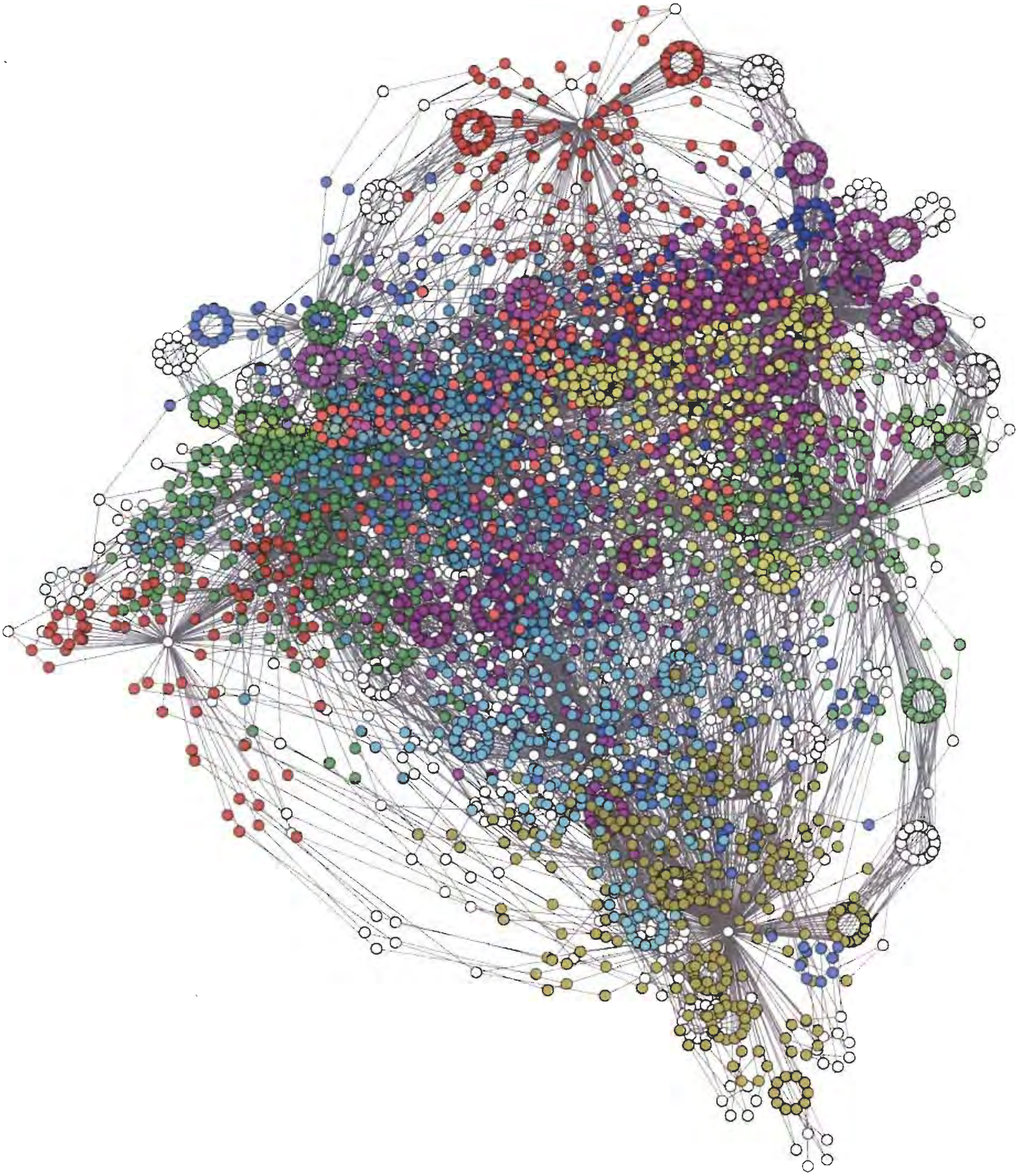


MultiNet: An interactive program for analysing and visualizing complex networks

by

Andrew J. Seary
B.Sc. McGill University, 1968

THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

In the
Faculty of Applied Science
under Special Arrangements

Simon Fraser University

Summer 2005

DECLARATION OF PARTIAL COPYRIGHT LICENCE

Approval

Name:	Andrew J. Seary
Degree:	Doctor of Philosophy
Title of Thesis:	MultiNet: An interactive program for analysing and visualizing complex networks

Examining Committee:

Chair: Dr. J. Driver
Department of Archaeology

Senior Supervisor: Dr. W. D. Richards
School of Communication

Member: Dr. E. D. Crozier
Department of Physics

Member: Dr. L. Goddyn
Department of Mathematics

Internal/External: Dr. G. Barnett
State University of New York, Buffalo NY

External: Dr. R. Breiger
University of Arizona, Tucson AZ

Date Approved --

Abstract

MultiNet is a Windows-based computer program designed for exploratory data analysis of social and other networks. MultiNet is highly interactive and always provides both textual and visual representations of results. The visualizations are innovative in the use of colour and interaction, and some are unique to MultiNet.

MultiNet was designed from the beginning to handle large amounts of data, and uses compact data formats, special storage schemes, and calculation methods that are highly efficient in terms of both space and time. MultiNet was also designed to handle large numbers of variables, both attribute (node) and network (link); it allows easy construction of new variables of either type by means of various operations on existing ones. Hybrid variables are easily constructed: node variables derived from networks; link variables derived from attributes. These capabilities provide crucial links among other parts of the program.

The application of spectral methods to large, sparse networks is both the theoretical and practical centre of the research and development that has gone into MultiNet. Spectral methods provide *analytic* visualizations of network data: pictures that not only provide understanding, but that provide numerical values that can be used in further analysis. The results of the spectral methods, as well as other attribute and network data, are used together with simple, standard statistical methods such as cross-tabulations, analysis of variance and correlations for testing hypotheses about relationships among the data. MultiNet provides unique methods that allow attributes and networks to be freely mixed in such analyses, and presents results in both textual and interactive visualizations that include two or three discrete or continuous variables.

The largest part of this thesis consists of descriptions of the seven main MultiNet program modules. Supplementary sections describe the theoretical background for spectral analysis and provide specific examples of spectral analysis, including a peer-reviewed, published paper that uses most of the parts of MultiNet together. In addition, a separate CD-ROM provides a working version of the program, electronic documentation, sample datasets, software aids and videos showing how the program is used.

Dedication

To my partner in all things: Percilla.

Acknowledgments

I would like to thank:

- The International Network for Social Network Analysis (INSNA), both as an organisation as well as selected members, for the acceptance of and support for this research through many presentations at Sunbelt conferences.

- The Special Arrangements program at Simon Fraser University for providing the opportunity to combine interdisciplinary research, applications and scholarship.

- Drs. Crozier and Goddyn, members of my committee, for simple listening, penetrating questioning, and open-minded encouragement.

- Dr. William Richards, my senior supervisor, for introducing me to the fascinating field of social networks, and for encouraging and guiding all aspects of this work.

- My family, for their belief that whatever I did was both good and worthwhile.

Technical Note

The printed version (or microfilm or photocopy) of this dissertation has the colour images rendered as grey-scale halftones, and much detail is lost. It is highly recommended that you get a copy of the free electronic version in PDF file format which shows all images in full colour.

This dissertation is available in electronic form as a PDF file from the library at Simon Fraser University. See SFU institutional repository:
http://ir.lib.sfu.ca/handle/1892/112

This dissertation is also available in electronic form as a PDF file at:
http://www.sfu.ca/~richards/Papers/DissAJS.pdf

List of Figures

Preface: Spectral methods for analyzing and visualizing networks

I. Motivation

Network analysis begins with data that describes the set of relationships among the members of a system. The goal of analysis is to obtain from the low-level relational data a higher-level description of the structure of the system which identifies various kinds of patterns in the set of relationships. These patterns will be based on the way individuals are related to other individuals in the network. Some approaches to network analysis look for clusters of individuals who are tightly connected to one another; some look for sets of individuals who have similar patterns of relations to the rest of the network. Other methods don't "look for" anything in particular — instead, they construct a continuous multidimensional representation of the network in which the coordinates of the individuals can be further analyzed to obtain a variety of kinds of information about them and their relation to the rest of the network.

One approach to this is to choose a set of axes in the multidimensional space occupied by the network and rotate them so that the first axis points in the direction of the greatest variability in the data; the second axis, orthogonal to the first, points in the direction of greatest remaining variability, and so on. This set of axes is a coordinate system that can be used to describe the relative positions of the set of points in the data. Most of the variability in the locations of points will be accounted for by the first few dimensions of this coordinate system. The coordinates of the points along each axis will be an *eigenvector*, and the length of the projection will be an *eigenvalue*. The set of all eigenvalues is the *spectrum* of the network.

Spectral methods (eigendecomposition) have been a part of graph theory for over a century. Network researchers have used spectral methods either implicitly or explicitly since the late 1960's, when computers became generally accessible in most universities. The eigenvalues of a network are intimately connected to important topological features such as maximum distance across the network (diameter), presence of cohesive clusters, long paths and bottlenecks, and how random the network is. The associated eigenvectors can be used as a natural coordinate system for graph visualization; they also provide methods for discovering clusters and other local features. When combined with other, easily obtained network statistics (e.g., node *degree*), they can be used to describe a variety of network properties, such as degree of robustness (i.e., tolerance to removal of selected nodes or links), and other structural properties, and the relationship of these properties to node or link attributes in large, complex, multivariate networks.

Researchers who use methods like Correspondence Analysis (Greenacre, 1984) or Principle Components Analysis (Jolliffe, 1986) are generally aware that they are using eigendecomposition (or spectral) methods. However, those who use computer programs such as CONCOR (Breiger, Boorman and Arabie, 1975) or NEGOPY (Richards and Rice, 1981) may not be aware of this because the eigenvalues or eigenvectors are hidden more or less deeply within the analytic package. Nevertheless, spectral methods lie at the heart of these analytic tools.

II. History of spectral analysis of networks

There has been a recent renewal of interest in spectral methods for partitioning large sparse arrays, based on results that first appeared in the literature over thirty years ago. Hagen (1992) speculates that this is because growth in problem complexity (more than 10,000 nodes) has exposed scaling weaknesses in iterative "local" methods such as Kernighan and Lin (1970), Kirkpatrick, et al., (1983). The methods currently being developed exploit "global" properties of the underlying graph representation of networks and so should be of interest to social network researchers. The author has examined the literature in Algebraic Graph Theory (Biggs, 1993; Cvetkovic, Doob, and Sachs, 1995; Cvetkovic, Doob, Guttman, and Torgasev, 1988), focussing on structural aspects of networks and attempting to discern the significance of the eigenvectors generated by standard eigendecomposition methods (Friedman, 1993; Powers, 1988; Dodziuk, 1984).

Early attempts to understand network structure from eigenvectors include Gould (1967) in geography and Bonacich (1972) in social networks. Both argue that components of the dominant (Frobenius) eigenvector of the adjacency matrix are a measure of "accessibility" or "centrality". However, at the end of their survey article, Cvetkovic & Rowlinson (1990) state that the Frobenius eigenvector of an adjacency matrix does nothing more than satisfy the Perron-Frobenius theorem (Senata, 1981), i.e. it cannot be used as a measure of "centrality" since it is a mixture of background and leakage from other dimensions.

More recent work has attempted to deal with leakage by combining eigenvectors or by devising better "filters" by loading the diagonal. The use of eigenvectors of the adjacency matrix ("*Standard*" eigenvectors) to provide clusters/partitions is examined in papers by Donath and Hoffman (1973) and Barnes (1982). Barnes uses a technique that finds partitions with the minimum number of connections between the parts. The technique is interesting since it uses both first and second positive eigenvectors to find a 2-way partition, the first three to find a 3-way partition, and the method can evidently be extended. Donath and Hoffman take a different approach: they add negative numbers to the diagonal of the adjacency matrix before performing the eigendecomposition. This is an example of the application of a transformation to the matrix that causes the results of eigendecomposition to be more useful in some way.

While methods that find partitions which minimize distance between connected nodes result in cohesive clusters of connected nodes, there are also methods that find partitions which maximize the distances between connected nodes. These partitions can be used to identify *structurally equivalent* (Lorrain & White, 1971) sets of nodes (roles which are "colourings"). Aspvall and Gilbert (1984) present a procedure for colouring graphs based on combinations of the most negative eigenvectors.

It is evident that eigendecomposition of the adjacency matrix, though it has many useful applications in Algebraic Graph Theory, requires that we either form combinations of its eigenvectors to apply it to problems such as clustering and colouring (and thus overcome "leakage" of useful information into the family of eigenvectors), or devise better "filters" by transforming the adjacency matrix before eigendecomposition.

The Laplacian matrix of a graph (Biggs, 1993; Merris, 1994) is formed by subtracting the original matrix from one in which the elements in the diagonal are the sums in the rows of the original matrix. Although this additive variation of the adjacency matrix may seem like a very simple one, a large literature has developed on the Laplacian in the last 20 years, mainly because this discrete analogue shares so many important properties with its continuous version, which is undoubtedly the most important and best understood operator in mathematical physics. Some useful discussions of its properties are found in Grone and Merris (1994); Friedman (1993); Grone, Merris, and Sunder (1990); Bien (1989); Alon (1986); Alon and Millman (1985); Anderson and Morley (1985); and Dodziuk (1984). Discussions of practical use in partitioning large sparse matrices are found in Barnard and Simon (1994); Hagen (1992); Pothen, Simon, and Liou (1990). Although Mohar (1991) presents a convincing case for considering the Laplacian as being a more fundamental matrix than the adjacency matrix for algebraic graph theory, the Laplacian has not been used in social network analysis, with the exception of Wilson (1982).

Because of the discussion in Fiedler (1975), the eigenvector belonging to the second-smallest eigenvalue of the Laplacian matrix has been called the Fiedler eigenvector (the smallest eigenvalue is a trivial 0, with constant eigenvector). The Fiedler eigenvector has been used to recursively partition a graph into subgraphs until the sub-graphs all satisfy some maximum size criterion. This approach is called Recursive Spectral Bisection (RSB) in Simon (1991); Barnard and Simon (1994); Walshaw and Berzins (1995); Hendrickson and Leland (1995). RSB is very similar in approach to CONCOR (McQuitty and Clark, 1968; Breiger, Boorman, and Arabie, 1975), a procedure commonly used to find structurally equivalent blocks in a network (Breiger, et al., 1975). The analysis by Schwartz (1977) shows the partitions of a graph produced by CONCOR to be closely related to "Standard" eigenvectors with positive or negative eigenvalues.

Correspondence analysis (CA) has become popular in social network research in the last 20 years (Greenacre, 1984; Noma and Smith, 1985; Hoffman and Franke, 1986; Barnett, 1993). It has been used mostly for clique detection, and it can be shown (Richards and Seary, 1997) that it produces the same cliques as NEGOPY (Richards and Rice, 1981). CA has not been used for general block-modelling. I have shown that CA is very closely related to a graph theoretic spectrum I call the "Normal" spectrum (its eigenvalues range from -1.0 to 1.0), which in turn shares some important properties with the Laplacian (Seary and Richards, 1995, 1996). This has allowed a generalisation of CA to produce blockmodels which contain both on- and off-diagonal blocks. The resulting partitions are generally similar to those of CONCOR, but with a firm graph-theoretic basis. The "Normal" spectrum uses a multiplicative transformation of the adjacency matrix and is the main subject of my research; it is described in Sections 5 and 8-10. CA has an interesting history, in that it has been re-invented so many times with so many names in so many different fields ("reciprocal averaging": Horst (1935) in ecology; "canonical analysis": Fisher (1940) in biometrics; "dual scaling": Gutman (1941) in psychometrics; "analyse des correspondences": Benzecri (1963) in linguistics; "Q-spectrum": Runge (Cvetkovic, et.al.,1976) in graph theory; "combinatorial Laplacian": Dodziuk and Kendall (1985) in physics; "multiplicative reversiblization": Fill (1991) in probabilistics; "Normal graph spectrum": Seary and Richards (1995) in social networks).

Another branch of research, complementary to the partitioning efforts, is based on the early work of Cheeger (1970), who proved a lower bound on Laplacian eigenvalues on a Riemannian manifold in terms of an isoperimetric constant of that manifold. Buser (1982) proved an upper bound, based on the same constant. This constant was generalised by Dodziuk (1984) and Alon (1986) for discrete graphs, and these results led to the first explicit construction of expander graphs. These graphs are of great importance in Computing Science, since they have the short distances of random graphs and have long been known to exist only by existence proofs. What makes them complementary to the partitioning research is that they have large gaps in the Laplacian spectrum, whereas the partitions are feasible because of small gaps in that spectrum. This result was soon generalised further to Markov chains by Lawler and Sokal (1988) and Diaconis and Stroock (1991) who noted that the probability transition matrix equation could be expressed as a Laplacian (indeed, this matrix differs from that of the Normal matrix of a graph by an identity matrix). Markov chains with large spectral gaps produce rapid mixing in which almost uniform distributions are reached very quickly, and this property is very important in the theory of Randomized Algorithms (Motwani and Prabhakar, 1995). Fill (1991) extended this result to Markov chains for which the underlying graph is directed, and in so doing re-invented Correspondence Analysis once again. Research in this area has been concentrated on large spectral gaps, and avoids the use of negative eigenvalues. My work has concentrated on results for small gaps (and local structure), and allows the negative part of the spectrum.

III. History of MultiNet

MultiNet began as an attempt to combine two methods of examining social network data devised by Dr. W.D. Richards: FATCAT (Richards, 1986) and NEGOPY (Richards & Rice, 1981). The former used simple counting methods (such as cross-tabulations) to look for significant associations among node attribute data and network link variables, but there was no way to extract node attributes based on network structure. The latter found network structure such as cohesive groups of nodes, but did not allow the use of node attributes. The author began programming and extending graphical presentations of FATCAT analyses (*panigrams*) based on ideas from (Richards, 1988). The author also examined the algorithms used in NEGOPY and determined that an important early part of the computer program was calculating a mixture of eigenvectors of a Markov chain transition matrix (Senata, 1981) derived from the network. At about the same time, (Barnett & Richards, 1991) observed that this part of the program was producing vectors which had very high correlations (>0.99) with the eigenvectors of Correspondence Analysis (Barnett, 1993) applied to the network. Recognition of the close relationship between these two apparently different types of analysis did not appear in the literature of graph spectra, and the author began research on developing a coherent theory of the graph spectrum which united these two ideas, and which he called the Normal spectrum (because of the many normalizations that occur in both the matrix that is analysed and the resulting spectrum).

One of the first results of this research was the recognition of the importance and usefulness of negative eigenvalues (and associated eigenvectors) (Seary & Richards, 1995). Negative eigenvalues are ordinarily suppressed in Markov analyses (Richards & Seary, 1997) and eigenvalue signs are "lost" in Correspondence Analysis (so that results can sometimes be misleading if CA is not done correctly). The importance of negative eigenpairs in capturing "off-diagonal" sets of nodes with similar sets of connections had already been exploited by CONCOR (Breiger, et.al., 1975) as a basis for *block-modelling,* but CONCOR remained poorly understood, despite the analysis by (Schwartz, 1972). It appeared that Normal analysis could yield similar results while based on a simpler theory.

Adding block-models derived from Normal analysis to the statistical analyses available in FATCAT was accepted as a useful strategy, and this was accomplished in the first DOS-based version of MultiNet (Seary, 1995), so-called because it allowed multiple networks (as in NEGOPY) along with multiple attributes (as in FATCAT). This version, as was common to other social network programs at the time, was limited in the size of networks it could handle.

IV. Research goals

When the research program for this thesis was proposed, the author outlined three main goals:

1. To continue development of a coherent, unified theoretical formulation of spectral approaches for social network analysis
2. To continue development of applications of the theoretical material for social network researchers, and to extend existing approaches so they provide more kinds of information about network structure and make it easier to relate this information to data that describes the people and the social context in which the networks function;
3. To implement the theoretical material in a computerized analytic package that enables social network researchers to apply the ideas and concepts in their efforts to describe and understand social networks. This package will be interactive and graphical, and will use sparse matrix methods throughout, allowing very large networks to be analysed quickly.

These goals have been realized as follows:

1. The theory of the Normal graph spectrum is described in sections 5.7, 9, 10 and 11.
2. Extensions of simple statistical methods to network analysis are described in sections 2 and 4.

   These include support for 3-mode analysis and discrete-continuous and continuous-continuous methods such as Analysis of Variance and Correlations. In addition, support for confirmatory models has been added in the Pstar module.
3. The current Windows-based version of MultiNet allows rapid analysis of large social networks by use of sparse matrix methods. Some limits on the modules are currently 5,000 nodes for Eigenspace and Pstar, although the code has been tested successfully for up to 25,000 nodes. The other modules are limited only by available memory. The Windows-based graphics are fast and interactive, with built-in support for both bitmap and PostScript outputs. In addition, other Windows-based programs may be used to capture interactive analyses.

Finally, the following documentation contains many examples that show how the goals can be achieved by social (and other) network researchers. Indeed, as the final sections show, the methods available in MultiNet have application outside of what is usually considered social network research.

V. References

Alon, N. and Millman, V. (1985). λ_1, Isoperimetric Inequalities for Graphs, and Superconcentrators. *J. Comb. Theory B.* **38**: 73-88.

Alon, N. (1986). Eigenvalues and Expanders. *Combinatorica* **6** (2),73-88.

Anderson, M. and Morley, T. (1985). Eigenvalues of the Laplacian of a Graph. *Linear and Multilinear Algebra* 18: 141-145.

Aspvall, B. and Gilbert, J. (1984). Graph Coloring using Eigenvalue Decomposition. *SIAM J. Alg. Disc. Math.* **5** (4), 526-538.

Barnard S. and Simon, H. (1994). Fast Implementation of Recursive Spectral Bisection for Partitioning Unstructured Problems. *Concurrency: Practice and Experience* **6** (2), 101-117.

Barnes, E. (1982). An Algorithm for Partitioning the Nodes of a Graph, *SIAM J. Alg. Disc. Math.* **3** (4), 541-550.

Barnett, G.A. and Richards, W.D. (1991, February). A comparison of NEGOPY's clique detection algorithm with correspondence analysis. Paper presented to the Sunbelt Social Networks Conference, Tampa. FL.

Barnett, G., (1993). Correspondence Analysis: A Method for the Description of Communication Networks. In W. Richards, and G. Barnett (Eds.). *Progress in Communication Sciences XII.* Newark NJ: Ablex . 135-164.

Benzecri, L. (1969). Statistical Analysis as a Tool to Make Patterns Emerge from Data. In S. Watanabe (ed.). *Methodologies of Pattern Recognition,* Academic Press. 35-74.

Bien, F. (1989). Constructions of Telephone Networks by Group Representations. *Notices of the Am. Math. Soc.* 36: 5-22.

Biggs, N. (1993). *Algebraic Graph Theory.* New York: Cambridge University Press.

Bonacich P (1972). Factoring and Weighting Approaches to status scores and clique identification. *Journal of Mathematical Sociology* 2: 113-120.

Breiger, R., Boorman, S. and Arabie, P. (1975). An Algorithm for Clustering Relational Data with Applications to Social Network Analysis and Comparison with Multidimensional Scaling. *J. Math. Psych.* **12** (3), 328-382.

Buser, P. (1982). A Note on the Isoperimetric Constant, *Ann. Sci. Ecole Norm. Sup.* 15: 213-230

Cheeger, J. (1970). A lower Bound for the Lowest Eigenvalue of the Laplacian. In R. C. Gunning (ed.). *Problems in Analysis,* Princeton Univ. Press, 195-199

Cvetkovic , D., Doob, M., and Sachs, H. (1995). *Spectra of Graphs,*.Academic Press.

Cvetkovic , D., Doob, M., Gutman, I., Torgasev, A. (1988). *Recent Developments in the Theory of*

Graph Spectra. North-Holland.

Cvetkovic , D. and Rowlinson, P. (1990). The largest eigenvalue of a graph: a survey. *Linear and Multilinear Algebra.* **28** 3-33.

Diaconis, P. and Stroock, D. (1991). Geometric Bounds for Eigenvalues of Markov Chains, *Ann. Appl. Prob.* 1: 36-61

Dodziuk, J. (1984). Difference Equations, Isoperimetric Inequality and the Transience of Certain Random Walks. *American Mathematical Society.* **284** (2), 787-794.

Dodziuk, J. and Kendall, W. S. (1985). Combinatorial Laplacians and Isoperimetric Inequality. In K. D. Ellworthy (ed.). *From Local Times to Global Geometry*, Pitman Research Notes in Mathematics Series **150** 68-74

Donath, W. and Hoffman, A. (1973). Algorithms for Partitioning Graphs and Computer Logic Based on Eigenvectors of Connection Matrices, *IBM Technical Disclosure Bulletin.* **15** 938-944.

Fiedler, M. (1975). A Property of Eigenvectors of Non-negative Symmetric Matrices and its Application to Graph Theory. *Czech. Math. J.* 85: 619-633.

Fill, J. A. (1991). Eigenvalue Bounds on Convergence to Stationarity for Nonreversible Markov Chains, with an Application to the Exclusion Process, *Ann. Appl. Prob* 1: 62-87

Fisher, R. A. (1940). The Precision of Discriminant Functions, *Annals of Eugenics* 10: 422-429

Friedman, J. (1993). Some Geometrical Aspects of Graphs and their eigenfunctions. *Duke Mathematical Journal* **69** 487-525.

Gould, P. (1967). The Geographical Interpretation of Eigenvalues, *Institute of British Geographers Transactions*, **42** 53-85.

Greenacre, M., (1984). *Theory and Application of Correspondence Analysis.* Academic Press.

Grone, R., Merris, R. and Sunder, V. (1990). The Laplacian Spectrum of a Graph. *SIAM J. Matrix Anal. App.* **11** (2),218-238.

Grone, R. and Merris, R. (1994). The Laplacian Spectrum of a Graph II, *SIAM J. Discrete Math.* **7** (2), 221-229.

Guttman, L. (1941).The Quantification of a Class of Attributes. In The Committee of Social Adjustment (ed.). *The Prediction of Social Adjustment*, New York: Social Research Council, 319-348

Hagen, L. (1992). New Spectral Methods for Ratio Cut Partitioning and Clustering, *IEEE Trans. CAD*, **11** (9),1074-1085.

Hendrickson, B. and Leland, R. (1995). An improved spectral graph partitioning algorithm for mapping parallel computations. *SIAM J. Comput. Sci.* **16** (2), 452-469.

Hoffman, D., Franke, G. (1986). Correspondence Analysis: Graphical Representation ofCategorical

Data in Marketing Research. *J. Market Research* 23: 213-227.

Horst, P. (1935). Measuring Complex Attitudes, *Journal of Social Psychology* 6: 369-374

Jolliffe, I. T. (1986). *Principal Components Analysis.* Springer-Verlag.

Kernighan, B. and Lin, S. (1970). An Efficient Heuristic Procedure for Partitioning Graphs, *Bell System Tech. J.* **49** 291-307.

Kirkpatrick, S., Gelatt, C., and Vecchi, M. (1983). Optimization by Simulated Annealing, *Science.* **220** 671-680.

Lawler, G. F. and Sokal, A. D. (1988). Bounds on the L^2 Spectrum for Markov Chains and Markov Processes: a Generalization of Cheeger's Inequality, *Trans. Amer. Math. Soc.* 309: 557-580

Lorrain, F. & White, H.C. (1971) Structural equivalence of individuals in social networks, *J. Mathematical Sociology*, **1** 49-80

McQuitty, L.L., and Clark, J.A. (1968). Clusters from iterative intercolumnar correlational analysis, *Educational and Psychological Measurement.* 28: 211-238.

Merris, R. (1994). Laplacian Matrices of Graphs: A Survey, *Linear Alg. App.* **198,199**: 143-176

Mohar, B. (1991). The Laplacian Spectrum of Graphs. In Alavi, Chartrand, Ollermann and Schwenk (eds.). *Graph Theory Combinatorics and Applications*, Wiley, pp. 871-898

Motwani, R. and Prabhakar, R. (1996). *Randomized Algorithms*, Cambridge Univ. Press

Noma, E. and Smith, D.R. (1985). Scaling sociomatrices by optimising and explicit function: correspondence analysis of binary single response sociomatrices. *Multivariate research.* 20: 179-197.

Pothen, A., Simon, H, and Liou, K-P., (1990). Partitioning Sparse Matrices with Eigenvalues of Graphs. *SIAM J. Matrix Anal. App.* **11** (3), 430-452.

Powers, D. (1988). Graph partitioning by eigenvectors. *Linear Algebra Appl.* **101** 121-133.

Richards, W.D. (1986) FATCAT: a different kind of network analysis program, http://www.sfu.ca/~richards/Pdf-ZipFiles/fatman2.pdf

Richards, W.D. and Rice, R. (1981). The NEGOPY Network Analysis Program, *Social Networks*, **3** (3): 215

Richards, W.D. and Seary, A.J. (1997). Convergence Analysis of Communication Networks. *Advances in Communication Science.* Barnett, G. and Thayer, L. (eds). Norwood: Ablex.

Schwartz, J.E. (1977). An examination of CONCOR and related methods for blocking sociometric data, in Heise, D.R. (ed). *Sociological Methodology* 1977. 255-282. San Francisco: Jossey-Bass.

Seary, A.J. and Richards W.D. (1999). Eigenvalue Bounds, presented to International Network for Social Network Analysis, Charleston, S.C. February, 18-21 http://www.sfu.ca/~richards/Pages/insna99x.pdf.

Seary, A.J. and Richards W.D. (1996). Trivial Expectations: the 'trivial' eigenvector and removal of the expecteds, presented to International Network for Social Network Analysis, Charleston, S.C. February, 14-17.

Seary, A.J. and Richards, W.D. (1995). Partitioning Networks by Eigenvectors. Presented to European Network Conference, London. Published in Everett, M.G. and Rennolls, K. (eds). (1996). *Proceedings of the International Conference on Social Networks, Volume 1: Methodology*. 47-58.

Seary, A.J. (1995) MultiNet for DOS, Presented at International Conference on Social Networks (Sunbelt XV), London, UK

Senata, E. (1981). *Non-negative Matrices and Markov Chains*. Springer-Verlag.

Simon, H. (1991). Partitioning of Unstructured Problems for Parallel Processing. *Computing Systems in Eng.* **2** (2/3), 135-148.

Walshaw, C. and Berzins, M. (1995). Dynamic Load-balancing for PDE Solvers on Adaptive Unstructured Meshes, *Concurrency: Practice and Experience*. **7** (1),17-28.

Wilson, T.P.(1982). Relational networks: an extension of sociometric concepts, *Social Networks*. **4** (2), 105-116.

0. Overview of MultiNet

0.1 Introduction

MultiNet is a Windows-based computer program designed for interactive exploratory data analysis of social and other networks. It is divided into modules that allow for analysis and visualization of complex networks, and of details of the values of the link and node variables that make up the networks. The modules currently available are

1. File: Load or Save MultiNet data files, Import, Export or View ASCII data files
2. Analyse: Perform statistical analyses on two, three or four link and/or node variables.
3. Variables: Univariate statistics and transform, combine, create and delete link or node variables.
4. Groupings: Create or delete sets of link variables that are treated as a unit.
5. Eigenspaces: Visualize networks and create node variables and partitions from graph spectra
6. Models: Fit networks to exponential random graph model p* and create link variables from fit
7. Preferences: Select defaults for displays and reports and save or load them.

Each module except the first and last always produces a visual display as well as a textual report.

Sections 1-7 describe each of these modules in detail. The purpose of this section is to introduce some concepts and definitions that are used throughout the rest of this document. These include:

- introductory concepts such as graph and social network terminology
- data concepts such as sparse methods and missing data
- interaction concept such as window types and common buttons
- file concepts such as file types and automation
- graphics concepts such as colour and window size
- error-handling concepts such as error trapping and warnings

Throughout this document, the following conventions are used:

- words which are *italicized* the first time they are used are described in the Glossary
- MultiNet menu items, choices and labels are presented in **bold Arial**
- MultiNet textual reports are presented in `Courier`
- Words are emphasized with underline

0.2 Introductory definitions

Social network analysis (SNA) is largely concerned with two types of objects: 1) people (usually referred to as *actors)* and 2) the relationships between people (usually referred to as *ties*). Graph theory abstracts the two types of objects into 1) *nodes*, *vertices*, or *points* and 2) *links*, *edges*, or *lines*. The terms node and link appear to be the most commonly used. SNA often refers to ties sent from actors and ties received by actors. Some types of relationships are inherently *symmetric* or *undirected* since a link is always two-way (e.g., "married to"). The term undirected describes the type of relationship, while the term symmetric describes how the relationship can be stored and manipulated. Some types of relationships are inherently *directed* (e.g. "child of"). Some relationships can be directed and yet result in symmetric data (e.g., "get advice") if a directed link from A to B is *reciprocated* by a directed link from B to A. Links may have values other than the binary presence or absence of a relationship (e.g., "duration" of a contact). A reciprocated directed link need not be symmetric if the values are different in each direction. A link that is sent and received by the same node is called a *self-link* (necessarily symmetric). These are not common in SNA, but are sometimes meaningful (e.g., "voted for ").

Networks based on undirected relationships are called *undirected networks* and may be stored and manipulated in an efficient way since they are symmetric (so they are also called symmetric networks). If a network results from directed links, then it is a *directed* network, even though many (or even all) of the relationships are reciprocated. A network of completely reciprocated links (of equal value) may be treated as symmetric. However, in general directed networks are not symmetric.

A set of links connecting two nodes is called a *path*, and the shortest path between nodes is called a *geodesic* (and there may be more than one). In an undirected network, if there is a path from every node to every other node, the network is said to be *connected*. The length of the longest geodesic is called the *diameter*. If there are nodes with no paths between them, then the network is *disconnected* and consists of a number of smaller connected networks called components, and the diameter is not defined (or is considered infinite). Therefore a connected graph has exactly one component. In directed graphs, we can have two types of path: *weak* (which ignores direction) and *strong* (which does not). A network is said to be *weakly* (*strongly*) *connected* if there is a weak (strong) path between every pair of nodes. A weakly (strongly) disconnected directed network consists of more than one weak (strong) component. A network in which there is a link between every pair of nodes is called *complete*. A subset of a network which is complete (every node linked to every other node) is called a *clique*. If the relationship on which the network is based is undirected, the resulting network should be undirected and all links should be reciprocated. If some are not, there is measurement error. There is no such simple check for error in directed networks.

A central concern in SNA is relating network structure (such as cliques) to attributes of the actors (such as age, sex, education level, etc.). In MultiNet terminology, actor attributes are referred to as *Node attributes* and are represented by *Node variables.* Attributes of relationships between nodes are called *Link attributes* and are represented by *Link variables.* Some node attributes can be derived from link attributes. For example, in a symmetric network, the number of links that a node has is called the *degree.* For a directed network, the number of links sent from a node is called the *out-degree*, while the number of links received by the node is called the *in-degree.* An example that is used often in this document is KIDS2. This is data collected from a day care center (Richards, 1988). The .node variables describe the sex of the children (SEX) and their ages (AGE), We call the set of all node variables the *node attributes*, so there are 2 node attributes The link variables describe who children say they play with (SAY), and who they are observed to play with (PLAY), so there are 2 link variables. Link variable PLAY is inherently symmetric: a link is reported when two children are seen to play togther so all links are reciprocated. Link variable SAY is not inherently symmetric: a child may nominate a number of others (large out-degree), but be nominated by few or none of them (small in-degree). The complete set of all node and link variables (along with optional subsidiary data such as labels and comments) is called a *dataset.* There is no limit (apart from available memory) on the number of node and link variables that MultiNet can handle simultaneously in a dataset.

A type of network that is common in SNA but not discussed much in Graph Theory is *ego-centric* data. This is network data collected by asking a sample of people to nominate, for example, who their friends are (so the link variable is "friendship"), who they play a sport with ("play tennis with"), and so on. In general, there is no reason to expect that the nominees in these networks will be the same for any two people doing the nominating (so the resulting network will not be connected), that any of those nominated are also nominators (so there are no reciprocated links), or even that there will be overlap among the nominees of a given nominator for different relationships (so the number of people nominated may be much larger than the number of nominators). This leads to a very fragmented set of small directed "*star graphs*" (one central node with many spokes to other nodes) which is not very interesting from a Graph Theory point of view. Many standard SNA measures are also useless. Nevertheless, this type of data is handled by MultiNet since it was originally designed for this purpose. The example dataset used in this document is 301 (Richards, 1988), which is self-reported data collected by students in a course (CMNS 301). It describes every interaction the students had over a one week period. This dataset also contains examples of non-binary link variables. For example, "Duration" contains real values which describe the length of time taken by some form of communication; "When" contains integer values describing the hour of the day in which the communication took place. The non-binary values of a link variable are generally referred to as the *strength*, since these are often a measure of the amount or intensity of an interaction, and can be used to weight sums of interactions. This weighting would be valid for "Duration", but not for "When", which does not measure amount or intensity.

Another type of network that is common in SNA is the *2-mode* network. This is data that links two different type of nodes, for example people and the events they attend. A link is defined when a person attends an event. There cannot be links among people or among events so that the network divides into two parts: it is a *bipartite* graph (in Graph Theory they are also called *hypergraphs* in which there are *hyper-links* between the two types of nodes) . The example dataset used in this document is SYM-EXP, which describes people, their medical symptoms, and the exposures that caused the symptoms (so this is actually a 3-mode dataset).

The last example is a network that, by definition, cannot have links between every possible pair of nodes. For ego-centric data, this is not impossible but very unlikely. In fact, for social networks in general it is unlikely that every pair of nodes is linked in some way, and this becomes even more unlikely as the number of nodes increases.

0.3 Data concepts

0.3.1 Sparse methods

A social network consisting of N nodes can have a maximum of N^2 links. If we remove self-links (usually these are not meaningful), this reduces to N(N-1). If we further stipulate that the relationship is symmetric, this drops to N(N-1)/2. We can measure the *density* of a network by taking the ratio of actual links to the maximum possible (N^2, N(N-1) or N(N-1)/2, depending on the type of relationship). A complete graph has density of 1 and a set of nodes with no links has density of 0. Social networks generally have densities much less than 0.1, so that the amount of storage required for a link variable is significantly less than N^2. Though there is no standard for this definition, we will say that any network with density less than 0.1 is *sparse*. The essence of *sparse methods* is to store only actual data, and assume that everything not stored is 0. (Other default values are possible, but this is by far the commonest and most tractable case). Further, no action can be taken on this data that would require more storage. For example, it is possible to calculate paths between nodes by matrix multiplication, but this eventually requires more and more storage for the matrix powers. In fact, all of the N^2 storage will eventually be required to calculate the diameter of the network by this method. For undirected networks with no self-links the storage is smaller, but still contains a term in N^2. This is not the only problem: the storage will be N^2, but the number of calculations required will be N^3, so that a network which is twice as large will take 8 times longer for this calculation. However, it is quite possible to find paths between nodes and even the diameter by methods that require much less space and time, using sparse methods such as link list representation and breadth-first search (Aho, et al., 1983)

MultiNet was designed from the beginning to handle large, sparse networks. There is currently no limit (apart from memory) on the number of nodes and links that can be handled by the Analyse and Variables modules. There is similarly no limit on the number of node and link variables that may belong to a MultiNet dataset, and it is very easy to create new node and link variables when desired. The Eigenspaces and Models modules currently restrict network size to no more than 5,000 nodes for the practical reason that few social network datasets are this large.

MultiNet for Windows is designed to handle large datasets with multiple attributes and multiple connections (hence the name). The best way to describe the internal sparse representation used by MultiNet is to define the external format used to initially bring data into the program. This is described in some detail in Section 1: The Files Module. Here follows a brief overview of data types and internal storage requirements.

0.3.2 Data representations and storage

A) Node variables

Typically, a node variable consists of a single value for each node. The nodes themselves are identified by numeric codes, called *ID numbers*, stored as 4 byte integers (allowing up to 2^{31} = 2,147,483,648 distinct ID numbers). Node attribute variables may be binary, integer or real (floating point). Binary values are stored as single bits. Integers may be stored as single bytes, double bytes, or as 4-byte values depending on the actual values. There is also one special node variable called IDLABEL which, if defined, consists of character data, and is used to label nodes where applicable (e.g., in Variable, Eigenspace and Pstar displays and reports).

B) Link variables

MultiNet does not use *adjacency matrices*, either as data files or internally. All calculations use sparse methods. Link variables are stored in a manner very similar to node variables, except each link is identified by a pair of ID numbers. There is no need to store all possible pairs of ID numbers, only those for which there is data for at least one link variable. As for Node variables, binary (presence/absence of a relationship) data is stored as single bits. This is very common for SNA link data. Combined with the sparse method of storing only ID pairs which have link values, this results in very efficient storage of network data. MultiNet does not use special storage methods for symmetric networks. A pair of ID numbers and a data value is required for each direction. However, the Eigenspace module can perform automatic symmetrization. In addition, the Variables module can generate all missing ID pairs to make a fully symmetric network from one which has links for only one direction. For non-binary data, the link values are also stored in 1, 2 or 4-byte integer formats, or 8-byte IEEE floating format as required.

C) Derived data

MultiNet allows new node variables to be derived from existing node variables (Variables module), or from existing link variables (Variables and Eigenspaces modules); new link variables can also be defined from combinations of node and link variables (Variables and Pstar modules). In each case the program automatically determines the most efficient storage format for the data based on its values.

D) Missing data

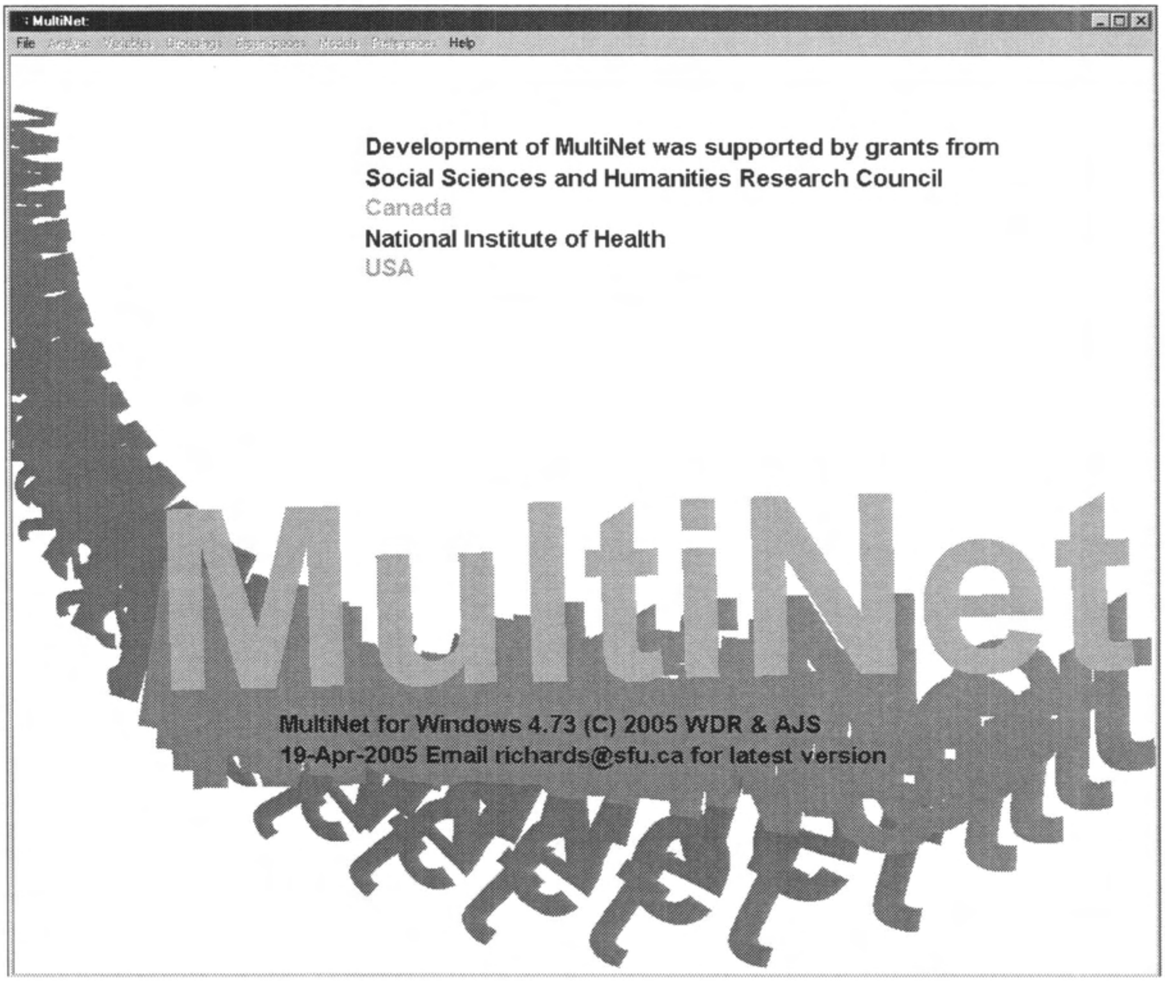
MultiNet keeps track of missing data for both node and link variables. Derived node or link variables can have non-missing values only for the intersection of the non-missing values of the variables used to construct them. Node variables derived from link variables must have missing values for any link ID numbers that do not appear as node ID numbers. Similarly, some node ID numbers may not appear in any link ID pairs. Whenever MultiNet reads in a dataset, either by IMPORTing from ASCII files, or LOADing system files, a report on such ID numbers is available from the "View Summary" menu item. This report lists all Node ID numbers that do not occur in any Link ID pairs, and all Link ID numbers that do not occur as a Node ID. Such problems should be considered as possible data entry errors.

0.3 Interaction concepts

MultNet was designed for interactive exploratory data analysis. The program is mostly menu-driven using the mouse much more than the keyboard. Most of the interaction takes place using standard Windows methods, and should be very familiar to Windows users. In particular, the program consists of a *Main Menu* (figures 0.1 and 0.4), with a descriptive *title bar*, and a menu bar consisting of eight choices. Any item in a menu bar may be selected by left mouse button single-click (left-click, usually referred to as simply "click") with the mouse pointer over the item. Menu items may also be activated (less conveniently) by accelerator keys on the keyboard: pressing the Alt- key, then the underlined key (e.g. F in **File**). The right mouse button has special uses in the **Variables** and **Eigenspaces** modules, but otherwise has no effect on menu items.

Selecting any of the Main Menu choices (except **File** and **Help**) starts a MultiNet module which replaces the main menu with a sub-menu with new choices. Menu items are available only if they are *enabled*; otherwise they are *disabled* and *greyed-out*, so that selecting them has no effect. Figure 0.1a shows the opening screen with initial main menu bar after the program has just started. Only the items **File** and **Help** are enabled, since no dataset has been read into memory yet.

The **Help** button is usually available on <u>every</u> MultiNet menu and most windows. The information provided by clicking on the **Help** item is context-sensitive. **Help** on a menu bar produces information describing what the current module does. In this case, since we are in the main menu, it provides an overview of all the modules in MultiNet, with detailed information about every choice available from the menu bars. To get at this information, **Help** opens up a *selection window* (figure 0.2).



Development of MultiNet was supported by grants from
Social Sciences and Humanities Research Council
Canada
National Institute of Health
USA

MultiNet for Windows 4.73 (C) 2005 WDR & AJS
19-Apr-2005 Email richards@sfu.ca for latest version

Figure 0.1. Initial MultiNet window with only **File** and **Help** enabled.

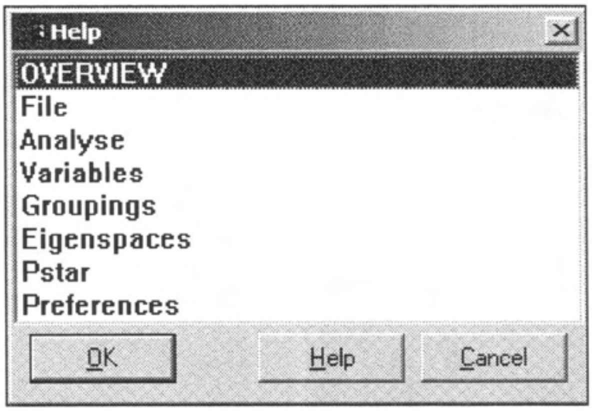


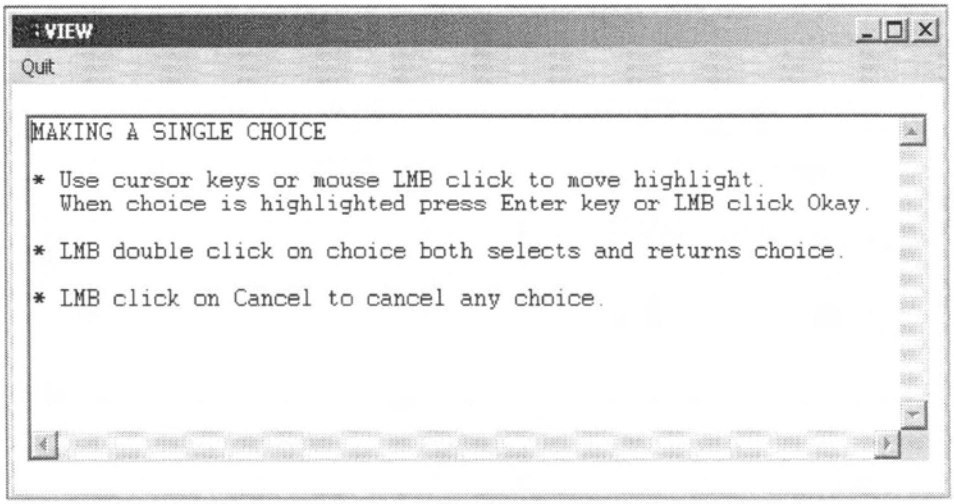
Figure 0.2. A selection window.
Title shows it is a **Help** window.
Vertical list of items describe the **Help** topics.
First item in list is default.

A selection window is a list of items presented vertically for selection by:

- double-click on an item
- single-click, then click on OK
- single-click, then press Alt-O (the accelerator key in OK)

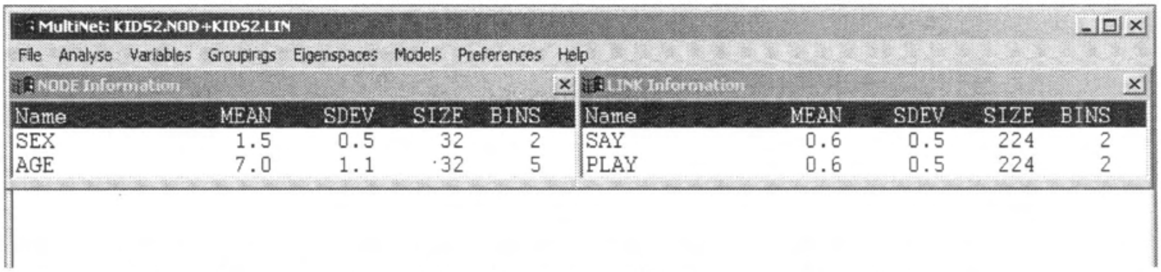
Clicking on **Cancel** (or pressing Alt-C) closes the window with no selection made. Selection windows appear in many contexts in MultiNet where a single item in a list of choices is to be selected.

When **Help** appears on a button (as in figure 0.2) it is used to describe what the program expects at this point. For example, clicking the **Help** button (or pressing Alt-H) for this selection window opens the *view window* shown in figure 0.3. A view window is a simplified version of an *edit window*. Both are used to display textual results. An edit window also allows text to be changed and saved. Text is highlighted by holding down the left mouse button while sweeping the mouse cursor over the text. This text can be copied to the clipboard by Ctrl-C.(both are standard Windows methods). View windows are used throughout MultiNet to display textual data such as **Help** screens and Reports and have **VIEW** in the title bar and **Quit** as the only menu item. Edit windows have **EDIT** in the title bar, and **Save** as an additional menu item, allowing text replacement.



```
MAKING A SINGLE CHOICE

* Use cursor keys or mouse LMB click to move highlight.
  When choice is highlighted press Enter key or LMB click Okay.

* LMB double click on choice both selects and returns choice.

* LMB click on Cancel to cancel any choice.
```

Figure 0.3. A view window contains text. Clicking on **Quit** or Alt-Q exits.
This help text is the default for selection windows.



Name	MEAN	SDEV	SIZE	BINS
SEX	1.5	0.5	32	2
AGE	7.0	1.1	32	5

Name	MEAN	SDEV	SIZE	BINS
SAY	0.6	0.5	224	2
PLAY	0.6	0.5	224	2

Figure 0.4. Main menu screen with a dataset in memory. All menu items are enabled.
Two display windows summarize the dataset

Figure 0.4 shows the main menu display after the dataset KIDS2.NOD and KIDS2.LIN has been loaded using the **File** menu item (see Section 1: The File Menu). The display now includes:

- The name of the file(s) in the coloured title bar at the top
- All menu items are now enabled and show as black rather than grey
- Two *display windows* show lists of the node and link variables along with some simple descriptive statistics.

A display window is a list of items for viewing only. The two display windows are intended as a helpful summary of the node and link variables currently in memory, and only appear when there is no other graphic display. They can be moved around individually by clicking on the title bar and dragging the mouse. If the number of variables in a window is more than 30 a scroll bar appears, allowing all items in the list to be viewed by left mouse button dragging on the scroll tab or using the arrow cursor keys. It the mouse has a wheel, this also can be used to scroll vertically through the list. Any window with a vertical scroll bar created by MultiNet scrolls in the same way. Horizontal scroll bars may appear for wide windows in which case left mouse button drag on the horizontal tab or arrow cursor keys must be used, since the mouse wheel only affects vertical scrolling. The display windows cannot be resized, but they can be closed. Closing either one closes both. These two display windows always reappear when returning to the main menu, or when variables have been deleted or reordered by the **Manage** menu item in the **Variables** module. The simple descriptive statistics include MEAN and SDEV (Standard deviation), which may not be strictly meaningful for categorical data. They also include SIZE (total number of data values) and BINS (total number of unique data values). Missing data means that not all variables need to be the same size. The number of Bins is used to determine whether a variable can be treated as categorical. For link variables (where not all possible node pairs need appear) MEAN, SDEV, SIZE and BINS refer only to the sparse link data that is actually defined for the link variables.

Another common window type in MultiNet is the *multiple selection window*, which is similar to a selection window, but allows for more than one selection to be made. Figure 0.5 shows a multiple selection window as it first appears. The example is the **Delete** choice of the **Manage** menu item in the **Variables** module. A common notation for this is **Manage→Delete**. All items are initially selected. If there is a vertical scroll bar, the list is automatically scrolled down to the last item. Selections are made using standard Windows methods:

- select a range by selecting the top (or bottom) item in a range with mouse click, then selecting the bottom (or top) item in the range with Shift-click.
- toggle selections on or off by Ctrl-click

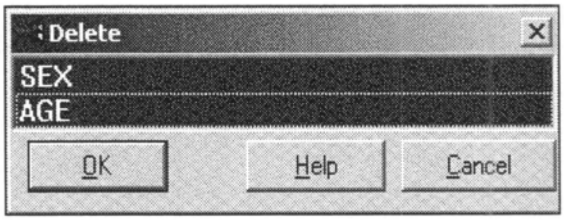
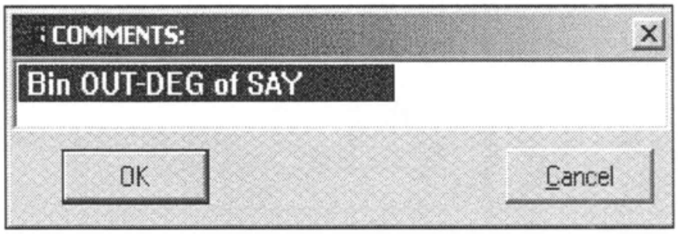


Figure 0.5. Multiple selection window. Initially all items are selected. The example is from **Manage→Delete** in the **Variables** module.

As for selection windows, selection(s) are confirmed by clicking on the the **OK** button (or Alt-O). Clicking on **Cancel** (or Alt-C) closes the window with no selection made.



Bin OUT-DEG of SAY

Figure 0.6. Text window for entering comments during variable creation. The default comment is provided by the program.

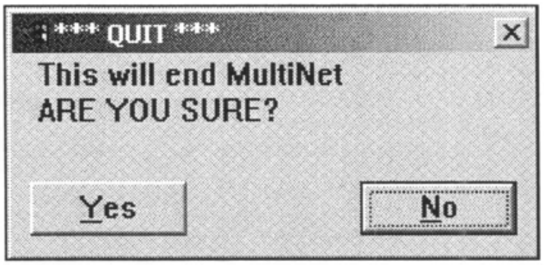
When keyboard input is required, for no more than one line of characters, a special edit window called a *text window* is used (figure 0.6). A text window often has some default characters which can be accepted with either clicking on the the **OK** button (or Alt-O), or by immediately pressing the Enter key. Text windows are used for defining variable names and comments, for entering equations, and for entering single numbers when required. When entering names and comments, the default characters are initially all highlighted, so that any editing immediately erases them (this is a standard Windows method). The actions that follow a text window (e.g., creation of a new variable) may be cancelled by clicking on **Cancel** (or Alt-C).

When it is neccessary to get confirmation before some important action is completed, a *YesNo window* is used. This window has a title which describes the action about to be taken, up to three lines of descriptive text, and two buttons labelled "Yes" and "No". The default action depends on the context. For example, there are three methods for exiting MultiNet:

- Alt-F4 (a Windows standard method)
- Click on the X button in the upper right of the main window (also a Windows standard)

- Select **File→Exit** from the main menu

Any one of these opens a YesNo confirmation window (Figure 0.7) before closing the program. Choosing No returns MultiNet to the main menu.



This will end MultiNet
ARE YOU SURE?

Figure 0.7. YesNo window to confirm exit from MultiNet. Default (dotted and emphasized) button is "No".

There are also some special purpose window types which are used only within one module. Examples are the highly interactive "Explore" windows in the **Variables, Eigenspaces,** and **Pspar** modules, and the interactive help windows in the **Analyse** module. These will be described in detail in the Sections where they are used.

0.4 File concepts

MultiNet deals with six types of files:

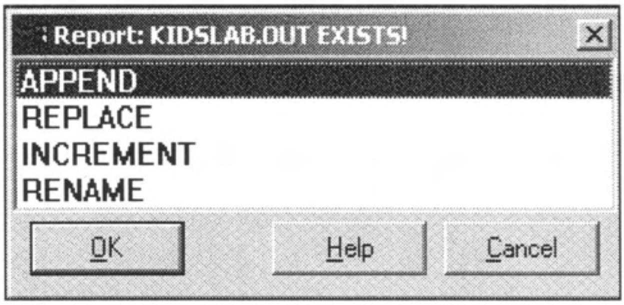
- NOD and .LIN files for importing and exporting data in MultiNet format (text files)
- .CSV files for importing and exporting data in comma-separated format (text files)
- .MNW system files (complete dataset in internal format)
- .OUT files produced by **Report→File** (text files used to report results)
- .PS files (PostScript text files that reproduce graphic displays)
- .BMP files (Graphic displays in internal compressed bitmap format)

The first three are used to get data into and out of MultiNet, and will be described in more detail in Section 1: The Files Module. The last three may be created in any MultiNet module that does any kind of analysis or fitting with a graphic display (all but **Files, Groupings** and **Preferences**), and always work the same way. Files are opened only while required for input or output, and are then closed . This means that any Reports or Graphics that have been written to files can be used immediately by other programs.

In every module which has a **Report** choice on the menu bar, clicking on **Report** produces the following choices:

- **Report→View** opens a scrollable, resizable view window which contains the text Report.
- **Report→File** saves the Report as an ASCII text file, with default extension .OUT without displaying it.

When Filing a Report, the first part of the current output file name is derived from the data file(s)currently in use, in this example KIDS2. If the file already exists, MultiNet will do one of two things, depending on how preferences have been set. It may use a standard method based on a setting made in the **Preferences** module (**File→Always Append**), which directs MultiNet to append the current Report to the current output file automatically. Alternatively, a selection window will appear (Figure 0.8), allowing the Report to a) be appended, b) replace the current output file, c) increment the current output file name (KIDS2.OUT becomes KIDS2__2.OUT), or d) rename the current output file (which opens a text window for the new file name). Choices c) and d) cause the new file to become the current output file. Once a Report has been filed, it is immediately available to other Windows programs, such as a printer or text editor.



Report: KIDSLAB.OUT EXISTS!

Figure 0.8. Selection window for **Report→File** User may select "**Always Append**" in the **Preferences** module, in which case this window does not appear and the first selection (Append) is automatic.

0.6 Graphics concepts

In every case where it is possible, MultiNet produces a graphic display which visualizes the latest interaction. The types of displays are mostly unique to each module (Pspar and Eigenspaces 1-D are similar), and some are quite innovative. The amount of interaction available in the Eigenspaces display is unusual for social network programs, as is the analglyphic (3-D) display. Panigrams (for XTABS) are both more complete and interactive than any other method for visualizing cross-tab results. The display of fit results in the Pstar module is also unique. Each of these displays will be described in more detail in their Sections, but all share common methods of displaying categorical data through colour. Table 0.1 in the technical appendix shows the colours used for up to twelve categories (the initial MultiNet default) with RGB values. Colours are always chosen in this order.

If there are more than 12 categories, the colours begin again at dark blue. This order was chosen to give the most contrast between succeeding colours, and the RGB values were chosen to give colours which are easy to distinguish from each other, and which appear very similar in both screen (additive) and print (subtractive) colour images. This order and these colours are used throughout MultiNet when displaying categories. For example, a 2-way partition in Eigenspaces displays as dark blue and dark red node labels. The same order and colours are used for the first two colours in any panigram display, for colouring the discrete variables in an ANOVA display, or for colouring the planes in a correlation display. The same order and colouring is used to show the diagonal or labels in a Pstar fit display using a node variable for blocking. To ensure that this mapping of colour to categories holds throughout all modules, empty categories should not be deleted from displays created in the **Analyse** module (this is the default and may be changed in the **Preferences** module). The alternative means the Analyse module will choose colours from this list for only non-empty categories when displaying the results of analyses. This is discussed further in Section 2: The Analyse Module.

Six other colours are used in graphics displays for other graphic purposes. These are shown in Table 0.2 in the technical appendix. These 18 colours are all of those used in graphics displays. Only the Arial (Helvetica) font is used for the text in graphic displays. Windows allows for smoothing the edges of screen fonts (anti-aliasing), which can produce many more colours. For this reason MultiNet saves bitmaps in the 256 colour format, using the 256 most common colours in any display. This allows run-length encoding compression with little loss of detail.

The MultiNet window is optimized for 1024horizontal by 768 vertical pixels, but automatically rescales to take up the upper left 3/4 of the screen at any resolution. In addition, the window may be *resized* by the standard Windows methods:

- Click on the square (full screen window) icon in the upper right corner of the title bar, or double-click on the MultiNet title bar results in the MultiNet window taking over the entire screen. This "full screen" window has the greatest resolution possible on the physical display. The square icon becomes overlapping double squares which will return MultiNet to windowed size with mouse-click.
- Click on the lower right corner of the MultiNet window and drag with left mouse button held down allows the window to be resized to any rectangular shape desired. As the window is resized, both the graphic image and any graphic text are resized to fit the new window. In the case of graphic text, this results in changes in font size. The aspect ratio (width to height) is not restricted, so the image may become quite distorted with text no longer lined up properly. It is generally easy to keep the image with the proper ratio, and this makes it easy to produce much smaller (but lower resolution) bitmap files for inclusion in documents. The PostScript images are not affected, and always fit in 8.5" by 6.5" (the bottom half of a page).

To return to the initial 3/4 screen windowed size, choose one of the methods to exit MultiNet, but click on "No" in the "Are you sure?" YesNo window. This also returns MultiNet to the main menu display with the default 3/4 screen resolution.

View and Edit windows may also be resized (to full screen or different size and shape) by standard Windows methods. All other MultiNet windows lack the square windowing icon on the right of the title bar, and cannot be resized.

All MultiNet modules that produce graphic displays have a **Graphics** item on the menu bar, and all behave similarly. clicking produces a selection window with the following choices:

- **Graphics→PostScript** reproduces the current display as a series of PostScript commands. The result of this selection is an ASCII text file and this output is treated exactly like the Report, with the default output having extension .PS (e.g., KIDS2.PS). The same method as for Reports is used to deal with .PS files that already exist, including automatic appending. This ASCII text file should produce graphics on any PostScript printer, or in any program that can interpret the PostScript language (The shareware program GhostScript is used to test all the PostScript output from MultiNet, and so is recommended). Technical appendix 0 contains more information on the PostScript translation

- **Graphics→Bitmap** captures the current screen display as a 256-colour bitmap, which is then run-length encoded. The result is saved as a Windows .BMP file, with the first part of the name coming from the data file(s) (e.g., KIDS2.BMP). These files can be imported into any Windows program that can read .BMP files. Since .BMP files can only contain one image Append cannot be used. If the file already exists, the choices become Replace, Increment or Rename. Run-length encoding can dramatically reduce the size of a bitmap, even at high screen resolution. Even so, note that complex images made at high resolution may produce very large files.

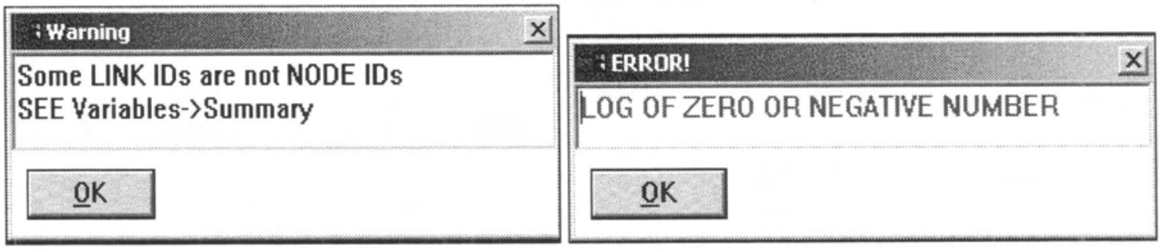
0.7 Error handling concepts

MultiNet traps all errors and reacts to three types of error by opening an *error window* (figures 0.9 and 0.10). The three types are:

- (Anticipated) Warning: An informational message. The current procedure can be continued.

- (Anticipated) Error: An error of a type the program checks for has occurred. The current procedure cannot be continued until the error condition is removed.

- Internal (Unexpected) Error: An error of a type that the program does not check for has occurred (figure 0.10).



Some LINK IDs are not NODE IDs
SEE Variables->Summary

LOG OF ZERO OR NEGATIVE NUMBER

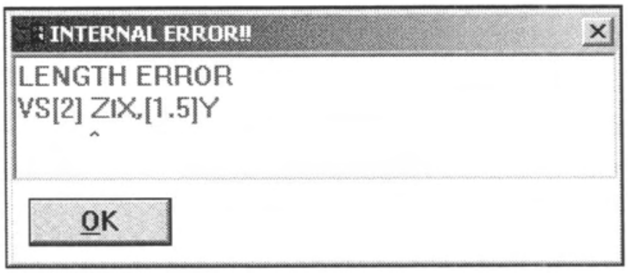
a) (Anticipated) Warning

b) (Anticipated) Error!

Figure 0.9. Anticipated error handling that is programmed into MultiNet

Anticipated Warnings and Errors are programmed into MultiNet. That is, the conditions that produce these errors are checked for, and if the condition exists a warning or error window is opened describing the condition. MultiNet will not continue until the error window is closed by one of the usual methods. The Warning example is discussed further in Section 1: The File Module, which describes the extensive error checking based on experience with common errors related to importing data. In the example, the program can proceed, but this warning indicates a possible error related to ID numbers. The Error example occurs in the **Variables** module for **Recode→Continuous** with an attempt to take the log of a variable which has data values of 0. The program cannot proceed with this, but the message is clear and the solution is simple: either do not try to take the log of this variable, or remove the 0 data values with **Recode→Zero->Missing** before proceeding. A list of all relevant programmed warnings and errors are appended at the end of each Section. Most of them occur in the **File** and **Variables** modules, based on the errors associated with importing datasets and transforming variables. However, not all possible errors can be anticipated.

An unexpected error is more serious, since it may indicate a bug in the program, or some error in the data that is not currently checked for. The message describes the type of error (in this case "LENGTH ERROR"), and the sub-routine and line number in which the error was detected (in this example "VS[2]"). The rest of the message may be unreadable. Errors of this type should be reported to this author. The report should include the text in the Internal Error window and, if possible, the dataset for which the error occurred. In addition, since the error may have occurred in some sub-routine, further unexpected



```
LENGTH ERROR
VS[2] ZIX,[1.5]Y
       ^
```

Figure 0.10. Unexpected error window
Labelled as an "Internal Error!!"
Type of error is "LENGTH ERROR"
Location of error is "VS[2]"

errors may occur when the first error window is closed. Each window should be closed in turn, but only the first error needs to be recorded by the user. Finally, MultiNet will return to the main menu, no matter which module caused the error. This allows the data to be saved before proceeding any further. It is usually possible to continue unless the problem is with the data.

0.8 Help concepts

Every MultiNet module, and many windows, have either **Help** as a menu bar item or a labeled button. For example, the Main Menu bar has **Help** as an item, and it is one of the two that are enabled when MultiNet first starts (the other is **File**). When **Help** is a menu bar item, clicking it provides a selection window listing all the other items on the menu bar for this module. Selecting one of these provides details about the actions of that menu choice in the current module (Figure 0.11) . On the Main window, **Help** also provides "OVERVIEW", which is a collection of all the Help information for all the MultiNet modules. This is a large amount of information, so the screens of information for each module are separated by a special character (Page Break or Form Feed). This technique is also used in the **Analyse** and **Pstar** modules when representing a large amount of information. The presence of the Page Break character causes the VIEW window to become a MULTIVIEW window, which has two extra items on the menu bar: **Last** and **Next**. These allow the user to step through the information which has been separated by the Page Break character, while still allowing the usual scroll bar controls within each screen (Figure 0.12). Last and Next are also menu items in the **Analyse**, **Variables**, and **Eigenspaces** menu bars, where they preform conceptually similar tasks.

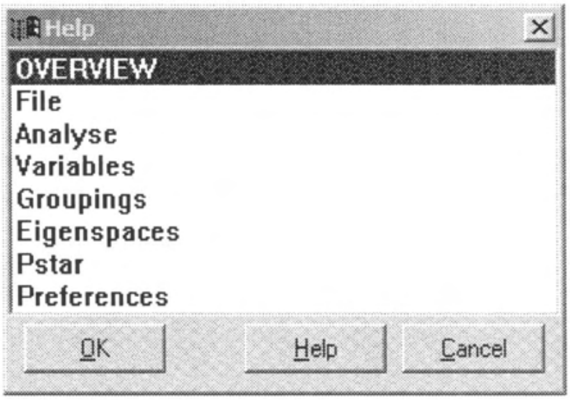
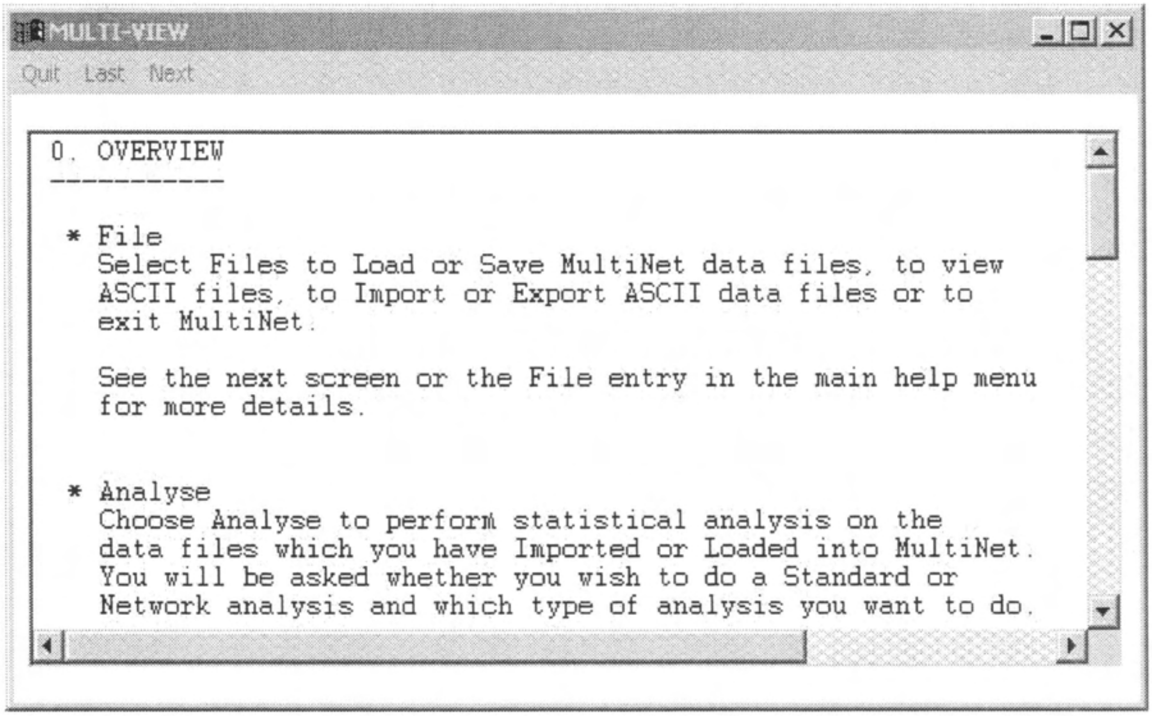


Figure 0.11. Selection window form Main window includes item OVERVIEW, which provides Help information from all the other modules.

When **Help** appears in a button on a window (e.g., Figure 0.11), it provides context-sensitive help, that is, information which provides an explanation about the response that is expected at this particular place in the program. For example, the response to clicking **Help** in figure 0.11 is shown in figure 0.3, which describes the standard methods for making a choice from a selection window.



```
0. OVERVIEW
-----------

 * File
   Select Files to Load or Save MultiNet data files, to view
   ASCII files, to Import or Export ASCII data files or to
   exit MultiNet.

   See the next screen or the File entry in the main help menu
   for more details.


 * Analyse
   Choose Analyse to perform statistical analysis on the
   data files which you have Imported or Loaded into MultiNet.
   You will be asked whether you wish to do a Standard or
   Network analysis and which type of analysis you want to do.
```

Figure 0.12. Main **Help** provides OVERVIEW which uses a MULTI-VIEW window to display a number of screens separated by Page Breaks. As well as the usual scroll bars, the menu items **Last** and **Next** are available for moving between screens.

0.9 Technical appendix 0

0.9.1 Data specifications

A) Dataset details

Every variable is stored as parsimoniously as possible. Binary data is stored as bit strings. Discrete data is stored as 1-byte (up to 256 values), 2-byte (up to 65536 values) or 4-byte integers. Real-valued data is stored as 8-byte IEEE format. All data is kept in memory for speed. Disk access is used only with the Main menu **File** selections, **Report→File**, and the **Graphics** selections.

There are two important data items which define the structure of all the node and link variables:

Node ID numbers (in the .NOD file) identify each node. Not every node needs to have a value for every attribute: a value may be "missing". The complete set of node ID numbers should equal the complete set of ID numbers that appear in the link pairs. If there are nodes that do not appear in the link pairs, this means that these nodes are not part of any network. This may indicate a coding error, so an informative message lists these nodes by ID numbers. If there are ID numbers in the link pairs that are not in the set of node ID numbers this is more serious, since it means that any node variable defined from networks which include these nodes must ignore these nodes. This may also indicate a coding error, so a warning message lists these node pair ID numbers.

Link ID number pairs (in the .LIN file) identify each link. In a network of n nodes, there are potentially n(n-1) ordered pairs of different nodes. However, most large networks are very sparse: not all node pairs have any values (other than 0). We need only store the ID pairs which have non-zero values. If there are multiple networks on the same nodes, we need only store those pairs which have a non-zero value for *any* link variable. In the KIDS2 data, for example, the 4th pair (1, 17) has a SAY value of 0, and a PLAY value of 1. At least one network (PLAY) has a non-zero value for this pair, so the pair must be stored. See KIDS2.LIN or Figures 1.1-1.3 in Section 1: The File Module.

Node and link pair ID numbers are generally stored as 4-byte integers, meaning that there is an upper limit of about 2 billion nodes. If necessary, ID values up to 2^{48} can be stored.

Every variable is associated with a group of supplementary descriptive variables, which are used to label displays and reports. These include:

- **variable name**: These are character strings of maximum length 24. They are defined in the Import file, and may be generated automatically when variables are created. The automatically generated names may be edited by the user.
- **value labels**: These may be defined in the Import file. They are character strings of maximum length 12. For categorical data with up to 20 discrete values, each discrete value may be given a textual descriptive label. These are also generated automatically in, for example, **Define→Partition**. For continuous data, value labels are undefined. If no value labels are available (e.g., for continuous data), then the actual values are used. For example, **Dots→Values** will use actual values to label nodes for a continuous attribute in **Eigenspaces**. Automatically generated value labels may be edited by the user.
- **comments**: These are textual descriptions of the variables which are up to 80 characters. Comments may be defined in the Import file. They are also generated automatically when variables are created. The automatically generated comments may be edited by the user.

The variable itself consists of two parts:

- **data values**: stored in the most parsimonious form.
- **missing data bits**: If there is no missing data, this is a single bit. If there is missing data, the number of 1-bits must equal the number of data values. The 0-bits indicate missing data for this node or link (ID pair). The total number of bits must equal the number of node ID numbers (for node variables) or link ID pairs (for link variables).

When a variable is chosen for any analysis, it is expanded from the parsimonious *passive* storage format, (data values), but still handled as a pair of items (data values, missing bit string) to minimize memory requirements. This is particularly important for link variables, which may be very large.

B) Storage rules

I. passive storage: Data is stored in one of two forms. The *passive* form is most compact, and is used both in files and in memory where data is not in use. When data is being used in a calculation, it is stored in *active* form, which is not as compact, but only exists in this form while in use. After use, the memory is released.

Table 0.1. Passive data storage

Data range	storage type
[0,1]	8 bits per byte
[0,255]	single byte
[0,65535]	double byte
[-2^{31},2^{31}]	4 byte
[-10^{308},10^{308}]	8 byte IEEE floating point format
any decimal values	8 byte IEEE floating point format
missing data	8 bits per byte. 0 if missing. 1's match data items
	If no missing data, a single bit.
ID numbers	Either 4-byte integer if $0<ID\#<2^{31}$ = 2,147,483,648, or
	8-byte IEEE floating point if $0<ID\#<2^{48}$ = 281,474,976,710,656
Labels	1 byte per character
Link ID pairs	As above (4-byte integers, or 8-byte IEEE floating point)
(Files only)	Or as unique items along with compressed indices into unique items, whichever takes less space

As an example of the last case, consider that in a network the same ID numbers may appear many times. For file storage purposes, it may be possible to store this information more efficiently by storing only the unique ID numbers along with indices into these unique numbers (in compressed form if possible). For example, the KIDS2 dataset has 227 ID pairs. The FROM ID numbers can be stored in 998 bytes. However, storing the 32 unique 4-byte ID numbers and 227 1-byte indices takes only 335 bytes. Similarly, the TO ID numbers can also be stored in 335 bytes. MultiNet calculates the storage required for both methods and selects the smallest. The savings can be substantial for large networks.

ii. active storage: When a variable is required in a calculation, MultiNet makes a copy of the data, converting from the passive storage form to a more convenient form for the calculation. After the calculation is complete, the memory required for the active storage is released for later calculations. The active storage of data can take advantage of the type of calculation to remain efficient in terms of storage or calculation or both. For example, the same method used to store ID pairs in files is used by the Analyse module for Network cross-tabs, ANOVA and correlation. This works well because the number of nodes is generally much smaller than the number of links, which are actually node pairs. Thus the same node attributes will occur multiple times, so that storing unique values and indices into the unique values is actually a very efficient way of handling the active storage of data. Also, in the Eigenspaces and Pstar modules, only the indices into link IDs are used, so that IDs may have a much larger range (up to 2^{48} for 8-byte values) than the values actually used in the calculations (up to 2^{31} for 4-byte values). So far these limits have been vastly larger than needed for any actual dataset.

Table 0.2. Active data storage

Data range	Storage type
[0,1]	8 bits per byte
[0,255]	1 byte
[-2^{31} , 2^{31}]	4-byte integer
[-10^{308},10^{308}]	8-byte IEEE floating point
any decimal number	8-byte IEEE floating point
missing data	8 bits per byte. 1's match data items. Single bit if no missing data.
Labels	1 byte per character
Node attributes of Links	As above for unique items, and 4-byte indices into unique values

0.9.2 Graphics specifications

A) Colour details

The following tables provide details about the colour used in MultiNet graphics displays. Table 0.1 specifies the mapping between colours and categories.

Table 0.3. Colours used to display categories in MultiNet graphic displays

Colour name	Colour number	category number	RGB values
dark blue	1	1st category	80, 80, 210
dark red	4	2nd category	200, 80, 80
dark green	2	3rd category	80, 200, 80
dark magenta	5	4th category	175, 75, 175
dark cyan	3	5th category	80, 180, 180
dark yellow	6	6th category	175, 175, 65
light blue	9	7th category	130, 130, 250
light red	12	8th category	250, 110, 110
light green	10	9th category	100, 220, 100
light magenta	13	10th category	220, 100, 220
light cyan	11	11th category	100, 220, 220
light yellow	14	12th category	190, 200, 100

Table 0.4. Other colours used in graphic displays. The two anaglyphic colours replace 8 and 7 only in anaglyphic 3-D displays, which do not use light or dark grey.

Colour name	Colour number	Graphic purpose	RGB values
black	0	default foreground colour	0, 0, 0
dark grey	8	cumulative curve, 3-D	140, 140, 140
light grey	7	axes, 3-D, de-emphasis	200, 200, 200
pure red	8	anaglyphic 3-D	255, 0, 0
pure cyan	7	anaglyphic 3-D	0, 255, 255
pure white	15	default background colour	255, 255, 255

B) PostScript details

PostScript is generated directly from the graphics primitives that are used to create the graphics displays. These primitives were chosen with this translation in mind, so the translation is fairly straightforward. The graphics screen is defined to have the same coordinates as a PostScript page, so that no translation of coordinates is required. Colours were chosen to appear similar in both additive computer displays and subtractive print displays. All the graphics primitives are translated by the set of definitions (Table 0.3) which is prepended to all PostScript output files.

Table 0.5. PostScript definitions prepended to every PostScript file. Comments added.

```
/A { .1 I {N 0 360 arc C G F R 0 P S} repeat 1 I} def % circle
/B {M 3 {L} repeat C G F R 0 P S} def % box with colour fill
/C {closepath} def
/D {3 1 roll M {L} repeat S} def % draw poly-lines
/E { 1 add 2 idiv {M L S} repeat} def % draw disconnected edges
/F {RGB aload pop setrgbcolor fill} def % fill area with colour
/G {gsave} def
/H {RGB astore aload pop} def % set RGB fill
/I {setlinewidth} def          /K {setrgbcolor} def
/L {lineto} def                /M {moveto} def
/N {newpath} def               /P {setgray} def
/R {grestore} def              /S {stroke} def
/T { {M 1 0 rlineto S} repeat} def % draw disconnected dots
/U {/Helvetica findfont exch scalefont setfont} def % set font
/V {0 eq {-90}{90} ifelse rotate U} def % set rotation state
/W {show} def
/X { 0.5 mul /x exch def x I % disconnected multi-coloured dots
    {M 0 x rlineto 1 eq { K } if S } repeat 1 I} def
/RGB [0 0 0] def % to hold current fill state
1 I 16 U 0 0 0 H 0 0 0 K % initialize
```

1. The File Module

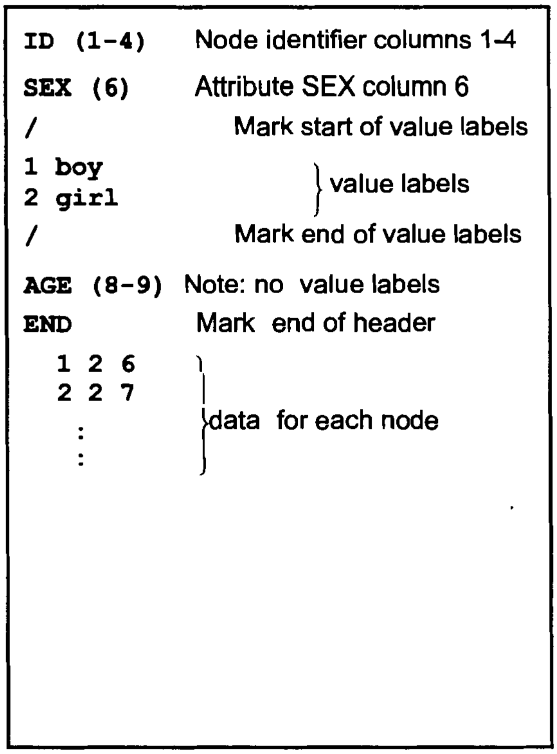
1.1 Introduction

File is the first of the eight items on the Main Menu bar, and is also one of the two items enabled when MultiNet first starts (the other is **Help**). While **Help** can be useful as an introduction to the MultiNet modules, no work can be done until a dataset has been read into MultiNet, and this requires the **File** module.

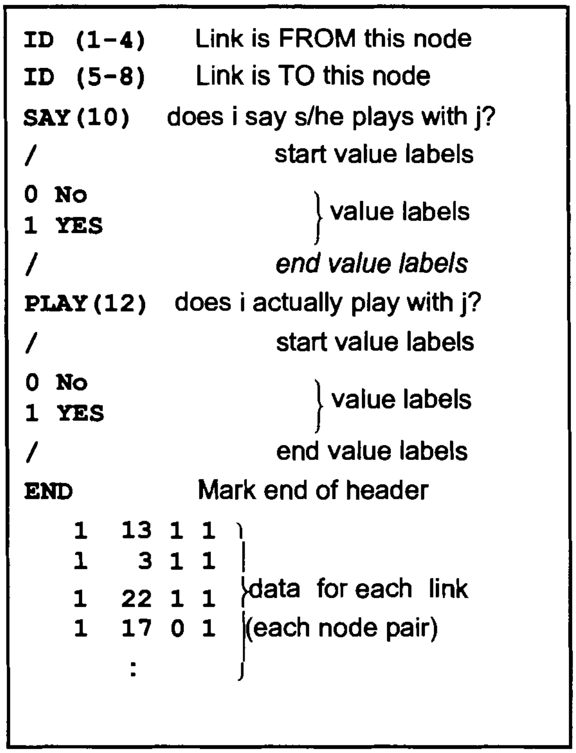
The **File** module is deceptively simple: it does not have its own menu bar, does not produce any graphics or reports, and does not seem particularly interactive – as long as no problem arise. Experience has shown that one of the most error-prone areas of data analysis is in the preparation of data for use by analysis programs. However, the same experience with potential data problems has been built into this module: There are many ways in which errors may creep into data, and as they have been encountered, more checks have been added. It is impossible to anticipate all possible sources of error, and some errors are subtle. An appendix to this section contains a list of all the checks against import error that are currently part of MultiNet, both those automatically performed by the **File** routines and those available in other parts of MultiNet.

1.2 Multiplex and sparse data representations

MultiNet was designed from the beginning to handle multiplex data and large, sparse networks. Multiplex data may have many attributes: there may be multiple attributes defined for the same node IDs or link ID pairs There is currently no limit (apart from memory) on the number of nodes and links that may belong to a MultiNet dataset, and it is very easy to create new node and link variables when desired. This flexibility requires a careful attention to the handling of large numbers of possible variables of both types. In addition, large network data is best handled by using one of the methods for representing such data in a sparse format, which includes only the ID pairs for which data actually exists. MultiNet uses a pair of files with similar names for the Node and Link variables. These file pairs can also come in two different but closely related file types: MultiNet text files and CSV (comma-separated variable) files. MultiNet text files will be described first, since there is a close relationship between the way variables are defined in the files, and the way they are stored internally.



```
ID (1-4)    Node identifier columns 1-4
SEX (6)     Attribute SEX column 6
/              Mark start of value labels
1 boy
2 girl             } value labels
/              Mark end of value labels
AGE (8-9)  Note: no  value labels
END          Mark  end of header
  1 2 6
  2 2 7
    :            }data  for each node
    :
```



```
ID  (1-4)    Link is FROM this node
ID  (5-8)    Link is TO this node
SAY(10)   does i say s/he plays with j?
/                         start value labels
0 No
1 YES                 } value labels
/                         end value labels
PLAY(12)   does i actually play with j?
/                         start value labels
0 No
1 YES                 } value labels
/                         end value labels
END            Mark end of header
   1   13 1 1
   1    3 1 1
   1   22 1 1   }data  for each  link
   1   17 0 1   (each node pair)
         :
```

a) KIDS2.NOD b).KIDS2.LIN

Figure 1.1. MultiNet Node and Link text files

To illustrate how sparse matrix methods are implemented in MultiNet, we now describe MultiNet data formats, and how to get network data into MultiNet. The data formats for both attribute (node) and network (link) data are intended to:

- allow for multiple attributes (node variables) and networks (link variables)
- use a minimum amount of space (both on disk and in memory)
- be easy to edit and maintain (the files are self-documenting)

The data format is an outgrowth of formats used by both NEGOPY and FATCAT (Richards, 1989,1995). MultiNet uses a pair of files with the same name and different extensions. The extensions are always .NOD for node (actor attributes) and .LIN for link (network attributes).

1.2.1 MultiNet text files

The example given here is KIDS2.NOD and KIDS2.LIN, which is data collected from a day care center (Richards, 1988). Each file begins with a *header* which defines what the data are, and where they occur in the file (by character position). Figures 1.1a and 1.1b show these two files, with the first few rows of data. The .NOD file describes the sex of the children (SEX) and their ages (AGE), so there are 2 node attribute variables. Individual children are identified by ID numbers, which is a common practice used to provide anonymity. Each .NOD file must begin with the declaration of the variable "ID", and the first character(s) of data must be the ID numbers. The first node variable is declared in the second line of the header. The variable name is taken as all characters (including blanks) up to the "(" character and may be up to 24 characters long. The position description is contained between "(" and ")" and may consist of a single number, or a pair of numbers separated by "-" for multi-position values.

The node variable SEX is in position 6, and can consist of two values: male or female. These are *categorical* values: descriptive labels with no natural ordering. It is common, and sometimes necessary (as it is for MultiNet and many statistical programs), to externally code such values as integers, and to represent them internally as integers. Because this is an integer encoding of a categorical variable, it is also very useful to provide *value labels* or category names that indicate the meanings of the numbers in the data. The MultiNet text files use a special format immediately following a variable declaration consisting of

- the character "/" to mark the beginning of the integer-to-value label descriptions
- a list of integers and value labels, one pair to each line (e.g. "1 male"), and with one pair for each integer that appears in the data for this variable
- a final "/" character to mark the end of the description

MultiNet makes a distinction between two types of variable: categorical (where the numerical representation is merely an encoding) and any other (where the numerical representation is an actual measurement). The type is decided automatically based on the number of unique values that a variable has (the number of *bins*). By default, a variable with no more than 12 bins is considered categorical. This number may be set as high as 20 in the **Preferences** module. Certain types of analyses (e.g. **XTABS**) are only allowed for categorical variables. In addition, categorical variables are expected to have value labels defined. If no value labels have been defined, the program will generate them automatically, based on the actual integer coding. Value labels may also be automatically created in the **Variables** and **Eigenspaces** module when a newly created variable has no more than the limited number of bins. On the other hand, MultiNet is forgiving when using categorical variables in types of analyses that expect non-categorical data (such as the calculation of simple descriptive statistics like mean and standard deviation).

The second node variable defined in KIDS2.NOD is AGE. This is not categorical and so no value labels are defined. However, there are only 5 bins and because the number of bins is not greater than the limit, MultiNet will allow this variable to be used wherever a categorical variable is expected, and will generate automatic value labels when required.

Even though both SEX and AGE can be treated as categorical data, they are distinctly different types of variables. AGE is a continuous phenomenon, though most people round it to the nearest number of years. It makes sense to talk of the mean and standard deviation of AGE, but not of SEX. Apart from the discrete/continuous distinction based on number of bins, MultiNet does not distinguish among nominal, ordinal and continuous data (Richards, 2004). For example, the program presents means and standard deviations of all variables. It is up to the user to determine when these results are meaningful.

The actual node variable data follows the set of declarations in the "header" (Figure 1.1a). Each variable takes up the column(s) described in the header. The internal storage of a node variable then consists of:

- the variable name (.e.g., SEX)
- the variable values (as defined by the contents of the position(s) declared in the header)
- optional value labels for categorical data
- an optional comment.

The comment can be up to 80 characters long, and is taken to be any text on a header line after the name and column declarations. For SEX in Figure 1.1a, the comment would be "Attribute sex column 6". Comments are very useful for describing data and MultiNet makes frequent use of them: whenever a new variable is created, the user is prompted for a descriptive comment.

The .LIN file header declares two ID numbers, since each link describes a relationship between a pair of nodes. ID1 is the ID numbers of the nodes that send links, and ID2 is ID the numbers of the nodes that receive links. The .LIN file must begin with the declaration of ID1 and ID2, and the first two variables must be pairs of node ID numbers. For every such pair, there may be any number of link variables, each of which describes a type of interaction or relation which defines a network. In this example, there are two link variables: SAY and PLAY. Both link variables may be considered categorical, with binary value yes or no. For SAY, a value of 1 means "ID1 says he or she plays with ID2". For PLAY, a value of 1 means "ID1 was seen playing with ID2". The coding 1 for yes and 0 for no allows the data to be stored internally as a single bit for each link value. Since binary link variables are very common, this is a very efficient way of storing such data. Thus a network of 65,536 binary links can be stored in only 8,192 bytes. For non-binary data, MultiNet will store the data in the most compact way, but integer and real link variables will obviously take up much more space.

MultiNet stores and manipulates networks using sparse methods based on the .LIN data format. That is, only the pairs of node ID numbers actually described in the .LIN file are stored. When multiple link variables are defined in a dataset, we need only describe those node pairs which have a non-zero value for any link variable. In the KIDS2 data, for example, in Figure 1.1b the 4th pair (1, 17) has a SAY value of 0, and a PLAY value of 1. At least one network (PLAY) has a non-zero value for this pair, so this node pair must be included, but since 1 did not nominate 17, the SAY value must be 0.

1.2.2 Missing data

Obviously, it is not necessary or desirable to describe pairs of nodes with a link value of 0, unless the pair takes or is given a non-zero value for another link variable. The value 0 (representing absence of interaction) is special for link variables, which must generally be non-negative. For node variables, 0 does not have any special meaning: a category may be encoded with value 0 without creating problems, and node values of 0 are never ignored. However, often a network dataset has *missing data* due to non-response during data collection. It is desirable to have a special data value other than 0 to represent missing data and MultiNet allows for the external dataset to have a missing data character. The default is a blank character, but the user may replace this with something else (preferably not a numeric character). Any variable with missing data is represented internally as a pair of lists: data values (in the most efficient storage) and a bit string marking which values are available (1) and missing (0). Any combination of variables can only use the intersection of non-missing values. For some types of variable, it may make sense to treat 0 data as missing (e.g., out-degree for one part of a two mode network, in-degree for the other part), or to treat missing data as 0 (e.g. for any link variable), so MultiNet allows these transformations in the Variables module. Working with missing data is discussed further, where relevant, in other sections in this document.

1.2.3 MultiNet CSV files

While MultiNet text files have certain advantages since they are in a *fixed* format, with the values of any variable always occupying the same character positions, making them easy to read and maintain. (Also, the MultiNet distribution package includes the programs FREE2FIX and ADJ2NEG, both written by W.D. Richards, that simplify translation of other common SNA formats to a fixed format). However, more recently SNA data has been appearing in spreadsheets such as Lotus, QuattroPro and Excel. Most of the advantages of the fixed format are supplied by the spreadsheet programs themselves. These programs also allow data to be imported and exported in a special type of text file with extension CSV for Comma-Separated Variables. These files are not fixed into columns. Rather, the data that would appear in columns is separated by a special character (usually a comma). This makes the files hard to read and maintain as text files, but hardly anyone actually does this, since the spreadsheet programs automatically resolve the separators so that the data appears in spreadsheet columns as if it was indeed in fixed format. Probably the biggest advantage of this format (apart from how common it has become) is that it allows easy insertion of new variables as columns. Adding new rows to text files is easy; adding columns requires much more knowledge of text-editing programs. In spreadsheet programs, it is easy. To illustrate these comments, Figure 1.1 presents the same data as Figure 1.2 in .CSV text format. Figure 1.3 shows the displays that result from reading this data into a spreadsheet program (QuattroPro).

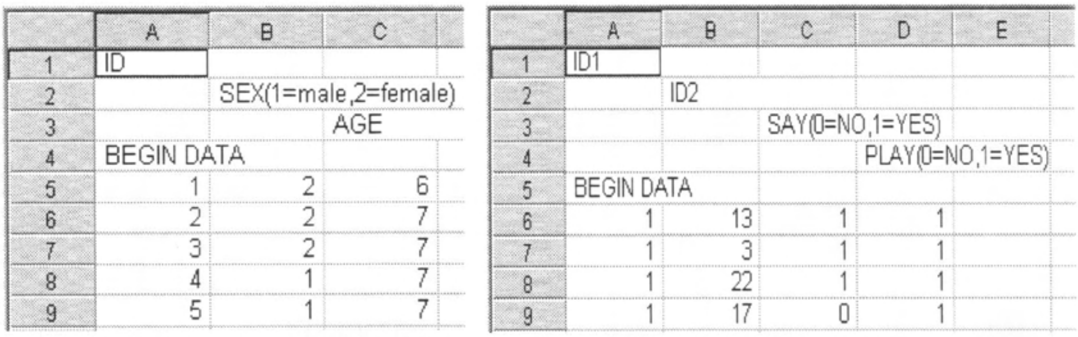
```
ID
,"SEX(1=male,2=female)"
,,AGE
BEGIN DATA
1,2,6
2,2,7
3,2,7
4,1,7
      :
```

a) Node variables NODKIDS2.CSV

```
ID1
,ID2
,,"SAY(0=NO,1=YES)"
,,,"PLAY(0=NO,1=YES)"
BEGIN DATA
1,13,1,1
1,3,1,1
1,22,1,1
1,17,0,1
      :
```

b) link variables LINKIDS2.CSV

Figure 1.2. Text display of CSV files



	A	B	C
1	ID		
2		SEX(1=male,2=female)	
3			AGE
4	BEGIN DATA		
5	1	2	6
6	2	2	7
7	3	2	7
8	4	1	7
9	5	1	7

a) Node variables NODKIDS2.CSV

	A	B	C	D	E
1	ID1				
2		ID2			
3			SAY(0=NO,1=YES)		
4				PLAY(0=NO,1=YES)	
5	BEGIN DATA				
6	1	13	1	1	
7	1	3	1	1	
8	1	22	1	1	
9	1	17	0	1	

b) link variables LINKIDS2.CSV

Figure 1.3. Spreadsheet display of CSV files

The biggest difference between CSV and the MultiNet formats is the treatment of value labels. These may be entered in the spreadsheet as shown – e.g., **`SEX(1=male,2=female)`** can be typed as is into cell B2. The extra pair of " quotes is provided during the production of the CSV files. It is not strictly necessary to put each variable name on a separate line, nor in different columns, though this makes the data easier to read and maintain and is the format used by MultiNet when creating CSV files. Because data is delimited by commas, there is no need for a special "missing data" character: any empty spreadsheet cell produces a pair of commas with no data between them and this is treated by MultiNet as missing data. However, some people use a special code for missing data (e.g., '*'), so this is also supported for CSV files. Comments are currently not supported in CSV files. The names of the CSV files must start with 'NOD' and 'LIN', since the '.CSV' extension is required by the spreadsheet programs.

Node files in either MultiNet or CSV format also support a special type of variable which can be used to label nodes with text data (e.g., names of individuals). The variable must be named IDLABEL (upper, lower or mixed case), with the usual definition of up to 20 characters for fixed format. These positions may contain non-numeric data which is then used in addition to ID numbers to label nodes. An example is NAMKIDS2.NOD in the distribution kit. Similarly, for CSV files the spreadsheet column for IDLABEL may contain non-numeric data. If IDLABEL is defined, these labels will be added to ID numbers in reports, and will replace ID numbers in Eigenspace displays.

Note that node or link files of either type may contain non-numeric data, as long as no declarations are made in the header that attempt to read these character positions or spreadsheet columns as numeric. Examples of this for IDLABEL are shown in Figure 1.4. In fact, any positions or columns that are not declared in the header are ignored when data is read in.

```
ID (1-3)
IDLABEL (5-12)
SEX (13-14)
/
1 male
2 female
/
AGE (15-17)
END
  1 Linda     2  6
  2 Jemima    2  7
  3 Bertha    2  7
  4 Jimmy     1  7
  5 Fred      1  7
  6 Rose      2  7
    :
```

a) IDLABEL declared

```
ID (1-3)
SEX (13-14)
/
1 male
2 female
/
AGE (15-17)
END
  1 Linda     2  6
  2 Jemima    2  7
  3 Bertha    2  7
  4 Jimmy     1  7
  5 Fred      1  7
  6 Rose      2  7
    :
```

b) not declared

Figure 1.4. IDLABEL as an example of non-numeric and non-declared data. In a) the declared IDLABEL columns result in the ID labels becoming part of the dataset. In b) the undeclared columns are completely ignored.

1.3 File menu selections

Figure 1.5 shows the selections available after pressing the **File** item on the Main Menu. Selecting one of these items (except the last) starts a process which either reads data from a file into MultiNet, or saves data from MultiNet into a file. The last selection provides one of the three methods for exiting MultiNet (all equivalent). Each selection has its own set of windows and associated Help buttons. The Main Menu **Help** item also provides a brief overview of the **File** choices which are summarized here:

1. **Load** reads a MultiNet system dataset.
2. **Save** saves a MultiNet system dataset
3. **View** allows viewing any text file
4. **Import** reads a MultiNet text or CSV dataset
5. **Export** saves a MultiNet text or CSV dataset
6. **Exit** exits MultiNet (after confirmation)

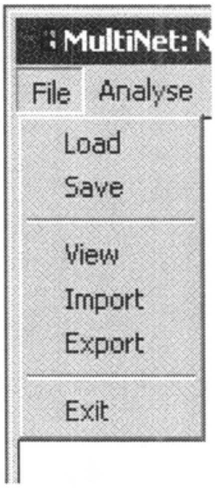
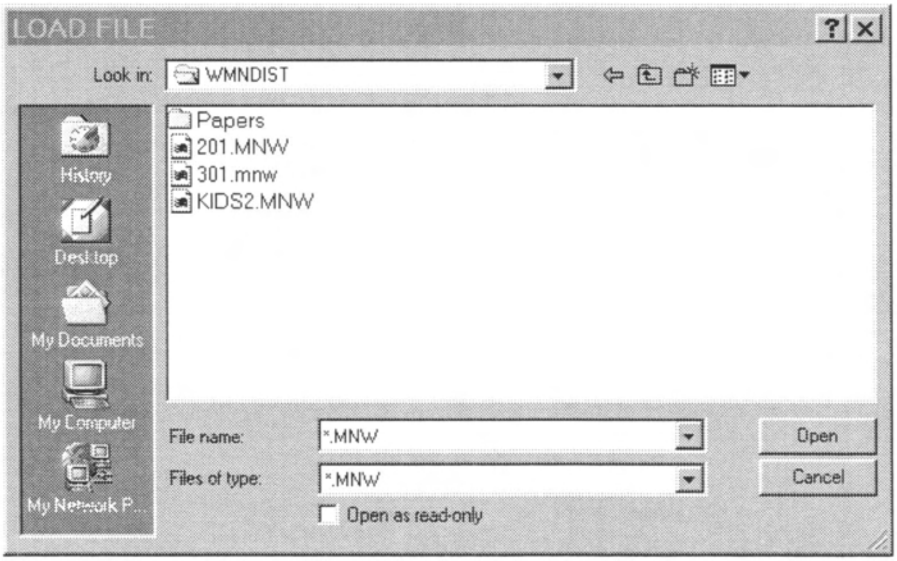


Figure 1.5.
File selections

1.3.1 Load

Load reads a MultiNet system file (with extension .MNW) into memory. MultiNet datasets can be very large as text or spreadsheet files. Once the data has been read into the program, everything has been translated into an internal format which usually takes much less space: the system file. The biggest savings in space come from storage of binary data, especially for link variables, where 1 and 0 are stored as single bits. As an example, one large dataset consisting of over 300 node and link variables takes up over 18 Megabytes as a MultiNet text file and takes many seconds for **Import**. In contrast, the corresponding system file takes up less than 3 megabytes and **Loads** almost instantly. The big disadvantage is that these system files can only be written and read by MultiNet.


LOAD FILE

Look in: WMNDIST

Papers
201.MNW
301.mnw
KIDS2.MNW

File name: *.MNW

Files of type: *.MNW

□ Open as read-only

Figure 1.6 Windows File selection

Load does not prompt before reading in the file. All current variable definitions are lost unless proceeded by a **Save** operation. The name of the file to be read in is chosen by the usual Windows methods (see Figure 1.6). The MultiNet title bar is updated to show the name of the system file that has just been read in.

1.3.2 Save

Save performs the action complementary to **Load**, saving the currently defined MultiNet node and link variables into a highly compressed MultiNet system file with extension MNW. The name of the file to be saved is chosen by the usual Windows methods (see Figure 1.6). If a file by that name already exists, a YesNo window asks whether it should be over-written. If No is chosen, the **Save** operation is terminated. If Yes is chosen the data is saved and the MultiNet title bar is updated to show the name under which the dataset has been **Save**d If the name (apart from the .MNW extension) under which the data is **Save**d is different from the previous name as shown in the MultiNet title bar, the program opens another YesNo window. This window asks whether the default names of certain automatically generated files should also be changed. These files have extensions .NOD, .LIN, .OUT, .PS and .BMP. For example, if the original file was named OLD.MNW and was **Save**d as NEW.MNW, then the program asks whether, for example, reports (PostScript files, Bitmap files) should now also be renamed as NEW.OUT (NEW.PS, NEW.BMP).

1.3.3 Import

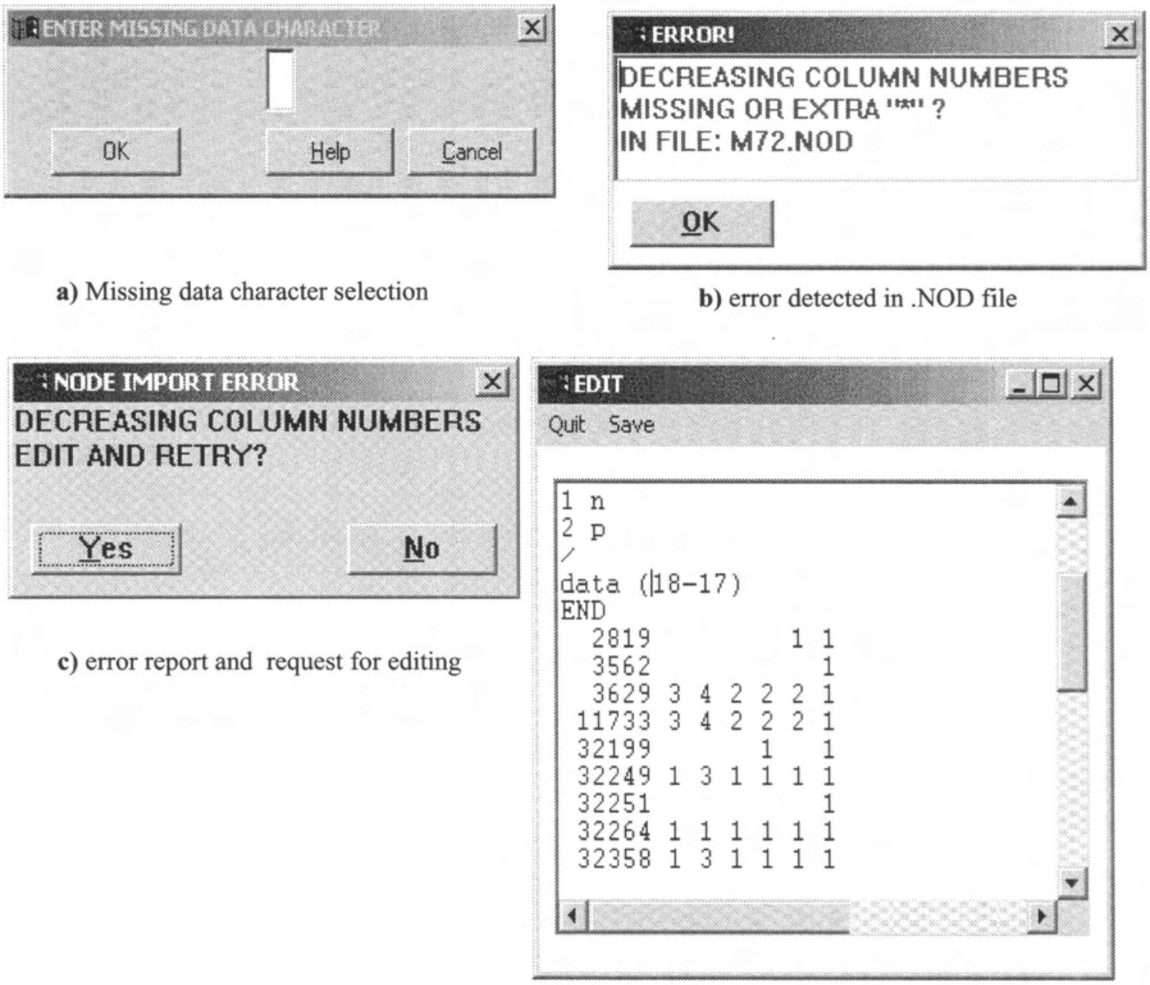
Although **Load** and **Save** are the most convenient and efficient methods for handling MultiNet datasets, **Import** is essential for first getting the a dataset into MultiNet. The most common format for moving data between different programs or even different operating systems is ASCII text files, and that is the type of file that **Import** deals with. These files are in one of the two formats described above: either MultiNet text files or spreadsheet-created CSV files. After selecting **Import**, a selection window opens for choice of one of the two file types. Once this choice has been made, a standard Windows file selection window opens. If MultiNet network is selected, all files are shown that:

- end with .NOD and
- match another file name that ends with .LIN (e.g., KIDS2.NOD and KIDS2.LIN)

If CSV is selected, all files are shown that:

- start with NOD and have extension .CSV and
- start with LIN, otherwise match the NOD file name and have extension .CSV (e.g. NODKIDS2.CSV and LINKIDS2.CSV)

Import then asks for a missing data character (Figure 1.7a). The default is blank (" "), unless it has been changed to some other character in character in the **Preferences** module. If the missing data character appears wherever numeric data is expected, the data item is marked as missing, and is not used in any analyses (though such missing data values may converted to 0's in the **Variables** module). **Import** then reads the NOD file into memory and converts it to internal format. If there are no errors, the LIN file is then read in and converted. If there are still no errors, the current dataset is completely replaced with the new dataset. The MultiNet title bar shows that the data has been read in as a pair of .NOD+.LIN files. Selecting "**Cancel**" at any time it is available halts the **Import**.



DECREASING COLUMN NUMBERS
MISSING OR EXTRA "*" ?
IN FILE: M72.NOD

DECREASING COLUMN NUMBERS
EDIT AND RETRY?

```
1 n
2 p
/
data (18-17)
END
  2819         1 1
  3562           1
  3629 3 4 2 2 2 1
 11733 3 4 2 2 2 1
 32199       1   1
 32249 1 3 1 1 1 1
 32251           1
 32264 1 1 1 1 1 1
 32358 1 3 1 1 1 1
```

a) Missing data character selection

b) error detected in .NOD file

c) error report and request for editing

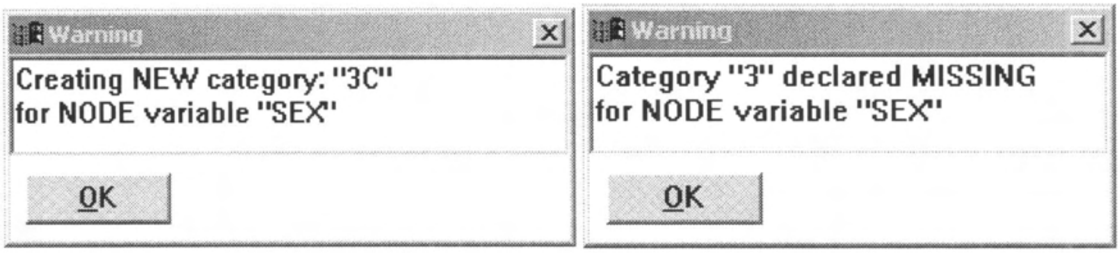
d) Edit window. Cursor is located where error detected.

Figure 1.7. Import interactions

There are many ways in which errors can creep into datasets, especially very large ones (see appendix), and the computer code for **Import** of MultiNet text files has many checks built into it that recognize certain common types of error (and some uncommon ones that have shown up). The more recent addition of CSV file support uses these checks by first converting the CSV files into the older fixed format MultiNet format. Some known errors are recoverable (e.g., positions in decreasing order in Figure 1.7b) Such recoverable errors cause the **Import** to suspend temporarily. An error window shows the detected error (Figure 1.7b). Then a YesNo window asks whether an attempt should be made to correct the error (Figure 1.7c). If “No” is selected, **Import** stops and the current dataset remains unchanged. If “Yes” is chosen, an edit window is opened holding the offending section of the text file (Figure 1.7d). The text cursor is placed at the position where the error has been detected. This allows correction of the error (in this example it is obvious that the character positions are incorrect). After making the correction(s), the edit window can be closed by either Quit or Save. Choosing Quit (or the standard Windows “X” or Alt-F4) discards any changes and halts the **Import**, leaving the current dataset unchanged. Choosing Save allows the **Import** to continue, as long as no more errors are detected. After one or more Saves and a successful **Import** another YesNo window asks whether to replace the text file that caused the error(s) with the Saved correction(s). If Yes is chosen, the original text file is renamed by replacing the last letter of the extension with “B” (for Backup), and the corrections are saved with the name of the original file. If No is chosen the text file has been successfully imported, but all corrections are lost and the original text file remains unchanged.

Data may have valid numeric values, but still be in error since the values are outside of the range expected (possibly due to a transcription error). If this can be the case for categorical data, it is one good reasons for declaring value labels. If **Import** finds a data value that is a valid numeric, but outside the declared range of values in the value labels declaration, it may:

- treat the offending value as an error and open an Edit window for correction, or
- create a new value label with label = value for the undeclared category, or
- treat the offending data item as missing data



Creating NEW category: "3C"
for NODE variable "SEX"

Category "3" declared MISSING
for NODE variable "SEX"

a) creating new category warning

b) undeclared category marked as missing

Figure 1.8. Handling categorical data outside the range declared by value labels

The first case may lead to a flurry of Edit windows, so is not used. Rather one of the last two actions is taken, based on a setting made in the **Preferences** module (**Categories→Create New**). In either case, a Warning window describes the action taken. Creating a new value label allows the data to be read in unchanged (Figure 1.8a), but the existence of an undeclared value shows that there may be an error either in the data or the value label declarations. Rejecting undeclared values as missing data (the default) gets rid of possible errors (Figure 1.8b), but at the cost of introducing missing data where none is expected. In either case, the original data should be checked for the source of the possible error. The example in Figure 1.8 suggests that the error is in the data but this is not always clear, and there is no general method to resolve the problem automatically.

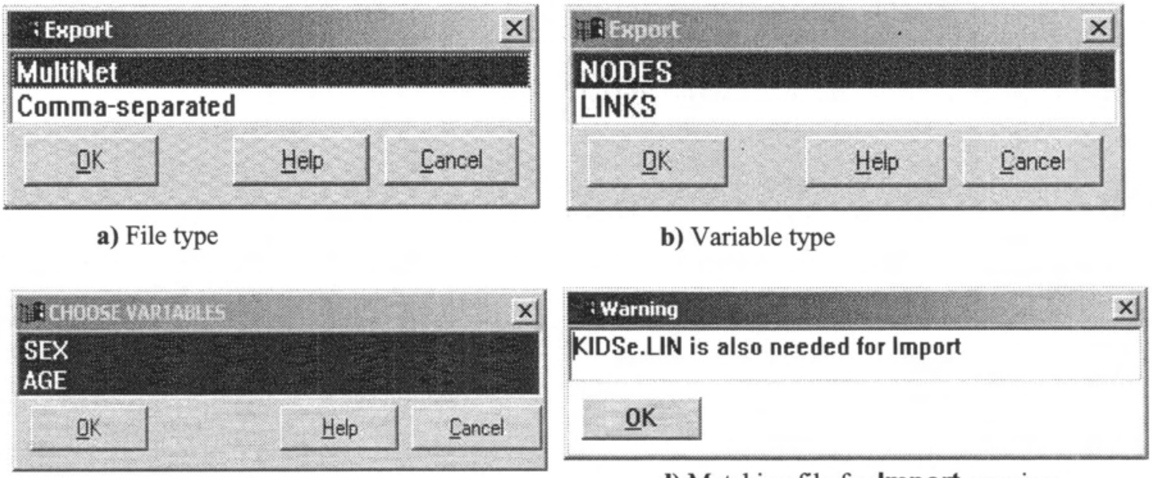
There is one other setting in the **Preferences** module that can affect the **Import**-ing of ASCII data. **Defaults→Categories** is set to a default value of 12, and may be set as large as 20. If a categorical variable has a set of unique values which is larger than this setting, then **Import** will treat the variable as continuous rather than categorical, and any declared value labels will be ignored. If categorical data seems to be missing declared value labels, this **Preferences** choice may have been set too small.

1.3.4 Export

MultiNet makes it easy to create new Node and Link variables, and **Export** makes it easy to write these variables out into ASCII text files for use by other programs. Just as for **Import**, both MultiNet fixed format and CSV files are supported, and as for **Import** this is the first choice made from a selection window (Figure 1.9a). This is followed by a selection window to choose Node or Link variables (Figure 1.9b) and then a multiple selection window to select which of the Node (or Link) variables to **Export** (default is all as in Figure 1.9c). Once these selections have been made, a standard Windows file menu is used to select the file name, with defaults:

- *.NOD (or .LIN) for MultiNet fixed format (e.g. KIDS2.NOD or KIDS2.LIN)
- NOD*.CSV (or LIN*.CSV) for CSV format (e.g., NODKIDS2.CSV or LINKIDS2.CSV)

where * is based on the file name of the current dataset. If the file does not exist, the chosen variables are **Export**ed to an ASCII text file in the chosen format. **Export** then checks to see
if the corresponding Link (or Node) file also exists, and displays a warning window if it does not (Figure 1.9d). Here the default name was replaced with KIDSe.NOD, so KIDSe.LIN would also be needed for **Import**.(if you wanted to **Import** the new file at another time).



KIDSe.LIN is also needed for Import

a) File type

b) Variable type

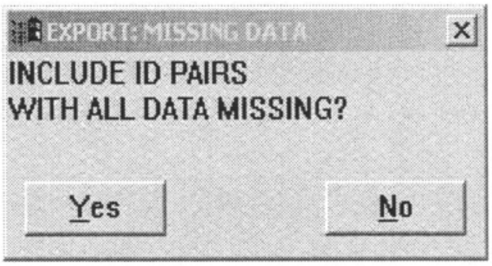
c) Variable choice

d) Matching file for **Import** warning

Figure 1.9. Export interactions

Export produces ASCII files that conform exactly with the rules **Import** requires for these files (either MultiNet fixed format or CSV files) while being as easy to read and maintain as possible. For example, all columns have spaces between them in fixed format (Figure 1.1); all variable definitions have their own row and columns in CSV format (Figure 1.2, 1.3). **Export** files of either type may be used as an example of how to construct such files, and it is instructive to use **Import** and **Export** to convert between the two types of files.

Export always uses the blank character for missing data. If **Export** detects that all of the variables chosen contain missing data for at least one ID number (Nodes) or ID pair (Links), another YesNo window (Figure 1.10) is opened before the file is actually written. It may or may not be useful to include these all-blank rows in either the fixed or CSV files. For example, updating an existing CSV file within a spreadsheet with a derived variable which has missing data would require including the missing data rows, so that IDs match up. In contrast, preparing a text file as input for another program would likely require eliminating all-blank rows.



INCLUDE ID PAIRS
WITH ALL DATA MISSING?

Figure 1.10.
YesNo window for **Export** of Link variables with missing data.

Selecting Yes will include all missing data values as blanks. Selecting No removes all ID pairs with completely blank rows.

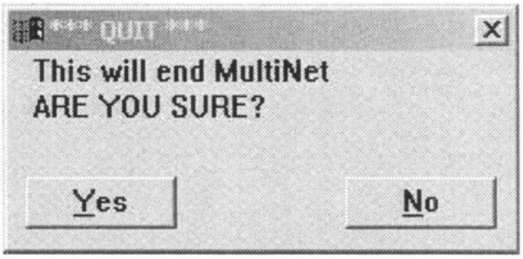
1.3.5 **View**

Since Windows allows multiple programs to be open, it is quite possible to view or edit textual or graphic input or output files created by MultiNet without leaving the program. However, this generally requires manoeuvring the file reading program (e.g., Notepad) into the current directory where the MultiNet input and output files are found. As a convenience **View** allows reading the text files that are in whatever directory MultiNet has been set to for reading and writing. **View** opens a standard Windows file selection window showing files with the output extension .OUT (where Reports go). The other built-in file extensions (in the "Files of type" menu) are .NOD, .LIN , and .CSV for the ASCII **Import** and **Export** file types. The extension may be changed to something else in the "File Name" input area, but the **View**er makes no special attempt to handle non-ASCII files, so the result may contain many unreadable characters. For ASCII files, the MultiNet View window will show the readable contents, along with standard scroll-bars for moving through the file. Since this is a View window, no editing is allowed, and the window is closed by clicking on Quit or "X"

1.3.6 **Exit**

It is possible to exit from MultiNet from within any module by using either of the standard Windows methods:

- Press Alt-F4 on the keyboard
- Left-click on the "X" button in the upper right corner of the MultiNet window.

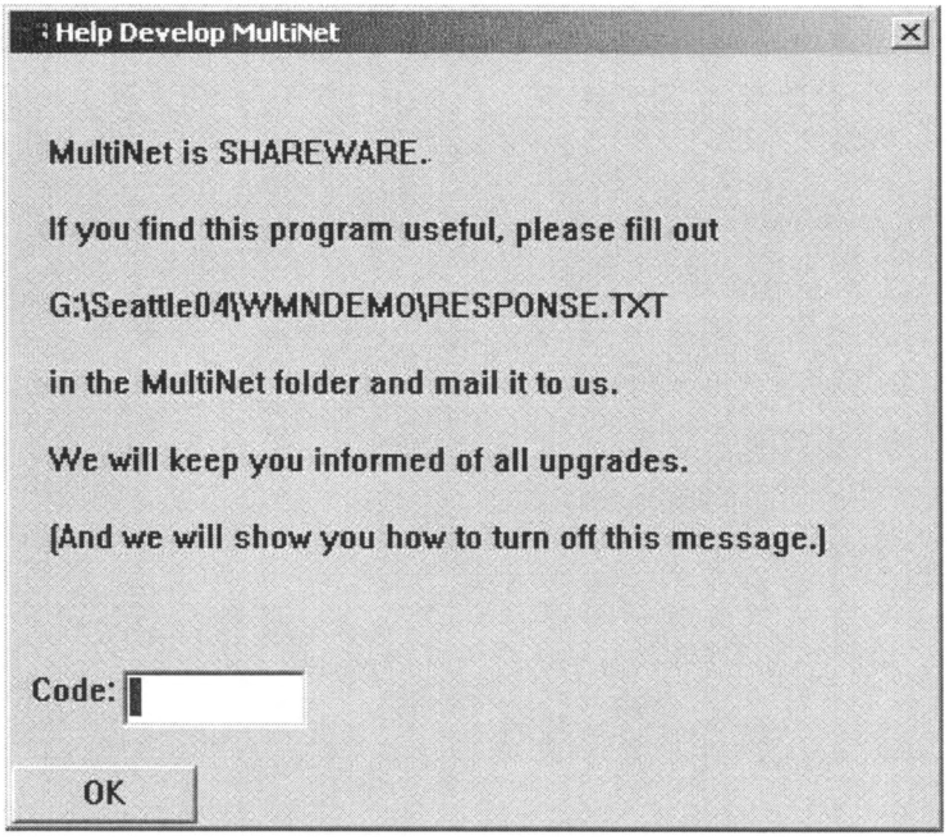


*** QUIT ***

This will end MultiNet
ARE YOU SURE?

Figure 1.11. YesNo window provides a last chance to avoid exiting.

The **File** menu provides the third, and preferred, method which is the **Exit** item. All three methods produce the same results. First, a YesNo window (Figure 1.11) opens confirming that the program should be terminated. If one of the exit methods has been chosen by accident, this is the last chance to cancel the exit (and cancelling a deliberate exit may be used to return to the default MultiNet screen ratio). This is a good time to confirm that any valuable work has been saved.



MultiNet is SHAREWARE..

If you find this program useful, please fill out

G:\Seattle04\WMNDEMO\RESPONSE.TXT

in the MultiNet folder and mail it to us.

We will keep you informed of all upgrades.

(And we will show you how to turn off this message.)

Figure 1.12. Final "nag" window.

Once the exit has been confirmed, a final window "nags" the user to send money for further development of MultiNet (Figure 1.12). The "nag" window can be permanently turned off by entering a code number in the Code field. This code is provided when a user has actually registered for MultiNet by sending a payment. The payment is small ($20 is suggested), and is used mostly to keep track of registered users to provide bug fixes and other up-grades.

1.4 References

Richards, W.D. (1988). *Private Communication*

Richards, W.D. (1989). *A Manual for FATCAT* (2nd edition). School of Communication, SFU.

Richards, W.D. (1995). *NEGOPY 4.30 Manual and user's Guide*, School of Communication, SFU.

Richards, W. D. (2004) *The Zen of Empirical Research*, Empirical Press

1.5 Technical appendix

1.5.1 List of errors anticipated by **Import.**

Generic values for File name, column numbers and variable name are represented by <File>, <n1-n2> and <Name>, respectively.

`Error Text` is followed by

- Explanation.

A) "Expected" errors

`<File> DOES NOT EXIST`

- Attempt to import .NOD file for which there is no .corresponding .LIN file

`<File> DOES NOT START WITH "ID"`

- Both .NOD and .LIN files must have "ID" (in either upper or lower case) as the first two characters of the file. These are part of the ID or ID1 data declarations.

`EXPECTED VARIABLE DEFINITION`
`WITH COLUMN NUMBERS AS (n1-n2).`
`MISSING "/" OR "END"?`

- While reading header information, expected to find a variable definition of the form <Name> (1-2) which defines a variable name and the columns the data values are in. Most likely this error was caused by an unfinished value label declaration, or by forgetting to mark the end of the header with "END", or "BEGIN DATA".

`EXPECTED TWO COLUMN NUMBERS`
`(n1-n2) AFTER VARIABLE NAME`

- Ill-formed expression after variable name. Possibly missing parentheses, or un-matched parentheses, or parentheses as part of variable name or missing "-" character between column numbers.

`DECREASING COLUMN NUMBERS`

- The column numbers in the (n1-n2) expressions must increase in value within the parentheses and for succeeding variable declarations.

MORE THAN 20 COLUMNS

- (n1-n2) must span no more than 20 columns. This error probably caused by typing a error.

DUPLICATE COLUMN NUMBERS

- The same columns (or overlapping columns) used for more than one variable. Probably a typing error.

EXPECTED VALUE LABELS
MISSING "/"?

- Probably missing ending "/" character in value label declaration.

NO DATA IN COLUMNS <n1-n2>
FOR VARIABLE "<Name>"

- The column described by (n1-n2) for variable <Name> is completely blank.
  Probably a typing error for column declaration.

EXPECTED NUMBERS IN COLUMNS <n1-n2>
FOR VARIABLE "<Name>"

- At least on non-numeric character in the columns declared for the variable.
  In Edit window, cursor will be placed at first such character.

DUPLICATE NODE ID

- Only for .NOD files. At least one ID number appears more than once.

NUMBER OF LINKS DO NOT MATCH
FROM: <n1>   TO: <n2>

- Only for .LIN files. The number of ID numbers in the from column do not match the number of ID numbers in the TO column. Probably at least one blank ID number.

NO DATA IN ANY SPECIFIED COLUMNS

- None of the declared data columns contained valid data. This can happen with a NEGOPY link file, which does not require actual data besides the ID pairs.

`NO FROM ID NUMBERS`

or

`NO TO ID NUMBERS`

- Only for .LIN files. Probably forgot to declare either ID1 or ID2 in header.

`ID NUMBER = 0 IN FILE <Name>`

- ID numbers must be positive and less than 248

B) "Unexpected" errors

The following error message follows an unexpected error that cannot be recovered from. The data could not be read in, and cannot be edited, but the current dataset (if any) is unchanged.

`UNEXPECTED ERROR IN FILE <File>`

- New type of error. Cannot be edited, since location is unknown.

C) Warning messages

The following messages appear after both files have been read in. They are warnings rather than errors. These warnings are repeated for any dataset (ASCII or system).

`SOME NODE IDs ARE NOT LINK IDs`

- Some nodes do not appear in any link variable. Since they are not part of any network, this may indicate transcription errors.

`SOME LINK IDs ARE NOT NODE Ids`

- Node that appear in networks are not declared, and so can have no node attributes. Any data derived from networks (e.g., degree, partition) must be declared as missing since there are no node IDs to attribute them to. This may also indicate transcription errors.

1.5.2 Other data checks

The following are not error messages, but are conditions that can be checked in the **Variables** module, under **Recode**. Some items in the **Recode** menu are enabled only when the data satisfies certain conditions. If an item is enabled unexpectedly, recheck both the original data and ASCII files. For nodes and links, **Select** variables, click **Recode**, and check the following:

For either Node or Link variables:

Zero->Missing is enabled only when at least one 0 is present as a variable value.

Missing->Zero is enabled only when at least one data item is marked as missing.

For Link variables:

Reduce MultiLinks is enabled only when the same From ID and To ID pairs appear more than once. For some network data this is meaningful, but for others it is not, and so indicates an error in the data.

No Diagonal is enabled only if at least one ID number appears as both sender and receiver, with a non-zero link value. For some network data this is meaningful, but for others it is not, and so indicates an error in the data.

2. The Analyse Module

2.1 Introduction

Analyse is the most complex MultiNet module given the number of different tasks it performs, and the number of menus, displays and reports it uses, but is also the simplest in the actual calculations it does, most of which are simple counting. While the counting calculations may be simple, organizing the things that need to be counted, displaying the results, calculating the relevant statistics and switching between types of counting makes up most of the complexity of this module. All of the tasks performed by **Analyse** follow the same pattern:

- Selection of type of counting to be done
- Selection of (at least two) variables
- Sorting the variables by increasing values
- partitioning the variables by unique values
- counting the number (and possibly the amount) within each partition
- performing appropriate statistical calculations and tests
- displaying the results with an appropriate graphic
- responding to user requests for more detail (**Report** and **Help**)

There are two main approaches to analysis: *Standard* and *Network*:

- Standard analysis is performed by most statistics packages that work with a single unit of analysis and a single rectangular cases by variables file of data. Here, the counting involves the values of Node or Link variables, and the total N is either the number of Nodes (actors or people); or the number of Links (ties or interactions) for which non-missing data exists. All variables must be either Node or Link.

- Network analysis allows mixing Node and Link variables in a way that is unique to MultiNet. In Network analysis, the total N is the number of Links (or amount of interaction if Link variables have non-binary values) between the Nodes in the network which have non-missing data. Network analysis always involves directed links. Links come 'FROM Senders' and go 'TO Receivers'. For analytic purposes, the 'sender' is the Node describing the Link, and the 'receiver' is the Node the sender says the Link is to. (Usually Nodes are people, and Links are interactions.) The same pair of Nodes may have multiple Links between them (for example, at different times). The networks may be completely directed and have many components (as is often the case with *ego-centric* network data).

For each of these two approaches, there are three kinds of analysis which depend on whether the variables are categorical/discrete or numeric/continuous (this choice is partly left up to the user, though warnings are given if a choice is inappropriate):

- Crosstab (**XTAB**) analysis for pairs of categorical or numeric variable
- Analysis of Variance (**ANOVA**) for one numeric/continuous and one categorical or numeric variable
- Correlation (**CORREL**) analysis for pairs of continuous numeric variables.

MultiNet generally considers the number of unique values (Bins) for each variable to determine whether a variable could be considered categorical/discrete or numeric/continuous. The user has some control over this, and may choose (in Preferences) a maximum of 20 for 'Categories', so that a variable with 20 or fewer bins is considered allowable as a discrete/categorical variable. Care must be taken here since this could lead to 20-by-20 crosstab tables or an **ANOVA** with an independent variable that has 20 different values (resulting in 20 groups). Either is a challenge to absorb and interpret and may be problematic because the number of cases in each cell or group may be too small unless the data describes a reasonably large network. Any warning about an inappropriate datatype generated by the program may be ignored by the user (also with care!) if a variable with a small number of bins is a legitimate numeric/continuous datatype (e.g., childrens' ages).

A minimum of two variables is necessary to begin one of the analyses available in this module. However, MultiNet allows the user to **Stack** an analysis on a third variable (which must be categorical/discrete) and then to **Rotate** the variables (which swaps the current independent variable and the Stack variable) and does the appropriate analysis; a second **Rotate** swaps the current dependent variable with the original independent variable and does the appropriate analysis; a third **Rotate** swaps the current dependent variable (the **Stack** variable) with the original dependent variable, going back to the original analysis. MultiNet is unique in providing the choice of a third variable, and then allowing the selection of any pair of the three variables, with automatic choice of the type of display and report to accompany the selected pair. Further, if the currently unselected variable is categorical/discrete, the user may "step" through the values of this variable in either direction for both visual display and textual report. For Network analyses, the user may also select a fourth variable which may be a Link (thus defining a network) or the result of an equation based on one or more link variables. The use of multiple link variables is further extended by support for the concept of a Grouping, which is a user-defined set of Link variables (see Section 4: The

Groupings Module). A grouping is considered categorical, so it can include no more than 20 link variables. It must also be chosen as the first or second variable where a categorical/discrete variable is expected (**XTABS** or **ANOVA**, but not **CORREL**).

Table 2.1 shows the type of analysis automatically selected for each possible pair of variables in a trio of variables. More information on the displays and reports available for each type of analysis is provided later in this section.

Table 2.1. MultiNet automatic selection of display/report based on data type.

	Type of 1st variable	
Type of 2nd variable	**Discrete**	**Continuous**
Discrete	**XTAB**	**ANOVA**
Continuous	**ANOVA**	**CORREL**

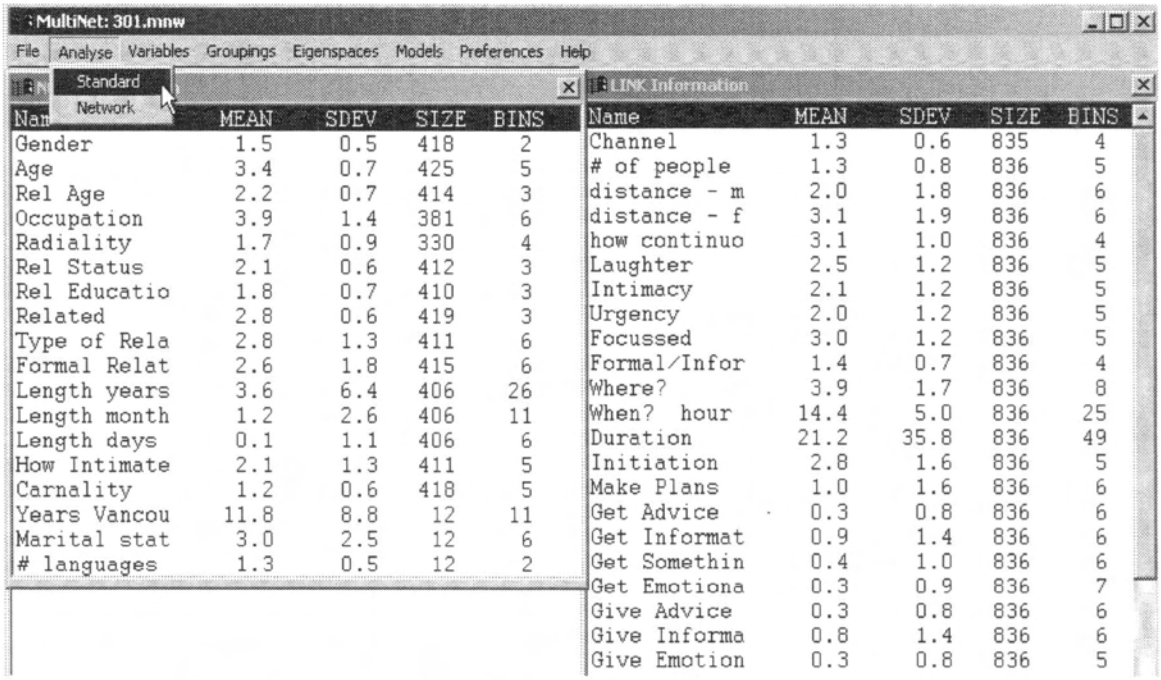
The dataset used to illustrate most of the capabilities of the Analyse module is 301.MNW. This data was collected by 12 students in Communications 301 at Simon Fraser University, a course given by Prof. W.D. Richards. The students recorded every interaction they had over a one-week period[1], including personal details about the other person such as gender (categorical), age (categorical) and years of residence (numeric) as well as communication channel (categorical), duration (continuous), time of day (numeric), and how much of each of several purposes and about each of several contents (allowing categorical groupings). The dataset thus allows for all possible combinations of both Node and Link variable datatypes. The network(s) are ego-centric and have multiple interactions between pairs of nodes. These networks are also completely directed and disconnected, with 12 components (one for each student). The dataset is thus too fragmented for detailed network visualization using the Eigenspaces module. However, the methods of the Analyse module allow visualization of the results of analysis that showed some patterns about which the students would complain (without irony) that it made them look stereotypical.

2.2 Standard Analysis

Most of the concepts used in doing any of the Standard analyses should be very familiar, so this section will be used to introduce the new and unique techniques available in MultiNet for visualization and interaction with both displays and reports. Figure 2.1 shows the MultiNet display, after

[1] By the end of this period, students would cross the street to avoid people they knew, so they wouldn't have to record the interactions with them

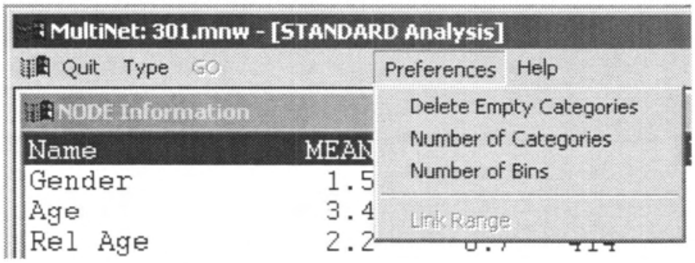
clicking the **Analyse** item on the Main Menu..Also shown are the 18 Node variables and 30 of the 45 Link variables of the 301.MNW dataset, Clicking on the **Standard** item results in the Standard Analyse menu bar shown in figure 2.2. Four items are enabled. **Quit** returns to the main menu; **Help** provides context-sensitive descriptions of the choices obtained by clicking **Type** (Figure 2.3). The **Preferences** choices are displayed in Figure 2.2, and described in more detail in Section 7:



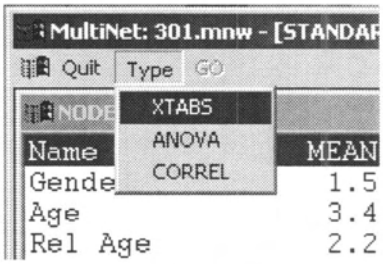
Name	MEAN	SDEV	SIZE	BINS
Gender	1.5	0.5	418	2
Age	3.4	0.7	425	5
Rel Age	2.2	0.7	414	3
Occupation	3.9	1.4	381	6
Radiality	1.7	0.9	330	4
Rel Status	2.1	0.6	412	3
Rel Educatio	1.8	0.7	410	3
Related	2.8	0.6	419	3
Type of Rela	2.8	1.3	411	6
Formal Relat	2.6	1.8	415	6
Length years	3.6	6.4	406	26
Length month	1.2	2.6	406	11
Length days	0.1	1.1	406	6
How Intimate	2.1	1.3	411	5
Carnality	1.2	0.6	418	5
Years Vancou	11.8	8.8	12	11
Marital stat	3.0	2.5	12	6
# languages	1.3	0.5	12	2

Name	MEAN	SDEV	SIZE	BINS
Channel	1.3	0.6	835	4
# of people	1.3	0.8	836	5
distance - m	2.0	1.8	836	6
distance - f	3.1	1.9	836	6
how continuo	3.1	1.0	836	4
Laughter	2.5	1.2	836	5
Intimacy	2.1	1.2	836	5
Urgency	2.0	1.2	836	5
Focussed	3.0	1.2	836	5
Formal/Infor	1.4	0.7	836	4
Where?	3.9	1.7	836	8
When? hour	14.4	5.0	836	25
Duration	21.2	35.8	836	49
Initiation	2.8	1.6	836	5
Make Plans	1.0	1.6	836	6
Get Advice	0.3	0.8	836	6
Get Informat	0.9	1.4	836	6
Get Somethin	0.4	1.0	836	6
Get Emotiona	0.3	0.9	836	7
Give Advice	0.3	0.8	836	6
Give Informa	0.8	1.4	836	6
Give Emotion	0.3	0.8	836	5

Figure 2.1. Selecting **Analyse →Standard** from the Main Menu for dataset 301.MNW



Name	MEAN
Gender	1.5
Age	3.4
Rel Age	2.2



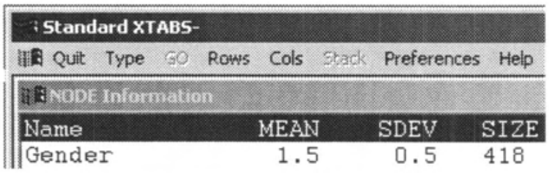
Name	MEAN
Gende	1.5
Age	3.4
Rel Age	2.2

Figure 2.2. Standard Analysis menu with **Preferences** menu displayed. These choices are made available here as a convenience.

Figure 2.3. Analysis Type Choices

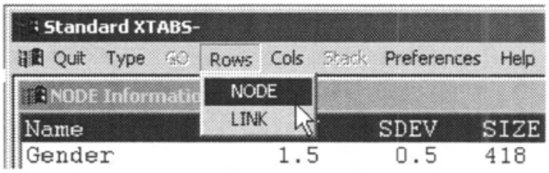
2.2.1 Cross-tabulations (XTABS)

To start a cross-tabulation analysis, click **Type** and then click **XTABS** as shown in Figure 2.3. At this point the menu bar changes as shown in Figure 2.4a. At this point, the title bar of the MultiNet window says "**Standard XTABS**". Since no variables have been selected, none appear on the title bar. The menu choices are now appropriate to building a cross-tabulation table, which requires selecting Row and Column variables for counting (represented as **Rows** and **Cols**). Either may be chosen first. Two items are greyed-out (disabled): **GO** performs the analysis, and **Stack** allows a third variable to be selected. Neither can be chosen until at least two variables have been selected. Clicking on either **Rows** or **Cols** produces the choice of Node or Link variables shown in Figure 2.4b. Standard analyses does not allow choosing both Node and Link variables together, so that once a Node (Link) variable is chosen, the choice of Link (Node) variables is disabled. You may switch between Node and Link by starting over with choosing analysis Type (Figure 2.3). At any time up to clicking the **GO** menu item, the choices presented in Figures 2.3 (type of analysis) or 2.4 (selection of variables) may be changed.



Name	MEAN	SDEV	SIZE
Gender	1.5	0.5	418

Figure 2.4a. Menu bar after choosing **Xtabs**



Name	SDEV	SIZE
Gender	1.5	0.5	418

Figure 2.4b. Menu bar after choosing **Rows**

Assume we wish to look at the relationships among gender of the respondents (**Rows:** Gender), the ages of the students (**Cols:** Age) and the ages of the people they reported communicating with (**Stack:** Rel Age). These are all NODE variables. Each of these variables is chosen from the selection list of Node variables. Figures 2.4a and b show both **GO** and **Stack** are greyed out as they are not yet available.

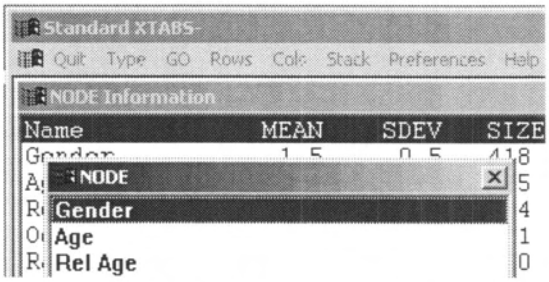
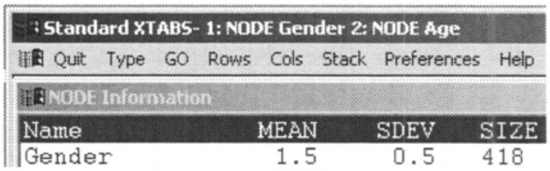


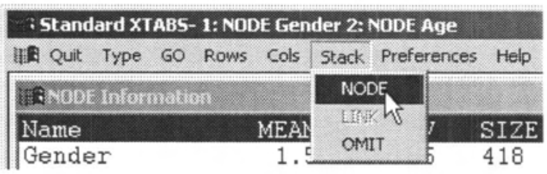
Figure 2.4c. Gender being selected for **Rows**



Name	MEAN	SDEV	SIZE
Gender	1.5	0.5	418

Figure 2.4d. Menu bar after choosing variables for **Rows** and **Cols**.

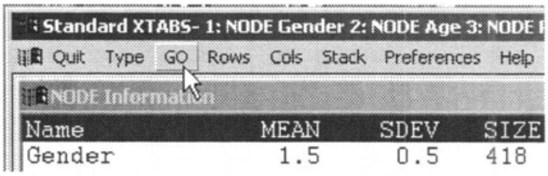
As variables for **Rows** and **Cols** are chosen, the texts "**1: NODE Gender" and " 2: NODE Age**" are appended to the title bar. After they have been chosen, both **GO** and **Stack** are enabled (Figure 2.4d), allowing an optional third variable to be chosen to **Stack** the analysis (Figure 2.4e). After the third variable has been chosen, all three variables appear on the title bar (Figure 2.4f).



Standard XTABS- 1: NODE Gender 2: NODE Age

Name	MEAN	SDEV	SIZE
Gender	1.5	0.5	418

Figure 2.4e. First step to select a third variable to stack the analysis on

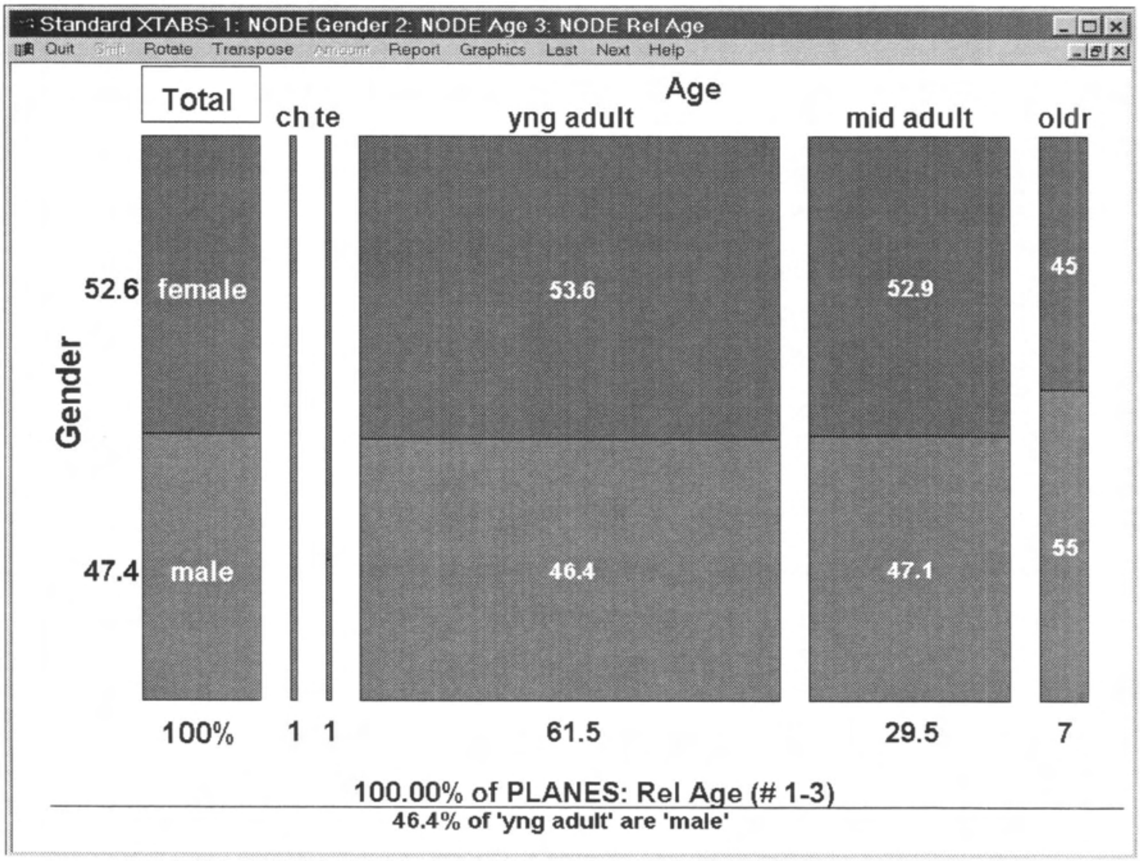


Standard XTABS- 1: NODE Gender 2: NODE Age 3: NODE R

Name	MEAN	SDEV	SIZE
Gender	1.5	0.5	418

Figure 2.4f. Ready to click **GO** to perform analysis

Clicking **GO** performs the analysis and produces the display shown in figure 2.5. Before discussing the details of this display, notice that the overall impression is that the blue and red proportions are roughly the same for all of the vertical columns -- the heights of all of the blue segments are about the same as the height of the blue segment in the Total column. This indicates there is a very weak relationship or no relationship between the row and column variables in the crosstabulation – that they are more-or-less independent. The actual cross-tab table can be seen by clicking **Report** and **View** (Table 2.2a). The impression is confirmed in Table 2.2b, which shows that Chi-squared for this table is 4.948, so small that the probability that this could be due to sampling variability is high (p>0.10), enough that we should fail to reject the null hypothesis which says that there is no relationship between Gender and Age for this dataset.



Standard XTABS- 1: NODE Gender 2: NODE Age 3: NODE Rel Age

Figure 2.5. Panigram showing the Column percents of the Gender by Age crosstabulation table.

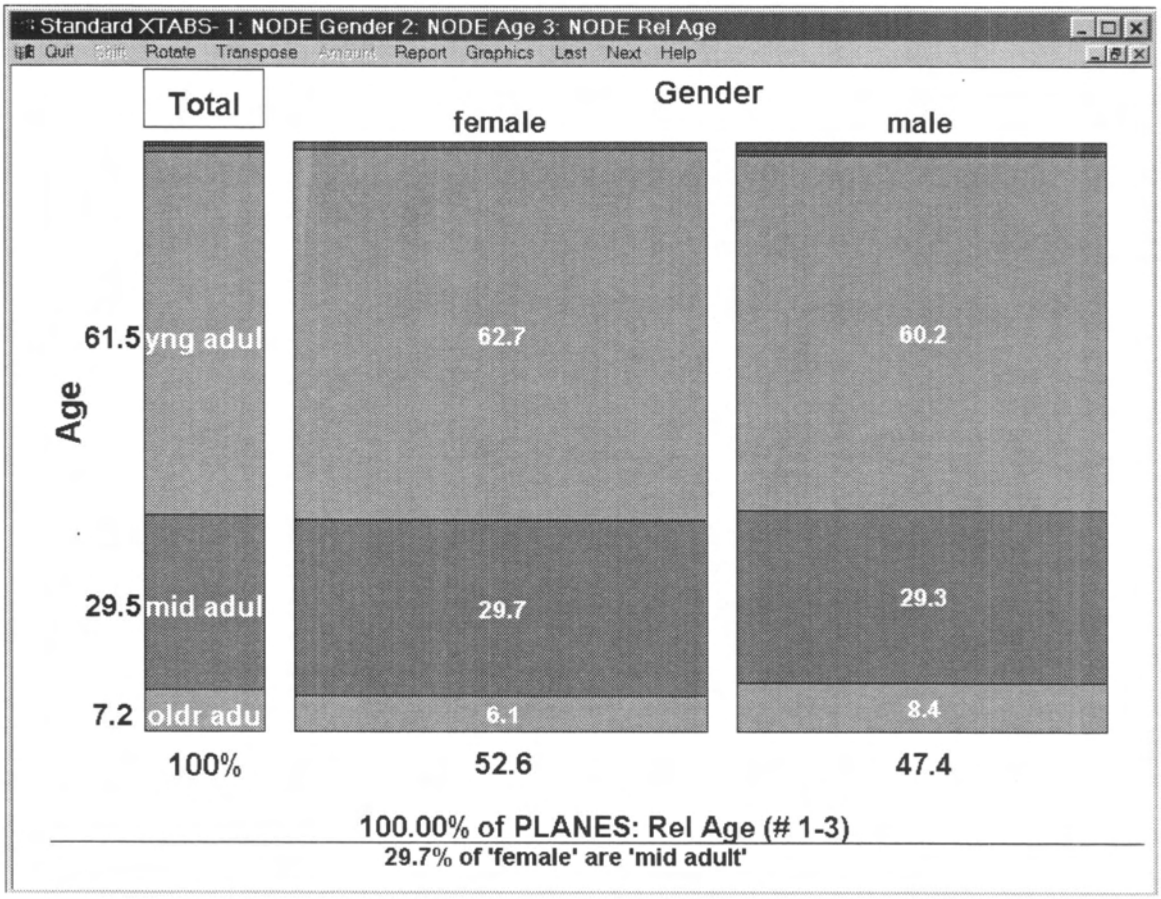
Table 2.2a. Counts, Row percents and Column percents of Gender by Age crosstabulation.

```
MultiNet 3-D STANDARD XTABS REPORT ON "301.mnw" 19/05/2004  16:21:16
ROWS: "Gender"   COLUMNS: "Age"   PLANES: "Rel Age"
 -------------------------------------------------------------
  100.00% of PLANES: Rel Age ( # 1-3)
Crosstabulation of Gender with Age

COUNT
ROW %                    ROWS = Gender
COL %                    COLS = Age
                      teen            mid adul
            child          yng adul          oldr adu
                                                     TOTAL
        ----------------------------------------------------
        |     0       3       133       63       13      212
 female |   0.0 %   1.42%   62.74%   29.72%    6.13%   52.61%
        |   0.0 %  75.0 %   53.63%   52.94%   44.83%
        |
        |     3       1       115       56       16      191
   male |   1.57%   0.52%   60.21%   29.32%    8.38%   47.39%
        | 100.0 %  25.0 %   46.37%   47.06%   55.17%
        |
        |     3       4       248      119       29      403
  TOTAL |
        |   0.74%   0.99%   61.54%   29.53%    7.2 %
```

Table 2.2b. % Difference from expected values, Chi-squared, and *Cramer's Phi.*
For 396 out of 403 (98%) of the counts, there is little variation from the expected values

```
  100.00% of PLANES: Rel Age ( # 1-3)
Crosstabulation of Gender with Age
  % Difference from Expected Values
 (Assuming row independent from column)

 % Diff                  ROWS = Gender
                         COLS = Age
                    teen           mid adul
          child           yng adul         oldr adu
        ---------------------------------------------
 female | -100.%     43.%       2.%      1.%    -15.%
        |
   male |  111.%    -47.%      -2.%     -1.%     16.%

 CHI-SQUARE =     4.948   D.F. = 4   P > 0.10  Cramer's Phi= 0.11
```

Table 2.2a is an ordinary crosstab table. It has three sets of numbers: Counts, Row percents and Column percents. The display in Figure 2.5 shows the Column percents, which assumes that Gender is the *independent* variable, and. is enough to produce a visualization that displays no relationship. The numbers in the left-most vertical column are 'female = 52.6%' and 'male = 47.4%', which are the *row marginal percents*. The numbers in each cell of the other vertical columns are the column percents for each of the values of the variable Age. They show, for example, what proportion of the people in the 'yng adult' column are women (53.6%) and what proportion are men (46.4%). The width of each of these columns correspond to the column marginal percents , and these are labelled at the bottom of each column. This display, called a ***panigram***, is a two-dimensional analogue of a histogram.



100.00% of PLANES: Rel Age (# 1-3)

29.7% of 'female' are 'mid adult'

Figure 2.6. Transpose of the panigram in Figure 2.5 is based on the Row percents.

To get a panigram based on the Row percents of Table 2.2a, with Gender treated as the independent variable, click **Transpose**. This will produce the display in Figure 2.6. The row percents shown in Table 2.2a correspond with the numbers shown in each cell in Figure 2.6, while the numbers to the left of the Total column in the panigram are the column marginal percents and the numbers at the bottom are the row marginal percents in Table 2.2a..

Notice that the marginal percents and the labels shown in Figure 2.5 are reproduced in Figure 2.6. Note also that the percents in the cells are not the same for the two figures. Figure 5.2 shows that 53.6% of 'yng adult' are female (blue), while Figure 6.2 shows that 62.7% of 'female' are 'yng adult'(green). This is due to the differences between row and column percents in Table 2.2a. The colours in the boxes come from the colours assigned to the values in the 'Total' column.

The rules for labelling a panigram are:

- each column is given enough width to display at least the first two letters of each value label at the top
- each column is labelled at the bottom with no less than 1 and no more than 3 significant digits of the marginal percentages.
- each box is labelled only if there is both sufficient height and width to hold at least 1 and no more than 3 significant digits of the percentage corresponding to its area
- each row is labelled at the left with 3 significant digits of marginal percentages and in the Totals column with up to 8 characters of value label if the height of the row is sufficient
- The boxes are coloured based on the values of the row variable, with colours ordered as described in Appendix 0, Table 0.1
- An explanatory description of the second labelled box in the first column is shown below the line at the bottom of the display.

Figure 2.6 lacks labels and percents for the 'child' and 'teen' rows, since their marginal percents are so small. The explanatory comment below the line **27% of 'Female' are 'Mid Adult'** helps to interpret what the panigram is displaying. This comment always describes the contents of the second cell which shows a value. The colours in each cell are determined from the panigram's row variable ('Gender' in Figure 2.5, 'Age' in Figure 2.6). Thus for Figure 2.5, 53.6 % of 'yng adults' are 'female' and 46.4% are 'male'; for Figure 2.6, 62.7% of 'female' are 'yng adult', 26.7% are 'mid adult', and so on. The values for all of the cells in either figure are shown in the textual Report (Table 2.2a), and are also available by using Interactive **Help** (described below).

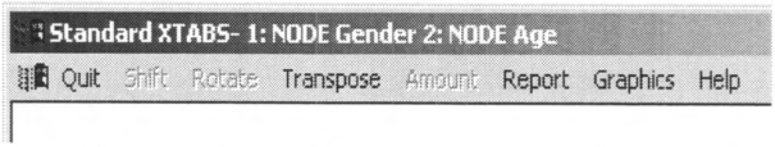


Figure 2.7a. Menu bar for **XTABS** with no stack variable

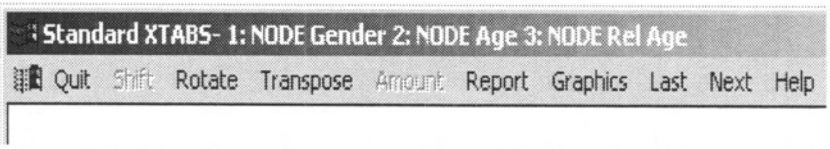
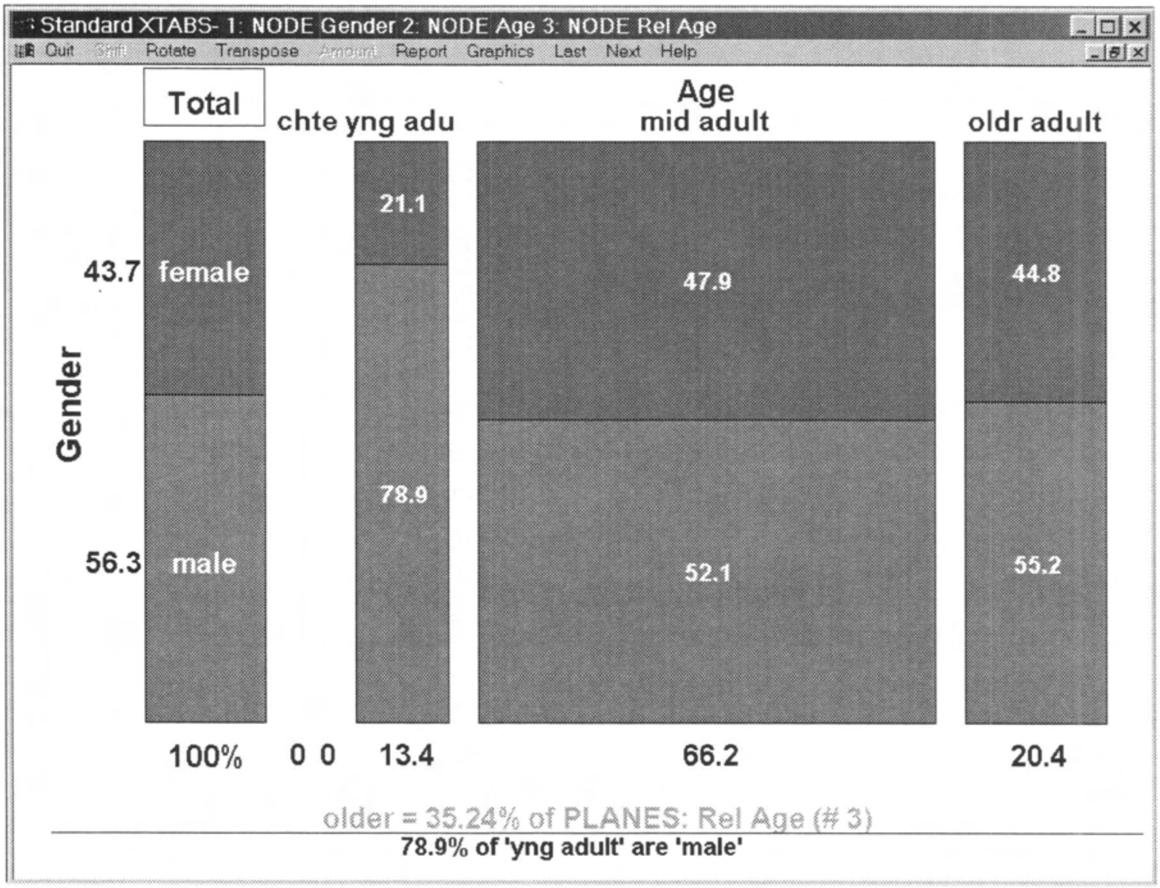


Figure 2.7b. Menu bar with stack variable includes **Rotate**, **Next** and **Last**

If only two variables had been chosen for the **XTABS** analysis, the menu items **Rotate, Next** and **Last** would be missing (Figure 2.7a). The title bar and menu items in Figure 2.7b shows that a third variable 'Rel Age' has been included with **Stack**.

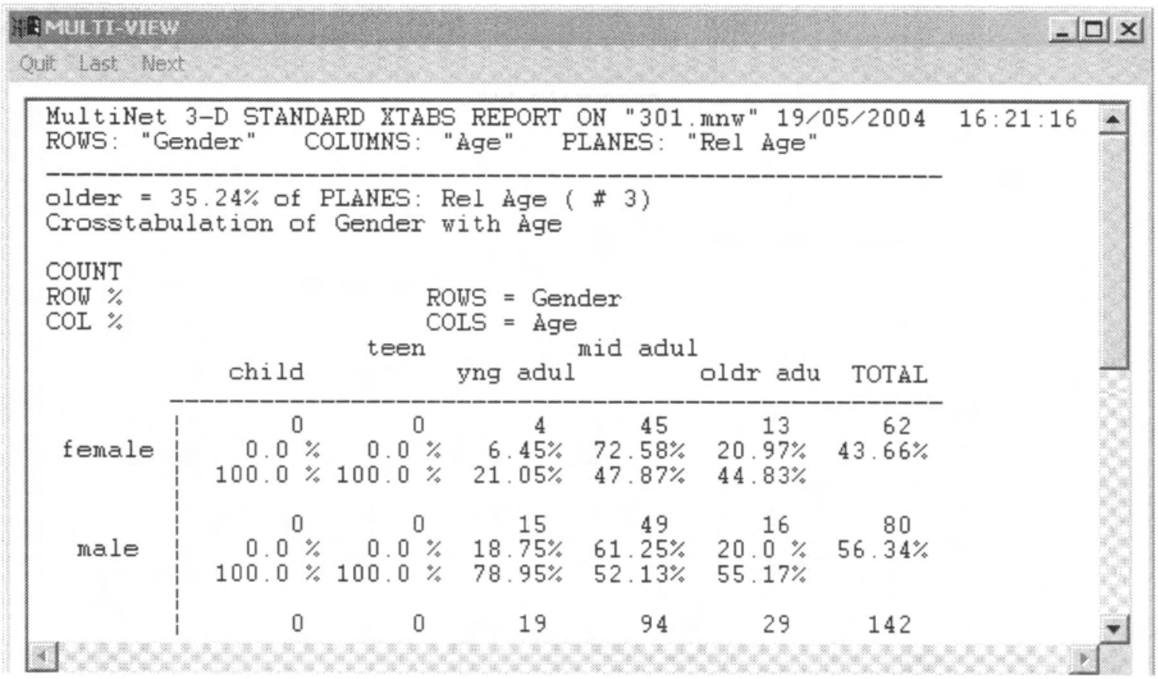
We can step through the set of all panigrams for the categories of the third variable by clicking **Next** and **Last**. The results for 'older', the third value of 'Rel Age', are shown in Figure 2.8. Note that Figure 2.8b is a Multi-View window for the text Report, so it also has menu items **Next** and **Last**.. For both visual display and textual report there is a plane for an analysis that includes all values of the third variable, as well as planes for each category of this variable (in this case, one plane for each value of "Rel Age" = "younger", "same", "older"). Each plane is automatically labelled (with a standard colour) above the line with a description of the plane being shown, and how much the plane contributes to the total. In Figure 2.8a, "Rel Age" = "older" (coloured dark green for the 3rd category) contributes 35.24% of the total. The Reports are similarly labelled. For "Rel Age"= "older", there are only 3 non-empty columns, since there are= no "child" and "teen" counts for "Age". This means that the degrees of freedom for this plane is 2, rather than 4 for all the data as in Table 2.2b. Indeed, it can happen that a plane has no counts at all, and this is displayed as 'NO DATA ON THIS PLANE' for both the panigram and Report (and for any other type of stacked analysis as well).



older = 35.24% of PLANES: Rel Age (# 3)

78.9% of 'yng adult' are 'male'

Figure 2.8a. Panigram for third category of 'Rel Age', obtained by clicking **Last** once, or **Next** twice.



```
MultiNet 3-D STANDARD XTABS REPORT ON "301.mnw" 19/05/2004  16:21:16
ROWS: "Gender"   COLUMNS: "Age"   PLANES: "Rel Age"
-----------------------------------------------------------
older = 35.24% of PLANES: Rel Age ( # 3)
Crosstabulation of Gender with Age

COUNT
ROW %                        ROWS = Gender
COL %                        COLS = Age
                    teen             mid adul
          child          yng adul          oldr adu  TOTAL
         --------------------------------------------------
         |     0        0       4      45       13      62
  female |   0.0 %    0.0 %   6.45%  72.58%  20.97%  43.66%
         | 100.0 %  100.0 %  21.05%  47.87%  44.83%
         |
         |     0        0      15      49       16      80
   male  |   0.0 %    0.0 %  18.75%  61.25%  20.0 %  56.34%
         | 100.0 %  100.0 %  78.95%  52.13%  55.17%
         |
         |     0        0      19      94       29     142
```

Figure 2.8b. Partial **XTAB** results for **Stack**ed variable 'Rel Age'. Report for third category 'older' obtained by pressing **Last** once or **Next** twice.

Even with the lower degrees of freedom for the cross-tabulations of 'Gender' and 'Age' when 'Rel Age' is restricted to only one of its three values, chi-square is not significantly large enough to reject the null hypothesis of no relationship between these two variables. However, this does not imply no relationship between any other pairs of the three variables. The menu item **Rotate** is used..to exchange variables so that these other pairs may be analysed.

Rotate is available for <u>any</u> stacked analysis, and clicking it will result in the stacked variable becoming the first variable, the first variable becoming the second, and the second variable becoming stacked. When all three variables are being used in an **XTABS** analysis, the rotated display will remain a panigram and the reports will remain crosstab and related tables. For other types of analysis, rotating causes the displays and reports to change, as will be described later.

In this case the results are shown in Figure 2.9 and Table 2.3. It is immediately obvious from Figure 2.9 that, in contrast to the "Gender" vs 'Age' results, 'Gender' and 'Rel Age' are <u>not</u> independent of each other. The panigram indicates that fewer 'male' are 'younger' and more are 'older' than for 'female'. The chi-square calculation confirms the visualization: chi-square is 9.676 and for 2 degrees of freedom this is significant at $p<0.01$. We can reject the null hypothesis that 'Gender' is unrelated to 'Rel Age'.

We now look at the remaining pair of variables 'Rel Age' and 'Age' by clicking **Rotate** (which makes 'Age' and 'Rel Age' the row and column variables) followed by **Transpose** (which makes 'Rel Age' the independent row variable). The results are shown in Figure 2.10, which can be compared with

Table 2.3. Crosstab table, Chi-squared and Cramer's Phi for Rows: 'Rel Age' and Columns: 'Gender'

```
ROWS:"Rel Age" COLUMNS:"Gender"  PLANES:"Age"
  -----------------------------------------
  100.00% of PLANES: Age ( # 1-5)
Crosstabulation of Rel Age with Gender

COUNT                       ROWS = Rel Age
ROW %                       COLS = Gender
COL %               male
            female           TOTAL
        --------------------------
        |      40       20       60
younger |   66.67%   33.33%   14.89%
        |   18.87%   10.47%
        |
        |     110       91      201
  same  |   54.73%   45.27%   49.88%
        |   51.89%   47.64%
        |
        |      62       80      142
 older  |   43.66%   56.34%   35.24%
        |   29.25%   41.88%
        |
        |     212      191      403
  TOTAL |
        |   52.61%   47.39%
CHI-SQUARE =  9.676   D.F. = 2    P < 0.01    Cramer's Phi= 0.15
```

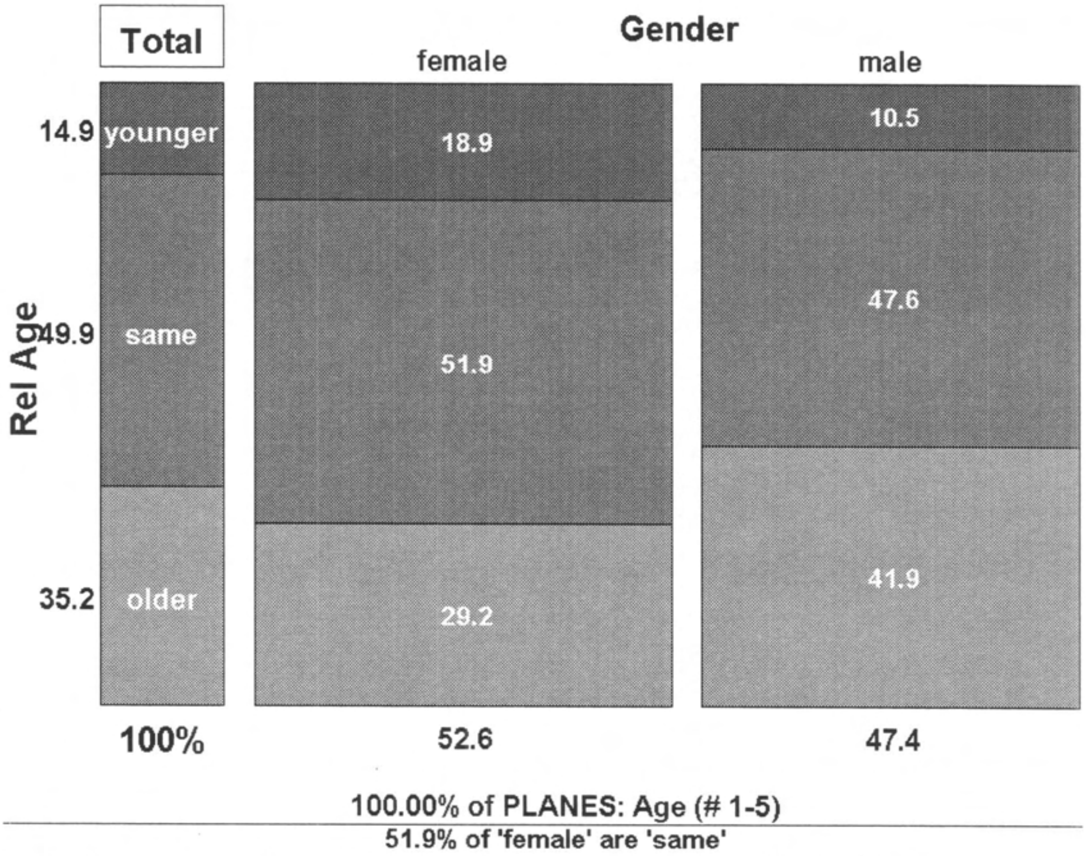


Figure 2.9. Panigram of Rows:'Rel Age' and Cols: 'Gender' after clicking **Rotate**.

Figures 2.5 and 2.9, and Table 2.4, which can be compared with Tables 2.2 and 2.3 The panigram in Figure 2.10 shows a strong relationship between these two variables. Almost all of 'mid adult' and 'oldr adult' are 'older' (than the respondent), while most of 'yng adult' are 'same'. In Table 2.4 we see that even with 8 degrees of freedom Chi-squared is so large that we can easily reject the null hypothesis. Finally, Cramer's phi is over 0.5, the largest value of all for the three pairs of variables, indicating a relationship between these two variables. Notice that **Transpose** has no effect on the tables: 'Age' was **Rotate**d into the first (row) variable, and 'Rel Age' was **Rotate**d into the second (column) variable. Initially, the panigram shows the column percents in the crosstab table. Clicking **Transpose** once changes the panigram to show the row percents instead, with no change to the entries in the table. A second click on **Transpose** switches back to the display of column percents.

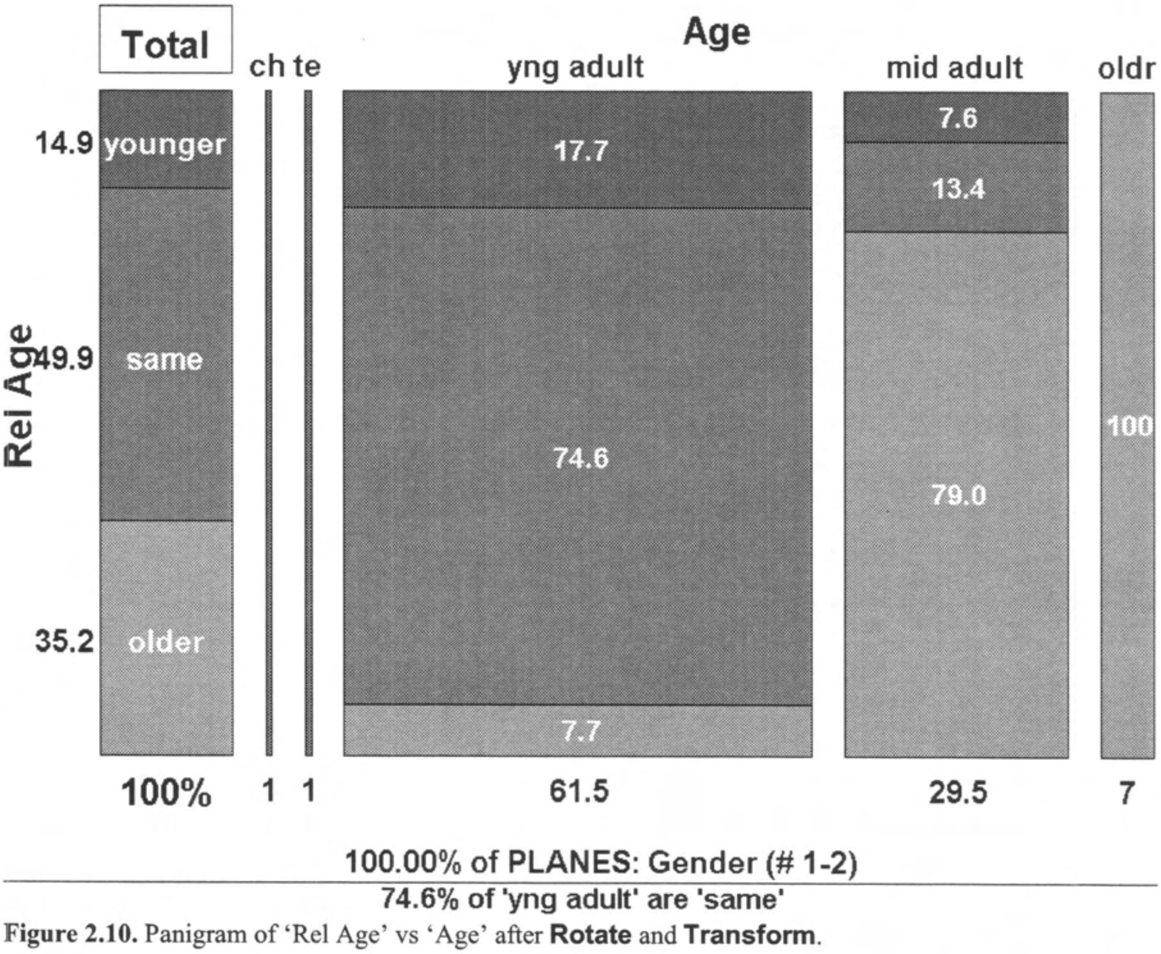


Figure 2.10. Panigram of 'Rel Age' vs 'Age' after **Rotate** and **Transform**.

Up to this point, we have concentrated on the mechanics of getting panigrams and cross-tab and related tables without considering whether these results tell us anything useful about the dataset, or

Table 2.4. Crosstab, Chi-squared and Cramer's Phi of "Age" vs "Rel Age" after **Rotate**

```
  100.00% of PLANES: Gender ( # 1-2)
Crosstabulation of Age with Rel Age
COUNT
ROW %                    ROWS = Age
COL %                    COLS = Rel Age
                     same
          younger           older    TOTAL
         ------------------------------------
        |    3       0       0       3
 child  | 100.0 %  0.0 %   0.0 %   0.74%    100% of PLANES: Gender
        |   5.0 %  0.0 %   0.0 %            CHI-SQUARE = 279.2 D.F. = 8
        |
        |    4       0       0       4             P < 0.01
  teen  | 100.0 %  0.0 %   0.0 %   0.99%
        |   6.67%  0.0 %   0.0 %                Cramer's Phi= 0.59
        |
        |   44     185      19     248
yng adul|  17.74%  74.6 %   7.66%  61.54%      ---------------------
        |  73.33%  92.04%  13.38%                female = 52.61% of Gender
        |                                         Cramer's Phi= 0.58
        |    9      16      94     119
mid adul|   7.56%  13.45%  78.99%  29.53%       ----------------------
        |  15.0 %   7.96%  66.2 %                 male = 47.39%
        |                                         Cramer's Phi= 0.62
        |    0       0      29      29
oldr adu|   0.0 %   0.0 % 100.0 %   7.2 %
        |   0.0 %   0.0 %  20.42%
        |
        |   60     201     142     403
 TOTAL  |  14.89%  49.88%  35.24%
```

even whether cross-tabs are appropriate for these variables. To do this, we need to know more about what these variables are measuring.

The 301.MNW dataset was collected by 12 students in a project they did for a course taught by Richards (CMNS 301), for which they recorded details about <u>every</u> interaction they had with another person over a one-week period. Node variables include information about the students and about the people they reported interactions with. In the case of 'Gender', this was easy to do (and there is missing data in only 17 out of a total of 435 people). In the case of 'Age', it was also easy to estimate to which of the five categories they or their respondents belonged to (data missing for 10 people). However, 'Rel Age' (relative age – younger, about the same, older) is different, since it records information comparing the ages of the students to those of the people they interacted with, so it has no values for the students themselves. This variable has data missing for 21 cases.

Because this analysis is stacked on a third variable (the gender of the students), only the people for which there are values for <u>all three</u> of these variables appear in the analysis, so the total number of people is 403 for all of the tables presented here. Because there is no value for 'Rel Age' for the twelve students in the course, the students are not included in the tables. The data in Figure 2.9 and

Table 2.4 tell us that 74.6% of the people who the students described as 'yng adults' were also described as 'Rel Age' = 'same' (as the students). 78.99% of those described as 'mid adult' and 100% of those described as 'older adult' were also described as 'older' (than the students). This makes sense, as does the classification of 'child' and 'teen' as 'younger' (than the students). The inserts in Table 2.4 also show that there is little difference in Cramer's Phi for the age descriptions made by male and female students.

The problem associated with variables that may have a small overlap of non-missing data is not important in this example, but it could be significant if, for example, there was extensive data on personal attributes such as 'Age', but only estimates for the data of nominees, as for 'Rel Age'. In this case, the overlap of data for nominators and nominees might be empty for Standard analysis (but not for Network analysis, as we shall see). While analysis with three variables would seem to be generally more informative and flexible (as in this example), it is possible that only one or two of the possible three pairings will be useful, since there may be very small (or no) overlap between one of the pairs. Knowledge of what the variables are actually measuring and how they could interact should always be part of any analysis.

2.2.2 Analysis of Variance (ANOVA)

Where crosstabs are appropriate when both variables are categorical, if one is categorical and the other is continuous, ANOVA is the method of choice (unless the continuous variable has been made discrete by one of the methods available in the Variables module, although this results in a loss of detail). *Analysis of Variance* or **ANOVA** is used to test the statistical significance of the difference between means. To do this it compares the variance between categories to the differences within categories to determine whether the differences between categories is larger than the differences within the categories. If the between-categories differences are larger than the within-categories differences, there must be a relationship between the categorical variable (which sorts individuals into categories) and the continuous variable that is used to calculate the means. To illustrate this type of analysis, consider three variables, all of which describe nominees: 'Length Years', 'Type of Relation' and 'Rel Age'. The variable 'Length Years' is continuous and numerically describes the length of time the nominator (student) has known the nominee; we have data for 411 nominees. 'Type of Relation' categorically describes the relationship the students have with their nominees (so there are no values for the students, we have data for 408 nominees. To demonstrate the effects of **Rotate** with an **ANOVA** analysis, the example also includes 'Rel Age' with data for 403 nominees. The overlap of the three variables results in 403 nominees being included in the analysis.

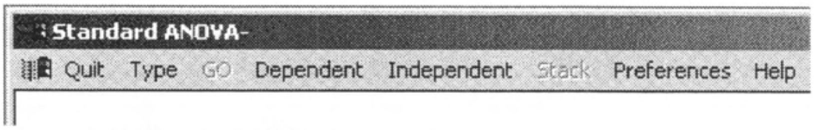


Figure 2.11. Standard **ANOVA** menu choices.

Figure 2.11 shows the menu bar after choosing **ANOVA**. The choices are different from those that follow choice of **XTABS** (and context-sensitive **Help** also provides different information). In particular, rather than **Rows** and **Cols**, the choice of the first two variables are labelled **Dependent** and **Independent**. The Independent variable sorts nominees into categories; the Dependent variable is the numeric/continuous one used to calculate means for each category. We want to determine whether these means depend on the Independent discrete/ categorical variable. For data collected by the students, who reported who they communicated with, the questions are:

- Is the type of relationship a student has with a nominee related to how long the student has known the nominee?
- Is the relative age of the nominee related to how long the student has known the nominee?
- Is the relative age of the nominee related to the type of relationship the student has with the nominee?

Is very likely that the students would describe a 'relative' as someone they have known a long time, and 'stranger' as people they have known a short time, but let us see what the data tell us.

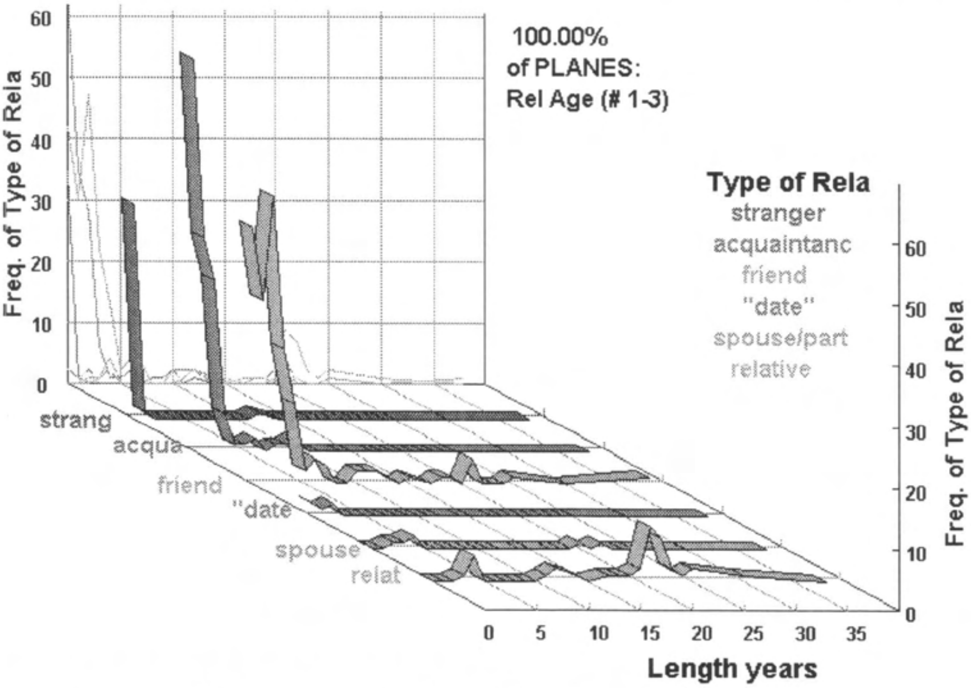


Figure 2.12. Standard **ANOVA** display for Node variables **Dependent**:'Length years' and **Independent**:'Type of Relation', **Stack**ed on 'Rel Age'

Table 2.5a. Report on ANOVA analysis of 'Length years' and 'Type of Relation'

```
MultiNet 3-D STANDARD ANOVA REPORT ON "301.mnw" 21/05/2004  13:17:56
ROWS: "Type of Relation"   COLUMNS: "Length years"   PLANES: "Rel Age"

  100.00% of PLANES: Rel Age ( # 1-3)
Independent: "Type of Relation"       Dependent: "Length years"
SUMMARY OF "Type of Relation"
   LABEL    |  SIZE     |  MEAN     |CONF. INT.|
------------|-----------|-----------|----------|
 stranger   |   37      |     0.35  |    1.17  |
 acquaintan |  141      |     1.18  |    0.6   |
 friend     |  178      |     3.16  |    0.53  |
 "date"     |    5      |     1.0   |    3.17  |
 spouse/par |    6      |     9.17  |    2.89  |
 relative   |   36      |    18.19  |    1.18  |

 ANOVA TABLE
 ---------------------------------------------------------------
 SOURCE OF | Sum of   | Deg. of   | Variance | Obtained |  P
 VARIATION |Squares SS| Freedom   | Estimate |  ratio   |
 ---------------------------------------------------------------
 BETWEEN   | 9135.68  |      5    | 1827.14  |    98.43 |< 0.01
 GROUPS    |          |           |          |          |
 -----------------------------------------------------------
 WITHIN    | 7369.71  |    397    |    18.56 |
 GROUPS    |          |           |          |
 ---------------------------------------------
 TOTAL     | 16505.38 |    402    |
```

Table 2.5b. Report on Statistics and distribution for dependent variable 'Length year'

```
 STATISTICS OF "Length years"            DISTRIBUTION OF "Length years"
                                              VALUES    COUNTS    %age
     MEAN  3.62                                 0         143     35.5%
     VAR   40.96                                1          67     16.6%
     SDEV  6.4                                  2          78     19.4%
     MIN   0                                    3          30      7.4%
     MAX   38                                   4          20      5.0%
     MED   1                                    5           3      0.7%
     SIZE  403                                  6           6      1.5%
     BINS  26                                   7           3      0.7%
                                                8           1      0.2%
                                                :           :       :
                                               25           3      0.7%
                                               30           1      0.2%
                                               38           1      0.2%
```

Figure 2.12 uses a set of 'ribbon' plots to show the number of people who have 'Length years' values from 0 to 38 in each of the six categories of 'Type of Relation', and each category is labelled and coloured in the standard way. In this case, the counts are exact, since the number of values for 'Length years' is less than the MultiNet default of 30 for '**Number of Bins**'. For a variable with a large number of unique values ('bins'), the program will automatically put the variable into a number of bins close to the default (or the number chosen in the Preference module) and the display will then show counts for the number in each bin. Although the *binned display* can look quite different for different '**Number of Bins**', this does not affect the **ANOVA** calculations. It is clear from Figure 2.12 that the curves for 'Type of Relation' = 'stranger', 'acquaintance' and 'friend' are different from those for 'date','spouse' and 'relation'. The former all have high counts at low values. Table 2.5 provides much more detail about each distribution, as well as a summary of the distribution and statistics of the dependent variable 'Length years'. The details about each distribution are used in the Analysis of Variance calculations, and follow the standards for such tables. The first table shows the number, mean values and confidence intervals of 'Length years' for each category. These give a rough idea of how different these distributions are, and confirm the initial and sensible guess that 'strangers' are known for shorter times than 'relatives'. The **ANOVA** table compares the variance within each category (group) to the variance between categories to calculate a statistic ('Obtained ratio') which can be used to determine whether the differences between groups are statistically significant. The entry 'p<0.01" means that, when the groups come from the same population, the probability of getting differences between groups as large as the ones we see is less than 0.01, so we can reject the null hypothesis: the means are indeed different; there is a relationship between 'Length years' and 'Type of Relation'.

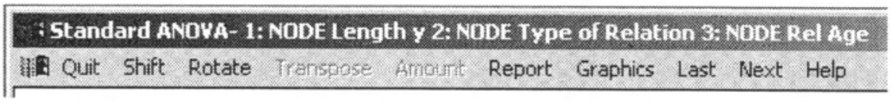

Standard ANOVA- 1: NODE Length y 2: NODE Type of Relation 3: NODE Rel Age
Quit
Shift
Rotate
Transpose
Amount
Report
Graphics
Last
Next
Help

Figure 2.13. Menu bar for Standard **ANOVA**

The menu bar for this display (Figure 2.13) is similar to that for **XTABS**, except that **Transpose** is disabled (greyed out) since the action does not make sense for this display. **Shift** is enabled, and allows interactive selection of the aspect that this display should present. Figure 2.14 shows the result after pressing the other '**X**', '**Y**' and '**Z**' buttons a few times (defaults **Home1** and **Home2** are shown in Figures 2.12 and 2.15). This view can be accepted as the default (and the setting then saved in the Preferences module) by clicking **Okay**. Pressing **Cancel** restores the current default display. **Shift** is provided since not all **ANOVA** displays are easy to understand in the default **Home1** or **Home2** aspects.

Next and **Last** act the same as they did for **XTABS**, displaying the relationship between 'Length years' and 'Type of Relation' as we step through and restrict 'Rel Age' successively to each of its values both for the visual display and the textual report (in a Multi-View window). Categories are labelled and coloured as they were for **XTABS**. **Rotate** acts for **ANOVA** almost the same as for **XTABS**. Clicking it will switch 'Type of Relation' to the Stacked variable, and make 'Rel Age' the new independent variable (Figure 2.15), to give another **ANOVA** analysis. This time the aspect defaults to **Home2**. The relationship for this pair of variables is not as strong (Table 2.6).

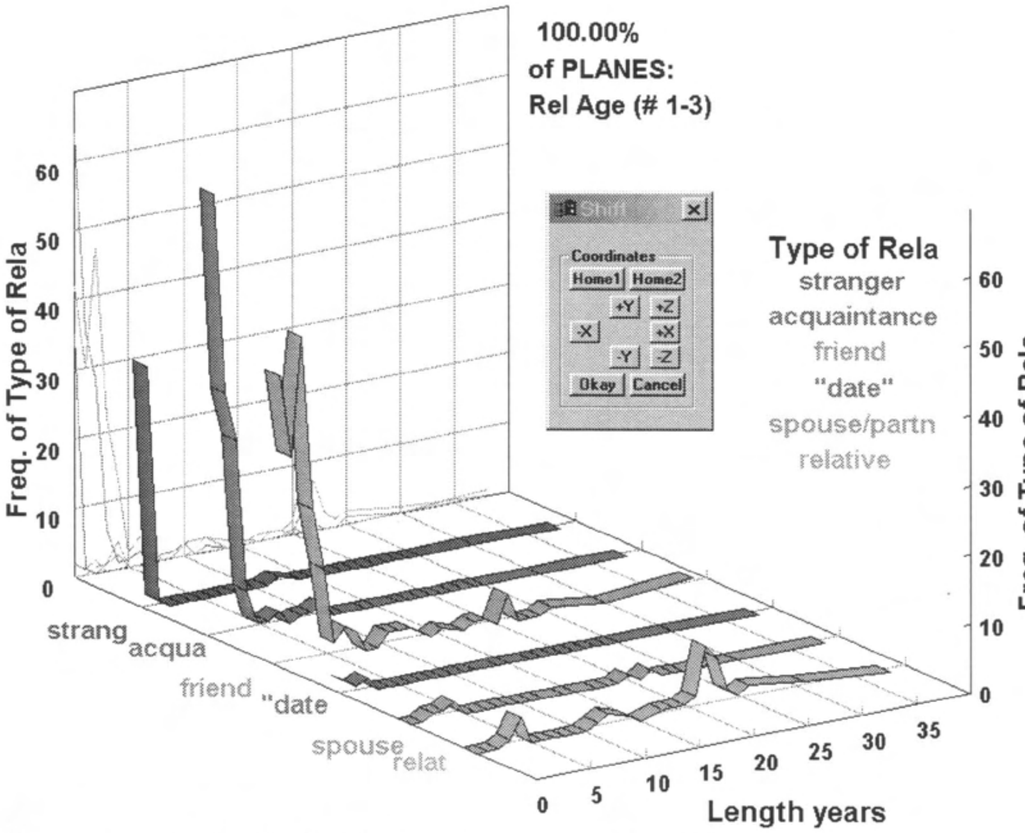


Figure 2.14. Using the **Shift** window to change aspect of the **ANOVA** display

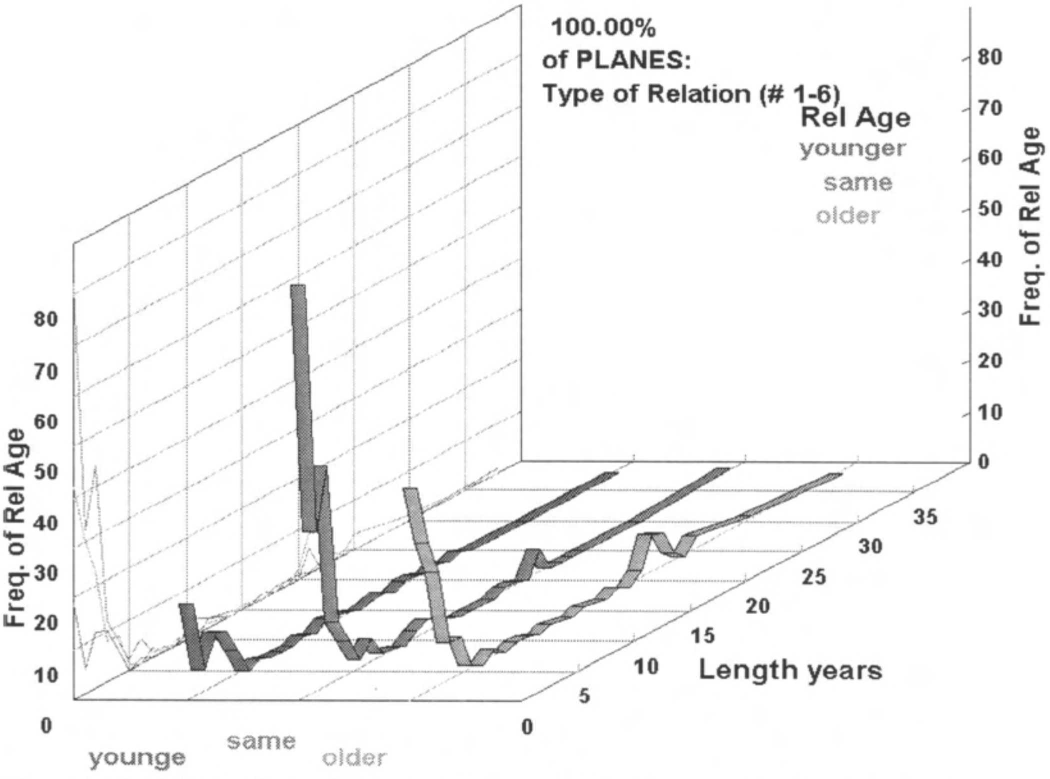


Figure 2.15. ANOVA display for 'Length years' and 'Rel Age' after **Rotate.** Aspect is **Home2**.

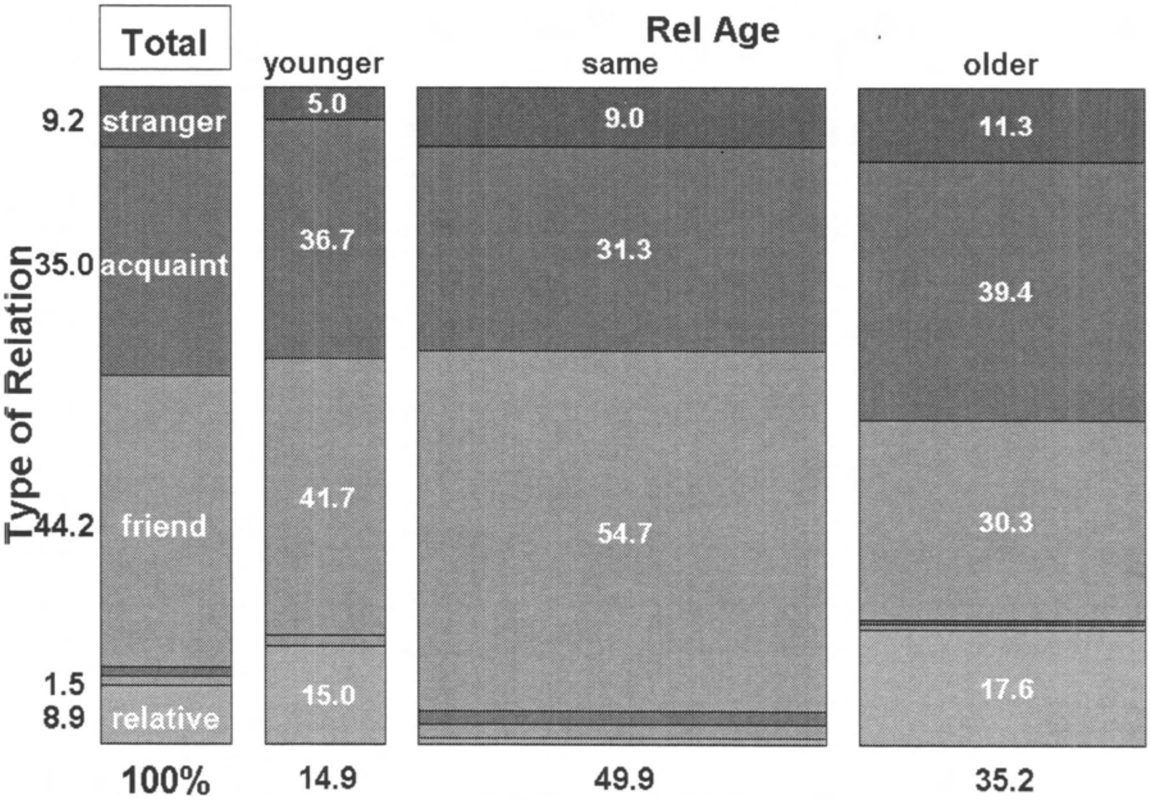
Table 2.6. **ANOVA** analysis for Dependent: 'Length years' and Independent: 'Rel Age'

```
   100.00% of PLANES: Type of Relation ( # 1-6)
Independent: "Rel Age"    Dependent: "Length years"
 SUMMARY OF "Rel Age"
  LABEL    |  SIZE     |  MEAN     |CONF. INT.|
----------|----------|----------|----------|
younger   |   60     |     3.88 |     1.34 |
same      |   201    |     2.47 |     0.73 |
older     |   142    |     5.13 |     0.87 |

ANOVA TABLE
---------------------------------------------------------------
SOURCE OF | Sum of   | Deg. of  | Variance | Obtained |  P
VARIATION |Squares SS| Freedom  | Estimate |  ratio   |
---------------------------------------------------------------
BETWEEN   |   593.44 |        2 |   296.72 |     7.46 |< 0.01
GROUPS    |          |          |          |          |
---------------------------------------------------------------
WITHIN    | 15911.94 |    400   |    39.78 |
GROUPS    |          |          |          |
-----------------------------------------------
TOTAL     | 16505.38 |    402   |
```

Table 2.7. **Rotate** produces crosstab of 'Rel Age' and 'Type of Relation' for all values of 'Length years'

```
 100.00% of PLANES: Length years
 Crosstabulation of Rel Age with Type of Relation
 COUNT
 ROW %                     ROWS = Rel Age
 COL %                     COLS = Type of Relation
                    acquaint          "date"          relative
             stranger          friend        spouse/p           TOTAL
          ------------------------------------------------------------
          |      3      22      25       0       1       9       60
 younger  |   5.0 %  36.67%  41.67%   0.0 %   1.67%  15.0 %  14.89%
          |   8.11%  15.6 %  14.04%   0.0 %  16.67%  25.0 %
          |
          |     18      63     110       4       4       2      201
   same   |   8.96%  31.34%  54.73%   1.99%   1.99%   1.0 %  49.88%
          |  48.65%  44.68%  61.8 %  80.0 %  66.67%   5.56%
          |
          |     16      56      43       1       1      25      142
  older   |  11.27%  39.44%  30.28%   0.7 %   0.7 %  17.61%  35.24%
          |  43.24%  39.72%  24.16%  20.0 %  16.67%  69.44%
          |
          |     37     141     178       5       6      36      403
   TOTAL  |   9.18%  34.99%  44.17%   1.24%   1.49%   8.93%
```

For this **ANOVA** analysis **Shift**, **Next** and **Last** act exactly the same as for the previous **ANOVA** analysis. However, now **Rotate** must do something quite different, since it will exchange the continuous variable 'Length year' for the discrete/categorical variable 'Type of Relation', resulting in a comparison of two categorical variables. Logically, the result should be a cross-tabulation, as shown in Figure 2.16 and Table 2.7. Since this is a crosstab, **Shift** is disabled and **Transpose** is enabled. **Next** and **Last** must also be disabled, since there is no obvious way to 'step through' a continuous variable such as 'Length year'.



100.00% of PLANES: Length years

36.7% of 'younger' are 'acquaintance'

Figure 2.16. Crosstab of 'Rel Age' and 'Type of Relation' for all values of 'Length years'.

The results show that there is a relationship between these two variables. For example, the majority of people the same age are 'friends', while 'relatives' appear as either 'younger' or 'older'. In this case, the result is different from that obtained by applying **XTABS** to these two variables by themselves since the overlap with third variable 'Length years' forced the loss of a small number (4) of data pairs.

Both **XTABS** and **ANOVA** include at least one categorical variable in the visualizations, with the values of the variable mapped to the set of colours used by MultiNet in a standard way (first = dark blue, second = dark red, etc.). For stacked displays, the third variable is also mapped to these colours in the same way, and these colours show up in comments and titles which show the percent each value contributes to the total. In Figure 2.8a, 'Rel Age' = 'older' is the third category, mapped to dark green, which shows in the comment: 'older = 34.35% of Rel Age'. The next type of analysis does not include a categorical variable, so colours mapped to the stacked variable are available for both the comments and the displayed graphics.

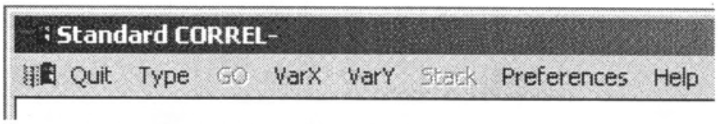
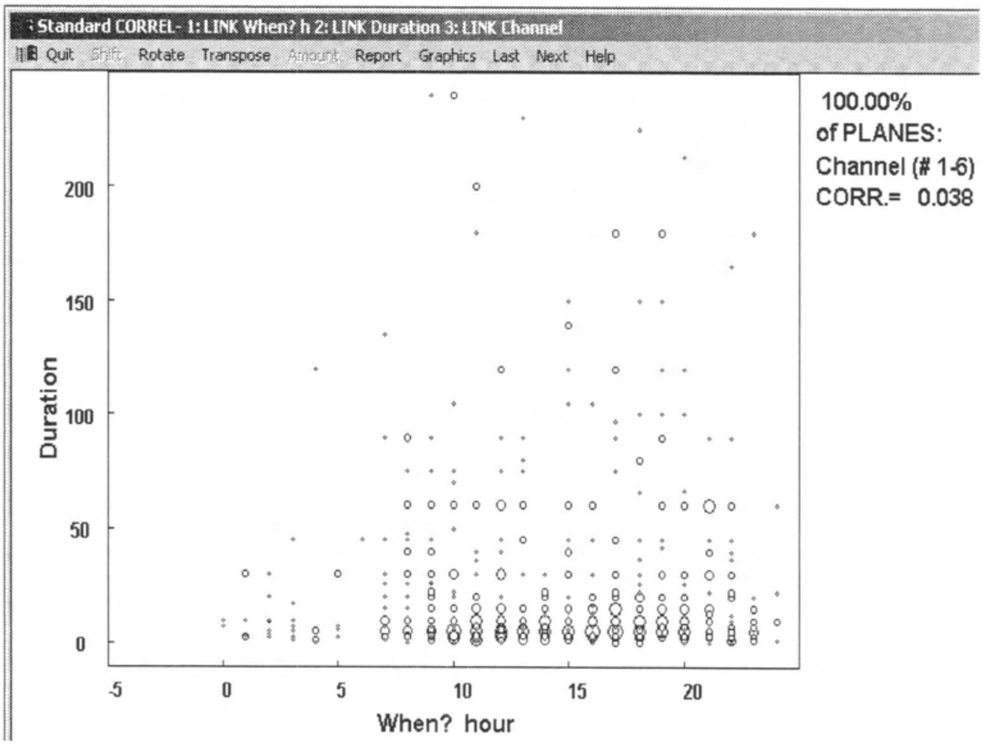


Figure 2.17. Initial **CORREL** menu items

2.2.3 Correlations and scatterplots (CORREL)

A common method for measuring any relationship between pair of numeric/continuous variables is *correlation,* and a common method for visualizing the data is the *scatterplot*. MultiNet provides both of these by selecting the **Analyse→Type→CORREL**. The menu bar changes to show choices for 'VarX' and 'VarY' (Figure 2.17). These names represent the axes assigned to the two variables, but do not otherwise distinguish them (Table 2.8). Once both have been selected, the usual **Stack** choice for a third discrete/categorical variable is enabled. To illustrate **CORREL**, we will select a pair of Link variables: 'When? Hour' describes the hour when an interaction took place; 'Duration' describes how long in minutes the interaction was. To further illustrate the effects of stacking, the third categorical variable is 'Channel', which describes the communication channel over which the interaction took place. Figure 2.18 shows the visual display over all channels as a scatterplot, while Table 2.8 shows the accompanying textual report. Once again, the Next and Last items are available both from the menu bar of the display and the Multi-View window. For this display the **Shift** item is disabled, while **Transpose** is enabled and switches the X and Y axes. It should be no surprise that **Rotate** produces a pair of **ANOVA** displays and reports: one for 'Channel' and 'Duration', the other for 'Channel' and 'When? Hour'.



100.00%
of PLANES:
Channel (# 1-6)
CORR.= 0.038

Figure 2.18. Visual display for stacked **CORREL**.

Table 2.8. Report for Standard **CORREL** analysis of 'When? Hour' and 'Duration,' **Stack**ed on 'Channel'. **VarX** and **VarY** are treated the same way: neither is assumed to be dependent or independent. Each is regressed against the other to give linear fits, with regression coefficient r and coefficient of determination r^2.

```
MultiNet 3-D STANDARD CORRELATION REPORT ON "301.mnw" 22/05/2004  16:23:21
 ROWS: "When?  hour"    COLUMNS: "Duration"    PLANES: "Channel"
100.00% of PLANES: Channel ( # 1-6)
 CORRELATION OF "When?  hour" WITH "Duration":   0.038
 REGRESSION ANALYSIS:
 "Duration"        =      17.351 +      0.271 * "When?  hour"
 STD. ERR.         =       3.799        0.250
 ERROR VARIANCE    =    1280.088     COEFF. OF DETERMINATION =      0.001
 "When?  hour"     =      14.266 +      0.005 * "Duration"
 STD. ERR.         =       0.200        0.005
 ERROR VARIANCE    =      24.585     COEFF. OF DETERMINATION =      0.001

  STATISTICS OF "When?  hour"           STATISTICS OF "Duration"

    MEAN  14.38                           MEAN  21.24
    VAR   24.56                           VAR   1279
    SDEV  4.956                           SDEV  35.76
    MIN   0                               MIN   0
    MAX   24                              MAX   240
    MED   14                              MED   7
    SIZE  835                             SIZE  835
    BINS  25                              BINS  49

  DISTRIBUTION OF "When?  hour"         DISTRIBUTION OF "Duration"
      VALUES      COUNTS      %age          VALUES      COUNTS      %age
        0              2      0.2%            0            431     51.6%
        1              6      0.7%           10            171     20.5%
        2              7      0.8%           20             51      6.1%
        3              7      0.8%           30             49      5.9%
        4              7      0.8%           40             32      3.8%
        5              5      0.6%           50              1      0.1%
        6              1      0.1%           60             39      4.7%
        7             21      2.5%           70              7      0.8%
        8             32      3.8%           80              3      0.4%
        9             44      5.3%           90             15      1.8%
       10             63      7.5%          100              6      0.7%
       11             63      7.5%          110              0      0.0%
       12             69      8.3%          120              8      1.0%
       13             46      5.5%          130              1      0.1%
       14             53      6.3%          140              2      0.2%
       15             42      5.0%          150              3      0.4%
       16             48      5.7%          160              1      0.1%
       17             62      7.4%          170              0      0.0%
       18             56      6.7%          180              7      0.8%
       19             54      6.5%          190              0      0.0%
       20             46      5.5%          200              2      0.2%
       21             44      5.3%          210              1      0.1%
       22             32      3.8%          220              1      0.1%
       23             20      2.4%          230              1      0.1%
       24              5      0.6%          240              3      0.4%
```

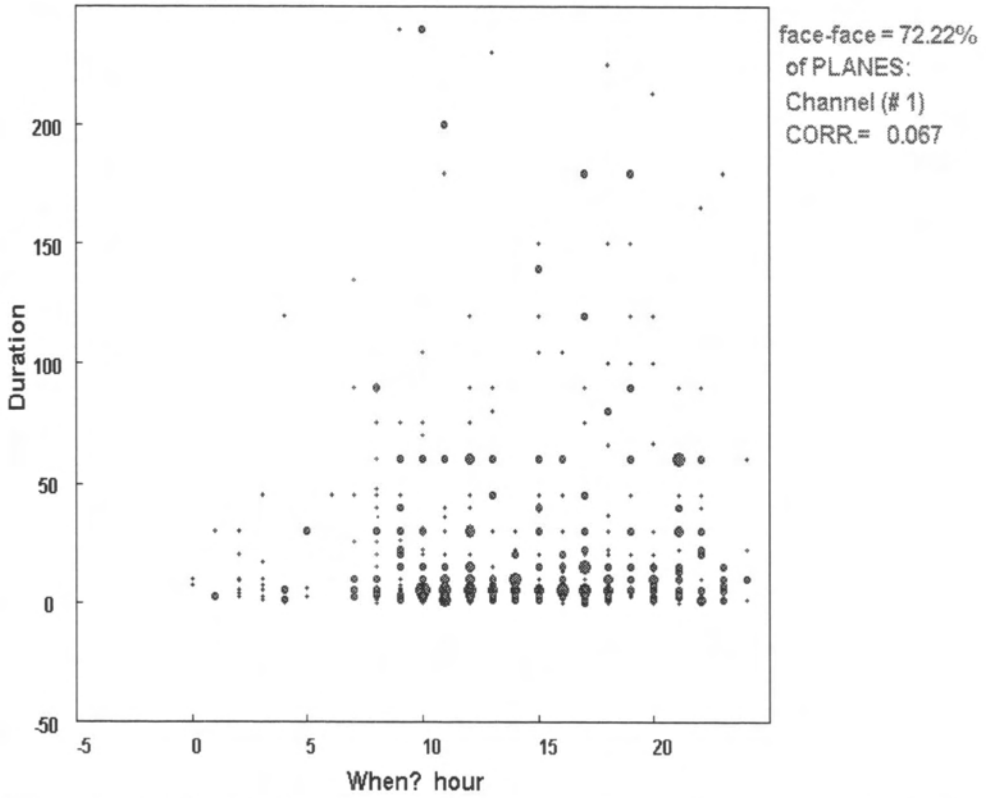


Figure 2.19 a. **CORREL** display 'When? Hour' and 'Duration' after clicking**Next** once to movefrom all values of **Stack**ed variable 'Channel' to first (#1) category = 'face-face' only.

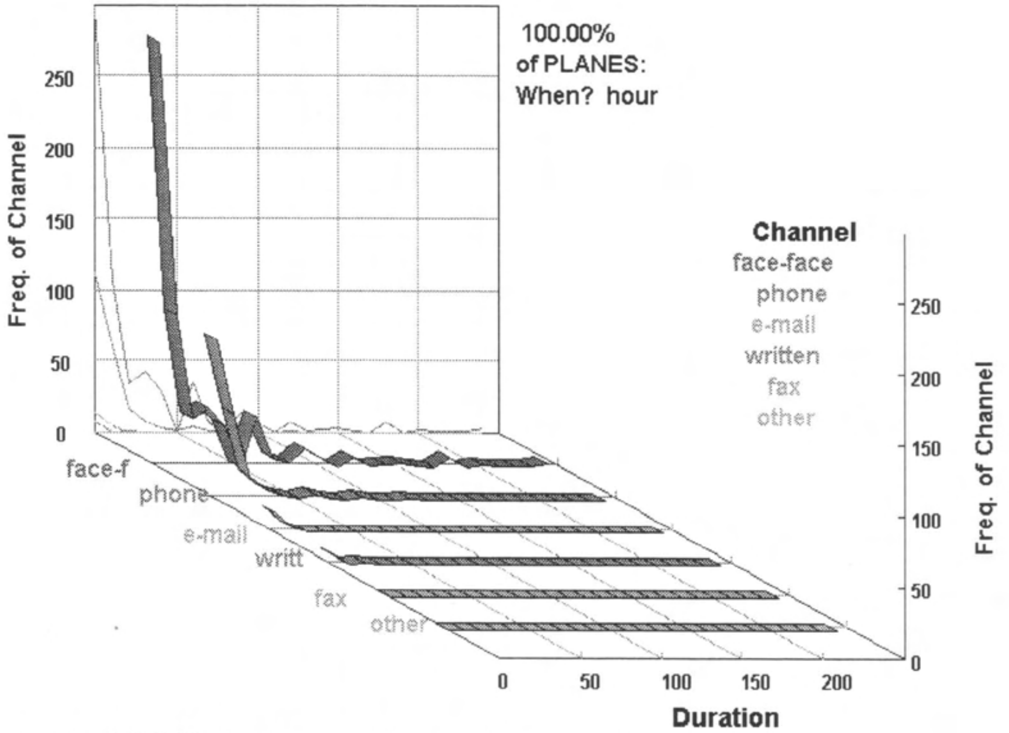


Figure 2.19 b. **ANOVA** display of all 'Channel' categories and 'Duration'

Table 2.9. CORREL statistics for 'Duration' with a) 'Channel'='face-face', b) 'Channel' = 'phone' c) **ANOVA** summary of means after **Rotate** with dependent: 'Duration' and independent: 'Channel

.**a)** 'face-face'

```
STATISTICS OF
 "Duration"
 MEAN   25.06
 VAR    1626
 SDEV   40.32
 MIN    0
 MAX    240
 MED    10
 SIZE   603
 BINS   45
```

b) 'phone'

```
STATISTICS OF
"Duration"
 MEAN   12.04
 VAR    262.4
 SDEV   16.2
 MIN    1
 MAX    120
 MED    7
 SIZE   207
 BINS   24
```

c) ANOVA report for 'Duration' and 'Channel'

```
SUMMARY OF "Channel"
LABEL      |SIZE    | MEAN     |CON.INT
----------|-------|---------|-------
face-face |  603   |  25.06 |  2.36
phone     |  207   |  12.04 |  4.03
e-mail    |   17   |   5.24 | 14.06
written   |    8   |   5.38 | 20.5
fax       |        |        |
other     |        |        |
```

Figure 2.18 does not visually show a strong relationship between the length of an interaction and the time it took place, and the correlation of 0.038 (CORR=0.038) confirms this. The display does show, both by the density of points and the larger circles at the low values, that 'Duration' is skewed towards smaller values, The areas of the circles are proportional to the number of data points that the circles represent. There are 24 distinct values for 'When? Hour' and 49 values for 'Duration'. This means that the values of 'Duration' must be collected into about 30 'bins', adding to the nuber of data points and making the circles larger for shorter durations. The actual bins and counts are shown in the report, which includes the statistics and distribution for <u>both</u> continuous variables.

The correlation does not change much as we step through the 'Channels' of interaction (Figure 2.19) using **Next** (or **Last**). For example, 'Channel' = 'face-face' shows that interaction takes place at all hours, mostly for short durations (Figure 2.19a), and also for longer times than for 'phone' (Figure 2.23). However the same result is easier to see after **Rotate** produces the **ANOVA** display of Figure 2.19b and report of Table 2.9c. The **CORREL** and **ANOVA** displays suggest a higher mean for 'face-face' and this is confirmed in the reports (Table 2.9)

2.3 Analyse display menu choices

Although the menus used to get to a finished display in the Analyse module may differ slightly, the final Analyse display menu bar always contains the same items. Some (e.g., **Shift**, **Transpose**, **Amount**) may be greyed out if not applicable to the current analysis and display. Others (**Rotate**, **Next**, **Last**) are only used with three variables. This section gives a brief description of each menu item (Figures 2.5, 2.13, 2.17, 2.20).

Quit

Exit from the Analyse Module back to the Main MultiNet menu.

Shift

Enabled for ANOVA displays. Allows interactively changing display aspect.

Rotate

Enabled for three variables. **Rotate** replaces the second variable with the **Stack**ed one, the **Stack**ed variable with the first one, and the first variable with the second one. If the analysis type is **ANOVA**, exchange first and second to make continuous first (dependent). If the **Stack**ed variable is discrete, the program enables **Next** and **Last** for displays and reports.

Transpose

Enabled for **XTABS** and **CORREL**. **Transpose** exchanges first and second variables in display, but has no effect on reports.

Amount/Number

Enabled for pure Network **XTABS** (i.e., not for stacked **ANOVA**) . If the link variable has non-binary values, **Amount** shows cumulative values of links, and changes this menu item to **Number**. **Number** (the default) shows count of links and changes this menu item to **Amount**.

Report

This menu item is common to most MultiNet modules, and acts similarly in each one. Left-click on **Report** produce the following choices:

- .**Report→View** opens a View (or Multi-View) window containing the textual report for the latest analysis. The contents of this report depend on the analysis just performed. If there is a discrete **Stack**ed variable, the window will be Multi-View, with **Next** and **Last** available for stepping through the report for each value of the Stacked variable.
- **Report→File** saves the **Report** as an ASCII text file, with default extension .OUT without displaying it.

Graphics

Like Report, this menu item is common to most MultiNet modules, and acts similarly in each one. Left-click on Graphics produces the following choices:

- **Graphics→PostScript** reproduces the current display as a PostScript program. This is a text file which produces graphics with a PostScript interpreter (e.g., printer).
- **Graphics→Bitmap** captures the screen display as a 256-colour bitmap, which is then run-length encoded. The result is a compressed Windows .BMP file.

For more details, see Section 0: Overview and Technical appendix 0.

Next

This is another menu item that appears in a number of MultiNet modules. In the Analyse module it is enabled for three variables if the **Stack**ed variable is discrete. Clicking on **Next** steps through the display for each value of the Stack-ed variable, from first to last.

Last

This is another menu item that appears in a number of MultiNet modules. In the Analyse module it is enabled for three variables if the **Stack**ed variable is discrete. Clicking on **Last** steps through the display for each value of the **Stack**ed variable, from last to first.

Help

Every MultiNet module includes a **Help** button which provides a selection window for choosing information about the other items on the menu bar. The **Analyse** displays include this as the first of two choices: 'Text' and 'Interactive' (Figure 2.20). The choice of 'Text' produces the usual selection window for the display and the menu bar items. **Interactive Help** is similar to the '**Explore**' windows in other modules since it allows the user to step through all values of any display while receiving detailed information about each part being examined. Figure 2.21 shows **Interactive Help** for Panigrams (**XTABS**), Figure 2.22 for **ANOVA** displays, and Figure 2.23 for **CORREL** displays. **Interactive Help** has two main purposes:

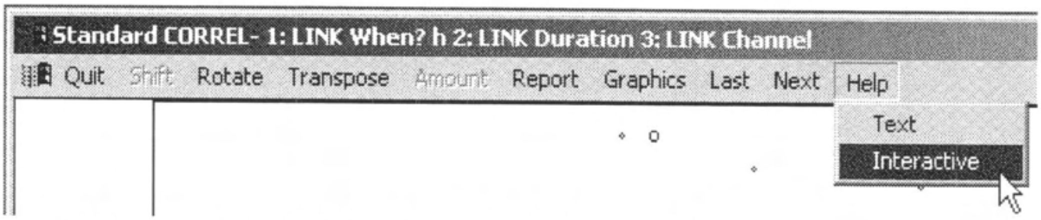
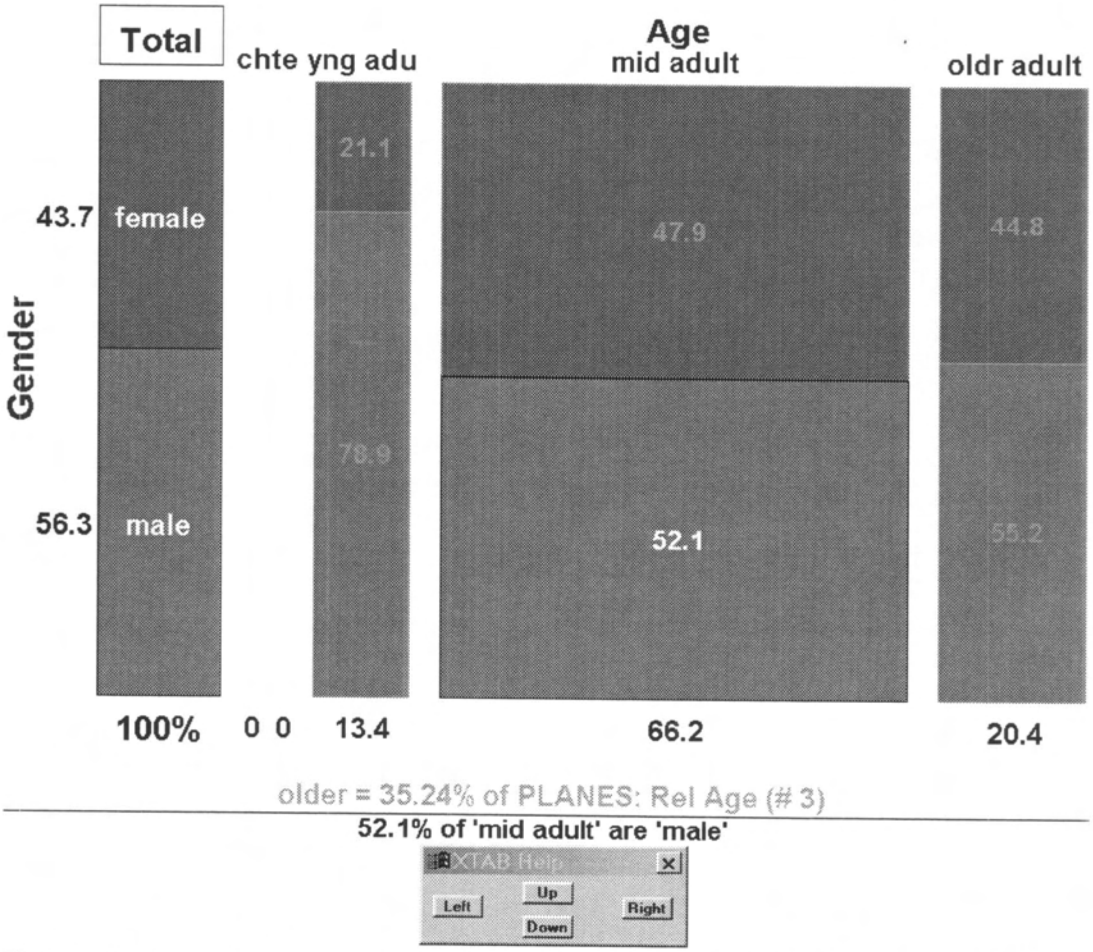


Figure 2.20. Selecting Interactive **Help** for any Analyse display window

- To aid in understanding the display
- To provide details about the display not otherwise available

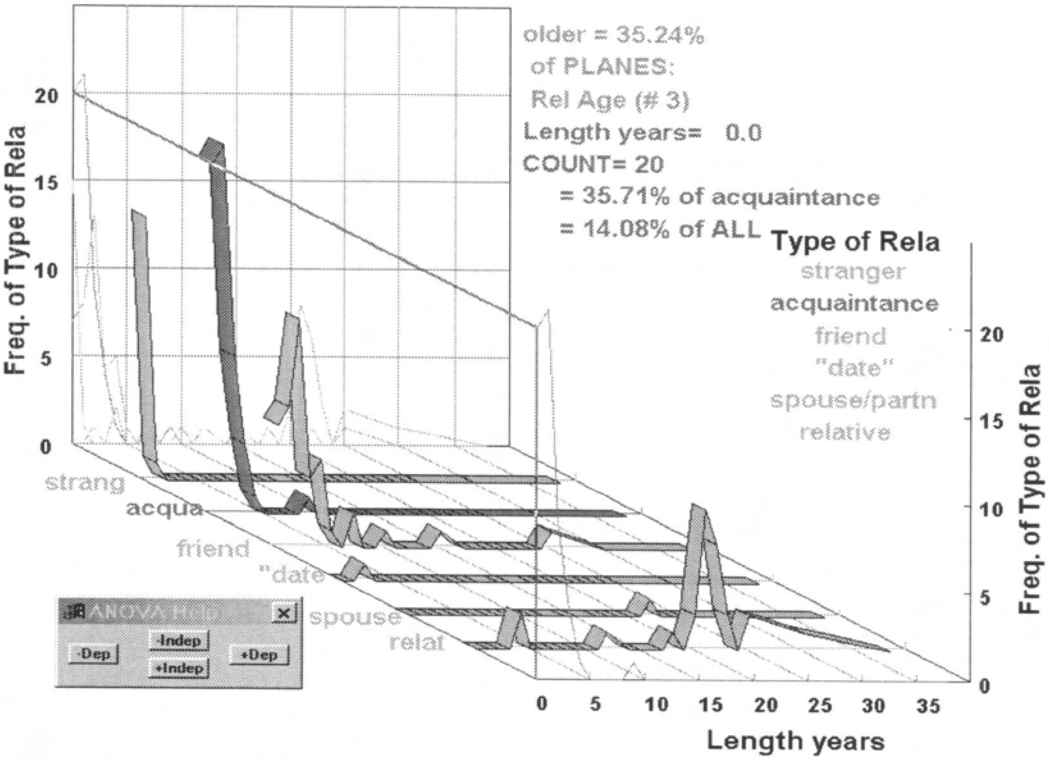


older = 35.24% of PLANES: Rel Age (# 3)

52.1% of 'mid adult' are 'male'

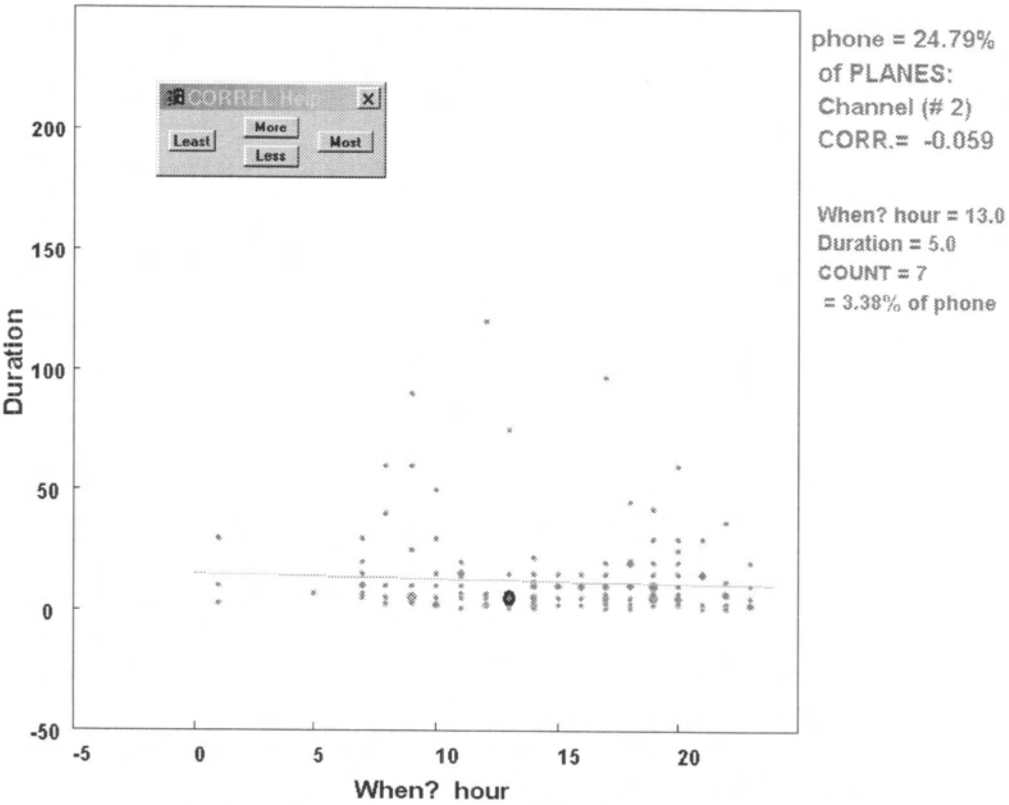
Figure 2.21. Interactive help for panigrams uses **XTAB Help** window to move focus up, down, left and right. As a cell is selected, the outline and percent are high-lighted, and an explanatory description is printed below the line. This also shows the percent when the cell is too small for it.

For example, **XTAB Help** (Figure 2.21) provides not just percents whether they are visible or not, it also provides explanatory detail that helps to interpret what each individual cell represents; this is particularly helpful for Network crosstabs, which can have meanings quite distinct from the relatively simple Standard crosstabs. **ANOVA Help** (Figure 2.22) high-lights each category of the discrete variable by greying out all the others, and by showing the corresponding (binned) distribution curve of the continuous variable at both front and back of the display; it allows stepping through each category of the discrete variable as well as the bins of the continuous variable giving count, percent within category, and overall percent for each point selected. **CORREL Help** (Figure 2.23) high-lights each circle chosen, giving counts, percents, and values for both of the continuous variables. Circles may be selected by size, or by clicking the mouse pointer anywhere in the display. The regression line is also shown in grey.



older = 35.24%
of PLANES:
Rel Age (# 3)
Length years= 0.0
COUNT= 20
= 35.71% of acquaintance
= 14.08% of ALL

Figure 2.22. ANOVA Help window allows stepping through values of the discrete and continuous variables. As values are selected, the count and percents of the variable are shown.



phone = 24.79%
of PLANES:
Channel (# 2)
CORR.= -0.059

When? hour = 13.0
Duration = 5.0
COUNT = 7
= 3.38% of phone

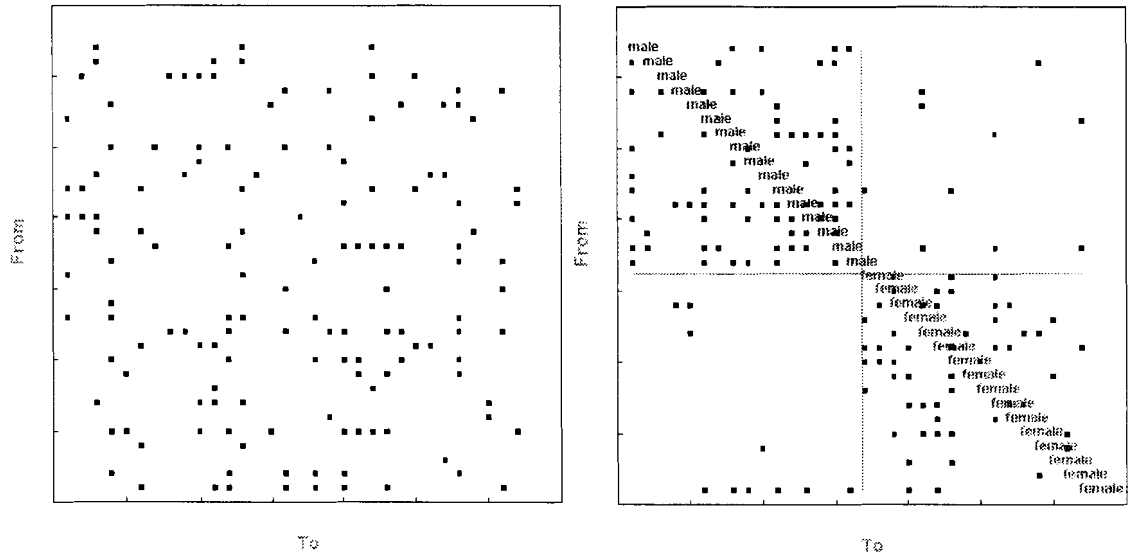
Figure 2.23. CORREL Help allows circles to be chosen by size or with a mouse click. Details include value, count and percent. The regression line is also shown in grey.

2.4 Network analysis

The previous sections introduced the methods, displays and reports available in the **Analyse** module with examples for standard crosstabs, **ANOVA**, and correlations. The same methods can be applied to networks by letting the variables represent (among other things) the attributes at each end of a link. This application of standard methods to networks is unique to MultiNet. A simple and obvious example comes from the KIDS2 dataset, using the **1-D** displays discussed in the **Eigenspaces** module. Figure 2.24 shows the adjacency matrix of the SAY link (who each kid says he or she plays with). In Figure 2.24a, the rows and columns are ordered according to the ID numbers of the nodes, and it is difficult to see any pattern. In Figure 2.24b, the rows and columns have been permuted by the values of the node variable 'SEX' and the values (1='male' and 2='female') have been shown along the diagonal by clicking **Dots→Values**. The relationship between this node variable and link variable is obvious and made even more clear by also selecting **Show→Grid**. Even though the relation is obvious, it would be helpful to have a measure of the statistical significance of this relationship between the two types of variables. The easiest way to do this would be to simply count the number of links in each of the grid boxes of Figure 2.24b, and consider the result as a crosstabulation of two variables based on SEX:

- the value of SEX for whom the link is **FROM** (the y-axis in Figure 2.24b) and
- the value of SEX for whom the link is **TO** (the x-axis in Figure 2.24b)

This cross-tabulation would then allow a chi-squared significance test and other measures as we have seen. It would also allow a calculation of the mean and variance of link values within each grid box



a) Adjacency matrix for link variable SAY

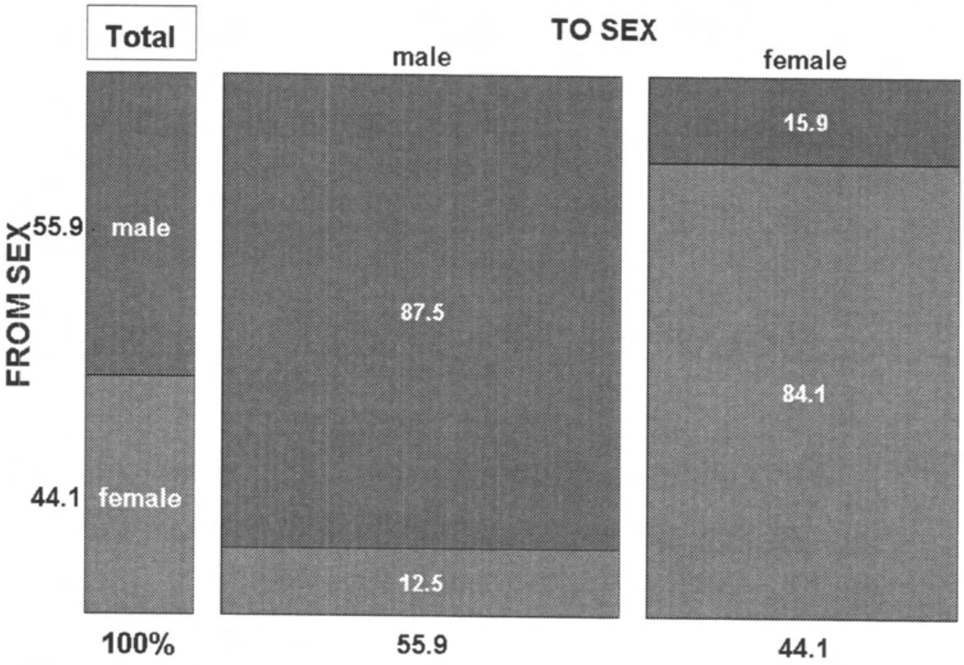
b) SAY permuted by Node variable SEX (See Section 5.6 for details)

Figure 2.24. Eigenspaces 1-D displays of SAY

(although in this case all links are binary so these are not useful). These calculations are exactly what is done by a network crosstab (Figure 2.25 and Table 2.10).

Table 2.10. Network crosstab table for Node SEX and Link SAY.
Chi-squared shows there is a significant relationship between the SEX of sender and receiver.

```
Crosstabulation of FROM SEX with TO SEX
 LINK=SAY
 COUNT                    ROWS = FROM SEX
 ROW %                    COLS = TO SEX
 COL %                    female
         male                    TOTAL
         ---------------------------
        |      70      10       80
   male |   87.5 %  12.5 %   55.94%
        |   87.5 %  15.87%
        |
        |      10      53       63
 female |   15.87%  84.13%   44.06%
        |   12.5 %  84.13%
        |
        |      80      63      143
  TOTAL |
        |   55.94%  44.06%
CHI-SQUARE =  73.365   DF = 1   P < 0.01 Cramer's Phi= 0.72
```



12.5% of the links to members of 'male' come from members of 'female'

Figure 2.25. Panigram of **FROM** SAX and **TO** SEX for link variable SAY.
The crosstab counts are the number of links in each grid box of Figure 2.24b.

When **Analyse→Network** is selected from the main menu, the resulting menus and displays are similar, but have some additional choices. For example, when **XTABS** is selected, the **Rows** and **Cols** menu items allow selection of **FROM** and **TO** Node variables (Figure 2.26), as well Links and Groupings if any are defined (Groupings will be discussed in the Groupings Module section). When selecting a **FROM** (**TO**) node variable, we are really selecting the values of the node variable at the sending (receiving) end of the links defined by a variable which must also be selected to perform the analysis.

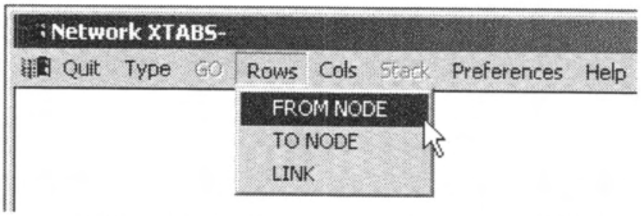


Figure 2.26. Row variable selection for Network **XTABS**. No Groupings are defined, so 'Grouping' is not a choice.

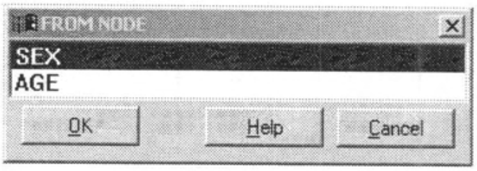


Figure 2.27. FROM NODE selection window provides a list of Node variables.

Since we are selecting the two ends of a network, Network XTABS differs from Standard XTABS since choosing a **FROM** or **TO** node variable does not prevent us from also selecting a link variable (or a **Grouping** of them), which allows mixing of Node and Link variables. After a variable type for **Rows** or **Cols** is chosen a selection window opens, allowing the actual variable to be selected (Figure 2.27). Either **FROM** and **TO** Node windows provide a list of Node variables. **Link** or **Groupings** windows provide a list of **Link** or **Groupings** variables.

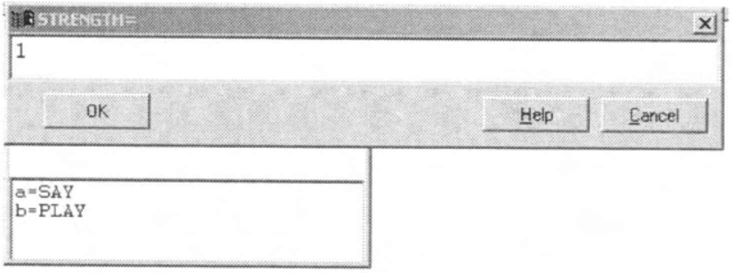
Once the variables to be tabulated have been chosen, the '**GO**' menu item is clicked, which then provides a further choice of **Link** variable to define the network (Figure 2.28). In fact, two choices are provided:

- select a single Link variable from a selection window of all Link variables
- enter an Equation which may combine a number of Link variables (including none)

If a single link variable is chosen, a final Edit window allows the user to enter a descriptive label for the resulting display (useful for keeping track of analyses). If a **Stack** variable has also been chosen, descriptive labels are created automatically.

If **Equation** is chosen, and Edit window opens, along with a display window showing single-letter codes to be used for the Link variables. This is very similar to the windows used for **Recode→Equation** in the Variables module. Here the Edit window has a title bar labelled 'STRENGTH ='. For historical reasons the values of the links in a network defined by an equation are called the link 'strengths' (another valid term is link 'weights'). The values returned by the Strength equation are more restricted than they are for the **Recode→Equation** values, since they are not allowed to be negative (this would cause problems for the accumulation of strengths). As Figure 2.28 shows, a constant value of 1 is valid: this simply chooses all pairs of IDs, and is useful when a link variable is already one of the variables to be tabulated: in this case only the ID pairs for which the chosen link variable has a non-zero value will be selected for analysis. For more complex equations, only those ID pairs that have a non-zero and non-missing value as the result of the strength equation are counted. As an example, since both SAY and PLAY are binary, entering **'a*b'** as the equation in Figure 2.28 results in the logical 'AND' of the two link variables. This selects all ID pairs which have non-zero values for both SAY and PLAY, resulting in only the SAY values that are confirmed by PLAY observations. Once a Strength equation has been accepted, the program continues as in the case of a single link variable. Also, in either case the user may choose to select only a range of link values. This setting is described in the **Preferences** Module section.

Once the link variable or equation has been selected, and the description entered for non-**Stack**ed analyses, the resulting display looks and behaves very much like the Standard Analysis display, although the interpretations are very different. For example, Figure 2.25 does not show that 12.5% of 'males' are 'females'. It shows that 12.5% of links '**TO** males' are '**FROM** females', that is, counting the number of males that receive links from males (70) and from females (10) gives us 10/(10+70) = 1/8 = .125 = 12.5% of all links to males. This is spelled out in the helpful description underneath the panigram. The unit of analysis in Figure 2.25 and Table 2.10 is the link SAY (child *I* SAYs it plays with child *j*), and the variables in the XTAB analysis are the attributes (in this case SEX) of the people at the ends of the link. The TOTAL count in the table is the number of links (SAY) in the network, not the number of nodes (children). The helpful descriptions are helpful, especially with interactive Help and with all the permutations allowed by **Transpose** and **Rotate**.



STRENGTH=

1

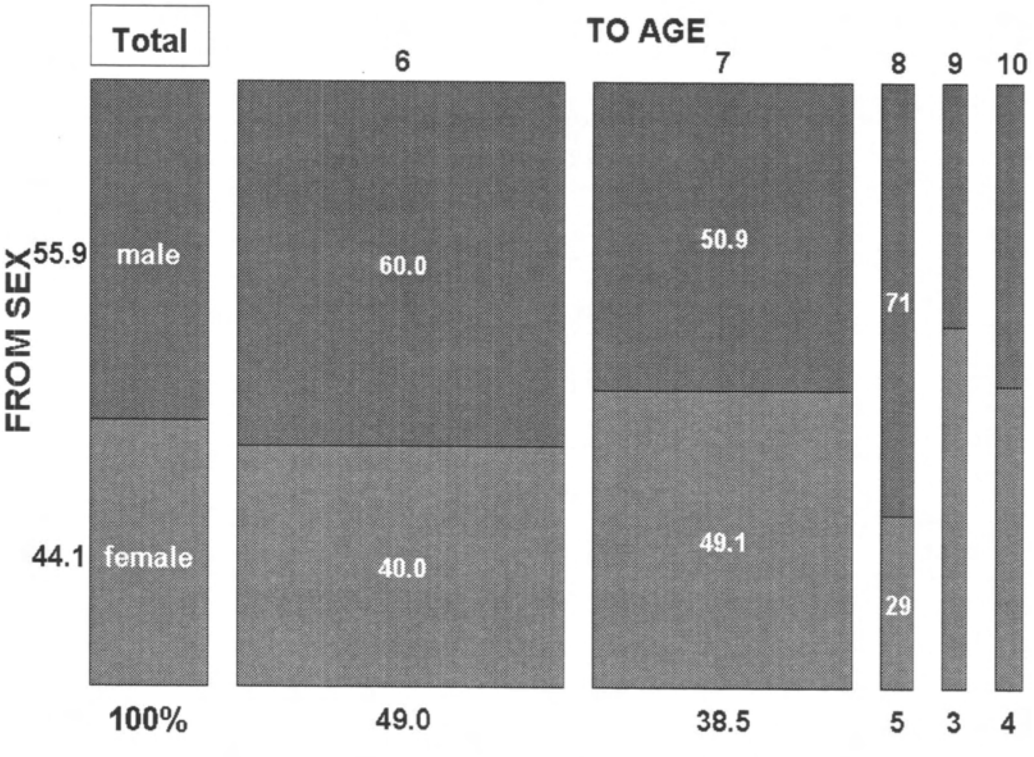
a=SAY
b=PLAY

Figure 2.28. Edit window for entering Strength equation

After **Transpose**, the descriptive line reads "**12.5% of the links from members of 'male' go to members of 'female'**". Although the numbers are the same, the interpretation is different. This happens even though the network has not been symmetrized, since many children say they play with someone who does not agree. The crosstab table turns out to be symmetric when the Row and Column are the SEX of the children (but not the children's AGE).

Table 2.11a) Crosstabulation report of **FROM** SEX and **TO** AGE shows little significance

```
Crosstabulation of FROM SEX with TO AGE
LINK=SAY
COUNT                       ROWS = FROM SEX
ROW %                       COLS = TO AGE
COL %                7                 9
              6                 8                 10      TOTAL
          ---------------------------------------------------------
          |      42       28        5        2        3        80
   male   |   52.5 %   35.0 %    6.25%    2.5 %    3.75%   55.94%
          |   60.0 %   50.91%   71.43%   40.0 %   50.0 %
          |
          |      28       27        2        3        3        63
  female  |   44.44%   42.86%    3.17%    4.76%    4.76%   44.06%
          |   40.0 %   49.09%   28.57%   60.0 %   50.0 %
          |
          |      70       55        7        5        6       143
   TOTAL  |
          |   48.95%   38.46%    4.9 %    3.5 %    4.2 %

CHI-SQUARE =   2.316      DF = 4        P > 0.50    Cramer's Phi= 0.13
```



40.0% of the links to members of '6' come from members of 'female'

Figure 2.29a. Age (and hence **TO** Age) treated as discrete (there are only 5 distinct values).

Table 2.11b) **ANOVA** report of Dependent:**TO** AGE and Independent: **FROM** SEX shows little significance.

```
LINK=SAY
   100.00% of PLANES: TO AGE ( # 1-5)
 Independent: "FROM SEX"                   Dependent: "TO AGE"

 SUMMARY OF "FROM SEX"
   LABEL    |   SIZE    |   MEAN    |CONF. INT.|
 ----------|----------|----------|----------|
 male       |    80     |     6.7   |     0.19 |
 female     |    63     |     6.83  |     0.21 |

 ANOVA TABLE
 -------------------------------------------------------------
 SOURCE OF |  Sum of   | Deg. of   | Variance | Obtained |  P
 VARIATION |Squares SS| Freedom   | Estimate |  ratio   |
 -------------------------------------------------------------
 BETWEEN   |      0.55 |     1     |     0.55 |     0.55 |> 0.10
 GROUPS    |           |           |          |          |
 -------------------------------------------------------------
 WITHIN    |   141.88 |   141     |     1.01 |
 GROUPS    |           |           |          |
 --------------------------------------------
 TOTAL     |   142.43 |   142     |
```

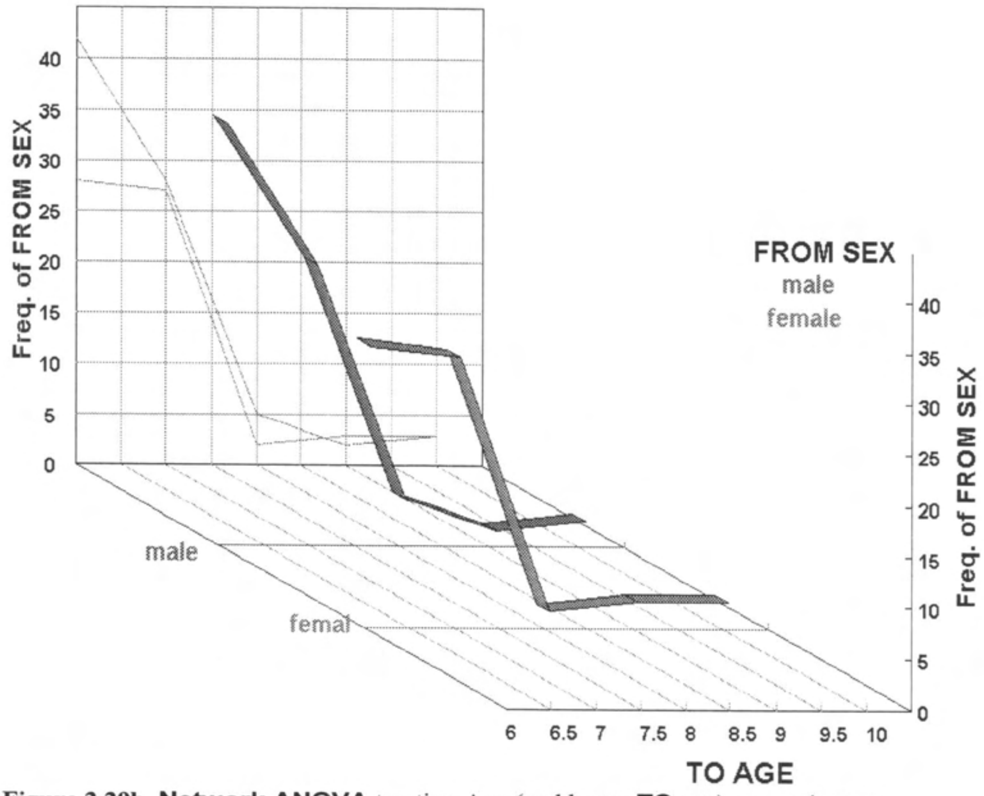


Figure 2.29b. **Network ANOVA** treating Age (and hence **TO** age) as continuous.

The display of panigrams including Link strengths are controlled by the **Amount** menu item. In this case the strength of all links is '1', so this display is not different. The Report includes crosstabs for link strength as well as a table of means and variances. These details are discussed below for examples with non-binary link variables.

The example above is simple because it uses the same Node variable for both **FROM** and **TO**, but there is nothing to prevent them being different. Although the **Eigenspaces 1-D** display does not allow separate permutation of rows and columns by different variables, there is no such restriction in any of the Network analysis types. Tables 2.11 shows the result for '**FROM** SEX' and '**TO** AGE' first as a crosstab, then as an **ANOVA**. AGE can be treated as a numeric/continuous variable, but it has only 5 values, so **ANOVA** is not really appropriate. In fact, Tables 2.11 and Figures 2.29 are the results for **Stack**ed Network **ANOVA** with dependent:**TO** AGE, independent: **FROM** SEX, **Stack**ed on **TO** AGE (again). This shows that MultiNet can be flexible about using variables as either discrete or continuous (although it does issue warnings). This also demonstrates that there is nothing to prevent two or even three variables from being all **FROM** or all **TO** (this would make the interpretations similar to standard methods). The **ANOVA** analysis proceeds in a manner very similar to standard **ANOVA**, except for the extra choice of Link or Equation after pressing '**GO**'. For **ANOVA** and **CORREL**, the actual values of the variable chosen as 'Link' are not currently used, only whether they are non-zero (i.e., they are treated as binary).

The application of these standard methods to networks provides an extremely powerful and flexible set of tools for exploratory analysis of complex network data: that is, network data which consists of many Node attribute variables, and many Link attribute variables each of which can describe a different aspect of a set of related networks. Further, these variables may be either discrete/categorical or numeric/continuous, and Node and Link variables may be considered together. This flexibility allows many combinations, not all of which may make sense for a particular dataset. It is up to the user to consider what each combination actually means, and whether the combinations are useful for understanding the data..

The following is a list of variable types currently allowed for each of the first, second and third variables. The strength equation allows many Link variables to be combined together. In addition, the **Variables**, **Eigenspaces** and **Pstar** modules allow new variables to be created, derived from combinations of node attributes and network structure.

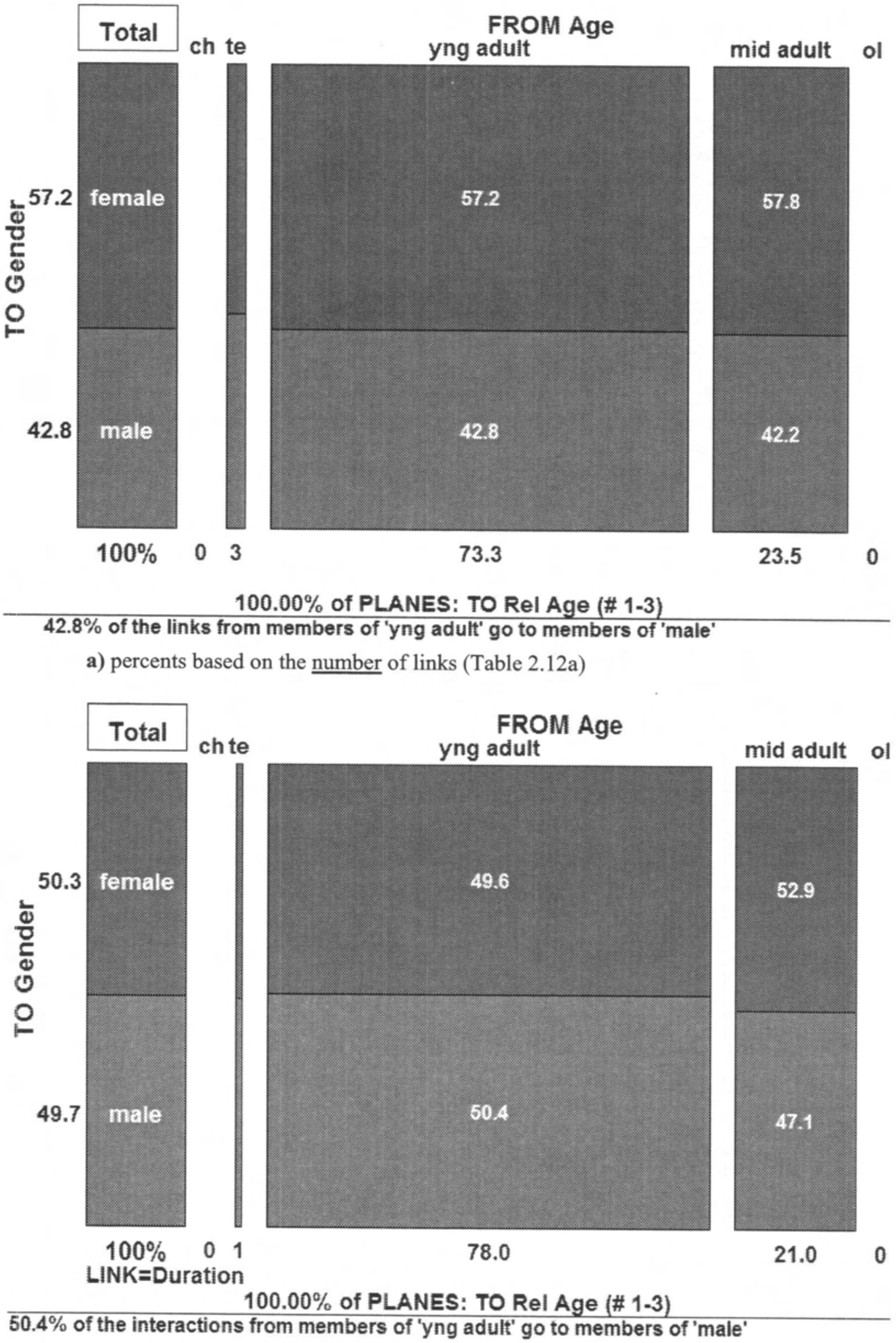
- First variable: **FROM** Node, **TO** Node, Link, Grouping; discrete or continuous
- Second variable: **FROM** Node, **TO** Node, Link, Grouping; discrete or continuous

- Third variable: **FROM** Node, **TO** Node, Link: discrete only
- Equation: any combination of Link variables and constants using **+ - * / ^ < > =**

  May be discrete (including binary) or continuous.

To provide some examples of these combinations, including non-binary link values for the Link variable or Strength equation, we return to the 301.MNW dataset. To further illustrate the difference between standard and networks analysis, we look at the very first example: Gender vs Age. In the network context, we need to consider both ends of any relationship, as well as the type of relationship we are interested in. This dataset is egocentric, with only a few nodes (12) at the sender end. Much of the node variables have values only for the receivers (e.g., 'Rel Age'), while a few have values for both senders or receivers (e.g., 'Age'). Given the kind of link data available, we might ask: 'Do the younger senders ('**FROM** Age') spend more time ('**Link**: Duration') with those of the same age ('**TO** Rel Age')? Of either sex ('**TO** Gender')? This leads to a stacked cross-tabulation with Link variable 'Duration'.

One result is shown in Figure 2.30a and Table 2.12a (compare with Figure 2.5 and Table 2.2a, where the numbers are half as large). The first impression from the panigram is that there is no difference in the number of links to males and females from the different ages. Table 2.12a confirms that any differences are not statistically significant..However, these results are only for the number of relationships and we are interested in the total amount of time spent. This is where the **Amount** menu item is used. Clicking **Amount** presents a different display (Figure 2.30b) with much larger numbers (Table 2.12b) since each contact is weighted by the Duration in minutes. This time the relationship is significant, and the % differences in Table 2.12b shows why: more time is spent by mid-adults talking to females and less time to males than would be expected if there were no relationship between these node attributes. The effect is small, but suggests a question that could be resolved by more data or given more confirmation by other statistical methods: it is an example of hypothesis generation. Note that the tables have rows based on **TO** Gender by selecting this as the **Rows** variable, **FROM** Age as the **Cols** variable and **TO** Rel Age as the **Stack**ed variable. These selections can usually be made in an order that produces the most convenient tabulations.



a) percents based on the number of links (Table 2.12a)

b) percents based on the Duration of links (Table 2.12b)

Figure 2.30. Panigrams showing **FROM** Age **TO** Gender

Table 2.12. Crosstabs for FROM age TO Gender for 100% of Rel Age

a) Crosstab table for the number of links FROM Age TO Gender. Chi-squared is not significant

```
COUNT
 ROW %                         ROWS = TO Gender
 COL %                         COLS = FROM Age
                          teen          mid adul
                 child          yng adul          oldr adu
                                                          TOTAL
           ------------------------------------------------------
           |      0       14      334      108        0      456
   female  |   0.0 %    3.07%   73.25%   23.68%   0.0 %   57.21%
           | 100.0 %   53.85%   57.19%   57.75% 100.0 %
           |
           |      0       12      250       79        0      341
     male  |   0.0 %    3.52%   73.31%   23.17%   0.0 %   42.79%
           | 100.0 %   46.15%   42.81%   42.25% 100.0 %
           |
           |      0       26      584      187        0      797
    TOTAL  |   0.0 %    3.26%   73.27%   23.46%   0.0 %

 CHI-SQUARE =        0.143      D.F. = 2
           P > 0.50             Cramer's Phi= 0.01
```

b) Crosstab table for the amount of contact FROM Age TO Gender. % Difference table shows contact FROM "Age" = "mid-adult" is higher than expected TO females and lower than expected TO males.

```
LINK=Duration
COUNT
ROW %                          ROWS = TO Gender
COL %                          COLS = FROM Age
                          teen          mid adul
                 child          yng adul          oldr adu   TOTAL
           ------------------------------------------------------
           |      0       85     6364     1829        0     8278
   female  |   0.0 %    1.03%   76.88%   22.09%   0.0 %   50.34%
           | 100.0 %   50.9 %   49.63%   52.94% 100.0 %
           |
           |      0       82     6458     1626        0     8166
     male  |   0.0 %    1.0 %   79.08%   19.91%   0.0 %   49.66%
           | 100.0 %   49.1 %   50.37%   47.06% 100.0 %
           |
           |      0      167    12822     3455        0    16444
    TOTAL  |   0.0 %    1.02%   77.97%   21.01%   0.0 %
```

```
% Difference from Expected Values
LINK=Duration                  ROWS = TO Gender
                               COLS = FROM Age
                 teen          mid adul
         child         yng adul         oldr adu
           -----------------------------------
   female  |   0.%    1.%   -1.%    5.%    0.%
           |
     male  |   0.%   -1.%    1.%   -5.%    0.%

CHI-SQUARE =        11.908      D.F. = 2
          P < 0.01              Cramer's Phi= 0.03
```

Since this is a three-variable analysis, we can also step through the values of "Rel Age" to see if the pattern holds for contacts with "younger", "same" and "older" receivers. Table 2.13a shows that most of the signal comes from the links **FROM** "Age"="mid adult" **TO** "Rel Age" = "same" (which accounts for 71.58% of total amount of time interacting), and that this signal is partially offset by "Rel Age"="older" in Table 2.13b (which accounts for 16.95%). So: mid adults spend more time in contact with females of the same age than with males. Does this have anything to do with the sex of the sender? We haven't included this, but it suggests another cross-tabulation: **FROM** Gender **TO** Gender. To further illustrate the flexibility of network crosstabs, this time we will **Stack** on Link variable Channel and ask "Does one sex have more contact with the other? Does this depend on the form of contact?"

Table 2.13. Significant differences from expected for two TO Rel Age planes.

a) % Difference from expected FROM Age TO Gender and TO Rel Age=same

```
same = 71.58% of PLANES: TO Rel Age ( # 2)
% Difference from Expected Values
LINK=Duration              ROWS = TO Gender
                           COLS = FROM Age
                teen           mid adul
         child          yng adul        oldr adu
       ---------------------------------------
female |    0.%   80.%   -6.%   33.%    0.%
       |
 male  |    0.%  -77.%    6.%  -32.%    0.%

CHI-SQUARE =        240.576      D.F. = 2
          P < 0.01                Cramer's Phi= 0.14
```

b) %Difference from expected FROM Age TO Gender and TO Rel Age=older

```
older = 16.95% of PLANES: TO Rel Age ( # 3)
% Difference from Expected Values
LINK=Duration              ROWS = TO Gender
                           COLS = FROM Age
                teen           mid adul
         child          yng adul        oldr adu
       ------------------------------------
 female |    0.%  -19.%   10.%  -28.%    0.%
        |
  male  |    0.%   17.%   -8.%   24.%    0.%

CHI-SQUARE =        63.146      D.F. = 2
         P < 0.01              Cramer's Phi= 0.15
```

Although **FROM** Gender **TO** Gender seems similar to the example with KIDS2, this time the **Eigenspaces 1-D** display is not helpful: the data is egocentric with few senders and uses non-binary link weights for Link Duration. Table 2.14a shows the %difference for number of links and Table 2.14b shows the % difference for the amount of linkage. An obvious difference between these two tables is that within (same sex) and between (different sex) gender difference from expected are switched: the **amount** of contact is significantly higher between sexes, although the **number** of contacts is actually higher than expected within sexes. Obviously, some contacts take longer than others, as summarized in Table 2.14c, which shows the mean and standard deviation of the Duration of contacts. It is 50% higher **FROM** female **TO** male than for the other pairs.

Table 2.14. Differences from expected **FROM** Gender **TO** Gender for 100% of Channels

a) % difference from expected for number of links is not significant

```
% Difference from Expected Values
LINK=Duration              ROWS = FROM Gender
                           COLS = TO Gender
                     male
              female
              ---------------
      female |    3.%    -4.%
             |
        male |   -7.%     9.%

 CHI-SQUARE =    2.063    DF = 1    P > 0.10
 Cramer's Phi= 0.05
```

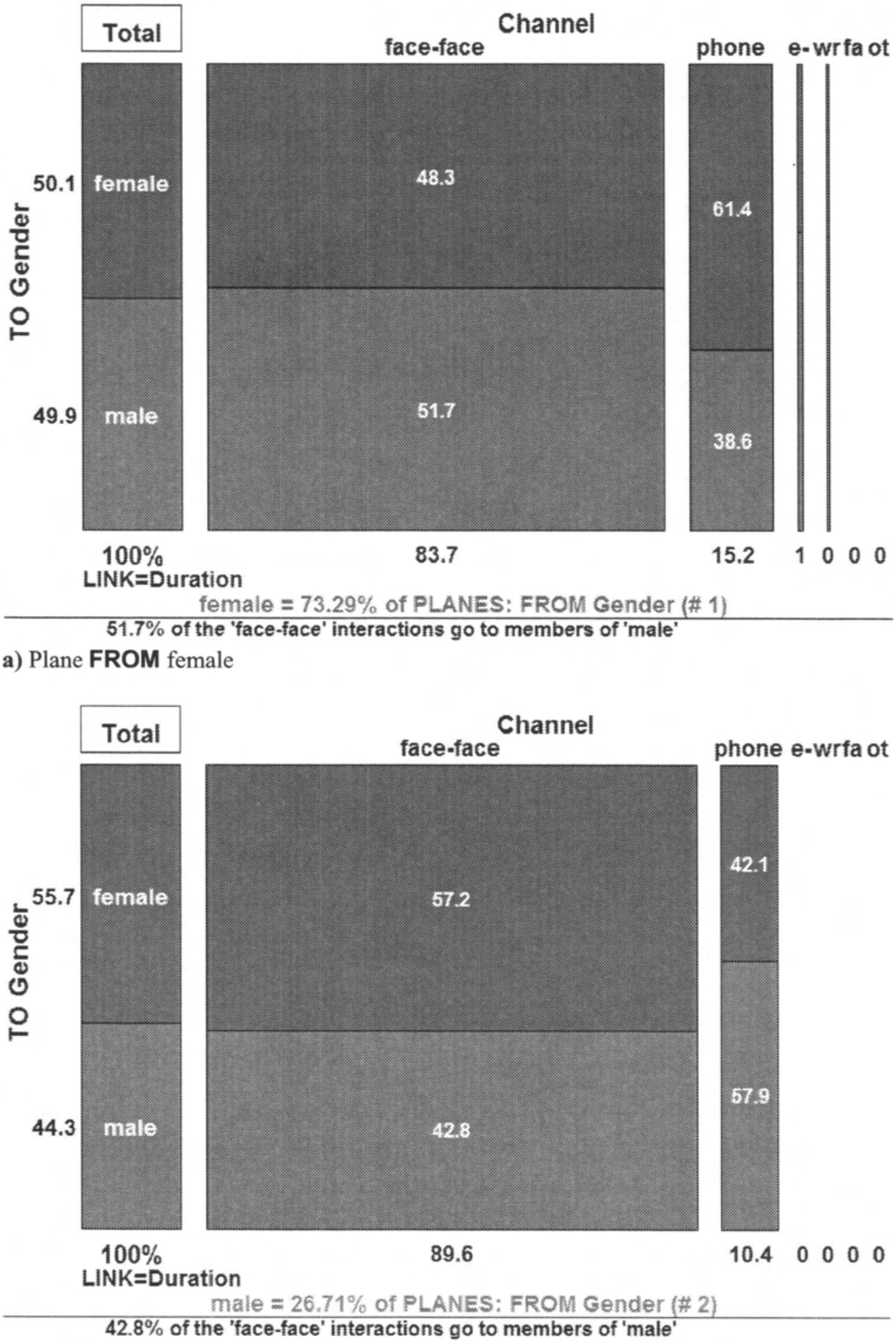
b) % difference from expected for amount of linkage is significant.

```
% Difference from Expected Values
LINK=Duration              ROWS = FROM Gender
                           COLS = TO Gender
                  male
           female
           ---------------
    female |   -3.%     3.%
           |
      male |    8.%    -8.%

   CHI-SQUARE =    40.241    DF = 1    P < 0.01
   Cramer's Phi= 0.05
```

c) Mean and Standard Deviation of Duration. Mean Duration is 50% higher for female to male.

```
Mean strength of links from row I to col j
LINK=Duration
Mean                        ROWS = FROM Gender
Std. Dev.                   COLS = TO Gender
                      male
              female
           ------------------
    female |    18.60   27.03
           |    28.21   45.67
           |
      male |    19.13   17.83
           |    34.21   31.25
```

Does this difference depend on communication Channel? **Rotate** and **Next** show the effect for 'Channel'='face-face', and the opposite effect for 'Channel'='phone' (Figure 2.31a and b), and also show that only females use 'email' and 'written' (and that this data is rather old).

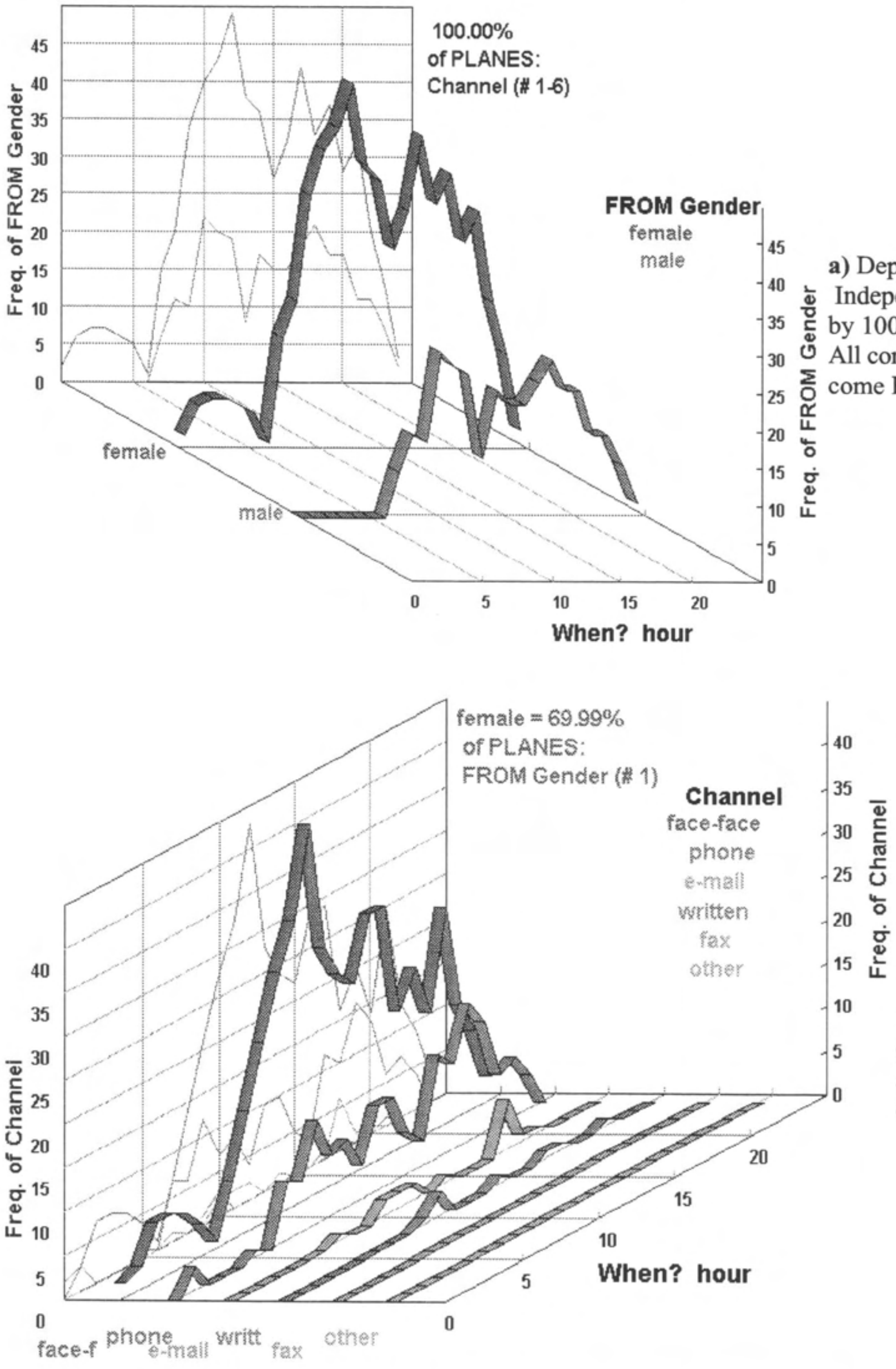


a) Plane **FROM** female

b) Plane **FROM** male

Figure 2.31. TO Gender by Channel shows more between-sex 'face-face'; more within-sex 'phone'

The 301 dataset also has Link variable 'When? Hour' which records the hour of day (0-23) when each contact took place. Using this with network **ANOVA**, we see (Figure 2.32a) that all the contacts from hours 0 (midnight) to 6AM come **FROM** females. Using **Rotate** and **Next**, we can also see that these contacts are 'face-face' (mostly) or 'phone' (Figure 2.32b).



a) Dependent 'When? hour' Independent FROM Gender by 100% of Channels All contacts between 0 and 6 come FROM females.

b) Rotate and **Next** show that contacts **FROM** females at 0 (midnight) to 6AM are 'face-face' or 'phone'.

Figure 2.32. ANOVA displays of **FROM** Gender by Links :'When? Hour'(Dependent) and Channel.

The example for **CORREL** will also be used to demonstrate how the different modules of MultiNet can work together. The questions are:'Is the length of time someone has been known related to the average duration of contact? Are either of these related to the communication channel?'. This suggests **ANOVA** of Dependent:'**TO** Length years' with Independent: Channel and link variable Duration. But **ANOVA** (and **CORREL**) does not use the values of a link variable, only the binary presence/absence. How do we get the average Duration (a link variable) into this analyses?

The Variables module allows hybrid derived variables: node variables derived from link variables (as does Eigenspaces), and link variables derived from node variables (as does Pstar). We will do the former, in two steps. First, **Variables→Link** is used to calculate the average Duration of contacts between distinct pairs of individuals. For this egocentric dataset, there may be a number of links between any pair of nodes, and each may have a different value for Duration. To get the average value, **Recode→Reduce Multilinks** is used to count the number of links between each distinct pair and to accumulate the total Duration for these pairs..The Link variables **#Duration** and **$Duration** are automatically created containing this information, which is exactly what we need to get the average. Since they have just been created, **Recode→Equation** with equation **b/a (= $Duration/#Duration)** will calculate this average. We will use **Create** on this new link variable and name it 'Avg Duration'. Second, **Variables→Node** is used with **Recode→ Degree** and choice **Weighted In-degree** to automatically create '**I<Avg Duration**', a single value associated with each receiver which measures the average amount of time spent in all contact with the sender. Now we are ready for the analysis.

We select **Network→CORREL** with **VarX=TO Node:**'Length Years', **TO** I<Avg Duration' and **Stack=Link:**'Channel'. For Link variable, we select '**Equation=1**", since VarY already selects all the receivers. The stacked scatterplot is shown in Figure 2.33, and the correlation of -0.155 is small: there is little relationship between these two variables. Using **Next** to step through the values of Channel shows that all the correlations are small, except for 'written' = -0.625, but this has only 1% of the data (8 out of 811 points). **Rotate** shows that 'Length years' is related to Channel (p<0.01), mostly since a large number of 'face-face' contacts are to people with small 'Length years' values (Figure 2.34 and Table 2.16). **Rotate** again shows that the relation between Channel and '**TO** I<Avg Duration' is not significant (p>0.10): the different means for 'face-face' and 'phone' are not large enough to overcome the large spread in values shown by the confidence intervals (Table 2.15).

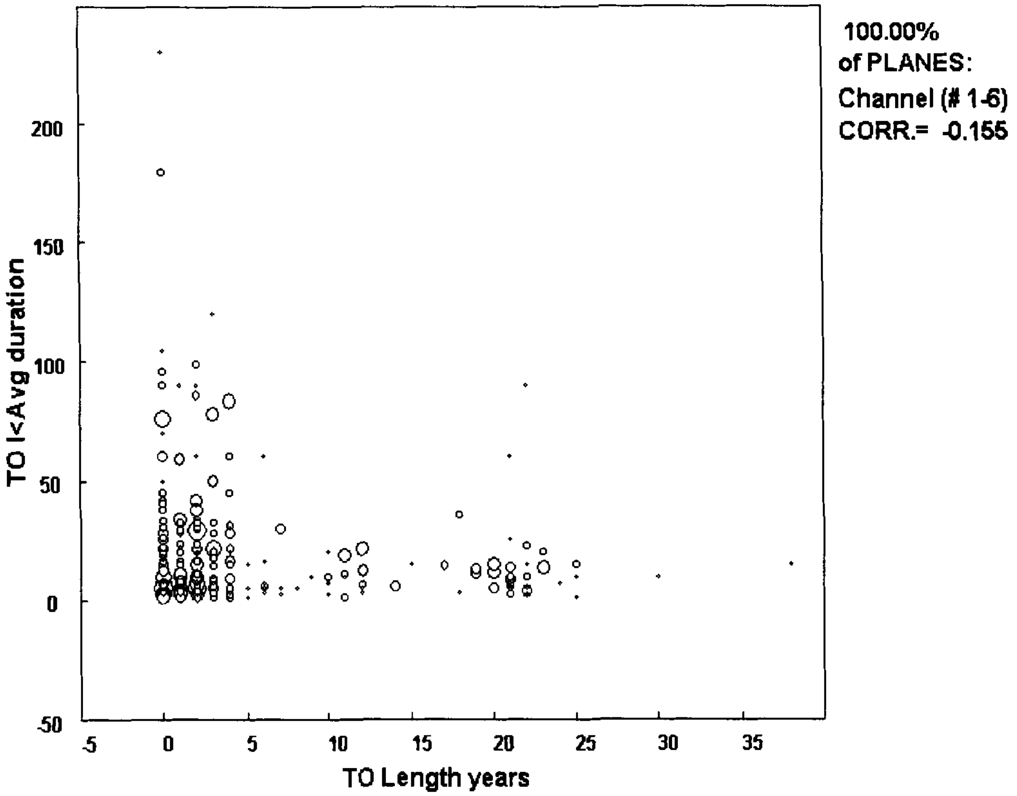


Figure 2.33. **CORREL** analysis of '**TO** Length years' vs '**TO** I<Avg Duration' on Channel. Correlations for Channel (Nbr of points) are: `'face-face=-0.139(583), 'phone'=-0.176(203), 'e-mail'=-0.16(17), 'written'=-0.625(8)`

Table 2.15. **ANOVA** tables for Dependent: '**TO** I<Avg Duration' and Independent: 'Channel'

```
STRENGTH=1
Independent: "Channel"                    Dependent: "TO I<Avg duration"
 SUMMARY OF "Channel"

 LABEL     |  SIZE     |  MEAN     |CONF. INT.|
----------|----------|----------|----------|
face-face |   583    |    22.1   |     1.66 |
phone     |   203    |    18.39  |     2.81 |
e-mail    |    17    |    15.75  |     9.7  |
written   |     8    |    16.26  |    14.13 |
fax       |          |           |          |
other     |          |           |          |

ANOVA TABLE
------------------------------------------------------------
SOURCE OF |  Sum of   | Deg. of   | Variance | Obtained |  P
VARIATION |Squares SS| Freedom   | Estimate |  ratio   |
------------------------------------------------------------
BETWEEN   |  2730.2   |      3    |   910.07 |     1.54 |> 0.10
GROUPS    |           |           |          |          |
------------------------------------------------------------
WITHIN    |477800.44  |    807    |   592.07 |
GROUPS    |           |           |          |
-----------------------------------------------
TOTAL     |480530.64  |    810    |
```

Table 2.16. ANOVA tables for Dependent: **'TO** Length years' and Independent: 'Channel'

```
STRENGTH=1
 Independent: "Channel"          Dependent: "TO Length years"
 SUMMARY OF "Channel"
  LABEL    |   SIZE    |   MEAN    |CONF. INT.|
----------|----------|----------|----------|
face-face |    583    |     4.13 |     0.48 |
phone     |    203    |     7.53 |     0.82 |
e-mail    |     17    |     3.53 |     2.82 |
written   |      8    |     5.25 |     4.11 |
fax       |           |          |          |
other     |           |          |          |
ANOVA TABLE
 ------------------------------------------------------------
SOURCE OF |  Sum of   | Deg. of   | Variance | Obtained |  P
VARIATION |Squares SS| Freedom   | Estimate |  ratio   |
-----------------------------------------------------------
BETWEEN   |  1781.01 |       3   |   593.67 |    11.87 |< 0.01
GROUPS    |          |           |          |          |
-----------------------------------------------------------
WITHIN    | 40355.63 |     807   |    50.01 |
GROUPS    |          |           |          |
--------------------------------------------
TOTAL     | 42136.64 |     810
```

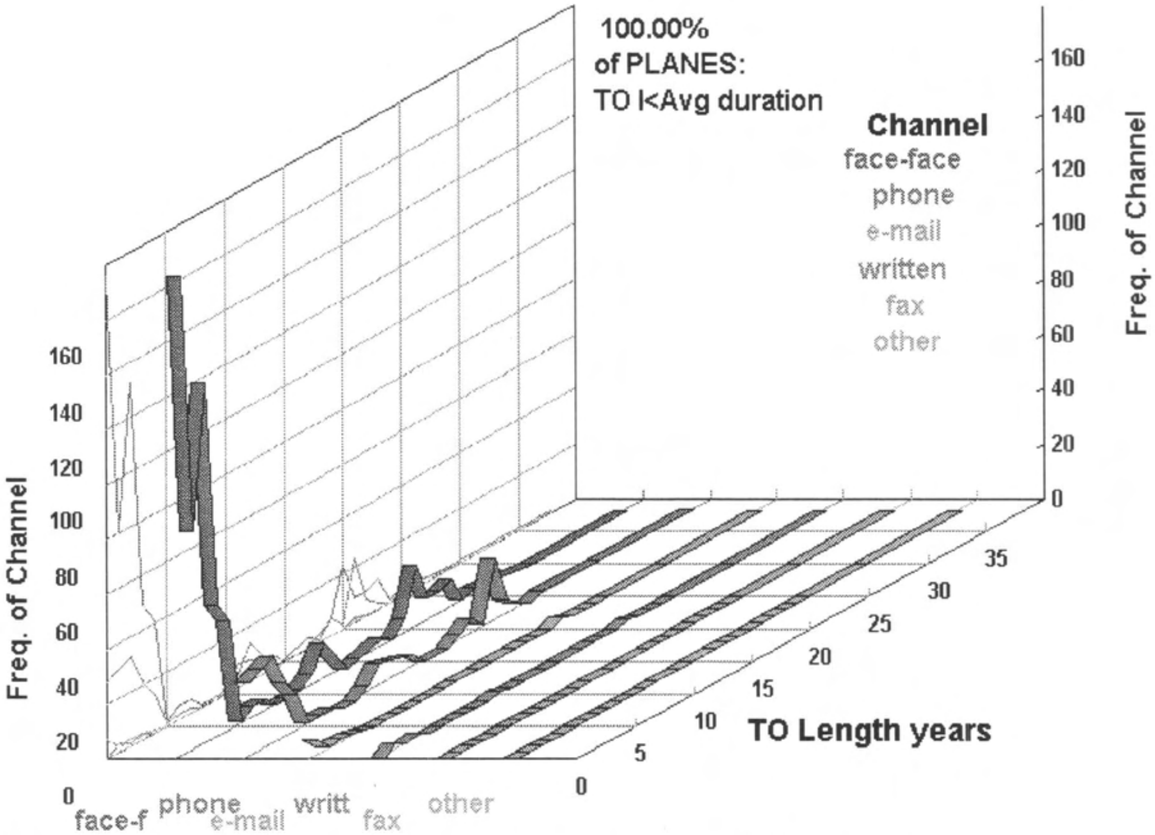


Figure 2.34. ANOVA display for Dependent: **'TO** Length years' and Independent: 'Channel'

Obviously, the examples shown in this section are only a small sample of what can be done in MultiNet. There are also examples in the Groupings section which explore a somewhat different type of network analysis. To use MultiNet effectively requires a knowledge of what the variables in a dataset measure individually; with some practice the program can help in understanding the relationships among variables and eventually the dataset as a whole. Sections 9 and 10 contain some more detailed examples showing how the various parts of MultiNet can be used together to explore large complex datasets.

2.5 Technical appendix

2.5.1. List of errors anticipated by **Analyse** module.

Generic values for Any variable, or for Node, and Link variable names represented by <Aname>, <Nname>, <Lname> respectively. <Aname> can refer to FROM/TO Nodes or Links. Numbers are represented by <n#>.

`Error Text` is followed by

- Explanation
- Solution

`TOO MANY VALUES (<n1>)`

- A categorical variable is expected. The chosen variable has more distinct values than the current setting for Number of Categories.
- Solution: If the variable has less than 20 distinct values, Use **Preferences→Number of Categories** to increase the setting. Otherwise, choose a different variable.

`ONLY <n1> VALUES`

- This is a warning message, not an error, and analysis can proceed. A numeric/continuous variable is expected, and the chosen variable has distinct values which are no more than the current maximum for categorical variables.
- Solution: Proceed only if the analysis makes sense given the small number of distinct values.

`NO LINKS FROM/TO <Nname>`

- Network analysis error. A Node attribute which is defined only for receivers (**`NO LINKS FROM`**) or only for senders (**NO LINKS TO**) is being used in the wrong direction.

- Solution: Change the direction of use for this Node attribute.

NO <Aname1> is <Aname2>

- Analysis cannot proceed since the intersection of these two variables is empty.
- Solution: These two variables cannot be analysed together.

NO <Lname> AMONG <Aname1> <Aname2> [<Aname3>]

- Network analysis error. Directions are correct, and intersection of variables is non-empty, but the chosen link (strength or multiplier) has only 0 values for these variables. This means that there is no network connecting these Nodes.
- Solution: Choose a different node or link variables.

NO DATA ON THIS PLANE

- Warning message. Three variable analysis with Empty Categories can result in planes with no data. This message appears in both the graphic display and the text report.
- Solution: No action needed. This message can be avoided by choosing

  Preferences→Categories→Delete Empty Categories

MORE THAN 52 VARIABLES

- Network analysis warning message. Pressing **GO** followed by selecting **Equation** opens a multiple selection window if more than 52 link variables are available. Selecting a subset of 152 or less of these allows the 52 symbols **a-zA-Z** to be associated with these link variables. Otherwise, this error appears.
- Solution: Choose 52 or less link variables from multiple selection window.

NO EQUATION

- Network analysis error message. Pressing **GO** followed by selecting **Equation** opens a text window along with a window showing association of symbols **a-zA-Z** with link variables. Returning a blank text window causes this error.
- Solution: Do not return blank text window.

MISPLACED PARENTHESES

- Network analysis error message. Pressing **GO** followed by selecting **Equation** opens a text -

window along with a window showing association of symbols **a-zA-Z** with link variables. This error occur if parsing the equation fails due to misplaced or non-matching parentheses.

- Solution: Enter syntactically correct equation.

MISSING OR INVALID SYMBOL/OPERATOR

- Network analysis error message. Pressing **GO** followed by selecting **Equation** opens a text window along with a window showing association of symbols **a-zA-Z** with link variables. These errors occur if parsing the equation fails due to two symbols (**a-zA-Z**) or operators (^*/+-=<>) beside each other.
- Solution: Enter syntactically correct equation.

ALL VALUES = 0

- Network analysis error message. Pressing **GO** followed by selecting **Equation** opens a text window along with a window showing association of symbols **a-zA-Z** with link variables. This error occurs if all Equation values evaluate to 0.
- Solution: Use link values that evaluate to at least some non-zero values.

SOME VALUES < 0

- Network analysis error message. Pressing **GO** followed by selecting Link or **Equation** is followed by selecting link variable(s). These must evaluate to non-negative values.
- Solution: Use link values that evaluate to non-negative values.

2.5.2 History of Panigrams

Panigrams were devised by Dr. William D. Richards as a way of visualizing network cross-tabulation results for a research client who found the percentages in a crosstabulation table just too confusing to understand, even with pages of text explaining the tables. Originally, panigrams were available only for FROM/TO node attributes in the computer program FATCAT, but were then extended to standard crosstabs and groupings (see Groupings section). This author further extended the methods to include continuous variables (ANOVA and CORREL) in MultiNet and added stacking and the various interpretations available with Rotate (Seary, 1997). This author also gave panigrams their name, in analogy to histograms which were named from the Greek istos ('mast'). Early histograms resembled a forest of one-dimensional ship's masts. Panigrams are named from the Greek panis ('sail'), which are the two-dimensional objects attached to masts.

2.6 References

Richards, W.D. (1986) FATCAT: a different kind of network analysis program, http://www.sfu.ca/~richards/Pdf-ZipFiles/fatman2.pdf

Richards, W.D. (1988). *Private Communication.*

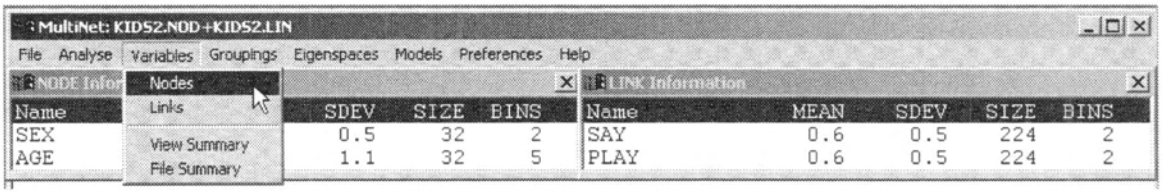
Seary, A.J.(1995) MultiNet for DOS, presented at International Conference on Social Networks, London UK

3. The Variables Module

3.1 Introduction

One of the important features of MultiNet is the ease with which new variables may be created for use in subsequent analyses. Though it is possible to create node variables in the Eigenspaces module, and link variables in the **Models →Pstar** module, most of the management of variables -- creating, modifying, arranging, and deleting – is handled by the Variables module

From the main menu, the Variables module is selected with a single click. This produces a sub-menu (Figure 3.1) allowing further choices: the first two (above the separator) are used to actually start the Variables module for either node or link variables. The remaining two selections (below the separator) do not start the Variables module. Rather, they present a summary of both node and link variables as a textual report, which may be either viewed of filed. These two selections are repeated within the Variables module (and will be described later), and are present in this sub-menu as a convenience.

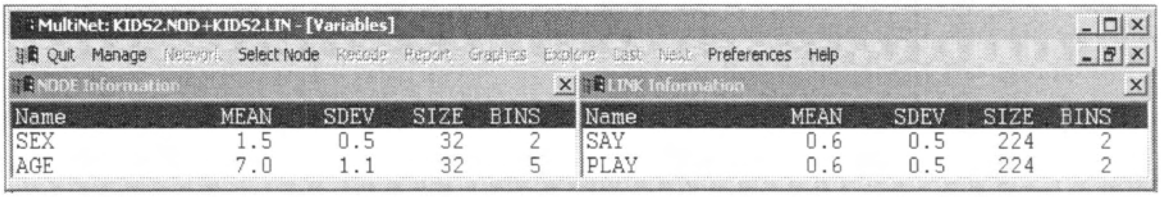


Name	SDEV	SIZE	BINS
SEX	0.5	32	2
AGE	1.1	32	5

Name	MEAN	SDEV	SIZE	BINS
SAY	0.6	0.5	224	2
PLAY	0.6	0.5	224	2

Figure 3.1. Starting the Variables module

Once the Variables module has started by selection of either node or link variables, the menu bar has everything greyed out (can't be selected) except **Quit, Manage, Select** and **Help** (figure 3.2). Help will produce descriptions of each of the sub-menu items. **Quit** will leave the sub-menu entirely and return to the Main menu. As a convenience **Manage** initially allows some management of node or link variables (and will be described in the section on **Manage**), but most of the functions in this module are unavailable until a variable is selected by **Select** (**Node** or **Link**). The menu bar otherwise looks similar for node and link variables, but some functions may be different or not available. These exceptions will be noted in the relevant sections.



Name	MEAN	SDEV	SIZE	BINS
SEX	1.5	0.5	32	2
AGE	7.0	1.1	32	5

Name	MEAN	SDEV	SIZE	BINS
SAY	0.6	0.5	224	2
PLAY	0.6	0.5	224	2

Figure 3.2. Initial display for Variables module

3.1.1 Visualizations

Clicking on **Select** produces a new selection window containing a list of all the node or link variables. Once a variable has been selected by the usual method, it is displayed as a black histogram, with an overlayed dark grey cumulative distribution (figure 3.3). The axis markings for the values of the variable are along the horizontal axis, while the cumulative and frequency values are displayed on the left (in dark grey) and the right (in black) vertical axes. In addition, a pair of dotted cursor lines intersect the cumulative distribution horizontally and the actual values vertically. Initially, this cursor is set to the lowest value of the variable. Finally, a separate, movable "**Explore Data**" window is created which allows for controlled movement of the cursor. The left end of the cursor is labelled in dark grey with the cumulative proportion up to this value (above) and the percent this value contributes (below). The bottom end of the cursor is labelled with the actual data value, and the number of data items with this value.

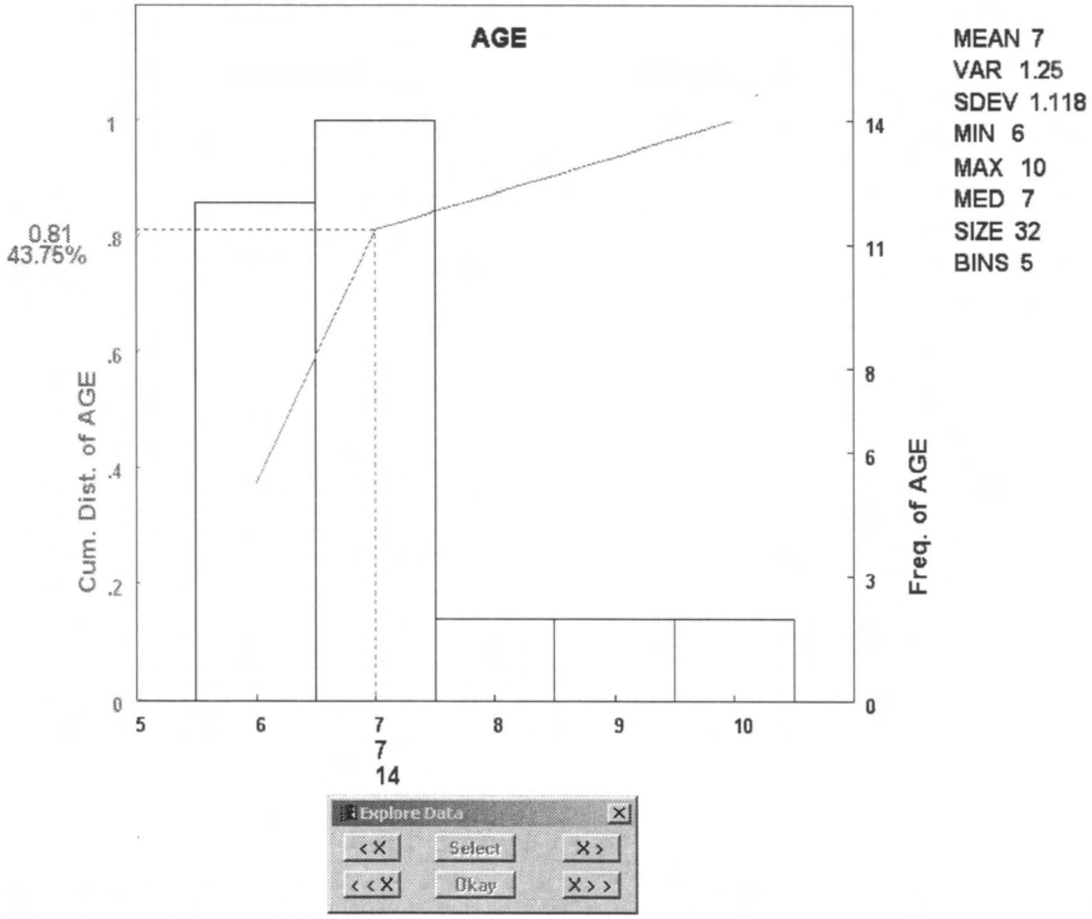


Figure 3.3. Visualization of node variable AGE. The cursor has been moved from the initial position.

The name of the variable labels the upper centre of the graphic display On the right of the graphic, a text area lists some important summary statistics. Note that some of these statistics must be interpreted with care for categorical variables coded as integers. For example, all are meaningful for the variable AGE shown above. However, MEAN, VAR[iance] and S[tandard] DEV[iation] are hard to interpret for SEX coded as 1=Male and 2=Female. The statistics are:

- MEAN: the average value of the data items
- VAR: the variance of the data items
- SDEV: the standard deviation of the data items (square root of variance)
- MIN: the minimum value
- MAX: the maximum value
- MED: the median value, for which the cumulative proportion is exactly 0.5
- SIZE: the number of data items
- BINS: the number of unique data values
- NODES: *for link variables only* the number of unique nodes in the network defined by the link variable

The presence of the last statistic is one of the differences between the displays of node and link variables. If value labels have been defined for this variable, they are also displayed below the statistics. Figure 3.4 shows the display for a link variable with value labels.

In figure 3.3, we see by looking at the values below the cursor that there are 14 nodes (out .of a possible SIZE=32) with AGE value of 7. We also see by looking at the labels to the left of the cursor that this corresponds to a cumulative total of 0.81 and that these 14 values constitute 43.75% of all the nodes that do not have missing values for AGE. In figure 3.4, we see that there are 82 data values of 0 (no link) out of a possible SIZE=224, and that this constitutes 37.7% of the data values. Care must be taken here, since this is the percent of the sparse representation of the link variable.

The histogram is an exact representation of data frequency when the number of unique values (*bins*) is not greater than a user selected limit (set by **Preferences**) with default 30 and maximum 100. If the number of bins is greater than this value, there may be more than one data value in each of the histogram bars. The cursor actually follows the dark grey cumulative distribution curve, not the bins of the histogram, so that the cursor steps through every data value.

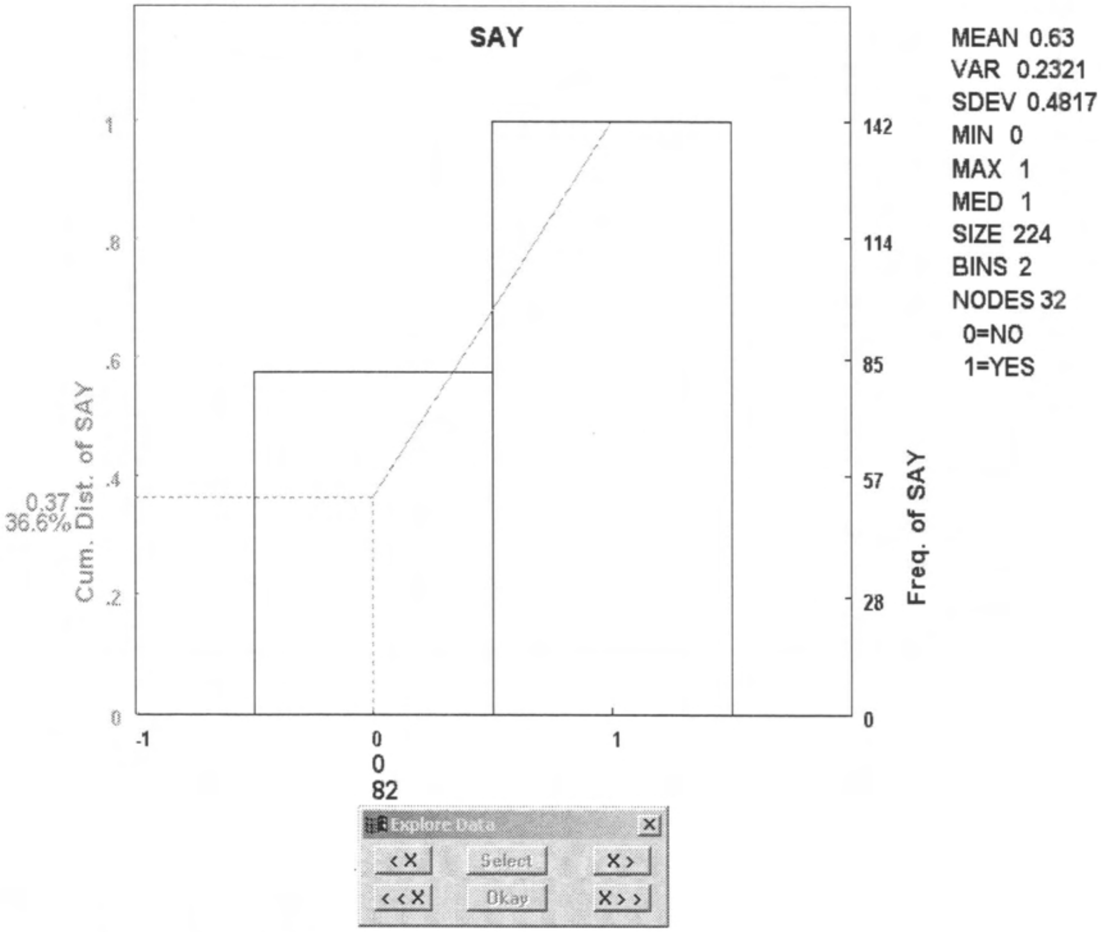


Figure 3.4. Initial display of a link variable with value labels. MAX, MIN and BINS show this to be a binary valued variable.

3.1.2 The **Explore Data** window

At the bottom is the **Explore Data** window, which is a control center for the display. This is a separate window which can be moved around by dragging the title bar: a standard Windows method. The Variables module provides an interactive display for stepping through data values, using the buttons on the **Explore Data** window. In addition, this window contains buttons for selecting data ranges within the **Recode →Discrete→User** function.

These are otherwise greyed-out and will be discussed later. The cursor movement buttons are:

- **<X** moves the cursor one data value to the left with left-click
- **X>** moves the cursor one data value to the right with left-click.
- **<<X** moves the cursor to the left continuously while the left mouse button is held down.
- **X>>** moves the cursor to the right continuously while the left mouse button is held down.

If right button is used instead of left, all movements occur in steps of 10. In every case cursor movement stops at the minimum or maximum value.

3.2 Variables Menu Bar

The menu bar contains 12 items (Figure 2.3). The content and availability of some of these items depends on whether node or link variables have been selected. This will be noted in the descriptions below.

Quit

Exit from the Variables Module back to the Main MultiNet menu.

Manage

Create, delete, arrange, edit and modify the display of node and link variables.

- **Manage→Create** is enabled once a variable has been selected. When a variable is changed with **Recode**, the changes are said to be "pending" until you instruct MultiNet to accept the changes by pressing **Manage→Create**, upon which a new variable is created with name and comments you select in edit windows. Selecting "Cancel" for either of these windows cancels the variable creation. For convenience both the variable Name and Comment default to the list of changes you have made. You cannot create a variable with the same name as another. Creating a variable results in automatic selection and display of the new variable. NOTE: Some variables are created automatically. See **Recode→Degree** and **Recode→Components**.

- **Manage→Delete** is always enabled. Select this to remove variables from the current dataset in memory. This does not affect any files (until you **File→Save** the current dataset after making changes). Choosing ths function opens a multiple selection window in which selections are made from the list of variables with Space+Click or Ctrl+Click, and selection is finished with Enter or the Okay button. You will be prompted to confirm each deletion, then you are put back in the Variables menu, and must repeat all the operations for more deletions. Deleting variables restores the Variables module to its initial state, waiting for a variable to be selected. NOTE: deleting variables is not reversible. The deletion process may be ended at any point by pressing the Cancel button, but those already deleted cannot be recovered.

- **Manage→Replace** is enabled once a variable has been selected, and allows replacement of any variable with the values, value labels, and comments of the (possibly pending) variable currently being displayed. **Manage→Replace** is a convenience provided to allow renaming of variables created automatically (e.g. from **Recode→Degree**), changing the value labels automatically generated by some **Recode** functions, and editing comments. Choosing **Manage→Replace** opens a selection window with a list of variable names. Choosing one of these then opens an edit window with the name of the chosen variable. You may then change or accept the variable name. Another edit window then opens for comments. The default comment is the list of **Recode** actions performed on the pending variable unless no changes have been made, in which case the original comment is used. If any value labels have been changed by **Manage→Labels**, they are also replaced. Selecting "Cancel" for either of these windows cancels the replacement.

- **Manage→Labels** is enabled whenever the current variable has value labels, and is a convenience that allows changing the value labels automatically generated by some **Recode** functions (e.g., **Quantiles**). Choosing **Manage→Labels** opens an edit window with two columns headed 1) "Values" followed by a column of all unique data values and 2) "Labels" followed by a column of the value labels currently assigned to the values. You may edit the contents of column 2. Any changes to column 1 produces an error message and all changes are ignored. Pressing Save then replaces the current value labels with the contents of column 2, while pressing **Quit** ignores any changes made. Changes affect only the pending variable, and do not become permanent until **Create** or **Replace** is chosen.
- **Manage→Arrange** is always enabled and is a convenience which allows you to arrange the order of variables. Selecting ths function opens a multiple selection window in which you toggle selections from the list of variables with Shift+Click or Ctrl+Click, and finish selecting with Enter or Okay button. The variables you have selected move to the beginning in any display or selection window of either node or link variables. Press Cancel to avoid any change. Any change in the order of variables restores the Variables module to its initial state, waiting for a variable to be selected.
- **Manage→Comments** is enabled when the variables list shows the brief descriptive statistics. This function allows you to have the descriptive comments associated with variables displayed in the variable list, rather than the brief descriptive statistics. A descriptive comment may be associated with any variable when it is first created. Descriptive comments may also be included in the original text data file after the column definitions. Choosing this function restores the Variables module to its initial state, waiting for a variable to be selected.
- **Manage→Statistics** is enabled when the variables list shows the descriptive comments. This function restores the brief descriptive statistics in the variables list. Choosing this function restores

the Variables module to its initial state, waiting for a variable to be selected.

- **Manage→View Summary** is always enabled and produces a textual report of all current Node and Link variables, and displays it in an edit window. Included are variable names, statistics and comments. Also included are lists of any Node IDs that do not appear as Links IDs, and Link IDs that do not appear as Node IDs. This duplicates a function available from the main menu under Variables.
- **Manage→File Summary** is always enabled and produces a textual report of all current Node and Link variables, and files it in the current .OUT file. Included are variable names, statistics and comments. Also included are lists of any Node IDs that do not appear as Links IDs, and Link IDs that do not appear as Node IDs. This duplicates a function available from the main menu under Variables.

Network

This menu item may be enabled only for node variables. Once you have selected a node variable, you may be interested in looking at the distributions of links sent **FROM** each node value, and **TO** each node value. If you press **Network** you may look at these for a particular link variable, which is chosen from a selection window by standard methods. MultiNet will give a warning if there are no **FROM** or **TO** links to display. Initially, the Report includes node statistics and distribution only. Using Network to select a link variable will result in adding both **FROM** and **TO** statistics and distributions for the selected link variable to the Report. The graphic display becomes a histogram counting the number of links sent **FROM** (or **TO**) each unique value of the node variable. You may restore the node distribution display by selecting Network→Standard, but this does not remove the additional information from the Report. To restore the initial report (without link information), **re-Select** the node variable. NOTE: **Recode** is disabled while **FROM** or **TO** distributions are displayed. Also **Network** is not enabled for *pending* variables that have been recoded but not yet saved into a permanent variable. The statistics and distribution tables provided by **Network** is a convenience, since they are also produced by the network analyses available in the MultiNet Analyse module.

Select (Node or Link)

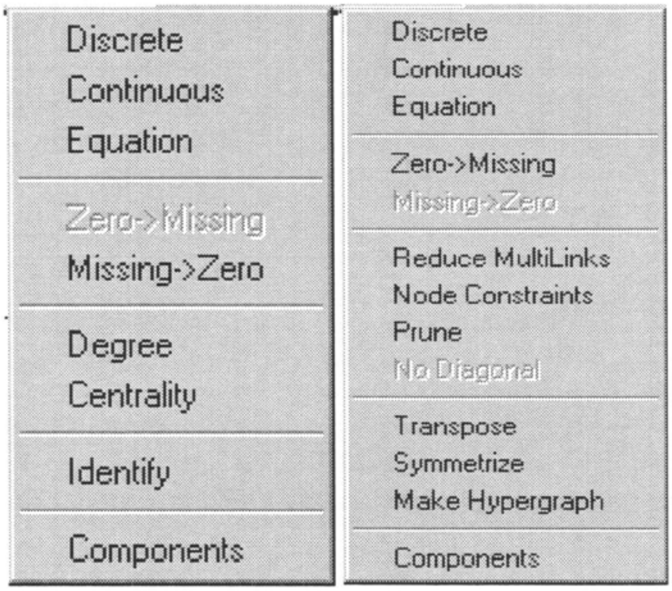
The text of this menu item depends on whether you have chosen to examine node or link variables. This menu item is always enabled, and is generally the first item chosen from the menu bar. Press Select to examine the distribution of values for an individual variable. The variable is chosen from a selection window using standard methods. Once the variable has been selected, a graphic display shows the distribution of the variable and a textual report containing tables of statistics and distribution of the variable is created. Most of the other menu items also become available.

3.3 Recode

Recode allows variables to be transformed in various ways. Clicking on this item produces a menu of further choices (figure 3.5). Some transformations are appropriate for node variables but not for link variables and vice-versa, so some of the choices are different for nodes and links. This menu item is quite complex, and so the description of **Recode** is broken down into three sets of descriptions:

- The first five choices (all those above the second separator) are quite similar in effect for nodes and links so can be discussed together.
- choices that apply to node variables are discussed together.
- choices that apply to link variables are discussed together.

Choices are only enabled when a transformation is possible. For example, in figure 3.5a the selected node variable has no data values of 0, so the choice **Zero->Missing** is not enabled.



a) node variable choices. **b)** link variable choices

Figure 3.5. Recode menus for Node and Link variables.

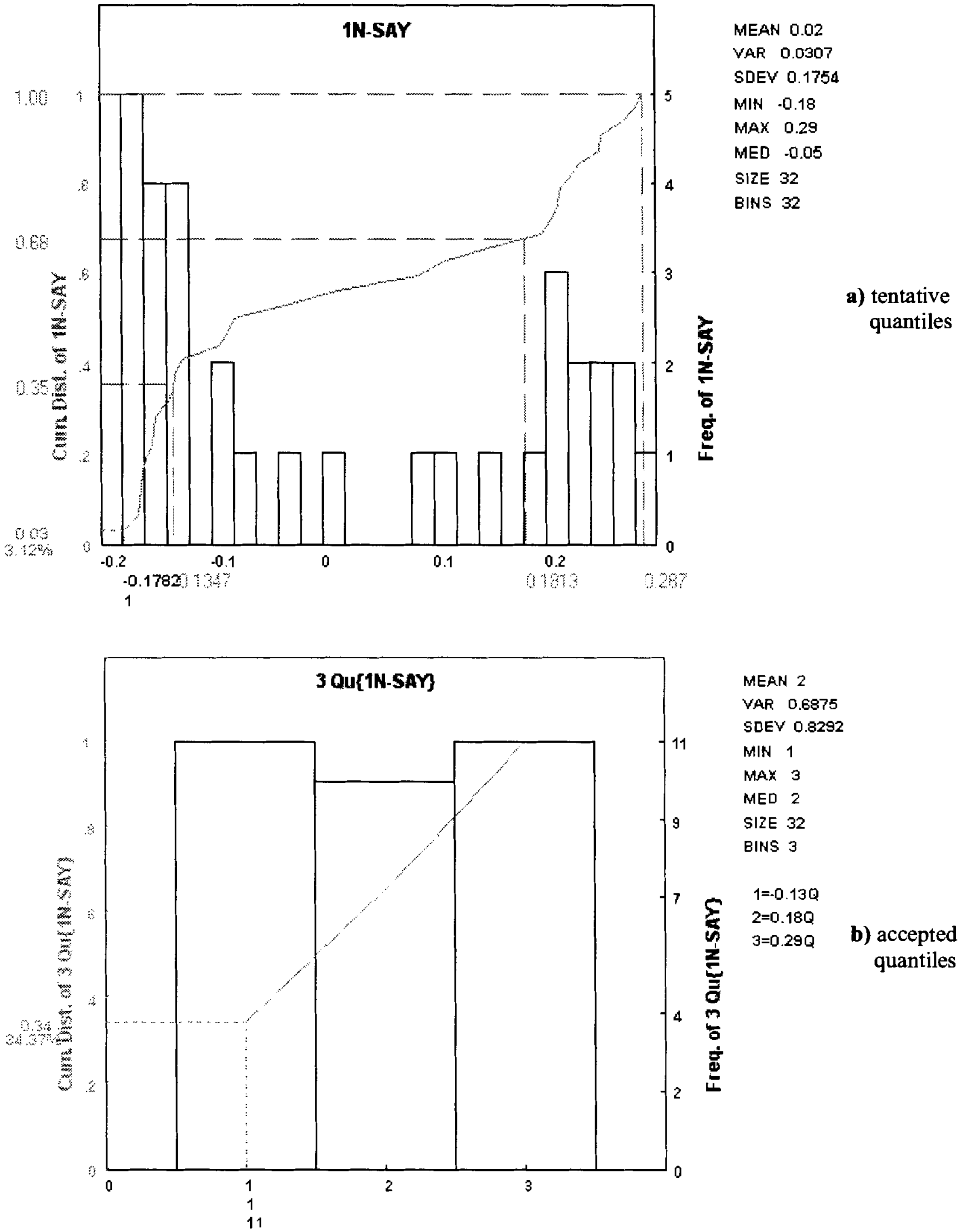
The following descriptions of choices apply to both node and link variables.

3.3.1 Discrete

This choice allows transformations that result in a discrete, categorical variable, with a small number of unique integer or binary values. Selecting this choice with a click opens a selection window with the following choices:

- **Quantiles** will recode the variable currently selected into a specified number of quantiles. Type the number of quantiles into the edit window and press enter. This number must not be more than a user selected limit (set by **Preferences**) with default 12 and maximum 20. Initially the new quantiles are shown superimposed on the variable distribution as green dashed lines and labels (Figure 3.6). These lines intersect along the cumulative distribution, and are as equally spaced as possible along the Y-axis. The left ends of the horizontal parts are labeled with the cumulative values (and these values should be approximately equidistant), while the bottoms of the vertical parts are labeled with the corresponding data values. This display persists until a YesNo window is clicked. A No response requests a new number of quantiles. A Yes response displays the new pending variable, which corresponds to the quantiling of the original selected variable, with value 1 corresponding to the first quantile, 2 to the second and so on. Value labels are automatically created to correspond to the original data value to the rightmost in each quantile. For example, quantile 1 contains data up to -0.1347, so it is labeled as **-0.13Q** (rounded to 2 decimal figures) quantile. The display of the recoded (and pending) variable is labeled as **3 Qu{1N-SAY}** to show how it was created. This label will automatically become the descriptive comment if a new variable is now created. In this example the number of nodes in each quantile cannot be the same (since there are 32, which is not divisible by 3). It will generally be the case that not all quantiles contain exactly the same number of items.

* **Bins** will recode the variable currently selected into a specified number of bins. The results are very similar to **Quantiles**, except the data is divided as evenly as possible into bins based on data values, rather than cumulative distribution. Type the number of bins into the edit window and press enter. This number must not be more than a user selected limit (set by **Preferences**) with default 12 and maximum 20. Initially the new bins are shown superimposed on the variable distribution as blue dashed lines and labels. These lines intersect along the cumulative distribution, and are as equally spaced as possible along the X-axis. The left ends of the horizontal parts are labeled with the cumulative values, while the bottoms of the vertical parts are labeled with the corresponding data values (and these values should be approximately



a) tentative quantiles

b) accepted quantiles

Figure 3.6. a) shows the tentative 3-quantiling of a continuous variable, with cumulative values on the left and actual data values below. The green dashed lines intersect on the cumulative distribution.

b) shows the results after the tentative 3-quantiling is accepted. The value labels on the right are based on the upper data value in each quantile. The pending variable is labeled to show how it was created. This label will automatically become the descriptive comment if a new variable is now created.

equidistant). This display persists until a *yes/no* window is clicked. A No response requests a new number of bins. A Yes response displays the new pending variable, which corresponds to the binning of the original selected variable, with value 1 corresponding to the first bin, 2 to the second and so on. Value labels are automatically created to correspond to the original data value to the rightmost in each bin. For example, assume a 3-binning of the data shown in figure 3.6. Then bin 1 contains data up to -0.0231, so it is labeled as **-0.02B** (rounded to 2 decimal figures). The display of the recoded (and pending) variable is labeled as **3 Bi{1N-SAY}** to show how it was created. This label will automatically become the descriptive comment if a new variable is now created. It will generally be the case that not all bins contain exactly the same number of items.

- **User** allows interactive discretization of data with ranges chosen by the user. When this function is selected, two items become enabled in the **Explore Data** window: **Select** and **Okay**. The cursor is moved as usual, with the labels on both axes helpfully showing data value and cumulative value. When the cursor is as a desired value, pressing Select draws a red dashed line and labels, defining the end-point of a tentative category of data. This tentative category may be removed by moving the cursor to an existing red tentative end-point and pressing Select again. This allows considerable interaction in setting up categories. When the tentative discretization is complete, pressing **Okay** produces a result like figure 3.6b. For example, assume 3 categories have been selected. Then if category 1 ends at -0.1524, it is given label **-0.15U**,. The display label (and descriptive comment) is **3 Us{1N-SAY}**, showing how the variable was discretized.
- **Rank** is available as a choice if the variable consists of not more than a user selected limit (set by Preferences) with default 12 and maximum 20. The data values are categorized by their rank (least =1, greatest = number of unique values). Value labels are **1R**, **2R**,..., up to number of unique values. The display label (and descriptive comment) is **Ra{...}**. Rank is a convenient way to convert categories with non-contiguous coding values (e.g., 0, 3, 4, 9,...) into contiguous integer values (1, 2, 3, 4, ...).

3.3.2 Continuous

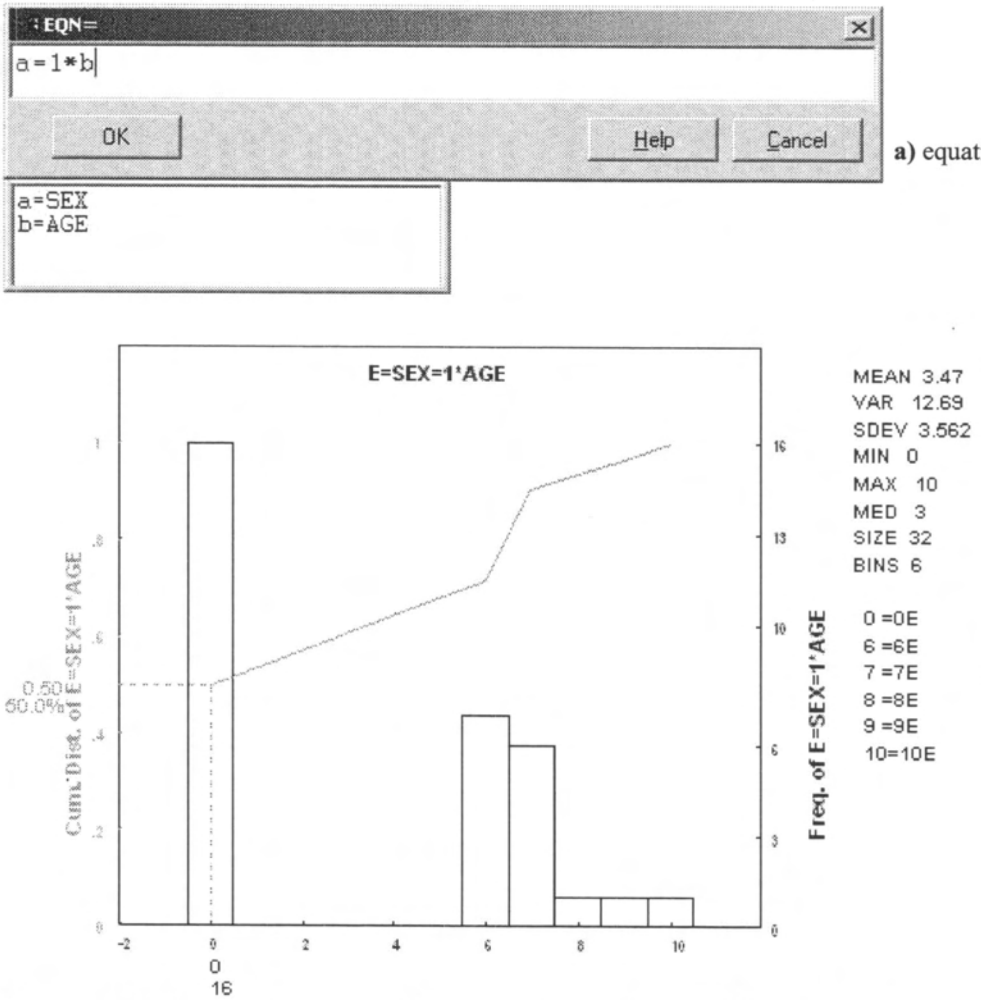
This choice allows transformations that result in a continuous (real-valued) variable, and is most appropriate when applied to such data. However, no warning is made if these functions are applied to integer (or even binary) data, and it is up to the user to make sure the transformation is reasonable. These transformations affect only the data values, not the frequencies, though the resulting histograms and distribution tables may be somewhat different. Selecting this choice with a click opens a selection window with the following choices:

- **Lg10** takes the base-10 log of the data values. The new label (and comment) is **Lg{...}**. An error message results if any data value is less than or equal to 0.
- **Pwr** allows powers and roots to be taken. An edit window opens in which the power is entered. A positive integer such as n means take the nth power of the data values (e.g, 2 means take the square). A fraction such as 1/n means take the nth root of the data values (e.g., 0.5 means take the square root). For power n, the new label (and comment) is **n Pw{...}**. An error message results if extracting a root would result in imaginary values.
- **Std** subtracts the mean and divides by the standard deviation of the data values. This rescales the data values to have mean = 0 and Std. Dev. = 1. The new label (and comment) is **St{...}**.
- **Rank** replaces each data value with its rank in a rank-ordering (increasing values). Strictly speaking, this produces an integer result. This function would be useful, for example, in replacing dates (YYMMDD) with day numbers. The new label (and comment) is **Ra{...}**.

3.3.3 Equation

This very powerful function allows arithmetic and logical combinations of a number of variables. When this is chosen, two windows open. One is a display window, showing alphabetic symbols **(a-z,A-Z)** beside each of the actual variable names. (If there are more than 52 variables, a multiple selection window opens first, allowing choice of up to 52 variables). Above the display window is an edit window in which an equation may be typed. The symbols beside the variable names are used to construct an equation describing how to define the a new variable. Standard precedence rules are used for arithmetic. e.g., **a*(b+c)** means construct a new variable from **a** times the sum of **b** and **c,** where **a, b** and **c** are symbols representing variables.

You may also use **=** and **<** and **>** to construct new variables. These evaluate to the logical values 1 or 0. For example: **10=a** evaluates to 1 only for values of a exactly equal to 10. **10<a** (or **a>10**) evaluates to 1 for all values of a greater than 10. **a<10** (or **10>a**) evaluates to 1 for all values of a less than 10. These expressions may be used in an equation to select ranges of a variable. For example: **a*(10<a)** (or **a*(a>10)**) makes a variable which is the value of a if a is greater than 10 or is 0 if a is 10 or less. Such logical expressions may also be used to select subsets of a variable which depend on another variable. In conjunction with **=** or < or **>**, * acts like logical AND, while + acts like logical OR. A *logical equation* uses only these operations to produce the logical results 0 or 1, as in **Recode→Node Constraints** For example, in figure 3.7 the equation **a=1*b** is used to select the ages (variable **b**) for which the sex (variable **a**) is 1 (male). Figure 3.7a shows the windows and equation, while figure 3.7b shows the result. Notice that all the females have value 0 and could be removed.



```
a=1*b
```

```
a=SEX
b=AGE
```

a) equation and symbols

```
MEAN 3.47
VAR  12.69
SDEV 3.562
MIN  0
MAX  10
MED  3
SIZE 32
BINS 6

 0 =0E
 6 =6E
 7 =7E
 8 =8E
 9 =9E
10=10E
```

b) result

Figure 3.7. a) shows the equation **a=1*b** being entered, where **a**=SEX and **b**=AGE
b) shows the result. Symbols are replaced with variable names in label (and comment).

The equation in figure 3.7 also demonstrates the precedence rules: The equation is parsed left-to-right, with precedence

+- Addition and Subtraction have lowest precedence

***/** Multiplication and Division have higher precedence

^ Exponentiation has still higher precedence.

These are standard rules which make it easy to write polynomials.

2*a^2+3*a-1 is interpreted as **(2*(a^2))+(3*a)-1**

For convenience, **=**, **<** and **>** are given highest precedence. This makes it easy to write a=1*b without parentheses as in figure 3.7, since it is interpreted as **(a=1)*b**. You may choose to use extra parentheses for clarity

Since the result is a combination of variables with a small number of unique values, it also has a small number of unique values, not more than a user selected limit (set by **Preferences**) with default 12 and maximum 20. This allows the value labels **0E** to **10E** to be created. The display label (and descriptive comment) is **E=SEX=1*AGE** showing how this variable was created from an equation. In this case it is straightforward to interpret all the 0 values as belonging to the females. However, these zero values affect the statistics. It would be useful to consider these values as "not there" in some way, which is the topic of the next section.

In the case where one or more of the variables used in the equation have missing data, the result of any equation must also have missing data. It is possible that the result of an equation consists entirely of missing data, which results in an error message and no change in the current variable. This can happen, for example, when there are two types of nodes (e.g., people and events) which may have variables describing people (with missing data for events) and variables describing events (with missing data for people). How this may be handled in an equation is also described in the next section.

3.3.4 Zero->Missing

Zero->Missing is enabled if there are any data values of 0 in the current variable. This function marks all data values of 0 as missing data and removes the zeros. A number of **Recode** functions produce data values of 0 to show that data may be removed (declared as missing data). Examples are: **Equation, Missing->Zero**, **Prune**, **NoDiag**, **Components**. The example in the previous section resulted in a variable with statistics skewed by the logical 0's. Applying **Zero->Missing** would produce the correct statistics, as shown in figure 3.8. The label (and descriptive comment) is now **Zm{E=SEX=1*AGE}** showing how the variable was created.

3.3.5 Missing->Zero

Missing->Zero in the inverse of **Zero->Missing**, with label **Mz{...}**. It is enabled if there is any missing data in the current variable and it replaces all missing data values with zeros. This can be useful when it is necessary to combine two variables whose non-missing intersection is empty (e.g., the people and event variables described above). This would involve creating a pair of temporary variables with **Missing->Zero**, putting them together in an equation, then deleting the temporary variables. This function should be used with care, especially for data where 0 can be a meaningful value, or for large non-binary link variables with many missing values. Note that sparse representation means that link data exists (possibly with value 0) or is declared missing for node ID pairs with a value for at least one link variable: any other ID pairs are not part of the dataset.

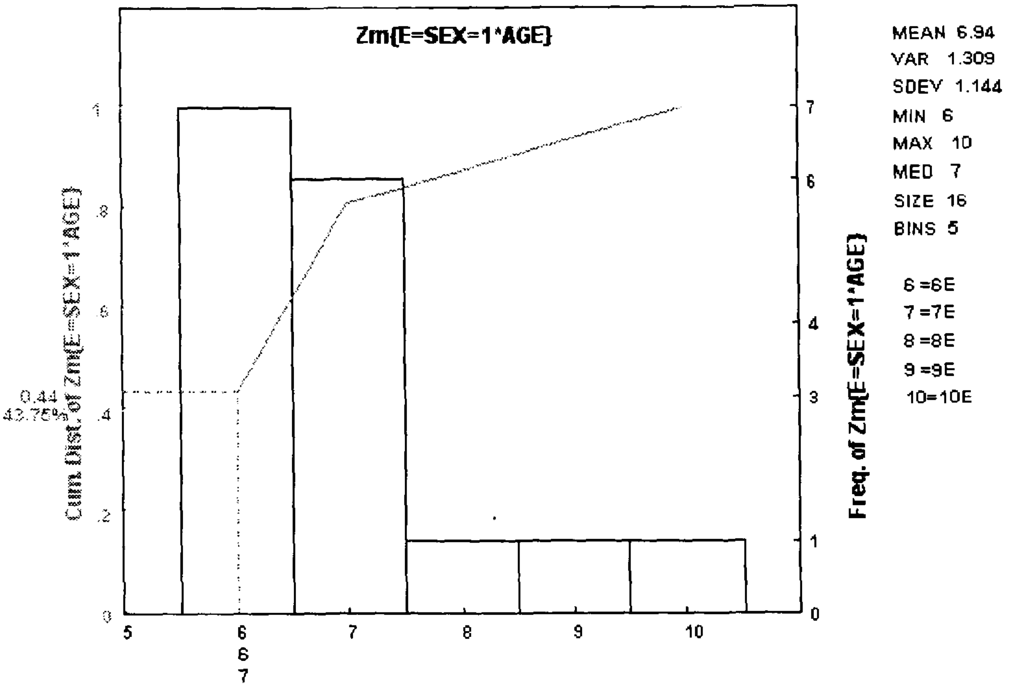


Figure 3.8. Using **Zero->Missing** on the result of the equation in figure 3.7.

<u>The following descriptions of choices apply to node variables only.</u>

3.3.6 Degree

Degree produces a node variable that depends on which network (link variable) is being measured. Choosing **Degree** opens a selection window for choosing a link variable. After choosing a link, a multiple selection window is opened which may have two or four choices. If the link variable is strictly binary, there will be choices for out-degree and in-degree only. If the link variable is not strictly binary (though it must be non-negative), two more choices appear: weighted out- and in-degrees. Once the choices have been made, the new variables are created automatically. This requires a naming convention as well as a convention for comments. The names are derived from the link variable, with additional characters prepended as shown in the table below.

o> binary out-degree counts the number of links sent from each node (integer valued).

I< binary in-degree counts the number of links received by each node (integer valued).

O> weighted out-degree counts the sum of all link values sent from each node.

I< weighted in-degree counts the sum of all link values received by each node.

For example, the binary out-degree for each node of link variable SAY is automatically given the variable name **o>SAY**, and the comment "**Bin OUT-DEG of SAY**". Each potential node variable is presented before creation with variable name and then with comment, either of which can be changed by the user in an *edit window*. The creation process can be cancelled at any time, though variables already created remain. If any variables are created, the first becomes the selected variable, which is then displayed.

If it is known that the link variable is also non-directed (symmetric), these both [weighted] out- and in- calculations produce the same result. Otherwise, for directed networks [weighted] out- and in-degree will have different distributions. It is quite possible, especially for directed or bipartite networks, that some nodes have zero values for either out- or in-degree, in which case **Zero->Missing** could also be applied. **Degree** produces values only for those nodes that take part in the network defined by the selected link variable. All other nodes are marked as having missing data for the new variable.

3.3.7 Centrality

A large number of methods have been devised that assign descriptive measures of networks to nodes. One of the aims is to identify "important" nodes. A simple example is out-and in-degree, and indeed nodes with very high degree are generally "well-connected". The most popular measures include:

- Degree: number of connections from (out) or to (in) each node (Freeman, 1979)
- Betweenness: number of geodesics that each node is on (Freeman, 1979; Brandes, 2000)
- Closeness: reciprocal of sum of distances to or from all other nodes (Freeman, 1979)
- Influence: derived from the walk-generating matrix (Biggs, 1993) with attenuation (Katz, 1953; Foster, et.al., 2001)
- Integration: sum of diameter minus distances from all nodes (Valente & Foreman, 1998)
- Radiality: sum of diameter minus distances to all nodes (ibid)
- Eigenvector: Frobenius eigenvector of symmetrized adjacency matrix (Bonacich, 1972)

This list is by no means exhaustive. For details, see cited works. In each case the measure is "normalized" to run between 0 and 1 so that networks of different sizes may be compared. MultiNet calculates binary Degree (out- and in-) in **Recode→Degree**, and these may be normalized using **Recode→Equation** simply by dividing by the number of nodes-1. Similarly, Eigenvector centrality may be calculated in the **Eigenspaces** module by choosing Standard, saving the largest eigenvector, and dividing by number of nodes-1. The **Recode→Centrality** choice provides some centrality measures directly and others may be added by user request if they can be calculated using sparse methods.

Betweenness has become very important in recent studies of large scale-free networks, and a recent algorithm (Brandes, 2000) allows efficient calculation for large networks using sparse methods. Betweenness is very useful since it is always defined even for disconnected networks, and is the same measure for both a directed graph and its transpose (out- and in- measures are the same). Closeness is not included since it is not defined unless the network is at least strongly connected. Instead the related measures of Integration (in) and Radiality (out) are included, since these are always defined for disconnected networks and can be efficiently calculated with a variant of the (Brandes , 2000) algorithm. Influence (both out- and in-) is also defined for disconnected networks and can be efficiently calculated for large networks with a recent sparse algorithm (Foster et. al., 2001).

Once **Recode→Centrality** has been chosen, a selection window opens for choice of link variable. Then another selection window opens for choice of centrality measure. Choosing one of these results in the calculation and display of a new pending node variable. (The variable is not automatically created). The new label (and eventual comment) for each choice is:

Betweenness	**Bt{...}**
Influence (Out)	**Io{...}**
Influence (In)	**Ii{...}**
Integration	**In{...}**
Radiality	**Rd{...}**

3.3.8 Identify

Identify defines a new node variable by producing a set of ID numbers that correspond to the unique values of the node attribute, and then gives each new node its own value. For this reason, Identify can only be used with integer-valued variables. Since new node IDs are created, all other node variables must be given missing data values for these new nodes. **Identify** is useful in connection with **Make Hypergraph** for link variables, which also creates new ID numbers based on attribute values, but does not create a new node variable and so does not give the new IDs any values as **Identify** does.

3.3.9 Count

Count is similar to Identify, since it can produce a set of new ID numbers that correspond to the unique values of a node variable (which must be positive integers). Whereas **Identify** gives each new node ID a value equal to its ID number **Count** gives each ID a value equal to the number of times the value appears. Thus the result of **Count** is the frequency distribution of values for a discrete-valued node variable.

Table 3.1. Partial report on strong components of link SAY as a node variable

```
MultiNet VARIABLE REPORT ON "KIDS2.NOD+KIDS2.LIN" 13/06/2003  10:33:31
NODE VARIABLE NAME: ST{SAY} <pending>
STATISTICS OF ST{SAY} <pending>      DISTRIBUTION OF ST{SAY} <pending>
  MEAN  0.97                        LABELS        VALUES    COUNTS      %age
  VAR   0.0302                      0S                 0         1      3.1%
  SDEV  0.174                       1S                 1        31     96.9%
  MIN   0
  MAX   1
  MED   1
  SIZE  32
  BINS  2
```

3.3.10 Components

If there is a path between every pair of nodes in a graph, the graph is said to be connected. If a graph is connected, it consists of a single component. A disconnected graph does not have a path between all pairs of nodes, and may consist of several components. **Recode→Components** produces a node variable with values that depend on which component each node belongs to in the network defined by the selected link variable. Choosing **Components** opens a selection window for choosing a link variable. After choosing a link variable, a selection window is opened which has two choices: **Weak (UNDIRECTED)** and **STRONG (DIRECTED)**. Weak components ignore direction (and therefore are the same components found in the **Eigenspaces** module, which symmetrizes networks as part of the eigendecomposition). Strong components do not ignore direction. Once a selection is made, the function calculates the components for the chosen link variable, and orders them according to decreasing order (number of nodes). Nodes are then assigned an integer value depending on which component they are part of, from 1 (largest) to n (smallest). Nodes belonging to a component of order 1 (trivial components or *isolates*) are assigned a value of 0, which makes it easy to re-assign them all as missing data values (not belonging to any non-trivial component). If all nodes belong to trivial components, the result is an error message. This can occur, for example, with strong components of ego-centric data.

Once components are calculated, the result is a pending variable with default label (and comment) based on the chosen link variable. For example, the weak components of link SAY are labelled **We{SAY}**, while the strong components are labelled **ST{SAY}** (see Table 3.1). If the number of components is less than a user selected limit (set by Preferences), then value labels are also created, with component number as prefix, and suffix '**W**' for weak and '**S**' for strong. Components produces values only for those nodes that take part in the network defined by the selected link variable. All other nodes are marked as missing data.

When a variable derived from **Components** has been created, it may be used with Equation, combined with other node variables, to isolate sets of nodes that belong to individual components (or all isolates), and examine differences in distributions. It may also be used in the MultiNet Analyse module in combination with other node variables to examine differences in components. The next section describes a method that creates a new link variable for each component.

The following descriptions of choices apply to link variables only.

3.3.11 Reduce MultiLinks

Reduce MultiLinks is enabled if there are any multiple links in the selected link variable. Multiple links are From->To pairs of nodes that appear more than once. This may occur if the link variable is time-dependent; e.g., I speaks to j at time t, and again at time t+1. (It may also occur as a data error, so it is worth checking to see if **Reduce MultiLinks** is enabled when you do not expect it to be.). Choosing this function opens a multiple selection window with choices:

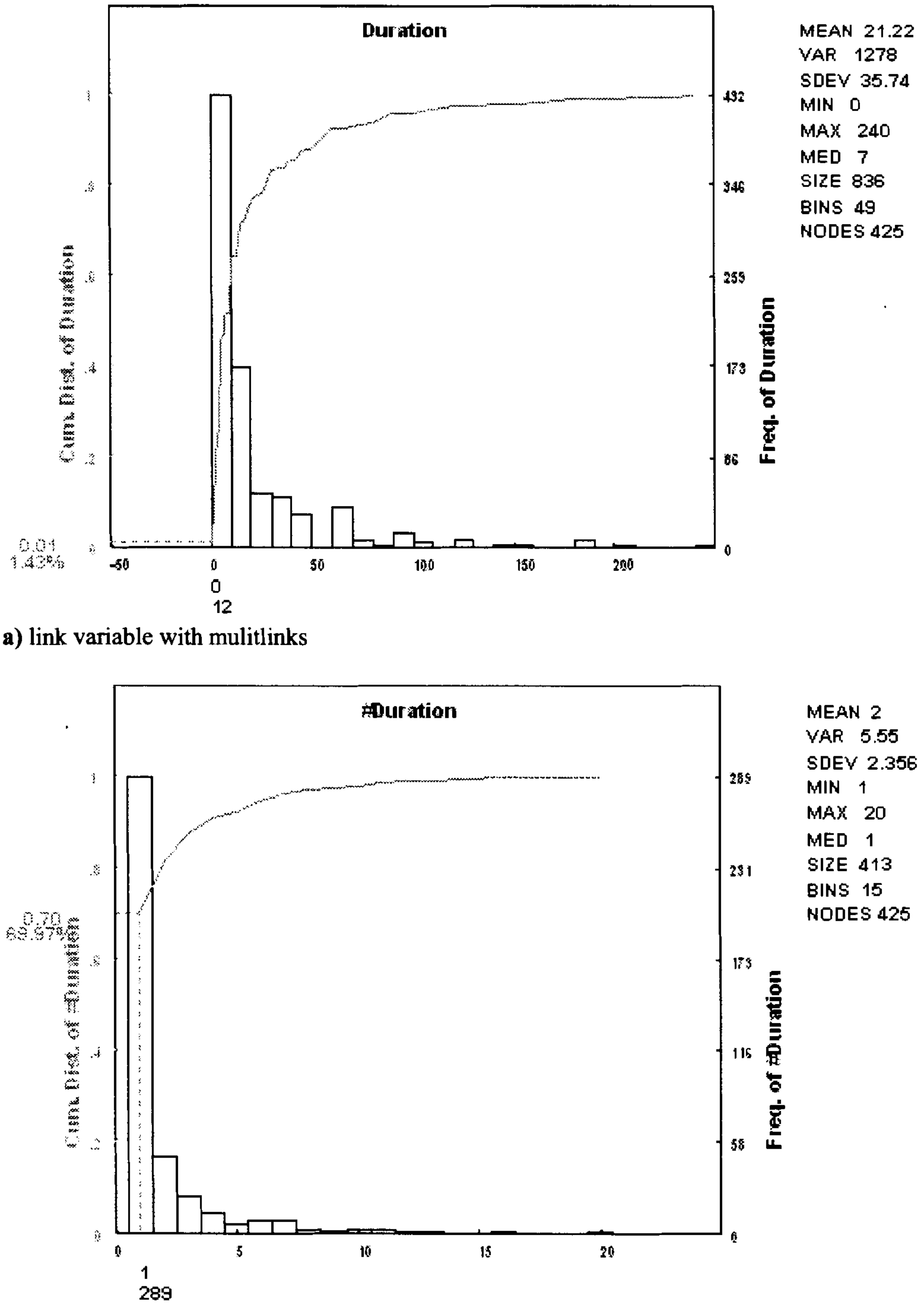
- **Number of links** COUNT the number of multiple From->To pairs
- **Amount of Linkage** SUM the link strengths of multiple From->To pairs

The second choice appears only if the link variable is non-binary, but may not make sense if the link variable is categorical (e.g., form of communication: 1=phone, 2=email,...). Once choice(s) are made, the function counts the number of multilinks and/or sums the total value of multilinks and assigns the value(s) to the last node pair of each multilink. All the other values of each unique node pair is marked as missing data. This can dramatically reduce the size of the network.

The resulting link variables are automatically created, with default prefix **#** for number of links and **$** for amount of linkage. Figures 3.9 and 3.11 show results for link variable Duration, which is length of time of each contact so calculating Amount does make sense. The automatic variable names are **#Duration** and **$Duration** with comments "**Count Duration MultiLinks**" and "**Sum Duration MultiLinks**" respectively. Each potential link variable is presented before creation with variable name and then with comment, either of which can be changed by the user in an edit window. The creation process can be cancelled at any time, though variables already created remain. If any variables are created, the last created becomes the selected variable, which is then displayed.

3.3.12 Node Constraints

Node Constraints allows the selection of subsets of a link variable based on node attributes. You may select allowed values for one or more node variables by using a logical equation to define the allowed values You initially choose whether these allowed values apply only to senders (**From**),



a) link variable with mulitlinks

b) counts of number of multilinks

Figure 3.9. a) link variable Duration has 836 multiple links which measure the length of each interaction. **b)** link variable #Duration counts the number of interactions for each of 413 unique node pairs. See also figure 3.11.

receivers (**To**), **Both** senders and receivers (logical And), or **Either** senders or receivers (logical Or). As an example, consider the KIDS2 network of boys and girls of ages 6-10. To select a sub-network of only boys of age less than 8 select **Both** and use the logical equation

(a=1)*(b<8) (where **a** represents sex, and **b** represents age).

Only links for which Both nodes satisfy these node constraints will remain non-zero. The resulting link variable is automatically named (and commented) with the constraints in [...] brackets. In this example, the name is "**SAY[B:(SEX=1)*(AGE<8)]**". Note that the new network may not be connected.

More complex sub-networks may be constructed by combining **Node Constraints** results using **Recode→Equation**. For example, to get a network consisting only of links From boys To girls, select **From** and **a=1**, then **To** and **a=2**. Then combine the two new link variables using **Recode→Equation** and the "*" (logical And) operation.

3.3.13 Transpose

Transpose is provided for convenience, and is not suitable for data which is highly directed such as ego-centric and 2-mode networks. **Transpose** simply reverses the direction of all links, keeping the values. This can produce many new node pairs, and will cause a permanent increase in the size of the dataset in memory until the new link variable(s) and all other variables derived from them are Deleted. The extra node pairs are marked as missing data for all other link variables. Thus **Missing->Zero** may produce much larger link variables, with many extra zeros

3.3.14 Symmetrize

Symmetrize is provided as a convenience, and is not suitable for data which is highly directed such as ego-centric and 2-mode networks. This function uses **Transpose** which can add many new node ID pairs to the dataset and will cause a permanent increase in the size of the dataset in memory until the new link variable(s) and all other variables derived from them are Deleted. The extra node pairs are marked as missing data for all other link variables. Thus **Missing->Zero** may produce much larger link variables, with many extra zeros. The convenience comes from not having to enter every node pair in both directions in a .LIN file when the network is known to be symmetric, and using **Symmetrize** to produce all the reciprocated node pairs.

The network is combined with its transpose in on of four ways:

- **MAX (OR)** For binary data, this reciprocates every link.
- **MIN (AND)** For binary data, this drops all non-reciprocated links.

- **SUM** which may be converted to mean by **Recode→Equation**.
- **ABS DIFF** which subtracts the network from its transpose and takes the absolute value.

The result is given label **Mx{...}**, **Mn{...}**, **Su{...}** or **Ad{...}**. If the number of unique values produced is not greater than the current maximum number of categories (default 12 or maximum 20), then value labels are produced with "**S**" appended to the actual values.

3.3.15 Make Hypergraph

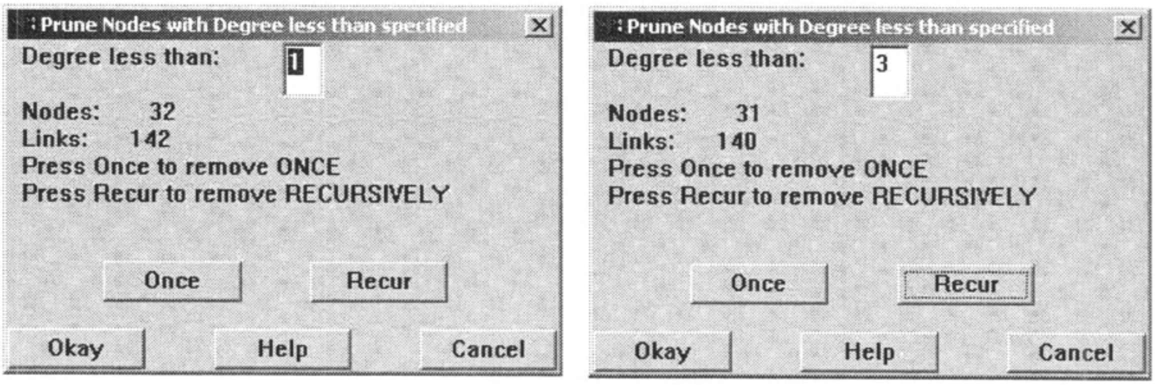
Make Hypergraph combines node attributes to produce a binary network of co-occurrences, with existing nodes as rows, and the values of the node attributes as columns. New ID numbers are created to account for the attribute values. For this reason, the node attribute values should be integer, and there should be no overlap with existing ID numbers and the two sets of attribute values. Thus this function not only creates new node pairs, it also creates new "node"s and IDs for the attribute values. Eigenspaces will treat this network as a bipartite graph, with nodes as rows and attribute values as columns. The new node ID numbers can receive values either as variables or partitions. See also the **Identify** function for Node variables.

Make Hypergraph uses Selection windows to ask for a pair of multi-valued node variables. Use **Recode→Equation** to ensure that the range of values for these two variables do not overlap with each other or existing IDs. These variables are attributes of nodes and for each node that has a pair of such attributes, **Make Hypergraph** produces new node pairs consisting of the node ID and the actual attributes. This results in a binary contingency table of co-occurrences which may then be further analysed in the Eigenspaces module to examine whether the attributes co-cluster. Sections 9.4, 9.5 and 10 contain detailed examples of this type of network analysis. The new link variable is automatically named "**NVAR1;NVAR2**", where "**NVAR1**"and "**NVAR2**"are the names of the chosen node variables in the order chosen.

The functions **Identify**, **Count**, **Transpose**, **Symmetrize** and **Make Hypergraph** can produce new nodes and node pairs. When the **Manage→Delete** command removes node or link variables, it checks whether remaining nodes or node pairs have only missing values, and if so deletes such nodes or node pairs and the corresponding missing data markers from all remaining variables. Thus the number of unique IDs may fluctuate as new variables are created by these functions and then later deleted.

3.3.16 Prune

It is often useful to remove nodes of very low degree from a network (e.g., isolates). Some networks have long paths of low-degree nodes, which result in large negative eigenvalues (since they are locally bipartite). Since the nodes are all of low degree, removing them does not generally have much effect on the overall behavior of the network, and can produce better visualizations of the more important high-degree nodes. Choosing **Prune** opens the control window shown in figure 3.10.



Prune Nodes with Degree less than specified

Degree less than: 1

Nodes: 32

Links: 142

Press Once to remove ONCE

Press Recur to remove RECURSIVELY

a) initial display

Prune Nodes with Degree less than specified

Degree less than: 3

Nodes: 31

Links: 140

Press Once to remove ONCE

Press Recur to remove RECURSIVELY

b) final display

Figure 3.10. Prune control window. Initially set to remove isolates (degree < 1). For link SAY there are 32 nodes with 142 links. After selecting degree < 3 and pressing **Recur**, one node with 2 links has been removed.

Prune will remove nodes with less than a specified number of links, equal to the total sum of out- and in-degrees. The initial default is less than 1, which removes isolates (figure 3.10a). The example uses link variable SAY, which has no isolates or nodes of degree less than 2. However there is one node with total degree of 2. Selecting degree < 3, and pressing **Once** removes that node. Removal of nodes can result in other nodes dropping below the threshold for total degree, so pressing **Once** again may remove more nodes and so on. This iteration is automated by pressing **Recur**, which repeatedly removes nodes below the threshold until there is no more change. It is possible to end up with no nodes (or links) left. Nothing happens to the selected variable until **Okay** is pressed, and **Cancel** ends **Prune** with no change. The result is a pending variable. This function works by setting all values of the link variable to 0 for all node pairs containing a node that is being removed. These links may then be declared missing with **Zero->Missing**. The new label (and comment) is **Pr{...}**.

3.3.17 No Diagonal

No Diagonal is only enabled whenever there are <u>any</u> **FROM** ->**TO** node pairs for which both **FROM** and **TO** are the same node (i.e., self-loops). Though self-loops are perfectly reasonable for some networks, for most they are not, and so if **No Diagonal** is enabled unexpectedly this may indicate a data error. The purpose of this function is to eliminate all self-loops by setting the link variable to 0 for links where both sender and receiver are the same. These links may then be declared missing with **Zero->Missing**. The pending variable has label (and comment) **Nd{...}**.

3.3.18 Components

This function is very similar to the version for node variables. The difference is that instead of assigning values to nodes depending on which component they belong to, new link variables are created, one for each component of size at least 2. The link version of **Components** proceeds as the node version, until the components (weak or strong) have been identified, and sorted into decreasing order (number of nodes). At this point, a multiple choice window is opened with link variable names constructed from the type of component, sort position, colon and selected link variable name. The comment is "Nodes:" and the order of the component. For example, link variable SAY has one strong component of order 31, and one trivial component with one node. The multiple selection list consists of all non-trivial components, in this case the single component labeled **S1:SAY** with comment "**Nodes:31**". Each potential link variable is presented before creation with variable name and then with comment, either of which can be changed by the user in an edit window. The creation process can be cancelled at any time, though variables already created remain.

Whether or not any variables are created, a new pending link variable is created showing the distribution of component orders, with label (and comment) **We{...}** or **ST{...}**. This link variable replaces the value of each link with the number of the component that the nodes at each end belong to. Links that do not belong to a non-trivial component are set to 0. These links may then be declared missing with **Zero->Missing**.

3.3.19 Hybrid variables

Node variable which are derived form link variables, and link variables which are derived from node variables are called *Hybrid variables*. There are many examples of hybrid node variables, such as variables and partitions (from Eigenspaces), components and the various degree and centrality measures. Hybrid link variables include those defined from p* fits (which may result from a blocking based on a node variable), **Node Constraints** and **Make Hypergraph**.

3.3.20 Composition of Recode functions

The **Recode** functions have been designed to allow for easy composition. That is, the result of one function can immediately become the input to another function. Any compositions are accumulated in the automatic display label (and eventual comment), so it is easy to keep track of how a variable was created. As an example of this, suppose we wish to construct a Likert-style categorical variable based on the Duration of interactions (figure 3.9) for use in cross-tabs or colouring links in module **Eigenspaces**. Duration is strongly skewed to the left and has a long tail (Figure 3.9a), suggesting that a **Lg10** transformation is reasonable for this type of interaction (Crow and Shimizu, 1988). The steps to produce the categorical variable are:

- **Select Link** Duration
- **Recode→Reduce MultiLinks** with choice **Amount of Linkage** to get the total duration of all interactions between node pairs. This automatically creates and displays the variable named **$Duration** with comment "**Sum Duration MultiLinks**"
- **Recode→Continuous→Lg10**, which takes the logarithm and produces label (and comment) **Lg{$Duration}**; and
- **Recode→Discrete→Bins**, and selecting 5 bins, produces the variable with automatic label "**5 Bi{Lg{$Duration}}**" shown in figure 3.11. Then **Manage→Labels** is used to produce the descriptive Likert-style labels describing the amount of interaction.

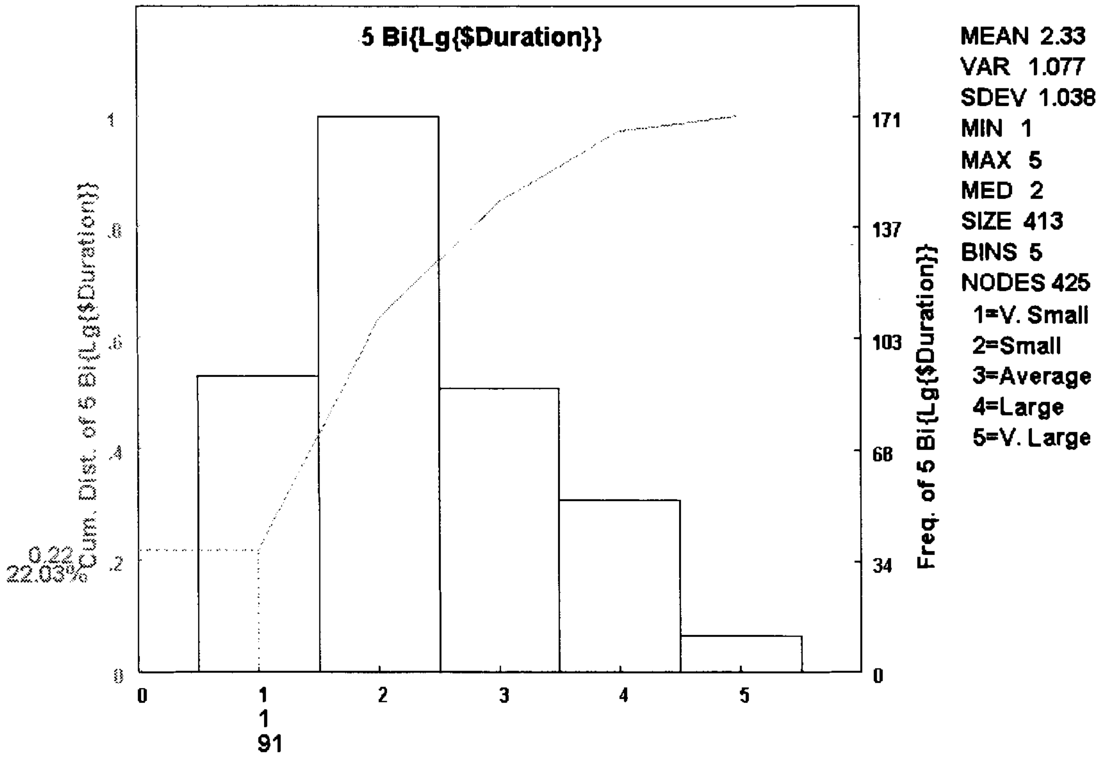


Figure 3.11. Composition of **Recode** functions on a link variable with automatic variable creation.

As another example, consider a non-binary link variable called "links". We may want to find the weak components when link strength is restricted to values above a certain threshold. The steps are:

- **Select Link** "links"
- **Recode→Equation** with equation **a>16*a**, where **a** is the symbol for links, creates a pending variable with values 0 when variable "links" has values less than 17; otherwise, the values are the same as in the variable "links". The program produces label **"E=links>16*links"** for the display of the pending variable
- **Recode→Components**, selecting **Weak**, and then selecting the largest component of size 153, which is automatically created and labeled **W1:E=links>16*links**.
- **Select** this new link variable (which will later be replaced).
- **Recode→Zero->Missing** results in the pending variable with label **Zm{W1:E=links>16*links}.**
- **Manage→Replace** then replaces **W1:E=links>16*links** with the non-zero links only as shown in figure 3.12, using the default label automatically generated.

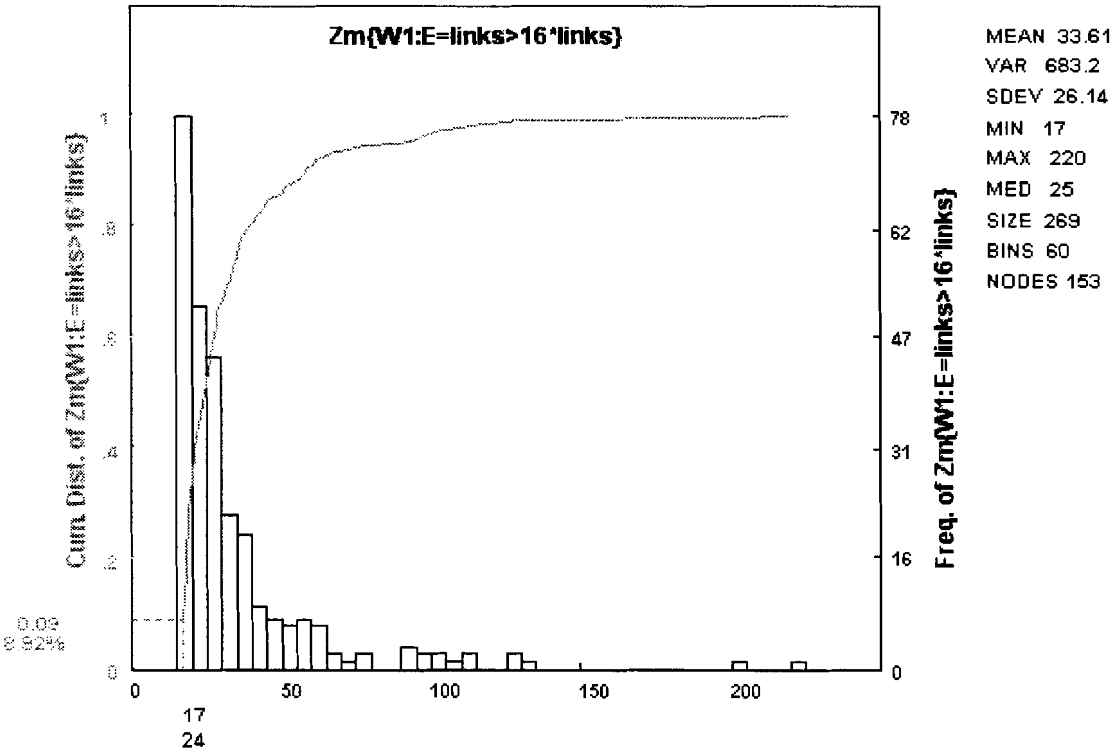


Figure 3.12. Composition of **Recode** functions on a link variable using **Replace** on temporary intermediate result.

3.3.21 Automatic creation of variables by **Recode** functions

Certain **Recode** functions automatically create variables either because they use multiple selection menus as a convenience, or because these variables may be created as a side-effect. These are:

- **Recode→Degree** for node variables presents a multiple selection window as a convenience since both out- and in- degrees (and their weighted forms for non-binary link variables) are generally of interest. The selected node variables are created in reverse order, subject to user choice, and the last variable created is then automatically selected and displayed.
- **Recode→MultiLinks** for link variables presents a multiple selection window with one choice for binary (**Number**) or two choices for non-binary link variables (**Number** and **Amount**). The selected link variables are then created in reverse order, subject to user choice, and the last variable created is then automatically selected and displayed.
- **Recode→Components** for link variables will construct a new link variable with a unique integer link value for each component. As a side-effect, new link variables with the original link values can also be created, one for each component. A multiple selection window of all non-trivial components is presented. The selected link variables are then created in reverse order, subject to user choice. The distribution of component sizes (either **We** or **ST**) is the new and pending link variable displayed.

In each case the automatic creation can be ended by pressing the Cancel button for either variable name or comment, although variables already created are not affected and will remain as variables. Creation in reverse order ensures that the first variable chosen in the multiple selection list is the last created, and therefore is the one automatically selected and displayed. All other **Recode** functions produce a pending variable, that is, a temporary variable which can be made permanent by **Manage→Create** or **Manage→Replace**.

This ends the section describing the functions available with Recode.

Report

The report for any variable includes a table of simple statistics and a table of the distribution using the same bins used in the graphic histogram. The number of bins has default value of 30, and may be set by the user in module **Preferences** to a maximum of 100. If the number of unique values of the variable is not more than this setting, all values are used in the histogram and distribution table.

For node variables, additional detail is also available. Node variables are generally much smaller than link variables, so it is practical to include a list of all the nodes and their values. In fact, there are two lists side-by-side, the first ordered by ID number (to make it easy to find a values associated with any node), the second sorted by value (to make it easy to find nodes associated with any value). If variable IDLABEL is available, it is also included in these lists. If the **Network** function has been applied, then the Report also contains statistics and distributions for the links sent From each node and for the links sent To each node. Table 3.2 is an example with node variable SEX, IDLABEL, and link variable SAY selected by **Network**.

Table 3.2. Full report for node variable SEX and link variable SAY

```
MultiNet VARIABLE REPORT ON "NODkids.CSV+LINkids.CSV" 13/06/2003  16:23:36
 NODE VARIABLE NAME: SEX

 STATISTICS OF SEX

   MEAN  1.5
   VAR   0.25
   SDEV  0.5
   MIN   1
   MAX   2
   MED   1.5
   SIZE  32
   BINS  2

 DISTRIBUTION OF SEX

 LABELS           VALUES      COUNTS     %age
 male               1            16     50.0%
 female             2            16     50.0%

 --------------------------------------------------
 REPORT OF SAY LINKS SENT FROM SEX

 STATISTICS OF SAY LINKS FROM SEX

   MEAN  1.44
   VAR   0.2468
   SDEV  0.4968
   MIN   1
   MAX   2
   MED   1
   SIZE  142
   BINS  2

DISTRIBUTION OF SAY LINKS FROM SEX
 LABELS           VALUES      COUNTS     %age
 male               1            79     55.6%
 female             2            63     44.4%
```

```
--------------------------------------------------
REPORT OF SAY LINKS SENT TO SEX

STATISTICS OF SAY LINKS TO SEX

  MEAN  1.44
  VAR   0.2468
  SDEV  0.4968
  MIN   1
  MAX   2
  MED   1
  SIZE  142
  BINS  2

DISTRIBUTION OF SAY LINKS TO SEX

LABELS          VALUES      COUNTS    %age
male              1             79    55.6%
female            2             63    44.4%

--------------------------------------------------
DETAILS OF NODE VARIABLE SEX
SORTED BY ID#                        |          SORTED BY VALUE
     ID#        ID       VALUE       |      ID#       ID       VALUE
       1 Linda             2         |        4 Jimmy             1
       2 Jemima            2         |        5 Fred              1
       3 Bertha            2         |        7 Bill              1
       4 Jimmy             1         |        8 Andrew            1
       5 Fred              1         |        9 Tom               1
       6 Rose              2         |       12 Dweevi            1
       7 Bill              1         |       15 Russel            1
       8 Andrew            1         |       16 Emil              1
       9 Tom               1         |       18 Harry             1
      10 Susan             2         |       19 Ivan              1
      11 Amber             2         |       20 Victor            1
      12 Dweevi            1         |       21 Pavel             1
      13 Moon U            2         |       23 Mark              1
      14 Katie             2         |       24 Trevor            1
      15 Russel            1         |       28 Matthe            1
      16 Emil              1         |       31 Luke              1
      17 Mary              2         |        1 Linda             2
      18 Harry             1         |        2 Jemima            2
      19 Ivan              1         |        3 Bertha            2
      20 Victor            1         |        6 Rose              2
      21 Pavel             1         |       10 Susan             2
      22 Nancy             2         |       11 Amber             2
      23 Mark              1         |       13 Moon U            2
      24 Trevor            1         |       14 Katie             2
      25 Julie             2         |       17 Mary              2
      26 Joan              2         |       22 Nancy             2
      27 Janet             2         |       25 Julie             2
      28 Matthe            1         |       26 Joan              2
      29 Lucie             2         |       27 Janet             2
      30 Irma              2         |       29 Lucie             2
      31 Luke              1         |       30 Irma              2
      32 Prisci            2         |       32 Prisci            2
```

As is the case throughout MultiNet where the **Report** menu item is available, the user may choose to **View** the textual report in an edit window, or **File** the report to the current output file. Filing produces the usual choices: **Append, Replace, Increment** or **Rename**. The user may also set **Automatic Append** in the module **Preferences**.

Graphics

Like **Report**, this menu item is common to most MultiNet modules, and acts similarly in each one. Clicking on **Graphics** produces the following choices:

- **Graphics→PostScript** reproduces the current display as a PostScript program. This is a text file which produces graphics with a PostScript interpreter (e.g., printer).
- **Graphics→Bitmap** captures the screen display as a 256-colour bitmap, which is then run-length encoded. The result is a compressed Windows .BMP file.

For more details, see Section 0: Overview and Technical appendix 0.

Explore

This menu item is included as a convenience, and is initially disabled (greyed-out). The **Explore Data** window may be closed by clicking on the 'X' in the upper right on this item. This may be helpful when using other screen-capture software. Closing the **Explore Data** window enables this menu item, so that clicking on **Explore** re-opens the **Explore Data** window.

Next

This is another menu item that appears in a number of MultiNet modules. In the **Variables** module it is enabled once a variable is selected. Clicking on **Next** replaces the current variable with the next one in the list of variables. This is useful for stepping through the a set of variables to look for interesting patterns.

Last

This is another menu item that appears in a number of MultiNet modules. In the **Variables** module it is enabled once a variable has been selected. Clicking on **Last** replaces the current variable with the previous one in the list of variables. This is useful for stepping through the variable looking for interesting patterns.

Preferences

As a convenience, two useful choices from the **Preferences** module are made available from the main menu of the Variables module. They are:

- **Number of Categories** (default 12, maximum 20). A variable is considered categorical if the number of unique values is no more than this number. Most **Recode** functions will produce automatic value labels for categorical variables.
- **Number of Bins** (default 30, maximum 100). Histograms and Report distribution tables attempt to collect the data into this many bins. This number affects the number of "bars" in the histogram display and in the Report.

These choices behave exactly the same way as in the **Preferences** module. The action takes affect the next time the display changes (by **Network**, **Select**, **Recode**, **Next** or **Last**).

Help

This is a menu item that appears in all the MultiNet modules. Clicking on **Help** opens a selection window which lists all items on the current menu bar. Selecting any of these opens a view window containing details about the menu item. **Help** is also a common button on many other temporary windows, and always provides a context-sensitive description of what the program is doing and what kinds of inputs it expects at the point the **Help** button is pressed.

3.4 References

Brandes, U. (2001), A faster algorithm for betweenness centrality, *Journal of Mathematical Sociology* **25**, 163-177

Foster, K.C.,. Muth, S.Q., Potterat, J.J and Rothenberg, R.B. (2001), A Faster Katz Status Score Algorithm, *Computational & Mathematical Organization Theory* **7**, 275–285

Freeman, L.C. (1978), Centrality in Social Networks: Conceptual Clarification, *Social Networks* **1**, 215–239.

Crow, E. L.& Shimizu, Eds.(1988).*Lognormal distributions: theory and applications*, Statistics, textbooks and monographs ; vol. **88**. M. Dekker

Valente,T.W. & Foreman, R.K. (1998) Integration and Radiality: measuring the extent of an individual's connectedness and reachability in a network, *Social Networks* **20**, 89-105

3.5 Technical appendix

3.5.1 List of errors anticipated by **Variables** module.

Generic values for Any variable, or for Node, Link and Groupings variable names represented by <Aname>, <Nname>, <Lname> and <Gname>, respectively.

`Error Text` is followed by

- Explanation
- Solution

`<Lname> IS  IN GROUPING <Gname>`

- Attempt to **Manage→Delete** link variable <Lname>, which has been defined as part of Grouping <Gname> in the **Groupings** module.
- Solution: Use **Groupings→Disband** to delete <Gname>.

- Attempt to **Manage→Replace** link variable <Name1>, which has been defined as part of Grouping <Name2> in the **Groupings** module.
- Solution: Use **Groupings→Disband** to delete <Name2> or replace a different variable.

`BLANK NAME`

- Attempt to **Manage→Create** or **Manage→Replace** variable using a name consisting of all blanks.
- Solution: Use helpful, descriptive names for variables (and comments!)

`CANNOT DELETE ALL NODES!` (or `LINKS`)

- Sensible precaution to prevent mistakes in **Manage→Delete**, which starts with a multiple selection window that includes all variables.
- Solution: don't attempt to delete all variables.

<Name>`: NAME IN USE`

- Attempt to **Manage→Create** or **Manage→Replace** with a name already in use. Node variable names must all be distinct, as must Link and Grouping names. However, the same name can be used for a Node, Link and Grouping variable.

- Solution: Choose a different name.

- **Recode** is attempting to automatically create a name which is in use.
- Solution: Choose a different name.

DO NOT ADD OR DELETE ROWS

- In **Manage→Labels** editor, the number of categories changed by deleting a row.
- Solution: Do not change number of rows.

DO NOT CHANGE VALUES, ONLY LABELS

- In **Manage→Labels** editor, attempt to change items in the values columns.
- Solution: Change only items in the labels columns.

NO <Lname> LINKS FROM (or TO) <Nname>

- **Select Node** <Nname> followed by **Network→From**, and link variable <Lname> chosen. In the <Lname> network, there are no links From node variable <Nname>. This may occur in a directed (e.g.,ego-centric) network where <Nname> describes an attribute only of the nominees. Similarly, selecting **Network→To** causes this error when the node variable describes attributes only of the nominators.
- Solution: Check the node variables and Link variables for implied direction.
- This error may also occur when a symmetric link variable is defined only for the forward half of the ID pairs for convenience.

  Solution: Use **Symmetrize** to generate the missing half of the symmetric network.

<Number> IS NOT A POSITIVE INTEGER

- In **Recode→Discrete→Quantiles** or **Bins**, the number entered into edit window asking for number of Quantiles or bins is not an integer.
- Solution: Only integer values allowed.

<Number> IS TOO MANY!
MAXIMUM IS <Max>

- In **Recode→Discrete→Quantiles** or **Bins**, the number entered into edit window asking for number of Quantiles or bins is larger than the number specified in **Preferences→Ranges → Number of Categories**

- In **Recode→Discrete→User**, the number of categories created is larger than the number specified in .**Preferences→Ranges → Number of Categories.**
- Solution: Reset maximum in **Preferences→Ranges → Number of Categories** or choose number of categories no larger than the current maximum.

WARNING: LESS THAN 5% RECIPROCATION!

<Lname> MAY BE EGO-CENTRIC OR 2-MODE

- Warning before applying **Recode→Transpose** or **Recode→Symmetrize** to a link variable with very few reciprocated links.
- Solution: Ensure link data should be considered as symmetric before proceeding.

<Lname> ALREADY SYMMETRIC

- Attempt to apply **Recode→Symmetrize** to a link variable which already has all links reciprocated.
- Solution: No need to apply **Recode→Symmetrize** to this link variable.

<Lname>: ALL LINKS ZERO

- Attempt to apply **Recode→Symmetrize →Absolute Difference** to a link variable removes all links.
- Solution: Do not use Absolute difference with this link variable.

FIRST variable MUST be different from SECOND variable

- Attempt to apply **Recode→Make Hypergraph** to a pair of identical node variables
- Solution: Choose two different node variables.

<Nname1>;<Nname2>: NO NODES IN COMMON

- Attempt to apply **Recode→Make Hypergraph** to a pair of node variables that have no nodes in common: none of the nodes with attribute <Nname1> also have attribute <Nname2>, so the matrix of co-occurrences is empty.
- Solution: **Recode→Make Hypergraph** cannot be used with this pair of node variables.

<Nname> IS NOT POSITIVE INTEGERS

- Attempt to apply **Recode→Identify, Recode→Count** or **Recode→Make Hypergraph** to a node variable which has values that are not positive integers.

- Solution: Use only node variables with positive integer values with these functions.

3.5.2 Time and space efficiency of Recode functions

Because of the size of network data, two restrictions determine the functions available under **Recode** :

- The space required must satisfy the rules of sparse methods. That is, the size of the data must remain as a small fraction of N^2, where N is the number of nodes in the network
- The time required must be at most N(logN).

a) Space

There are no **Recode** . functions that violate sparsity restrictions. Some functions can produce results that require much larger amounts of storage. For example, any function that replaces binary data with floating point data would require 64 times as much memory to hold the result, but this is still only a constant multiplier. If this is part of a larger calculation that evaluates to a more compact (e.g, .binary) result, the final data would be stored in the more compact form. In general, intermediate results may take larger amounts of memory but the memory will eventually be released, and the final results stored in the most efficient form. For example, the following equation combines two binary link variables with a logical expression to select only those sums above a certain threshold:

(a+b)>2

The **(a+b)** expression produces an intermediate result which must be stored as 4-byte integer, but the logical **>2** expression results in data that can be stored as bits. Note that this expression is equivalent to **a*b**, which does not require 4-byte intermediate storage. Thinking in terms of logical equations can often simplify such equations as well as avoiding large intermediate results.

For link variables **Symmetrize**, **Reduce MultiLinks** and **Components** can produce results that require more storage than the original link variable. **Components** replaces binary links with integer values, one for each non-trivial component **Reduce MultiLinks** summarizes all links between the same ID pairs into a link value for one pair and sets the others to 0. Some space can be saved for this result by using **Zero->Missing**, since 0 link values are essentially "missing" links., and these are stored as bit values of 0. This method can also be used with **Node Constraints**, **Prune** and **No Diag**, which set selected link values to 0, but this is only worthwhile if the original link was non-binary. The **Transpose** and **Symmetrize** functions can double the number of node ID pairs, but this does not violate the sparsity restriction. Since the functions **Transpose**, **Symmetrize** and **Make Hypergraph** add new ID pairs, all link variables must have missing data added for the new pairs. This can mean that **Missing->Zero** can add many non-zero links to non-binary data, increasing the storage requirements.

b) Time

Many of the **Recode** functions are dominated by sort procedures, which are N(logN) in time, but no functions require more time than this. For example, Out-Degree uses a sort to put From IDs in increasing order, then counts the numbers (or weights) for each ID. In-Degree does the same for To ID. Betweenness, Radiality and Integration Centrality use the Brandes algorithm, which is N(logN). Influence uses the iterative algorithm described in Foster, et al (2002) which has an upper bound on number of iterations based on network diameter; in practice convergence is much faster than this and appears to be nearly linear in network size.

4. The Groupings Module

4.1 Introduction

The **Analyse** module described network crosstabs and **ANOVA**, where one or more of the discrete/categorical variables could be **FROM** Node, **TO** Node or Link. This section describes another type of network analysis where one categorical variable is a set of *proportional* link variables (called a *grouping*). The link variables must have discrete/categorical values, and in a grouping each link variable becomes a category. This section will demonstrate the usefulness of this type of network analysis, as well as how to manage (create, delete, invert and recode) the groupings Examples of crosstabs and **ANOVA** analysis using groupings will be presented using the 301 dataset.

Groupings are used only with network analysis. Although the interpretations are different, the displays and reports are very similar to those described in the Analyse section, which should be referred to for descriptions of these.

4.2. Purpose, content and proportional Link variables

When people interact, they generally do so in a variety of ways. They may use a variety of channels of communication: telephone, email, fax, or face-to-face. Alternatively, people may have different reasons for the interaction which are generally directional since the people involved may each have their own understanding of the *purpose* of interaction. Examples include:

- to get information
- to get advice
- to get something done
- to give information
- to give emotional support

In addition to *purpose* of interaction, each contact may include discussion of a variety of subject areas which constitute the *content* of the communication. Examples are:

- school
- sports
- family
- business/work

Purpose and content have no obvious implied direction since the content may be mutually agreed upon, but it's likely that different people will have different perceptions of the purpose and content of the interaction, so it makes sense to say that links come **FROM** the person who describes them to the researcher. Except in very formal situations, the purpose and content of contacts between people are generally *multiplex*: there are a number of purposes and a number of content areas. If we wish to study relationships between what people talk about and why they talk, we need to collect this information from the people involved. Ideally, we could ask each person to produce a list of percentages describing the *proportion* of each of a set of purposes (or contents) in a contact; e.g., 47% to get something done, 22% to get advice and 31% to get information. A glance at these numbers shows that it is unlikely that they could be collected with such precision. This is not just because the contact was described through later recollection rather than while it was happening but also since it is unlikely that people will try to make sure everything adds up to 100%. Also people tend to "round" any estimates (to 25%, 50%, etc.). This suggests that such data could be collected in a more approximate way, less likely to produce false precision.

The *Likert scale* is a standard method that is often used to collect subjective information. It is categorical, and generally runs from 1=strongly disagree through 3=neutral to 5=strongly agree. Much social data has been collected in a similar way, with people describing in a rough, subjective way the *proportions* of each of a set of purposes or contents involved in a contact. A typical scale for such a description (with approximate percentages) is:

0	None	(0%)
1	Little	(1-20%)
2	Some	(21-40%)
3	Half	(41-60%)
4	Most	(61-80%)
5	All	(81-100%)

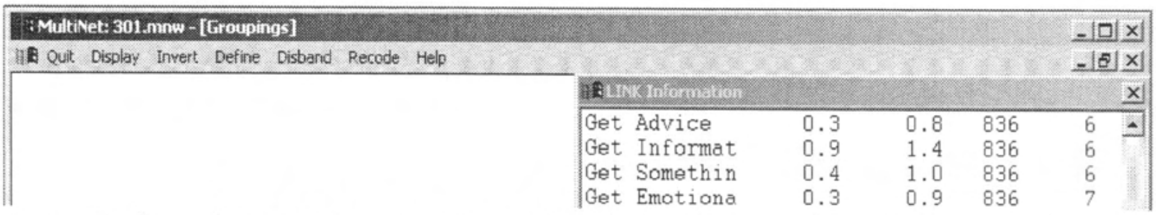
Obviously, "None" is a special case: for the sparse networks associated with social contacts the most common "interaction" is no contact at all. These categorical "proportions" may be assigned to a set of Link variables which describe different purposes such as "Get information", "Get Advice" and "Get Something Done" or to a set of Link variables which describe content such as "School", "Work" and "Family" or a set of Link variables describing the communication channel as "face-to-face", "phone", "email", etc. In collecting such descriptions, we accept that the categories are subjective and approximate and we do not insist that the numbers "add up" (e.g., all three content areas above could receive 3=”Half” or even 0=”None”). Such data can be used to describe each of many contacts between the same pair of people as well as multiple contacts between many other pairs of people. Once such data has been collected, it is natural to look for relationships between purpose and content over all the node pairs. We may also be interested in relations between these proportional variables and other link variables and node attributes. This requires treating a set of purposes or contents together, which is exactly what a *grouping* does. A grouping is a set of Link variables which describes proportions of purposes or proportions of contents for each of many contacts. The "proportions" may be precise, or they may be an approximate categorical scale as described above.

The Groupings module of MultiNet allows collecting proportional Link variables into groupings using **Define**. In network analysis the program treats grouped link data as if the categories represent constant ranges of proportion as suggested by the percentages given above. For data actually collected this way, and with the assumption that the proportions "add up" to 100%, the calculation of overall proportions based on totals over the categories is crude, but straightforward. As discussed above this is generally not the case for real data, so that results should be interpreted with care. This is a problem of data collection, not MultiNet. However, it is the case that with the coding used above, larger numbers are associated with higher proportions. Other codings may reverse this (e.g., NONE = 5 and ALL = 1) so we need a method to convert categorical descriptions so that larger numbers are associated with higher proportions. The Groupings module provides **Recode** for this purpose. Examples showing how to do this will be given later.

When a grouping is a variable in network **XTABS** or **ANOVA**, each link variable is considered as a separate category. For **ANOVA** or **XTABS** analysis based on the number of links, only the number of links are counted (Figure 4.8 and Table 4.1). For **XTABS** analysis based on the amount of linkage, each link is weighted by its proportional value, so the resulting percentages in the table and panigram depend on the scale values (Figure 4.9 and Table 4.2). Ideally, scale values should start at 0 and increase by a constant amount for easy interpretation of proportions as in all the examples given below.

4.3 The **Groupings** menu bar

The **Groupings** module is used to work with the grouped link variables described above. Groupings may be defined, undefined, examined, inverted and recoded. The items available in the **Groupings** module provide these methods (Figure 4.1).



```
Get Advice      0.3     0.8   836     6
Get Informat    0.9     1.4   836     6
Get Somethin    0.4     1.0   836     6
Get Emotiona    0.3     0.9   836     7
```

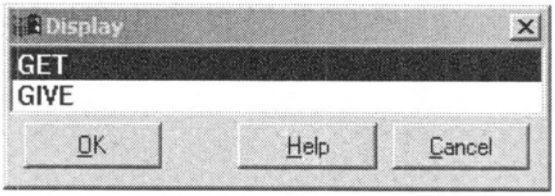
Figure 4.1. Groupings menu bar. The Link variables display window has be scrolled down to show some of the variables that measure purpose of interaction.

Quit

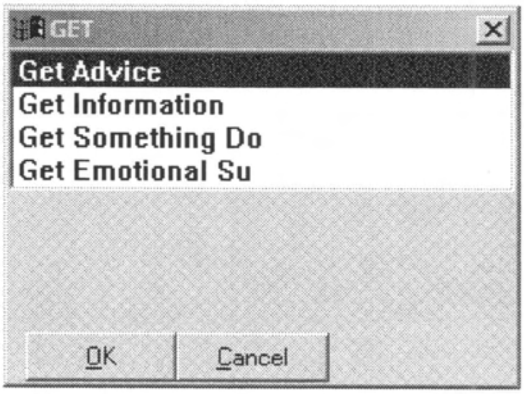
Exit from the **Groupings** Module back to the Main MultiNet menu.

Display

Display is enabled only if at least one grouping has been defined. If any groupings are defined, clicking **Display** opens a selection window showing a list of all defined groupings (Figure 4.2a). Selecting one of these by the ususal Windows methods opens a display window (Figure 4.2b) showing all the Link variables that are contained in this grouping.



a) Selection window for **Groupings**.

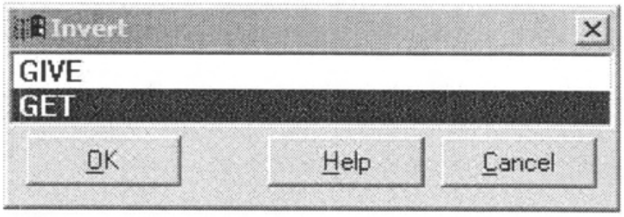


b) **Display** window of Link variables in GET

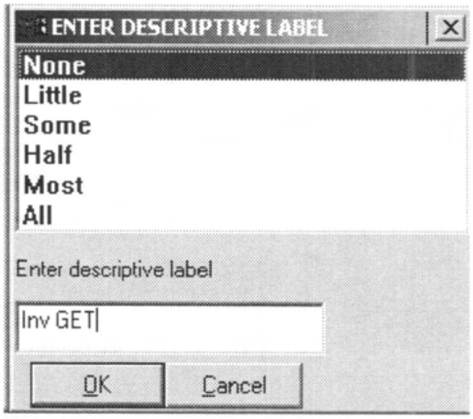
Figure 4.2. Display selection and display windows.

Invert

Invert is enabled only if at least one grouping has been defined. **Invert** is used to exchange between a grouping of Link variables and the values taken by these variables. For example, if a grouping contains two variables called "TALK"'and "LISTEN", each with the values "little", "some" and "much", a new grouping is formed with three variables called "little", "some" and "much", and with values "TALK"=1 and "LISTEN"=2. All the variables in a grouping must be categorical and share the same categories for **Invert** to succeed. MultiNet will not invert a grouping unless all the



a) **Invert** selection window. "GET" is selected.



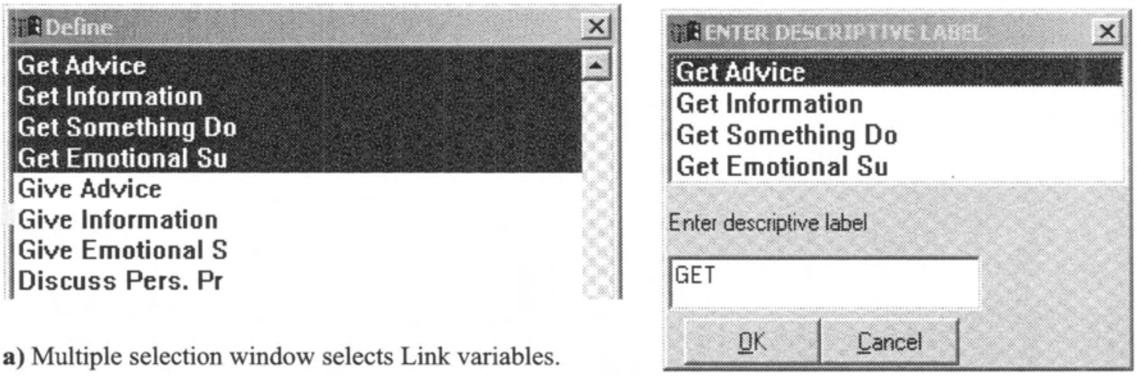
Enter descriptive label

Inv GET

b) **Invert** confirmation window shows value labels from links. Grouping name "Inv GET" has just been entered.

Figure 4.3. Invert selection and confirmation windows. "Cancel" in either window ends **Invert**.without defining a grouping.

value labels (defined in the Input LINK file) are the same. If there are no value labels, the actual numeric values are used as variable names. Figure 4.3 shows the steps involved in **Invert**-ing the Grouping "GET" (defined below).



Enter descriptive label: GET

a) Multiple selection window selects Link variables.

b) Confirmation window lists Link variables just selected. Grouping name. "GET" has just been entered.

Figure 4.4. Define selection and confirmation windows. "Cancel" in either window ends **Define** without creating a new grouping.

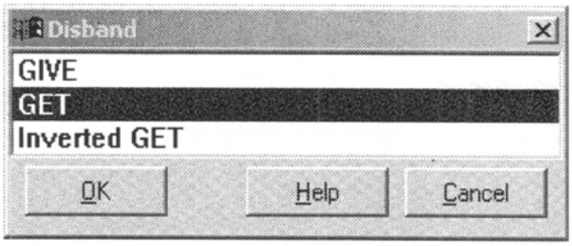
Define

Define is always enabled and is used to select the set of Link variables that are part of a grouping, and to give the grouping a name. In Figure 4.4 shows the steps involved in **Define**-ing the grouping "GET". In Figure 4.4a, four link variables are chosen from a multi-selection list. Figure 4.4b shows a confirmation window, which lists the link variables just chosen and requests a name for the grouping.

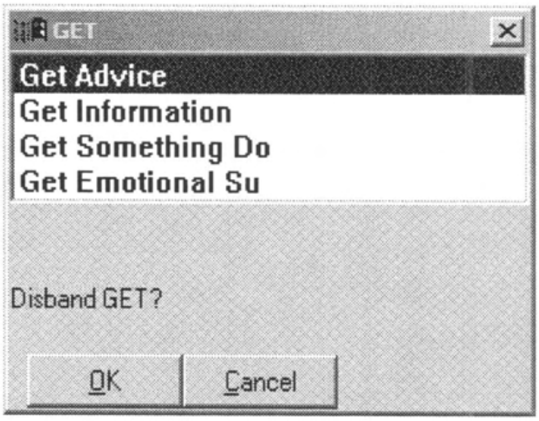
Invert and **Define** may be interrupted without the creation of a new grouping by clicking "Cancel" in either of the selection or confirmation windows. Both generate an error message if a name has not been entered for the grouping. **Define** also checks whether exactly the same set of links is already used in the definition of a grouping, and if so will generate a warning message. Error messages will also be generated in the Variables module if any attempt is made to **Delete** or **Replace** a Link variable that is part of a grouping (or Inverted grouping). A grouping must be **Disband**-ed before the Link variables that belong to it can be changed.

Disband

Disband is enabled only if at least one grouping has been defined. **Disband** is used to remove (undefine) a grouping. The link variables that are collected as the grouping are not affected, except that they may now be **Deleted** or **Replaced.** Figure 4.5 shows grouping GET selected and about to be disbanded. "Inverted GET", which was created by Inverting GET (Figure 4.3) will not be affected. Clicking "Cancel" in either window ends **Disband** without any changes.



a) **Disband** selection window



Disband GET?

b) **Disband** confirmation window. Selecting "OK" Disbands grouping "GET".

Figure 4.5. Disband selection and confirmation windows. "Cancel" in either window ends **Disband** without any changes.

Recode

Recode is enabled only if at least one grouping has been defined. **Recode** is used to make a simple linear transformation on the vales of the Link variables in the grouping. For example, Link variables in a grouping may have been coded as follows:

1	ALL
3	SOME
5	NONE

Because we wish the highest proportion (ALL) category to have the largest contribution, and the category NONE to have no contribution, we would like the category values to be:

2	ALL
1	SOME
0	NONE

While this can be done for each Link variable in the Variables module, it is more convenient to do this all at once for all the Link variables in a grouping. **Recode** requests two numeric values:

- a multiplicative constant (Figure 4.6a) with default 1.
- an additive constant (Figure 4.6b) with default 0

Whenever a grouping is used in an analysis, all the categorical values of every link variable that make up the grouping are transformed as follows:

NEW_VALUE = MULT_CONST(ADD_CONST+Link category)

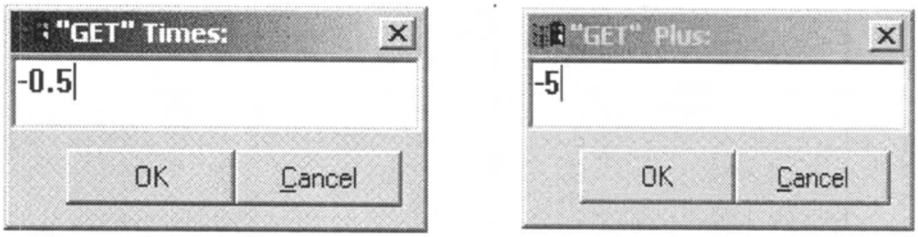
In the example above, if we wish ALL to be coded as 2 and NONE to be coded as 0, we would use an additive constant of -5, and a multiplicative constant of -0.5 (as in Figure 4.6). Then

(5 = NONE -5)(-0.5) -> 0 = NEW_NONE

(3 = SOME -5)(-0.5) -> 1 = NEW_SOME

(1 = ALL -5)(-0.5) -> 2 = NEW_ALL

As long as both the old and new category scales are equally spaced (e.g., 0,2,4,6 or 9,5,1), this linear transformation is always possible. Note that each grouping can have a different pair of **Recode** constants, and the default values of 0 for additive and 1 for multiplicative cause no change in category values.



"GET" Times:

-0.5

"GET" Plus:

-5

a) **Recode** text window for Multiplicative constant

b) **Recode** text window for Additive constant

Figure 4.6. Recode input windows to convert 1,3,5 to 2,1,0

Help

This is a menu item that appears in all the MultiNet modules. Left-click on **Help** opens a selection window which lists all items on the current menu bar. Selecting any of these opens a text window containing details about the menu item. **Help** is also a common button on many other temporary windows, and always provides a context-sensitive description of what the program is doing and what kinds of inputs it expects at the point the **Help** button is pressed.

4.4 Using groupings in the Analyse module

It is easiest to understand and interpret the results of network analyses using groupings by considering some actual cases. Here are some examples of the kinds of questions that may be examined by including a grouping as a variable:

- a) How do the relationships people have with different professions differ in the purpose of interaction?
- b) How do the purposes of men differ from those of women for getting and receiving different types of information and support?

- c) How do the responses of men and women differ for different types of questions?

Each of these questions will be demonstrated by examples taken from the 301 dataset.

4.4.1 Purpose by formal role

How do the relationships people have with different professions differ in the purpose of interaction? For this example, we will consider the set of purposes in the GET grouping (Figure 4.5b) and the set of contacts **TO** the Node variable "Occupation". To perform this analysis, choose the grouping GET as either the first or second variable in network **XTABS**. For convenience in showing the reports, we will choose GET (4 values) as the first variable, and **TO** Occupation (6 cases) as the second variable.

Pressing '**GO**' after these selections presents only the 'Link' (not 'Equation') selection, since only a single link variable may be used as a multiplier, and this should be a link variable that encodes frequency or importance. An extra choice of 'NONE' is also available, meaning that no multiplication is performed (Figure 4.7). We will first choose 'NONE' (no multiplier) and then compare the results using Duration (time in minutes of each contact) as a multiplier.

The panigram in Figure 4.8 and the counts in Table 4.1 show the proportions of the number of contacts for each purpose, but do not include the proportional values for each link variable in the grouping. The relationship is weak, with Cramer's phi = 0.08, and not significant with Chi-squared = 12.161 for 15 degrees of freedom.

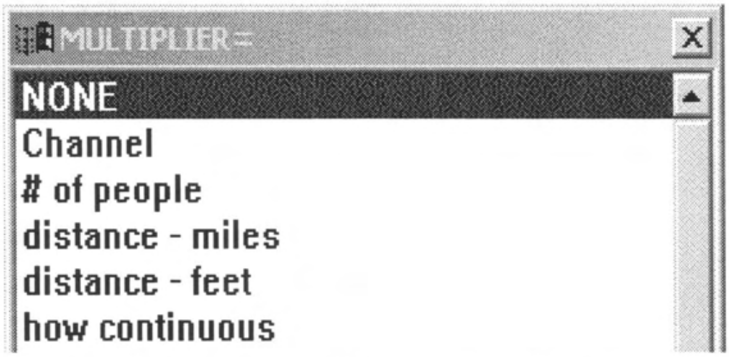
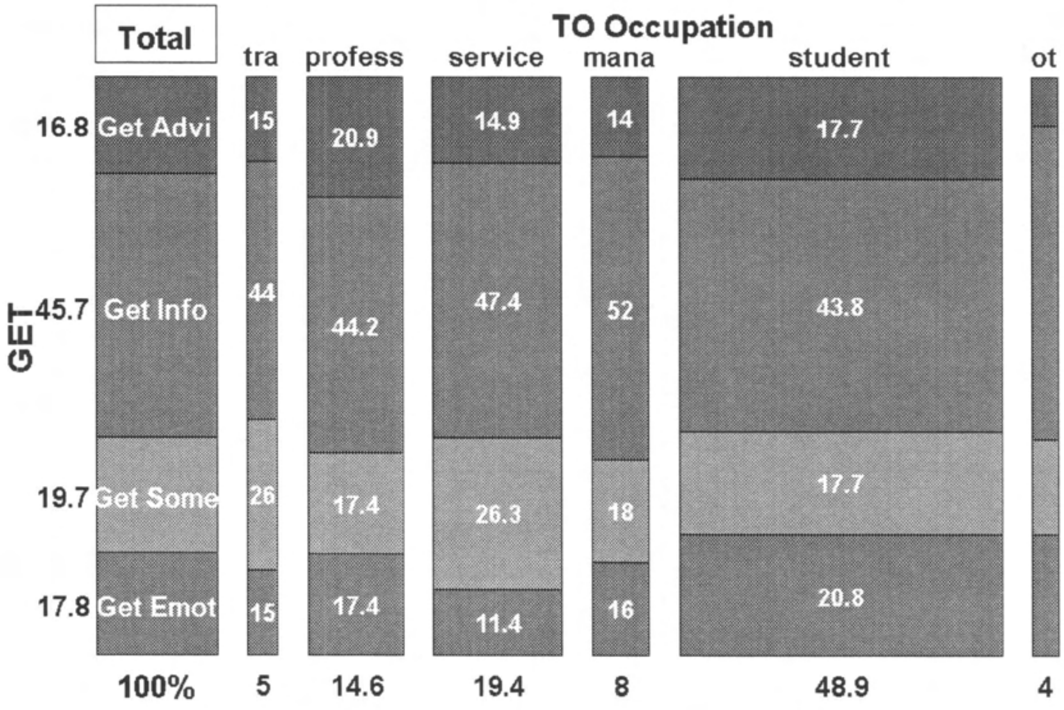


Figure 4.7. Selection window with choice of Multiplier for network **XTABS**, showing the first few Links.

Table 4.1. Number of links **TO** Occupation for purposes GET does not show a relationship.

```
The following tables deal with # OF LINKS not AMOUNT OF LINKAGE.
 COUNT
 ROW %                    ROWS = GET
 COL %                    COLS = TO Occupation
                  professi         manageri         other
            trade          service         student          TOTAL
 ---------------------------------------------------------------
         |      4      18      17       7      51       2      99
 Get Advi|  4.04%  18.18%  17.17%   7.07%  51.52%   2.02%  16.81%
         | 14.81%  20.93%  14.91%  14.0 %  17.71%   8.33%
         |
         |     12      38      54      26     126      13     269
 Get Info|  4.46%  14.13%  20.07%   9.67%  46.84%   4.83%  45.67%
         | 44.44%  44.19%  47.37%  52.0 %  43.75%  54.17%
         |
         |      7      15      30       9      51       4     116
 Get Some|  6.03%  12.93%  25.86%   7.76%  43.97%   3.45%  19.69%
         | 25.93%  17.44%  26.32%  18.0 %  17.71%  16.67%
         |
         |      4      15      13       8      60       5     105
 Get Emot|  3.81%  14.29%  12.38%   7.62%  57.14%   4.76%  17.83%
         | 14.81%  17.44%  11.4 %  16.0 %  20.83%  20.83%
         |
         |     27      86     114      50     288      24     589
   TOTAL |  4.58%  14.6 %  19.35%   8.49%  48.9 %   4.07%

CHI-SQUARE =  12.261     DF = 15   P > 0.50      Cramer's Phi= 0.08
```

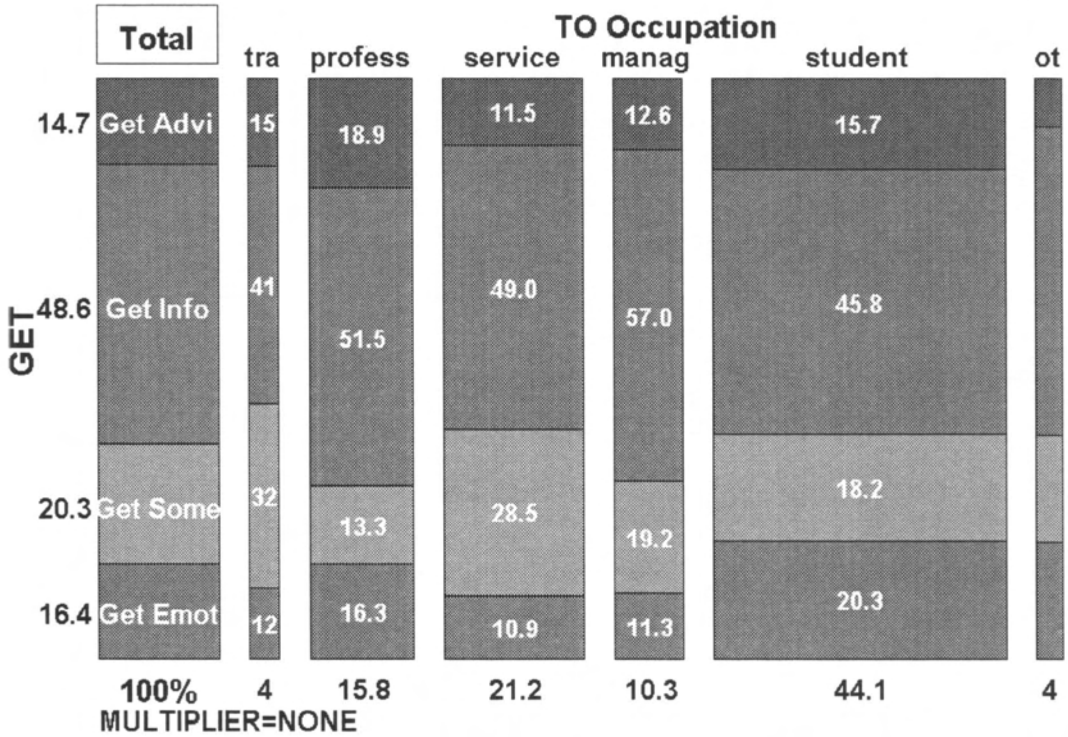


44.4% of the links to members of 'trade' are 'Get Information'

Figure 4.8. Panigram showing proportion of links **TO** each Occupation for each purpose in GET

Table 4.2. Amount of Interaction **TO** Occupation for purposes GET <u>does</u> show a relationship

```
The following tables deal with AMOUNT OF LINKAGE not # OF LINKS.
 MULTIPLIER=NONE
 ROW %                       ROWS = GET
 COL %                       COLS = TO Occupation
                   professi          manageri          other
            trade           service           student           TOTAL
          ---------------------------------------------------------------
          |     10      44      36      19     102       5     216
 Get Advi|   4.63%  20.37%  16.67%   8.8 %  47.22%   2.31%  14.68%
          |  15.15%  18.88%  11.54%  12.58%  15.72%   8.33%
          |
          |     27     120     153      86     297      32     715
 Get Info|   3.78%  16.78%  21.4 %  12.03%  41.54%   4.48%  48.61%
          |  40.91%  51.5 %  49.04%  56.95%  45.76%  53.33%
          |
          |     21      31      89      29     118      11     299
 Get Some|   7.02%  10.37%  29.77%   9.7 %  39.46%   3.68%  20.33%
          |  31.82%  13.3 %  28.53%  19.21%  18.18%  18.33%
          |
          |      8      38      34      17     132      12     241
 Get Emot|   3.32%  15.77%  14.11%   7.05%  54.77%   4.98%  16.38%
          |  12.12%  16.31%  10.9 %  11.26%  20.34%  20.0 %
          |
          |     66     233     312     151     649      60    1471
   TOTAL |   4.49%  15.84%  21.21%  10.27%  44.12%   4.08%

 CHI-SQUARE =  49.723     DF = 15     P < 0.01      Cramer's Phi= 0.11
```

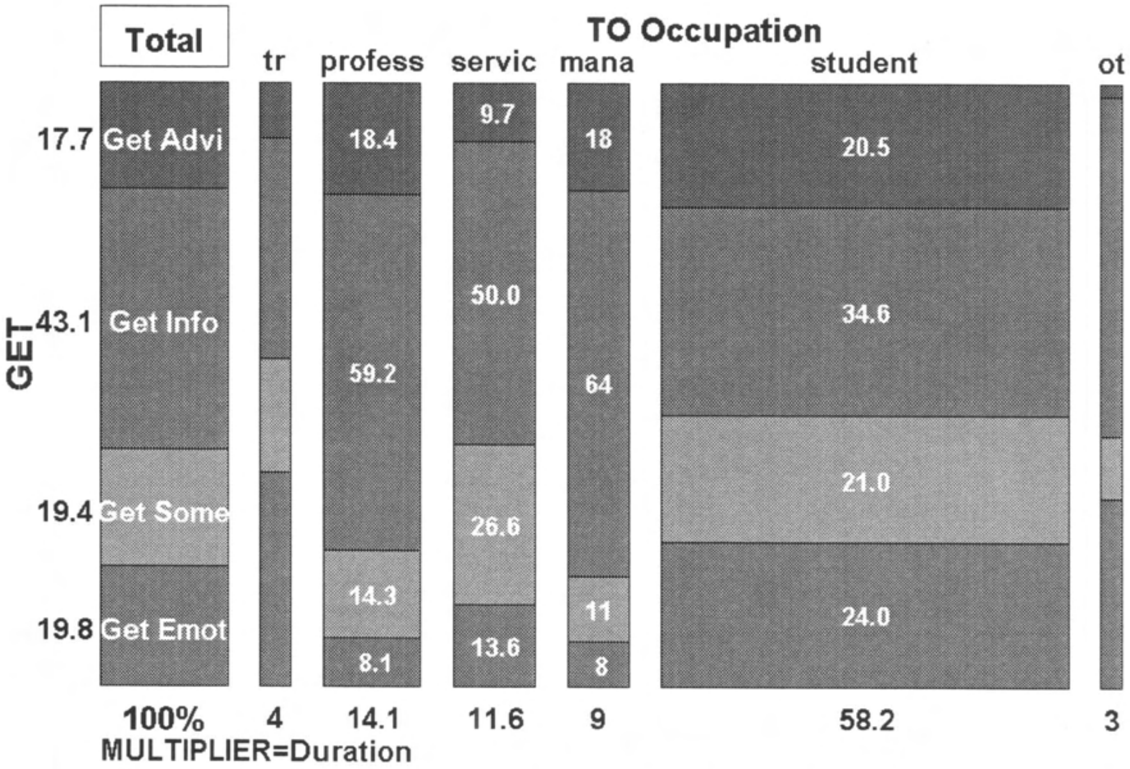


40.9% of the interactions to members of 'trade' are 'Get Information'

Figure 4.9. Panigram showing Amount of linkage **TO** Occupation for purposes GET

Table 4.3. Average and standard deviation of Likert scale values for each Link in grouping GET **TO** each Occupation.

```
Mean strength of links from row I to col j
 MULTIPLIER=NONE              ROWS = GET
 Mean                         COLS = TO Occupation
 Std. Dev.            professi            manageri            other
                trade            service             student
           --------------------------------------------------------
 Get Advi|       2.50     2.44     2.12     2.71     2.00     2.50
         |       1.12     0.96     0.96     1.39     0.91     0.50
         |
 Get Info|       2.25     3.16     2.83     3.31     2.36     2.46
         |       1.53     1.29     1.26     1.17     0.94     0.50
         |
 Get Some|       3.00     2.07     2.97     3.22     2.31     2.75
         |       1.77     0.44     1.14     1.13     0.94     1.30
         |
 Get Emot|       2.00     2.53     2.62     2.13     2.20     2.40
         |       0.71     1.02     0.84     0.78     1.01     1.36
```



59.2% of the interactions to members of 'professional' are 'Get Information'

Figure 4.10. Amount of linkage with multiplier Duration **TO** Occupation for purposes GET

Figure 4.8 and Table 4.1 present the results only for the number of contacts, and do not include the values which describe the "proportions" of the purposes for each contact as coded in each proportional link variable in the grouping. Clicking on **Amount** produces Figure 4.9, while the

Table 4.4. Amount of Interaction with multiplier Duration **TO** Occupation for purposes GET.

```
MULTIPLIER=Duration
ROW %                    ROWS = GET
COL %                    COLS = TO Occupation
                  professi        manageri          other
           trade          service          student           TOTAL
    ------------------------------------------------------------
          |   120      779      338      470     3581      20     5308
Get Advi|   2.26%   14.68%    6.37%    8.85%   67.46%   0.38%   17.72%
          |   9.26%   18.62%    9.74%   17.9 %   20.5 %   2.21%
          |
          |   474     2502     1737     1672     6037     510    12932
Get Info|   3.67%   19.35%   13.43%   12.93%   46.68%   3.94%   43.18%
          |  36.57%   59.8 %   50.04%   63.67%   34.57%  56.23%
          |
          |   242      606      925      286     3663      94     5816
Get Some|   4.16%   10.42%   15.9 %    4.92%   62.98%   1.62%   19.42%
          |  18.67%   14.48%   26.65%   10.89%   20.97%  10.36%
          |
          |   460      297      471      198     4184     283     5893
Get Emot|   7.81%    5.04%    7.99%    3.36%   71.0 %   4.8 %   19.68%
          |  35.49%    7.1 %   13.57%    7.54%   23.96%  31.2 %
          |
          |  1296     4184     3471     2626    17465     907    29949
   TOTAL  |   4.33%   13.97%   11.59%    8.77%   58.32%   3.03%

CHI-SQUARE =  2580.471     DF = 15     P < 0.01    Cramer's Phi= 0.17
```

corresponding Table 4.2 is further down in the report. This time the results are different: there is a significant relationship between amount of contact for various purposes and the occupation of the receiver. The mean and standard deviation of the proportions for each link variable in the grouping **TO** each occupation is also part of the report and is shown in Table 4.3. Notice that the helpful description (below the line) in Figure 4.8 refers to "number of links", while that in Figure 4.9 refers to "amount of interaction".

There is yet one more aspect of each interaction that can be taken into account: the total amount of time of each contact, which is measured by the Link variable Duration. Using Duration as a multiplier (Figure 4.7) will not change the display shown in Figure 4.8 or the counts in Table 4.1, but it will affect the display, crosstab table and strength table when Amount is clicked. The first two are shown in Figure 4.10 and Table 4.4. The relationship between purpose and Occupation is stronger when Duration is used as a multiplier, as measured by both Chi-squared and Cramer's phi.

The effect of Duration raises the question: are different amounts of time taken for the different purposes? This type of question can be answered by network **ANOVA**, and to make sure we count exactly the same links (since there may be zero Durations and missing data), we perform a 3 variable **ANOVA**, with dependent (Link) variable: Duration, and Independent variable GET, stacked on **TO** Occupation. Network **ANOVA** with groupings is currently somewhat limited, since a grouping cannot be the third variable (so there is no **CORREL** with groupings), and only the number of links, not the amount of interaction, is calculated and displayed. This is enough to answer the question, since we only want to know how long each contact lasted, not what proportion it was. The results are shown in Figure 4.11 and Table 4.5. Though "Get advice" and "Get emotional support" take more time, the spread of values (confidence intervals) is so large that the effect is not significant, and this is the case for all occupations (not shown, but seen by using **Next** on the **Report**).

Table 4.5. Network **ANOVA** on **Dependent:** Duration and **Independent:** GET **stack**ed on **TO** Occupation shows small significance for any value (plane)

```
MULTIPLIER=NONE
   100.00% of PLANES: TO Occupation ( # 1-6)
 Independent: "GET"                    Dependent: "Duration"

 SUMMARY OF "GET"
   LABEL    |  SIZE     |  MEAN     |CONF. INT.|
 ----------|----------|----------|----------|
 Get Advice|    99     |    27.66 |     6.4   |
 Get Inform|   269     |    20.63 |     3.88  |
 Get Someth|   116     |    22.49 |     5.91  |
 Get Emotio|   105     |    28.44 |     6.22  |

 ANOVA TABLE
 -------------------------------------------------------------
 SOURCE OF | Sum of    | Deg. of   | Variance | Obtained |  P
 VARIATION |Squares SS| Freedom   | Estimate |  ratio   |
 -------------------------------------------------------------
 BETWEEN   |  6604.48 |        3  |  2201.49 |     1.47 |> 0.10
 GROUPS    |           |           |          |          |
 -------------------------------------------------------------
 WITHIN    |878329.99 |      585  |  1501.42 |
 GROUPS    |           |           |          |
 ---------------------------------------------
 TOTAL     |884934.47 |      588  |
```

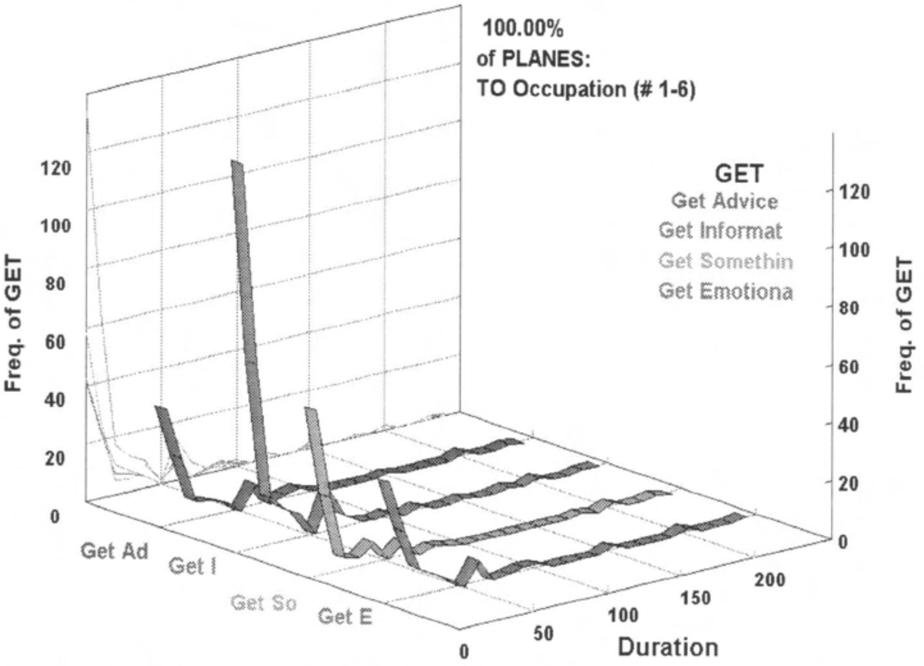


Figure 4.11. Network ANOVA display of Dependent: Duration with Independent: GET, **Stack**ed on**TO** Occupation.

4.4.2 Purpose by gender

How do the purposes of men differ from those of women for getting and receiving different types of information and support? This question suggests stacked **XTABS** with first variable GET, second variable **FROM** Gender and stacked variable **TO** Gender (a grouping must be the first or second variable). Table 4.6 and Figure 4.12 shows significant difference from expected for the Amount of interaction **TO** Gender for purposes GET, with no multiplier. Females are higher and males lower than expected for "Get Emotional Support", and this result holds whether the sender (**FROM** Gender) is female or male (This was one of the results that caused the students who collected the data to complain that it made them seem like "stereotypes").

Table 4.6. Differences from expected for Amount of interaction **TO** Gender for purposes GET over both **FROM** Gender values. Results are also significant **FROM** female and **FROM** male.

```
% Difference from Expected Values
  100.00% of PLANES: FROM Gender ( # 1-2)
MULTIPLIER=NONE
% Diff                        ROWS = TO Gender
                              COLS = GET
               Get Info          Get Emot
        Get Advi        Get Some
        -----------------------------
 female |     3.%    -9.%   12.%   10.%
        |
  male  |    -5.%    18.%  -23.%  -21.%

CHI-SQUARE =  27.043   DF = 3   P < 0.01   Cramer's Phi= 0.13
```

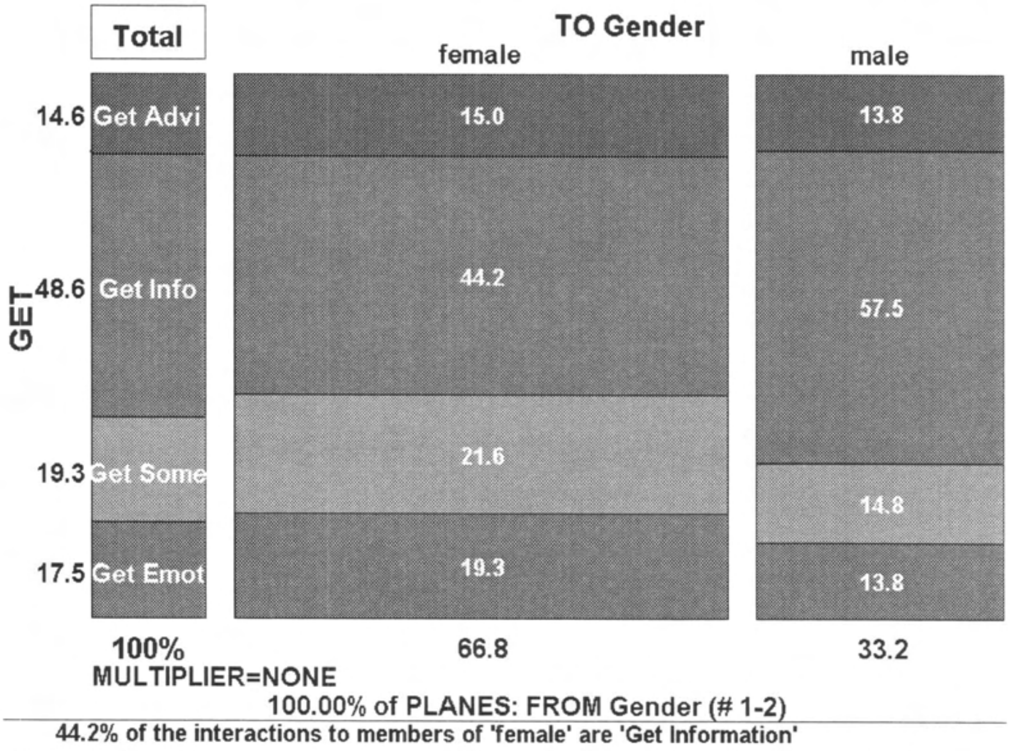


Figure 4.12. Panigram showing Amount of interaction TO Gender for purposes GET

Very similar results are obtained when the "purpose" grouping variable is GIVE, which consists of the Link variables "Give Advice", "Give Information" and "Give Emotional support" as shown in Table 4.7. **TO** female is significantly higher and **TO** male is lower for both "Give Advice" and "Give Emotional support", and the opposite for "Give Information". This relation holds as well when the interactions come **FROM** females, but disappears (no significance) for interactions **FROM** males.

Table 4.7. Differences from expected for Amount of interaction **TO** Gender for purposes GIVE over both **FROM** Gender values. Results are also significant **FROM** female but not **FROM** male.

```
% Difference from Expected Values
  100.00% of PLANES: FROM Gender ( # 1-2)

MULTIPLIER=NONE
% Diff                      ROWS = TO Gender
                            COLS = GIVE
               Give Inf
       Give Adv        Give Emo
       ----------------------
 female |    7.%   -9.%   18.%
        |
  male  |  -13.%   16.%  -32.%

CHI-SQUARE =        25.715     DF = 2
         P < 0.01      Cramer's Phi= 0.15
```

4.4.3 Inverted groupings

The results of the previous example suggest a different type of question: do the responses of men differ from women for the purposes collected in the grouping GET? That is, for the four different purposes, does one sex answer "None", "Some" or "All" more often than the other? To answer this question, we first use **Groupings→Invert** to invert the grouping GET to produce a set of categories "None", "Little","Some", Half", "Most" and "All". This inverted grouping is given the name "Inv GET". Each category can take on the values 1= "Get Advice" to 4= "Get Emotional Support" but these values will not be used in the analysis. We will use Network **XTABS** to count the number of "None" responses over all four types of "GET" purposes, the number of "Little" responses, and so on. Since we are only interested in number, the multiplier "NONE" is used. We are mainly interested in the counts **FROM** Gender, since these people are the ones who chose the values assigned to each of the contacts.

To create convenient tables, we select the three variables in the order: "**FROM** Gender", "Inv GET" and "**TO** Gender". The results for all values of "**TO** Gender" are shown in Figure 4.14 and Table 4.8. The figure shows that the responses **FROM** men are higher than expected for the two categories "Some" and "Most", and that this deviation from expected is significant. The same significant pattern is also seen for responses **TO** females and **TO** males. It appears that the men tend to choose these responses in describing the proportions of the four GET purposes, no matter what the purpose, or which gender the link was **TO**. (It could also be said that the females tend to avoid these responses).

Table 4.8. Differences from expected for types of response of GET purposes **FROM** Gender

```
 100.00% of PLANES: TO Gender ( # 1-2)    MULTIPLIER=NONE
       % Diff                    ROWS = FROM Gender
                                 COLS = Inv GET

                  Little          Half            All
            None          Some           Most
         ------------------------------------------
  female |   17.%    12.%  -25.%   17.%  -24.%    0.%
         |
   male  |  -40.%  -29.%   58.%  -39.%   56.%    0.%

CHI-SQUARE = 38.679    DF = 4    P < 0.01   Cramer's Phi= 0.28
```

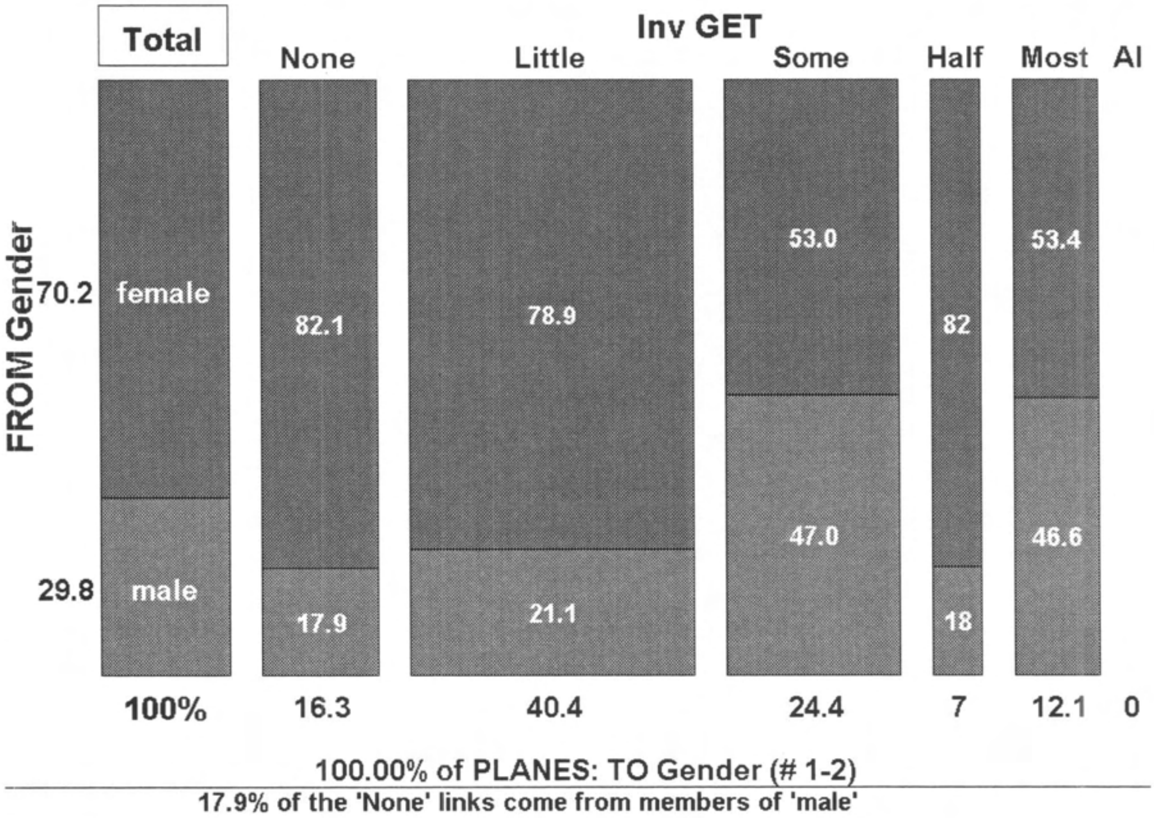


Figure 4.13. Types of responses to the GET purposes **FROM** Gender. Men tend to choose two response values significantly more than expected.

An obvious question is: does this pattern of responses also appear in the GIVE grouping of purposes? To answer this, we need to use **Groupings→Invert** to invert the grouping GIVE, and repeat the network **XTABS** with "Inv GIVE". The results are shown in Figure 4.14 and Table 4.9. The results are similar and significant overall as well as for links **TO** female and **TO** male.

Table 4.9. Differences from expected for types of response of GIVE purposes **FROM** Gender

```
100.00% of PLANES: TO Gender ( # 1-2)  MULTIPLIER=NONE
 % Diff                        ROWS = FROM Gender
                               COLS = Inv GIVE


                    Little          Half           All
               None          Some           Most
          ---------------------------------------------
  female |    20.%     7.%  -22.%    19.%  -26.%     0.%
         |
   male  |   -60.%  -20.%    65.%  -56.%    78.%     0.%

 CHI-SQUARE =    32.605    DF = 4      P < 0.01   Cramer's Phi= 0.29
```

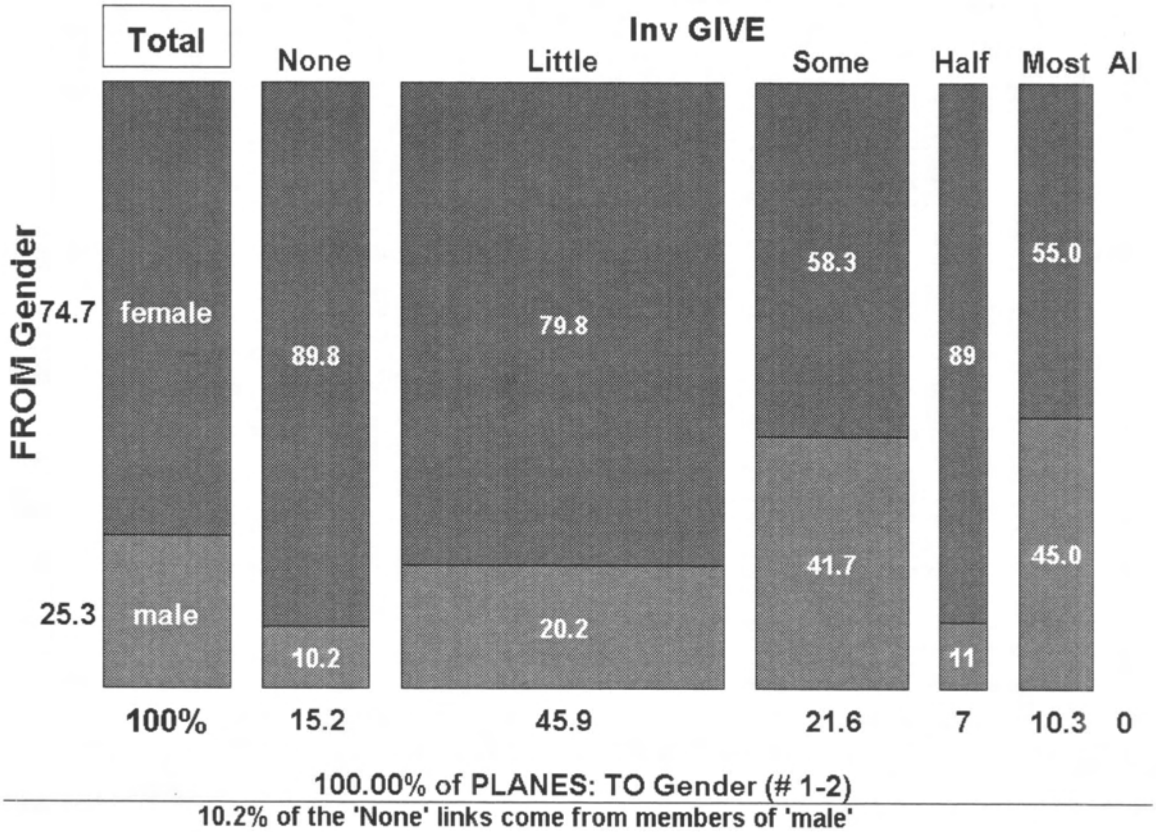


Figure 4.15. Types of responses to the GIVE purposes **FROM** Gender.

4.5 Technical Appendix

4.5.1 List of errors anticipated by Groupings module.

Generic values for Grouping names represented by <Gname> Numbers are represented by <n1>.
`Error Text` is followed by an Explanation and suggested Solution.

"<Gname>": SAME VARIABLES

- **Define** has just been used to create a grouping. The link variables in this grouping are exactly the same as those in <Gname>.

  Solution: Cancel definition to avoid duplication. **Define** grouping with different variables.

"<Gname>": NAME IN USE

- Attempt to give a duplicate name to grouping during **Define**.

  Solution: Choose a different name for the grouping.

NO LABEL

- Attempt to give blank name to grouping during **Define**

  Solution: Blank names are not allowed.

VALUE LABELS NOT ALL THE SAME!

- Attempt to **Invert** a grouping for which Link variables do not all have the same value labels. **Invert** exchanges Link names and Link value labels which must therefore all be the same.

  Solution: **Disband** the grouping. Make sure value labels are all the same for all Link variables. Re-**Define** the grouping and **Invert** it.

NO VALUE LABELS! NUMERIC VALUES WILL BE USED

- Warning given when **Invert** is used on a grouping for which no Link variable has value labels. The actual values are used as labels and exchanged with Link names.

  Solution: Warning only. No action needed.

<n1> NOT NUMERIC

- Error report from **Recode** whenever either an additive or multiplicative entry is not numeric.

  Solution: These Edit windows accept numbers only.

4.5.2 Storing and retrieving groupings

Groupings are stored and retrieved as part of MultiNet system files. The only way to review the contents of a grouping is by using **Groupings→Display** command. **Files→Export** does not currently allow saving groupings. **Files→Import** allows groupings to be defined as part of the ASCII .LIN files in a rather ad hoc manner. 301.LIN contains the following lines just before the data begins:

```
END     (Signals end of header)
USER GET : 16 17 18 19 ; 0 1
USER GIVE : 20 21 22 ; 0 1
```

These two lines define the groupings GET and GIVE by listing:

```
USER GET: (USER signals that the definition of a grouping follows, with name “GET”)
16 17 18 19  (these are the indices of the 16th - 19th Link variables defined in header)
;0 1         (Additive constant and multiplicative constant)
```

Inverted groupings are signalled by having at least one of the Link indices negative. E.g., Inverted Get would be coded as:

```
USER Inv GET:  -16 17 18 19 ; 0 1
```

4.5.3 History of analysis with groupings

Network analysis of groupings of Link variables that describe purpose and importance was devised by Dr. William Richards for use with to proportional link analysis in the program FATCAT, along with the use of panigrams to visualize the results of this type of analysis. The present author extended this type of analysis to 3 dimensions and to the **ANOVA** analysis in MultiNet (Seary, 1997).

4.6 References

Richards, W.D. (1986). *FATCAT: a different kind of network analysis program,* http://www.sfu.ca/~richards/Pdf-ZipFiles/fatman2.pdf

Seary, A.J. (1995) *MultiNet for DOS*, Presented at International Conference on Social Networks (Sunbelt XV), London, UK

5. The Eigenspaces Module

5.1 Introduction

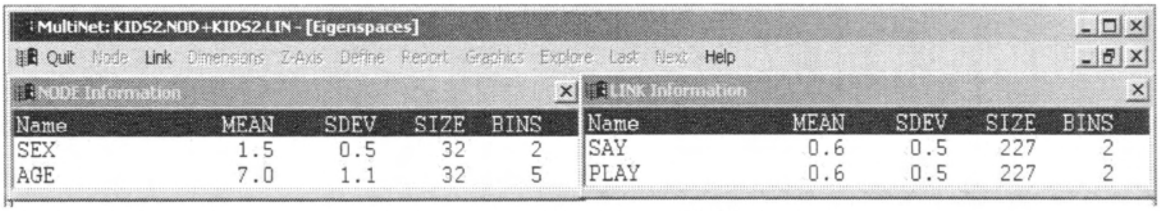
The Eigenspaces module is the main tool provided by MultiNet for visualizing networks, and the main method used for these visualizations is location of nodes based on their *eigenvector* coordinates. In effect, *eigendecomposition* or *spectral analysis* allows networks to "draw themselves". There are a number of advantages to this method of visualizing networks:

- The method is completely reproducible
- The eigenvector coordinates and distances between them are meaningful
- The related eigenvalues are also meaningful
- The method is efficient for large networks using sparse methods

Because eigendecomposition produces multi-dimensional results, the Eigenspaces module is the most interactive part of MultiNet, allowing real-time manipulation of up to 3 spatial dimensions which can be freely chosen, and with displays that use colour to represent node, link, spatial and directional attributes to help understand any patterns in the network. This module also allows new node variables to be defined, based on the eigenvector coordinates or their sign patterns. These variables can then be used in other MultiNet modules for analysis or modelling.

Not all the visualization tools use eigenvector coordinates. The 2- and 3-D displays allow a node attribute to replace an eigenvector. The 1-D display makes extensive use of node attributes to produce permutations of the (virtual) *adjacency matrix* of a network, producing very helpful displays of the relationship between node and link variables.

The Eigenspaces and Pstar modules currently restrict network size to no more than 5,000 nodes for the practical reason that few social network datasets are this large. The code for Eigenspaces has been tested successfully for networks of 20,000 nodes.



Name	MEAN	SDEV	SIZE	BINS
SEX	1.5	0.5	32	2
AGE	7.0	1.1	32	5

Name	MEAN	SDEV	SIZE	BINS
SAY	0.6	0.5	227	2
PLAY	0.6	0.5	227	2

Figure 5.1.. Initial Eigenspaces display with KIDS2.NOD and KIDS2.LIN

5.2 Data manipulation I

From the main menu, the Eigenspaces module is started with a single mouse button click. The **Eigenspaces** menu starts with everything greyed out (can't be selected) except **Quit**, **Link** and **Help** (Figure 5.1). Selecting **Help** will produce descriptions of each of the menu items. **Quit** will leave the **Eigenspaces** menu entirely and return to the Main menu. **Link** is where to start: select a network (link variable) for eigendecomposition. Clicking on **Link** produces a new selection window containing a list of all the link variables (fig. 2).

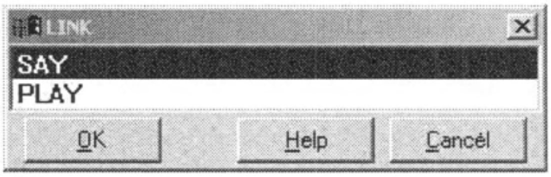


Figure 5.2. Link variable selection

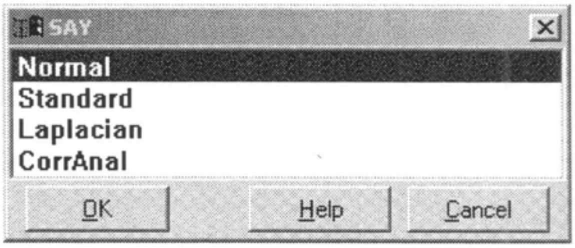


Figure 5.3. Graph spectrum selection

In this example there are two choices: "SAY" and "PLAY". This type of selection window is used throughout MultiNet for selection from a list of variables (node or link) or other options. In this case a selection may be made by clicking on a variable name and then click on the “OK” button. Another option which is always available for this type of window is double-clicking on a selection. The “Help” and “Cancel” buttons are also always available for context-sensitive help, or to cancel the selection and delete the selection window.

Once a network variable has been selected, another selection window appears for choice of type of eigendecomposition. These will be described briefly here, and in more detail Technical appendix 5. Much more information is available in Section 8. Mathematical Background. The eigendecompositions (also known as graph spectra) currently supported are:

- *Normal*: This is basically *Correspondence Analysis* of the symmetrized binary matrix of links. Eigenvalues may be positive or negative. Eigenvectors are *normalized.*
- *Standard*: This is the standard eigendecomposition of the symmetrized binary adjacency matrix. Eigenvectors are normalized.
- *Laplacian*: This eigendecomposition first loads the diagonal of the symmetrized binary matrix with the node degree. Eigenvalues are always positive. Eigenvectors are normalized.
- *CorrAnal*: This is Correspondence Analysis, with the diagonal loaded with node degree. The eigenvalues are always positive. The eigenvectors are the same as those of Normal, but weighted by eigenvalue and not normalized.

5.2.1 The default display

The result of an eigendecomposition is an *eigenspace*: a set of *eigenpairs* of eigenvalues and corresponding eigenvectors. For all 4 graph spectra the eigenvectors are immediately useful for visualization purposes. The Normal and closely-related CorrAnal eigenvalues are also immediately useful. For the Standard spectrum the eigenvalues are generally less easy to interpret except in special cases. For the Laplacian, the eigenvalues have some useful interpretations since this eigenspace shares many properties with the continuous Laplacian operator of mathematical physics. Of the four, the Normal is most generally useful, and has been the main subject of the author's Ph.D. research program.

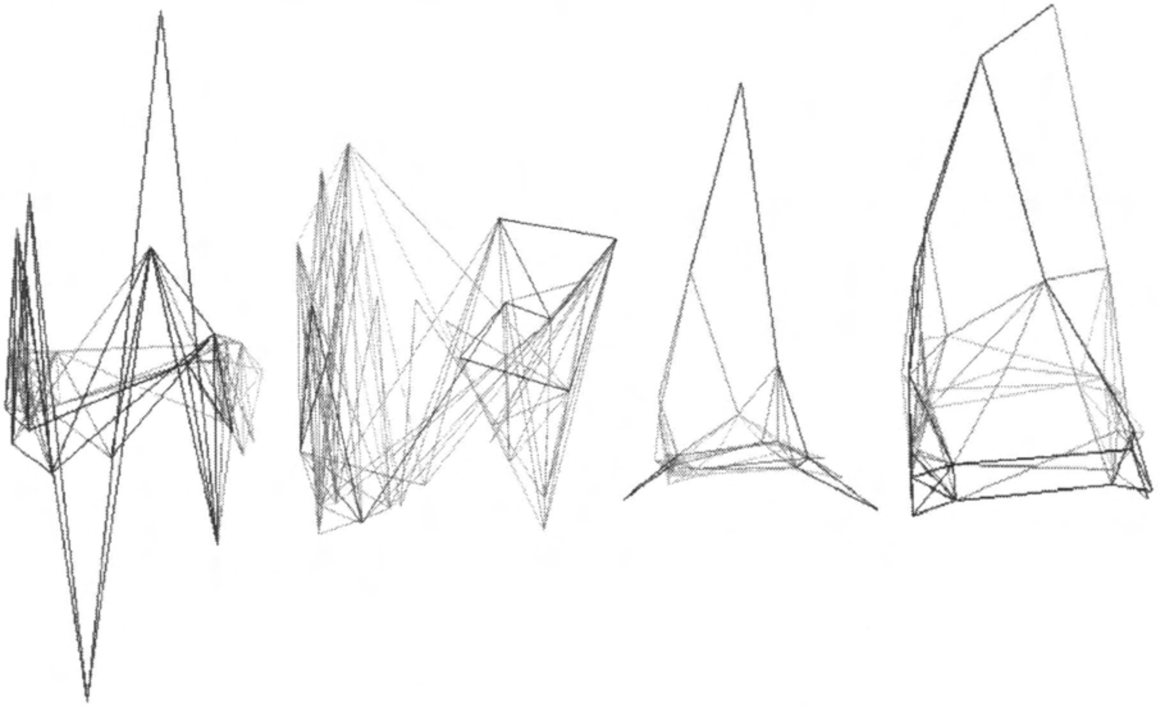
Once the eigendecomposition is completed the network can be displayed by using the coordinates of selected eigenvectors to place the nodes in a 3-, 2-, or 1-dimensional display. The default is 3-dimensions. The default eigenvectors depend on the eigenspace chosen. For the Normal and CorrAnal eigenspaces, there is always a largest eigenvalue of 1 with corresponding constant ("trivial") eigenvector, so the eigenpairs are first ordered by <u>descending</u> absolute values of eigenvalues, then rotated so that the trivial constant eigenpair is placed last (and the second largest becomes first and so on). Then the first three are chosen for the default initial display. For the Standard, a similar method is used since the largest eigenvalue corresponds to an eigenvector of constant <u>sign</u>. For the Laplacian, there is always an eigenvalue of 0 with corresponding trivial eigenvector, and the (non-negative) eigenvalues are ordered by <u>ascending</u> values, then the trivial rotated to the end. Figures 5.4a-d show the results for the KIDS2 variable SAY. Tables 5.2a-d show how the eigenvalues are ordered in each case.

In these displays, each node is given x and y coordinates based on the coordinates of the 1st and 2nd eigenvectors. Lines are drawn between nodes that are connected as defined by non-negative values of the link variable. Since the link variable was binarized and symmetrized for the eigendecomposition, this is an accurate representation of the result. The default 3-D display also includes a simple "grey-scale" effect that indicates the signs of the 3rd eigenvector. There are a number of other defaults at work in this display which will be discussed in the section on the "Explore" window.

Notice that for the SAY network, both Normal and Standard spectra have negative 2nd eigenvalue, which is associated with oscillations in the 2nd eigenvector (Y-direction). This never happens with the Laplacian and CorrAnal, which have only non-negative eigenvalues. Eigenpairs with negative eigenvalues can be very useful, which is one of the great strengths of the Normal spectrum (Section 9).

Tables 5.1. a-d. Examples of how eigenvalues are ordered for SAY

a. Normal	b. Standard	c. Laplacian	d. CorrAnal
1 = 0.82348	1 = 5.85387	1 = 1.00079	1 = 0.91174
2 = -0.65417	2 = -3.7221	2 = 1.32385	2 = 0.79943
3 = 0.59885	3 = -3.61207	3 = 1.85425	3 = 0.71044
⋮	⋮	⋮	⋮
30 = 0.01302	30 = 0.08302	30 = 13.63041	30 = 0.25305
31 = -0.01205	31 = -0.05364	31 = 14.41048	31 = 0.17291
32 = 1.0	32 = 8.23625	32 = 0.0	32 = 1.0



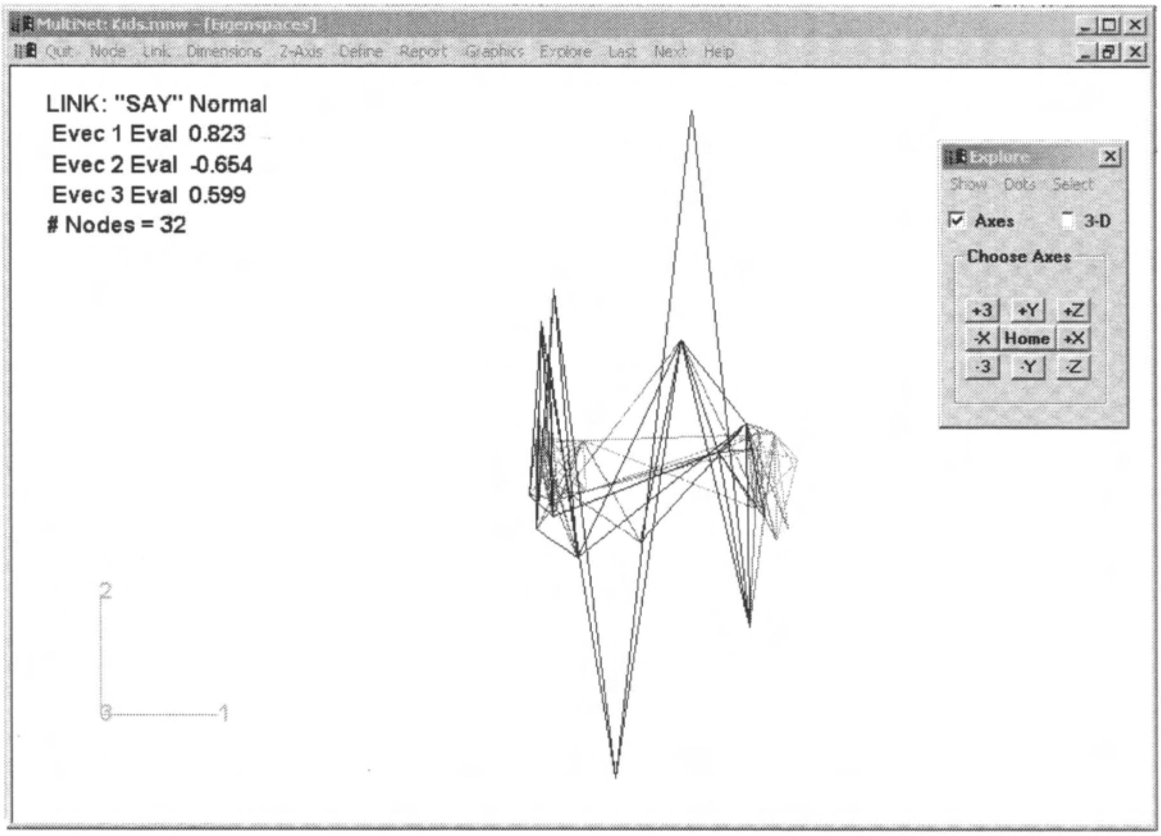
a) Normal b) Standard c) Laplacian d) CorrAnal

Figure 5.4. Default display of SAY from a) Normal, b) Standard, c) Laplacian and d) CorrAnal spectra, based on the first three eigenvectors as ordered by eigenvalues

5.3 Data visualizations I: the 3-D and 2-D displays

Figure 5.5 shows the complete Eigenspaces window with the default display for link variable SAY. The left part of the display contains additional information about the display. The upper left lists the link variable and the type of eigenspace; the numbers and values of the eigenvectors being displayed in the order X, Y, Z; and the total number of nodes in the network.

More textual information is also displayed as more graphical information is added to the display. The lower left shows a set of axes pointing in the directions that the current eigenvectors have been rotated into. Initially this is the X, Y, and Z directions. The lengths of the axes lines are significant and show the scale of the display. All eigenvectors except CorrAnal are normalized, and each axis vector has length 0.2. For CorrAnal, the lengths are 1. The default 3-D display is initially placed so that the origin is at the centre of the screen.



MultiNet: Kids.mnw - [Eigenspaces]
Quit Node Link Dimensions Z-Axis Define Report Graphics Explore Last Next Help
Explore
Show Dots Select
☑ Axes
□ 3-D
Choose Axes
+3
+Y
+Z
-X
Home
+X
-3
-Y
-Z
2
3
1

LINK: "SAY" Normal
Evec 1 Eval 0.823
Evec 2 Eval -0.654
Evec 3 Eval 0.599
Nodes = 32

Figure 5.5. Complete MultiNet **Eigenspaces** default window with **Axes** checked

5.3.1 The **Explore** window

At the upper right is the **Explore** window, which is a control and command center for the display. This is a separate window which can be moved around by clicking on the menu bar and dragging - a standard Windows method. The MultiNet Eigenspaces module provides highly interactive displays, and almost all of the functions are available from the **Explore** window. There are 3 sets of commands available from the menu bar, two check boxes, and either 9 or 7 buttons depending on the **Axes** checkbox.

The Explore window has two modes, controlled by the **Axes** checkbox. If **Axes** is checked, (as in Figure 5.5) the buttons will change the eigenvectors being displayed:

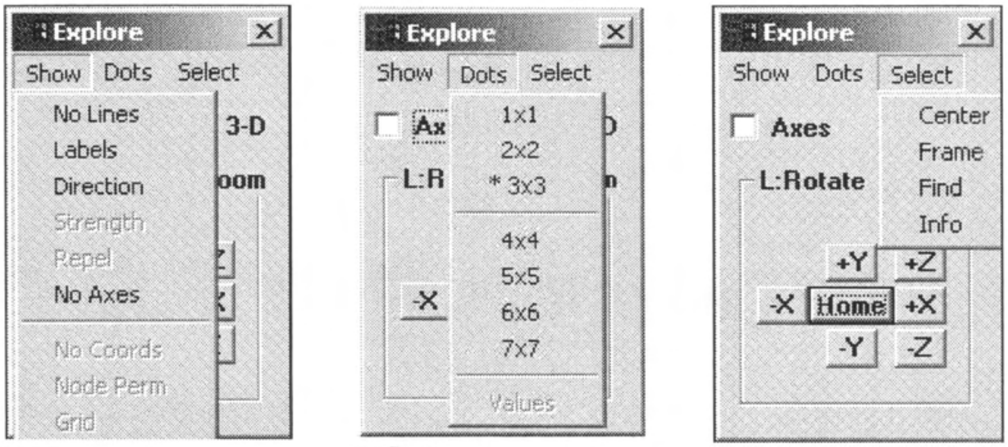
- +X and -X changes the eigenvector in the X direction (horizontal).
- +Y and -Y changes the eigenvector in the Y direction (vertical).
- +Z and -Z changes the eigenvector in the Z direction (in and out of the screen).

- +3 and -3 are a convenience to allow changing all 3 eigenvectors together. Each are changed by +1 or -1 simultaneously.
- **Home** (left-click or Left-**Home**) brings the display to default position which shows eigenvectors 1, 2 and 3 on the X, Y, Z axes.

Selecting a new eigenvector for any axis will change the informative text display in the upper left, restore the axes to point in along the X, Y, and Z directions, and restore the scale so that the axes displayed in the lower left again have length 0.2 (or 1 for CorrAnal).

If the **Axes** checkbox is unchecked, then the buttons control movement around the X, Y and Z axes (figures 7, 8). Rotating on any axis continues as long as the LEFT mouse button is pressed. Rotation follows the convention of a right-handed coordinate system with positive Z out of the screen, so that pressing +X rotates into the positive Z direction, +Y rotates into the positive X direction, and +Z rotates into the positive Y direction. These directions are reversed for the buttons -X, -Y and -Z, respectively.

By pressing the RIGHT mouse button, the action becomes Zoom for +Z and -Z and Pan (translation) for +X, -X, +Y, -Y. Zoom changes the scale, and so affects the axes display. All three axes are magnified (+Z) or shrunk (-Z) by the same amount and the center of the zoom is the rotation origin (initially the eigenspace origin). Pan horizontal is to the right for +X, to the left for -X. Pan vertical is up for +Y and down for -Y. These actions only affect the center of the display, so that any rotations still take place about the eigenspace origin. Left-**Home** restores the display to its original scale and orientation (no magnification, translation, or rotation). Right-**Home** restores the orientation (no rotation), but not the magnification or translation.



a) **Show** menu b) **Dots** menu c) **Select** Menu

Figure 5.6. Explore window and menus

The **Show** menu item of the **Explore** window controls a number of display defaults. In Figure 5.6a, some of these are "greyed out" and so not accessible. This occurs when a selection is not applicable to the network being displayed. These menu items are all "toggles" which turn something off or on. If it is already on, the word "**No** " is prepended to the menu item to show that choosing this item will turn something off. In order, these defaults are:

- **Show →No Lines** turns off the display of lines. This is useful when the network is very large or very dense. Only the nodes are shown (as dots unless **Labels** or **Values** has been selected). This selection is "sticky" and persists over an entire MultiNet session.
- **Show →Labels** turns on the display of ID numbers for each node. Displaying labels can be slow for large networks, so this choice initially defaults to No Labels whenever the network or number of dimensions is changed.
- **Show→Direction** is available (not greyed out) if the network is *directed.* Solid lines are replaced by a combination of lines dashed at the sender end and dotted at the receiver end. Reciprocated links remain as solid lines. A description of this convention appears in the upper right of the display. Displaying direction can be slow for large or dense networks. An example of **Show→Direction** is seen in Figure 5.20.
- **Show →Strength** is available (not greyed out) if the link variable has non-binary values (non-negative integer or real), <u>and</u> if the number of dimensions being displayed is 1 or 2. In 3-D, this selection is not available since a grey scale is used. Displaying strength can be slow for large or dense networks. Non-binary link variables usually measure the amount or strength of a relationship, hence the name. See Figure 5.18, where "strength" is actually a fit probability.
- **Show →Repel** is used to displace nodes that have the same coordinates (because they have the same connections) around the common coordinate. This affects <u>only</u> the display, not the eigenvectors. This is greyed out if all nodes have unique coordinates.
- **Show →No Axes** toggles the axis display at the lower left off or on.

The next 3 **Show** choices (below the separator) are available only for the 1-D display and will be discussed later.

With a little practice, it becomes simple to move around the display with a combination of right and left mouse translations and rotations. Even more control is available by using the **Select** menu item from the **Explore** window (figure 5.6c). Clicking on **Select** produces a drop-down menu which allows changing the rotation/zoom origin (**Center**, **Find**) or scale (**Frame**), as well as detailed information about a node and its immediate neighbours (**Info**).

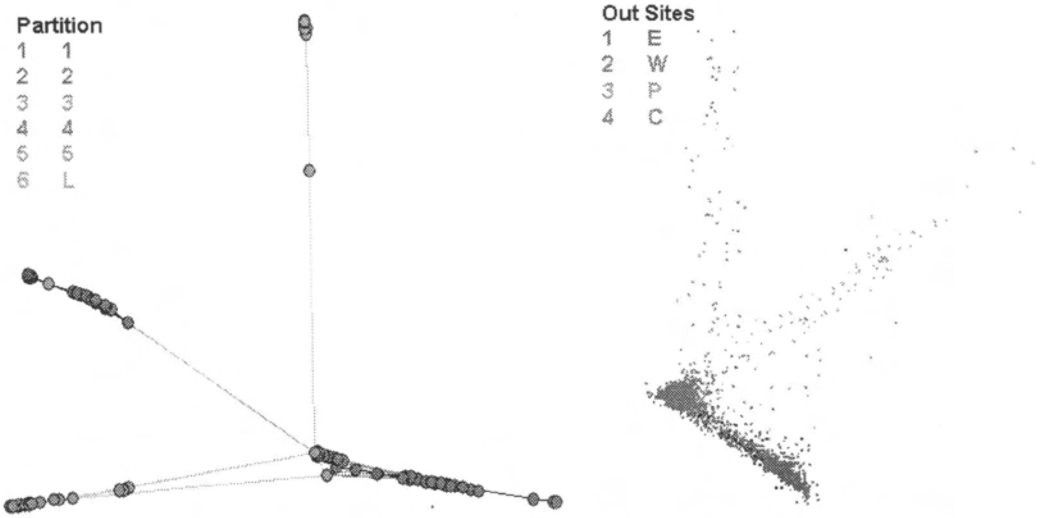
- **Select →Center** changes the cursor to a cross-hair. Move the cursor around the display. Clicking then selects the node nearest the cross-hair cursor as the new rotation origin for the display so that any further rotations (around X, Y, or Z) will not move this node. The center of rotation is restored to the eigenspace origin by pressing left-click on **Home**.
- **Select →Frame** changes the cursor to a cross-hair. Move the cursor to one corner of a new frame, left-click and hold down the button while moving to another corner of the new frame, then release the button. This changes the scale so that the part of the network within the frame fills the display screen, and the axes display at the lower left shows the new scale (which will not generally be the same for each axis). The default scale and display is restored by pressing left-click on Home.
- **Select →Find** produces a selection window containing the ID numbers of all the nodes in the network. Selecting one of these makes this node the new rotation origin. Combining this with Zoom makes it simple to find any node (and its *neighbourhood*).
- **Select →Info** allows detailed information on any node. Upon left-click the node nearest the cross-hair cursor is chosen and a window opens with information about: node ID number and node attribute and value (if any chosen); ID numbers that receive links from this node, along with their values and link strengths; ID numbers that send links to the chosen node, along with their values and link strengths. An example is shown in Figure 5.9.

The **Dots** menu item of the Explore window is used to over-ride the automatic selection of dot size (for **No Lines** or the **1-D** display) or to show node attribute values if a node variable has been selected. The program always attempts to select an appropriate dot size (smaller dots for larger number of nodes). The current setting is marked with '*' and can be changed by selecting one of.

- Sizes 1 to 3 are drawn as pixels, and are most useful for large networks and with **No Lines**.
- Sizes 4 to 7 are drawn as circles, and are most useful with smaller networks or with **Lines**.

Dot size is always recalculated with any change of link or dimension. For either style of Dot size, colour is used to represent values if a Node variable has been chosen. **Dots →Values** replaces dots with node value labels (eg. **Node** below). Examples are shown in Figures 5.7, 5.11 and 5.18.

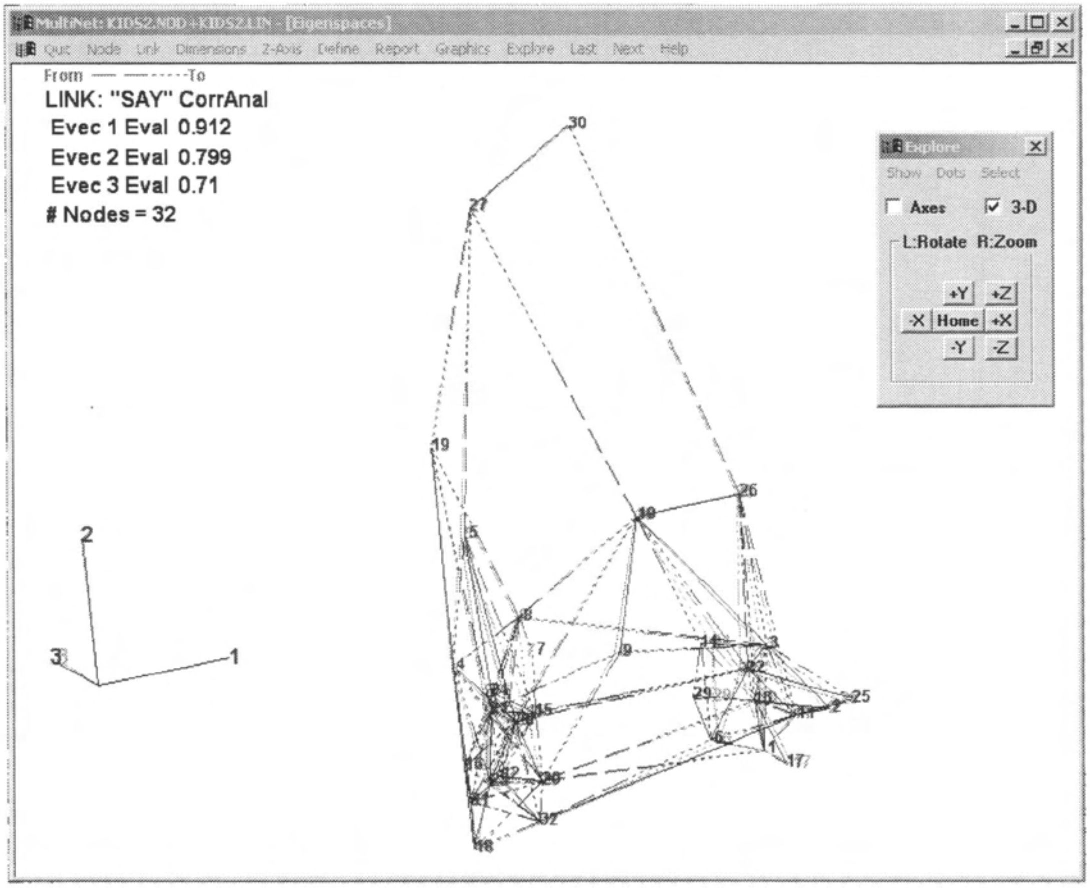
Finally, the **3-D** checkbox is enabled for the 3-D display. Checking this switches the display to *anaglyphic* red/cyan 3-D, which gives the illusion of depth when used with red-green 3-D glasses (which works best with the red lens on the left eye). Figure 5.8 shows a rotated anaglyphic 3-D display of SAY CorrAnal eigenvectors, with **Direction** and **Labels** Selected.



a) **Lines, Dots** = 4 for 145 nodes

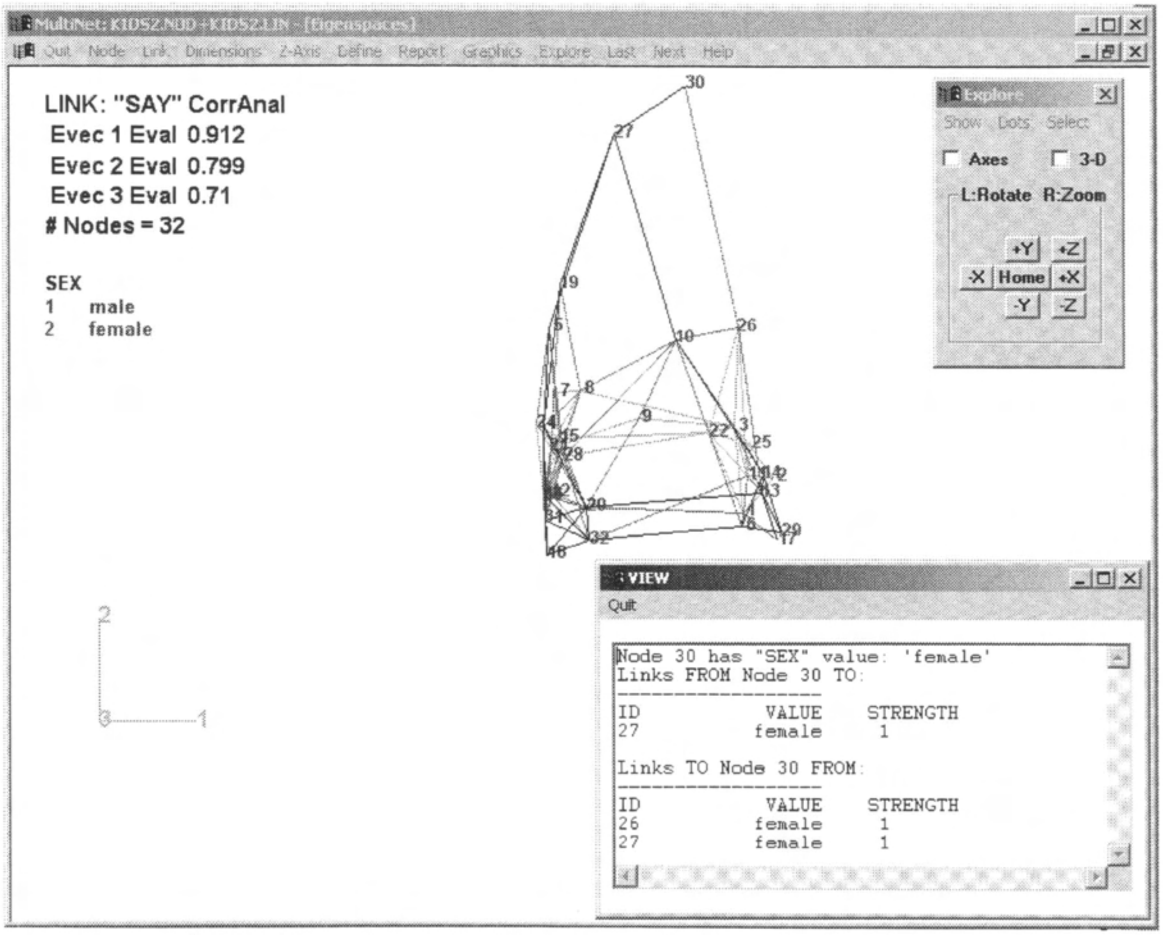
b) **No Lines, Dots** = 1 for 2700 Nodes.

Figure 5.7. Dots and Lines styles for a small and large network. Colour shows Node relationships.



```
From —— -----To
LINK: "SAY" CorrAnal
 Evec 1 Eval 0.912
 Evec 2 Eval 0.799
 Evec 3 Eval 0.71
 # Nodes = 32
```

Figure 5.8. Anaglyphic 3-D display of SAY CorrAnal eigenvectors.
Rotated with **Direction** and **Labels** Selected and **Axes** unchecked.


MultiNet: KIDS2.NOD+KIDS2.LIN - [Eigenspaces]
Quit Node Link Dimensions Z-Axis Define Report Graphics Explore Last Next Help
Explore
Show Dots Select
□ Axes □ 3-D
L:Rotate R:Zoom
+Y +Z
-X Home +X
-Y -Z
30
27
19
26
10
7
8
9
22
3
25
15
28
20
32
29
17
2
3
1
VIEW
Quit

```
LINK: "SAY" CorrAnal
 Evec 1 Eval 0.912
 Evec 2 Eval 0.799
 Evec 3 Eval 0.71
# Nodes = 32

SEX
1    male
2    female
```

```
Node 30 has "SEX" value: 'female'
Links FROM Node 30 TO:
-------------------
ID            VALUE    STRENGTH
27           female      1

Links TO Node 30 FROM:
-------------------
ID            VALUE    STRENGTH
26           female      1
27           female      1
```

Figure 5.9. 3-D CorrAnal SAY with node variable SEX and **Show→Labels**.
Select →Info display shows a neighbourhood, with node and link values.

5.4 **Eigenspaces** Menu Bar

Although much of the interaction in Eigenspaces is controlled from the **Explore** window, a number of useful functions are also available from the Eigenspaces Menu Bar. All are activated with a single mouse click. They include:

5.4.1 **Quit**

Exit from the Eigenspaces Module back to the Main MultiNet menu.

5.4.2 **Node**

Select a node variable for inclusion in the display. Its values are then used to colour the nodes of the network and also appear in the Report and the **Select →Info** display. When a node variable is selected, a list of values and value labels appears to the left using the available colours, and the nodes show as dots which are coloured accordingly. When **Show→Labels** or **Dots →Values** are selected from the **Explore** window, node positions are labelled with ID numbers or value labels which are coloured according to node values. If both are chosen, only ID numbers are shown. If the node variable has too many values (more than the default 12 or a user-selected number in Module Preferences, maximum 20), the list tells you how many, and each value gets one of the available colours. Figure 5.9 shows CorrAnal SAY with node variable SEX, the **Select →Info** display and **Show→Labels**. The relationship between SEX and the clusters at each end of the X-axis are immediately obvious: boys say they play with boys, and girls say they play with girls.

5.4.3 Link

As described above, **Link** is used to select a link variable (which defines a network). One of four types of eigendecomposition is then calculated and displayed. Once this is done, the other menu items and the **Explore** window become available for manipulating the network display.

5.4.4 Dimensions

Dimensions is used to select the number of *active* eigenvectors in the display. 3- and 2-D displays are superficially very similar (1-D is described in a separate section below), and both allow for full 3-D rotations and translations. The main differences are the number of eigenvectors involved in any *partition* and the lack of grey scale (and anaglyphic 3-D) for the 2-D display. Lack of grey scale means the 2-D display can be used to visualize link values (weights or strengths) for non-binary link variables. The number of active dimensions is indicated by the informative display in the upper left. The eigenvectors that do not contribute to any partition are greyed out. For Normal and CorrAnal, and if the network is at least *weakly connected*, a further choice is available: **Dimensions→Subsets** will show a display of number of subsets (abscissa) vs upper bound on *distance* between subsets (ordinate). This may be used as a guide to the number of "clusters" or "groups" present in the network. Also, an estimated upper bound on the *diameter* of the network is displayed as a grey horizontal line. These estimates are based on Normal eigenvalues (Chung, 1995) and also appear in the Report.

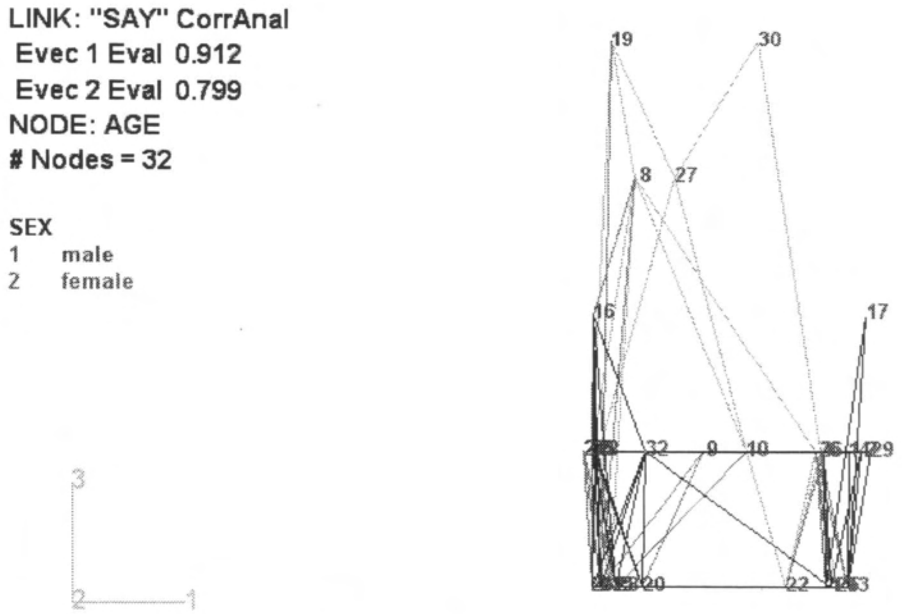
5.4.5 Z-axis

Z-axis allows mixing eigenvector coordinates with node variables in the display.

- **Z-axis→Node** produces a selection window containing names of node variables. Selecting one of these causes the third eigenvector to be replaced by this node attribute, suitably scaled to fit the display. Rotating around either X or Y axis brings this node variable into view in the display. This is a useful way to display more than one node variable at a time (Figure 5.10). It can be useful if the node variable is a measure of prominence such as betweenness centrality (Brandes & Cornelsen, 2001) and in finding clusters in sub-groups related to a node variable. It is also useful for separating different types of nodes in multi-mode analysis (Figures 9.9 and 9.10).

- **Z-axis→ Evec** restores the 3rd selected eigenvector along the Z-axis.

Selecting either one causes the display to return to the default **Home** position. Note that **Z-axis→Node** affects only the display and has no effect on the Report or on derived partitions.



LINK: "SAY" CorrAnal
Evec 1 Eval 0.912
Evec 2 Eval 0.799
NODE: AGE
Nodes = 32

SEX
1 male
2 female

Figure 5.10. Z-axis eigenvector replaced by node variable AGE, and rotated +90 deg on the X-axis Clustering by SEX on X-axis is still evident. AGE is seen to have 5 distinct values, with the older (top) having fewer links.

5.4.6 Define

Define is used to create new node variables, either directly based on eigenvector coordinates, or discrete partitions from the pattern of eigenvector coordinate signs.

- **Define→Variable** displays a selection window of eigenvector variable names. These selection names are generated automatically, and are intended only to make choice easy. The names start with the first letter describing type of spectrum (Normal, Standard, Laplacian, CorrAnal) and ending with a number showing the position of this vector in the order described in Section 2.

Any number of these may be selected by standard Windows methods (e.g., using Shift-click and/or Ctrl-click). Once this has been done the program presents an edit window containing a default name consisting of the last selection, a "-" character, and the name of the link variable. E.g., the 1st eigenvector of Normal spectrum of link variable SAY is called **1N-SAY**. The user may accept the default name, rename the variable or cancel the definition (which cancels all subsequent definitions as well). Once the name has been accepted, another Edit window appears with a default descriptive comment describing the variable. The default comment consists of the selection name for the eigenvector and the corresponding eigenvalue. Again, this may be accepted, replaced, or the definition (and all subsequent definitions) cancelled. The process is repeated for next-to-last until all choices have been dealt with (or cancelled).

- **Define→Partition** calculates a discrete-valued polynomial based on the signs of the coordinates using the current active dimensions (Seary & Richards, 1995). For 3-D this will be all 3 current eigenvectors being displayed, for 2-D the first 2, for 1-D only the first. The maximum number of distinct values in **d** dimensions is 3^d rather than 2^d since some coordinates may be exactly 0. This happens when there are more than one weak component (i.e., the network is not weakly connected). The value labels are derived from the actual sign patterns – using 'n' for negative coordinate, '0' for 0 coordinate, 'p' for positive coordinate – and are useful for interpretation and are used in displays and reports. For example, Table 5.3 (taken from the Report) shows the Normal SAY coordinates of nodes 1 to 5 from eigenvectors 1, 2 and 3 and the value label from the sign pattern. Figure 5.11 shows the corresponding display. Values are stored internally as indices into the value labels ordered by 'n', '0', 'p', which produces contiguous small integers. These values are not otherwise useful. Once the partition has been calculated, an Edit selection window appears with a default variable name for the partition. This name is derived from **d**, the number of dimensions, the type of spectrum and the link variable name as follows:

  Start with 2^d, followed by 'P',
  followed by the first letter of spectrum type
  (**N**ormal, **S**tandard, **L**aplacian, **C**orrAnal),
  followed by '.' and the link variable name.
  Example: 3-D partition from CorrAnal of SAY produces **8PC.SAY**

The user may accept the default name, rename the variable or cancel the definition. Once the name has been accepted, another Edit window appears with a default descriptive comment describing the variable. The default comment consists of the equation used to calculate the polynomial, showing the eigenvector numbers and their position. Again, this may be accepted, replaced, or the definition cancelled.

Once these new node variables are defined, they are treated exactly like any other node variable, and are immediately available for use. Both types of node variables are hybrid in the sense that they are node attributes based on network structure. For example, partitions derived from eigenvectors are very useful in finding *clusters* in networks, since sets of nodes with similar patterns of links tend to have similar sign patterns (Seary & Richards, 2002).

Table 5.2. Partition value labels derived from eigenvector sign patterns

```
Normal EIGENVECTORS OF "SAY"                   |    ID#    8PN.SAY
      0.21226      -0.0655      -0.10602   |      1    pnn
      0.27031      -0.09755     -0.04968   |      2    pnn
      0.19777       0.07406      0.0332    |      3    ppp
     -0.16447      -0.10022      0.03426   |      4    nnp
     -0.15384       0.23857      0.1955    |      5    npp
```

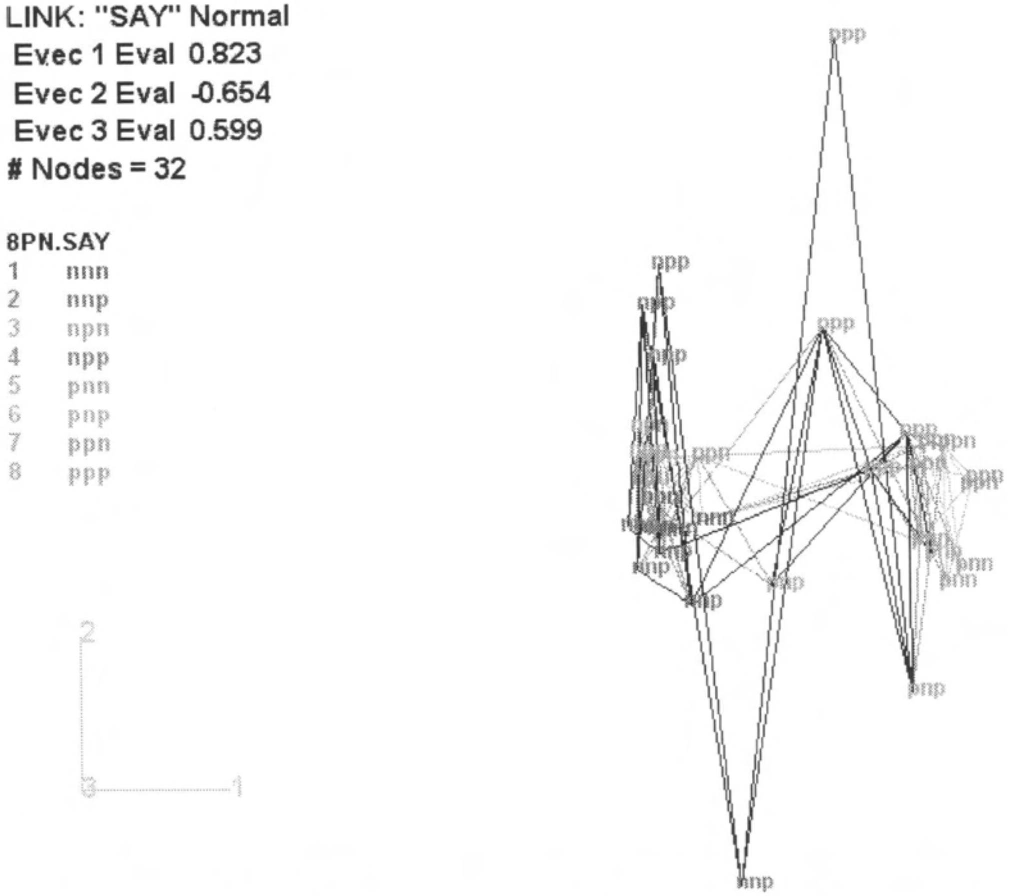


Figure 5.11. Value labels of 3-D partition used to label nodes for SAY with **Dots →Values**

5.4.7 Report

A textual report is generated whenever there is any major change in the display such as:

- Change in link variable and eigendecomposition
- Change in number of dimensions
- Change in node variable

The Report includes a large amount of detail about the current eigenspace, and includes information about the node variable if one has been chosen. Table 5.4 shows the complete report for the Normal eigendecomposition of SAY. Normal and CorrAnal allow extra information to be calculated, and these are shown in ***Courier Italic***. (The **bold** numbers on the right are not part of the report; they are there to help with the description.)

Table 5.3. Complete report for CorrAnal eigendecomposition of link variable SAY

```
MultiNet CorrAnal GRAPH SPECTRUM REPORT ON "KIDS2.NOD" 12/04/2003  15:14:25    1
LINK: "SAY"   LINKS =    142   NODES =   32  DENSITY = 0.1387                  2
CorrAnal EIGENVALUES OF "SAY"                                                  3
           1          2
     0.91174     0.79943                                                       4

PERCENT OF CHI-SQUARED =       806.16
        11.945%        6.317%                                                  5

NUMBER OF NON-ZERO COORDINATES
          32          32                                                       6

CorrAnal EIGENVECTORS OF "SAY"         |   ID#   SEX
     1.15232     -0.6225               |    1    female
     1.46749     -0.29169              |    2    female
     1.07366      0.19494              |    3    female
    -0.89289      0.20114              |    4    male
    -0.83521      1.14788              |    5    male
     1.14428     -0.7595               |    6    female
    -0.75637      0.5248               |    7    male
    -0.5114       0.54558              |    8    male
     0.07922      0.27134              |    9    male
     0.4659       1.03829              |   10    female
     1.20403     -0.27651              |   11    female
    -0.82341     -0.43877              |   12    male
     1.34767     -0.44246              |   13    female
     1.3576      -0.26163              |   14    female
    -0.73431      0.08297              |   15    male
    -0.8836      -0.47064              |   16    male
     1.51795     -0.88917              |   17    female
    -0.86171     -1.0359               |   18    male
    -0.7251       1.57134              |   19    male
    -0.45139     -0.57595              |   20    male
    -0.82721     -0.00731              |   21    male
     0.80622      0.13916              |   22    female
    -0.90003     -0.47095              |   23    male
    -0.96759      0.22055              |   24    male
```

```
 1.24832     0.03205              |    25   female
 1.10811     1.15248              |    26   female
-0.15692     3.02458              |    27   female
-0.68719    -0.09221              |    28   male
 1.55824    -0.81466              |    29   female
 0.57754     3.48757              |    30   female
-0.88832    -0.69142              |    31   male
-0.42467    -0.8928               |    32   female
```

```
PARTITION OF "SAY" BY SIGNS OF CorrAnal EIGENVECTORS 1 2                          7

   ID#  SIGNS PERMUTED ADJACENCY MATRIX
              111222233    1122   11112  12223
              26801381245785947126134793902560   SEX
    12   --    ..*..*.*.......................   male
    16   --   . *...**.*......................   male
    18   --   .* ..*.*........................   male
    20   --   *.* ..*..*.......*,..*..........   male
    21   --   **.* ***....*.**.........*......   male
    23   --   *.*** ●..*.....*................   male
    28   --   *..*** .***..*.............**...   male
    31   --   ****..* .*......................   male
    32   --   ****.*.* ..........**...........   female
     4   -+   .*....**. ....*.................   male
     5   -+   ......*..● ..*.**...............   male
     7   -+   ........... ....................   male
     8   -+   **.......*.* .*............*....   male
    15   -+   *..****....*. .*.............*...   male
    19   -+   .........*.... .................   male
    24   -+   ....***...*.... ................   male
    27   -+   ..............*. ...............*  female
     1   +-   ................. ...*...*..*...   female
     2   +-   .................. .**...*......   female
     6   +-   .................*. ....*...*...   female
    11   +-   ........*........*** *......**..   female
    13   +-   .................**.. .*.*......   female
    14   +-   ...................*.* .**......   female
    17   +-   .................*...*. ........   female
    29   +-   ...................*.*.. .......   female
     3   ++   ............*.....*.*.... ****..   female
     9   ++   ...*...................... *....   male
    10   ++   ................*.....*..** ..*.   female
    22   ++   ...................**......* **.   female
    25   ++   ....................*.......* ..   female
    26   ++   ....................**...*.*.. *   female
    30   ++   ................*..............    female
                                                                                  8
ALL EIGENVECTORS HAVE 32 NON-ZERO COORDINATES                                     9
LIST OF ACCEPTED CorrAnal EIGENVALUES FOR "SAY"
  1 =     0.91174       9 =     0.60125      17 =     0.43927      25 =     0.33142
  2 =     0.79943      10 =     0.56886      18 =     0.42597      26 =     0.30798
  3 =     0.71044      11 =     0.53721      19 =     0.40772      27 =     0.29063
  4 =     0.70473      12 =     0.52197      20 =     0.40067      28 =     0.28225
  5 =     0.69353      13 =     0.51669      21 =     0.39669      29 =     0.26815
  6 =     0.67049      14 =     0.50651      22 =     0.38281      30 =     0.25305
  7 =     0.64088      15 =     0.49398      23 =     0.3549       31 =     0.17291
  8 =     0.61331      16 =     0.45891      24 =     0.33569      32 =     1.0
```

```
EIGENVALUE BOUNDS ON DISTANCE BETWEEN SUBSETS
BOUND ON DIAMETER =        7                                            10

# SUBSETS  BOUND                                                        11
       2     19
       3      9
       4      6
       5      6
       6      6
       7      5
       8      5
       9      4
      10      4
      11      4
      12      4
      13      4
      14      3
      15      3
```

Comments

1. Describes the type of analysis, the name of the dataset, and gives a time stamp. This header is common to all MultiNet reports.

2. Describes the link variable. Number of links and density do not include any symmetrization (required for the eigendecomposition).

3. Describes the active eigenvectors being displayed and shows the corresponding eigenvalues.

4. For Normal and CorrAnal only, uses the close relationship between these spectra and Correspondence Analysis to calculate the percent of total chi-squared accounted for by each of the active eigenvectors.

5. This information is useful when the network is not weakly connected, since it shows whether the chosen eigenvectors are describing the same component. If the number of non-zero coordinates in the eigenvectors are not equal, then they contain information on different components. See Table 5.5 for a sample report on a disconnected network (one with more than one component).

6. This is a detailed list showing the eigenvector coordinates for each node (in order of ID number) and for each active eigenvector being displayed. If a node variable has been selected, the report appends the value and value label of this attribute for each node.

7. Any network may be represented by an adjacency matrix, where the value v in row I, column j shows there is a link from node I to node j of strength v. The eigendecomposition algorithm currently ignores strength, and dichotomizes the network into binary form. The resulting binary adjacency matrix representing the network is permuted on both rows and columns according to the sign patterns of the active eigenvectors. The permuted matrix shows clusters of nodes with similar patterns of connections, with non-zero links represented by '*', and 0 represented by '.'. The clusters

are either on (positive eigenvalue) or off (negative eigenvalue) the diagonal, which is left blank since self-loops are ignored. If a node variable has been selected, it is permuted as well, and the result is appended to the right of the matrix. In Table 5.4, it is clear that there are on-diagonal clusters along the diagonal, and that they are related to the node variable SEX. This matrix is omitted (with a brief explanation) if the number of nodes exceeds 500.

8. This line indicates that the network is weakly connected, since all eigenvectors have the same number of non-zero coordinates. If this were not the case, the message would describe the number of weak components, and the size of the 20 largest. See Table 5.5 for an example. The next table following contains additional information about the number of non-zero coordinates in each eigenvector. This information may be used to determine which eigenvectors contain information on the same components.

9. This table lists all accepted eigenvalues in the order defined for each type of spectrum:

- Normal, Standard and CorrAnal in descending order of absolute values, with largest (always trivial 1 for Normal and CorrAnal) rotated to end
- Laplacian in ascending order with trivial 0 rotated to end.

If the network is not weakly connected, the number of non-zero components for the associated eigenvector is also listed. This helps identify which eigenvectors correspond to the same components. E.g., Table 5.5 (8) shows that there are components of size 26, 5, 4, 4, 3 and 2.

Eigenvectors 2 to 15 and 21 to 29 contain information about the largest component with 26 nodes.

Not all eigenpairs are calculated, unless there are 500 or less nodes. Rather, a number of eigenpairs is requested from the eigendecomposition algorithm, which returns information about how many were successfully found. For example, for more than 1000 nodes, only 200 eigenpairs are requested, and of these it may happen that only 100 are accepted. The remainder have not converged. This behaviour is very dependent on the network, and in general more than enough eigenpairs are requested and returned.

10. The theory of the Normal and CorrAnal spectra allows calculation of upper bounds on distances between subsets. That is, if we select N subsets of nodes= from the network, what is the minimum *geodesic* distance among all pairs? For N=2, this amounts to an upper bound on the diameter. This appears only for Normal and CorrAnal and only for weakly connected networks.

11. For N>2 a large upper bound implies there may be as many as N clusters, while a small upper bound implies that N is too many subsets since it forces short distances. Thus we can use the eigenvalues to estimate how many subsets we should look for in a network without forcing distances that are too short (and hence too many subsets). As a rule of thumb, when the bound for N+1 is more than half the bound for N, we should look for at most N subsets. In Table 5.4, this suggests that we should look for at most 3 clusters. This appears only for Normal and CorrAnal and only for weakly connected networks.

Table 5.4. Partial report for a network which is not weakly connected.

```
MultiNet Normal GRAPH SPECTRUM REPORT ON "Stork2.mnw" 12/04/2003  20:41:11
LINK: "goodfriend"   LINKS =     52   NODES =   44  DENSITY = 0.0269

Normal EIGENVALUES OF "goodfriend"
                  1               2            3
               -1.0          -0.97105      0.95063

PERCENT OF CHI-SQUARED =       1845.25
              2.818%        2.657%        2.547%

NUMBER OF NON-ZERO COORDINATES                                            5
                 18            26            26
```

... (Omitting list of coordinates and permuted adjacency matrix)

```
NETWORK HAS 6 WEAK COMPONENTS.                                            8
 THE 6 LARGEST HAVE # NODES:  26 5 4 4 3 2

 LIST OF ACCEPTED Normal EIGENVALUES FOR "goodfriend" AND NUMBER OF NON-ZERO
COORDINATES                                                               9
  1 =   -1.0        18    16 =   -0.5       19    31 =    0.0       8
  2 =   -0.97105    26    17 =   -0.5       19    32 =    0.0       8
  3 =    0.95063    26    18 =    0.5        8    33 =    0.0       8
  4 =    0.92717    26    19 =    0.5        8    34 =    0.0       8
  5 =   -0.92286    26    20 =   -0.5       19    35 =    1.0      44
  6 =   -0.89513    26    21 =    0.42509   26    36 =    1.0      44
  7 =    0.8732     26    22 =   -0.41985   26    37 =    1.0      44
  8 =    0.77278    26    23 =   -0.32966   26    38 =    1.0      44
  9 =   -0.71018    26    24 =   -0.27134   26    39 =    1.0      44
 10 =   -0.708      26    25 =    0.23762   26    40 =    1.0      44
 11 =    0.70684    26    26 =   -0.20428   26    41 =   -1.0      18
 12 =   -0.61309    26    27 =    0.19429   26    42 =   -1.0      18
 13 =    0.60531    26    28 =   -0.17061   26    43 =   -1.0      18
 14 =   -0.56699    26    29 =    0.08718   26    44 =   -1.0      18
    .   15 =    0.50294    26    30 =    0.0          8

NETWORK IS NOT WEAKLY CONNECTED. NO EIGENVALUE BOUNDS AVAILABLE.         10
```

Report comments if not weakly connected

If the network is not weakly connected, the report is different for section **8** and **9** and sections **10** and **11** are missing, since distance is not defined with more than one component. Table 5.5 shows parts **1-5** and **8-9** for a network which is not weakly connected.

5. Reports that eigenvector 1 and eigenvectors 2-3 contain information on different components.

8. Reports the size of up to 20 weak components, in descending order of size.

9. The table of eigenvalues also includes information about the number of non-zero coordinates in the corresponding eigenvector. This is useful in finding the set of eigenvectors that describe (a) given component(s). Notice that the multiplicity of eigenvalue 1 is 6: the number of weak components. The multiplicity of -1 shows that 5 of them are bipartite (see Section 10).

The **Report** menu item is common to most MultiNet modules, and acts similarly in each one. Clicking on **Report** produces the following choices:

- **Report→View** opens a scrollable, resizable text display window which contains the Report.
- **Report→File** saves the Report as an ASCII text file, with default extension .OUT without displaying it. The first part of the current output file name is derived from the dataset(s), in this example KIDS2. If the file already exists, MultiNet uses a standard method based on a setting made in the Preferences module. The user can direct MultiNet to append the current Report to the current output file automatically. Alternatively, a selection menu appears (Figure 5.12), allowing the Report to a) be appended, b) replace the current output file, c) increment the current output file name (KIDS2.OUT becomes KIDS2__2.OUT), or d) rename the current output file (which opens an Edit window for the new file name). Choices c) and d) cause the new file to become the current output file. Once a Report has been filed, it is immediately available to other Windows programs, such as a printer or text editor.

For more details, see Section 0: Overview and 0.9 Technical appendix.

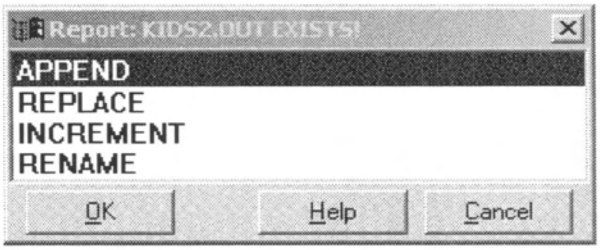


Figure 5.12.:
Standard selection list for **Report→File**
User may select to "Always Append"
in which case this window does not appear,
and the first selection (Append) is automatic

5.4.8 Graphics

Like **Report**, this menu item is common to most MultiNet modules, and acts similarly in each one. Clicking on **Graphics** produces the following choices:

- **Graphics→PostScript** reproduces the current display as a PostScript program. This is a text file which produces graphics with a PostScript interpreter (e.g., printer).

- **Graphics→Bitmap** captures the screen display as a 256-colour bitmap, which is then run-length encoded. The result is a compressed Windows .BMP file.

For more details, see Section 0: Overview and 0.9 Technical appendix.

5.4.9 Explore

This menu item is included as a convenience, and is initially disabled (greyed-out). The **Explore** window may be closed by clicking on the 'X' in the upper right on this item. This may be helpful when using other screen-capture software. Closing the Explore window enables this menu item, so that clicking on **Explore** re-opens the Explore window.

5.4.10 Next

This is another menu item that appears in a number of MultiNet modules. In the Eigenspaces module it is enabled when any node variable is selected. Clicking on **Next** replaces the current node variable with the next one in the list of node variables. This is useful for stepping through the node attributes to look for patterns related to network structure, especially with the 1-D display.

5.4.11 Last

This is another menu item that appears in a number of MultiNet modules. In the Eigenspaces module it is enabled when any node variable is selected. Clicking on **Last** replaces the current node variable with the previous one in the list of node variables. This is useful for stepping through the node attributes to look for patterns related to network structure, especially with the 1-D display.

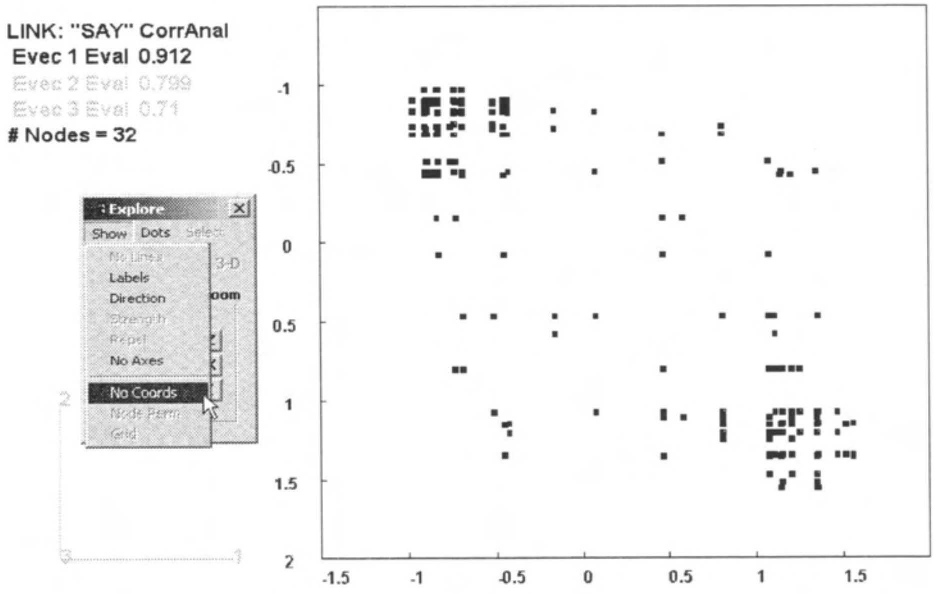
5.4.12 Help

This is a menu item that appears in all the MultiNet modules. Clicking on **Help** opens a selection window which lists all items on the current menu bar. Selecting any of these opens a text window containing details about the menu item. **Help** is also a common button on many other temporary windows, and always provides a context-sensitive description of what the program is doing and what kinds of inputs it expects at the point the **Help** button is pressed.

5.5 Data visualizations II: The 1-D display

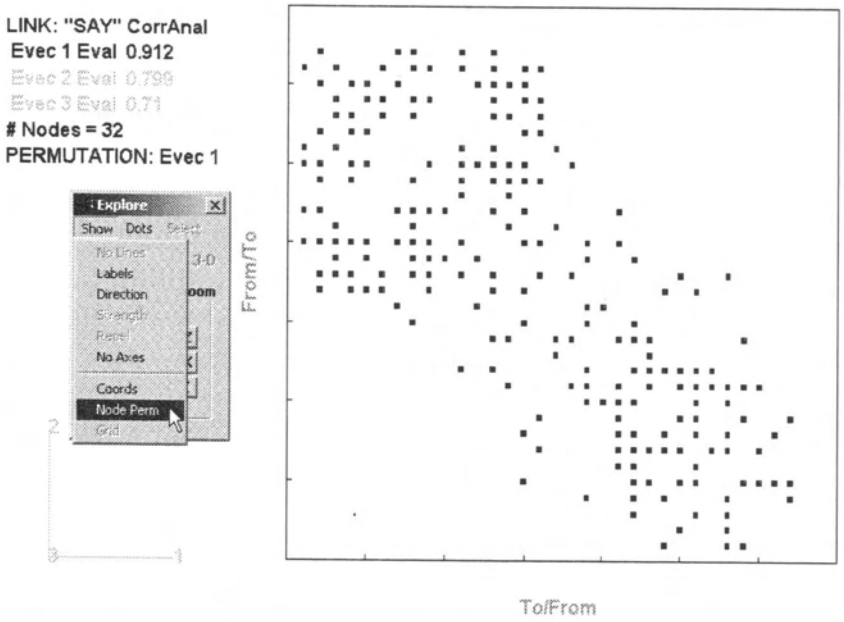
Dimensions→1-D not only reduces the number of active eigenvectors to the first of those currently selected, it also produces a dramatically different display of the network, and enables some extra Explore window options, and disables others.

The main idea behind the 1-D display is the adjacency matrix. The 1-D display is actually a virtual adjacency matrix (only the existing links are shown by using sparse methods). The display uses an off-diagonal dot to represent a link between two nodes, while the nodes are placed along the diagonal. At first the display places nodes at the coordinates of the single active eigenvector. If there is any clustering in the node coordinates, this results in the links being clustered as well, clearly showing the similar patterns of connections that produces the clustering in eigenvector coordinates, especially for Normal and CorrAnal eigendecompositions. Figure 5.13 shows the first SAY CorrAnal eigenvector 1-D display. **Show→No Coords** is a now enabled in the Explore window. The X and Y axes are labelled to show the actual eigenvector coordinates. Clicking on **Show→No Coords** produces the display in Figure 5.14. This actually is a display of an adjacency matrix, with rows and columns permuted by the eigenvector coordinates. The **Explore** window now allows **Show→Node Perm**, which allows the permutation by the values of a node variable.



LINK: "SAY" CorrAnal
Evec 1 Eval 0.912
Evec 2 Eval 0.799
Evec 3 Eval 0.71
Nodes = 32

Figure 5.13. Initial 1-D display of CorrAnal SAY uses eigenvector 1 coordinates to locate links



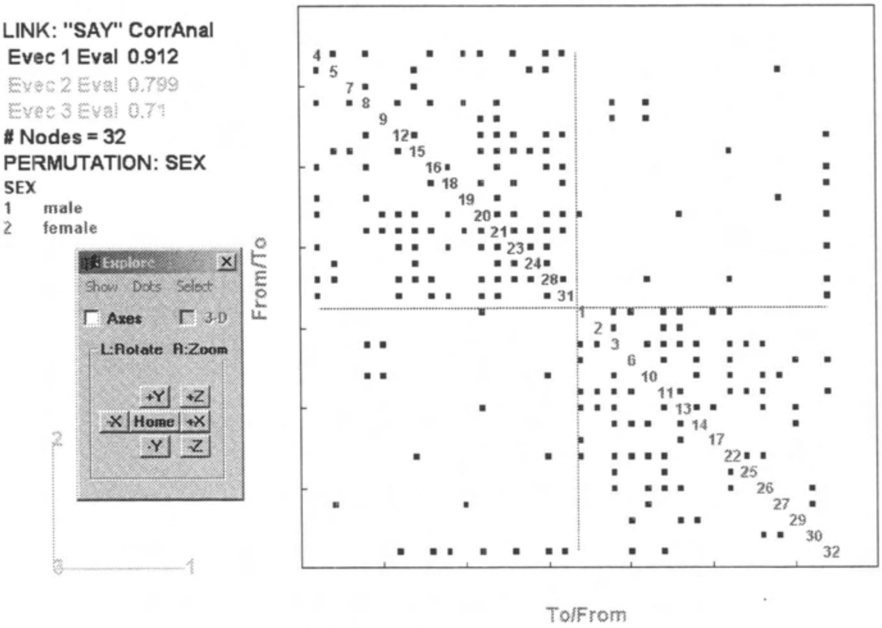
LINK: "SAY" CorrAnal
Evec 1 Eval 0.912
Evec 2 Eval 0.799
Evec 3 Eval 0.71
Nodes = 32
PERMUTATION: Evec 1

Figure 5.14. No Coords display of CorrAnal SAY uses eigenvector coordinates to permute links

At this point, since no node variable has been selected, the result is no permutation at all, although actually this means that the rows and columns are permuted by the ID numbers (lowest to highest) irrespective of how the link data were originally organized. This is actually a useful result, since ID numbers may be related to network structure since they are not usually generated completely randomly. For example, increasing ID numbers can be related to increasing time, so the "unpermuted" adjacency matrix may show some time-related effects.

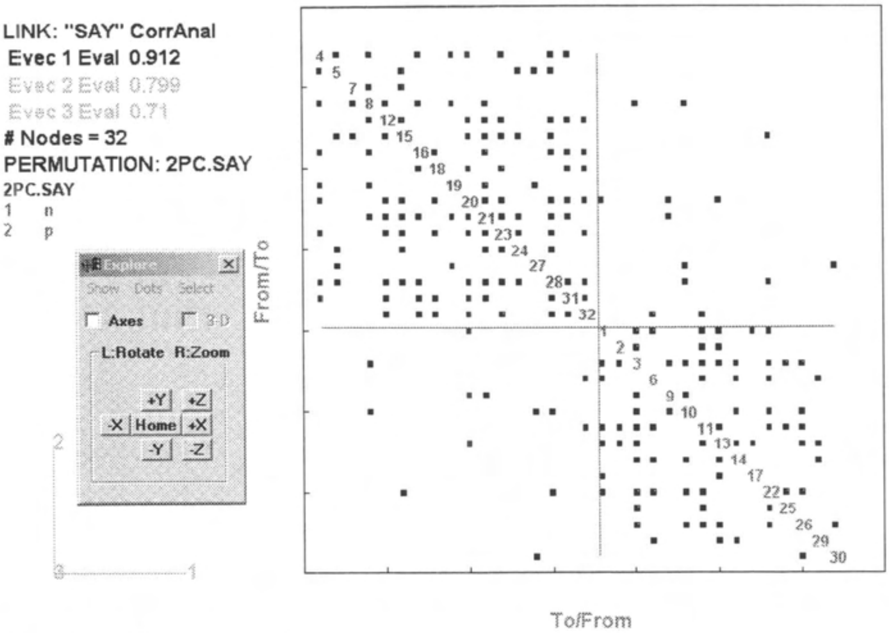
Selecting and integrating a node variable into the 1-D display is conceptually the same as for the 3- or 2-D displays, but the resulting display is again different. The blank diagonal is used to hold the dots representing each node, with colours assigned according to the values of the node variable. The dots may be replaced by ID numbers with **Show→Labels**, or with value labels using **Dots→Values**. As before, a list including the node variable name, values, value labels and assigned colours appears on the left below the informative text display. Figure 5.15 shows the results for CorrAnal SAY with node variable SEX permuting the display, and with **Show→Grid** selected to display the blocks of the matrix with the same values. The **Show→Grid** setting is initially off, but once set or reset retains the setting for the entire Eigenspaces session.

Now **Define→Partition** may be used to create a node variable based on the signs of the active eigenvector, with up to 3 possible values: 'n' for negative signs, 'p' for positive signs, and '0' for coordinates of 0. Since the SAY network is weakly connected, there are no '0' values for eigenvector 1. This new node variable, automatically named 2PC.SAY, provides a different and slightly better permutation of the SAY network (in the sense that chi-squared for this permutation is larger than for SEX): there are more links within the on-diagonal blocks, and fewer in the off-diagonal blocks (Figure 5.16). This is to be expected since this eigenvector has a positive eigenvalue.



LINK: "SAY" CorrAnal
Evec 1 Eval 0.912
Evec 2 Eval 0.799
Evec 3 Eval 0.71
Nodes = 32
PERMUTATION: SEX
SEX
1 male
2 female

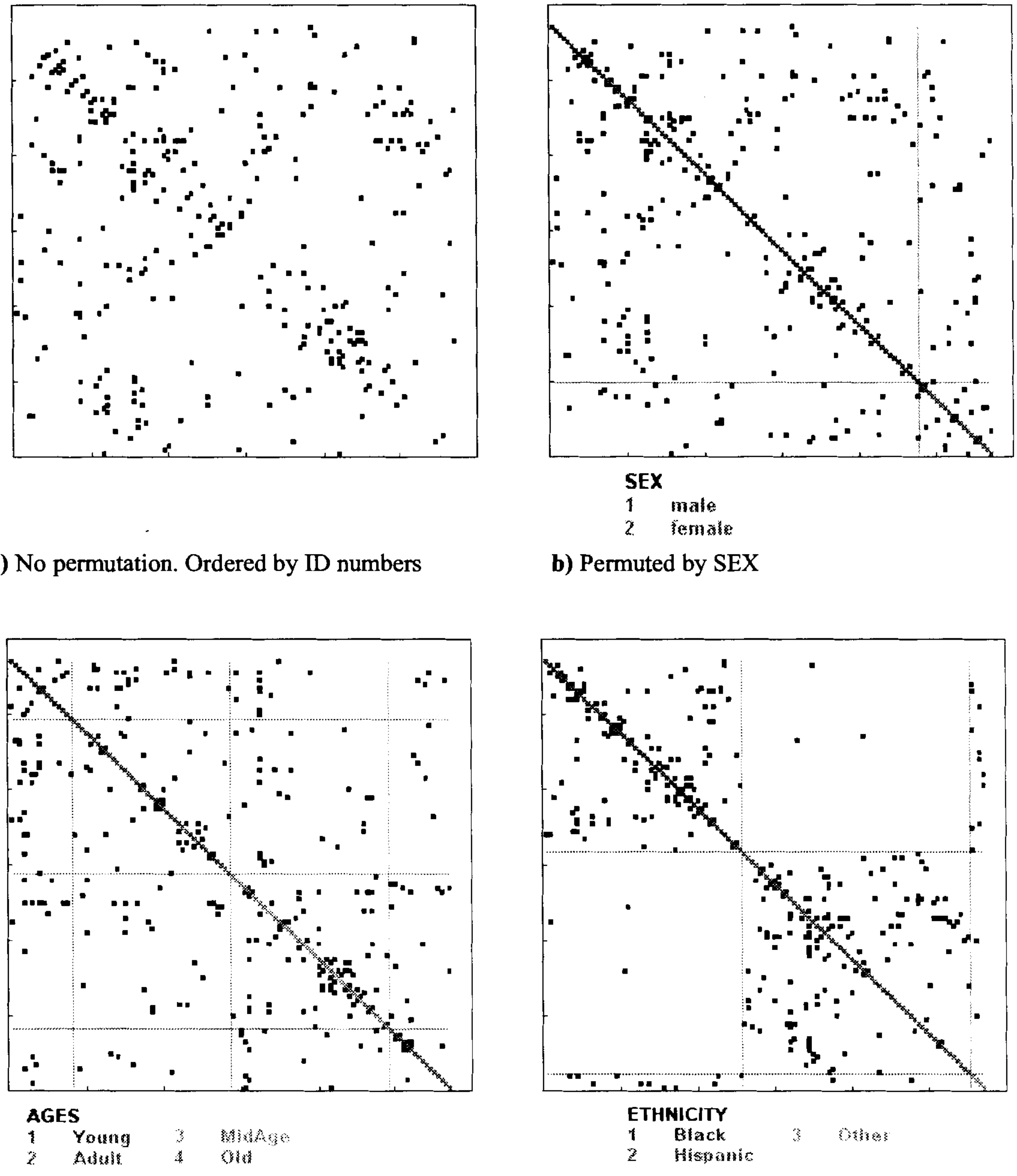
Figure 5.15. SAY permuted by SEX. Most links are from/to same sex. Only 18 between different sexes



LINK: "SAY" CorrAnal
Evec 1 Eval 0.912
Evec 2 Eval 0.799
Evec 3 Eval 0.71
Nodes = 32
PERMUTATION: 2PC.SAY
2PC.SAY
1 n
2 p

Figure 5.16. SAY permuted by 2PC.SAY derived from eigenvector 1 signs. Only 13 links between different signs.

At this point, with **Show→No Coords**, **Show→Node Perm**, and **Show→Grid** all selected, the **Next** and **Last** items on the Eigenspaces menu bar become particularly useful for exploring possible relations between a network link variable and the available node variables.



a) No permutation. Ordered by ID numbers

b) Permuted by SEX

c) Permuted by AGE

d) Permuted by ETHNICITY

Figure 5.17. Four permutations of link variable INJW, showing that this link variable is strongly associated with the node variable ETHNICITY. This network has 114 nodes.

Clicking repeatedly on **Next** (or **Last**) allows for a systematic search through all node variables, while displaying possible relations to the current link variable. Figure 5.17a-d shows the results of such a search for a large network with the link variable INJW. It is clear that there is a strong relationship between INJW and node variable ETHNICITY, but not for the others shown.

5.5.1 Restrictions in 1-D

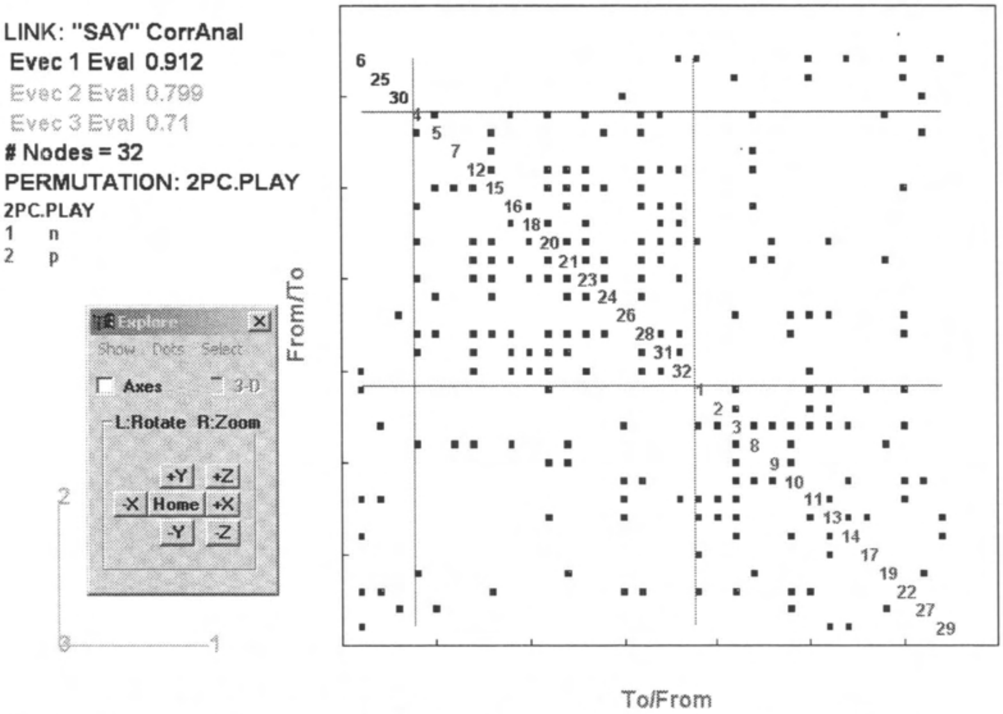
With the 1-D display some new capabilities are available to the **Show** menu item of the **Explore** window, but some other actions are not available or have no affect on the display. Rotation around X, Y, or Z is allowed (although rotation around X only affects the 2nd and 3rd eigenvectors, so does not change the display which only shows the 1st). These rotations produce a display which uses the projection on the X-axis instead of the pure first eigenvector for node placement or permutations. Rotations with **Show→Coords** around Y and Z involves continuous re-calculation of the display scale. With **Show→Node Perm** rotation will not change the display (although it is carried out, and shown in the axis display). All of the **Show** and **Dots** selections are available and meaningful, but the Select menu item is (currently) greyed out. The right-click zoom and pan buttons (currently) produce no effects. With **Axes** checked, only +X , -X , +3 and -3 have a visible affect (although the selected eigenvectors do change).

5.6 Data manipulation II

MultiNet is designed for exploratory data analysis of networks, and the **Eigenspaces** module produces analytical results in the form of eigenpairs for various spectral methods and also provides tools that allow for a certain amount of exploratory visualizations. The data analysis is done by the other modules of MultiNet and Eigenspaces communicates its results to these other modules by the creation of variables. These may be real-valued eigenvectors created by **Define→Variable** which may be used in the Analyse module for either **ANOVA** or **Correlation** analysis. **Define→Partition** produces integer-valued discrete (categorical) variables which are more suitable for the cross-tab analyses available in the **Analysis** module. Either type of variable may also be recoded (transformed) in a number of ways in the **Variables** module.

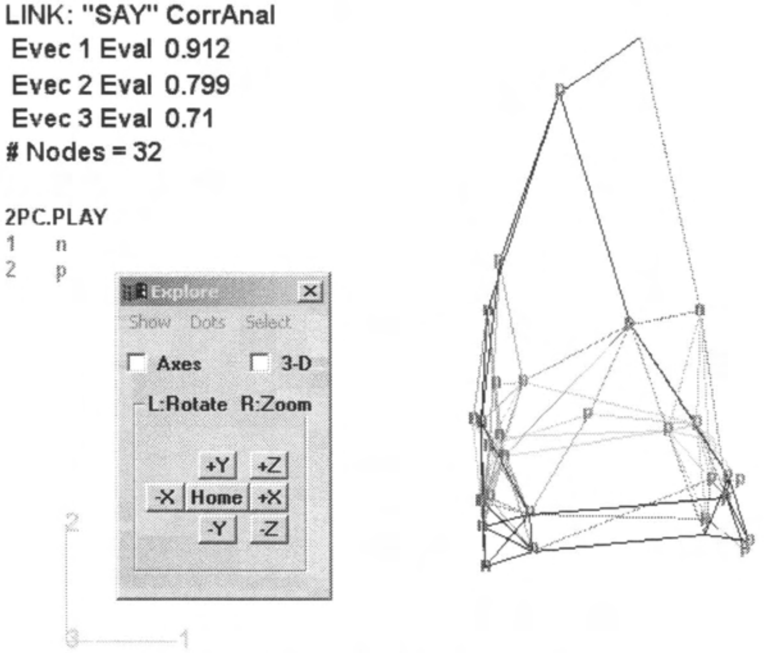
5.6.1 Missing data

MultiNet allows for missing data. For example, Importing data allows for a missing data character. The Variables module can actually mark parts of a variable (node or link) as "missing".



LINK: "SAY" CorrAnal
Evec 1 Eval 0.912
Evec 2 Eval 0.799
Evec 3 Eval 0.71
Nodes = 32
PERMUTATION: 2PC.PLAY
2PC.PLAY
1 n
2 p

a) 1-D display of SAY permuted by 2PC.PLAY, with **Show→Labels**. The missing data is treated as 0 in the permutation, and given the colour black for ID numbers.



LINK: "SAY" CorrAnal
Evec 1 Eval 0.912
Evec 2 Eval 0.799
Evec 3 Eval 0.71
Nodes = 32

2PC.PLAY
1 n
2 p

b) 3-D display with **Dots→Values** for node variable 2PC.PLAY. Node 30 (highest node on Y axis) has missing value, so is left blank

Figure 5.18. Missing data displays

The Analysis module (which can work with up to four variables simultaneously) finds the intersection of all data for which variables are not missing before performing further analysis (sometimes there is no data in the intersection and a warning is given).

When the **Eigenspaces Define** menu item is used to create a new variable, the result has a value for every node which is in the current network (link variable). Since this may not include all the known nodes, the variable is given "missing" values for those nodes which are part of the dataset, but not in the network for which the variable is being defined. The **Variables** module allows missing values to be replaced by 0 so that every node has a value. Note that there may be a number of link variables which together include all nodes, while separately including only subsets of the nodes.

Since it is quite possible for a node variable to have missing values, the **Eigenspaces** module uses a simple method for representing them in any display:

- Nodes which have missing values for the current node variable are coloured black. This applies to both the dot and ID number representation.
- For the case **Dots→Values** which displays value labels, the text is left blank.
- Missing data is treated as 0 in **Show→Node Perm**. Partitions derived from eigenspaces are positive integers, so the missing values are collected in the upper right of the 1-D display.

Examples of these rules is shown in figures 18. Link variable PLAY describes which pairs of children were actually observed to play together (and is substantially different from SAY). PLAY does not include all the children in link variable SAY, so the derived variable 2PC.PLAY has missing data for nodes 6, 25 and 30.

5.6.2 Link values

The eigendecomposition algorithm currently ignores any non-zero values of the link variable by dichotomizing the network into binary form. However other MultiNet modules do not ignore link value, and can create and use real or integer-valued link variables; in fact the dichotomization from non-binary to binary values (or to a discrete range) can be performed explicitly in the **Variables** module. The 2-D and 1-D displays can display link values using a "rainbow" colour scheme, which runs from "cool" to "hot" colours, and uses all available colours (up to 15).

Figure 5.19 shows 1-D and 2-D displays of the p* fit to SAY (from the **Models→Pstar** module) based on 2PC.SAY, where Link value is related to probability level at which a link is predicted. Both **Show→Direction** and **Show→Labels** are selected, where IDLABELS holds

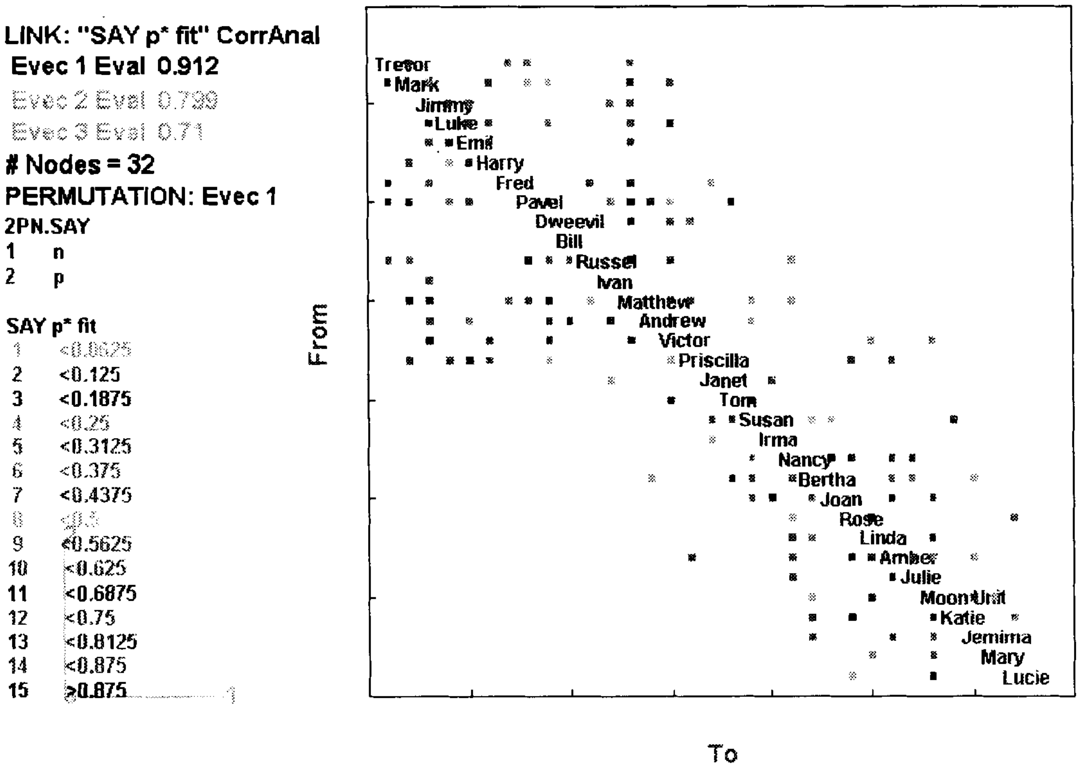
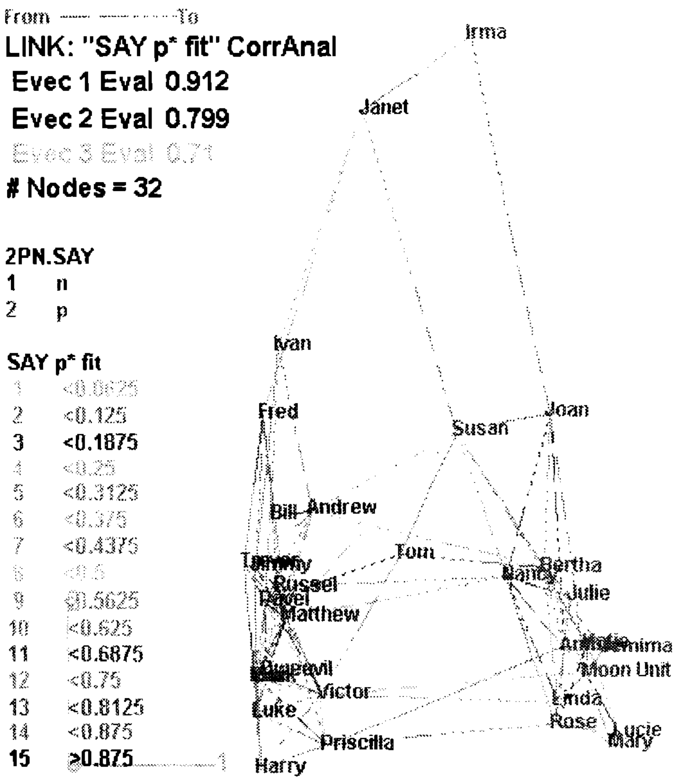


Figure 5.19. Link values of p* fit to SAY using partition 2PN.SAY



Values from p* fit represent probability level at which link is predicted, **Show**ing **Direction** and **Labels** with childrens' names.

Upper: 1-D visualization permuted by 2PC.SAY, which was used in the fit.

Lower: 2-D visualization.

Long dashes show sender nodes. Reciprocated links are solid. The link between Janet and Irma (at top) shows two colours, since the probability level for a link from Irma to Janet is different from the probability level for a link from Janet to Irma.

the (fictitious) names of the children. SAY is not symmetric, so link values may be different in each direction. In the 2-D display, reciprocated links with different values in each direction change colour half-way. The 1-D display also shows the two different link values, but the 2-D display makes the unequal reciprocation more clear. For one-way links there can be only one value, and in the 2-D display the direction is shown with long dashes at the sender end and short dots at the receiver end. This is not a conventional representation of directed links, but avoids the clutter of arrowheads at the nodes. Also, it is immediately obvious whether a link in one-way, and which way it goes, without looking at the ends.

In this example, the Link value is categorical: it is descriptive of how well the p* model predicts each link, and it would not make sense to perform arithmetic on such values (though it would make sense to collect them into fewer or more categories). Another example of a categorical Link variable is "When" (from the 301. dataset) where the 24 categories are the times ate which the interaction took place. In other cases, the Link values may represent continuous measurements such as "Duration" (from the 301 dataset) which measures the length of an interaction. Link values for this type of variable are also referred to as "Weights" or "Strengths" (e.g., in the **Analyse** module, where means and variances of "link strengths" are calculated). For link variables with a large number of distinct values, it can be useful either to form sub-networks for a number of ranges of link values and visualize each separately, or to use Bins or Quantiles in the **Variables** module to produce a small number of categories for network visualization in **Eigenspaces**. While the **Explore** window and the **Analyse** module use the term "Strength" to refer to Link values, it is up to the user to determine whether the link values are to be interpreted as categorical descriptions or numerical strengths.

5.7 Technical appendix

5.7.1 Mathematical definitions

Assume G is an undirected connected loopless graph without multiple edges which is not complete. (The definitions below can be extended to weighted graphs, but for simplicity we will not consider multigraphs here. Assuming G is connected and not complete avoids certain trivial results). G has nodes V and edges E, with $|V| = n$.

The Adjacency (Standard) Matrix **A**(G) of graph G is a binary matrix with

$\mathbf{A}(i,j) = 1$ if i is connected to j

$= 0$ otherwise

The eigenpairs of **A** are $(\alpha_i, \mathbf{a}_i)$ such that $\mathbf{A}\mathbf{a}_i = \alpha_i\mathbf{a}_i$

If G is k-regular, then $\mathbf{a}_0 = \mathbf{1}/\sqrt{n}$, with $\alpha_0 = \max(\alpha_i) = k$.

If G is *bipartite*, then eigenvalues appear as pairs with opposite signs (Biggs, 1993).

The Laplacian Matrix **L**(G) is a matrix with

$\mathbf{L}(i,j) = -1$ if i is connected to j

$\mathbf{L}(i,i) = \deg(i)$ where deg(i) is the degree of node i

$\mathbf{L}(i,j) = 0$ otherwise

The eigenpairs of **L** are $(\lambda_i, \mathbf{l}_i)$ with $\lambda_0 = 0$ and $\mathbf{l}_1 = \mathbf{1}/\sqrt{n}$

The $\mathbf{l}_i$ are mutually orthogonal and $0=\lambda_0 \leq \lambda_1 \leq ,..., \leq \lambda_{n-1} \leq n$

The multiplicity of 0 as an eigenvalue is equal to the number of components in G.

There are a number of other equivalent definitions of **L** the simplest being:

$\mathbf{L} = \mathbf{D} - \mathbf{A}$

where **D** is the diagonal matrix of node degrees (Kirchoff, 1847; Grone et. al., 1990).

The Normal matrix N(G) is $\mathbf{D}^{-1}\mathbf{A}$ so that

$N(i,j) = 1/(\deg(i)$ if i is connected to j

$N(i,j) = 0$ otherwise

N is similar to $\mathbf{M} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$, which is symmetric.

Let $(\mu_i, \mathbf{m}_i)$ be the eigenpairs of **M**.

Then the eigenpairs **N** are

$(\gamma_i , \mathbf{D}^{-1/2}\mathbf{m}_i) = (\nu_i , \mathbf{n}_i)$

The orthonormalization condition is:

$\mathbf{n}_i\mathbf{D}\mathbf{n}_i = \delta_{ij}$

That is, the vectors are orthonormal in the **D** (or χ^2) metric (Richards & Seary, 1995).

The Normal spectrum is referred to as the Q-spectrum in (Cvetkovic, et. al., 1995).

We have $1=\nu_0 \geq \nu_1 \geq ,..., \geq \nu_{n-1} \geq -1$

The multiplicity of 1 as an eigenvalue is equal to the number of components in G.

If G is bipartite, then eigenvalues appear as pairs with opposite signs. Thus -1 is an eigenvalue if and only if G is bipartite.

Adding a constant c to the diagonal of **A**, **L** or **N** does not change the eigenvectors, but adds c to all the eigenvalues. Similarly, multiplying **A**, **L** or **N** by a constant d does not change the eigenvectors, but multiplies all eigenvalues by d.

We may combine these facts to form **C** = (**I** + **N**)/2, which has eigenvalues

$\gamma_i = (1+\nu_i) / 2$

This makes all eigenvalues positive: positive ν_i become closer to 1, while negative ν_i become close to zero. In particular, any eigenvalues of exactly 1 are unchanged. Any eigenvalues of exactly -1 become 0 (Richards &Seary, 1997).

The CorrAnal matrix **C**(G) is a matrix with

$\mathbf{C} = (\mathbf{N} + \mathbf{I})/2$

Let $(\gamma_i , \mathbf{c}_i)$ be the eigenpairs of **C**.

We have $1=\gamma_0 \geq \gamma_1 \geq ,..., \geq \gamma_{n-1} \geq 0$

Then CorrAnal calculates the pairs

$(\gamma_i , \mathbf{D}^{-1/2}\gamma_i \, \mathbf{c}_i)$

The multiplicity of 1 as an eigenvalue is equal to the number of components in G.

Most implementations of Correspondence Analysis (CA) generally remove the (normalized) χ^2 expecteds, which removes the eigenpair belonging to eigenvalue 1 and produces a "trivial" vector of length 0. Also most implementations of CA assume that G is not symmetric and ignore the signs of γ_i, though these are important.

MultiNet actually derives the CA results from the Normal eigenspace, by applying the equation $\gamma_i = (1+\nu_i) / 2$ to the Normal eigenvalues, then ordering the Normal eigenvectors according to descending γ_i (with $\gamma_0 = 1$ rotated to the end). The eigenvectors are also multiplied by γ_i rather than being normalized, to conform with standard CA procedures (Greenacre, 1984, Benzecri, 1992).

5.7.2 The eigendecomposition algorithm

In general, a real symmetric matrix has as many eigenpairs as there are rows (columns).
Let n = number of rows. There are three major problems when the matrix is large:

1. Standard eigendecomposition routines (Press, et. al., 1986) produce *all* eigenpairs which must be stored as 8-byte real. This requires 8*n*(n+1) bytes of storage.
2. Storage as a symmetric matrix requires n*(n-1) memory locations. For **A** these locations may be binary, for **L** they are at least 2-byte integer, and for **N** they are 8-byte real.
3. Standard eigendecomposition routines are n^3 in time, meaning that a matrix of size 2n takes 8 times longer than for size n.

Thus the standard methods, which work with the entire matrix and produce all eigenpairs, must be avoided for large problems. We can avoid problem 2 by using the link list representation of the network, storing *only* the node pairs which are linked (along with the link value, if not binary). We can avoid problems 1 and 3 if there is a method for extracting a small number of eigenpairs, preferably the most important ones.

The Power method is such a method (Hotelling, 1933) for finding a small number of eigenpairs. We can find the eigenpair with largest eigenvalue by:

Start with some random vector **p** normalized to length 1:

Repeat $\mathbf{p'} \leftarrow \mathbf{Ap}$, $\mathbf{q} \leftarrow \mathbf{p}$, $\mathbf{p} \leftarrow \mathbf{p'}$ until **p** is no longer changing in direction.

Then the largest eigenpair of **A** is (**p/q, p**). There are some bookkeeping details: **Ap** uses the link list representation and sparse matrix multiplication, and the entries of **p'** must be adjusted in size after each multiplication (for details see Richards & Seary, 2000), but the method will always work for any matrix without repeated eigenvalues, which is generally the case for social networks.

If we want more eigenpairs, we can iterate with

$$\mathbf{p'} \leftarrow \mathbf{Mp} - \alpha_0 \mathbf{a}_0 \mathbf{a}_0^{T}$$

to get the second, and with

$$\mathbf{p'} \leftarrow \mathbf{Mp} - \alpha_0\, \mathbf{a}_0\, \mathbf{a}_0^{\mathrm{T}} - \alpha_1\, \mathbf{a}_1\, \mathbf{a}_1^{\mathrm{T}}$$

to get the third, and so on, without destroying sparsity. However, we must store the (α_i , $\mathbf{a}_i$) eigenpairs somewhere; the procedure is subject to loss of precision on a computer; and the iterations may converge slowly if α_i / α_{i-1} is close to 1. There are better methods, such as Lanczos iteration (Lanczos, 1961; Parlett *et al.*,1982) which converge very rapidly and do not have problems with loss of precision.

MultiNet uses Lanczos iteration, a generalization of the Power method which allows calculation of a specified number of eigenpairs using sparse methods without loss of precision or orthogonality. Essentially Lanczos iteration works with a sample of the original matrix of size $\sqrt{n}$ which can be orders of magnitude smaller (Golub & Van Loan, 1989). This smaller eigenproblem is solved exactly using standard methods. The resulting eigenvectors are very good starting values for the Power method, so that convergence is rapid for a selected number or range of eigenvalues, with eigenpairs generally returned in order of absolute value of eigenvalue (largest first). Lanczos iteration is currently one of the best methods for eigendecomposition of large systems and has been extensively optimized in various ways. MultiNet uses a variation first developed by Simon (1984) and modified by Berry (1992) which has been further modified by the author to handle the data storage conventions of MultiNet and to produce all four types of eigendecomposition available in MultiNet.

With the version of Lanczos iteration used in MultiNet, the number of eigenpairs requested can be specified, as well as the range in which to search for eigenvalues. This is ideal if we know in advance the range in which important eigenpairs appear, such as for the Normal spectrum, which has important eigenvalues that are near 1 in absolute value. The default number of eigenpairs requested depends on the size of the network. For 500 nodes or less, all are requested. For 5,000 nodes (the current maximum), only 30 are requested. The amount of storage required depends on the number of links, the number of nodes, and the number of eigenpairs requested, and the maximum number of iterations allowed for any eigenpair. For 5,000 nodes this is about 2.5 Megabytes. Not all requested eigenpairs are actually returned. If an eigenvalue is not accurate to within 10^{-13} after the maximum number of steps, it is marked as not converged, and this eigenpair is not "accepted". It is possible that a large eigenvalue could be rejected but a smaller one accepted, so MultiNet checks for this, but it has never happened.

Currently, MultiNet constructs two vectors consisting of the unique pairs of node ID numbers which have non-zero value for the selected link variable. These are equivalent to the upper right triangle of the adjacency matrix, not including the diagonal. The eigendecomposition routine completes the symmetrization on-the-fly. The routine also does the calculations specified by the definitions to produce one of the four desired eigenspaces. In the case of **L**, we are most interested in the smallest (non-zero) eigenvalues, but the eigendecomposition algorithm returns the largest. To get around this, the matrix actually used is **A-D'**, where **D'** is a diagonal matrix of node degrees + Δ, where Δ = maximum node degree. This negates all the eigenvalues and shifts them so that 0 becomes Δ (the maximum possible), and the rest are in descending order. Converting back to the desired λ_i requires subtraction from Δ, giving the desired smallest eigenvalues.

5.8 References

Benzecri, J-P. (1992). *Correspondence Analysis Handbook*, Marcel Dekker Inc.

Berry, M. (1992). Large Scale Sparse Singular Value Computations, *Int. Jnl. Supercomputer Appl.*, **6**(1): 13-49

Biggs, N. (1993). *Algebraic Graph Theory.* Cambridge University Press.

Brandes, U., & Cornelsen, S. (2001). Visual Ranking of Link Structures, *Lecture notes in Computer Science*, **2125**, Springer-Verlag.

Cvetkovic , D., Doob, M., & Sachs, H. (1995). *Spectra of Graphs*. Academic Press.

Chung, F.K.R. (1995). *Spectral Graph Theory*, CBMS Lecture Notes, AMS Publication.

Golub, G.H. & Van Loan, C.F. (1989) *Matrix Computations*, Johns Hopkins Press

Greenacre, M., (1984). *Theory and Application of Correspondence Analysis*. Academic Press.

Grone, R., Merris, R. & Sunder, V. (1990). The Laplacian Spectrum of a Graph. *SIAM J. Matrix Anal. App.* **11**(2): 218-238.

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components, *J. Educ. Psychol.* **24**: 417-441, 498-520.

Kirchoff, G. (1847). Über die Auflösung der Gleichungen, auf welche man bei derUntersuchung der linearen Verteilung galvanischer Ströme gefürte wird, *Ann. Phys.Chem,* **72**: 497-508

Lanczos, C. (1961). *Linear Differential Operators*. Van Nostrand, Dover 1997.

Parlett, B., Simon, H. & Stringer, L. (1982). On Estimating the Largest Eigenvalue with the Lanczos Algorithm. *Mathematics of Computation.* **38**(157): 153-165

Press, W., Flannery, B., Teukolsky, S., Vetterling, W. (1986). *Numerical Recipes*, New York: Cambridge University Press

Richards, W.D. & Seary, A.J. (2000). Eigen Analysis of Networks, *J. Social Structure.* http://www.heinz.cmu.edu/project/INSNA/joss/index1.html

Richards, W.D. & Seary, A.J. (1997) .Convergence analysis of communication networks, in *Organization Communication: Emerging perspectives V*, Ed. G. Barnett, Ablex

Seary, A.J. and Richards, W.D. (2003). Spectral methods for analyzing and visualizing networks: an introduction. In National Research Council, *Dynamic Social Network Modelling and Analysis: Workshop Summary and Papers*, (209-228). Eds. Ronald Breiger, Kathleen Carley and Phillipa Pattison, Division of Behavioral and Social Sciences and Education. Washington DC. The National Academics Press.

Seary, A.J. & Richards, W.D. (2000). Negative eigenvectors, long paths and p*, Paper presented at INSNA XX, Vancouver BC http://www.sfu.ca/~richards/Pages/longpaths.pdf

Seary, A.J. & Richards, W.D. (1995). Partitioning Networks by Eigenvectors. Presented to European Network Conference, London. Published in Everett, M.G. and Rennolls, K. (eds). (1996). *Proceedings of the International Conference on Social Networks,* Volume **1**: Methodology. 47-58.

Simon, H. (1984). Analysis of the symmetric Lancozos Algorithm with reorthogonalization methods, *Lin. Alg. And Appl.* **61**:101-131

6. The Models Module

6.1 Introduction

The **Models** module is the part of MultiNet which implements theoretical models for networks. These models are intended both to allow "curve-fitting" to network measures, and as a confirmatory tool for evaluating and comparing the blockings (partitions) produced by the Eigenspace methods. While there is no shortage of models for common network statistics such as degree distribution (Albert & Barabasi, 2002; Newman et. al. 2001), only a few of them allow comparison by defining a measure for *quality of fit* which is independent of network size. Historically, the first such model for Social Networks was the p_1 model of (Holland and Leinhardt, 1981), but this model has two major drawbacks:

- It does not account for any structures more complex than *dyads*, although it can be extended to include blockings (Wang And Wong, 1987)
- It does not scale well to large networks, since the fit parameters grow as N^2.

More recently, models based on *exponential random graphs* were developed by (Frank & Strauss:FS, 1986; Strauss & Ikeda:SI, 1992; Wasserman & Pattison: WP, 1996). WP described the new family of models as p*, acknowledging the historical connection to the earlier p_1 model (which is included as a subset of p*). The p* family is based on fitting to a network given a set of *local statistics*, which can include not only in- and out-degree, but also various other *triad counts*. These models have recently been extended (Robins & Pattison, 2001) to include counts of 4-paths and 4-cycles, although it is clear that larger local structures (5- and higher paths and cycles) will take a prohibitively large amount of calculation. There is already a fair amount of literature on p* models, both applied and theoretical, and only a very basic review of p* methods is given in this section. A very readable introduction to p* fitting is (Crouch and Wasserman:CW, 1998); the current MultiNet text Report is largely based on the tables described in this article.

Most papers on p* use matrix terminology to define the various local statistics used in a p* fit and to describe the actual (approximate) fitting procedure using Logistic Regression. It was not clear that either of these calculations scale well for large networks until an analysis by this author showed that both could be done efficiently using sparse methods. The analysis led to a computer program called PSPAR, which was made freely available to the social network community in (Seary, 1999). This code was then included into MultiNet in (Seary, 2000), which also introduced new GUI and graphic displays for p* fits. The MultiNet implementation (and PSPAR code) is based on a subset of p* defined in FS as Markov Graphs. **Models →Pstar** is included as the first (and so far only) MultiNet **Models** choice since

- it is a local method that complements the global method of the **Eigenspaces** module.
- it is a network model that allows comparison of networks
- it can be implemented by sparse methods that are efficient in both space and time.

This section assumes some familiarity with the aims and methods of p* fitting, and is mostly concerned with the mechanics of p* fits in MultiNet. A more detailed description of p* fitting is found in (Pattison, Robins and Wasserman:PRW, 1999), which provides the matrix equations necessary to calculate the *change statistics* for various triad, *k-star* and comparative network counts for both directed and undirected networks.

6.2 Implementation

The methods outlined in WP, FS, SI, CW, and PRW all follow a similar pattern:

- A computer program (e.g., PREPSTAR in WP, CW; Matlab routines in PRW) is used to provide change statistics as intermediate files
- These files are submitted to a statistical package such as BMDP, SAS, or SPSS,
- The statistical package calculates the logistic regression of the network to the change statistics, giving a set of logistic parameters and useful (though approximate) measures of goodness-of-fit, standard errors, etc. Exact measures such as actual counts and parameter correlations are also very useful in interpreting the results of a p* fit.

These methods have some obvious drawbacks:

- A number of data transformations are required, including some programming in the language of the statistical package. This is awkward, and it is possible that errors can be introduced during the various intermediate steps. The number of steps also makes curve-fitting far from interactive: trying various combinations of parameters can be cumbersome.
- The calculation of the change statistics, which produces the intermediate files, involves a number of matrix calculations, where the initial matrices may be quite large, and (generally for social networks), quite sparse. These calculations are usually inefficient (slow) and generally destroy sparsity (large).
- The intermediate files can get very large (a storage problem), since they generally require O(N^2) terms for each parameter to be fitted. For example, with N=1000 and 15 parameters, an intermediate file can take up 30 Mbyte (or more, depending on formatting). A simple block

model can double this.

- A similar amount of space is required to do the logistic regression (a runtime problem). Although the actual matrices used in a logistic regression are much smaller – typically $O(P^2)$ -- where P is the number of parameters), the intermediate data must either fit into memory, or be read in repeatedly for calculation of the parameter matrices.

Implicit in the above is the assumption that we are dealing with network data in the form of adjacency matrices. The care and handling of such matrices for large sparse networks is quite a chore, and it is easy to miss errors in the original data itself. An earlier text-based DOS program called SPAR (Scary, 1999) was designed to avoid these problems. Adjacency matrices are not used. There are no intermediate files. No programming in a statistical package is required. All network calculations use sparse methods (fast and small). The user proceeds directly from reading a data file to selecting triad statistics, to examining fit parameters and other useful results. A network may be repeatedly analysed with different parameters (and block-models) in an interactive way. Typical social networks (N < 200) may be analysed in a couple of seconds. Larger networks may take a few minutes. It was always intended that the code used in SPAR would become part of MultiNet, but two problems needed to be addressed: how to represent the results of p* fits textually and graphically. The text report was based on the format used in CW. The graphic display was an adaption of the Eigenspaces 1-D display to show how well each link is predicted.

The sparse matrix implementation of p* is based on the simple observation that <u>all</u> of the triad change statistics can be calculated for <u>each</u> node <u>independently</u>. (In fact, they may be calculated <u>in parallel</u>, but most desk-top computers don't allow for that). One other observation is that <u>repeating</u> the triad calculations — rather than storing them — saves a <u>lot</u> of space. Whether this is a good strategy depends on how much work these calculations take, and how often they have to be repeated. Fortunately, the triad change statistics are fast and easy to calculate for the classic Markov Graph model (FS). This model is also one for which the conditional dependencies are well-understood, and so the current MultiNet implementation allows p* fit <u>only</u> to these (see below for more details on these 15 statistics). Also fortunately, the logistic regression loop which finds p* fit parameters uses a Newton-Raphson algorithm which converges quadratically fast, meaning that only a few (3-8) iterations are generally needed.

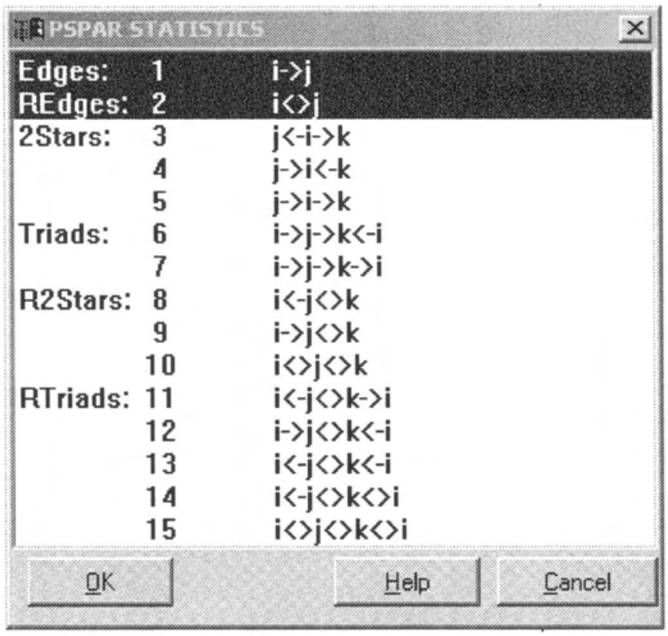
6.3 Parameters

MultiNet currently allows fitting to the 15 (non-null) triads described by Frank and Strauss (1986) and Pattison, Robins, and Wasserman (1999), and adds the block-model parameters described in Wasserman and Pattison (1996). There is some variation in notation among these papers, and MultiNet adds one other. The notation is intended to be descriptive and helpful: the symbol **I** is the important one in each case. The correspondences among the notations are all one-to-one, as described in Table 6.1 (see also Figure 6.1: screen shot of triad selection). Triads with some reflexivity have (purely descriptive) names that start with an R. The order of parameters has no analytic implication; in MultiNet they are roughly ordered by number of edges.

Table 6.1: Notations for p* parameters (R* = reflexive)

Description	**MultiNet**		**FS(U|MAN)**	**PRW**
Choice	**Edges:**	**1 i->j**	**012**	**T15_1**
Mutuality	**Redges:**	**2 i<>j**	**102**	**T11_11**
out-star	**2Stars:**	**3 j<-i->k**	**021D**	**T12_11**
in-star		**4 j->i<-k**	**021U**	**T14_11**
mixed-star		**5 j->i->k**	**021C**	**T13_11**
transitivity	**Triads:**	**6 i->j->k<-i**	**030T**	**T9_111**
cyclicity		**7 i->j->k->i**	**030C**	**T10_111**
R* 2-stars	**R2Stars:**	**8 i<-j<>k**	**111U**	**T8_111**
		9 i->j<>k	**111D**	**T7_111**
		10 i<>j<>k	**201**	**T6_1111**
R* triads	**Rtriads:**	**11 i<-j<>k->i**	**120U**	**T5_1111**
		12 i->j<>k<-i	**120D**	**T4_1111**
		13 i<-j<>k<-i	**120C**	**T3_1111**
		14 i<-j<>k<>i	**210**	**T2_11111**
		15 i<>j<>k<>i	**300**	**T1_111111**

The MultiNet notation is a compromise between compactness and convenience. As will be seen later, the triad numbers (1-15) are used to indicate the parameter fits and correlations in the p* fit report. Figure 6.1 shows how these choices are presented to the user in a multiple selection window. In this example, MultiNet will fit to '**Edges: 1 I->j**' and '**REdges: 2 i<>j**'. (These are also shown as Choice and Mutuality in Table 6.1). Selections are made using standard Windows methods. Initially, all triads are selected. By clicking on **Edges** all others are de-selected. Clicking on an item while holding down the Ctrl key toggles a selection. Clicking on an item while holding down the Shift key selects all items between the current item and any previously selected one. In this case, to select both **Edges** and **REdges**, either key could be used since there are no other items between these two.



```
Edges:    1    i->j
REdges:   2    i<>j
2Stars:   3    j<-i->k
          4    j->i<-k
          5    j->i->k
Triads:   6    i->j->k<-i
          7    i->j->k->i
R2Stars:  8    i<-j<>k
          9    i->j<>k
          10   i<>j<>k
RTriads:  11   i<-j<>k->i
          12   i->j<>k<-i
          13   i<-j<>k<-i
          14   i<-j<>k<>i
          15   i<>j<>k<>i
```

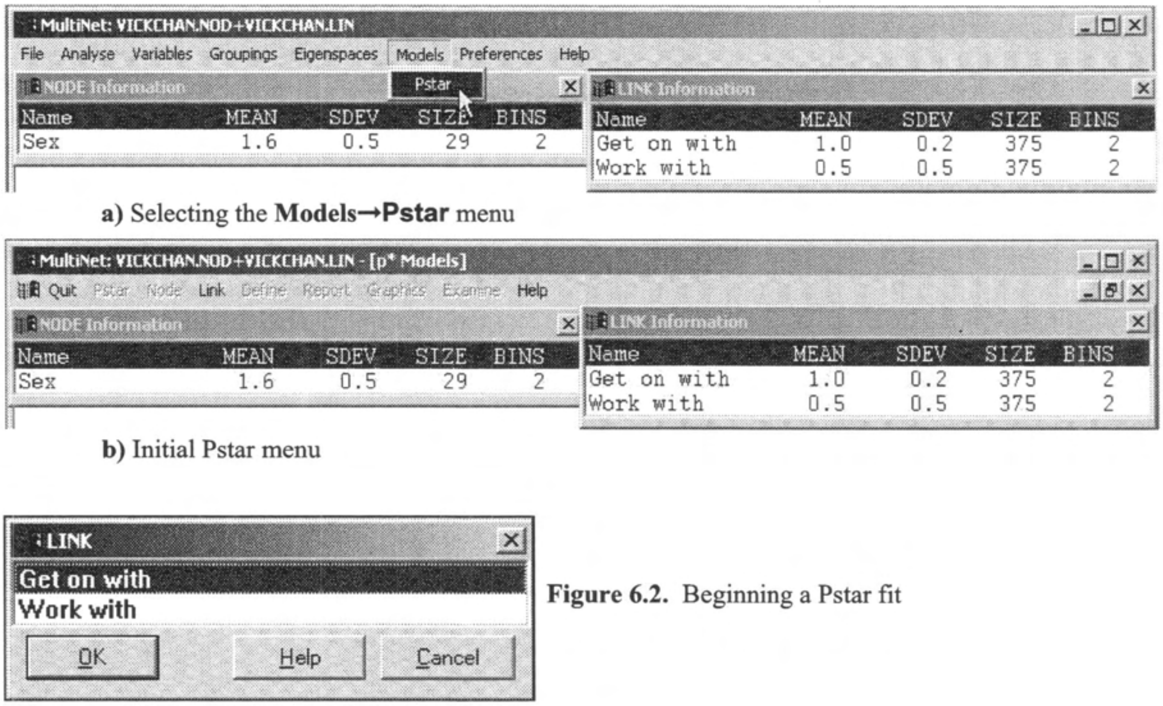
Figure 6.1. Selecting Pstar statistics

At this writing, only these 15 triad statistics are supported. Node level (heterogeneous) parameters such as the p_1 attractiveness and expansiveness are not available because they do not scale well to large networks. Other parameters (such as degree centralisation) are not included since they do not have well-defined dependency graphs, and so the Hammersley- Clifford theorem cannot be applied. Although k-stars would be easy to include, there has not been a consensus on how to select and report them (and there can be a lot of them). Recent work has shown how to implement 3-paths and 4-cycles (Pattison and Robins, 2000) although they are not currently included. Support for affiliation networks (bipartite graphs) and valued nodes can also be implemented efficiently using sparse methods (Seary, 2002), and will be included soon. Given these restrictions, even for small networks (<100), the MultiNet Pstar version makes it very easy way to do p* fitting. Because it is a familiar dataset in p* literature, the next sections will show how to repeat most of the model fits done in (WP), using the same Vickers and Chan data and link variable “Get on with”.

6.4 Example: the Vickers and Chan dataset

Figure 6.2a shows the MultiNet main menu after the VICKCHAN Node and Link files have been successfully imported, and **Models→Pstar** is being selected. Figure 6.2b shows the initial **Pstar** menu bar: everything except **Quit**, **Link** and **Help** is greyed out (disabled). Selecting **Help** will produce descriptions of each of the sub-menu items. **Quit** will leave the Pstar menu entirely. **Link** is where to start; it selects a network (Link variable) for p* fitting. Pressing **Link** produces the selection window shown in Figure 6.3c. For the Vickers&Chan data there are two choices: “Get on with” and “Work with”. The examples will use “Get on with” which is highlighted in Figure 6.3c.

Once the Link variable has been selected, both **Node** and **Pstar** menu items will be available. **Node** is used for blocking, which will be described later. Clicking **Pstar** brings up a menu of triad statistics to fit to (described earlier). After selecting Choice and Mutuality (‘**Edges: 1 i>j**’ and ‘**REdges: 2 I<>j**’) and clicking **OK**, the program very quickly produces the display shown in Figure


MultiNet: VICKCHAN.NOD+VICKCHAN.LIN
File Analyse Variables Groupings Eigenspaces Models Preferences Help
Pstar
NODE Information
LINK Information

Name	MEAN	SDEV	SIZE	BINS
Sex	1.6	0.5	29	2

Name	MEAN	SDEV	SIZE	BINS
Get on with	1.0	0.2	375	2
Work with	0.5	0.5	375	2

a) Selecting the **Models→Pstar** menu

MultiNet: VICKCHAN.NOD+VICKCHAN.LIN - [p* Models]
Quit Pstar Node Link Define Report Graphics Examine Help
NODE Information
LINK Information

Name	MEAN	SDEV	SIZE	BINS
Sex	1.6	0.5	29	2

Name	MEAN	SDEV	SIZE	BINS
Get on with	1.0	0.2	375	2
Work with	0.5	0.5	375	2

b) Initial Pstar menu

LINK
Get on with
Work with
OK
Help
Cancel

c) Selection window for Link variable

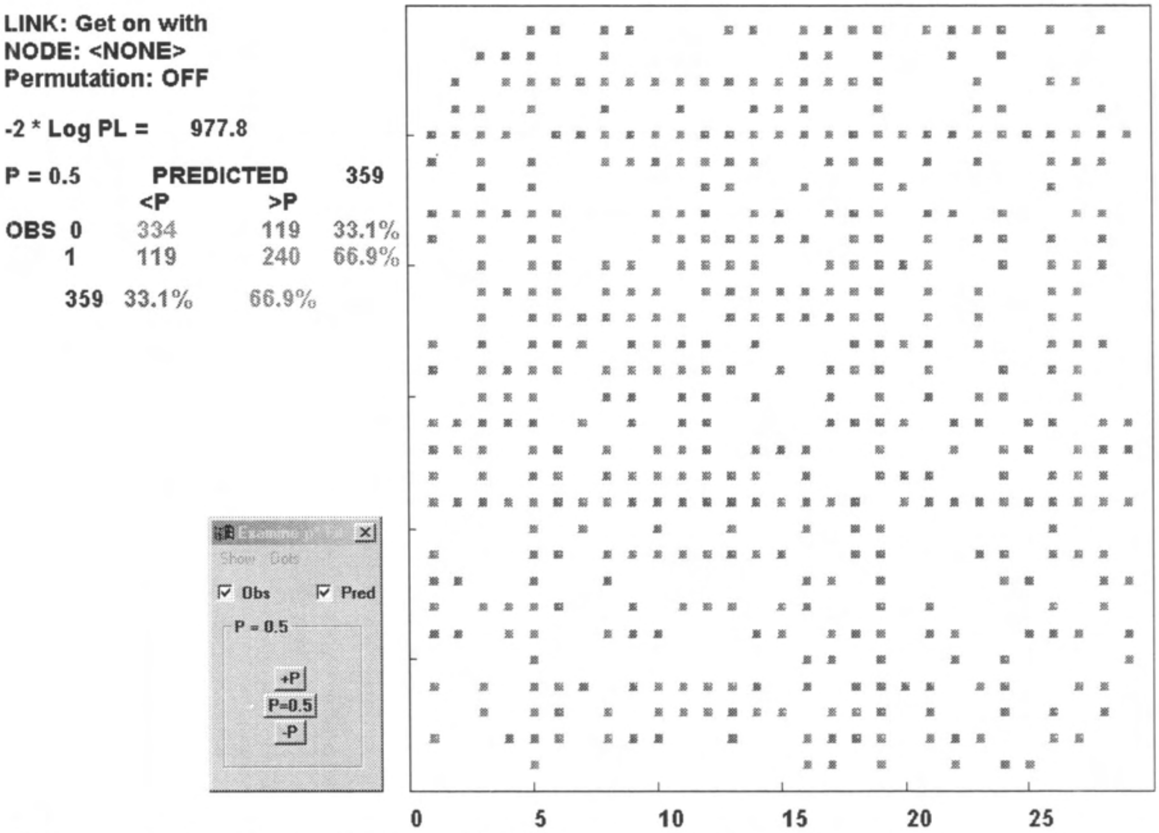
Figure 6.2. Beginning a Pstar fit

6.3. The fit corresponds to Model 2 in WP. At this point, the **Define**, **Report** and **Graphics** menu items are enabled. As in other MultiNet modules, **Graphics** is used to produce PostScript or Windows Bitmap copies of the graphic display and **Report** is used to see or save the textual results. **Define** will be described below.

The graphic on the right side of Figure 6.3 is the 1-D display of an adjacency matrix, similar to the display used in the Eigenspaces module. The numbers on the bottom of the matrix are column numbers of the matrix, but have no use other than indicating the number of nodes. The dots inside the box represent three kinds of links:

- **green:** a link exists and was successfully predicted at the selected probability level (default is P=0.5).
- **blue:** a link exists but was not successfully predicted: False Negative.
- **red:** a link does not exist but was falsely predicted: False Positive.

As mentioned, the initial (default) probability level is P=0.5, meaning that if the p* fit predicts a link with probability greater than or equal to 0.5, it is counted (and shown as either green or red). This brings us to the '**Examine p* fit**' window at the lower left. This window is a "control centre" for examining the p* fit. Clicking on the '**P-**' button lowers the probability level by 1/16 (to a minimum of 1/16, or P=0.0625). This generally means more green (successful) and less blue



```
LINK: Get on with
NODE: <NONE>
Permutation: OFF

-2 * Log PL =     977.8

P = 0.5        PREDICTED         359
                 <P          >P
OBS  0          334         119   33.1%
     1          119         240   66.9%

     359  33.1%       66.9%
```

Figure 6.3. The Pstar graphic display, with **Explore p* fit** window

(false negative) links predicted — but at the cost of more red (false positive) predictions. The '**P+**' button does the opposite, raising the probability level (to a maximum of P=0.9375). Playing with these buttons can help in understanding *what* network features p* is fitting to (especially with blockings — see below). The '**P=0.5**' button restores the default probability level of **P=0.5**. The check boxes '**OBS**' and '**PRED**' allow you to look at *only* the observed data (**PRED** not checked) or *only* the p* fit (**OBS** not checked).

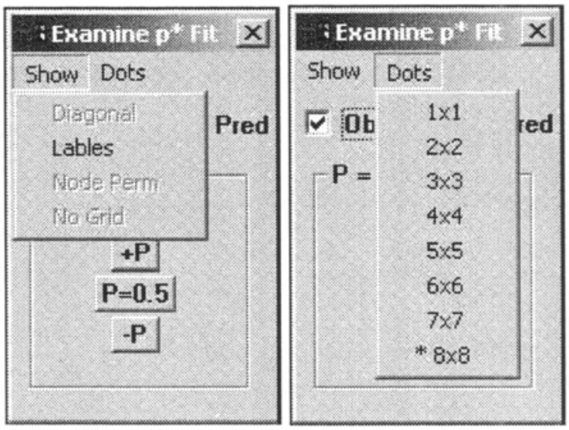
The upper left corner of the display (Figure 6.3) summarizes the p* fit results. Shown are:

- The Link (network) variable
- The Node (attribute) variable (in this case **<NONE>**)
- The Permutation state (in this case **OFF**. This is described further below)
- The value of -2*Log Pseudolikelihood for this fit. This is the p* quality of fit measure.

Below this is a table giving more details on the fit. The rows correspond to the OBSERVED links in the data, while the columns correspond to the PREDICTED links. The same colouring scheme is used here. The total unobserved (and unpredicted) links are shown in the upper left in grey (334). The total number of false negatives are shown in blue (119), of false positives in red (119), and of correctly predicted links in green (240). Below the table is shown the total number of observed links in black

(359), with the percentages (of this total) of false negatives (33.1%), and correct predictions (66.9%) in blue and green. To the right is a column with the total number of predicted links in black (359), and below this the percentages (of this total) of false positives (33.1%) in red and correct predictions (66.9%) in green. Note that the unobserved (and unpredicted) links shown in grey are not include in these percentage calculations. For large sparse networks, there are a lot of these, and including them can give a distorted picture of how successful the fit was.

The '**Examine p* fit**' window also allows you to control the size of the dots (from the '**Dots**' menu item), and to add other information to the graphic adjacency matrix display (Figure 6.4). At this point in this example, all items of the '**Show**' menu are greyed out (disabled) except '**Labels**', which puts the ID numbers (or names) down the diagonal of the graphic display. To turn this off, pick '**Show→No Labels**'. The other selections under '**Show**' become very useful with blockings.



a) **Show** menu b) **Dots** menu

Figure 6.4 Examine p* fit menu choices

As well as this summary display, there is a textual report showing the actual parameter values and other details. For these, press the **Report** menu item and select **View**. The text window is shown in Table 6.2. The report includes everything (except the graphics) shown in the display and a lot more:

> Number of iterations, File names, Link variable name, number of links and nodes, Node variable and blocking array (if any), -2*Log Pseudolikelihood, Goodness of fit and model Chi-squared (see CW and appendix for descriptions of these), absolute and squared residuals, parameter numbers and block numbers (if any), parameter values, along with Pseudolikelihood (PL)-"Standard error", PL-Wald statistic, PL-Wald significance, exp (parameter value), and Counts and Errors.

The PL-Wald statistics are estimates based on the assumption that the likelihood function is distributed as Chi-square. If this were the case, then the Wald statistics show how much each parameter contributes to -2*LogPL. The importance of this contribution is given in the "p" column: if p<0.01, then the PLWald statistic is significant for this parameter; for p>0.01 and higher, the parameter does not make a significant contribution to -2*LogPL and may be dropped from the model.

The Counts and Errors columns can be valuable in determining why a fit has failed (about which more later). Following the parameter values is a table of correlations among all the parameters. This table should always be examined for possible high correlations between (among) two (or more) parameters, and can be very useful in determining why a fit fails. Finally, the report includes a table of tables of all fit results at all probability levels from P=1/16 to P=15/16. (See Appendix).

6.5 The Report

The Report (Table 6.2) is very similar to that shown in CW. Other statistics will be added if users request them. This section describes the reported results, with some comments on their usefulness. The **bold** numbers have been added to the Report in Table 6.2.

- 1) -2 * Log Pseudolikelihood (-2*Log PL) is the same as that defined in CW. This measure appears in the p* literature as the most important result for evaluating the fit. One goal of p* fitting is to get the lowest such value with the fewest parameters (so -2*LogPL is a "badness of fit" measure).
- 2) Goodness of Fit and Model Chi-squared are as defined in CW. These do not appear much in the p* literature.
- 3) The table of Observed vs Predicted is also as defined in CW. The cut-off is P=0.5: a calculated frequency of 0.5 or more is considered a "hit". The percentages of Correct, False Negative and False Positive are exactly as described above.
- 4) Absolute residuals are as defined in WP. Squared residuals are also included. These can be useful in comparing fits with very close -2*Log PL.
- 5) The table of parameters, blocks, parameter values (b), PL-"Standard error" and Wald statistic, and exp(b) appear as defined in CW. The "p" column shows the significance of the PLWald statistic for each parameter. The columns headed "Count" and "Error" aid in tracking convergence problems (see Error Traps). Counts are not triad counts (except Choice and Mutuality), but rather the diagonal of the inner products matrix that the logistic regression actually fits to. Errors are the deviation of the fit from these values.
- 6) The correlation matrix. It is recommended that this table be examined closely.
- 7) A table of 5 rows by 3 columns, each item of which is a table of Observed vs Predicted links, for all probabilities from P=0.0625 to P=0.9372 (see Appendix).

Both the table of parameters and correlation matrix identify parameters by number and block number (if any). Any global parameter has a blank block number.

Table 6.2. Report for a simple Pstar fit. Sections 1-6 described in text. (see Figure 6.13 for 7)

```
MultiNet PSTAR REPORT ON "VICKCHAN.NOD"  4/05/2004  18:14:57
       RUN #1 ITERATIONS =  4

LINK: "Get on with"  LINKS =   359 NODES =   29  (DIAGONAL NOT INCLUDED)
                                                                Section
 -2 Log PseudoLikelihood =          977.812                           1
         Goodness of Fit =          812.000                           2
       Model Chi-squared =          147.859     df = 2                2

FIT AT P = 0.5                      RESIDUALS
                 PRED         359                                     3
           <P    |   >P             Absolute =          334.588       4
       ------------------           Squared  =          167.294       4
    0 |     334|      119  33.1%                                      3
OBS   |--------|--------   -----                                      3
    1 |     119|      240  66.9%                                      3
      |        |                                                      3
359   |   33.1%|   66.9%

PARM BLOCK    b     "Std.Err"   PLWald   p(df=1)  exp(b)  Counts    Errors 5
   1  -1.0320     0.1068   93.4479   < 0.01    0.36     359   0.00000
   2   1.7335     0.1548  125.3843   < 0.01    5.66     240   0.00000

CORRELATION MATRIX:                                                   6
  PARM              1         2
        BLOCK
     1        1.00000  -0.68959
     2       -0.68959   1.00000
```

Since this part of MultiNet is primarily concerned with curve-fitting to the p* model, **Report** behaves somewhat differently from the other modules. When a Link variable is initially fitted, the Report consists of the results of the single fit, with a simple **Quit** selection. The results of the initial fit (e.g., PLWald statistics, Errors, Correlation table) may suggest adding or removing parameters. The modeller may also want to compare fits with and without blockings based on Node attributes, or simple versus complex blockings. To allow easy comparisons of various fitting attempts, the program retains all reports including the initial fit. Reports of succeeding fits produce a Multi-View window, with additional '**Last**' and '**Next**' selections. Each fit attempt is numbered with a RUN # to simplify navigation through the reports. When a Report involves many fits, the two menu items '**Last**' and '**Next**' allow for paging through the results of the present and earlier fit attempts. If **Report →File** is chosen, all the fit results are saved, separated by Page Break characters. The results are stored in an ASCII disk file in the current directory with the same name as the Node and Link files (or system file), and with the extension '.OUT'.

6.6 Block-modelling

MultiNet allows for block-modelling by using the actor attribute information in the Node variables. You may choose any categorical Node variable to define a block structure, which then allows MultiNet to fit to both the whole network and subsets of actors which share certain Node attributes. The method employed is quite flexible and will be shown in a number of ways.

6.6.1 Simple blocks

MultiNet assumes (by default) that a Node attribute will be used to define a simple cohesive block structure, since this type of modelling is very common. Thus, if an attribute has four categories, MultiNet will present the following default block structure:

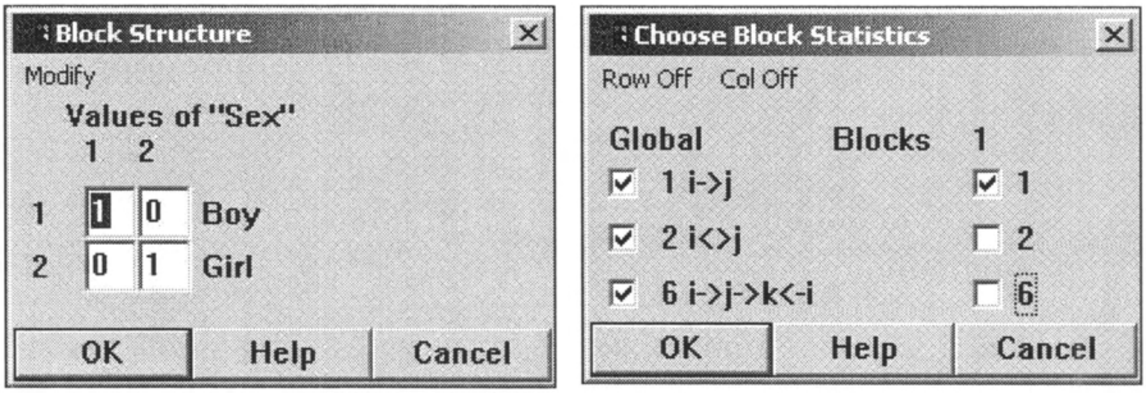
```
1 0 0 0
0 1 0 0
0 0 1 0
0 0 0 1
```

This block structure assumes that <u>any</u> pair of actors sharing the <u>same</u> Node category are considered as belonging to the same block. Since there is only one type of block (with value 1), <u>any</u> pair of actors with the same attribute are counted together. This is the type of block structure assumed for the example given in CW where fitting is done to the block parameters "Choice within Blocks" and "Mutuality within Blocks". This example has two possible values for the Block attribute (boy and girl), and so the block structure is:

```
1 0
0 1
```

MultiNet makes it easy to do this kind of fitting by allowing a click anywhere in the blocking display to change the value by keyboard entry (Figure 6.5a). Tab and Shift-Tab may also be used to get to any block. To demonstrate this with the Vickers and Chan data, the Node variable "Sex" has been selected as the blocking variable. This was followed by clicking on **Pstar** and selecting Choice and Mutuality, as well as '**Triads: 6 l→j →k←l**' (Transitivity). This time, since a blocking variable (Sex) has been selected, there is a little more work to do. Before the fit begins, MultiNet presents a blocking matrix as shown in Figure 6.6a. The rows correspond to Links FROM and the columns to Links TO actors with each category in the Node variable (in this case "girls" and "boys"). For this example, the diagonal blocking is accepted by clicking '**Okay**'.

Now there is one other set of choices to make (Figure 6.5b). Of the triads originally selected, and with the defined blocking, it is possible to fit some parameters globally (over the whole network) and others locally (only within certain blocks). This is demonstrated (and reproduces WP Model 15) by clicking *off* the block 1 choices for parameters 2 and 6 (Mutuality and Transitivity). The results are shown in Figure 6.6.



Block Structure

Modify

Values of "Sex"

	1	2	
1	1	0	Boy
2	0	1	Girl

Choose Block Statistics

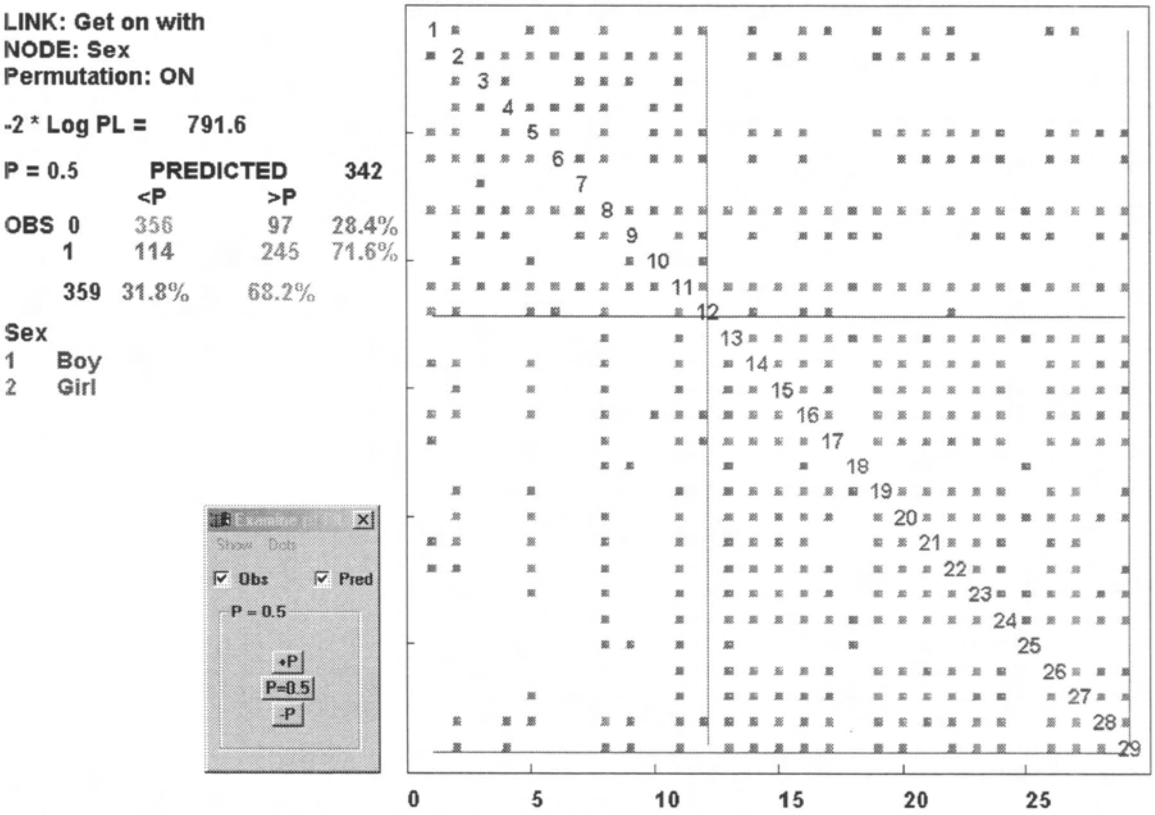
Row Off Col Off

Global	Blocks 1
☑ 1 i->j	☑ 1
☑ 2 i<>j	□ 2
☑ 6 i->j->k<-i	□ 6

a) Defining Block structure

b) Selecting global and block parameters

Figure 6.5. Control windows for Block Models



```
LINK: Get on with
NODE: Sex
Permutation: ON

-2 * Log PL =     791.6

P = 0.5        PREDICTED        342
               <P          >P
OBS  0         356          97    28.4%
     1         114         245    71.6%

     359     31.8%       68.2%

Sex
1    Boy
2    Girl
```

Examine

☑ Obs ☑ Pred

P = 0.5

Figure 6.6. Simple block model. **Labels**, **Node Perm** and **Grid** were enabled from the **Show** menu.

Compare Figures 6.3 and 6.6. Notice that -2*Log Pseudolikelihood is shown as smaller in Figure 6.6. However, there are now more parameters (4 instead of 2). Are the extra parameters worth it? To answer this, we need to know how the difference between the two -2*Log Pseudolikelihood is distributed, and this is currently unknown. However, if we can assume a Chi-squared distribution with degrees of freedom equal to the difference in the number of parameters, then the 4-parameter blocking model is significantly better.

There is more to Figure 6.6 than this. The diagonal labels ('**Labels**'), permutation ('**Node Perm**') and '**Grid**' items were selected from the '**Show**' menu of the '**Examine p* fit**' window. Notice that Permutation is shown as ON. The rows and columns of the adjacency display have been permuted by the values of the Node variable Sex (1=boys and 2=girls, so boys are in the upper left). This is also shown by the value label display on the right (below the P=0.5 table) and by the labels along the diagonal coloured by these Node categories. It is clear just by looking at this display that there is an important connection between the Node attribute "Sex" and the Link attribute "Get on with". The ability to permute the adjacency display by the value of Node variables is very useful (a similar display is also available under **Eigenspaces**). One can very quickly get a feeling for which (if any) actor attributes are related to network structure just by turning permutation on, and selecting each Node variable in turn. The ones that show obvious block structure are candidates for p* fits with blocking (among other things!)

Table 6.3 shows the result of this fit (excluding the 15 tables at various probability levels). This Report shows the RUN# as 2, since a new Link variable has not been chosen Since this is a block model, the report contains additional information: the Node variable name and block model are shown in section 0. The global parameters (choice, mutuality and transitivity or 1, 2 and 6) are listed with Block set to blank, while the "Choice within Block" parameter is listed with parameter 1 and Block set to 1. This is also done for the correlation matrix. No correlations are near 1 or -1, and all the PLWald statistics are significant, so it is worth checking the improvement in -2*LogPL. This value has decreased by more than 186 with only two additional variables. If we assume that -2*LogPL is distributed as Chi-squared with the degrees of freedom equal to the number of extra variables, then this is clearly a significantly better fit. A large part of the improvement comes from the block structure, which counts "Choice within Blocks" where most of the counts actually are, because the choices are indeed largely boy-boy and girl-girl. It is this ability to compare counting within different permutations that makes the p* model a useful confirmatory tool for the partitions that are produced by the **Eigenspaces** module.

Table 6.3. Report for simple block model. Sections are labelled as in Table 6.2.

```
MultiNet PSTAR REPORT ON "VICKCHAN.NOD"  5/05/2004  13:48:53
RUN #2 ITERATIONS =  6

LINK: "Get on with"  LINKS =   359  NODES =   29  (DIAGONAL NOT INCLUDED)
NODE = "Sex"                                                           0
BLOCKING                                                               0
   1  0                                                                0
   0  1                                                                0

 -2 Log PseudoLikelihood =            791.568                          1
        Goodness of Fit =             732.798                          2
      Model Chi-squared =             334.103      df = 4              2

FIT AT P = 0.5                        RESIDUALS
         PRED           342                                            3
          <P   |   >P                 Absolute =          262.574      4
         ------------------           Squared  =          132.991      4
      0 |     356|      97  28.4%                                      3
OBS     |--------|--------  -----                                      3
      1 |     114|     245  71.6%                                      3
        |        |                                                     3
359     |   31.8%|   68.2%

PARM BLOCK      b "Std.Err"   PLWald   p(df=1)    exp(b) Counts    Errors 5
 1       -3.3243  0.2422  188.4052   < 0.01       0.04    359   0.00000
 2        1.0655  0.1784   35.6699   < 0.01       2.90    240   0.00000
 6        0.1236  0.0111  124.7508   < 0.01       1.13   8289   0.00001
 1    1   0.7603  0.1778   18.2808   < 0.01       2.14    239   0.00000

CORRELATION MATRIX:                                                    6
  PARM                1          2          6          1
       BLOCK                                           1
   1           1.00000   -0.21823   -0.79371   -0.35061
   2          -0.21823    1.00000   -0.09712   -0.13897
   6          -0.79371   -0.09712    1.00000    0.00967
   1     1    -0.35061   -0.13897    0.00967    1.00000
```

6.6.2 Global and Local counts

To construct the p* parameter estimates in Table 6.3, Choice counts were made both globally (for the entire network) and locally (when both ends of the link are in block 1, that is sender and receiver are of the same sex). This results in double counting, so that the first number (359) in the Counts column of section **5** is the sum of all links (including those in block 1), while the last number in the Counts column (259) is the number of links in block 1 only. From this we can deduce that there are 100 links from one sex to the other (outside of block 1). The double counting has effects on the p* fit: The p* parameter listed for Choice in block 1 has the global value subtracted from it. That is, for block 1 the Choice estimate is -3.3243+0.7603 = -2.5641. This effect holds in general for any parameter that is estimated both globally and locally (but not if strictly global or local). It is simple to make the correction for any number of different blocks. However, this effect can also result in lower "Standard Error", PLWald and significance measures. Also, if a parameter is counted and estimated globally, the resulting p* fit parameter will be correlated with all the other p* parameters (table 6.3 part **6**). None of these effects have been described or commented on in the p* literature.

To remove all these side-effects, MultiNet makes it simple to remove any global counts and fits by using the '**Modify→Add 1**' menu choice of the '**Examine p* fit**' window. This adds 1 to all block numbers, transforming all 0-blocks to 1-blocks and so automatically deselecting all the global parameters. The resulting fit (Table 6.4) has exactly the same -2*LogPL (and related measures), while the fit parameters need no further corrections. In addition, the correlations are non-zero only within blocks, and the PLWald and significance measures are more useful.

Table 6.4. Simple block model with only local counts (within and between sexes) and fits for Choice. Compare with Table 6.3. Global parameters (2 and 6) are unchanged.

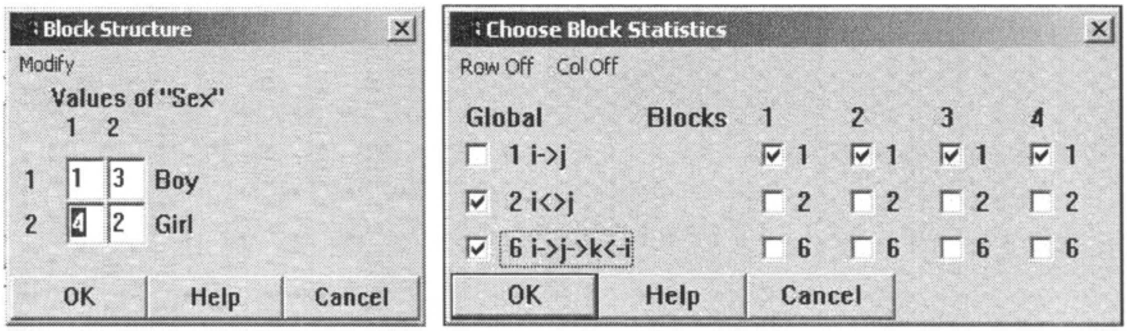
```
LINK: "Get on with"  LINKS =   359  NODES =   29  (DIAGONAL NOT INCLUDED)
NODE = "Sex"
BLOCKING
   2   1
   1   2

 -2 Log PseudoLikelihood =               791.568
         Goodness of Fit =               732.798
        Model Chi-squared =               334.103     df = 4

 FIT AT P = 0.5                           RESIDUALS
               PRED          342
            <P   |   >P                   Absolute =          262.574
        ------------------                Squared  =          132.991
       0 |     356|      97  28.4%
 OBS     |--------|--------  -----
       1 |     114|     245  71.6%
         |        |
 359     |  31.8%|   68.2%

PARM BLOCK     b "Std.Err"    PLWald   p(df=1)    exp(b) Counts      Errors
  2         1.0655  0.1784    35.6699  < 0.01      2.90     240     0.00000
  6         0.1236  0.0111   124.7508  < 0.01      1.13    8289     0.00001
  1    1   -3.3243  0.2422   188.4052  < 0.01      0.04     120     0.00000
  1    2   -2.5641  0.2451   109.4336  < 0.01      0.08     239     0.00000
```

6.6.3 Complex Blocks

While the simple block method may be sufficient for fitting to global and "within block" effects, some models may also require "between blocks" effects as well. An example is given in (WP p. 419, model 30) where a fit is made to Mutuality, Transitivity, and all four possible within and between blocks Choice parameters (in this case involving the two sexes). MultiNet allows this type of modelling by letting the user define the blocks as follows:

```
1 3
4 2
```

where the numbers are entered by point-and-click, tab, or shift-tab to select an attribute combination, then typing in the Block number. Here there are 4 distinct blocks, and recall that rows represent links sent while columns represent links received. So Block 1 represents boy "getting on with" boy, Block 3 boy "getting on with" girl, Block 4 girl "getting on with" boy, and Block 2 girl "getting on with" girl.



Values of "Sex"

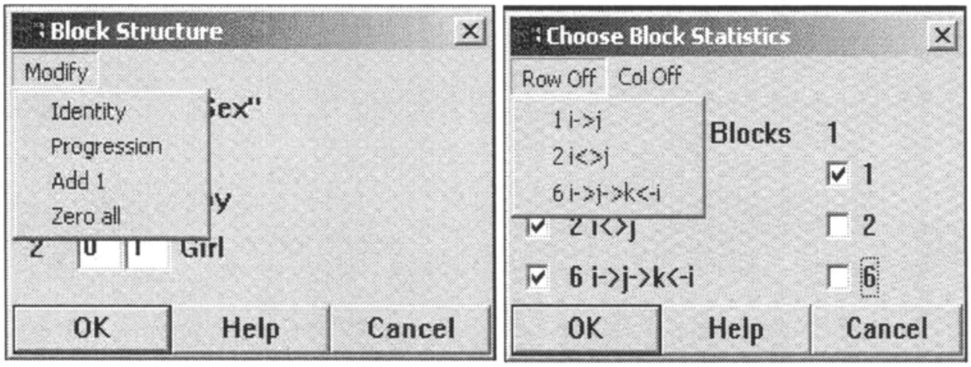
	1	2	
1	1	3	Boy
2	4	2	Girl

Global	Blocks 1	2	3	4
□ 1 i->j	☑ 1	☑ 1	☑ 1	☑ 1
☑ 2 i<>j	□ 2	□ 2	□ 2	□ 2
☑ 6 i->j->k<-i	□ 6	□ 6	□ 6	□ 6

a) Block definition **b)** Parameter selection

Figure 6.7. Complex block modelling

WP model 30 fits Choice to each block, and Mutuality and Transitivity globally, so after setting up the Block structure, parameter choices are set as shown in Figure 6.7. Notice that Choice ('1 i→j') is not selected globally: selecting it globally would surely lead to an error (discussed later). When there are no global (zero) blocks, MultiNet deselects all the global parameter choices, meaning these must be turned off for parameters 2 and 6 in Blocks 1 to 4 , and then turned them on as "Global" parameters, with the result shown in Table 6.5 (omitting the 15 probability level tables).



a) Modify choices **b)** Multiple parameter choices

Figure 6.8. Block structure control choice windows.

Table 6.5. Report for complex block model. -2*LogPL drops by 32.58 with 2 more parameters.

```
MultiNet PSTAR REPORT ON "VICKCHAN.NOD"  5/05/2004  15:53:32
RUN #3 ITERATIONS =  6

LINK: "Get on with"  LINKS =   359  NODES =   29  (DIAGONAL NOT INCLUDED)
NODE = "Sex"
BLOCKING
   1  3
   4  2

 -2 Log PseudoLikelihood =              752.992
          Goodness of Fit =              776.118
        Model Chi-squared =              372.679     df = 6

 FIT AT P = 0.5                          RESIDUALS
               PRED          355
             <P   |   >P                 Absolute =          246.987
        ------------------               Squared  =          124.022
      0 |      359|      94  26.5%
 OBS    |---------|--------  -----
      1 |       98|     261  73.5%
        |         |
 359    |   27.3%|    72.7%
 PARM BLOCK   b     "Std.Err"   PLWald   p(df=1)     exp(b) Counts    Errors
  2        1.3265   0.1960   45.7888   < 0.01       3.77    240    0.00000
  6        0.1319   0.0125  111.7546   < 0.01       1.14   8289    0.00000
  1    1  -2.2206   0.2737   65.8364   < 0.01       0.11     77    0.00000
  1    2  -3.1949   0.3139  103.6289   < 0.01       0.04    162    0.00000
  1    3  -2.9501   0.2834  108.3339   < 0.01       0.05     82    0.00000
  1    4  -4.3489   0.3314  172.2432   < 0.01       0.01     38    0.00000

 CORRELATION MATRIX:
  PARM              2          6          1          1          1          1
      BLOCK                               1          2          3          4
    2           1.00000   -0.09637   -0.31890   -0.26345   -0.05283   -0.36292
    6          -0.09637    1.00000   -0.53802   -0.80906   -0.77490   -0.64335
    1    1     -0.31890   -0.53802    1.00000    0.56305    0.46463    0.50514
    1    2     -0.26345   -0.80906    0.56305    1.00000    0.67089    0.66694
    1    3     -0.05283   -0.77490    0.46463    0.67089    1.00000    0.55322
    1    4     -0.36292   -0.64335    0.50514    0.66694    0.55322    1.00000
```

To make the selections and choices simpler for large numbers of blocks and parameters, each of the block structure control windows has additional menu items as shown in Figure 6.8. For the '**Block Structure**' window, the **Modify** choices are:

- **Identity**: restore default simple cohesive block structure (1's down the diagonal)
- **Progression**: diagonal contains progression form 1 to number of categories C (1,2,...,C)
- **Add 1**: Add 1 to every cell. This forces no Global parameters.
- **Zero All**: Start clean for special block structures.

For the '**Choose Block Statistics**' window the menu items allow the deselection of <u>all</u> parameters globally or in a block ('**Col Off**') or the deselection of any parameter in <u>all</u> blocks ('**Row Off**' shown in Figure 6.8b). These menu choices simplify the procedures described above, and are especially useful with many blocks or parameters.

The block-modelling scheme is very flexible, though it requires more inputs from the user. The number of possible blocks grows as the square of number of Node variable categories, and there are possibilities for the fit to fail (see below). Here are some examples of possible block structures with commentary below, using the two-category attribute gender:

a) $\begin{matrix}1 & 2\\ 2 & 1\end{matrix}$ b) $\begin{matrix}1 & 0\\ 0 & 2\end{matrix}$ c) $\begin{matrix}1 & 3\\ 3 & 2\end{matrix}$ d) $\begin{matrix}0 & 1\\ 1 & 0\end{matrix}$ e) $\begin{matrix}1 & 2\\ 1 & 2\end{matrix}$

- a) Fit to within- and between-categories (see WP Models 14-16)
- b) Fit within-categories separately and ignore between-category effects
- c) Fit within-categories separately and lump between-category effects together
- d) Ignore within-category; lump between-categories together
- e) Ignore sender; fit to receiver

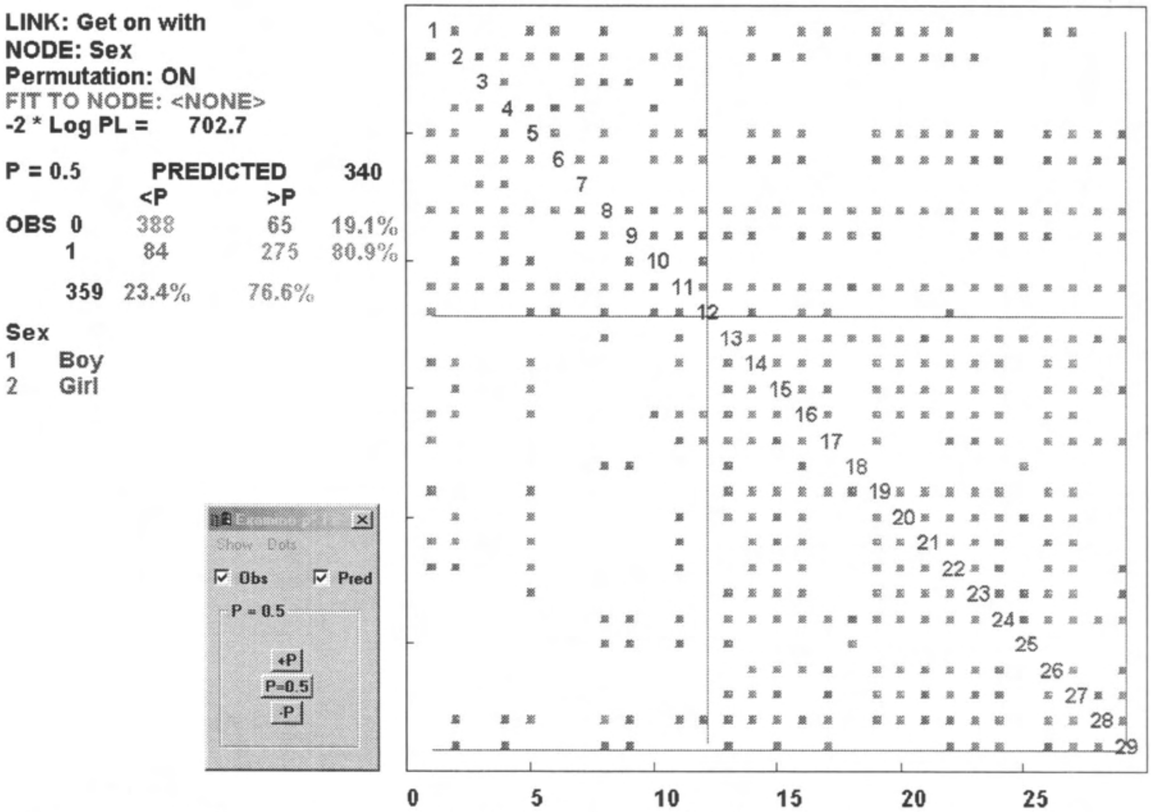
With this flexibility, the number of parameters can grow quickly, so some restrictions are imposed:

- 1) No more than 16 different blocks (4 or fewer categories can be fit to a "saturated" model, where each block has a different number)
- 2) Maximum of 76 parameters. This allows fitting all 15 triads globally and in 4 blocks (or 4 triads in 16 blocks). MultiNet will warn if too many parameters are chosen.

6.7 Evaluating p* fits in MultiNet

The 1-D visualisations available in this module can help in guiding the p* curve-fitting procedure. For example, Figure 6.3 shows that with only Choice and Mutuality as parameters, a number of nodes with high out-degree and low in-degree are not fitted well since Mutuality tends to symmetrize links. The problem is still evident in Figure 6.6, which includes both Transitivity and a block model for Choice based on node variable "Sex". These results suggest that using one of or more of the parameters sensitive to in-, out- or mixed degree might lead to better fits. Using only parameters 1-5 without a block model results in -2*LogPL = 786.6, which is better than the result for RUN #2. Adding the sex-based block-model results in -2*LogPL = 749.5, which is better than RUN #3, and which visually does a better job at fitting the node degrees. Adding Transitivity produces a still better fit, but now some parameters are not significant and in fact they are the block-related parameters. Removing all block structure leads to a fit with -2*LogPL = 702.7 with only 4 parameters and no block structure required. Figure 6.9 shows the resulting fit, and also demonstrates that a fit may be permuted by a node variable, even if the variable was not actually used in the fit. The message '**FIT TO NODE: <NONE>**' is added to the display in red to show that the fit did not use the currently displayed node variable "Sex".

In this way the rapid interaction between visual display and textual reports of parameters and their significance can help produce good p* fits with a moderate number of parameters. However, as with any "hill-climbing" algorithm, only locally optimal solutions are found, and there is no recipe for finding the best possible p* fit. Also, since the p* parameters are not in general independent (parameter correlations are not zero, except between blocks in a saturated model), there is no guarantee that changing some parameters will have no affect on those that are unchanged. For example, Mutuality and Transitivity for Figure 6.9 are higher than for Tables 3 and 4, with Transitivity almost doubled.



```
LINK: Get on with
NODE: Sex
Permutation: ON
FIT TO NODE: <NONE>
-2 * Log PL =     702.7

P = 0.5       PREDICTED        340
               <P          >P
OBS  0        388          65    19.1%
     1         84         275    80.9%

     359     23.4%       76.6%

Sex
1    Boy
2    Girl
```

Figure 6.9. A good 4 parameter p* fit. The display is permuted by Node attribute Sex, although this was not used in the fit. The red '**FIT TO NODE:<NONE>**' message indicates that the fit used a different node variable (in this case: NONE).

Table 6.6. Fit results for 4 parameter fit shown in figure 6.9

PARM	BLOCK	b	"Std.Err"	PLWald	p(df=1)	exp(b)	Counts	Errors
2		1.7563	0.2123	68.4456	< 0.01	5.79	240	0.00000
4		-0.1138	0.0260	19.1125	< 0.01	0.89	4620	0.00000
5		-0.1565	0.0136	132.9890	< 0.01	0.86	8822	0.00000
6		0.2275	0.0185	151.9222	< 0.01	1.26	8289	0.00000

Close examination of Figure 6.9 shows that some reciprocal links are predicted with different probabilities in each direction (e.g., for nodes 9 and 18), but not what the probabilities are. These can be found by stepping through the probability levels using the '**Examine p* fit**' window, or they can all be summarized by creating a new Link variable using **Define**. The resulting Link variable has categorical values (and value labels) that show the probability level below which each link is predicted. This variable may then be compared with other fits in the Variables module or displayed in the **Eigenspaces** module to get a complete view of how well the p* fit worked, and what network features were best captured by the fit. An example of this display is Figure 6.19 in the Eigenspaces section. **Define** can only produce a Link variable with values for correct (green) and false negative (blue) links. Producing links for the false positives (red) could destroy sparsity.

6.8 Error Traps

Again, we assume that the reader is familiar with p* fitting and concentrate on matters specific to the implementation in MultiNet. This section describes what to look for when the program fails — or almost fails — to converge. Generally, the program appears quite robust, but certain situations can cause problems. MultiNet traps errors and puts everything it "knows" up to that point into the Report, along with some helpful advice. Three types of error are detected. Each will produce an Error window describing the error condition, which also appears in the Report.

6.8.1 STATISTICS LINEARLY DEPENDENT!

At some point within the logistic regression procedure the code must invert a matrix. Initially this matrix is the inner product of the matrix of change statistics with its transpose. The **'Statistics Linearly Dependent'** message results when the matrix inversion fails the first time. The most common cause of this failure is the attempt to fit to a parameter with triad statistics of 0. This may occur either globally (in which case there is no point trying to fit to that statistic) or only within a Block (in which case don't use that parameter in that Block). Another common source of this error is the use of a parameter both globally and with a full set of blocks. For example, in WP Model 30 (see above), this error would appear if we tried to fit Choice (1) as a global parameter and to all four blocks. The matrix would then have a row corresponding to global Choice which is a linear combination of the four rows corresponding to Choice within each of the four blocks. To avoid this error, do not fit to the same parameter both globally and to all of a saturated (no zero blocks) block-model.

Less often, the error can occur for more subtle reasons, having to do with the structure of the network. If this happens, there is very little information available from the program because, at this point, it has just begun the analysis. The best advice is to try again with fewer parameters. One thing to look for in the triad counts is repeated values for related triads. For example, Choice and Mutuality will be equal (and dependent) in a network (or block!) which is symmetric. It doesn't hurt, and doesn't take much time (unless you have a very big network) to take a quick look at what possible problems will occur by initially attempting a fit to all the triads without any blocking.

6.8.2 LOGISTIC REGRESSION FAILED!

As the logistic regression iterations proceed, the initial matrix is updated by weights derived from the latest fit. SPAR monitors the change in parameters with each iteration, and if a parameter changes too rapidly this will be reported as '**Logistic Regression failure**', since it is very likely that at the next iteration, the matrix inversion will fail. The most common source of this error is a high correlation between (among) a pair of (or more) parameters. There is a lot more information available in the report since the latest estimates of parameter values, errors and correlations are all available, but this condition does not appear very common. If this occurs, the best advice is to look for parameters with both high estimated values and high correlations, and eliminate one or more of them.

6.8.3 TOO MANY ITERATIONS!

The SPAR code appears to be quite robust. Usually no more than 6 iterations are required. There is an upper limit of 10 iterations, after which the program will stop iterating and report the current fit parameters. There are two situations commonly responsible for this problem:

- If one of the parameters chosen has zero counts (as reported in the Counts column in both screen and file output), try again without this parameter. It is also useful to look for parameters with very few counts, and low pseudo-Wald statistic, especially within blocks. These may be removed from the model to give comparable fits with fewer parameters.
- If two (or more) parameters are highly correlated, the iterations will drive both (or more) to extreme values (generally greater than 10 in absolute value). Looking for a pair of extreme values (in the b column) can be helpful. If such a pair exists, try again without one of them. It may be necessary to examine the correlation matrix to understand why a fit does not converge. Look for

pairs (or more) of parameters with correlations very close to 1 in absolute value. The "Error" column in the Report is also a useful indicator of which parameter(s) are causing difficulties.

It is always a good idea to examine the correlation matrix in any case. Parameters with high correlations (> .9) indicate that the model can be fit with fewer parameters. This often occurs when a triad count is fit both globally and within blocks where most of the counts occur. To avoid this, use a "saturated" model, such as a) or c) above, with *no* global counts.

6.9 Technical appendix

6.9.1 Sparse matrix estimation of p* parameters

The use of logistic regression as a means of estimating p* parameters was first suggested by (Frank & Strauss, 1986). This suggestion was examined in detail in (Strauss & Ikeda, 1990). They showed that the maximum likelihood parameter values could be estimated by maximum pseudo-likelihood as originally defined by Besag (1974), using standard logistic regression procedures. They demonstrated that, for graphs that are small and simple enough, the approximation is fairly good. This was a very useful result, since calculating the actual maximum likelihood values is a very hard computational problem that has still not been solved (Handcock, 2003). In the paper that introduced the family of p* models (Wasserman & Pattison, 1996), it was assumed that using logistic regression to find maximum pseudo-likelihood estimates of p* parameters would be a sufficiently useful approximation until better methods were developed, and almost all of the applied p* literature to date uses this method of estimation (usually with a short statement that this is – temporarily – an approximation).

The use of "packaged" logistic regression routines (in SAS SPSS, BMDP, etc.) requires that the p* change statistics be pre-calculated for each statistic. Unfortunately, this destroys sparsity, since the change statistics occur not only for each exiting link , but also for each potential link. This means that for N nodes, each statistic consists of N^2 values (subtract N if there are no self-links, divide by 2 if the network is symmetric). For a network of 1000 nodes, this means each change statistic requires one million values. Passing these values to a "packaged" routine generally requires some ASCII formatting which requires one byte for each order of magnitude present in the statistic. Whereas Choice and Mutuality values are always single bytes (Choice is constant 1, Mutuality is the transpose of the binary-valued network), the various 2-stars and triangles are integer-valued counts that can be two or three bytes (or more) for each value. This leads to an estimate of 30 Megabytes for a 15-parameter fit when N=1000. The results of this fit are then usually boiled down into a set of tables that take up much less than 1 megabyte. It is the "data explosion" of the change statistics, which are after all only temporary values, that makes the application of a packaged logistic regression routines problematic for large networks. Not only are these intermediate results large, they are also slow to calculate, and large and slow for the packaged routines to use. All this makes any "experimental" p* curve-fitting slow and cumbersome: the opposite of what is required for an "exploratory" analysis.

The SPAR code was intended to avoid these problems by skipping over the intermediate stage entirely. Instead, the change statistics would be calculated "on-the-fly" inside the routine that calculated the logistic regression. The pseudo-code in Figure 6.10 describes one step in the iterative calculation of a logistic regression. The notation is as follows:

- input **y** is the network, represented as a binary vector of length N^2.
- input **X** is the matrix of change statistics with *g* columns, each column of length N^2.
- output **b** is the logistic regression fit of length *g*, one for each of *g* parameters.
- Lower case letters are vectors. Upper case letters are matrices.
- `INV(W)` represents Matrix inverse of **W**.
- `DIAG(v)` is a diagonal matrix with vector **v** along the diagonal and 0's off-diagonal.
- `SQ(X)` is square matrix **X'X**, i.e., matrix product of transpose of **X** and **X**.
- `Exp(x)` is element-by-element exponential of vector **x**.
- `Sqrt(x)` is element-by-element square root of vector **x**.

This looks complicated, but it is just a weighted linear least-squares fit. Other complications include finding good starting guesses for the b values and accumulating useful statistics such as the correlations, but these are not difficult. Also, the difference vector **d** is checked for large changes, which is a sign of potential regression failure due to highly correlated parameters. The key point is that the calculations involve matrix-matrix and matrix-vector products, which are all linear. This means that the results can be accumulated from the statistics of each node, one at a time. These results enter the logistic regression loop at the code points marked with (*), that is, any time the "input" vector of the network y or the matrix of statistics X is required. As a result, the space which would be required to store X is not needed. The network is stored sparsely as ID pairs, and only the part of vector y directly connected to the current node is needed. All other matrices require storage of only g^2, where *g* is the number of parameters. This solves the problem of allocating huge amounts of space for the intermediate change statistics, but it means that the statistics must be re-calculated at each iteration. For "reasonably well-behaved" statistics (correlations are less than 0.95) the least-squares code is equivalent to a Newton-Raphson least-squares minimizer, which converges quadratically fast (the number of significant digits doubles with each iteration). This means that usually only a few (3-6) iterations are needed. In fact, no more than 10 iterations are allowed since this many iterations is a sign that there is a problem with convergence due to highly correlated p* parameters.

```
b←initial guess                g random numbers between 0.01 and 0.99
oldb←b                         Keep a copy for next iteration
loop:                          Beginning of logistic regression loop
  s←Xb                         Vector s from matrix product of X and b (*)
  p←1÷(1+Exp(-s))              Logistic vector p from s
  w←p×1-p                      Weight vector w from logistic p
  s←s+(y-p)÷w                  Update vector s by weighted difference of data and logistic
  T←DIAG(Sqrt(w))              Matrix T is diagonal of square roots of weights
  V←INV(SQ(XT))                Matrix V is inverse of square of weighted input X (*)
  W←DIAG(w)                    Matrix W is diagonal of weights
  U←XW                         Matrix U is product of  input and weights (*)
  z←sU                         Temporary vector to form b=VsU
  b←Vz                         Logistic regression estimate b
  d←oldb-b                     Change from last estimate
  oldb←b                       Keep a copy for next time
IF d sufficiently small, then convergence, ELSE GOTO loop
```

Figure 6.10. Pseudo-code for logistic regression iteration.

```
c 2 outstars. What is out-degree of ifr?
c subtract 1 if ifr->ito.
        do 32 i=1,nn
          istats(i,3) = idegfr(ifr) - idat(I)
32      continue
c
c 2 instars. What is in-degree of ito?
c subtract 1 if ifr->ito.
        do 42 i=1,nn
          istats(i,4) = idegto(i) - idat(i)
42      continue
c
c 2 mixed stars. How many depend on ifr->ito?
c subtract 2 if ito->ifr (transpose - already calculated if needed)
        do 52 i=1,nn
          istats(i,5) = idegfr(i) + idegto(ifr) - 2 • istats(i,2)
52      continue
```

Figure 6.11. Code for calculating 2-stars. Idegfr and idegto are vectors of length nn=N which hold out- and in-degrees. They are calculated once when data is read in. Idat(N) and istats(N,2) are the network and its transpose as seen from node ifr, which are also used in a number of other calculations.

```
c for cyclic triads, count the number of ways each node can
c reach ifr in 2 steps by ifr<-j<-i
      l2 = idegto(ifr)
      if (l2.eq.0) goto 80
        l1 = lp2(ifr)
        l2 = l1 + l2 - 1
c look at who points to ifr and count who else points to them
        do 72 jj=l1,l2
          j = lid1(jj)
          l4 = idegto(j)
          if (l4.eq.0) goto 72
          l3 = lp2(j)
          l4 = l3 + l4 - 1
          do 74 ii=l3,l4
            i = lid1(ii)
c don't count if it points to ifr
            if (ifr.eq.i) goto 74
            istats(i,7) = istats(i,7) + 1
74        continue
72      continue
80     continue
```

Figure 6.12. Calculating cyclic triads. The same code is shared in the calculation of some of the reflexive triads (parameters 8, 9 , 10 and 13), though these extra steps have been removed for clarity. Most of this code uses pre-calculated vectors such as idegto and sparse matrix pointer variables lid1 and lp2.

Since the number of iterations can be kept small, it remains only to show that re-calculating the change statistics for each node inside the regression loop is reasonably efficient in time. In fact, some change statistics require no or very little calculation. Choice is constant. Mutuality requires the transpose of the network, which is also useful for many other statistics. Some other statistics are very easy to calculate, for example all 2-stars (Figure 6.11). The block-related local statistics use already-calculated global counts. The most complicated counts involve the triangles (transitive and cyclic). The cyclic counts code is shown in Figure 6.12. It requires following all links up to two steps away, and so the efficiency depends on the network having low density; this is generally the case for social networks. This example also shows how the change statistics calculations can make ready use of the pointer variables that are an essential part of sparse matrix representation.

6.9.2 Details of fit

Appended to every report is a 5 by 3 table of tables of observed and predicted links, referred to as section 7 in the main text. This information is also available with the 1-D display by using the '**Explore p* Window**' buttons '**P+**' and '**P-**' to step through 15 probability levels from 0.0625 to

```
                 P = 0.0625                                 P = 0.125                                  P = 0.1875
                    PRED            717                        PRED            624                        PRED            548
                 <P   |   >P                                <P   |   >P                                <P   |   >P
             -------------------                        -------------------                        -------------------
        0  |      91|     362  50.5%               0  |     174|     279  44.7%               0  |     244|     209  38.1%
OBS        |--------|--------  -----      OBS         |--------|--------  -----      OBS         |--------|--------  -----
        1  |       4|     355  49.5%               1  |      14|     345  55.3%               1  |      20|     339  61.9%
           |        |                                 |        |                                 |        |
359        |    1.1%|   98.9%             359         |    3.9%|   96.1%             359         |    5.6%|   94.4%

                 P = 0.25                                   P = 0.3125                                 P = 0.375
                    PRED            518                        PRED            485                        PRED            439
                 <P   |   >P                                <P   |   >P                                <P   |   >P
             -------------------                        -------------------                        -------------------
        0  |     267|     186  35.9%               0  |     288|     165  34.0%               0  |     313|     140  31.9%
OBS        |--------|--------  -----      OBS         |--------|--------  -----      OBS         |--------|--------  -----
        1  |      27|     332  64.1%               1  |      39|     320  66.0%               1  |      60|     299  68.1%
           |        |                                 |        |                                 |        |
359        |    7.5%|   92.5%             359         |   10.9%|   89.1%             359         |   16.7%|   83.3%

                 P = 0.4375                                 P = 0.5                                    P = 0.5625
                    PRED            392                        PRED            355                        PRED            311
                 <P   |   >P                                <P   |   >P                                <P   |   >P
             -------------------                        -------------------                        -------------------
        0  |     338|     115  29.3%               0  |     359|      94  26.5%               0  |     385|      68  21.9%
OBS        |--------|--------  -----      OBS         |--------|--------  -----      OBS         |--------|--------  -----
        1  |      82|     277  70.7%               1  |      98|     261  73.5%               1  |     116|     243  78.1%
           |        |                                 |        |                                 |        |
359        |   22.8%|   77.2%             359         |   27.3%|   72.7%             359         |   32.3%|   67.7%

                 P = 0.625                                  P = 0.6875                                 P = 0.75
                    PRED            261                        PRED            218                        PRED            185
                 <P   |   >P                                <P   |   >P                                <P   |   >P
             -------------------                        -------------------                        -------------------
        0  |     401|      52  19.9%               0  |     420|      33  15.1%               0  |     435|      18   9.7%
OBS        |--------|--------  -----      OBS         |--------|--------  -----      OBS         |--------|--------  -----
        1  |     150|     209  80.1%               1  |     174|     185  84.9%               1  |     192|     167  90.3%
           |        |                                 |        |                                 |        |
359        |   41.8%|   58.2%             359         |   48.5%|   51.5%             359         |   53.5%|   46.5%

                 P = 0.8125                                 P = 0.875                                  P = 0.9375
                    PRED            143                        PRED             94                        PRED             34
                 <P   |   >P                                <P   |   >P                                <P   |   >P
             -------------------                        -------------------                        -------------------
        0  |     440|      13   9.1%               0  |     447|       6   6.4%               0  |     452|       1   2.9%
OBS        |--------|--------  -----      OBS         |--------|--------  -----      OBS         |--------|--------  -----
        1  |     229|     130  90.9%               1  |     271|      88  93.6%               1  |     326|      33  97.1%
```

Figure 6.13. An example of the 'Details of Fit' section of every Pstar report, in this case for RUN #3.

0.9375, or by using **Define** to create a new link variable with these probabilities as the value of each link. Figure 6.7 shows the summary information for RUN #3 which is a more complex model, and so shows more complex behaviour at the different probability levels.

6.10 References

Albert, R. and Barabasi, A.-L. (2002) Statistical mechanics of complex networks, *Review of Modern Physics* **74**:47-97

Crouch, B. and Wasserman, S. (1998). A Practical Guide to Fitting p* Social Network Models, *Connections* **31**: 87-101.

Frank, O. and Strauss, D. (1986). Markov Graphs, *J. Am. Stat. Assoc.* **81**: 832-842.

Handcock, M. (2003) Statistical Models for Social Networks: Inference and Degeneracy. In National Research Council, *Dynamic Social Network Modeling and Analysis: Workshop Summary and Papers*, (229-240). Eds. Ronald Breiger, Kathleen Carley and Phillipa Pattison, Division of Behavioral and Social Sciences and Education. Washington DC. The National Academics Press

Holland, P.W. and Leinhardt, S. (1981) An exponential family of probability distribution for directed graphs, *J. Am. Stat. Assoc.*76:33-65

Newman, M., Strogatz, S. and Watts, D. (2001) Random graphs with arbitrary degree distributions and their applications, *Physical Reviews E* **64**:026118

Pattison, P. and Robins, G. (2000) Neighbourhood-based models for social networks, *preprint.* Presented at INSNA XX, Vancouver 2000.

Pattison, P., Robins, G., and Wasserman, S. (1999) Some computational notes on univariate and multivariate random graph models for social networks. *Preprint.*

Seary, A. (1999) PSPAR: Sparse Matrix Version of PSTAR. http://www.sfu.ca/~richards/pspar5k.zip

Strauss, D. and Ikeda, M. (1990). Pseudolikelihood Estimation for Social Networks, *J. Am. Stat. Assoc.* **85**: 205-212.

Wang,Y.J. and Wong, G.Y. (1987) Stochastic Blockmodels for directed graphs, *J. Am. Stat. Assoc.*, **82**:9-19

Wasserman, S. and Pattison, P. (1996). Logit Models and Logistic Regressions for Social Networks I: An Introduction to Markov Graphs and p*, *Psychometrika* **61**: 401-425.

7. The Preferences module

7.1 Introduction

The Preferences module is used to set or restore default behaviour in the other modules. Preferences does not perform any analyses, or produce reports or graphical displays. Nevertheless, the settings made in Preferences can have an important effect on displays, reports and importing data. Because it can affect the actions of other modules, a few of the most important menu items available in this module are also made available in other modules as a Preferences menu item.

When MultiNet first starts up, it looks for a file called "MNW.INI" which should be in the same directory as the MultiNet executables (.W3, .BAT and .DLL files). This file contains definitions for the settings that may be changed in Preferences. Unlike most .INI files, it is not a text file, and should only be changed within MultiNet. If this file is not found, MultiNet uses internal settings which are also used to restore all MultiNet defaults. In this case, the final "nag" screen will also show up when MultiNet exits, since the registration code is also saved in MNW.INI (one reason why it is not human-readable).

7.2 Preferences menu bar

The following is a list of Preferences menu items, their actions, and their effects on other MultiNet modules (Figure 7.1). All settings shown are the MultiNet defaults.

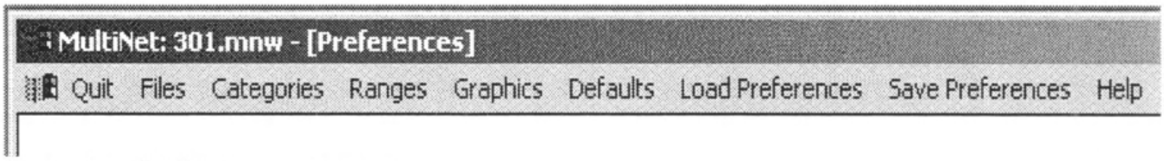


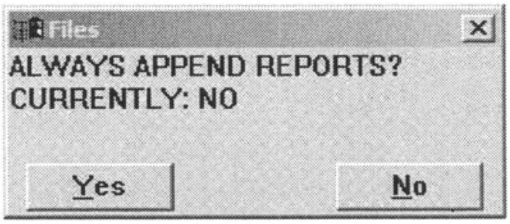
Figure 7.1. Preferences Menu bar

7.2.1 Quit

Exit from the Preferences Module back to the Main MultiNet menu.

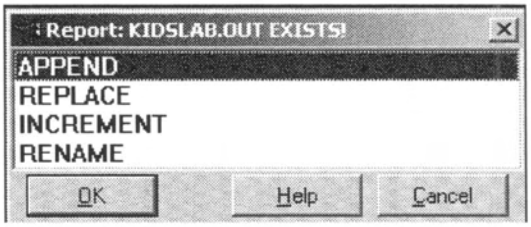
7.2.2 Files

Clicking this item produces a menu with one choice: **Always Append**. Selecting this opens a YesNo window (Figure 7.2) showing the current setting. When MultiNet files a report or PostScript graphic, it will always ask what to do if the output file already exists (Figure 7.3). The easiest choice is 'Append!' add to the end of the existing file. It is convenient to select this as the automatic behaviour, so that this selection does not have to be made every time there is some output. MultiNet uses the Form-Feed (Page Break) character to separate reports, and the View windows in MultiNet (including **File→View** from the Main Menu) will use Multi-View windows with **Next** and **Last** buttons for convenient scrolling through such text files. PostScript files containing multiple images also behave well. However, automatic append is not possible for Bitmap file.


Files

ALWAYS APPEND REPORTS?
CURRENTLY: NO

Figure 7.2. YesNo window to select **Always Append** as default behaviour.


Report: KIDSLAB.OUT EXISTS!

APPEND
REPLACE
INCREMENT
RENAME

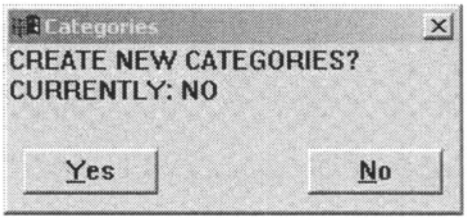
Figure 7.3. Always Append avoids standard selection window when saving to a file that exists.

7.2.3 Categories

Selecting this menu item provides the following automatic actions for managing missing category labels, empty categories in an analysis and missing data during Import and Export:

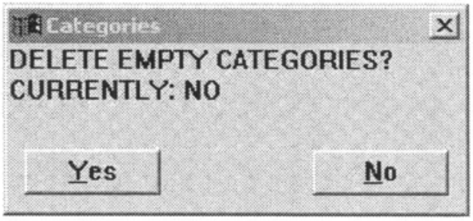
- **Create New** opens a YesNo window (Figure 7.4) showing the current setting. Discrete (categorical) data has the option of a list of descriptive labels (*value labels*) for the unique values in each category. These may be declared as part of the ASCII (Import) file. MultiNet also automatically generates labels whenever a variable is created that could be considered categorical, and these may be edited to be more informative. If a categorical variable does not have declared value labels, MultiNet generates labels based on the actual values when these are required in reports or displays. When a discrete variable with declared value labels is being Imported, MultiNet checks that every unique data value corresponds to a value label. If values without labels are encountered, and the "Yes" choice has been made, the program will generate a new label based on the undeclared value. However, it is likely that this situation results from an error in either value or declared labels. Selecting the default "No" response to this choice rejects the values that do not have declared labels, and marks these data items as missing data.

- **Delete Empty** opens a YesNo window (Figure 7.5) showing with the current setting. In the **Analyse** module, the intersection of three or even two variables can result in one or more categories having no counts for one or more of the variables. These empty counts are never used in the statistical calculations, but they may be included in reports and displays. For Stacked analyses, planes that have no counts are shown in reports and displays with "No data on this plane". For ANOVA displays, empty categories are displayed as flat ribbons which have constant counts of zero. In Panigrams, an empty column is given enough room for two letters of value label at the top and 0 percent at the bottom, but otherwise is blank. The empty categories can be removed from displays and reports by selecting the "Yes" choice to this setting (Figure 7.11 and Table 7.2). However, the mapping between colours and categories will not be guaranteed across modules, since the nth category is not necessarily the nth non-empty category. Selecting the default "No" always includes empty categories and guarantees the colour mapping across modules.



CREATE NEW CATEGORIES?
CURRENTLY: NO

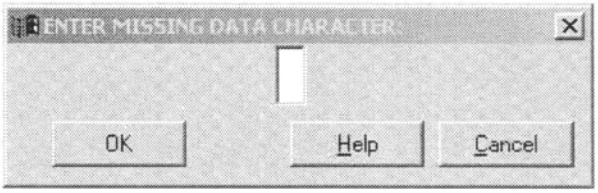
Figure 7.4. YesNo window for **Categories→Create New**



DELETE EMPTY CATEGORIES?
CURRENTLY: NO

Figure 7.5. YesNo window for **Categories→Delete Empty**

- **Missing Data** opens a single character Edit window (Figure 7.5) showing the current setting. By default, MultiNet uses the blank character (‘’) as a marker for missing data. Any other character may be chosen to represent missing data (for example, 0'may be treated as missing data as well). Whatever character is chosen, it will be used by both **File→Import** and **File→Export** to represent missing data. The request for a missing data character is repeated with **File→Import**, and choosing the character in Preferences makes this choice more convenient.

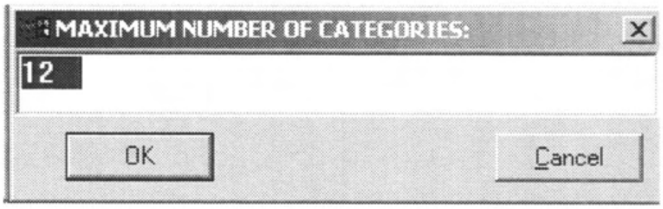


ENTER MISSING DATA CHARACTER:

Figure 7.6. One-character Edit window for **Categories→Missing Data** The current setting is ‘’ (blank).

7.2.4 Ranges

Selecting this menu item provides the following automatic actions for managing the number of categories, the number of bins and link strength range in network analysis.:

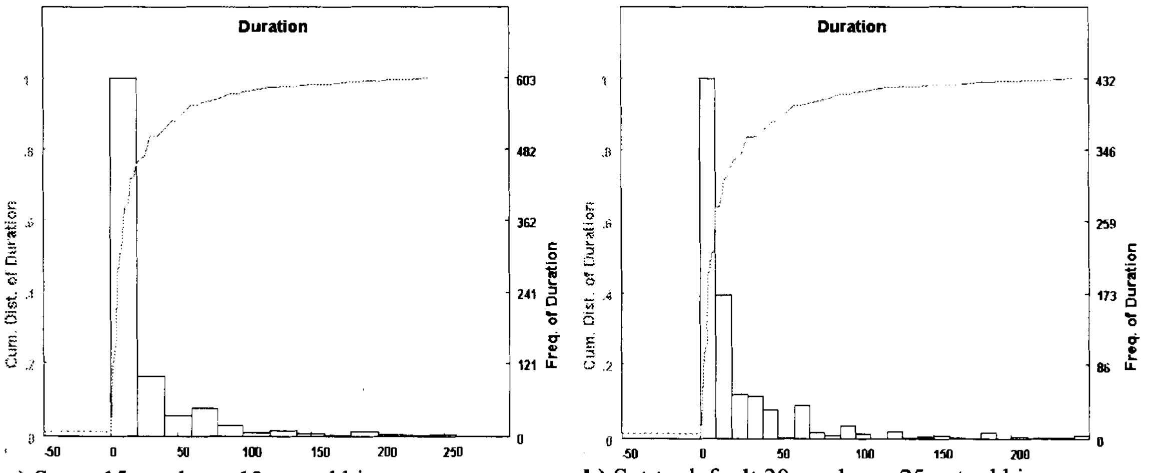


MAXIMUM NUMBER OF CATEGORIES:
12

Figure 7.7. Edit window for **Number of Categories**

Number of Categories opens an Edit window (Figure 7.7) showing the current setting for the maximum number of categories allowed in a categorical variable. The default is 12, which matches the number of distinct colours used in MultiNet displays. The maximum allowable number of categories is 20 (which can therefore create rather unwieldy 20 by 20 crosstab tables containing 1281 numbers). If more than 12 categories are selected, the 12 colours repeat. If less than 12 categories are allowed, only that number of colours is used. If a discrete variable is being Imported, it will be considered categorical only if the number of distinct values is not more than the current number of categories setting. In the **Analyse** module **XTABS**, **ANOVA** and stacked **CORREL** will not accept a variable with more distinct values than the current setting. In the **Variables** module, **Recode→Discrete** will not allow creation of more categories than the current setting. For these reasons, a **Preferences** menu item is available for the **Analyse** and **Variables** modules which allows changing this setting. The setting for **Number of Categories** is also checked whenever a numeric/continuous variable is expected in ANOVA or CORREL. If the variable does not have more distinct values than this setting, a warning message is given, but the analysis is allowed to proceed.

Number of Bins opens an Edit window showing the current setting for the maximum number of bins used in displaying and reporting numeric/continuous variables. Unlike **Number of Categories, Number of Bins** does not prevent analyses or variable recoding. The number is a goal for the display and reporting routines, and is only approximately attained. For numeric variables with the number of distinct items no more than this setting, both displays and reports are exact: every values is displayed and reported. If the number of distinct values is greater than this setting, the program attempts to find a set of bins that will give close to the requested result. Some results are shown in Figures 7.8 and 7.10 and Tables 7.1 and 7.2. The **Number of Bins** setting affects only the displays and reports, not the statistical calculations, which always use the exact values of variables.



a) Set to 15 produces 13 actual bins **b)** Set to default 30 produces 25 actual bins

Figure 7.8. Displays for Link Duration set at 15 and 30 bins.

Table 7.1. Bin values and statistics for requested bins 15 and 30 (Default) for Link variable Duration

a) Requested 15 bins, got 13

VALUES	COUNTS	%age
0	603	72.1%
20	100	12.0%
40	33	3.9%
60	46	5.5%
80	18	2.2%
100	6	0.7%
120	9	1.1%
140	5	0.6%
160	1	0.1%
180	7	0.8%
200	3	0.4%
220	2	0.2%
240	3	0.4%

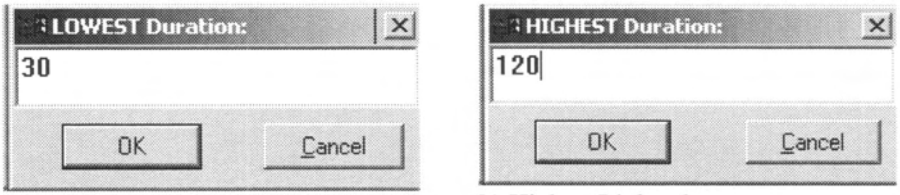
b) Requested 30 bins, got 25

VALUES	COUNTS	%age
0	432	51.7%
10	171	20.5%
20	51	6.1%
30	49	5.9%
40	32	3.8%
50	1	0.1%
60	39	4.7%
70	7	0.8%
80	3	0.4%
90	15	1.8%
100	6	0.7%
110	0	0.0%
120	8	1.0%
130	1	0.1%
140	2	0.2%
150	3	0.4%
160	1	0.1%
170	0	0.0%
180	7	0.8%
190	0	0.0%
200	2	0.2%
210	1	0.1%
220	1	0.1%
230	1	0.1%
240	3	0.4%

c) Statistics are the same for either bin selection

```
STATISTICS OF "Duration"
    MEAN  21.22
    VAR   1278
    SDEV  35.74
    MIN   0
    MAX   240
    MED   7
    SIZE  836
    BINS  49
    NODES 425
```

Link Range opens a YesNo window showing the current setting. In the **Analyse** module, any network analysis involves a single link variable or an equation using 0 or more link variables that are used to describe *link strengths*. For **XTABS**, separate tables summarize these strengths, and for any of the types of analysis, link strengths of 0 mean that those links are not counted. While it is possible in the **Variables** module to create new link variables with a range of values which is a subset of all the values, for analysis it is sometimes more convenient to do this "on-the-fly". For example, in an analysis of the 301 dataset, we may be interested in those



LOWEST Duration: 30

HIGHEST Duration: 120

a) Lowest Link value

b) Highest Link value

Figure 7.9. Edit windows for range selection of Link Duration

Table 7.2. ANOVA report for Dependent:Duration and Independent:Channel with No Empty Categories, Bins‡2 and 0Ðuration‡20. Compare with Table 2.9c in Section 2

```
STRENGTH=30<=Duration<=120
30<=Duration<=120, No Empty Categories; Bins=12
Independent: "Channel"                    Dependent: "Duration"

SUMMARY OF "Channel"
  LABEL   |  SIZE     |  MEAN    |CONF. INT.|
----------|----------|----------|----------|
face-face |   141     |   56.34 |     3.61 |
phone     |    19     |   53.47 |     9.82 |

ANOVA TABLE
-----------------------------------------------------------
SOURCE OF |  Sum of   | Deg. of  | Variance | Obtained |  P
VARIATION |Squares SS| Freedom  | Estimate |  ratio   |
-----------------------------------------------------------
BETWEEN   |   137.6  |        1 |   137.6  |      0.2 |> 0.50
GROUPS    |          |          |          |          |
-----------------------------------------------------------
WITHIN    |106308.4  |     158  |   672.84 |
GROUPS    |          |          |          |
-----------------------------------------
TOTAL     |106446.0  |     159  |

 STATISTICS OF "Duration"          DISTRIBUTION OF "Duration"
                                       VALUES     COUNTS      %age
   MEAN  56                              30          49      30.6%
   VAR   665.3                           40          32      20.0%
   SDEV  25.79                           50           1       0.6%
   MIN   30                              60          39      24.4%
   MAX   120                             70           7       4.4%
   MED   46.5                            80           3       1.9%
   SIZE  160                             90          15       9.4%
   BINS  19                             100           6       3.8%
                                        110           0       0.0%
                                        120           8       5.0%
```

contacts that take between 30 minutes and 2 hours rather than including all contacts. Selecting "Yes" for this choice results in two extra questions being asked for any network analysis (Figure 7.9). These questions show the current low and high values acceptable for link strength. Any links that fall outside this range are ignored in the analysis. The results of applying a link value range of 60Ðuration‡20 are shown in Figure 7.10 and Ta ble 7.2, which also show the effects of **No Empty Categories** and user-selected **Number of Bins**.

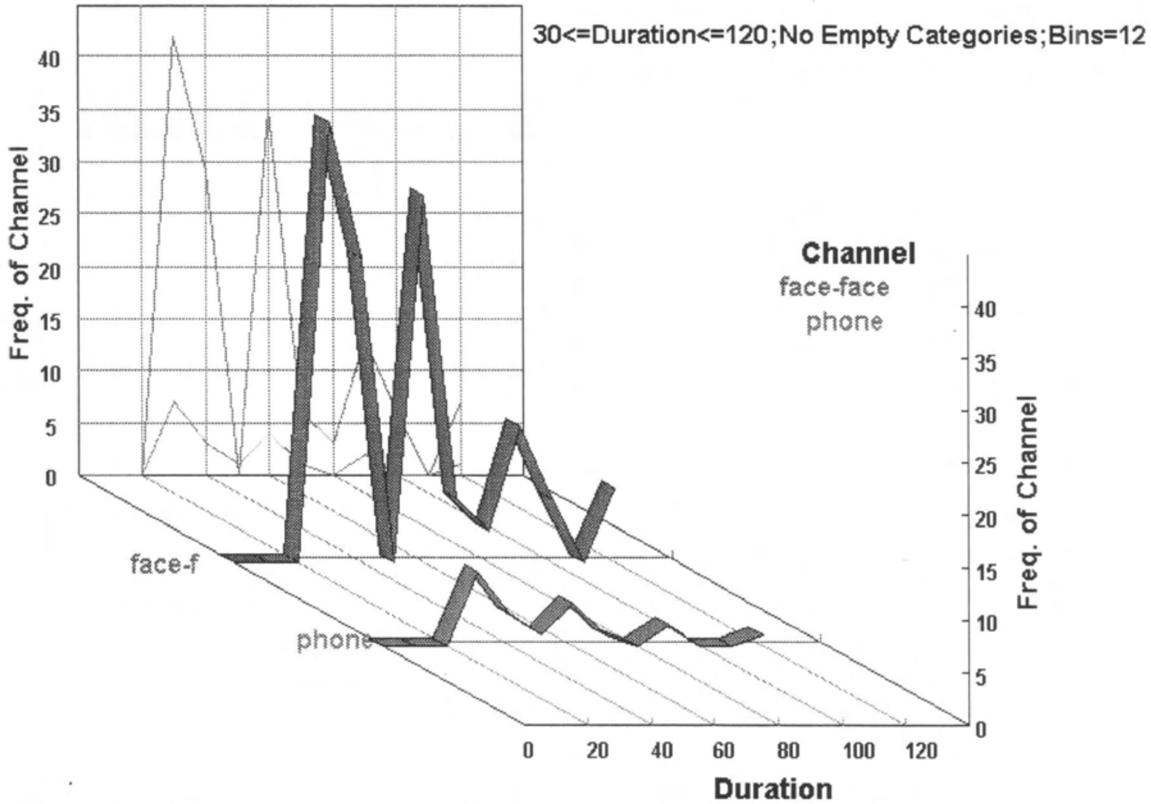


Figure 7.10. **ANOVA** display showing **a)** No Empty categories and **b)** User selected Bins‡2 and **c)** User selected Link range 30Ðuration‡20. Compare with Fig. 2.17b.

7.2.5 Graphics

Clicking this item produces a menu with two choices:

* **ANOVA Aspects**. Selecting this opens a YesNo window showing the current setting. Choosing "Yes" ensures that any changes to the ANOVA displays made using **Shift** will be saved along with other preference settings. There are two settings available, referred to as **Home1** (Figure 7.11) and **Home2** (which only appears in Stacked analyses; see Figure 2.14 in Analyse section). Each may be set separately (Figure 2.13 in Analyse Section). Choosing "No" ensures that **Home1** and **Home2** always have the default aspects for any **ANOVA** displays.

* **Panigram Text**: Each cell in a panigram contains a colour representing a category of the row variable, and text showing the column percent if the cell is large enough. By default, the text is black. However, printing the display (produced by PostScript, Bitmap or screen-capture) in grey-scale rather than colour may make these texts hard to read. Selecting this option opens a YesNo window which asks if you prefer white text. A Yes response will result in all text written to panigram cells to be white (as in the printed version of this documentation). This has no effect on any other graphic display.

7.2.6 Defaults

Clicking this item produces a menu with one choice: **Restore ALL Defaults**. Selecting this immediately restores all preferences to the MultiNet default values. These values in MNW.INI in the MultiNet directory are initially the same as the internally stored defaults. If the user is not satisfied with personal preference changes made to MNW.INI, the defaults may be restored by using this selection. These defaults are not stored in any file; they reside with the MultiNet program code. Selecting this choice has no effect on any preferences stored in MNW.INI files, only on the settings currently active in MultiNet.

7.2.7 Load Preferences

Clicking this item produces a menu with two choices:

- **From MultiNet Directory** is enabled if MNW.INI exists in the MultiNet startup directory. When this is selected, all preferences are set to the values in this file.
- **From Current Directory** is enabled if MNW.INI exists in the current working directory. If the file is found, all preferences are set with to values in this file.

This combination allows preferences which are customized for different types of datasets to reside with those datasets. The file MNW.INI in the MultiNet directory is always loaded when MultiNet starts up, allowing general customized settings to be used. If this file does not exist (has been deleted, renamed or moved) the internal defaults for all preferences are set, and the registration code must also be re-entered to avoid the final "nag" screen.

7.2.8 Save Preferences

Clicking this item produces a menu with two choices:

- **To MultiNet Directory**: When this is selected, the program saves the current preferences in the file MNW.INI in the MultiNet startup directory. The preferences in this file are always loaded when MultiNet starts up, allowing general customized settings to be used.
- **To Current Directory**: When this is selected, the program saves the current preferences in the current working directory. This allows preferences which are customized for different types of datasets to reside with those datasets

MNW.INI may be restored to its initial state by first selecting **Defaults→Restore ALL Defaults**, followed by **Save Preferences→To MultiNet Directory**. Once the registration code has been entered, it is also saved with **Save Preferences→To MultiNet Directory**.

7.2.9 Help

This is a menu item that appears in all the MultiNet modules. Left-click on **Help** opens a selection window which lists all items on the current menu bar. Selecting any of these opens a view window containing details about the menu item. **Help** is also a common button on many other temporary windows, and always provides a context-sensitive description of what the program is doing and what kinds of inputs it expects at the point the **Help** button is pressed.

7.3 Preferences in other modules

As a convenience the following Preferences settings are made available in three other MultiNet modules.

- **Missing data character** Edit window appears for **File→Import** with MultiNet files. It is not needed for CSV files.
- **Preferences** appears as a menu item in the **Analyse** and **Variables** modules. Clicking it produces a menu containing **Delete Empty Categories**, **Number of Categories**, **Number of Bins** and **Link Range**. For **Variables** only **Number of Categories** and **Number of Bins** are enabled. **Link Range** is enabled only for network **Analyse**. **Number of Bins** is enabled only for **Analyse** ANOVA and CORREL.

7.4 Technical appendix

7.4.1 Error messages

Generic values for a numeric variable is represented by n#

`Error Text` is followed by

- Explanation
- Solution

`<n1> IS TOO LARGE! MAXIMUM IS 20`

- Attempt to set more than 20 categories.
- Solution: No more than 20 categories currently allowed.

`<n1> IS TOO LARGE! MAXIMUM IS 100`

- Attempt to set more than 100 bins.
- Solution: No more than 100 bins currently allowed.

`<n1> IS TOO SMALL!  MAXIMUM IS <n2>`

- Attempt to set number of bins less than number of categories.
- Solution: Number of bins must be at least as big as number of categories.

`EXPECTING ONE POSITIVE INTEGER`

- Entry for either Number of Categories or Number of Bins is not a positive integer.
- Solution: Enter a single positive integer in these edit windows.

7.4.2 Bins and Categories.

The setting for number of categories is used to determine whether a variable should be considered as discrete/categorical or numeric/continuous. This is why the number of bins should be no less than the number of categories. In the **Variables** module, any new variable which can be considered discrete/categorical automatically has a list of value labels generated for each distinct category. These names may be changed by selecting **Manage→Labels**. If a new variable has too many categories, it is considered continuous and no value labels are available for editing. If value labels are required and the number of categories is not more than 20, **Preferences→ Number of Categories** can be set to the maximum and the variable re-created. For example, **Recode→Equation** using the single new variable will generate automatic value labels if possible.

The setting for number of bins is approximate. If a variable has no more distinct values than this setting, all are used. Otherwise, the bin size is calculated by dividing the range of the variable by the requested setting and rounding the result to the nearest "nice number" (a multiple of 2,5 or 10) to get the bin size. The new bin size then determines the actual number of bins.

8. Mathematical background

This section is substantially the invited paper (Seary & Richards, 2003) presented at the workshop on Dynamic Social Network Modeling and Analysis organized by the Office for Naval Research and the National Research Council held in November 2002, Washington DC.

8.0 Abstract

We introduce three types of spectral analysis for graphs and describe some of their mathematical properties. We discuss the strengths and weaknesses of each type and show how they can be used to understand network structure. These discussions are accompanied by graphical displays of small (n=100) and moderately large (n=20,000) networks. Throughout, we give special attention to sparse matrix methods which allow rapid, efficient storage and analysis of large networks. We briefly describe algorithms and analytic strategies that allow spectral analysis and identification of clusters in very large networks (n>1,000,000).

8.1 Introduction

A standard method in statistics for handling multivariate data is to find the directions of maximum variability, usually of variance-covariance or correlation matrices. These directions are called Principal Coordinates or *eigenvectors,* while the relative importance of each direction is represented by numbers called *eigenvalues*. (Jolliffe, 1986) Finding this coordinate system may be accomplished by a series of *rotations* (although this is not the most efficient method) that end up pointing along the direction of maximum variability, with the second largest maximum variability at right angles, and so on. As a result, the data matrix is reduced to a diagonal matrix, with diagonal entries corresponding to the *importance* (*eigenvalue*) of each direction (*eigenvector*). The collection of *all* eigenvalues is called the *spectrum*. One goal is to reduce the problem so that only the most important dimensions (those with the largest eigenvalues) contain most of the variability. Implicit in these methods (variance-covariance or correlation) is that some kind of "expected" or "background" signal has been subtracted: in the case of variances, these would be the means of each variable in the original data matrix. To find these *eigenvectors* and *eigenvalues* we need to solve the *eigenvalue equation*:

$$\mathbf{E}\,\mathbf{e} = \varepsilon\,\mathbf{e}$$

(we will derive this equation below) which states that along the direction represented by vector **e**, multiplication by data matrix **E** does not change the direction, but only the length (where ε may be any number, including 0).[1] The related pair (ε,**e**) is called an *eigenpair* of matrix **E**.

A *network* or *graph* G(V,E) is a set of *nodes* V (points, vertices) connected by a set of *links* E (lines, edges). We will consider networks that are *binary* (edges have logical value 1 if an edge exists, 0 if not), *symmetric* (an edge from node *i* to *j* implies an edge from node *j* to *i*), *connected* (there is a set of edges connecting any two nodes, consequently only one *component*), and *without self-loops* (no edges between i and i). We may represent such a network as the *adjacency* matrix **A** = **A**(G) with: 1 in row *i*, column *j* if *i* is connected to *j*,

0 otherwise.

We will not directly discuss *weighted* networks, where the entries for an edge may be a number other than 1, although most of the results that follow generalize to such networks. For many "real world" networks, **A** consists mostly of 0's: it is *sparse*. We will discuss efficient ways of storing and manipulating **A** using *sparse methods*.

Associated with **A** is the *degree distribution* **D**, a diagonal matrix with row-sums of **A** along the diagonal, and 0's elsewhere. **D** describes how many connections each node has. We call the number of nodes, m, the *order* of G and it is equal to the number of rows or columns of **A**. We represent the number of edges by |E|. We will also introduce two other matrices related to **A**:

- the *Laplacian:* $\mathbf{L} = \mathbf{D} - \mathbf{A}$
- the *Normal:* $\mathbf{N} = \mathbf{D}^{-1}\mathbf{A}$

and will discuss the properties of the spectrum and associated eigenvectors of **A**, **L**, and **N**.

8.2 Distances and diameter

One important property of a network is the set of *distances* between any pair of nodes *i* and *j*; that is, the least number of links between any pairs *i* and *j*. One way of calculating this is to take *powers* of the matrix **A** as follows (assuming **A** is not bipartite; see footnote 5):

1st power **A** = **A** by definition gives a matrix of all pairs of nodes linked to each other.

2nd power = **AA** has a non-zero in row *i* column *j* if *j* is at most two steps away from *i* ("at most" since *i* may be directly connected to *j*). Since *i* is 2 steps away from itself, the diagonal *i,i* entry counts the number of these 2-steps.

[1] We have introduced some notation which will be followed throughout:

- matrices are represented by bold capitals: **D**
- (column-)vectors are represented by bold lower case: **e**
- inner products of vectors are represented as $\mathbf{e}^T\mathbf{e}$ = n (a scalar) where $\mathbf{e}^T$ is the transpose of **e**.
- outer products of vectors are represented as $\mathbf{e}\mathbf{e}^T$ = **M** (a matrix)
- eigenvalues are represented by Greek letters, usually with some relationship to the latin letters representing a matrix and an eigenvector. E.g.,(α_i , $\mathbf{a}_i$) is an eigenpair of adjacency matrix **A**.

3rd power = **AAA** has a non-zero entry in row *i* column *j* if *j* is at most 3 steps away from *i*. Eventually, some power of **A**, say $\mathbf{A}^N$, will consist of entirely non-zero entries, meaning every node has been reached from every other node. We call N the *diameter* of the graph: the longest possible path between any pair of nodes.

This is a very inefficient way of calculating the diameter of a graph for two reasons:

1) calculating each power of **A** requires m^3 calculations

2) as more nodes are reached, the powers of **A** become less sparse until eventually no 0's remain: the amount of storage required approaches m^2.

If we continue taking powers of **A**, an interesting thing happens: all the columns become multiples of each other. Taking higher powers of **A** corresponds to taking longer "walks" along the edges and we can interpret this as a "loss of memory" about where we started from (Lovasz, 1995). We will see why this happens soon, as well as other examples of this phenomenon.

We can approach this problem another way by the properties of the *spectral decomposiiton* of **A** (Parlett, 1980). Let α_i be the eigenvalues of **A** and $\mathbf{a}_i$ the corresponding eigenvectors, with $\alpha_0 \geq \alpha_1 \geq \alpha_2 \ldots \geq \alpha_{m-1}$ and $\|\mathbf{a}_i\| = 1$ (the eigenvectors are *normalized* to length 1). Then the spectral decomposition of **A** is:

(1) $\mathbf{A} = \Sigma_i(\alpha_i)\mathbf{a}_i\mathbf{a}_i^T$ where $\mathbf{a}_i\mathbf{a}_i^T$ is an $m\times m$ matrix defining a 1-dimension subspace and $(\mathbf{a}_i\mathbf{a}_i^T)^N = \mathbf{a}_i\mathbf{a}_i^T$ if $i=j$; $\mathbf{a}_i\mathbf{a}_j^T = \mathbf{0}$ if $i \neq j$

therefore $\mathbf{A}^N = \Sigma_i(\alpha_i)^N\mathbf{a}_i\mathbf{a}_i^T$ for any power N, and this allows an easy way of calculating powers of **A**, assuming we have already calculated all the eigenpairs $(\alpha_i, \mathbf{a}_i)$.

Another important property of the spectral decomposition is the *approximation* property. If we take the first k of the eigenpairs $(\alpha_i, \mathbf{a}_i)$, then $\mathbf{A}_k = \sum_{i=0}^{k} \alpha_i \mathbf{a}_i \mathbf{a}_i^T$ is the *best least-squares approximation* to **A**, meaning that we have captured most of the variability of **A** in the important eigenpairs. For example, we can *estimate* an *upper bound* for the diameter using the second-largest eigenvalue α_1 (Chung, 1989):

$$\mathrm{Diam}(G) \leq \lceil \ln(m-1)/\ln(k/\alpha_1) \rceil$$

Unfortunately, this bound applies only to k-*regular* networks (all degrees = $k = \alpha_0$). We will get better bounds for general networks using different spectra. Nevertheless, this bound does show one relationship between the spectrum and an important property like diameter. In particular, when k/α_1 is *large* (there is a large *gap* between the first two eigenvalues), the upper bound on diameter is *small*, so all distances are short.

8.3 The Power Method and Sparse methods

Using **(1)** and eigenpair (α_0 , $\mathbf{a}_0$) we can see why taking large enough powers of **A** results in columns that are multiples of one another – in fact, multiples of eigenvector $\mathbf{a}_0$. This is the basis of the *Power method* (Hotelling, 1933) for finding eigenpairs. We have mentioned that taking powers of matrix **A** is not efficient, so we introduce a representation and methods that are far more efficient.

A very simple way of storing and manipulating a sparse matrix **A** is to use a *link list* representation, which stores only the non-zero entries of **A** as a list of pairs *i,j* for each link in **A**. We could then calculate the diameter of **A** by starting at *i*=1 and following each link until we have reached every node, repeat for *i*=2, and save the maximum number of steps. This requires about m|E| operations (Aho, *et al.*, 1987) and a very moderate amount storage equal to 2|E|. We can now use **(1)** to devise a very efficient version of the Power Method for finding the largest eigenpair:

Starting with some random vector **p** normalized to length 1:

Repeat $\mathbf{p}' \leftarrow \mathbf{Ap}$, $\mathbf{q} \leftarrow \mathbf{p}$, $\mathbf{p} \leftarrow \mathbf{p}'$ until **p** is no longer changing in direction.

Then the largest eigenpair of **A** is (**p**/**q**, **p**). There are some bookkeeping details: **Ap** uses the link list representation, and the entries of **p'** must be adjusted in size after each multiplication (for details see Richards & Seary, 2000), but the method will always work for any matrix *without repeated eigenvalues*, which is generally the case for social networks.

If we want more eigenpairs, we can iterate with

$$\mathbf{p}' \leftarrow \mathbf{Mp} - \alpha_0 \mathbf{a}_0 \mathbf{a}_0^{\mathrm{T}}$$

to get the second, and with

$$\mathbf{p}' \leftarrow \mathbf{Mp} - \alpha_0 \mathbf{a}_0 \mathbf{a}_0^{\mathrm{T}} - \alpha_1 \mathbf{a}_1 \mathbf{a}_1^{\mathrm{T}}$$

to get the third, and so on, without destroying sparsity. However, we must store the (α_i, $\mathbf{a}_i$) eigenpairs somewhere; the procedure is subject to loss of precision on a computer; and the iterations may converge slowly if α_i / α_{i-1} is close to 1. There are better methods, such as Lanczos iteration (Parlett *et al.*,1982) which converge very rapidly and do not have problems with loss of precision.

8.4 Some network invariants

Some properties of **A** remain unchanged (*invariant*) under the series of orthogonal rotations that diagonalize **A** (eigendecomposition). We will relate these to some *network invariants* of **A**.

The eigenvalues of any symmetric matrix **M** are the roots of the *characteristic polynomial*:

$$x^m + c_1 x^{m-1} + c_2 x^{m-2} + c_3 x^{m-3} \ldots + c_{m-1}$$

Therefore, $c_1 = \alpha_0 + \alpha_1 + \ldots + \alpha_{m-1}$ (sum over all eigenvalues) of **A**;

$c_2 = \alpha_0\,\alpha_1 + \alpha_0\,\alpha_2 \ldots + \alpha_0\,\alpha_{m-1} \ldots + \alpha_{m-3}\,\alpha_{m-1} + \alpha_{m-2}\,\alpha_{m-1}$ (sum over all pairs);

$c_3 = \alpha_0\,\alpha_1\,\alpha_2 + \alpha_0\,\alpha_1\,\alpha_3 + \ldots + \alpha_{m-3}\,\alpha_{m-2}\,\alpha_{m-1}$ (sum over all triples)

The *trace* of a matrix is the sum of the entries on the diagonal, and this is invariant under orthogonal rotations. Since **A** has trace of 0 (no self-loops), $c_1 = 0$. The sum of product pairs is equal to minus the number of edges so that $c_2 = -|E|$. Most important is c_3 which is twice the number of triangles in G. Higher coefficients are related to cycles of length 4, 5,... although they also contain contributions for shorter cycles (Biggs, 1993). It appears that the eigenvalues of **A** encode information about the cycles of a network as well as its diameter. We will see related results for the other two spectra.

A *bipartite* network is one that can be partitioned so that the nodes in one part have connections only to nodes in the other part, and vice-versa. Such a network cannot have odd cycles (of any length) and hence no triangles. This means *all* the odd coefficients c_{2k-1} must be 0. It can also be shown (Biggs, 1993) that, in bipartite networks, the eigenvalues occur in pairs with opposite signs, so that $\alpha_0 = -\alpha_{m-1}$ and so on. Bipartite networks can be used to represent *two-mode networks* (Wasserman & Faust, 1994), for example the network relating people and the events they attend.

These results scratch the surface of the information contained in the spectrum of A for k-regular graphs. For general graphs, we need to turn to other spectra.[2]

8.5 The Laplacian spectrum

The Laplacian of a network was originally discovered by Kirchoff (1847). There are a number of definitions and derivations, perhaps the most revealing due to Hall (1970), who was interested in situating the nodes of any network so that total edge lengths are minimized.

He considers the problem of finding the *minimum* of the weighted sum

(2) $$z = 1/2\sum_{i,j}(x_i - x_j)^2\, a_{ij}$$

where a_{ij} are the elements of the adjacency matrix **A**. The sum is over all pairs of squared distances between nodes *which are connected*, and so the solution should result in nodes with large numbers of inter-connections being clustered together.

Equation **(2)** can be re-written as:

$$= 1/2\sum_{i,j}(\mathbf{x}_i^2 - 2\mathbf{x}_i\,\mathbf{x}_j + \mathbf{x}_j^{\,2})\,a_{ij} \qquad = 1/2\sum_i \mathbf{x}_i^{\,2} a_{ij} - 1/2\sum_{i,j} 2\mathbf{x}_i\,\mathbf{x}_j\,a_{ij} + 1/2\sum_j \mathbf{x}_j^{\,2} a_{ij}$$

$$= \sum_i \mathbf{x}_i^2 a_{ij} + \sum_i \sum_{i,j} \mathbf{x}_i\,\mathbf{x}_j\; a_{ij} \qquad = \mathbf{X^T L X}$$

[2] Similar results may be obtained from the *moments* of the eigenvalue distribution (Farkas. *et al.,* 2001; Gho, *et al.*, 2001)

where $\mathbf{L} = \mathbf{D} - \mathbf{A}$ is the Laplacian. In addition to this, Hall supplies the condition that $\mathbf{X}^T\mathbf{X} = 1$, i.e., the distances are normalized. Using Lagrange multipliers (a standard method for solving problems with constraints), we have:

$$\mathbf{z} = \mathbf{X}^T\mathbf{L}\mathbf{X} - \lambda\mathbf{X}^T\mathbf{X}$$

and to minimize this expression, we take derivatives with respect to X to give:

$$\text{(3)} \qquad \mathbf{L}\mathbf{X} - \lambda\mathbf{X} = 0 \text{ or } \mathbf{L}\mathbf{X} = \lambda\mathbf{X}$$

which is the eigenvalue equation. It is not hard to show that $\lambda_0=0$ with $\mathbf{l}_0 = \mathbf{1}$, the constant (or *trivial*) eigenvector, and that $0=\lambda_0 \leq \lambda_1 \leq \dots \lambda_{m-1}$. For **L**, the most "important" eigenvectors belong to the smallest eigenvalues (Pothen, *et al.*, 1990).

It turns out that the discrete network Laplacian shares many important properties with the well-known continuous Laplacian *operator* ∇^2 of mathematical physics. This has led to an explosion of research and results, mostly concerned with λ_1 (Bein, 1991).

The definition of **L** shows that there is no loss of sparsity (except for the diagonal) and that the sparse methods mentioned earlier can be applied to find all or some of the eigenpairs. The requirement that we must find the *smallest* eigenpairs is easily overcome by subtracting a suitably large constant from the diagonal of -**L** (which subtracts that constant from the eigenvalues without changing the eigenvectors).This guarantees that the first eigenpairs returned by the Power Method or Lanczos iteration are associated with the smallest eigenvalues of **L**.

Some of the coefficients of the characteristic polynomial of **L** have an easy interpretation:

$c_1 = \text{Trace}(\mathbf{L}) = 2|E|$ (i.e., twice the number of edges)

$c_{m-1} = 0$ (since 0 is an eigenvalue)

$| m\, c_{m-1} | = \lambda_0\, \lambda_1 \dots \lambda_{m-1} + \lambda_0\, \lambda_2 \dots \lambda_{m-1} + \dots + \lambda_1\, \lambda_2 \dots \lambda_{m-1} = \lambda_1\, \lambda_2 \dots \lambda_{m-1}$ = the number of *spanning trees* of G (this is the Matrix-tree theorem of Kirchoff, 1847).

In general, the eigenvalues of **L** encode information about the *tree-structure* of G (Cvetkovic, *et al.*, 1995). The spectrum of **L** contains a 0 for every connected component. There is no such direct way to find the number of components of a network from the spectrum of **A**. There is also a bound on diameter related to λ_{m-1} and λ_1 for general graphs from Chung, *et al.,* (1994):

$$\text{Diam}(G) \leq \left\lceil \cosh^{-1}(m-1) / \cosh^{-1}((\lambda_{m-1} + \lambda_1)/(\lambda_{m-1} - \lambda_1)) \right\rceil$$

Intuitively, if λ_1 is close to 0, the graph is almost disconnected, while if $\lambda_1 >> \lambda_0$ (an eigenvalue gap) the diameter is small.

8.6 The Normal spectrum

We can repeat the same argument as Hall to derive the Normal spectrum, with the normalization constraint that $\mathbf{X}^T\mathbf{DX} = 1$ (Seary & Richards, 1995) to give:

$\mathbf{LX} = \mu\, \mathbf{DX}$, or assuming that **D** can be inverted,

$$\mathbf{D}^{-1}\, \mathbf{LX} = \mathbf{D}^{-1}\, (\mathbf{D} - \mathbf{A})\mathbf{X} = (\mathbf{I} - \mathbf{D}^{-1}\, \mathbf{A})\mathbf{X} = \mu\, \mathbf{X}$$

where **I** is an *identity* matrix of proper size. In fact, we usually take the defining equation to be

(4) $\quad \mathbf{D}^{-1}\, \mathbf{AX} = \mathbf{NX} = \nu\, \mathbf{X}$ with $\mathbf{D}^{-1}\, \mathbf{A} = \mathbf{N}$ and $\nu = 1-\mu$

since adding an identity matrix shifts the eigenvalues by 1 without changing the eigenvectors. Note that for connected networks **D** not only has an inverse, it also has an inverse *square root* $\mathbf{D}^{-1/2}$.

The *Normal* matrix **N** has a number of interesting properties:

1) It is a *generalized* Laplacian (with a different definition of orthonormality)

2) It therefore has a *trivial* eigenvector $\mathbf{n}_0$ with eigenvalue $\nu_0 = 1$

3) The spectrum of **N** is bounded by $1 = \nu_0\ \geq \nu_1 \ldots \geq \nu_{m-1}\ \geq -1$

4) The rows of **N** sum to 1 (it is a *stochastic* matrix)

5) The spectrum of **N** contains a 1 for every connected component

6) The eigenvalue -1 only occurs if G is bipartite, in which case all eigenvalues occur in pairs.

7) **N** has been rediscovered a number of times: generalized or combinatorial Laplacian (Dodziuk & Kendall, 1985; Chung,1995); Q-spectrum (Cvetkovic, *et al.*,1995).

The descriptive name Normal is suggested by points 2) - 5), although it is not standard terminology.

It is easy to see that there is no loss of sparsity in the definition of **N**. Each 1 in row *i* is simply replaced by $1/\mathbf{D}_{ii}$ and the 0's are unchanged, but **N** is no longer symmetric. However, the matrix $\mathbf{D}^{-1/2}\mathbf{A}\,\mathbf{D}^{-1/2}$ is *similar* to **N** (has the same eigenvalues) and we can apply the sparse methods described above to solve for the eigenpairs $(\nu_i, \mathbf{e}_i)$ and then calculate $\mathrm{D}^{-1/2}\, \mathbf{e}_i = \mathbf{n}_i$ to get the corresponding eigenvectors without losing precision or sparsity.

For **N**, the coefficients of the characteristic equation are harder to interpret except in special cases, but the eigenvalues encode information about *both* the cycle *and* tree structure of G (Cvetkovic, *et al.*, 1995). Some examples:

$c_1 = \mathrm{Trace}(\mathbf{N}) = 0$

$c_3 = c_{2k-1} = 0$ (no triangles or other odd cycles) if G is bipartite

$\prod_i \deg(i)/\sum_i \deg(i)\ \sum_{i=1}(1-v_i)$ = number of spanning trees of G

In the last example we see how details of the degree distribution are *also* encoded in the spectrum. Fan Chung uses this to derive two remarkable bounds (see Chung, 1995 for details):

$$Diam(G) \leq \max \left\lceil \frac{\cosh^{-1}\sqrt{\dfrac{vol\,\overline{X}\ vol\,\overline{Y}}{vol\,X\ vol\,Y}}}{\cosh^{-1}\dfrac{(v_{n-1}+v_1)}{(v_{n-1}-v_1)}} \right\rceil \qquad \min_{i\neq j}\ dist(X_i, X_j) = \max_{i\neq j} \left\lceil \frac{\ln\sqrt{\dfrac{vol\,\overline{X}_i\ vol\,\overline{X}_j}{vol\,X_i\ vol\,X_j}}}{\ln\dfrac{(v_{n-1}+v_k)}{(v_{n-1}-v_k)}} \right\rceil$$

where vol X is the total number of edges in a subset of nodes X⊂V and $\overline{X}$ is V-X. Chung's first bound applies to *any* graph (regular or not) and is much tighter than the previous bound (for the Laplacian). Intuitively, if v_1 is close to 1, the network has a long path or is almost disconnected, and if $v_1 << 1$, the diameter is small.

Chung's second bound describes the distance between subsets for any number k of subsets, based on the kth eigenvalue. The result suggests that we can use the eigenvalues to estimate how many subsets we should look for in a network without forcing distances that are too short (and hence too many subsets).

8.7 Interpreting the Spectra

Many important properties of the spectrum of **A**(G) where G is k-regular are true for **L**(G) and **N**(G), even when G is not regular. Another way of looking at this is that these properties of **A** are true because the spectrum of **A** is simply related to those of **L** and **N** for regular graphs: ($\alpha_i = k-\lambda_i = v_i/k$ for k-regular graphs, with the corresponding eigenvectors being identical). In other words, both **L** and **N** are more natural function of graphs. This point of view is shared by the authors of recent papers on the Laplacian (Grone, *et al.*, 1990, 1994). Mohar (1991) presents a collection of important results relating to the spectrum and eigenvectors of **L**. Chung (1995) has written several papers and a book about **N**.

We return to the goal expressed in the opening paragraph. We would like to find the most important global features of a network, after accounting for what could be considered "expected" for a random network with the same number of nodes and edges. The biggest problem with interpreting the spectrum of **A** is the lack of an "expected" eigenvector (again, except for k-regular graphs). There is a lot of literature on the so-called "main eigenvectors" of **A**: those which have a *projection* on the "all-ones" vector (e.g., Harary & Schwenk, 1979), but the results remain hard to interpret (Cvetkovic and Rowlinson,1990). Both **L** and **N** have an "expected" all-ones eigenvector for which the interpretation is clear (though different in each case).

To interpret **L**, we turn to physical analogy and the relation to ∇^2 as discussed by Friedman (1993). He considers a graph G as a discrete *manifold* (surface) subject to "free" boundary conditions.[3] For illustration, consider ∇^2 as the spatial part of the wave equation (Fisher, 1966, Chavel, 1984). Think of a fishing net subject to no forces. It just lies there at 0 energy with nothing happening. As we subject it to regular oscillations, the net vibrates with the most highly-connected regions moving together. Regions with many triangles tend to have the *same* signs, while those with few triangles are separated into parts with *different* signs. Friedman shows how the Hilbert Nodal theorem (Courant & Hilbert, 1965) can be applied to a discrete network, which generalizes Fiedler's result (described below): the signs of the k^{th} eigenvector partition the network into no more than k+1 *disconnected* components.[4]

To interpret **N**, we have a number of choices:

1) **N** is the Laplacian for a network of nodes, each *weighted* by its degree

2) **N** is the transition matrix of a Markov Chain for a simple random walk on the nodes

3) $\mathbf{N}^2$ is similar to the χ^2 matrix, thus treating **A** as a *contingency table*

The first leads to a physical analogy similar to that for **L**, so we consider 2) and 3):

8.8 The Normal spectrum and Random walks

Specifically, we consider nearest-neighbour random walks on a network (Lawler & Sokal, 1988). Define the probability-transition matrix for such a walk as

$$\mathbf{N} = \mathbf{D}^{-1}\,\mathbf{A}$$

Then the probability of moving from vertex *i* to any vertex adjacent to *i* is uniform. **N** is a row-stochastic matrix, and the random walk is a Markov chain. In this case **1** (the trivial all-ones eigenvector) is related to the *stationary state* of the Markov Chain: the probability is $1 = \nu_0$ that such a probability distribution is eventually reached.[5] The vector $\mathbf{p}_0 = \mathbf{1}^T\,\mathbf{N} = \mathbf{N}^T\,\mathbf{1}$ is the stationary state,

[3] no external constraints need to be satisfied

[4] This interpretation of the eigenvectors may be even more useful when considering ∇^2 as the spatial part of the *Diffusion* equation (for example, when considering diffusion of innovation or disease).

[5] A problem can arise with bipartite graphs: $\mathbf{p}_0$ does not exist since the chain oscillates between the two sets of vertices (period = 2). Probabilists deal with this by a simple trick: divide **N** by 2 and add a self-loop of probability 1/2 to every vertex: $\mathbf{N'} = \mathbf{I}/2 + \mathbf{N}/2$ The eigenvalues of **N'** are then: $1 = \nu'_0 \leq \nu'_1 \leq \ldots \leq \nu'_{m-1} \leq 0$ so that $\nu' = (1+\nu)/2$ and again the eigenvectors are not affected. In effect, this suppresses *all* negative

and it is *proportional* to the degree distribution. The second eigenpair (ν_1, $\mathbf{n}_1$) has become important in the analysis of *rapidly mixing* Markov chains – those that reach the stationary state quickly (Sinclair, 1995). From the previous discussion it should not be surprising that these are associated with $\nu_1 << 1$ (a large eigenvalue gap), which means that the walk quickly "forgets" where it started.[6] Moreover, when ν_1 is close to 1, there must be parts of the network that are hard to reach in a random walk, implying long paths or a nearly disconnected network.

8.9 Normal spectrum and χ^2

The χ^2 matrix is defined in terms of the row and column *marginals* (sums). A typical element is (Observed$_{ij}$ - Expected$_{ij}$)^2 /Expected$_{ij}$ which is not sparse. For a sparse network **A**, consider χ which has a typical element (Observed$_{ij}$ - Expected$_{ij}$) /√ Expected$_{ij}$ where

$$\text{Expected}_{ij} = \frac{\deg(i)\deg(j)}{\sum \deg(i)}$$

We can write χ as: $\frac{O}{\sqrt{E}} - \sqrt{E}$ so that non-zero elements of **A** become $A_{ij} / \sqrt{E_{ij}}$ while the 0 terms are unaffected, maintaining sparsity. The second term corresponds to the trivial eigenvector which can be dealt with separately. In matrix notation $\chi = \mathbf{D}^{-1/2}\mathbf{A}\,\mathbf{D}^{-1/2}$ which has eigenpairs (ν_i, $\mathbf{D}^{1/2}\mathbf{n}_i$) Thus we have

(5) $\chi^2 = \sum_{j=1} \nu_i^2 / \sum_i a_{ij}$ (omitting the $\nu_0 = 1$ expected term for $\mathbf{n}_i = \mathbf{1}$)

This equation shows how much each dimension contributes to χ^2 which is a measure of *dependence* between rows and columns. In this interpretation, if ν_1 is small ($\nu_1 << \nu_0 = 1$), then χ^2 is also small: there is no relation between rows and columns of **A**, and so there is no "signal" above the expected "background". If ν_1 is close to 1, then χ^2 will be large and there is a relation between rows and columns of **A**, with the first eigenvector pointing in the direction of the maximum variability in χ^2. If $\nu_2, \nu_3, \ldots \nu_k$ are also large, we need k+1 eigenvectors to describe the patterns in the χ^2 matrix. With **(5)** we can tell how many eigenvectors we need to explain most of the χ^2 of the network.[7]

eigenvalues. However, negative eigenvalues are useful for directed and bipartite networks.

[6] As we saw, $\mathbf{A}^K$ shows a similar phenomenon, but there is no simple relation to **D**, due to "leakage" from all the "main eigenvectors" (Cvetkovic, *et al.*, 1988)

[7] While PCA results tell how much of the variance each dimension accounts for, the Normal eigenvalues tell how much of the network's chi-squared each eigenvector (dimension) accounts for.

8.10 Compositions

The *Kronecker product* of two binary matrices $\mathbf{A}_1$ and $\mathbf{A}_2$ makes a *copy* of $\mathbf{A}_1$ for each 1 in $\mathbf{A}_2$. It is well-know that for two matrices $\mathbf{A}_1$ and $\mathbf{A}_2$ of order m_1 and m_2 the eigenpairs of the Kronecker product $\mathbf{A}_1 \otimes \mathbf{A}_2$ *behave well* (West, 1996):

If $\mathbf{A}_1$ has eigenpairs $(\alpha_i, \mathbf{a}_i)$ and $\mathbf{A}_2$ has eigenpairs $(\beta_j, \mathbf{b}_j)$, then

$\mathbf{A}_1 \otimes \mathbf{A}_2$ has eigenpairs $(\{\alpha_i \times \beta_j\}, \{\mathbf{a}_i \otimes \mathbf{b}_j\})$

It is also well-known that $\mathbf{A}_1$ and $\mathbf{A}_2$ behave well under *Cartesian sum*:

$\mathbf{A}_1 \oplus \mathbf{A}_2 = \mathbf{A}_1 \otimes \mathbf{I}_2 + \mathbf{A}_2 \otimes \mathbf{I}_1$ (where $\mathbf{I}_1$ and $\mathbf{I}_1$ are identity matrices of appropriate size)
has eigenpairs $(\{\alpha_i + \beta_j\}, \{\mathbf{a}_i \otimes \mathbf{b}_j\})$.

The Laplacian $\mathbf{L}$ also behaves well under Cartesian sum. For $\mathbf{L}_1 \oplus \mathbf{L}_2$ the eigenpairs are $(\{\lambda_i + \kappa_j\}, \{\mathbf{l}_i \otimes \mathbf{k}_j\})$

This fact is used by Pothen, *et al.*, (1990) to study the Laplacian of grids, which are Cartesian sums of paths. Further, the eigenvalues of $\mathbf{L}_1$ and $\mathbf{L}_2$ always contain a $\lambda_0 = 0$ with corresponding constant eigenvector, so that the corresponding eigenpairs of $\mathbf{L}_1 \oplus \mathbf{L}_2$ are $(\lambda_i + 0, \mathbf{l}_i \otimes \mathbf{1})$. The term $\mathbf{l}_i \otimes \mathbf{1}$ means that the components of $\mathbf{l}_i$ are *replicated* m_2 times. Since the Cartesian sum of two paths is a grid, this produces a perfectly rectangular representation (Fig. 8.1b). The Laplacian is therefore a useful tool in problems involving regular grids (or hypergrids). However, $\mathbf{N}$ does *not* behave well under Cartesian sum (Figure 8.1c).

The Laplacian does *not* behave well under Kronecker product. However, the Normal spectrum *does* (Chow, 1997), so that $\mathbf{N}_1 \otimes \mathbf{N}_2$ has eigenpairs

$(\{\nu_i \times \mu_j\}, \{\mathbf{n}_i \otimes \mathbf{m}_j\})$

Further, the eigenvalues of $\mathbf{N}_1$ and $\mathbf{N}_2$ always contain a $\nu_0 = 1$ with corresponding constant eigenvector, so that the corresponding eigenpairs of $\mathbf{N}_1 \otimes \mathbf{N}_2$ are $(\nu_i \times 1, \mathbf{n}_i \otimes \mathbf{1})$. The term $\mathbf{n}_i \otimes \mathbf{1}$ means that the components of $\mathbf{n}_i$ are *replicated* m_2 times. Because all the coordinates are the same within each copy, this produces *clustering* of the components of $\mathbf{N}_1 \otimes \mathbf{N}_2$ for these eigenvectors.

It appears that the behavior under Kronecker product explains why both the Adjacency and Normal eigenvectors are good at detecting both on- and off-diagonal *blocks* (clusters of edges).

8.11 Visualization

The Laplacian **L** can provide good visual representations of graphs which are Cartesian sums (such as grids and hypercubes); while **N** can provide good visual representations of graphs which are Kronecker products (such as graphs consisting of blocks). The reasons for this are suggested above and have mostly to do with the behavior of eigenpairs which are sums and products with 0 and 1, respectively. For graphs that are not k-regular, eigenpairs of **A** do not provide such good representations since, in general, there is no constant (expected, trivial) eigenvector to combine with.

Another way of describing these results is to consider the relationship between the eigenvector components for a node and those it is connected to (Seary & Richards, 1999). It is evident from the definition of eigendecomposition that (where "u~v" means "u is connected to v")

(6) $\mathbf{a}_i(u) = \Sigma_{u\sim v}\mathbf{a}_i(v)/\alpha_i$ for eigenpair i of **A**

(7) $\mathbf{l}_i(u) = \Sigma_{u\sim v}\mathbf{l}_i(v)/(\lambda_i\text{-deg}(u))$ for eigenpair i of **L**

(8) $\mathbf{n}_i(u) = \Sigma_{u\sim v}\mathbf{n}_i(v)/(v_i \times \text{deg}(u))$ for eigenpair i of **N**

Note that **A** has no control for node degree. Consider the effect for "important" eigenpairs: ($|\alpha| \simeq k \gg 1$, $\lambda \simeq 0$ and $|v| \simeq 1$) when deg(u) is small, **a**(u) will be folded toward the origin, while **l**(u) and **n**(u) will sit further *away* from the origin than its neighbours. This effect makes it difficult to interpret visual representations based on **A**, except for k-regular graphs where all three spectra are essentially the same (Figure 8.1)

The equation for $\mathbf{n}_i$ shows that for v_i near 1, each node is approximately *at the centroid* of those it is connected to. The exact difference from the centroid for node *u* of eigenvector $\mathbf{n}_i$ is:

$$\mathbf{n}_i(u)\text{-}\ \Sigma_{u\sim v}\mathbf{n}_i(v)/\text{deg}(u)) = (1\text{-}v_i)\mathbf{n}_i$$

For important eigenvalues v_i near 1, this produces very good visualization properties. In addition, the eigenvector representation may be combined with derived properties such as *betweenness* (Freeman, 1979) to produce very helpful displays of large networks (Brandes *et al.*, 2001)

8.12 Interpreting the eigenvectors

8.12.1 Partitions

Powers (1988) and others have shown how eigenvectors of **A** can be used to find partitions of highly connected subsets (clusters) of nodes, but these methods are not as general or as clear as those derived from **L** or **N**.

The first non-trivial eigenvector $\mathbf{l}_1$ of **L** is the subject of extensive literature (Lubotzky, 1994; Alon & Millman,1985). Fiedler (1975) first suggested that the eigenvector $\mathbf{l}_1$ associated with the *second-smallest* eigenvalue λ_1 could be used to solve the *min-cut* problem: separate the network into two approximately equal sets of nodes with the fewest number of connections between them, based on the *signs* of the components of $\mathbf{l}_1$.[8] In fact, more recent derivations of **L** use the min-cut property as a starting point (Walshaw, *et al.*, 1995) and the results are used to partition compute-intensive problems into sub-processes with minimal inter-process communication (Pothen, *et al.*, 1990). This technique is called *Recursive Spectral Bisection* (Simon, 1991). Other researchers have used $\mathbf{l}_2$, $\mathbf{l}_3$ and higher eigenvectors to produce multi-way partitions of networks (Hendrickson & Leland, 1995).

The graph bisection problem (Mohar & Poljak.,1991) is to find two nearly equal-sized subsets $V_1,V_2 \subset V$ such that $\text{cut}(V_1,V_2) = \Sigma_{ij}\, a_{ij}$ is minimized, where $i \in V_1$, $j \in V_2$. (i.e. nodes in V_1 and V_2 have few connections to each other).

This problem is known to be NP-hard (Garey & Johnson, 1979), but a good approximation is given by the *signs* of $\mathbf{l}_1$ (Walshaw & Berzins,1995). This gives two sets of nodes of roughly the same size, but has no control for the number of *edges* in each part, and so any clustering of nodes is a *side-effect* of the partition.

However, we can add an additional constraint that the number of edges in each part also be roughly equal by *weighting* the node sets by their total degrees. This is exactly what a partition based on $\mathbf{n}_1$ from **N** gives us, since $\mathbf{n}_1$ points in the direction of *maximum variability* in χ^2 (Greenacre, 1984).[9] Similarly, further partitions based on $\mathbf{n}_2$, $\mathbf{n}_3$, ...will also produce sets of nodes with a large number of edges in common (as long as v_2, v_3,...make significant contributions to χ^2). Partitions based on *positive* eigenvalues will produce blocks *on the diagonal* of **A** of edges associated with each set of nodes, while those based on *negative* eigenvalues produce nearly bipartite *off-diagonal* blocks (which occur in pairs if the network is symmetric) (Seary & Richards,1995).

8.12.2 Clustering

Ideally, the important eigenvectors should be at least *bimodal* to induce clustering based on sign-partitions, and often they are *multi-modal* (Hagen ,1992), suggesting that standard clustering methods can be used on the coordinates of these vectors. Equations (7) and (8) show that **L** and **N** place nodes approximately at the centroids of their neighbours. For **N**, the distances are actually measured in χ^2 space, meaning that nodes with very similar *patterns* of connections will be close together (Benzecri,

[8] Hagen also uses deviations from median

[9] see Dhillon (2001) for a formal derivation and proof

1992). This clustering happens with either positive or negative eigenvalues (on- or off-diagonal). The latter are important in *nearly bipartite* networks with few triangles (Figure 8.3).

8.12.3 Problems

Farkas, *et al.*, (2001) and Goh, *et al.*, (2001) report that the important eigenvectors of **A** are very *localized* on nodes of high degree, and suggest that this effect may be used to distinguish certain types of networks. This effect does not occur for **L** or **N** (Figure 8.2), since each include some control for degree, and so far no similar results for distinguishing network types have been reported for these spectra. The biggest problem for **L** and **N** is their sensitivity to "long paths", especially to pendant trees attached to the main body of the network (Seary & Richards, 2000). For **N**, these may be interpreted as nodes that are hard to reach (distant) in a random walk. For long paths *internal* to a network, this effect is actually an advantage, since these cycles are detected as "locally bipartite" and emphasized in important eigenvectors. Nodes on such paths can have a large effect on global properties such as diameter (Figure 8.3-8.4).

8.13 Two-mode networks

Two-mode networks mix two different kinds of nodes and connections. A simple example is an affiliation network such as people and the events they attend. We could be interested in finding sets of people with events in common (or, equivalently, sets of events attended by the same people): this is an example of *co-clustering*. Affiliation networks can be represented by bipartite graphs for which **A** and **N** are most suited, since they have symmetric spectra for these (the eigenvalues occur in pairs with opposite sign). Because of this we don't need the entire bipartite matrix: we can work with the rectangular representation, and infer the missing parts of the eigendecomposition. If we assume m_1 people and m_2 events, the resulting eigenvectors consist of m_1 components for people followed by m_2 components for events. The resulting blocks will be strictly off-diagonal and once again the eigenvectors of **N** provide a superior solution by maximizing χ^2. In fact, this solution is identical to that provided by Correspondence Analysis, a statistical technique for finding patterns in 2-mode data (Benzecri, 1992).

8.14 Partial Iteration

For large networks, it is not necessary or desirable to calculate the entire eigendecomposition. For *very* large networks, it may not be possible in terms of time and space to calculate even a *few* eigenpairs. Nevertheless, it is possible to get at a large amount of the global and local network structure by partially iterating using the Power Method. A few iterations of $\mathbf{N} = \mathbf{D}^{-1}\mathbf{A}$, with each iteration placing nodes at the means of their neighbours, will produce a *mixture* of the most important eigenvectors. Consider the spectral representation

$$\mathbf{N}^{K} = \Sigma_i(\nu_i)^K \mathbf{n}_i\mathbf{D}\mathbf{n}_i^T$$

We know that $(\mathbf{n}_i\mathbf{D}\mathbf{n}_i^T)^K = \mathbf{n}_i\mathbf{D}\mathbf{n}_i^T$ for all K, so the contributions of eigenvectors with small ν_i quickly drop out as these $(\nu_i)^K$ approach 0. This means that $\mathbf{N}^K$ is dominated by the dimensions with ν_i near 1. We start with a random vector, and quickly (6-10 iterations) produce such a mixture. Moody (2001) describes a procedure in which this process is repeated a number of times, each producing a slightly different mixture of the important eigenvectors (Figure 8.5). The results are then passed to a standard cluster analysis routine (such as k-means, Ward's method) to find any clusters of nodes.

8.15 Further analysis

The method of partial iteration of **N** has been used for years in the program NEGOPY (Richards and Rice, 1981; Richards, 1995), as the first step in a more complex analysis. A key concept in NEGOPY is that of *liaisons*. These are nodes which do not have most of their connections with members of a cohesive cluster of nodes, but rather act as connections *between* clusters (Figure 8.3-4). Often it is the liaisons that provide the connections that hold the whole network together. Finding the liaisons requires detailed knowledge about the members of (potential) clusters and *their* connections, and is not an immediate result of a partition based on eigenvectors or clustering methods. Nevertheless, eigendecomposition methods – full or partial – are an excellent strategy to begin such analysis.

8.16 Future prospects

More work needs to be done on the categorization of networks based on important eigenpairs of **L** and **N**. Recent reports (Koren, *et al.*, 2002; Walshaw, 2000) suggest we might not need to resort to partial methods after all; we can find important eigenpairs exactly for enormous networks ($m>10^5$) using "small" amounts of time and memory by first *reducing* the network in some way by sampling, solving the reduced eigenproblem, then interpolating back up with a very good "first guess" for the Power Method. Preliminary tests show that this should work equally well for Lanczos iteration.

8.17 References

Aho, A., Hopcroft, J., & Ullman, J. (1997). *Data structures and algorithms* Addison-Wesley.

Alon, N. and Millman, V. (1985). λ_1, Isoperimetric Inequalities for Graphs, and Superconcentrators. *J. Comb. Theory B.* **38**: 73-88.

Barnard S. and Simon, H. (1994). Fast Implementation of Recursive Spectral Bisection for Partitioning Unstructured Problems. *Concurrency: Practice and Experience* **6** (2): 101-117.

Benzecri, J-P. (1992). *Correspondence Analysis Handbook*, Marcel Dekker Inc.

Bien, F. (1989). Constructions of Telephone Networks by Group Representations. *Notices of the Am. Math. Soc.* **36**: 5-22.

Biggs, N. (1993). *Algebraic Graph Theory.* Cambridge University Press.

Brandes, U., Cornelsen, S. (2001). Visual Ranking of Link Structures, *Lecture notes in Computer Science*, 2125, Springer-Verlag.

Chavel, I. (1984). *Eigenvalues in Riemannian Geometry,* Academic Press.

Chow, T.Y. (1997). The Q-spectrum and spanning trees of tensor products of bipartite graphs", *Proc. Am. Math. Soc.*, **125** (11): 3155-3161.

Courant, R. and Hilbert, D. (1966). *Methods of Mathematical Physics.* Interscience Publishers.

Cvetkovic , D., Doob, M., and Sachs, H. (1995). *Spectra of Graphs.* Academic Press.

Cvetkovic, D., Rowlinson, P. (1990). The largest eigenvalue of a graph: a survey. *Linear and Multilinear Algebra.* **28**: 3-33.

Cvetkovic, D., Doob, M. Guttman, I, Torgasev, A. (1988). *Recent results in the theory of graph spectra*, North Holland.

Chung, F.K.R.(1988). Diameters and Eigenvalues, *J. Am. Math. Soc.* **2** (2): 187-196.

Chung, F.K.R, Faber, V.& Manteuffel, T.A. (1994). An upper bound in the diameter of a graph from eigenvalues associated with its Laplacian, *SIAM J. Disc. Math.* **7** (3): 443-457.

Chung, F.K.R. (1995). *Spectral Graph Theory,* CBMS Lecture Notes, AMS Publication.

Dhillon, I.S. (2001). Co-clustering documents and words using bipartite spectral graph partitioning, UT CS Technical Report #TR 2001-05.

Diaconis, P. and Stroock, D. (1991). Geometric Bounds for Eigenvalues of Markov Chains, *Ann. Appl. Prob.* **1**: 36-61.

Dodziuk, J. and Kendall, W. S. (1985). Combinatorial Laplacians and Isoperimetric Inequality. In K. D. Ellworthy (ed.). *From Local Times to Global Geometry*, Pitman Research Notes in Mathematics Series **150**: 68-74

Farkas, I., Derenyi, I., Barabasi, A-L., Viscek, T. (2001). Spectra of "Real-world" graphs: beyond the semi-circle law, cond-mat/100235

Fiedler, M. (1973). Algebraic Connectivity of Graphs, *Czech. Math. J.* **23**: 298-305.

Fisher, M. (1966). On hearing the shape of a drum, *J. Combin. Theory,* **1**: 105-125.

Freeman, L. (1979). Centrality in social networks: Conceptual clarification, *Social Networks,* **1**: 215-239.

Friedman, J. (1993). Some Geometrical Aspects of Graphs and their Eigenfunctions. *Duke Mathematical Journal,* **69**: 487-525.

Garey, M., Johnson, D. (1979). *Computers and Intractability.* W. H. Freeman.

Goh, K-I., Kahng, B. Kim, D. (2001). Spectra and eigenvectors of scale-free networks, cond-mat/0103337.

Greenacre, M., (1984). *Theory and Application of Correspondence Analysis.* Academic Press.

Grone, R., Merris, R. and Sunder, V. (1990). The Laplacian Spectrum of a Graph. *SIAM J. Matrix Anal. App.* **11** (2): 218-238.

Grone, R. and Merris, R. (1994). The Laplacian Spectrum of a Graph II. *SIAM J. Discrete Math.* **7** (2): 221-229.

Hagen, L. (1992). New Spectral Methods for Ratio Cut Partitioning and Clustering, *IEEE Trans. CAD,* **11** (9): 1074-1085.

Hall, K. (1970). R-dimensional Quadratic Placement Algorithm, *Management Sci.* **17**: 219-229.

Harary, F. & Schwenk, A. (1979). The spectral approach to determining the number of walks in a graph, *Pacific J. Math.* **80**: 443-449.

Hendrickson,B. and Leland, R. (1995). An improved spectral graph partitioning algorithm for mapping parallel computations. *SIAM J. Comput. Sci.* **16** (2): 452-469.

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components, *J. Educ. Psychol.* **24**: 417-441, 498-520.

Joliffe, I.T. (1986). *Principal Components Analysis*, Springer-Verlag, New York.

Kirchoff, G. (1847). Über die Auflösung der Gleichungen, auf welche man bei derUntersuchung der linearen Verteilung galvanischer Ströme gefürte wird, *Ann. Phys.Chem*, **72**: 497-508

Koren, Y. Carmel, L., Harel, D. (2002). ACE: a fast multiscale eigenvector computation for huge graphs, *IEEE Symposium On Information Visualization.*

Lawler, G. F. and Sokal, A. D. (1988). Bounds on the L^2 Spectrum for Markov Chains and Markov Processes: a Generalization of Cheeger's Inequality, *Trans. Amer. Math. Soc.* **309**: 557-580.

Lovasz, L. (1995). Random Walks, Eigenvalues and Resistance, Handbook of Combinatorics, Elsevier, 1740-1745.

Lubotzky, A. (1994). *Discrete Groups, Expanding Graphs, and Invariant Measures*, Birkhauser.

Mohar, B. (1991). The Laplacian Spectrum of Graphs. In Alavi, Chartrand, Ollermann and Schwenk (eds.). *Graph Theory Combinatorics and Applications*, Wiley, 871-898.

Mohar, B., Poljak, S. (1991). Eigenvalues and the Max-cut Problem, *Czech. Math. J.* **40**: 343-352.

Moody, J. (2001). Peer influence groups: identifying dense clusters in large networks, *Social Networks*, **23**: 261-283.

Parlett, B. (1980). *The Symmetric Eigenvalue Problem*, Prentice-Hall.

Parlett, B., Simon, H. and Stringer, L. (1982). On Estimating the Largest Eigenvalue with the Lanczos Algorithm. *Mathematics of Computation.* **38** (157): 153-165.

Pothen, A., Simon, H, and Liou, K-P., (1990). Partitioning Sparse Matrices with Eigenvalues of Graphs. *SIAM J. Matrix Anal. App.* **11** (3): 430-452.

Powers, D. (1988). Graph partitioning by eigenvectors. *Linear Algebra Appl.* **101**: 121-133.

Richards, W.D. & Seary, A.J. (2000). Eigen Analysis of Networks, *J. Social Structure.* http://www.heinz.cmu.edu/project/INSNA/joss/index1.html

Richards, W.D. and Rice, R. (1981). The NEGOPY Network Analysis Program, *Social Networks*, **3** (3): 215–224.

Richards, W.D.(1995). *NEGOPY 4.30 Manual and user's Guide.* School of Communication, SFU. http://www.sfu.ca/~richards/Pdf-ZipFiles/negman98.pdf

Seary, A.J. & Richards, W.D. (2000). Negative eigenvectors, long paths and p*, Paper presented at INSNA XX, Vancouver BC. http://www.sfu.ca/~richards/Pages/longpaths.pdf

Seary, A.J. and Richards, W.D. (1998). Some Spectral Facts. Presented at INSNA XIX, Charleston. http://www.sfu.ca/~richards/Pages/specfact.pdf

Seary, A.J. and Richards, W.D. (1995). Partitioning Networks by Eigenvectors. Presented to European Network Conference, London. Published in Everett, M.G. and Rennolls, K. (eds). (1996). *Proceedings of the International Conference on Social Networks,* Volume 1: Methodology. 47-58.

Simon,H. (1991). Partitioning of Unstructured Problems for Parallel Processing. *Computing Systems in Eng.* **2** (2/3): 135-148.

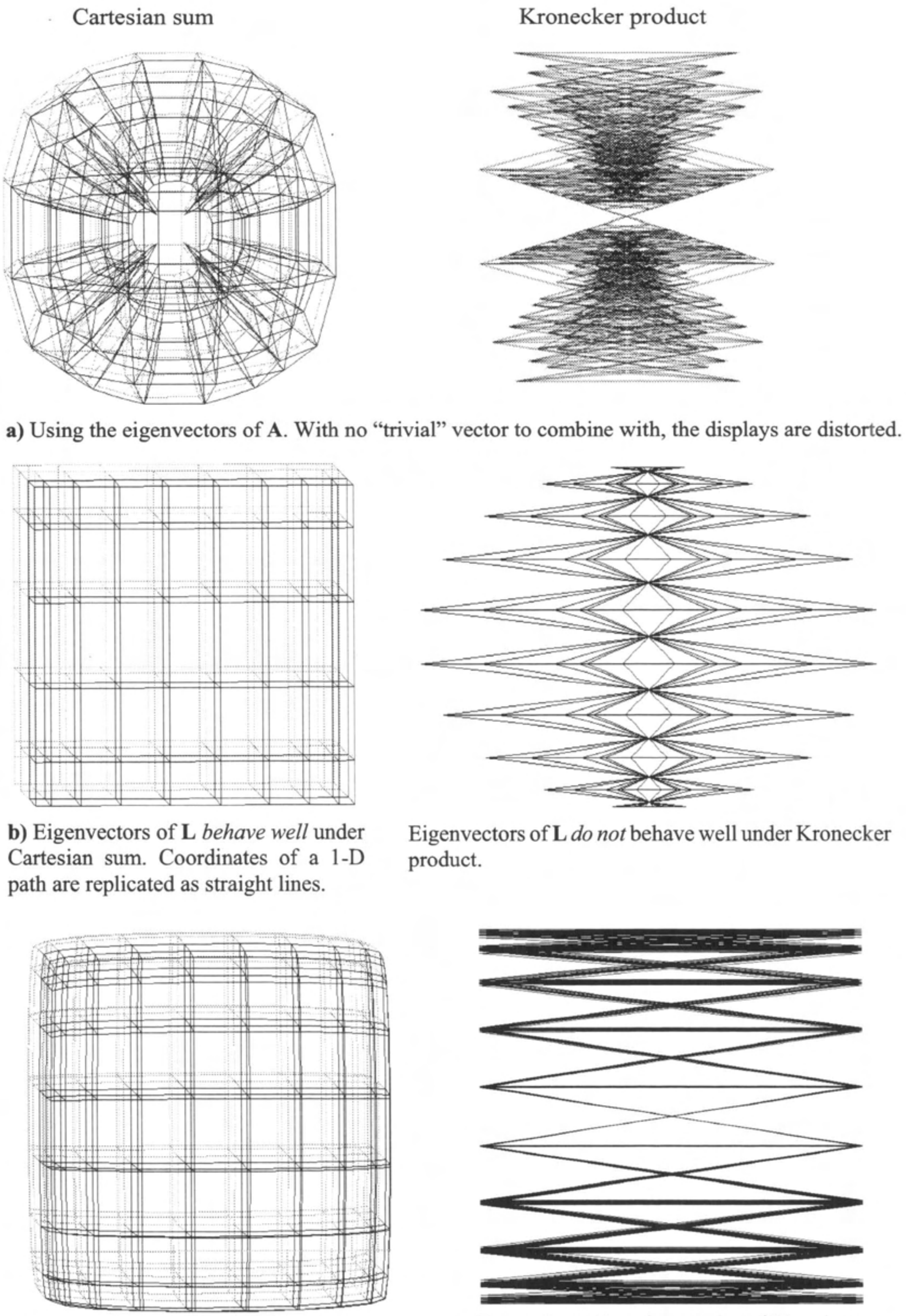
Sinclair, A. (1993). *Algorithms for Random Generation and Counting*, Birkhauser

Walshaw, C. (2000). A multilevel algorithm for force-directed graph drawing, *Proc. Graph Drawing 2000,* Lecture Notes in Computer Science, **1984**: 171-182.

Walshaw, C. and Berzins, M. (1995). Dynamic Load-balancing for PDE Solvers on Adaptive Unstructured Meshes, *Concurrency: Practice and Experience.* **7** (1): 17-28.

Wasserman, S. & Faust, K. (1994). *Social Network Analysis,* Cambridge

West, D. B. (1996). *Introduction to graph theory*, Prentice Hall, 1996.



Cartesian sum

Kronecker product

a) Using the eigenvectors of **A**. With no “trivial” vector to combine with, the displays are distorted.

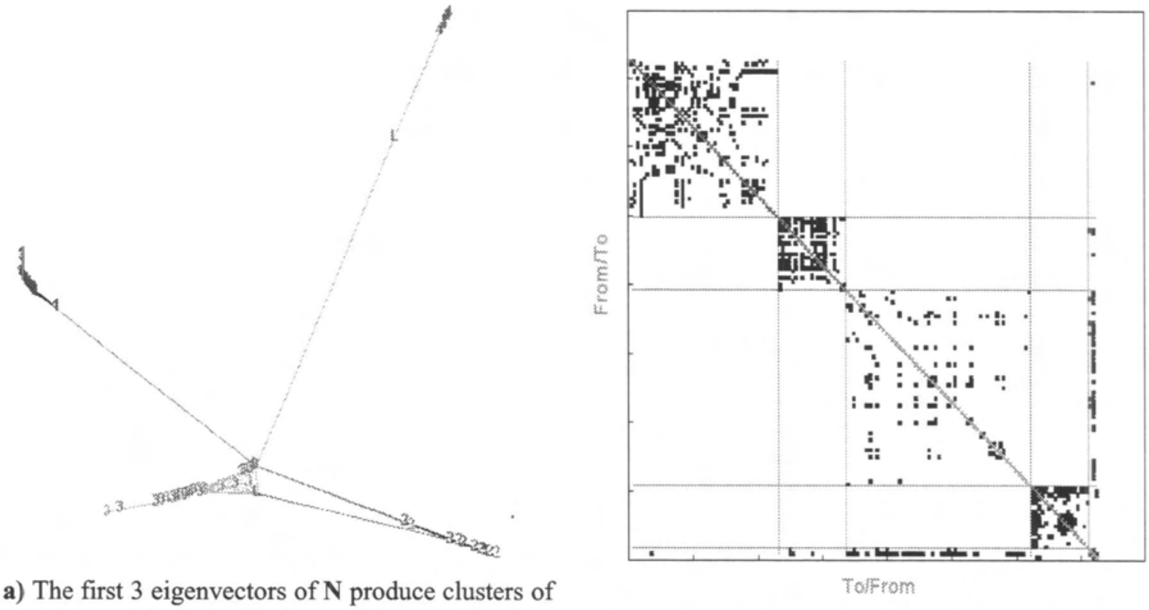
b) Eigenvectors of **L** *behave well* under Cartesian sum. Coordinates of a 1-D path are replicated as straight lines.

Eigenvectors of **L** *do not* behave well under Kronecker product.

c) Using eigenvectors of **N**. Lines are slightly curved, so the cube is slightly distorted.

Largest negative eigenvector of **N** captures bipartiteness perfectly.

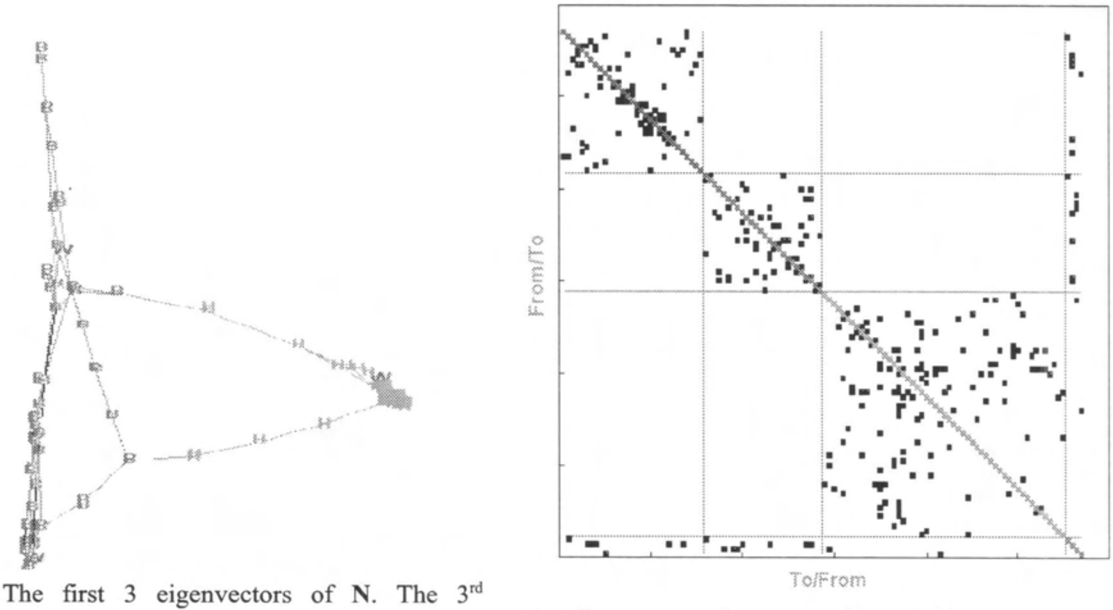
Figure 8.1. Six views of a 6 x 8 x 10 grid. Left: as the Cartesian sum of three paths producing a three-dimensional grid. Right: as the Kronecker product producing a bipartite graph.



a) The first 3 eigenvectors of **N** produce clusters of nodes, here labelled with the resulting partition into 4 blocks, along with the liaisons. The central liaison is of high degree and holds the network together. The figure is slightly rotated to show the clusters.

b) Adjacency matrix permuted by partition numbers. The blocks have no interconnections; the network is held together by the liaisons (right and bottom).

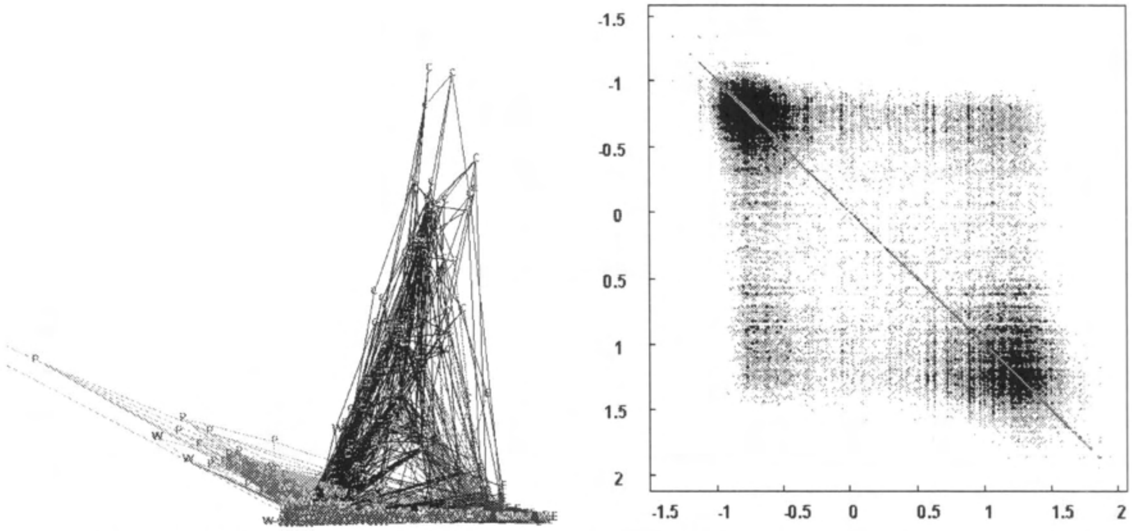
Figure 8.2. Two views of a small social network with 145 nodes. (Data Source: L. Koehly)



a) The first 3 eigenvectors of **N**. The 3rd eigenvalue is negative, since there are very few triangles in the rightmost cluster. The labeling is by ethnic group, which shows a close relation to structure. There are only two connections between the two main ethnic groups.

b) Adjacency matrix permuted by partition numbers. In this case the liaisons are of *low* degree.

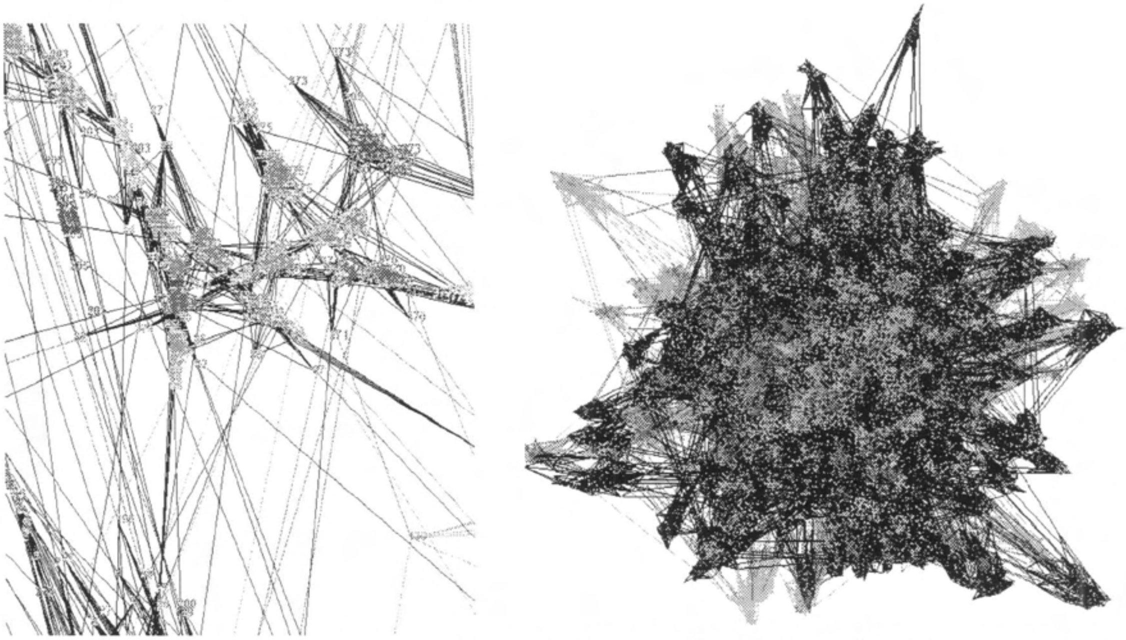
Figure 8.3. Two views of a small social network of drug users with 114 nodes. (Data Source: Scott Clair)



a) This figure shows the close relation between the first 3 eigenvectors of **N** and four needle exchange sites, which are used to label the nodes. The figure is rotated to make this clear.

b) This figure is like an Adjacency matrix, except links are located by the *coordinates* of the first eigenvector of **N**. The network is dominated by exchanges *within* sites E and W.

Figure 8.4. Two views of a needle exchange network. (Data Source: T. Valente and R. Foreman). This network is moderately large (N=2736) and roughly *scale-free* (k≈1.7). The eigenvectors of **A** are dominated by nodes of high degree (>100).



a) Close-up of clusters formed by the first 3 eigenvectors of **N**, labeled by construction.

b) Clusters formed by placing each node at the centroid of its neighbours, iterating 8 times with 3 random starts (multiple partial iteration). Labelled by construction.

Figure 8.5. A moderately large (N=20,000) artificial network (Data Source: J. Moody) constructed for testing purposes. The network was constructed from tightly connected groups of 50 nodes, with 400 of these groups more loosely connected into a single component.

9. Normal eigenspace and negative eigenvalues

9.0 Introduction

This section contains a comparison between Normal eigenspace and the spaces and eigenvalues of Correspondence Analysis (Greenacre, 1984) . The special properties of Normal analysis are exploited in four examples, with increasing emphasis on the use (and usefulness) of negative eigenvalues. These examples display results available in the **Eigenspaces** module. Other MultiNet modules (such as **Variables**, **Pstar** and **Analyse**) were also used for recoding variables, modelling and testing hypotheses.

9.1 Comparison of Normal and Correspondence Analysis

In a series of papers (Seary & Richards, 1995; 1998; 1999; 2000), the author developed a coherent theory relating the graph-theoretic Normal spectrum to the statistical technique known as Correspondence Analysis. MultiNet always computes the eigenspaces of the symmetrized, binarized network and treats the two spectra differently (see Technical appendix 5.7).

- The Normal matrix has a trace of 0 (the diagonal consists of 0s: self-loops are ignored), so that the spectrum consists of both positive and negative eigenvalues. In the extremes, eigenvalues of 1 are associated with disconnected components, while eigenvalues of -1 are associated with bipartite graphs. The positive Normal eigenvalues are shown to measure on-diagonal clustering: a fact exploited by NEGOPY (Richards & Seary, 2000). The negative eigenvalues measure off-diagonal clustering associated with local bipartiteness (Richards & Seary, 1997)

- Correspondence Analysis uses Singular Value Decomposition (SVD) (Press et. al., 1986) to find the related eigenspaces of the symmetric matrices of rows and columns formed by pre- and post-multiplying the data matrix by its transpose. Because of this squaring, the SVD “hides” signs of the singular values in the component signs of the singular vectors. This can lead to misleading results, where an important (near 1) CA singular positive singular value actually belongs to an eigenvector measuring bipartiteness. To avoid this, MultiNet makes the diagonal equal to (symmetrized) node degrees, so that the trace is positive and “negative“ singular values are suppressed.

For two-mode data there is also an important difference in the way symmetrization is handled for the two spectra.

- Normal analysis forms a symmetric matrix by constructing a (virtual) bipartite graph which retains all details about the rows and columns; the rows and columns are in a single eigenspace and are given their own sets of coordinates as the two parts of the bipartite graph. This results in a compete set of positive and matching (virtual) negative eigenvalues: virtual since with sparse matrix methods the full matrix is not required.
- Correspondence Analysis uses SVD resulting in separate row and column spaces. There are some controversies about how (or even whether) the two spaces can be represented together or compared (Carroll, et.al. 1986, 1987).

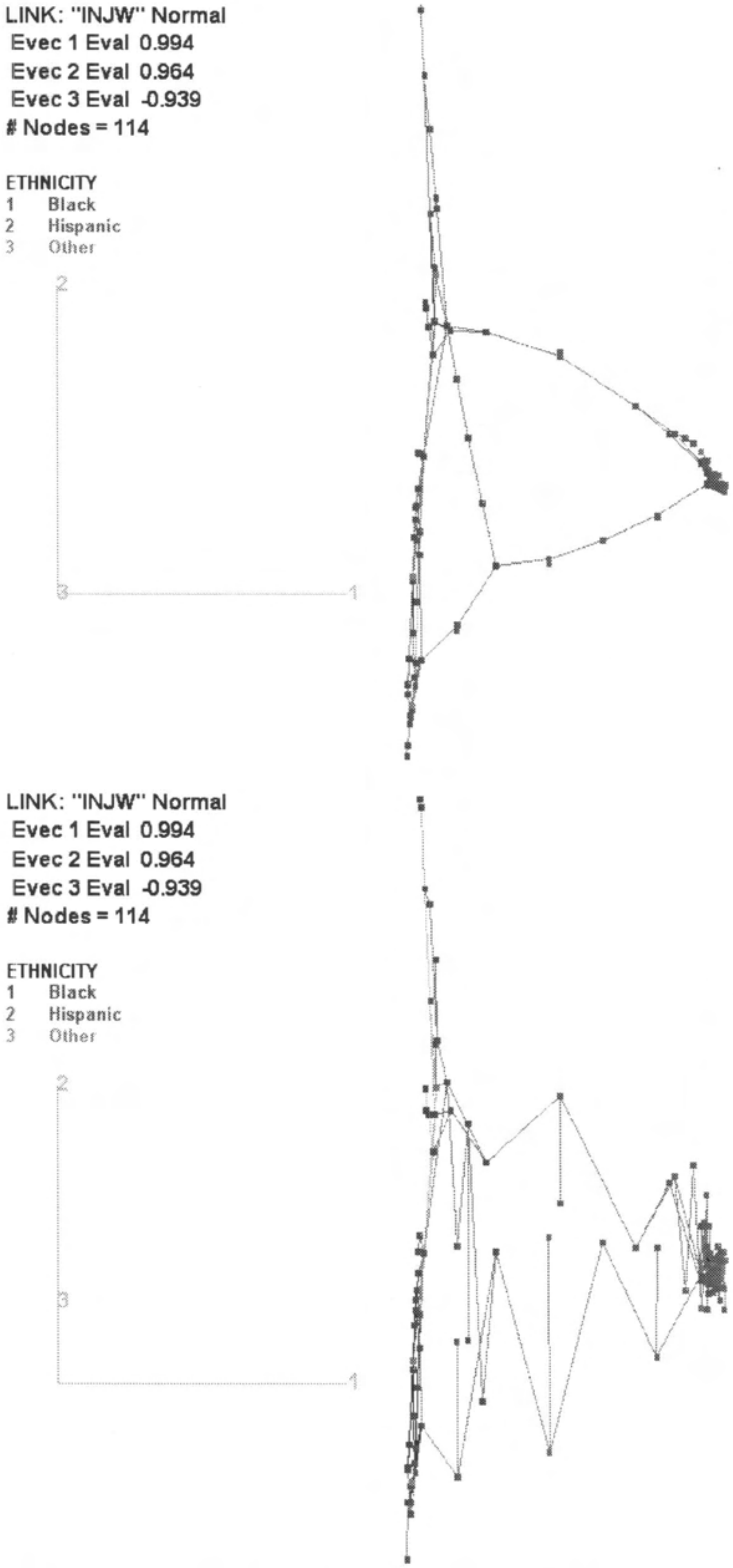
The following four examples demonstrate the usefulness of negative eigenvalues and the corresponding eigenvector.

9.2 Drug Injection behaviour

This data was collected from a northeastern US city by Scott Clair (Clair, 2000).There are a number of node and link variables, and in this example we look at the variables:

- ETHNICITY: which ethnic groups the actors belong to (node variable)
- INJW: who the actors report they have injected drugs with (link variable)

We have seen this network in the permuted 1-D displays of Section 5 (Figure 5.17) which showed a relationship between ETHNICITY and INJW. The 3-D displays of Figure 9.1 show that the first three eigenvalues are important, since all are close to 1 in absolute value. As shown in Figure 9.1a, the first eigenvector (X-axis) separates the two main ethnic groups (Blacks have many more links with Blacks, Hispanics have many more links with Hispanics). The second eigenvector (Y-axis) appears to detect two clusters within the Blacks, while the Hispanics appear tightly clustered. However, the third eigenvalue is negative, meaning that the third eigenvector has the "high-frequency" oscillations associated with bipartite graphs (Seary & Richards, 2000). Bipartite graphs have no triangles, so an important structural feature of this network is a lack of triangles. Fig 9.1b rotates the third eigenvector into the Y-axis and shows that this oscillation mostly happens at the right part of the display, which is mostly Hispanic. (This would be obvious without rotation in the anaglyphic 3-D display). What appeared to be a cluster is actually a "co-cluster": a bipartite-like pair of off-diagonal blocks.



a) The first eigenvector separates the two main ethnic groups.

The second further separates the group at the left.

The group at the right appears tightly clustered.

b) However, the third eigenvalue is negative.

Rotating the third eigenvector into the Y-axis shows that the third eigenvector is oscillatory for the group at the right, showing a lack of triangles. .

Figure 9.1. Link variable INJW and node variable ETHNICITY INJW links are pairs of people who report injecting drugs together. ETHNICITY is the self-reported ethnic group.

There are a number of reasons why this could be the case. Since this network results from people reporting that other people engaged in an illegal activity with them, this may just be the result of "missing" links since the people may be reluctant to report such activity. The existence of just two links between the two ethnic groups (Figure 1) suggests that people may indeed censor such links. However, this explanation does not work with missing triangles, since it requires too much collusion. In social activities, we expect transitivity: "if a knows b, and a knows c, then b knows c", and this forms triangles. With the Blacks we see these triangles, but not with Hispanics.

Scott Clair's explanation is based on ethnography. Blacks have a longer history in this city, have more money, inject drugs together as a social activity in each others' apartments. Hispanics are newer to the city, cannot afford apartments, and tend to inject together in alleys: this is not a social activity, it is one of convenience and contingency. **Model→Pstar** triad counts show 70 transitive and 24 cyclic triads for Blacks, and only 24 and 3 corresponding triads for Hispanics.

This example shows how the Normal eigenvector display not only makes clear the relation between network structure and actor attributes, it also raises questions about the quality and reliability of the data and suggests how these questions could be answered.

9.3 Physician advice networks

In this example we look at two physician advice networks, along with two node attributes: SPECIALTY (physician's specialty) and ZIPCODE (denotes geographical location of physician's office). The datasets were collected for a pharmaceutical company in order to find out who comes to doctors for advice, and who doctors go to for advice about medication for a particular medical problem. A non-disclosure agreement allows use of the data, but not the company name. The link variable COMESTO was collected by asking a sample of specialists to list up to 6 doctors who came to them for advice. The link variable GOESTO was collected by asking a different sample of doctors to list up to 6 doctors who they have gone to for advice. The relation of these two networks to the two node attributes is strikingly different.

In figure 9.2a we see COMESTO with node variable SPECIALTY. **Show→Direction** has been selected to show that the network is very directed.. Only 8 of the 412 physicians both give and receive advice. The directed nature of this network is captured by the first eigenvector (with negative eigenvalue). Those that give advice (and that answered the survey) are on the right, and the advice-seekers (those named in the survey) are on the right. Clustering is visible along the Y-axis, and is made clearer by rotation around the Y-axis to show eigenvectors 2 and 3 (both with positive eigenvalues). In figure 9.2b, we see that the clusters are related to groups of specialties, which leads to the hypothesis that specialists mostly report giving advice to other physicians within the same specialties. There is no obvious relationship with ZIPCODE here.

In figure 9.3a, we see GOESTO with node variable ZIPCODE. **Show→Direction** has been selected to show that the network is very directed.. Only 18 of the 375 physicians both give and receive advice. The directed nature of this network is captured by the first eigenvector (with negative eigenvalue). Those that give advice (those named in the survey) are on the right, and the advice-seekers (those who answered the survey) are on the left. Clustering is visible along the Y-axis, and is made clear by rotation around the Y-axis to show eigenvectors 2 and 3 (both with positive eigenvalues). In figure 9.3b, we see that the clusters are related to ZIPCODES, and there is no obvious relationship with SPECIALTY. This leads to the hypothesis that physicians report going to the geographically closest specialist, rather than to a physician with the same specialty. Another hypothesis is that these zip codes are related to concentrations of specialists, such as in a medical building or medical research facility, and suggests further data collection.

Both figures 9.2 and 9.3 show directed links as dashed at the sender end, dotted at the receiver end, and solid if reciprocated. This avoids the clutter of arrow-heads at nodes, and immediately identifies reciprocated links, and the direction of directed links from either end.

In Figures 9.4a and 9.4b we see the partitions that result from the first three Normal eigenvectors of each network. In both cases the off-diagonal nature of the directed links is captured very well by the first (negative) eigenvector.

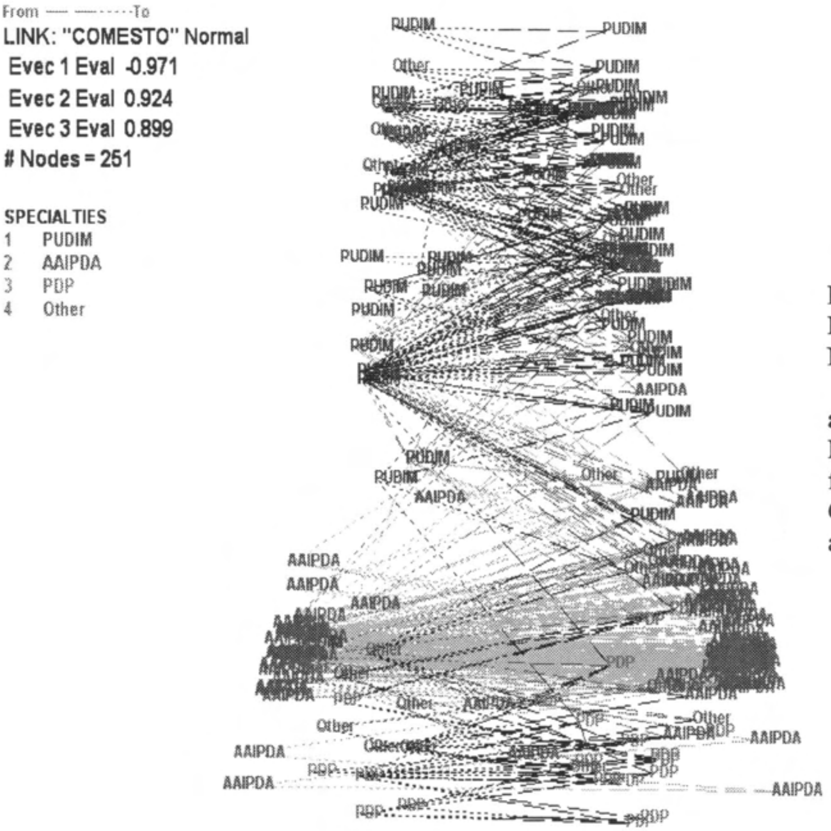
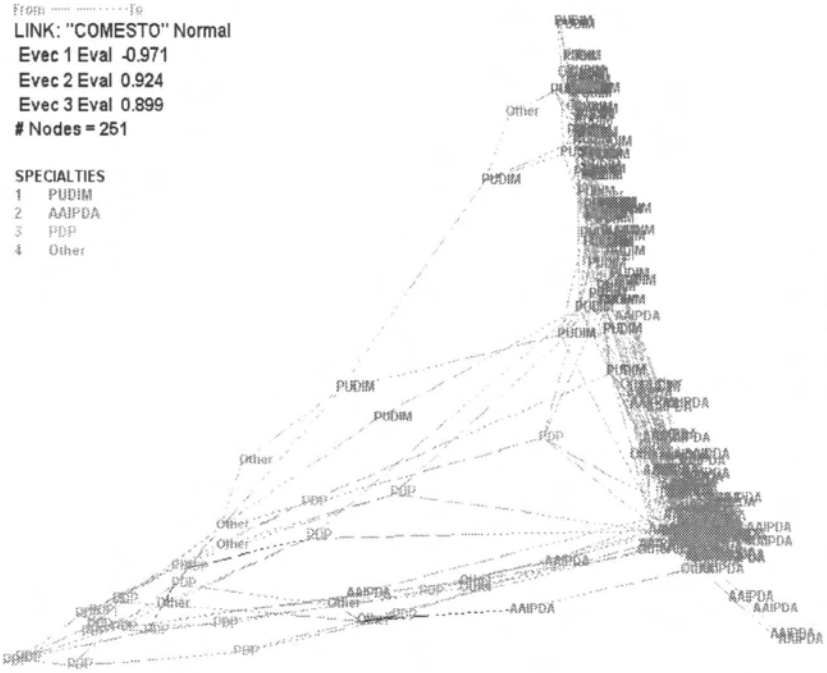


Figure 9.2.
Link variable COMESTO
Node variable SPECIALTY

a) First 2 eigenvectors. Direction is captured by first eigenvector. Clustering visible along Y-axis



b) 2nd and 3rd eigenvectors. Clusters at top and bottom right and (loosely) at left are related to SPECIALTY

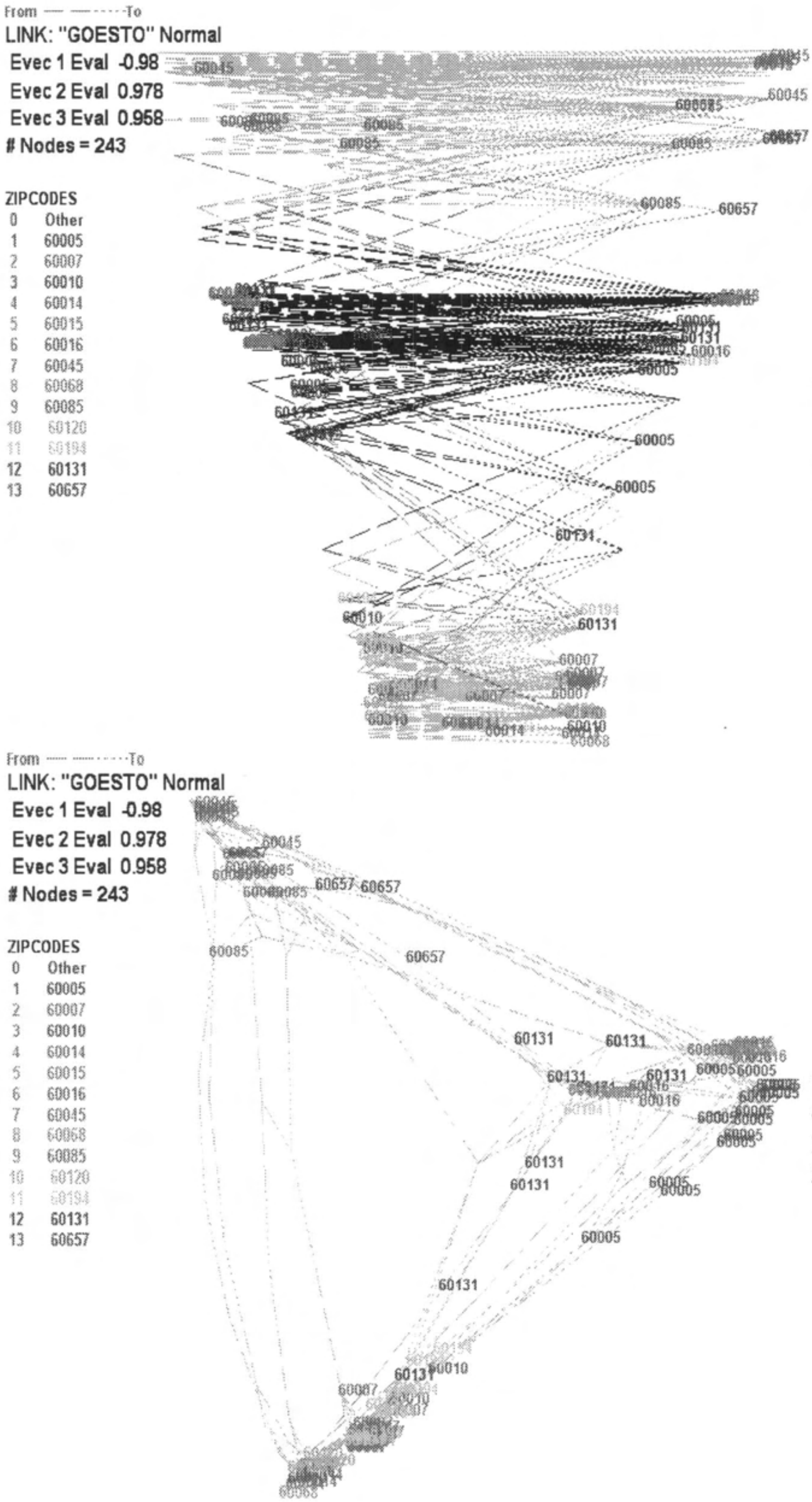
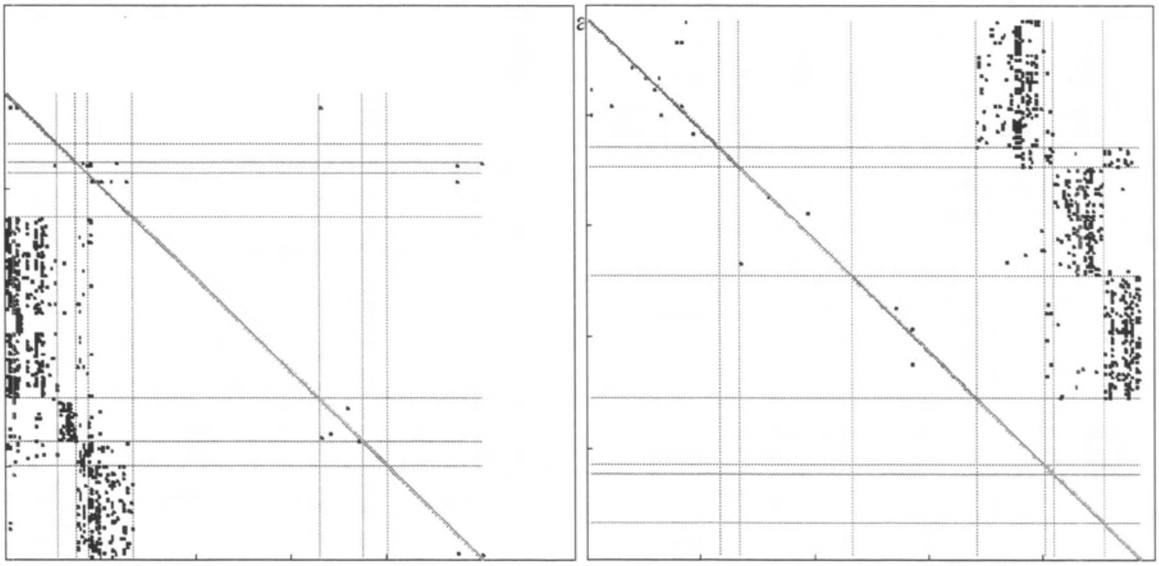


Figure 9.3.
Link variable GOESTO
Node variable ZIPCODES.

a) First 2 eigenvectors. Direction is captured by first eigenvector. Clustering visible along Y-axis.

b) 2nd and 3rd eigenvectors Clusters at top and bottom left and right are related to ZIPCODES.



a) COMESTO b) GOESTO

Figure 9.4. Adjacency matrices permuted by partitions from first three eigenvectors. In each case the first eigenvalue is negative and captures the directed nature of these networks, while next two are positive and capture the clustering behaviour.

9.4 Multi-mode networks I

Two-mode networks can be represented by bipartite graphs. An example of a two-mode network is a set of people (one mode) and the events they attend (the other mode). The relationship is "attends", so there are no links between people and people, or between events or events. This makes the network bipartite. If we represent this network as an adjacency matrix, all the links are in one or the other off-diagonal corners, and the matrix is mostly 0's. However with sparse methods this does not matter, since we only need store the links in one direction. The eigendecomposition routine automatically symmetrizes, which results in eigenvalues being returned as positive-negative pairs, as they should be for a bipartite network.

The same representation also works for three-mode networks, which have three types of objects and one relationship which is not meaningful within object types, only between types. An example is people and the types of symptoms and exposures they report. This data came from a University of Toronto study on Multiple Chemical Sensitivity conducted by Cornelia Baines and Gail McKeown-Eyssen (McKeown-Eyssen, et.al., 2001). Questionnaires were filled out by two different groups of people, in which they listed all medical symptoms they reported in the last year, and also listed all the materials that they thought may have caused the symptoms (the exposures). The group we will examine were patients of ordinary GPs, while the other group went to specialists in Environmental Health. Since people reported up to 63 symptoms and 61 exposures, it is impossible to relate individual symptoms to exposures. However, it is possible to find a relationship between types of symptoms and types of exposures which could not be found by factor analysis, and which the original researchers found surprising.

To define the network we define three types of nodes: 1340 people, 68 symptoms and 85 exposures (so the network has a total of 1493 nodes). A link is defined between a person and a symptom if the person reported that symptom, or between a person and an exposure if the person reported that exposure. The resulting link variable is called "sym-ex-pe". All links are binary. There are a total of 32,330 links out of a possible 205,020 giving a density is 0.16, which is rather high. (For the full adjacency matrix the density is 0.014, but this ignores the fact that most entries are not meaningful). Because of this Figures 9.5 and 9.6 have **Show→No lines** selected, and only the nodes are displayed (as dots or as attribute values).

Figure 9.5 shows eigenvectors 1, 5 and 3. The black dots correspond to people, the coloured dots to symptoms or exposures. Since the first eigenvector is -1 the two parts of the bipartite network would each lie along straight lines in the direction of the Y-axis, so the display is rotated slightly for clarity. The order of eigenvectors is 1,5,3 since the even eigenvalues 2 and 4 are the negative copies of 3 and 5, and since eigenvector 3 captures the frequency of symptoms or exposures. Eigenvector 5 on the other hand captures the coupling between symptoms and exposures, which is what we want.

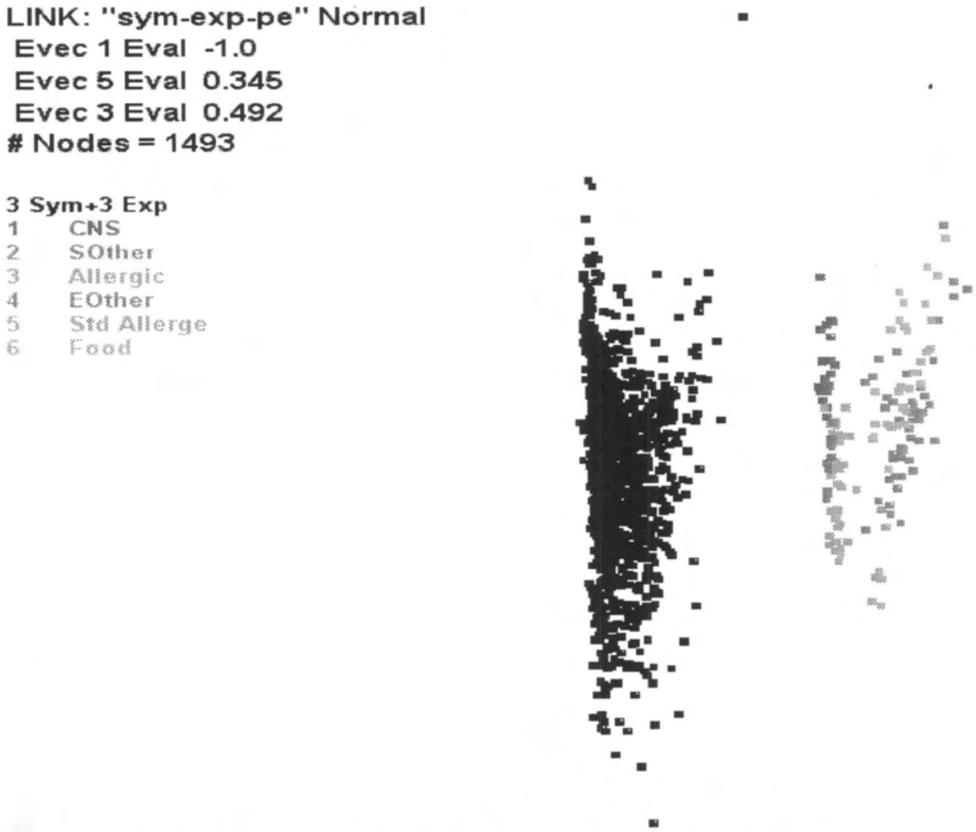


Figure 9.5. People (black), symptoms (1-3) and exposures (4-6) for GP dataset.

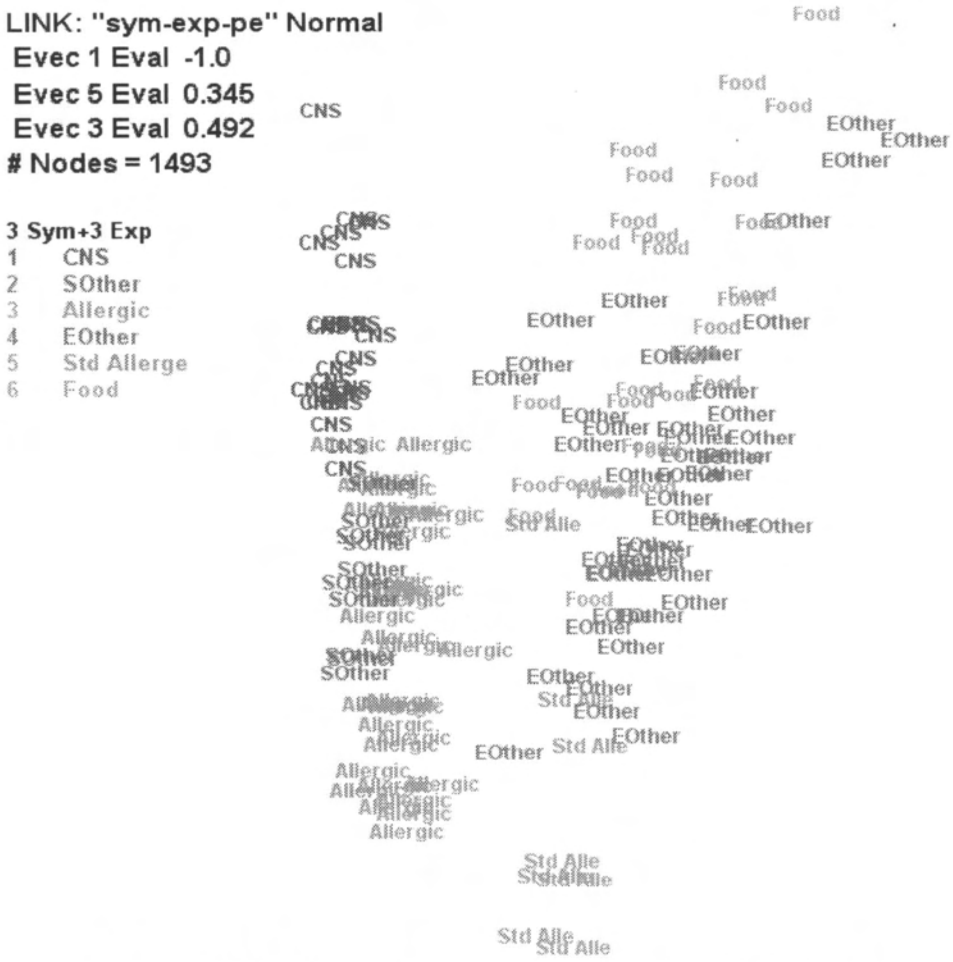


Figure 9.6. Dots→Values for node variable 3 Sym+3 Exp, showing coupling between symptom and exposure types with eigenvector 5 along Y-axis.

Figure 9.6 shows the same eigenvectors, but with **Dots→Values** selected. The node variable is 3 Sym+3 Exp which does not have values for people, so they are not shown. The categories refer to types of symptoms and types of exposures. For example, category 1 is CNS type symptoms such as "difficulty concentrating" and "depressed"; category 5 is Standard allergens such as "pollen" and "house dust". The categories and their membership were set up by the investigating physicians. The coupling between types of symptoms and types of exposures is clear, and in one case not surprising: Allergic (3) symptoms and Standard Allergen (5) exposures are mostly at the negative end of eigenvector 5, so that people who report one also tend to report the other (e.g., "pollen" is reported with "runny nose"). The other end of this axis does have a surprising association: CNS (1) symptoms and Food (6) exposures (e.g., "chocolate" is reported with "depressed"). This association, which is obvious from this visualization, had not been seen before but was confirmed by using **Define→Variable** to create node variables from the 1st and 5th eigenvectors which were then analysed with the more advanced statistical methods available in other MultiNet modules. See section 10 for a complete discussion.

9.5 Multi-mode networks II

Section 8.12.2 describes how important Normal eigenvectors point in a direction that maximizes chi-squared, by approximately diagonalizing the links of the adjacency matrix. This means that patterns of connections are similar for nearby nodes, and very different for nodes with the most different eigenvector coordinates. In fact, nodes with exactly the same connections will have exactly the same coordinates (this is the CA property of distributional equivalence discussed in Greenacre, 1984). Ordinarily, this property is considered a nuisance for visualizations since a number of nodes may be overlayed in exactly the same location. MultiNet provides **Show→Repel** which displays these nodes in a circle around the common centre (Figure 9.10). However, this property can also be very useful.

The Canadian subset of data from the Familial Colon Cancer Registry (FCCR) consists of 112,000 individuals in 4100 families and over 1,000 variables including 500+ epidemiological and lifestyle variables and 500+ diet and nutrient variables (Cotterchio, et.al., 2000). MultiNet is being used to examine which genes, and/or which environmental factors, account for familial clustering as well as the relative importance to familial clustering of genetic or environmental factors. Multi-mode analysis and visualisation techniques allow identification of patterns of connections among people, cancer sites and families. Questions that are being addressed:

- Which sites of cancer co-occur within families?
- Do the same sites co-occur with colonic and rectal cancers within families?
- Are there combinations of sites linked repeatedly within different families?

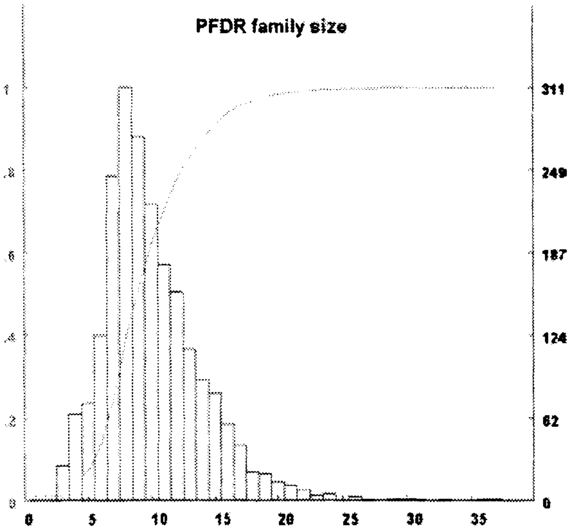


Figure 9.7. Distribution of family sizes

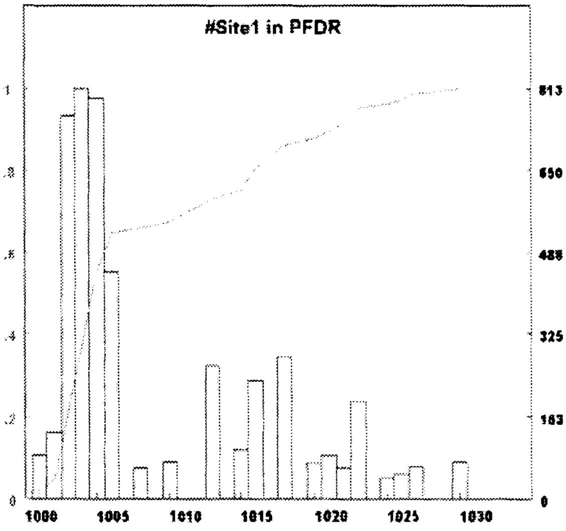


Figure 9.7. Distribution of cancer sites

Figure 9.7 shows the distribution of family sizes for the 21,813 family members of the 2158 probands selected from Ontario ColoRectal cancer clinics who volunteered to provide information about themselves and their first degree relatives. Figure 9.8 shows the distribution of the 20 most common types of cancer (over 90% of the cases) for these people; there is an obvious "spike" in the ColoRectal cancers (1003-1006) because of the nature of the data. See Figures 9.10-9.13 for details on the coding used for cancer sites.

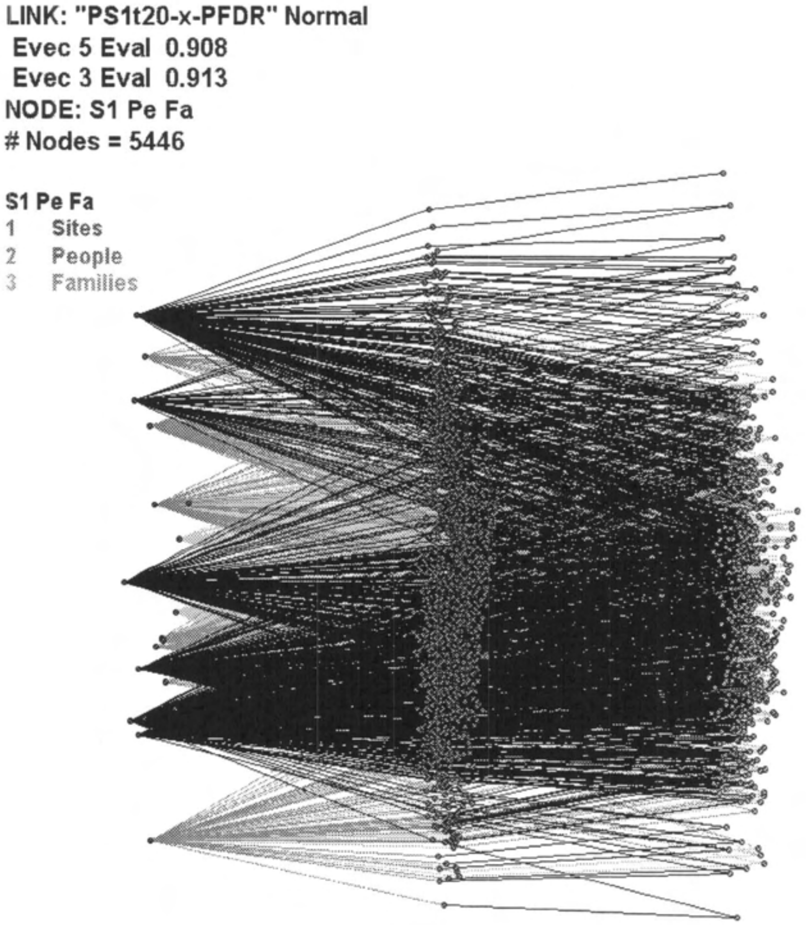


Figure 9.9. Cancer sites (blue) linked to people (red) linked to families (green), rotated to show the three modes, arranged along eigenvector 3 (vertical axis).

Recode→Make Hypergraph was used to make the set of connections going from cancer sites to people to families as shown in Figure 9.9. This figure uses **Z-axis→Node** with a node variable that distinguishes sites, people and families. This variable is also used to colour the nodes.

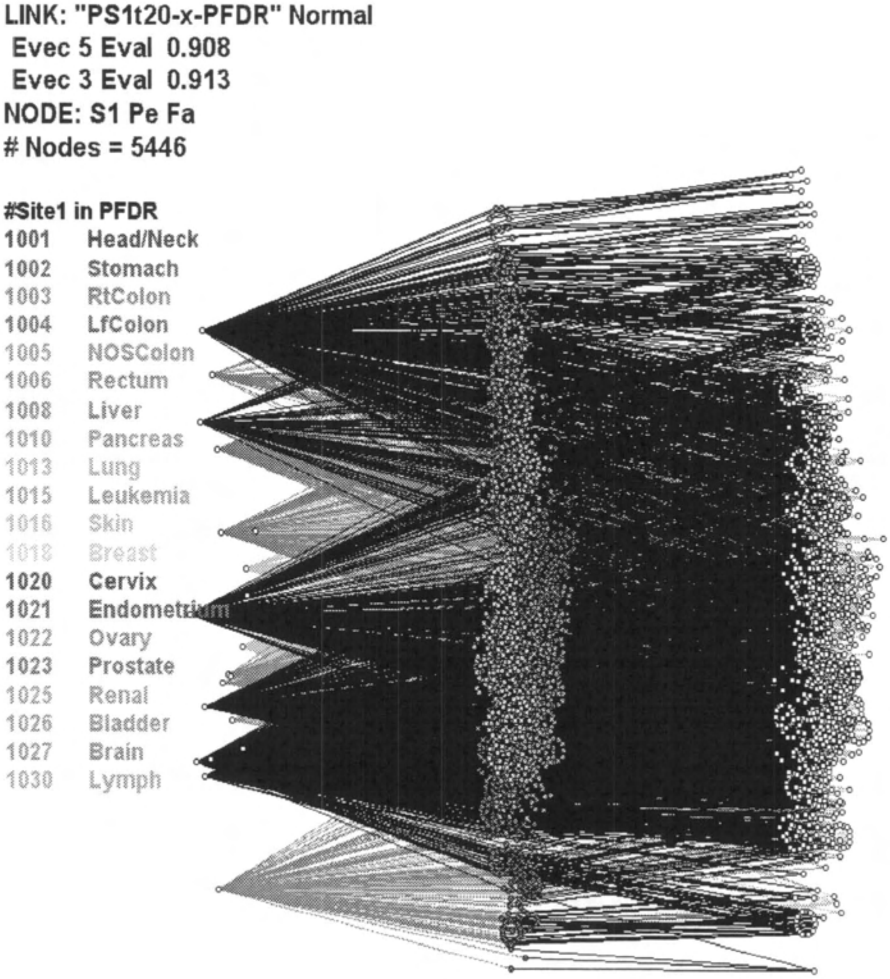


Figure 9.10. People coloured by cancer type, with **Show→Repel** on.

Families with the same sets of cancers are arranged in circles.

Figure 9.10 shows nodes coloured by cancer types (since this an attribute of people, only people are coloured). There is an obvious clustering of cancer types that changes from top (Rectum) to bottom (Stomach) of the figure (eigenvector 3). In addition, **Show→Repel** has been turned on, so that families (white circles on the right) with the same sets of connections are arranged in circles around the common centre. One goal has been satisfied: it is easy to see the co-occurrences of types of cancers in families.

The co-occurrences become even clearer when the figure is rotated into eigenvectors 3 and 5 (Figure 9.11). We know that the patterns are most different at the top and bottom of eigenvector 3, which suggests putting the coordinates of the families into deciles, and then examining the first and last decile. This is an example of analytic visualization. The results are shown in figures 9.12 and 9.13. In these figures, the number of people with a particular cancer in a family is shown by colouring the links. This uses a feature of MultiNet developed specially for this data, but not yet in general release. Figure 9.12 shows the sets of cancers which co-occur with Stomach in the first decile of families. It is clear that Left Colon is important, and that Rectum does not appear (nor do Brain, Skin, Prostate, Bladder) Figure 9.13 shows the last decile of families. Here we see Rectum at the centre, along with Brain, Skin, Prostate, Bladder, but without Left or Right Colon, Lymph, Breast, Head/Neck, Leukemia.

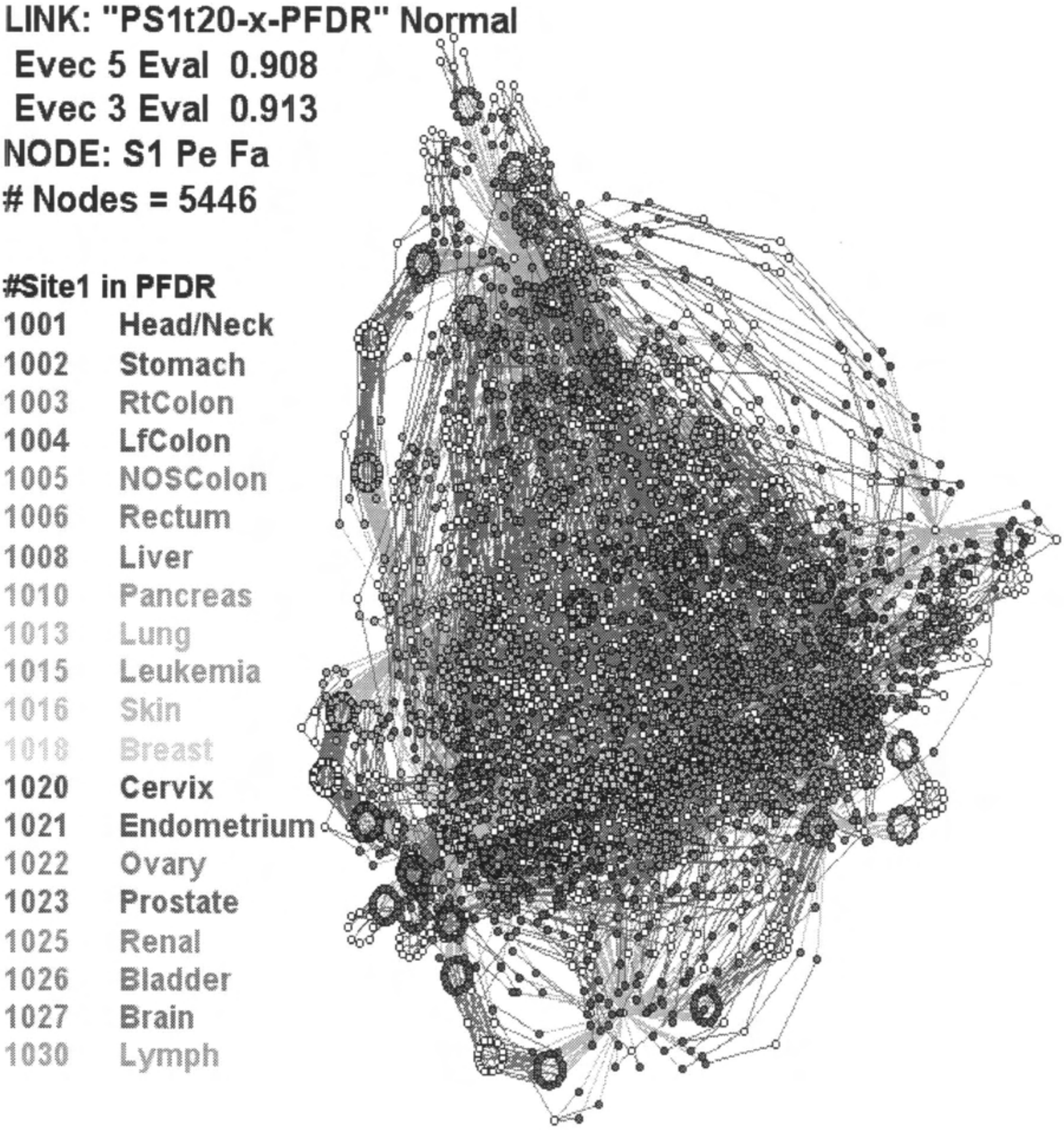
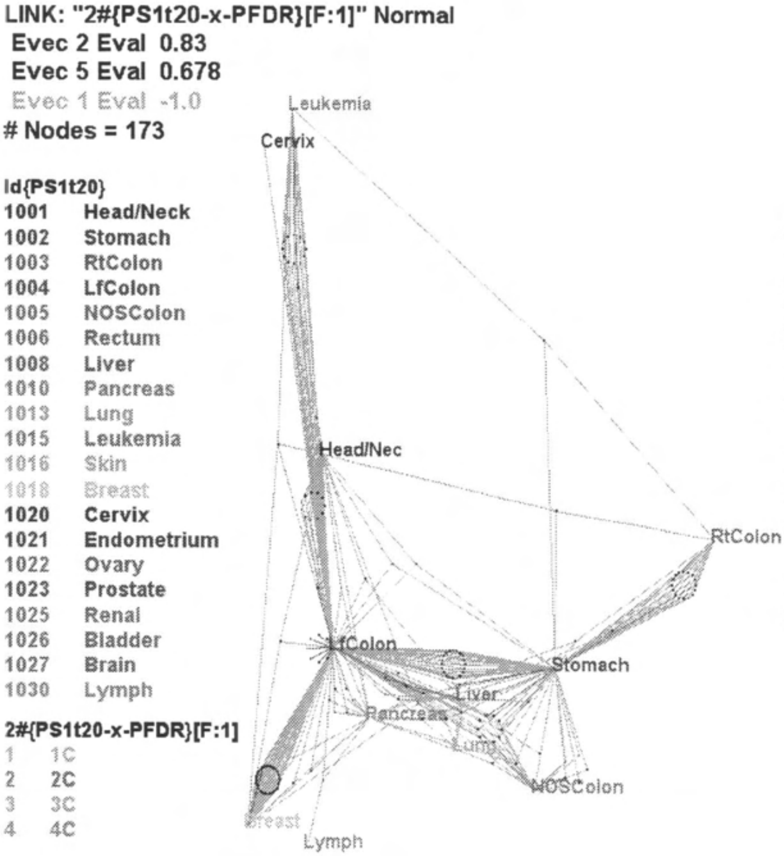


Figure 9.11. Familial cancer co-occurences are circles of open circles, connected to sets of cancer types.

Eigenvector 3 distinguishes Rectum (top) from Stomach (bottom).
Each is associated with different sets of cancer co-occurrences.



LINK: "2#{PS1t20-x-PFDR}[F:1]" Normal
Evec 2 Eval 0.83
Evec 5 Eval 0.678
Evec 1 Eval -1.0
Nodes = 173

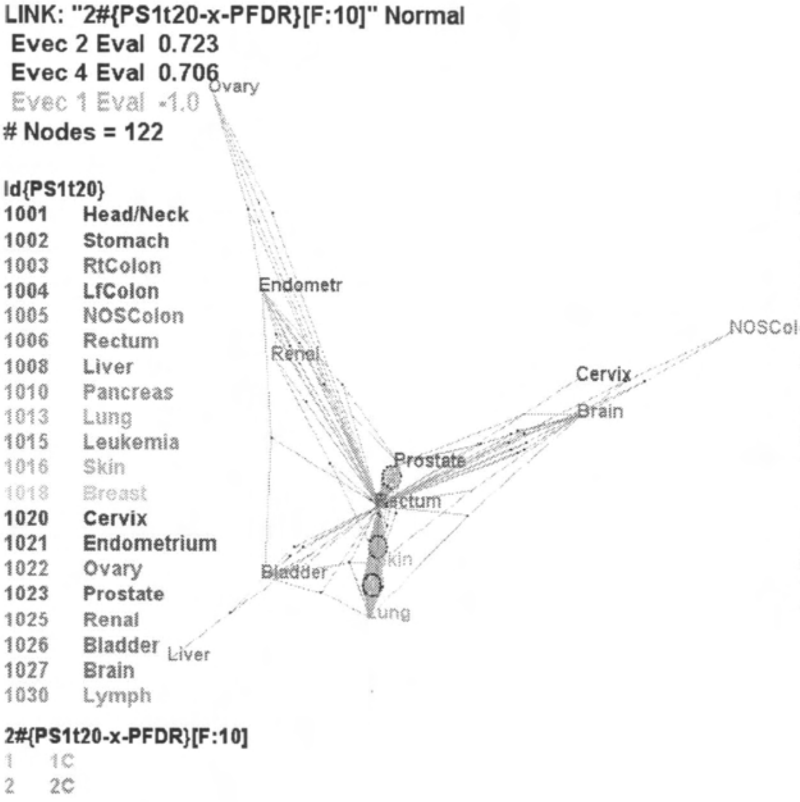
Id{PS1t20}
1001 Head/Neck
1002 Stomach
1003 RtColon
1004 LfColon
1005 NOSColon
1006 Rectum
1008 Liver
1010 Pancreas
1013 Lung
1015 Leukemia
1016 Skin
1018 Breast
1020 Cervix
1021 Endometrium
1022 Ovary
1023 Prostate
1025 Renal
1026 Bladder
1027 Brain
1030 Lymph

2#{PS1t20-x-PFDR}[F:1]
1 1C
2 2C
3 3C
4 4C

Figure 9.12. Familial clusters (circles of dots) of 150 families in the bottom decile.
Legend at bottom left shows that up to 4 cases of a cancer type may occur in a single family.

Head/Neck, Lymph, Breast, Pancreas, Leukemia co-occur mostly with Left Colon.

Rectum, Prostate, Brain, Bladder, Skin, Renal do not appear.



LINK: "2#{PS1t20-x-PFDR}[F:10]" Normal
Evec 2 Eval 0.723
Evec 4 Eval 0.706
Evec 1 Eval -1.0
Nodes = 122

Id{PS1t20}
1001 Head/Neck
1002 Stomach
1003 RtColon
1004 LfColon
1005 NOSColon
1006 Rectum
1008 Liver
1010 Pancreas
1013 Lung
1015 Leukemia
1016 Skin
1018 Breast
1020 Cervix
1021 Endometrium
1022 Ovary
1023 Prostate
1025 Renal
1026 Bladder
1027 Brain
1030 Lymph

2#{PS1t20-x-PFDR}[F:10]
1 1C
2 2C

Figure 9.13. Familial clusters (circles of dots) of 110 families in the top decile.
Legend at bottom left shows that up to 2 cases of a cancer type may occur in a single family

Prostate, Skin, Bladder, Brain, Lung co-occur with Rectum.

Stomach, Right and Left Colon, Lymph, Breast, Head/Neck, Pancreas, Leukemia
do not appear.

These results show two things:

- The Amsterdam criterion (Cotterchio, et.al., 2000) is a known risk factor for families with Colorectal cancer. Risk is high with co-occurrence of any of: Brain, Endometrium, Ovary, Liver, Lymph, Pancreas, Stomach. The first three co-occur more with Rectum, while the last three co-occur more with Colon. This is a new result.
- There is co-occurrence of Rectum with Skin, Bladder, Renal, Lung, which are not part of the Amsterdam criteria. This is also a new result.

These results, when confirmed by analysis of data from other centres, may be used in the search for new genes. Other questions that may be addressed are:

- What lifestyles co-occur within families?
- What lifestyles co-occur within individuals?
- Does lifestyle co-occur with health behaviour in individuals?

Papers discussing this new type of analysis and describing these new results are in preparation.

9.6 References

Carroll, J.D., Green, P.E., & Schaffer, C.M. (1986) Interpoint distance comparison in correspondence analysis, *J. Market Research* 23: 271-280

Carroll, J.D., Green, P.E., & Schaffer, C.M. (1987) Comparing interpoint distance comparison in correspondence analysis: a clarification, *J. Market Research* 24: 445-450

Clair, S. (2000) *Personal communication*

Cotterchio, M., McKeown-Eyssen, G., Sutherland, H., Buchan, G., Aronson, M.,Easson, A.M., Macey, J., Holowaty, E., & Gallinger, S. (2000) *Ontario Familial Colon Cancer Registry: Methods and first-year response rates*, Chronic Diseases in Cnaada **21** (2): 81-86

Greenacre, M., (1984). *Theory and Application of Correspondence Analysis.* Academic Press.

McKeown-Eyssen GE, Baines C, Marshall LM, Jazmaji V, Sokoloff E. (2001). Multiple Chemical Sensitivity: Discriminant Validity of Case. *Archives of Environmental Health*, **6**: 406-412.

Press, W., Flannery, B., Teukolsky, S., Vetterling, W. (1986). *Numerical Recipes*, New York: Cambridge University Press

Richards, W.D. & Seary, A.J. (2000). Eigen Analysis of Networks, *J. Social Structure.* http://www.heinz.cmu.edu/project/INSNA/joss/index1.html

Richards, W. D. & Seary, A.J. (1997) .Convergence analysis of communication networks, in *Organization Communication: Emerging perspectives V*, Ed. G. Barnett, Ablex.

Seary, A.J. & Richards, W.D. (2000). Negative eigenvectors, long paths and p*, Paper presented at INSNA XX, Vancouver BC. http://www.sfu.ca/~richards/Pages/longpaths.pdf

Seary, A.J. and Richards W.D. (1999). Eigenvalue Bounds, presented to International Network for Social Network Analysis, Charleston, S.C. February, 18-21 http://www.sfu.ca/~richards/Pages/insna99x.pdf.

Seary, A.J. and Richards, W.D. (1998). Some Spectral Facts. Presented at INSNA XIX, Charleston. http://www.sfu.ca/~richards/Pages/specfact.pdf

Seary, A.J. & Richards, W.D. (1995). Partitioning Networks by Eigenvectors. Presented to European Network Conference, London. Published in Everett, M.G. and Rennolls, K. (eds). (1996). *Proceedings of the International Conference on Social Networks,* Volume 1: Methodology. 47-58.

10. Networks of Symptoms and Exposures[1]

Andrew J. Seary(1), William D. Richards(1), Gail McKeown-Eyssen (2) and Cornelia Baines(3)[2]

Abstract

We present some novel methods for analysing and visualizing data from medical studies using methods originally developed for the study of social networks. The methods are based on spectral (eigendecomposition) properties of networks, in particular the so-called Normal spectrum. Among the many desirable properties of this spectrum is the natural handling of bipartite (2-mode) networks through negative eigenvalues, the clustering properties related to positive eigenvalues, and the relationship to the chi-squared measure of dependence in contingency tables.

Introduction

Social network analysis (SNA) begins with data that describes the set of relationships among the members of a system. One goal of analysis is to obtain from the low-level relational data a higher-level description of the structure of the system which identifies various kinds of patterns in the set of relationships. For example, it may be of interest to find cohesive clusters of network members: those which have most of their connections with each other. It may also be of interest to find members with similar roles: those with few mutual connections but many connections to other similar sets of members. These two goals may be combined in the search for general patterns, which is the aim of block-modelling (Wasserman and Faust, 1994).

As an illustration of block-modelling, consider a network or graph G(V,E) as a set of nodes V (points, vertices) connected by a set of links E (lines, edges). For simplicity here, we will consider networks that are binary (edges have logical value 1 if a relationship/connection between the nodes exists, 0 if not), symmetric (an edge from node *I* to *j* implies an edge from node *j* to *I*), and without self-loops (no edges between *I* and *I*). We may represent such a network as the (square) adjacency matrix **A** = **A**(G) with:

$$\mathbf{A}(i,j) = 1 \text{ if } I \text{ is connected to } j$$
$$\mathbf{A}(i,j) = 0 \text{ otherwise}$$

For example:

[1] This section has been accepted for publication in *Structure and Dynamics* as (Seary, et.al, 2005).

[2] (1) School of Communication, Simon Fraser University, B.C.; (2) Department of Public Health Sciences and Department of Nutritional Sciences, University of Toronto; (3) Department of Public Health Sciences, University of Toronto.

Adjacency matrix block-model

$$\begin{array}{c cccc|cc|cc}
 & a & b & c & d & e & f & g & h \\
a & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
b & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
c & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\
d & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ \hline
e & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
f & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ \hline
g & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
h & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0
\end{array}
\quad \rightarrow \quad
\begin{array}{ccc}
1 & 0 & 0 \\
0 & 0 & 1 \\
0 & 1 & 0
\end{array}$$

where the partitions of the network on the left map onto the blocks on the right. It is easy to see where the partitions (and hence blocks) should go in this example, since the rows and columns are ordered to make this obvious. In general, network data is not so conveniently ordered, nor is it so obvious where the blocks are. In this example, we see:

- the network is not connected. There are no links from the block in the upper left (*a-d*) to those in the lower right (*e-h*).
- The upper left block is on the diagonal. It is a clique (complete graph), with a link between every pair of nodes.
- The lower right blocks are off-diagonal and form a (complete) bipartite graph, with links from *e* and *f* to *g* and *h*, but no links between *e* and *f* or *g* and *h*.

There are a number of methods for finding an ordering and a blocking of network data. One approach is to choose a set of axes in the multidimensional space occupied by the network and rotate them so that the first axis points in the direction of the greatest variability in the data; the second axis, orthogonal to the first, points in the direction of greatest remaining variability, and so on. This set of axes is a coordinate system that can be used to describe the relative positions of the set of points in the data. Most of the variability in the locations of points will be accounted for by the first few dimensions of this coordinate system. The coordinates of the points along each axis will be an eigenvector, and the length of the projection will be an eigenvalue. The set of all eigenvalues is the spectrum of the network. Spectral methods (eigendecomposition) have been a part of graph theory for over a century. SNA researchers have used spectral methods either implicitly or explicitly since the late 1960's, when computers became generally accessible in most universities. Two of the earliest important programs were related to eigendecomposition: Negopy (Richards, 1971; Richards & Rice, 1981) was designed for finding cohesive clusters, and CONCOR (Breiger *et al.*, 1975) aimed to solve the more general block-modeling problem. The eigenvalues of a network are intimately connected to important topological features such as maximum distance across the network (diameter), presence of cohesive clusters, long paths and bottlenecks, bipartite-ness, and how random the network is (Chung,1995). The associated eigenvectors can be used as a natural coordinate system for graph visualization; they also provide methods for discovering clusters and other local features. For a more complete discussion of these matters, see (Seary & Richards, 2003).

As well as networks of people and relationships, SNA has long considered relationships between people and events (Davis *et al.*, 1941), co-authorship networks (Crane, 1972) and other examples of so-called 2-mode networks (Wasserman & Faust, 1994) which involve relationships between two types of nodes. These networks are usually shown as rectangular **R** (with n_1 rows and n_2 columns), since in general there are not the same numbers of the two types of nodes. As Breiger (1974) shows, 2-mode matrices can be made square by matrix multiplication of **R** and its transpose $\mathbf{R}^T$. Another approach is to make a square **A** (with $n_1 + n_2$ rows and columns) from **R** by appending $\mathbf{R}^T$ below and to the left of **R** along with necessary **0** matrices:

$$\mathsf{R} = \begin{array}{c|ccccc} & 1 & 2 & 3 & \dots & n2 \\ \hline 1 & 1 & 1 & 0 & \dots & 1 \\ 2 & 0 & 1 & 1 & \dots & 1 \\ : & : & : & : & \dots & : \\ n1 & 1 & 0 & 1 & \dots & 0 \end{array} \qquad \mathsf{A} = \begin{array}{|c|c|} \hline \mathbf{0} & \mathsf{R} \\ \hline \mathsf{R}^T & \mathbf{0} \\ \hline \end{array} = \begin{array}{c|cccc|ccccc} & 1 & 2 & \dots & n_1 & 1 & 2 & 3 & \dots & n_2 \\ \hline 1 & & & & & 1 & 1 & 0 & \dots & 1 \\ 2 & & 0 & & & 0 & 1 & 1 & \dots & 1 \\ : & & & & & : & : & : & \dots & : \\ n_1 & & & & & 1 & 0 & 1 & \dots & 0 \\ \hline 1 & 1 & 0 & \dots & 1 & & & & & \\ 2 & 1 & 1 & \dots & 0 & & & & & \\ 3 & 0 & 1 & \dots & 1 & & & 0 & & \\ : & : & : & \dots & : & & & & & \\ n_2 & 1 & 1 & \dots & 0 & & & & & \end{array}$$

This shows that 2-mode networks can be represented by the square adjacency matrices of symmetric bipartite graphs. (We will use this method for the data we describe later in this paper.) This representation is not generally used in SNA, probably because of the extra space taken up by the transpose and the **0** matrices. However, sparse matrix methods, which only store and manipulate actual links (Seary, 2005), can allow rectangular **R** to be treated as square **A** very efficiently.

The Normal Spectrum

The Normal Spectrum may be derived by considering the generalized quadratic placement problem (Hall, 1970; Seary & Richards, 1995) leading to the generalized eigenvalue equation:

$\mathbf{Lx} = \lambda\, \mathbf{Dx}$, where[3]

[3] We have introduced some notation which will be followed throughout:
- matrices are represented by bold capitals: **D**
- (column-)vectors are represented by bold lower case: **e**
- eigenvalues are represented by greek letters, usually with some relationship to the latin letters representing a matrix and an eigenvector. E.g.(v_i , $\mathbf{n}_i$) are the eigenpairs of Normal matrix **N**

- **D** is a diagonal matrix of node degrees of G
- **L** = **D** - **A** is the Laplacian matrix of G (Cvetkovic *et al.*, 1995, Seary & Richards, 2003)
- λ is an eigenvalue
- **x** is a corresponding (column-)eigenvector

Assuming that **D** can be inverted (which it can be if every node has at least one link; i.e. no nodes are isolated)

$$\mathbf{D}^{-1}\,\mathbf{LX} = \mathbf{D}^{-1}\,(\mathbf{D} - \mathbf{A})\mathbf{X} = (\mathbf{I} - \mathbf{D}^{-1}\,\mathbf{A})\mathbf{X} = \mu\,\mathbf{X}$$

where **A** is the adjacency matrix of G and **I** is an identity matrix of proper size. In fact, we usually take the defining equation to be

$\mathbf{D}^{-1}\,\mathbf{An} = \mathbf{Nn} = \nu\mathbf{n}$ with $\mathbf{D}^{-1}\,\mathbf{A} = \mathbf{N}$ and $\nu = 1-\lambda$, where

- ν (the Greek letter nu) is an eigenvalue of the Normal matrix **N** and
- **n** is the corresponding eigenvector.

Adding an identity matrix shifts the eigenvalues by 1 without changing the eigenvectors. Note that for networks without isolated nodes **D** has an inverse and therefore an inverse square root $\mathbf{D}^{-1/2}$. In networks with isolated nodes, the network size is effectively reduced by the number of isolates because the analysis uses only the nodes with links. The number of eigenpairs $(\nu_i\,, \mathbf{n}_i)$ is equal to the number of nodes n. We generally label these with i=0,...,–1 since i=0 corresponds to the trivial eigenpair $(\nu_0 = 1, \mathbf{n}_0 = \mathbf{1})$.

The Normal matrix **N**(G) is:

$\mathbf{N}(i,j) = 1/\deg(I)$ if *I* is connected to *j*

$\mathbf{N}(i,j) = 0$ otherwise

so that **N** is not symmetric. However, we can construct $\mathbf{M} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$, which is symmetric, and which is similar to **N** (it has the same eigenvalues).

Let $(\nu_i\;, \mathbf{m}_i)$ be the eigenpairs of **M**. Then the eigenpairs **N** are:

$$(\nu_i\;, \mathbf{n}_i) = (\nu_i\;, \mathbf{D}^{-1/2}\mathbf{m}_i)$$

The orthonormalization condition is:

$$\mathbf{n}_I\mathbf{D}\mathbf{n}_j = \delta_{ij} = 1 \text{ if i=j, 0 otherwise}$$

- a boldface **1** refers to the vector (1,1, ... ,1)

That is, the vectors are orthonormal in the **D** (or χ^2) metric (Richards & Seary, 1997). The Normal spectrum is referred to as the Q-spectrum in (Cvetkovic, *et. al.*, 1995). The multiplicity of 1 as an eigenvalue is equal to the number of connected components in G. If G is bipartite, then eigenvalues appear as pairs with opposite signs. Thus -1 is an eigenvalue if and only if G is bipartite.

The Normal matrix **N** has a number of interesting properties:

- It has a trivial constant eigenvector $\mathbf{n}_0 = \mathbf{1}$ with eigenvalue $v_0 = 1$
- The spectrum of **N** is bounded by $1 = v_0 \geq v_1 \ldots \geq v_{n-1} \geq -1$
- The rows of **N** sum to 1 (it is a stochastic matrix)
- The spectrum of **N** contains a 1 for every connected component
- The eigenvalue -1 occurs if and only if G is bipartite, in which case all eigenvalues occur in pairs with opposite signs.
- N has been rediscovered a number of times: generalized or combinatorial Laplacian (Dodziuk & Kendall, 1985; Chung,1995); Q-spectrum (Cvetkovic, *et al.*,1995).

Notice also that the similar matrix **M** satisfies the definition of Chi-squared. In practice, it is much simpler to solve the eigenproblem for **M**, since it is symmetric.

Four important properties of Normal eigenpairs

The following important properties of Normal eigenpairs will be useful in understanding the results obtained later.

1. Bipartite Representation of 2-mode networks

We can represent a 2-mode network by a square symmetric matrix with all the links in off-diagonal corners, so that the matrix is mostly 0's. The result is always a bipartite graph, so that all eigenvalues occur as positive and negative pairs (eg. 1, -1, 0.93, -0.93, ...). We don't generally need most of the negative eigenpairs, but the eigenvector belonging to eigenvalue -1 can be very useful. We don't need to explicitly construct the full matrix, nor calculate all eigenpairs. Using sparse methods and automatic symmetrization, we only need store the links in one direction, and can calculate only a few eigenpairs with largest eigenvalues (Seary, 2005).

Assume that there are n_1 items in one mode (the rows of the original matrix) and n_2 items in the other mode (the columns). Then the bipartite matrix will be square with (n_1+n_2) rows and columns. Thus each eigenvector also has (n_1+n_2) coordinates. By the bipartite construction, the first n_1 coordinates correspond to the n_1 items in the first mode (the rows), and the remaining n_2 coordinates correspond to the n_2 items in the second mode (the columns).

For a pair of positive and negative eigenvalues of a normal spectrum, the only difference between the corresponding eigenvectors is that the first n_1 coordinates of one have opposite signs of the first n_1 coordinates of the other. In particular, the eigenvector belonging to eigenvalue -1 is the trivial constant eigenvector (**1**), except that the first n_1 coordinates are negative. The difference in signs can be used to identify the two modes.

2. Visualization

The eigenvectors of **N** can provide good visual representations of graphs which consist of blocks of nodes with similar connections. This follows from the relationship between the eigenvector coordinates for a node and those it is connected to (Seary & Richards, 1998). It is evident from the definition of eigendecomposition that

(1) $$n_i(s) = \frac{\sum_{s\sim t} n_i(t)}{(v_i \times \deg(s))}$$ for the i^{th} eigenpair $(v_i, \mathbf{n}_i)$ of **N**

(where $n_i(s)$ is the s^{th} component of the i^{th} eigenvector; "$s\sim t$" means "s is connected to t")

This equation shows that for eigenvalues v_i near 1, each node is approximately at the centroid of those it is connected to. The exact difference from the centroid for node u for eigenvector $\mathbf{n}_i$ is:

$$n_i(s) - \sum_{s\sim t} n_i(t) \Big/ \deg(s) = (1 - v_i) n_i$$

For "important" eigenvalues v_i near 1, this produces very good visualization properties. Members of a block tend to be close to one another and not close to members of other blocks.

3. Relation to χ^2

The χ^2 matrix is defined in terms of the row and column *marginals* (sums). A typical element is $(\text{Observed}_{ij} - \text{Expected}_{ij})^2 / \text{Expected}_{ij}$ where

$$\text{Expected}_{ij} = \frac{\deg(i)\deg(j)}{\Sigma \deg(i)}$$

We can write χ as: $O/\sqrt{E} - \sqrt{E}$, where the second term corresponds to the trivial eigenvector which can be dealt with separately. In matrix notation $\chi = \mathbf{D}^{-1/2}\mathbf{A}\,\mathbf{D}^{-1/2}$ which has eigenpairs $(v_i, \mathbf{D}^{1/2}\mathbf{n}_i)$. Thus we have[4]

$$\chi^2 = \sum_{j=1}^{n-1} v_i^2 \sum_{i,j=1}^{n} a_{ij}$$

(omitting the expected term corresponding to trivial $v_0 = 1$, $\mathbf{n}_0 = \mathbf{1}$)

[4] For the bipartite representation, only the positive eigenvalues contribute to χ^2 of the 2-mode network.

This equation shows how much each dimension contributes to χ^2 which is a measure of *dependence* between rows and columns (or of deviation from what would be expected if the node degrees by themselves would give a complete description of the network's structure). In this interpretation, if $|v_1|$ is small ($|v_1| << v_0 = 1$), then χ^2 is also small: there is no structure or pattern to explain in the network beyond the node degrees, and so there is no "signal" above the expected "background." On the other hand, if $|v_1|$ is close to 1, then χ^2 will be large and there is a relation between rows and columns of **A**, with the first eigenvector pointing in the direction of the maximum variability in χ^2. If $|v_{2,} v_3, ... v_k|$ are also large, we need k+1 eigenvectors to describe the patterns in the χ^2 matrix. Thus we can tell from the eigenvalues how many eigenvectors we need to explain most of the χ^2 of the network, and which are the most "important" ones, since they contribute most to χ^2 (Greenacre, 1984).

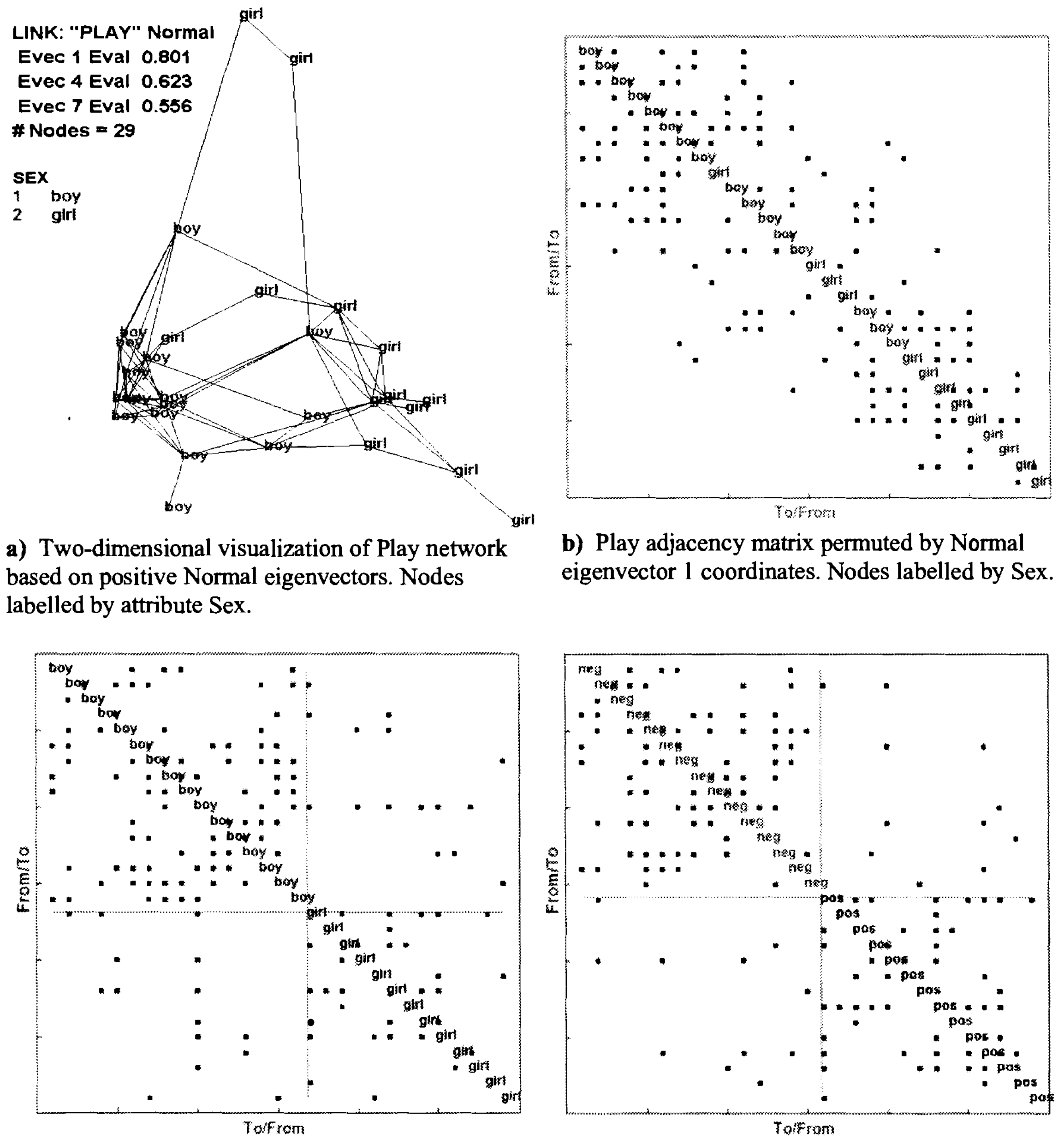
4. Partitions

There is a large literature on the use of eigenvector coordinates to provide partitions of graphs. Most of these methods use eigenpairs of the adjacency matrix (Powers, 1988) or the Laplacian (Pothen *et al.*, 1990). Fiedler (1975) was the first to show that Laplacian eigenpairs could provide good approximate solutions to the min-cut problem: partition a graph into parts of approximately equal number of nodes with few links between them. We can add an additional constraint that the *number of links* in each part also be roughly equal by weighting the node sets by their total degrees (Dhillon, 2001). This is exactly what a partition based on $\mathbf{n}_1$ from **N** gives us, since $\mathbf{n}_1$ points in the direction of maximum variability in χ^2. Similarly, further partitions based on $\mathbf{n}_2$, $\mathbf{n}_3$, ... will also produce sets of nodes with a large number of edges in common (as long as v_2, $v_{3,}$ ... make significant contributions to χ^2). Partitions based on positive eigenvalues will produce blocks of edges on the diagonal of **A**, while those based on negative eigenvalues produce nearly bipartite off-diagonal blocks (which occur in pairs if the network is symmetric) (Seary & Richards, 1995). In both cases, the concentration of links to specific parts of the network leads to a large value of χ^2 for the partition.

As an example of these properties, figure 1 shows visualizations of children at a day-care centre[5]. The network is defined by observing which children "Play" with each other (all links are therefore symmetric). Figure 1a is a two-dimensional visualization, labelled by the sex of the children. It is clear that the x-dimension (eigenvector 1) is important (eigenvalue = 0.801) and that the clusters on the left and right are related to sex. Fig. 1b is a one-dimensional visualization, showing the adjacency matrix as permuted by the coordinates on eigenvector 1. It is clear that this permutation based on the

[5] This data was collected by students in a course Richards taught in 1988. The students watched the children in a daycare centre (ages: 6 to 10) and, over the course of a day, noted the children they saw playing together and, later in the day, asked each who they had played with.

maximum variability in χ^2 has moved most of the links close to the diagonal. Fig. 1c shows the same adjacency matrix as permuted by the sex of the children (boys in upper left, girls in lower right). Some clustering is evident. Fig. 1d shows the adjacency matrix permuted by the *signs* of eigenvector 1 ("n" for negative, "p" for positive). The partition, which is now based on the network itself, is better than that for sex in a sense that will be described in the next section.



a) Two-dimensional visualization of Play network based on positive Normal eigenvectors. Nodes labelled by attribute Sex.

b) Play adjacency matrix permuted by Normal eigenvector 1 coordinates. Nodes labelled by Sex.

c) Play adjacency matrix permuted by Sex. Nodes labelled by Sex.

d) Play adjacency matrix permuted by signs of Normal eigenvector 1. Nodes labelled by signs.

Figure 1. Four views of the PLAY network.

Table 1a. Contingency tables for partitions of Play based on **a)** attribute Sex; **b)** Normal eigenvector 1

a) Crosstabulation of sex.
Chi-squared = 44.613

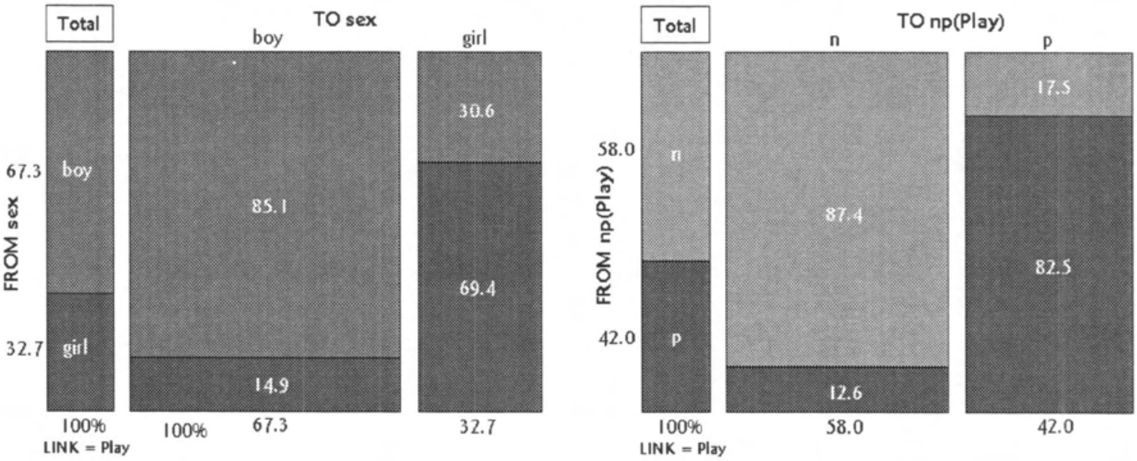
ROWS = FROM sex
COLS = TO sex

COUNT ROW % COL %	boy	girl	TOTAL
boy	86 85.15% 85.15%	15 14.85% 30.61%	101 67.33%
girl	15 30.61% 14.85%	34 69.39% 69.39%	49 32.67%
TOTAL	101 67.33%	49 32.67%	150

b) Crosstabulation of np(Play)
Chi-squared = 73.282

ROWS = FROM np(Play)
COLS = TO np(Play)

COUNT ROW % COL %	neg	pos	TOTAL
neg	76 87.36% 87.36%	11 12.64% 17.46%	87 58.0%
pos	11 17.46% 12.64%	52 82.54% 82.54%	63 42.0%
TOTAL	87 58.0%	63 42.0%	150



a) Partition of Play network based on Sex. $\chi^2 = 44.613$

b) Partition of Play network based on signs of Normal eigenvector 1. $\chi^2 = 73.282$

Figure 2. Panigrams for Play network with partitions based on: **a)** attribute Sex; **b)** Normal eigenvector 1

Contingency tables and panigrams

Once a partition has been found, it is a simple matter to form a contingency table by counting the number of links within and between each block. The quality of partitions can also be compared by calculating the χ^2 for each contingency table. These tables may be visualized by using *panigrams*[6]. A panigram contains most of the information in a contingency table. The percentages in the left column are the row marginals. The percentages under the columns are the column marginals. The

[6] Seary (1988) suggested the name "panigram" for the two-dimensional analog of histograms. "Histos" (ιστοσ) is Greek for the mast of a ship, whereas a 2-dimensional sail is "pani" (πανι) in Greek.

numbers in the cells are the column percents that would be seen in the corresponding cells of the contingency table. The heights of the segments in the "Totals" column are proportional to the row marginal percentages. The width of the other columns are proportional to the column marginal percentages. The areas of the segments are proportional to the counts in the corresponding cells in the contingency table. If the row variable is independent of the column variable, the segment heights in all columns are the same as the ones in the "Totals" column. This is not the case if the row variable is not independent of the column variable.

In table 1a and figure 2a (compare to figure 1c) we see the counts within and between a block-model based on sex. In table 1b and figure 2b (compare to figure 1d) we see the counts within and between a block-model based on the component signs of the first normal eigenvector. Clearly the latter is superior based both on a larger χ^2 and more within-block and fewer between-block counts.

The MultiNet Network analysis program

MultiNet is a Windows-based computer program designed for interactive exploratory data analysis of social and other large, sparse, multivariate networks[7]. It was designed for exploratory analysis and visualization of large, complex networks, and to provide details of the values of the link and node variables that make up the networks. Three aspects of the program are relevant to this discussion:

- **Eigenspaces**: Visualize networks and create variables and partitions from graph spectra.
- **Variables**: Univariate statistics and transform, combine, create and delete link or node variables. We will make use of the **Recode** function which allows a variable to be created by combining existing variables, then transformed into a categorical variable by quantiling, for use in a contingency table.
- **Analyse**: Perform statistical analyses on two or three link and/or node variables. We will create contingency tables visualized as Panigrams.

MultiNet always produces both graphical displays and textual reports; all the figures and tables in this paper were prepared using the program.

Figures 1d and 2b and table 1b use categorical partition variable np(Play) with two unique values ("n" and "p") based on the signs of Normal eigenvector 1 for the Play network. MultiNet makes it easy to define a real-valued variable based on actual eigenvector coordinates; this variable can then

[7] There is currently no limit (apart from memory) on the number of nodes and links that can be handled by the Analyse and Variables modules. There is similarly no limit on the number of node and link variables that may belong to a MultiNet data file, and new node and link variables can easily be created when desired.

be used to perform further operations on the eigenvector coordinates, such as selection of subsets of nodes and binning or quantiling into categories. Relationships between categorical variables can be easily examined with the resulting contingency tables visualized as panigrams. These definitions, transformations and analyses are all that will be used in this paper. Further and more detailed information on the program's capabilities can be found at (Seary, 2005).

A 3-mode medical network of people, symptoms and exposures

Bipartite representation can also be used for three-mode networks, which have three types of objects and one relationship which is meaningful only between but not within object types. An example is a) people, b) reported symptoms and c) exposures that were believed to produce the symptoms[8]. Using the method described above in the discussion of bipartite representation of 2-mode networks, we can represent the data as shown in Table 2.

Table 2. Bipartite representation of 2-mode networks

		1 2 .. n1	1 2 .. n2	1 2 .. n3
people	1 2 : n1	0	**reports of symptoms** *****	**reports of exposures** *****
symptoms	1 2 : n2	**symptoms reported** ******	**0**	**0**
exposures	1 2 : n3	**exposures reported** ******	**0**	**0**

** transposed data matrix * original data matrix

Questionnaires were filled out by patients in general practices. They listed symptoms they had experienced in the last year and any substances (exposures) that they thought might have caused symptoms. Respondents were not asked to link specific symptoms to specific exposures. The medical researchers initially categorized 68 selected symptoms into 14 types (Table 3a) and the 85 exposures into 8 types (Table 3b). The analysis described here did not link individual symptoms to individual exposures because of the large numbers of each reported by some patients (on average, 24.12 symptoms and/or exposures per person; in one case 63 symptoms and 61 exposures); however, it did find a relationship between types of symptoms and types of exposures reported by respondents which was unexpected to the medical researchers. Also, it suggested a further grouping of types of symptoms based on obvious monotonic trends in the Food (Group A) and Standard Allergens (Group C) exposures.

To define the network we begin with three types of nodes: 1340 people, 68 symptoms and 85 exposures (the network thus has a total of 1493 nodes). A link is defined between a person and a symptom if the person reported that symptom; a link is defined between a person and an exposure if

[8] This dataset came from a University of Toronto study conducted by co-authors and medical researchers Cornelia Baines and Gail McKeown-Eyssen (McKeown-Eyssen,G, Baines, C., *et al.* 2001).

Table 3a. Categories of Symptoms (n = number in category, N=total number reported)

	Category	n	N	Group	Examples
■	Neurocognitive	9	3147	A	forgetfulness, trouble finding words
■	Affect/Mood	6	3051	A	feeling tense, depressed
■	Vegetative	3	743	A	sleeping more, compulsive sleepiness
■	Energy	2	751	A	
■	Musculoskeletal	5	1283	A	tiredness, general weakness
■	Endocrine	1	253	A	muscle pain, muscle weakness
					fast heartbeat
▒	Headache	2	1250	B	other headache, migraine
▒	Gastro-intestinal	6	3129	B	excess gas, bloating
▒	Connective	1	292	B	burning eye
▒	Cardiovascular	1	277	B	irregular heartbeat
▒	Sensory	6	1236	B	light sensitive, bad taste
■	Infection	4	1841	C	sore throat, hoarse voice
■	Allergy	12	4323	C	itchy eye, watery eye
■	Miscellaneous	10	3641	C	sinus fullness, sinus headache
	TOTAL	68	25217		

Table 3b. Categories of Exposures (n=number in category, N=total number reported)

	Category	n	N	Group	Examples
■	Food	29	2002	A	coffee, dairy products
▒	Environmental	11	947	B	tobacco smoke, auto exhaust
▒	Home/Work	15	1005	B	household cleaners, disinfectants
▒	Furnishings	10	620	B	TV screen, carpet
▒	Grooming	4	587	B	perfume, cosmetics
▒	Renovation	3	232	B	paint, sawdust
▒	Pharmaceuticals	5	240	B	prescription, non-prescription medicine
■	Standard Allergens	6	1480	C	pollen, house dust
	TOTAL	85	7113		

the person reported that exposure. The resulting link variable is called “sym-exp” in figures 3 and 4. For each person, this variable has a value for each symptom and each exposure. The value is "1" if a person reports a particular symptom or a particular exposure; otherwise it is "0."

Figure 3 shows nodes placed according to the coordinates of eigenvectors 1, 3 and 5. In figure 3a and 3b the green dots correspond to people, the magenta dots to symptoms and the cyan dots to

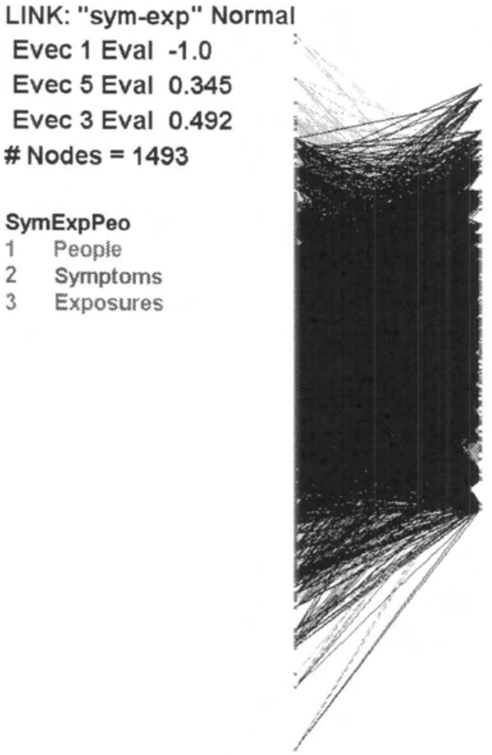


Figure 3a. Bipartite (people on left, symptoms and exposures on right) nature is captured perfectly by eigenvector 1, but lines representing links obscure relationships.

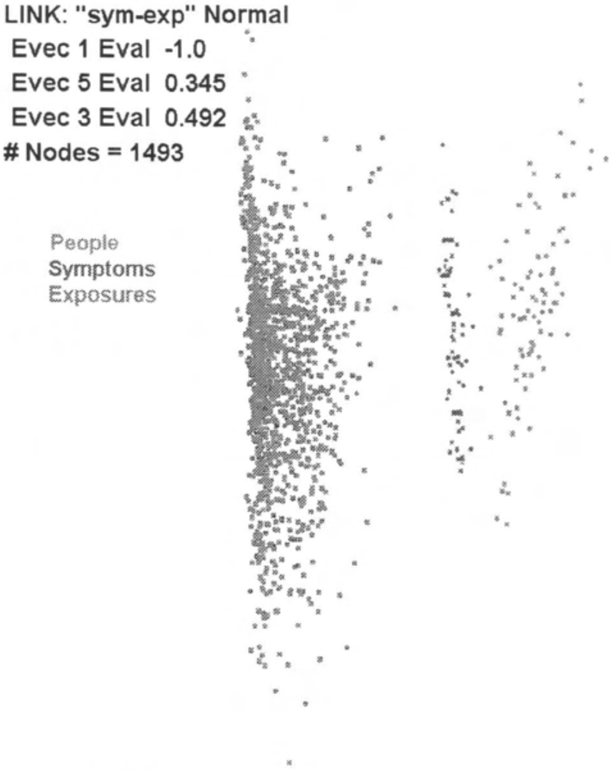


Figure 3b. With lines off and display rotated slightly, clustering on the right becomes clearly visible.

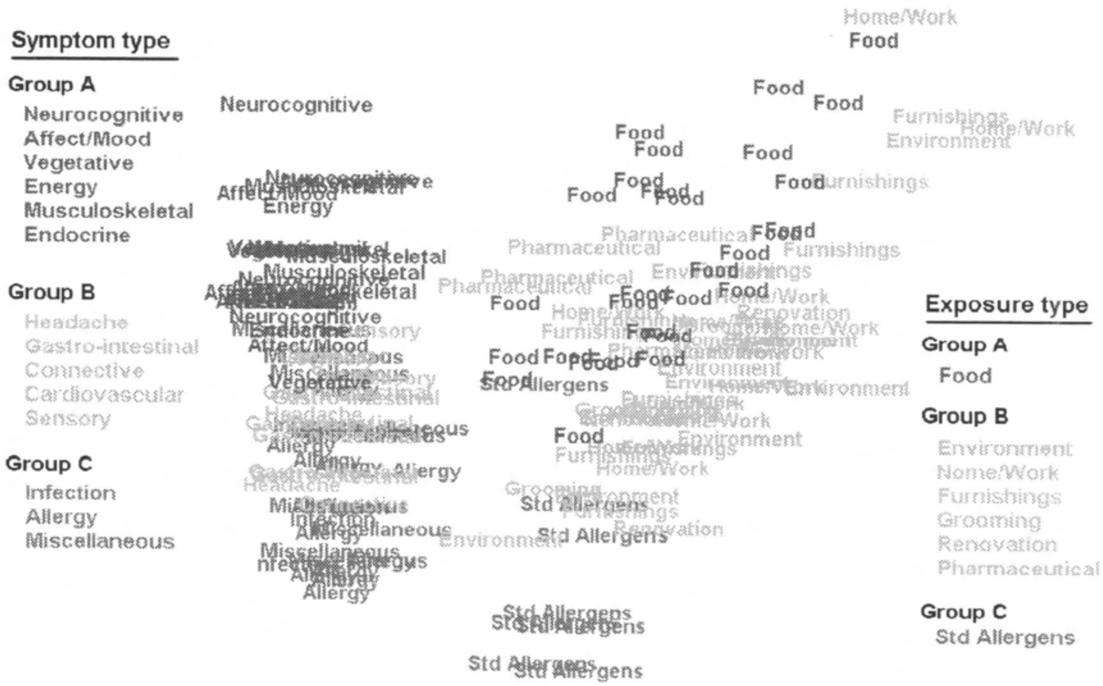


Figure 3c. Close-up of symptoms and exposures labelled by type. Upper and lower extremes show a relationship between symptom and exposure types. Coloured by symptom and exposure groups.

exposures. Since the first eigenvector with eigenvalue -1 perfectly captures bipartite-ness[9], the two parts of the bipartite network (people and symptoms or exposures) each lie along straight lines in the direction of the Y-axis. People report so many symptoms and exposures (high degrees) that the lines representing links obscure the display, so they are turned off in figure 3b, which is also rotated slightly around the Y-axis for clarity. Eigenvector 3 captures the difference in frequency of symptoms and exposures, separating the higher frequency symptoms from lower frequency exposures. As the totals in tables 3a and 3b show, the frequencies of symptoms and exposures are quite different with 25,217 symptoms reported (18.82 symptoms per person) and only 7,113 exposures reported (5.31 exposures per person). Eigenvector 5 captures the simultaneous clustering of symptoms and exposures. Figure 3c shows more detail of the symptoms and exposures nodes labelled by the types assigned by the medical researchers. Both symptoms and exposures cluster by type, with extremes belonging to Neurocognitive symptoms & Food exposures at the upper left and right, and Allergic Symptoms & Standard Allergen Exposures at the lower left and right.

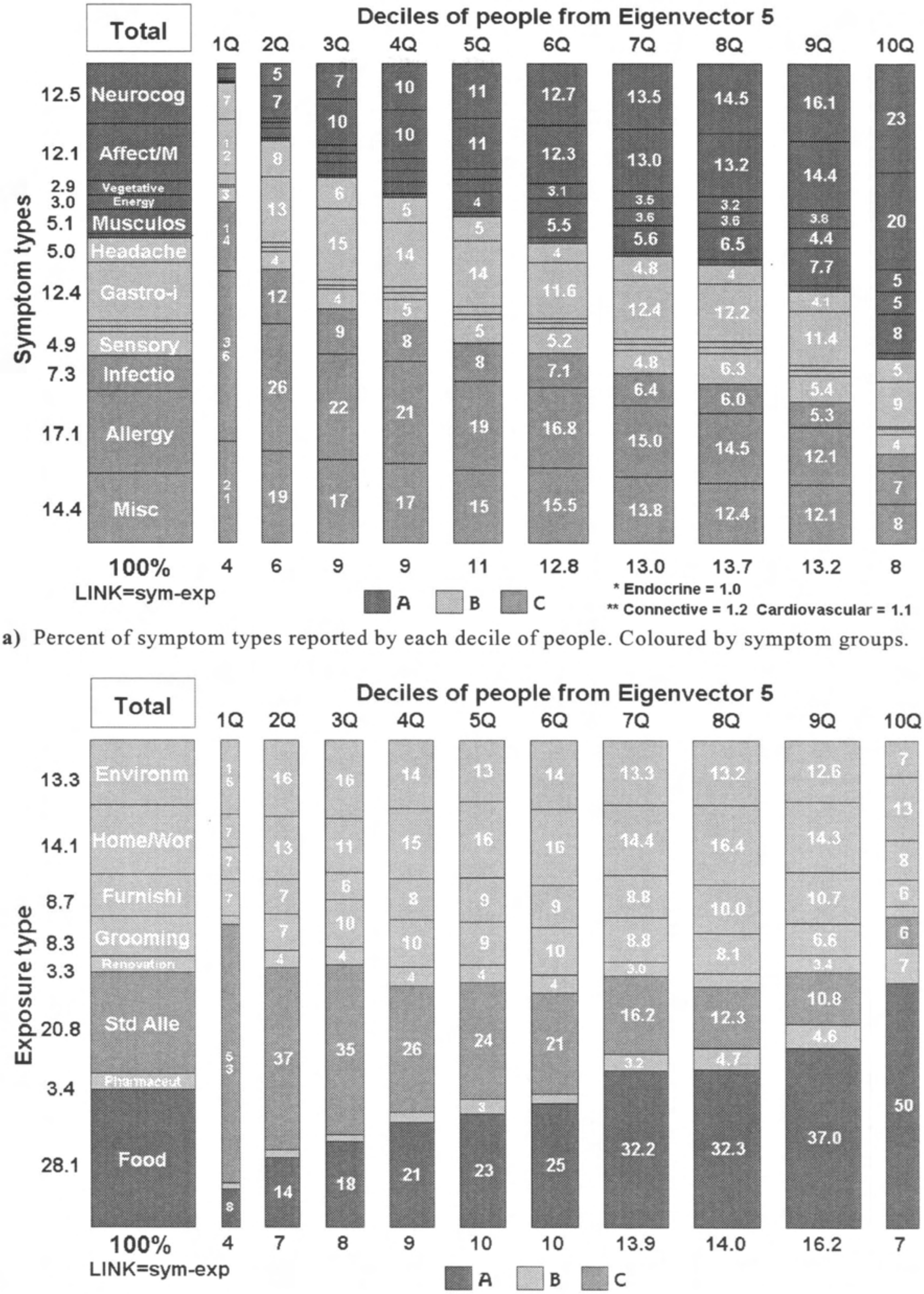
The analytic strategy

In order to quantify the clustering that appears visually in figure 3, we start with a Normal eigen-decomposition of the network using for "links" the variable that describes reported symptoms and exposures. The first and fifth eigenvectors were used to create new variables. The values of the fifth eigenvector for symptoms and exposures were converted to missing, resulting in a variable that contained only values for people. This variable was recoded into deciles so the lowest ten percent of people were "1"; the next ten percent were "2", etc.

We then performed a crosstabulation of symptom reports, using the symptom's type for rows and the person's Decile for columns. We did the same with exposure reports, using the exposure's type for rows and the person's Decile for columns. The set of steps used to do this analysis in MultiNet are explained in an appendix.

The results are shown graphically in panigrams in Figures 4a and 4b. In both cases, each column of the table is represented as a bar whose width is proportional to the column's marginal percentage. In a column, there is a segment corresponding to each row of the table. The heights of these segments are proportional to the column percentages in the corresponding cells of the table.

[9] In bipartite graphs, eigenvalues come in pairs with opposite signs. The eigenvectors associated with each pair contain the same values, but the component values for one part have signs reversed, repeating the bipartite-ness captured by eigenvalue -1. For this reason, we did not use eigenvectors 2 and 4 because the eigenvalues they are associated with are the negative copies of 3 and 5.



a) Percent of symptom types reported by each decile of people. Coloured by symptom groups.

b) Percent of exposure types reported by each decile of people. Coloured by exposure groups.

Figure 4. Panigrams based on symptom and exposure types, with people ordered by eigenvector 5. In both panigrams, unlabelled cells contain less than 3% of their column counts.

Discussion of results

Figure 3c suggests, and the tables[10] visualized in figure 4, confirm that there is a relationship between types of symptoms and types of exposures people report. People who report symptoms in certain categories tend to report exposures in certain categories. For example, more than 50% of the exposures reported by the people in decile 1 are Standard Allergens; almost 75% of the symptoms they report were in Group C (and more than 35% in "Allergy"). That people who report sensitivity to allergens tend to also report allergies is not a surprise, but at the other extreme (of both figure 3 and 4) is the result that more than 50% of the exposures reported by people in decile 10 were "Food" and more than 60% of the symptoms they report were in Group A, with most of these either Neurocognitive (23%) or Affect/Mood (20%). To our knowledge, the relationship between Food exposures and Neurocognitive and Affect/Mood symptoms has not been reported before.

It is clear that eigenvector 5 captures a difference between people who report symptoms related to allergens and those who report symptoms related to food. On the basis of the trends in eigenvector 5, we collected the categories of both symptoms and exposures into the following groups (table 2):

- Group A has column percents which *increase* steadily (almost monotonically) from decile 1 to decile 10. For Exposures, this group consists of Food. For symptoms, this group includes Neurocognitive, Affect/Mood, Vegetative, Energy, Musculoskeletal, Energy and Endocrine, with the first two contributing more than 50% to the counts. In figures 3c and 4, group A is coloured red.
- Group B does not change monotonically from decile 1 to decile 10. This group is coloured yellow.
- Group C has column percents which *decrease* steadily (monotonically except for one data point) from decile 1 to decile 10. For exposures, the group consists of Standard Allergens. For symptoms this includes categories Infection, Allergy and Miscellaneous, with the last two contributing about 80% of the counts. In Figures 3c and 4 group C is coloured blue.

These groupings and colourings are used in figures 3c and 4 to show the smooth relationship between categories (the deciles) of people and the symptoms and exposures they report.

The clusters shown in these figures arise from the relationship between the coordinates of any node in an eigenvector and the coordinates of the nodes it is connected to. This relationship is expressed by equation (1):

$$n_i(s) = \sum_{s \sim t} n_i(t) \Big/ (v_i \times \deg(s))$$

[10] The cross-tab tables visualized by these panigrams are large — the first one has 14 rows and 10 columns (plus marginals) resulting in 500 numbers (including row, column percentages and counts). Please email the authors if you wish to see these tables.

showing that coordinate $n_i(s)$, the s^{th} component of the i^{th} eigenvector, is approximately at the centroid of the coordinates of the nodes s is connected to. The approximation is exact for the constant trivial eigenvector with eigenvalue 1 (where every node has exactly the same coordinate). For eigenvalues far from 1 (which is the case for eigenvector 5 with eigenvalue 0.345), the coordinates can be quite far from the centroid, so that any clusters can be quite smeared out, as we see in figure 3c. Nevertheless, the analytic strategy outlined here can detect small signals and suggest directions for further analysis.

One reviewer suggested that comparable results could be found by using methods such as factor analysis. However, factor analysis would necessarily require reduction over the "cases" of the data (the people), while fitting to the "variables" (the symptoms and exposures). For example, the default SPSS "Factor" routine would apply Principal Components Analysis (Joliffe, 1986) to the symmetric (and therefore square) matrix produced by correlating the columns of variables, which loses all details about the people. Our method is similar to Correspondence Analysis (Greenacre, 1984) which uses Singular Value Decomposition (Press et. al., 1986) to find the related eigenspaces of the symmetric matrices of cases and variables formed by pre- and post-multiplying the data matrix by its transpose. Our method forms a symmetric matrix by constructing a bipartite graph which retains all details about the cases and variables; the cases and variables are in a single eigenspace and are given their own sets of coordinates as the two parts of the bipartite graph. This allows easy calculations and visualizations such as those shown in the panigrams of figure 4.

Another reviewer suggested relating Panigrams to Mosaic displays (proposed by Hartigan & Kleine, 1981) to represent contingency tables. Though there are superficial similarities, the two methods were developed independently and have evolved in different directions. Richards developed panigrams as a way to make the information in large crosstabulation tables easily comprehensible (Richards, 1987). Subsequent developments (Richards, 1988, 1993; Seary, 2005) include transposes, three-way contingency tables, three-mode ANOVA, and interactive exploration and interpretation (e.g., Fig 2 and 4). Panigrams have always included row and column marginals, which have never been part of Mosaic displays. In the early 1990's, Michael Friendly extended mosaic displays so they would incorporate residuals into the tiles, allowing the analyst to know whether the observed data deviates from an expected model. While panigrams are used to illustrate the column (or row) percentages and marginals in a two-dimensional crosstab table, with colours representing the categories of row or column variables, Friendly's method uses colour and shading to represent the sign and magnitude of standardized residuals from a specified loglinear model (Friendly, 1991, 1992).

Conclusions.

The results presented here show that the combination of spectral methods for visualizing and partitioning, and contingency tables with panigrams can lead to the extraction of unsuspected relationships, even with high degrees and low signal. In this case the categories given by the medical researchers were a good match to the patterns in the data. Without such pre-existing categorizations these methods can also suggest alternative ways of categorizing the data by block models which maximize χ^2 derived from spectral results.

Acknowledgment

This novel analysis of human epidemiological data would not have been initiated had it not been for the fact that Andrew D. Baines MD, PhD (Professor, Department of Laboratory Medicine and Pathobiology, University of Toronto) knew of the expertise in social network analysis of the late Steve Berkowitz, Professor of Sociology, University of Vermont. Andrew Baines urged Gail McKeown-Eyssen and Cornelia Baines to discuss their complex dataset with Berkowitz, who, with his customary enthusiasm for problem-solving, introduced them to Andrew Seary and William Richards. This chain of circumstances, a real network of persons developing methods of network analysis, may well have a major and lasting impact on how complex epidemiological databases are analysed in the future. The authors are grateful both to Andrew D. Baines and to the late Stephen D. Berkowitz.

References

Breiger, RL. 1974. The duality of persons and groups. *Social Forces*, **53**(2): 181-190

Breiger, RL, Boorman, S & Arabie, P. 1975. An Algorithm for Clustering Relational Data with Applications to Social Network Analysis and Comparison with Multidimensional Scaling. *Journal of Math. Psych.*, **12**(3): 328-382.

Chow, TY. 1997. The Q-spectrum and spanning trees of tensor products of bipartite graphs", *Proc. Am. Math. Soc.*, **125**(11): 3155-3161.

Chung, FKR. 1995. *Spectral Graph Theory*, CBMS Lecture Notes, AMS Publication.

Crane, D. 1972. *Invisible Colleges: Discussion of Knowledge in Scientific Communities*. Chicago: University of Chicago Press.

Cvetkovic , D, Doob, M, and Sachs, H. 1995. *Spectra of Graphs*. Academic Press.

Davis, A, Gardner, B & Gardner, MR. 1941. *Deep South*. Chicago University Press.

Dhillon, IS. 2001. Co-clustering documents and words using bipartite spectral graph partitioning, UT CS Technical Report #TR 2001-05.

Dodziuk, J & Kendall, WS. 1985. Combinatorial Laplacians and Isoperimetric Inequality. In K.D. Ellworthy (ed.). *From Local Times to Global Geometry*, Pitman Research Notes in Mathematics Series, **150**: 68-74

Fiedler, M. 1973. Algebraic Connectivity of Graphs, *Czech. Math. J.* **23**: 298-305.

Friendly, M. 1991. Mosaic displays for multi-way contingency tables. York Univ.: *Dept. of Psychology Reports*, 1991, No. 195.

Friendly, M. 1992. Mosaic displays for loglinear models. *American Statistical Association, Proceedings of the Statistical Graphics Section*, 1992: 61-68.

Greenacre, M. 1984. *Theory and Application of Correspondence Analysis*. Academic Press.

Hall, K. 1970. An r-Dimensional Quadratic Placement Algorithm. *Management Science*. **17**(3): 219-229.

Hartigan, JA, & Kleiner, B. 1981. Mosaics for contingency tables. In W. F. Eddy (Ed.), *Proceedings of the 13th symposium on the interface between computer science and statistics*, 268-273. New York: Springer-Verlag.

Joliffe, IT. (1986). *Principal Components Analysis*, Springer-Verlag, New York.

McKeown-Eyssen GE, Baines C, Marshall LM, Jazmaji V, & Sokoloff E. 2001. Multiple Chemical Sensitivity: Discriminant Validity of Case. Archives of Environmental Health, **6**: 406-412.

Pothen, A, Simon, H, & Liou, K-P, 1990. Partitioning Sparse Matrices with Eigenvalues of Graphs. *SIAM J. Matrix Anal. App.*, **11**(3): 430-452.

Powers, D. 1988. Graph partitioning by eigenvectors. *Linear Algebra Appl.*, **101**: 121-133.

Press, W, Flannery, B, Teukolsky, S, & Vetterling, W. (1986). *Numerical Recipes*, New York: Cambridge University Press

Richards, WD & Seary, AJ. 1997. Convergence analysis of communication networks, in Barnett, G.A. (ed.), *Advances in Communication Sciences, Vol 15*, Ablex: Norwood NJ, 141-189.

Richards, WD. 1993. FATCAT v4.1VP. A computer program for categorical analysis of multivariate multiplex communication network data. http://www.sfu.ca/~richards/Pdf-ZipFiles/fat41.zip

Richards, WD. 1988. FATCAT ... for Thick Data, *Connections* (the Bulletin of the International Network for Social Network Analysis), Vol. XI, No. 3.

Richards, WD & Rice, R. 1981. The NEGOPY Network Analysis Program, *Social Networks*, **3**(3): 215-224.

Richards, WD. 1971. An Improved Conceptually-Based Method for the Analysis of Communication Networks in Large Complex Organizations. Presented to International Communication Association, Phoenix, Arizona.

Seary, AJ 2005. MultiNet Manual. http://www.sfu.ca/~richards/Multinet/Pages/manual.pdf

Seary, A.J., Richards, W.D., McKeown-Eyssen, G. & Baines, C., "Networks of Symptoms and Exposures", (2005) *Structure and Dynamics: eJournal of Anthropological and Related Sciences* (To appear) http://repositories.cdlib.org/imbs/socdyn/sdeas/

Seary, AJ & Richards, WD. 2003. Spectral methods for analysing and visualizing networks: an introduction. In National Research Council, *Dynamic Social Network Modelling and Analysis: Workshop Summary and Papers*, (209-228). Eds. Ronald Breiger, Kathleen Carley and Phillipa Pattison, Division of Behavioral and Social Sciences and Education. Washington DC. The National Academics Press.

Seary, AJ & Richards, WD. 1998. Some Spectral Facts. Presented at INSNA XIX, Charleston. http://www.sfu.ca/~richards/Pages/specfact.pdf

Seary, AJ & Richards, WD. 1995. Partitioning Networks by Eigenvectors. Presented to European Network Conference, London. Published in Everett, M.G. and Rennolls, K. (eds). 1996. *Proceedings of the International Conference on Social Networks, Volume 1: Methodology*. 47-58. http://www.sfu.ca/~richards/Multinet/Pages/variables.pdf

Seary, AJ 1988. Private communication.

Wasserman, S and Faust, K. 1994. *Social Network Analysis: Methods and Applications*. Cambridge: Cambridge University Press.

Appendix

Perform a Normal eigendecomposition of the network using for "links" the variable that describes reported symptoms and exposures:

- Use **Eigenspaces →Normal** with "sym-exp" – the link variable that describes reported symptoms and exposures

- Use **Define →Variables** to create two new variables ("1N-sym-exp" and "5N-sym-exp") from eigenvectors 1 and 5

- Use **Recode →Equation** to select coordinates on the 5th eigenvector for people (and to exclude symptoms and exposures). This is a two-step process. First, take advantage of the fact that people have negative coordinates on eigenvector 1 (Figure 3): multiply the variable that contains eigenvector 5 by "1N-sym-exp<0" which evaluates to "1" if true and "0" if false. The equation (1N-sym-exp<0)*5N-sym will make the coordinates for Symptoms and Exposures equal to 0. This uses properties 1 (bipartiteness) and 2 (visualization) to isolate the people.

- Use **Recode →Zero ->Missing** to turn these 0 coordinates into missing data, excluding them from the next steps of the analysis. The distribution now includes only coordinates for people.

- Use **Recode →Discrete** option **Quantiles** to categorize the people into deciles. Create a new variable to specify which decile each of the 1340 people is in ("Deciles of people from Eigenvector 5" in figure 4). This uses properties 3 and 4 to produce a partition that should result in large χ^2.

Next, perform a network crosstabulation of symptom reports where symptom type is used for rows and Deciles for columns, then another with exposure types for rows and Deciles for columns:

- Use **Network →Xtabs** to form contingency tables counting the types of symptoms reported by people in each of the 10 deciles (Figure 4a).

- Use **Network →Xtabs** to form contingency tables counting the types of exposures reported by people in each of the same 10 deciles (Figure 4b).

11. Glossary

actor: In SNA, the nodes of a network are usually people, so this term commonly used to refer to a node (vertex, point) in a network.

adjacency matrix: A network (graph) may be represented by a matrix of zeros and ones, with a one indicating that two nodes are connected (adjacent), and a zero otherwise. In a weighted graph, the ones may be replaced by other positive numbers (e.g., a distance or cost).
A sample adjacency matrix is shown below. See *link list*

$$\begin{array}{c|cccccccc}
 & a & b & c & d & e & f & g & h \\
a & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
b & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
c & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\
d & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
e & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
f & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
g & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
h & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0
\end{array}$$

a has connections to b,c,d
b has connections to a,c,d
...etc...

Adjacency spectrum: The adjacency matrix of a graph, like any matrix, may be subject to an eigendecomposition. In graph theory, the resulting set of eigenvalues is referred to as ***the*** graph spectrum, in analogy to the continuous spectrum from continuous spectral analysis methods such as Fourier analysis. In Fourier analysis, the spectrum is understood to refer to the weighting of sines and cosines, whereas the discrete graph spectrum (eigenvalues) are weights of eigenvectors with unknown functional form. We sometimes use the term eigenpair refer to both eigenvalues and eigenvectors. Since there are other spectra associated with graphs, we refer to this one as the Adjacency or Standard spectrum.

anaglyphic 3-D: The same image is presented twice, once in red, and again in cyan after a slight rotation around the z-axis. When such an image is viewed through special glasses with a red filter over one eye (left is recommended) and cyan over the other eye, the brain combines the two images into one with a strong perception of depth.

Analysis of Variance: Statistical technique used to determine whether a continuous variable depends on individual values of a categorical variable by comparing the overall variation to the variation within each category.

attribute: one node (link) variable. The node (link) attributes are the complete set of node (link) variables. Sometimes used interchangeably with variable.

bipartite: network consisting of two parts, with links between the parts, but not within the parts. Bipartite graphs may be used to represent hypergraphs such as two-mode networks.

bin: The number of distinct values a variable attains is referred to as the number of bins.

binned display: Histograms of continuous variables must accumulate close values into "bins" which may be defined by range (e.g., 0-9, 10-19, 20-29,...). However a more general approach is to select a pre-defined number of bins (default 30 for MultiNet), but this also has problems. A small number of bins loses detail, whereas a large number may result in one value in each bin and an unhelpful binned display.

block: A block is a generalization of a clique in the sense that blocks are defined as sets of nodes that have similar patterns of links to nodes in the same or other sets, while cliques is a set of nodes that have most of their links to other nodes in their own set. All cliques are blocks, but some blocks are not cliques. One of the aims of block-modelling is to identify roles by clustering the nodes so that those with similar patterns of connections are next to one another in the matrix. The members of each block perform similar roles in the network.

block model: a higher-level description of a network, where roles (or blocks) are represented by a simplified graph. For the example adjacency matrix above, a block model would be:

```
1 0 0
0 0 1
0 1 0
```

Cartesian sum: A form of graph composition, which forms more complex graphs from simpler ones. Cartesian sum may be expressed in terms of Kronecker product as:

$$\mathbf{A}_1 \oplus \mathbf{A}_2 = \mathbf{A}_1 \otimes \mathbf{I}_2 + \mathbf{A}_2 \otimes \mathbf{I}_1 \qquad \text{(where } \mathbf{I}_1 \text{ and } \mathbf{I}_1 \text{ are identity matrices of appropriate size)}$$

As an example, the Cartesian sum of two paths is a rectangular grid.

change statistics: As implemented in MultiNet, estimation of pstar parameters depends not on local statistics, but on the difference between these statistics as a link is inserted and removed. Usually these are simple functions of the actual local statistics. E.g., the change statistic for Choice is constant

1; the change statistic for Mutuality is the existence of a reciprocal link in the transpose.

clique: In graph theory, a clique is a sub-graph in which all nodes are connected to each other. In social networks, a clique is a set of nodes with most of their connections with other members of the clique. This would generally correspond to an informal role (e.g, friendship). In the above matrix {a, b, c, d} form a clique.

cluster: A collection of points that are "close" to each other in some sense. Many definitions (and related techniques) are available. For networks, we should also insist that the points share connections, either within the cluster (clique) or with another cluster (see block model).

clique: a complete sub-graph. I.e., a subset of nodes with a link between each pair.

complete: a graph in which every pair of nodes is linked is said to be complete.

component: If a graph is connected, it consists of a single component. A disconnected graph does not have a path between any pair of nodes, and may consist of several components.

connected: If there is a path between every pair of nodes in a graph, the graph is said to be connected. A disconnected graph does not have a path between any pair of nodes, and so distances (and diameters) cannot be defined, except within each component. See weakly and strongly connected.

Correspondence Analysis: Eigendecomposition of the chi matrix derived from a data matrix by dividing entries by the square root of the chi-squared expected values. The resulting spectrum of eigenvalues squared is a partition of chi-squared/(sum of data entries). The eigenvectors are scaled by the corresponding eigenvalues (so they are not normalized), and the orthogonality metric is chi-squared Calculation of CA generally uses Singular Value Decomposition which hides the signs of the eigenvalues. In MultiNet, positive eigenvalues are always ensured by placing node degrees along the diagonal of the adjacency matrix. See also Normal spectrum.

database: the complete set of all node and link variables, along with all descriptive items such as comments and statistics, which are either stored together as a single file or as a pair of files (related by name) of the node and link variables.

degree: the number of links a node has to all other nodes is its degree. For directed networks, this may be further defined as

- * OUT-degree: the number of links TO other all other nodes, and
- * IN-degree: the number of links FROM all other nodes.

density: the number of links in a network compared to the maximum number possible. This maximum depends on whether the network is undirected (N/(N+1)/2) or directed (NN) and with or without diagonal (N/(N-1)/2 or N(N-1) respectively). Social networks of any of these types typically have low density (0.1 or less). Such networks are said to be sparse.

directed: Some relationships are inherently one-way only (e.g., "child of"). Some directed relationships may be reciprocated (e.g., "get advice")

disabled: Menu items that are not currently available are shown as greyed-out. Clicking on such items has no effect. E.g., In Eigenspaces, most items are disabled until a network (lin

distance: For graphs, the distance between nodes is defined as the smallest number of links connecting them. Also called geodesic distance.

diameter: the largest geodesic distance between any pair of nodes in a graph

disconnected: a network containing nodes with no paths connecting them is disconnected, and consists of a number of smaller connected networks called components.

display window: Two display windows are used by MultiNet: one displays the current list of Node variables and the other the current list of link variables. Both are scrollable, and items may be highlighted. Both are closed by closing either one. Both are displayed whenever MultiNet returns to the main window.

dyad: any pair of nodes, connected or not. In SNA, any dyad has a potential set of links (one-way and reciprocated).

edit window: A multi-line, scrollable window containing text which may be changed. An edit window has Edit in the title bar and menu choices Quit and Save. Selecting Save replaces old text with new. Pressing Quit results in no change.

ego-centric: network data collected by asking a (generally small) number of people to nominate other people and describe their relationships with these others. Such data is generally not connected or reciprocated.

eigendecomposition: Any symmetric matrix may be decomposed into a set of spaces defined by (usually orthogonal and normalized) eigenvectors, with the importance of each space determined by scalar eigenvalues. In general, the eigenvalues need not be unique and the dimensionality spanned by the eigenvectors need not be the size of the original matrix, although these matters are not usually important for social networks. What is important for SNA is that the most important (largest eigenvalues) dimensions give the best least-squares fit to the original matrix and hence describe the most important global properties of the network (e.g., connectedness, diameter, number of clusters). See also spectral analysis.

eigenpair: the pair (eigenvalue, eigenvector).

eigenspace: the complete set of eigenpairs. For distinct eigenvalues, each eigenvector defines a dimension, and the corresponding eigenvalue defines the importance of the dimension (contribution to the original network). Other properties of the eigenspace (e.g., orthonormality conditions) depend on how the original data matrix was treated (see Adjacency, Laplacian, Normal).

error window: MultiNet traps three types of errors: Warning (information is displayed, but the user may proceed after pressing Okay), Error (a condition was checked for and failed, the user may not proceed after pressing Okay), and Internal Error (an unexpected error has occurred). The last case may result in other Internal Error messages as Okay is pressed until the main menu is reached. Errors of the last type should be reported to the author.

exponential random graph: Model for directed graphs in which the probabilities of links is calculated based on an exponential function of fit weights times actual count statistics (see triad counts). Many graphs may produce the same counts, so some quality of fit measure is required to determine how well the fit describes a particular graph, in pstar this is -2LogPseudolikelihood.

gap: The term gap or spectral gap refers to large distances in the spectrum of eigenvalues, particularly between 0 and the second-smallest (Laplacian) or between 1 and the second-largest in absolute value (Normal). A small gap means that a graph can be disconnected with few edge-cuts;

a large gap means there are many paths between sets of nodes.

geodesic: the shortest path (i.e., the smallest number of links) connecting two nodes. There may be more than one geodesic between pairs of nodes.

global vs local methods: In graph theory, a local method is one that examines only a few neighbours of a node. A global method is one which examines the entire graph, such as an eigendecomposition.

grouping: a set of proportional link variable describing content or purpose that are collected together for analysis (cross-tabulation or ANOVA).

header: Before data files can be Imported, the data within them must be described. This is done in the header section, which describes mandatory variable names and locations, and optional value labels and comments. The header section ends in "end" or "begin data".

hypergraph: A simple graph consists of links between pairs of nodes. A hypergraph generalizes this to hyperlinks between sets of more than 2 nodes. An example is a (two-mode) network of people and the events they attend: an event may be attended by any number of people, and each person may attend many events. A hypergraph may always be represented by a bipartite graph.

ID number: Identity number. Used externally in collecting data to ensure privacy. Use internally to most analysis programs as an efficient method for handling data related to individuals.

isolate: a node with no connections.

k-star: The (out/in-) degree of a node. This is one of the local statistics that may be used in pstar fitting, though it is currently not implemented in MultiNet.

Kronecker product: A form of graph composition, which forms more complex graphs from simpler ones. An example of the Kronecker product is:

$$\begin{matrix} 1 & 1 \\ 1 & 0 \end{matrix} \otimes \begin{matrix} 1 & 0 \\ 0 & 1 \end{matrix} = \begin{matrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{matrix}$$

where every 1 in the first matrix has been replaced by a complete copy of the second matrix. In this example the first matrix is a block model, not a graph.

Lanczos iteration: a generalization of the power method which allows calculation of a specified number of eigenpairs without loss of precision or orthogonality. Currently one of the best methods for eigendecomposition of large systems.

Laplacian spectrum: The eigenvalues (and eigenvectors) of a matrix formed by subtracting the adjacency matrix from a diagonal matrix of node degrees. The eigenvalues are non-negative, with a "trivial" (constant) eigenvector of eigenvalue 0. This discrete analogue of the continuous Laplacian shares a great many of its important properties. For this reason, it has become the focus of much research in the last decade.

liaisons: according to NEGOPY, these come in two types. *Direct* liaisons are individuals who have most of their interaction with members of groups, but not with members of any one group. They provide direct connections between the groups they are connected to. *Indirect* liaisons are individuals who do not have most of their interaction with members of groups. They provide indirect or 'multi-step' connections between groups by connecting Direct Liaisons, who have direct connections with members of groups (Richards, 1995).

Likert scale: a standard method for categorizing subjective information, generally running from 1=strongly disagree through 3=neutral to 5=strongly agree.

link: A pair of nodes with some connection between them. In graph theory, links are also called edges or lines. In social networks, links are often called ties.

link list: A sparse format for storing information in a network. Only the pairs of nodes that are connected are in the link list. For symmetric graphs, only one pair is needed for each link. For weighted graphs, a third column may be used to hold the weights. For the symmetric adjacency matrix shown above, the link list is:

1	2
1	3
1	4
2	3

...and so on...

local statistics: Counts made in the immediate neighbourhood of nodes. These include e.g., degree, dyad and triad counts but not e.g. betwenness, diameter or other distance measures beyond nearest neighbours.

localized: As applied to an eigenvector means that most of the coordinates are near zero, and only a few have large values. Coordinates may be either positive or negative, and the eigenvectors are normalized to make the sum of squares of components 1, so the sum of 4th powers is generally used as a measure of localization. If this sum is near 1 only a small number of coordinates are important. If it is near 1/m, then all nodes contribute to the eigenvector.

logical equation: An equation which evaluates only to values of 0 or 1. These are generally constructed from assertions such as a>0 (=0 if a is 0 or negative; =1 if a is positive) which are combined with other variables to make selections. E.g. in the **Variables** module

a>0*b = 0 if a<=0
= b if a>0

which selects all values of b for which a is positive. The remaining zeros can then be removed by Zero->missing. The example shows that logical equations may be combined with + (logical or) and * (logical and).

main menu: horizontal list of MultiNet modules which may be selected by single left-click

missing data: MultiNet handles missing data by marking every variable with a bit string which has binary value 1 for valid data, and binary value 0 for missing data. There are a large number of reasons for missing data, the most common being non-response. Data with questionable values (outside a prescribed range) may also be converted to missing values. MultiNet ensures that combinations of variables are defined for non-missing values only. This may be over-ridden by converting missing values to 0's, although this causes fewer problems for link variables (where 0 means no relationship) than node variables (where 0 may be a valid data value).

multiple selection window: drop-down, scrollable list of items for making multiple selections. All items in the list are initially selected (and high-lighted), and the list is scrolled to the bottom. Shift -clicking selects a range of items, which are high-lighted. Control-clicking toggles selections on (high-lighted) or off. When selections are done, clicking on Okay completes the multiple selection of all high-lighted items. Help may provide context-sensitive help or general advice about selecting multiple items. Clicking on Cancel cancels all selection, which is detected by the program.

multiplex:Except in very formal situations, the purpose and content of contacts between people are generally multiplex: there are a number of purposes and a number of content areas. See proportional link variables and groupings.

neigbourhood: all the nodes which are connected to a given node. May be extended to all nodes connected to a *set* of nodes, but not including the original set.

NEGOPY: (NEGative entrOPY) (Richards and Rice, 1981, Richards, 1995) is a computer program designed to find clique structures. It uses a random starting vector, and multiplies it by the row-normalised adjacency matrix, subtracting off row means. Usually 6-8 such iterations are performed, resulting in a vector which is a mixture of the important Normal eigenvectors (Richards and Seary, 1997). This vector is then scanned for potential clique structures, which are tested against the original network and for some statistical properties (e.g., variance in the node degrees). Sparse matrix methods are used throughout, allowing large networks to be analysed rapidly.

Network analysis: In MultiNet, mixing node, link and group variables together into cross-tabulations, ANOVA and correlations that involve counting the number (or amount) of interactions that connect nodes of the same or different types in a network. E.g., The length of time spent talking about work (a link variable) between the sexes (the same node variable ate each end of the link).

node: An object that may have some kind of connection (link) to another object. In some cases, nodes are people, organizations, companies, countries, etc. In graph theory nodes are also called vertices and points. In social networks, nodes are often called actors.

Normal spectrum: The eigenvalues (and eigenvectors) of a row-normalised adjacency matrix. This matrix is row-stochastic, and similar to a symmetric matrix, so its eigenvalues are real and less than or equal to 1 in absolute value. It is closely related to the Laplacian (indeed, it may be defined to be the Laplacian in the χ^2 metric defined by the node degrees). In MultiNet, the diagonal of the adjacency matrix left at 0, so that the trace (sum of eigenvalues) is also 0, ensuring both positive and negative eigenvalues. See also Correspondence Analysis.

panigram: a two-dimensional display showing marginal and column percents in a cross-tabulation. Colours are used to represent the row categories.

partition: A partition of a graph is a division of the nodes into a collection of non-empty mutually exclusive sets. A partition of the adjacency matrix shown above could be: {a, b, c, d}, {e, f, g, h}, so that there are no links between the nodes in each part of the partition.

proportional link variable: a valued (non-binary) link variable with values that describe the importance of an interaction on some simple scale (e.g.,0 = none,...5=all). Proportional link variables may be used to describe both purpose and content of interactions (e.g., the meeting was mostly informal and we talked a bit about school), and are generally used in groupings that describe multiplex interactions over a certain time period.

quality of fit: measure used to describe how well a theoretical model predicts actual measurements. Usually based on the maximization or minimization of some well-defined function. In eigenspace, this is the mean-square difference between a low dimensional approximation and the actual network. In pstar, this is usually taken to be -2LogPseudolikelihood.

reciprocated: a directed relationship such as "get advice" may connect two actors in both directions. The link is then said to be reciprocated.

resizing: Any of the windows that show the resize controls in the upper left may be resized to either full-screen or larger or smaller under user control by clicking and holding the mouse at the lower right corner. In particular, the Report and Graphic display windows may be resized. The latter may be made smaller so that bitmap images take up less space (for, e.g., use in published results). For this reason, changing graphics sizes also changes displayed text font size, and some practice may be necessary to ensure that the results are satisfactory at smaller sizes.

selection window: drop-down, scrollable list of items for making a single selection. The first item in the list is initially selected (and high-lighted). Double-clicking on any item, or single-clicking to select an item (which high-lights it), followed by clicking on Okay selects the item. Clicking on Help may provide context-sensitive help or general advice about selecting an item. Clicking on Cancel cancels any selection, which is detected by the program.

self-link: A link that connects a node to itself. Some relationships allow such links (e.g. "Vote for"), while others forbid it (e.g., "married to"). In an adjacency matrix, a self-link appears on the diagonal.

sparse methods, sparse matrix techniques: In analysis of networks with more than 50 or 60 members, it is usually the case that each node is connected to only a fraction of the others. The adjacency matrix for such networks contain mostly zeroes, which indicates the absence of links. In these situations, it far is easier to work with a list of the links (link list) that are present, rather than the whole matrix which contains many times more numbers. Any array (such as an adjacency matrix) which consists mostly of some default number (usually zero) may be treated as a sparse matrix. Since this value is known, it does not need to be stored as part of the array. This allows the array to be stored in a much more efficient manner, e.g., for an adjacency matrix, we only need to store the links (pairs of nodes) when they exist. For a weighted adjacency matrix, we also need to store the values of the weights, one for each link. Many matrix operations (e.g., multiplying a matrix by a vector) can utilize this more efficient storage to run much faster as well. Sparse matrix techniques are those which avoid any manipulation of the matrix that would affect the sparseness property (e.g., taking the inverse will generally do this, as will correlating each row or column with all the others). It is quite possible to find eigenvalues and eigenvectors using sparse techniques.

spectral analysis or methods: Loosely speaking, another term for eigendecomposition. Mathematically speaking, a general term referring to any re-statement of some function in terms of a set of basis functions (e.g. sines and cosines for Fourier analysis). The spectrum is the weights of these basis functions. The Fourier transform is especially useful in mathematical physics since the sines and cosines (or e^z for complex z) are eigenfunctions of the ubiquitous derivative and integral operators. The terms function, operator and eigenfunction have the discrete analogues of vector, matrix and eigenvector.

Standard spectrum: see Adjacency spectrum

Standard Analysis: Ordinary Cross-tabs, ANOVA or correlations as applied to either node or link variables. Node variables may be combined only with other node variables, and similarly for link variables.

strongly connected: A network with directed links is said to be strongly connected if there is a path of directed links connecting every pair of nodes. A directed network may be weakly connected but not strongly connected. An undirected network which is connected is both weakly and strongly connected.

structural equivalence: Two nodes are structurally equivalent if they have identical links to all other nodes in the network. This strict definition is usually relaxed for block-modelling by replacing "all" with "most".

symmetric: A symmetric network is one in which every link is known to be reciprocated (see undirected) and so may be stored and manipulated in efficient ways. E.g., a symmetric network requires at most N(N+1)/2 storage (or N/(N-1)/2 if the diagonal is not included).

text window: a single-line edit window, generally with some default value high-lighted. Generally used to select variable names and comments.

tie: In SNA, the links of a network are usually between people, so this term is commonly used to refer to a link (edge, line) in a network.

title bar: In Windows, this is the coloured bar at the top of every window which generally contains the name of the program controlling the window. In MultiNet, this area may also contain additional information about what the current module is doing (e.g., Module name, File name(s), **Analyse** variable(s)).

triad counts: A dyad has potentially four types of linkage: 1) none, directed 2) I->j and 3) I<-j, and reciprocated 4) i<>j. A triad has potentially 16 types of linkage, based on the various possible dyad combinations. These are the counts that MultiNet uses in pstar fitting.

two-mode: network data which consists of two different types of nodes. A typical example is people and the events they attend together. May be represented by bipartite graphs.

undirected: an undirected link is one which is inherently reciprocated (e.g. "married to")

undirected network: An undirected network is one in which all the links are inherently reciprocated (e.g., "is married to"). See symmetric.

value labels: Categorical variables, or discrete variables with few enough distinct values, may use value labels to represent the actual values, as described in the header or when defined in the **Variables** module. This is convenient since data must be coded numerically but is much easier to understand with descriptive labels. For example, 1= female, 2 = male is easier to understand in tables and graphics by using "female" and "male" rather than the arbitrary "1" and "2".

view window: A multi-line, scrollable window containing text which may be viewed but not changed. A view window has View in the title bar and single menu choice Quit. Pressing Quit exits the View window with no change. View windows are used throughout MultiNet to present Report text results. Value labels are also useful as a check against error.

weakly connected: A network with directed links is said to be weakly connected if there is a path of links, ignoring direction, connecting every pair of nodes. A directed network may be weakly connected but not strongly connected. An undirected network which is connected is both weakly and strongly connected.

YesNo window: Before any action that may have important results (deleting a variable, starting a large calculation, ending MultiNet) a YesNo window asks a relevant question and waits for a Yes or No answer before proceeding. The context-dependent default answer is initially selected, and is the value returned by pressing Enter or clicking on x.

Bibliography

Aho, A., Hopcroft, J., & Ullman, J. (1997). *Data structures and algorithms* Addison-Wesley.

Albert, R. and Barabasi, A.-L. (2002) Statistical mechanics of complex networks, *Review of Modern Physics* **74**: 47-97

Alon, N. (1986). Eigenvalues and Expanders. *Combinatorica* **6** (2),73-88.

Alon, N. and Millman, V. (1985). λ_1, Isoperimetric Inequalities for Graphs, and Superconcentrators. *J. Comb. Theory B.* **38**: 73-88.

Anderson, M. and Morley, T. (1985). Eigenvalues of the Laplacian of a Graph. *Linear and Multilinear Algebra* **18**: 141-145.

Aspvall, B. and Gilbert, J. (1984). Graph Coloring using Eigenvalue Decomposition. *SIAM J. Alg. Disc. Math.* **5** (4), 526-538.

Barnard S. and Simon, H. (1994). Fast Implementation of Recursive Spectral Bisection for Partitioning Unstructured Problems. *Concurrency: Practice and Experience* **6** (2): 101-117.

Barnes, E. (1982). An Algorithm for Partitioning the Nodes of a Graph,.*SIAM J. Alg. Disc. Math.* **3** (4), 541-550.

Barnett, G.A. and Richards, W.D. (1991, February). A comparison of NEGOPY's clique detection algorithm with correspondence analysis. Paper presented to the Sunbelt Social Networks Conference, Tampa. FL.

Barnett, G., (1993). Correspondence Analysis: A Method for the Description of Communication Networks. In W. Richards, and G. Barnett (Eds.). *Progress in Communication Sciences XII.* Newark NJ: Ablex . 135-164.

Benzecri, J-P. (1992). *Correspondence Analysis Handbook*, Marcel Dekker Inc.

Benzecri, L. (1969). Statistical Analysis as a Tool to Make Patterns Emerge from Data. In S. Watanabe (ed.). *Methodologies of Pattern Recognition*, Academic Press. 35-74.

Berry, M. (1992). Large Scale Sparse Singular Value Computations, *Int. Jnl. Supercomputer Appl.*, **6** (1): 13-49

Bien, F. (1989). Constructions of Telephone Networks by Group Representations. *Notices of the Am. Math. Soc.* **36**: 5-22.

Biggs, N. (1993). *Algebraic Graph Theory*. Cambridge University Press.

Bonacich P (1972). Factoring and Weighting Approaches to status scores and clique identification. *Journal of Mathematical Sociology* 2: 113-120.

Brandes, U., & Cornelsen, S. (2001). Visual Ranking of Link Structures, *Lecture notes in Computer Science*, 2125, Springer-Verlag.

Brandes, U. (2001), A faster algorithm for betweenness centrality, *Journal of Mathematical Sociology* **25**, 163-177

Breiger, R. L. (1974). The duality of persons and groups. *Social Forces*, **53** (2): 181-190

Breiger, R.L., Boorman, S. and Arabie, P. (1975). An Algorithm for Clustering Relational Data with Applications to Social Network Analysis and Comparison with Multidimensional Scaling. *Journal of Math. Psych.*, **12** (3): 328-382.

Buser, P. (1982). A Note on the Isoperimetric Constant, *Ann. Sci. Ecole Norm. Sup.* **15**: 213-230

Carroll, J.D., Green, P.E., & Schaffer, C.M. (1986) Interpoint distance comparison in correspondence analysis, *J. Market Research* 23: 271-280

Carroll, J.D., Green, P.E., & Schaffer, C.M. (1987) Comparing interpoint distance comparison in correspondence analysis: a clarification, *J. Market Research* **24**: 445-450

Chavel, I. (1984). *Eigenvalues in Riemannian Geometry*, Academic Press.

Cheeger, J. (1970). A lower Bound for the Lowest Eigenvalue of the Laplacian. In R. C. Gunning (ed.). *Problems in Analysis*, Princeton Univ. Press, 195-199

Chow, T.Y. (1997). The Q-spectrum and spanning trees of tensor products of bipartite graphs", *Proc. Am. Math. Soc.*, **125** (11): 3155-3161.

Chung, F.K.R.(1988). Diameters and Eigenvalues, *J. Am. Math. Soc.* **2** (2): 187-196.

Chung, F.K.R, Faber, V.& Manteuffel, T.A. (1994). An upper bound in the diameter of a graph from eigenvalues associated with its Laplacian, *SIAM J. Disc. Math.* **7** (3): 443-457.

Chung, F.K.R. (1995). *Spectral Graph Theory*, CBMS Lecture Notes, AMS Publication.

Clair, S. (2000) *Personal communication*

Cotterchio, M., McKeown-Eyssen, G., Sutherland, H., Buchan, G., Aronson, M.,Easson, A.M., Macey, J., Holowaty, E., & Gallinger, S. (2000) *Ontario Familial Colon Cancer Registry: Methods and first-year response rates*, Chronic Diseases in Canada **21** (2): 81-86

Courant, R. and Hilbert, D. (1966). *Methods of Mathematical Physics*. Interscience Publishers.

Crane, D. (1972). *Invisible Colleges: Discussion of Knowledge in Scientific Communities*. Chicago: University of Chicago Press.

Crouch, B. and Wasserman, S. (1998). A Practical Guide to Fitting p* Social Network Models, *Connections* **31**: 87-101.

Crow, E. L.& Shimizu, Eds.(1988).*Lognormal distributions: theory and applications*, Statistics, textbooks and monographs ; vol. **88**. M. Dekker

Cvetkovic , D., Doob, M., and Sachs, H. (1995). *Spectra of Graphs*. Academic Press.

Cvetkovic , D., Doob, M., Gutman, I., Torgasev, A. (1988). *Recent Developements in the Theory of Graph Spectra*. North-Holland.

Cvetkovic, D., Rowlinson, P. (1990). The largest eigenvalue of a graph: a survey. *Linear and Multilinear Algebra*. **28**: 3-33.

Davis, A., Gardner, B. and Gardner, M.R. (1941). *Deep South*. Chicago University Press.

Dhillon, I.S. (2001). Co-clustering documents and words using bapartite spectral graph partitioning, UT CS Technical Report #TR 2001-05.

Diaconis, P. and Stroock, D. (1991). Geometric Bounds for Eigenvalues of Markov Chains, *Ann. Appl. Prob*. **1**: 36-61.

Dodziuk, J. (1984). Difference Equations, Isoperimetric Inequality and the Transience of Certain Random Walks. *American Mathematical Society*. **284** (2), 787-794.

Dodziuk, J. and Kendall, W. S. (1985). Combinatorial Laplacians and Isoperimetric Inequality. In K. D. Ellworthy (ed.). *From Local Times to Global Geometry*, Pitman Research Notes in Mathematics Series **150**: 68-74

Donath, W. and Hoffman, A. (1973). Algorithms for Partitioning Graphs and Computer Logic Based on Eigenvectors of Connection Matrices, *IBM Technical Disclosure Bulletin*. **15** 938-944.

Farkas, I., Derenyi, I., Barabasi, A-L., Viscek, T. (2001). Spectra of "Real-world" graphs: beyond the semi-circle law, cond-mat/100235

Fiedler, M. (1973). Algebraic Connectivity of Graphs, *Czech. Math. J.* **23**: 298-305.

Fill, J. A. (1991). Eigenvalue Bounds on Convergence to Stationarity for Nonreversible Markov Chains, with an Application to the Exclusion Process, *Ann. Appl. Prob* 1: 62-87

Fisher, M. (1966). On hearing the shape of a drum, *J. Combin. Theory,* **1**: 105-125.

Fisher, R. A. (1940). The Precision of Discriminant Functions, *Annals of Eugenics* **10**: 422-429

Foster, K.C.,. Muth, S.Q., Potterat, J.J and Rothenberg, R.B. (2001), A Faster Katz Status Score Algorithm, *Computational & Mathematical Organization Theory* **7**, 275–285

Frank, O. and Strauss, D. (1986). Markov Graphs, *J. Am. Stat. Assoc.* **81**: 832-842.

Freeman, L.C. (1978), Centrality in Social Networks: Conceptual Clarification, *Social Networks* **1**, 215–239.

Freidman, J. (1993). Some Geometrical Aspects of Graphs and their Eigenfunctions. *Duke Mathematical Journal,* **69**: 487-525.

Friendly, M. (1991). Mosaic displays for multi-way contingency tables. York Univ.: *Dept. of Psychology Reports*, 1991, No. 195.

Friendly, M. (1992). Mosaic displays for loglinear models. *American Statistical Association, Proceedings of the Statistical Graphics Section*, **1992**: 61-68.

Garey, M., Johnson, D. (1979). *Computers and Intractability*. W. H. Freeman.

Goh, K-I., Kahng, B. Kim, D. (2001). Spectra and eigenvectors of scale-free networks, cond-mat /0103337.

Gould, P. (1967). The Geographical Interpretation of Eigenvalues, *Institute of British Geographers Transactions*, **42** 53-85.

Greenacre, M., (1984). *Theory and Application of Correspondence Analysis*. Academic Press.

Grone, R. and Merris, R. (1994). The Laplacian Spectrum of a Graph II. *SIAM J. Discrete Math.* **7** (2): 221-229.

Grone, R., Merris, R. and Sunder, V. (1990). The Laplacian Spectrum of a Graph. *SIAM J. Matrix Anal. App.* **11** (2): 218-238.

Guttman, L. (1941).The Quantification of a Class of Attributes. In The Committee of Social Adjustment (ed.). *The Prediction of Social Adjustment*, New York: Social Research Council, 319-348

Hagen, L. (1992). New Spectral Methods for Ratio Cut Partitioning and Clustering, *IEEE Trans. CAD*, **11** (9): 1074-1085.

Hall, K. (1970). R-dimensional Quadratic Placement Algorithm, *Management Sci.* **17**: 219-229.

Handcock, M. (2003) Statistical Models for Social Networks: Inference and Degeneracy In National Research Council, *Dynamic Social Network Modeling and Analysis: Workshop Summary and Papers*, (229-240). Eds. Ronald Breiger, Kathleen Carley and Phillipa .Pattison, Division of Behavioral and Social Sciences and Education. Washington DC. The National Academics Press

Harary, F. & Schwenk, A. (1979). The spectral approach to determining the number of walks in a graph, *Paciific J. Math.* **80**: 443-449.

Hartigan, J. A., & Kleiner, B. (1981). Mosaics for contingency tables. In W. F. Eddy (Ed.), *Proceedings of the 13th symposium on the interface between computer science and statistics*, 268-273. New York: Springer-Verlag.

Hendrickson,B. and Leland, R. (1995). An improved spectral graph partitioning algorithm for mapping parallel computations. *SIAM J. Comput. Sci.* **16** (2): 452-469.

Hoffman, D., Franke, G. (1986). Correspondence Analysis: Graphical Representation of Categorical Data in Marketing Research. *J. Market Research* **23**: 213-227.

Holland, P.W. and Leinhardt,S. (1981) An exponential family of probability distributions for directed graphs, *J. Am. Stat. Assoc* **76**: 33-65

Horst, P. (1935). Measuring Complex Attitudes, *Journal of Social Psychology* **6**: 369-374

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components, *J. Educ. Psychol.* **24**: 417-441, 498-520.

Joliffe, I.T. (1986). *Principal Components Analysis*, Springer-Verlag, New York.

Kernighan, B. and Lin, S. (1970). An Efficient Heuristic Procedure for Partitioning Graphs, *Bell System Tech. J.* **49** 291-307.

Kirchoff, G. (1847). Über die Auflösung der Gleichungen, auf welche man bei derUntersuchung der linearen Verteilung galvanischer Ströme gefürte wird, *Ann. Phys.Chem*, **72**: 497-508

Kirkpatrick, S., Gelatt, C., and Vecchi, M. (1983). Optimization by Simulated Annealing, *Science.* **220**: 671-680.

Koren, Y. Carmel, L., Harel, D. (2002). ACE: a fast multiscale eigenvector computation for huge graphs, *IEEE Symposium On Information Visualization.*

Lanczos, C. (1961). *Linear Differential Operators*. Van Nostrand, Dover 1997.

Lawler, G. F. and Sokal, A. D. (1988). Bounds on the L^2 Spectrum for Markov Chains and Markov Processes: a Generalization of Cheeger's Inequality, *Trans. Amer. Math. Soc.* **309**: 557-580.

Lorrain, F. & White, H.C. (1971) Structural equivalence of individuals in social networks, *J. Mathematical Sociology*, **1**: 49-80

Lovasz, L. (1995). Random Walks, Eigenvalues and Resistance, Handbook of Combinatorics, Elsevier, 1740-1745.

Lubotzky, A. (1994). *Discrete Groups, Expanding Graphs, and Invariant Measures*, Birkhauser.

McKeown-Eyssen GE, Baines C, Marshall LM, Jazmaji V, Sokoloff E. (2001). Multiple Chemical Sensitivity: Discriminant Validity of Case. *Archives of Environmental Health*, **6**: 406-412.

McQuitty, L.L., and Clark, J.A. (1968). Clusters from iterative intercolumnar correlational analysis, *Educational and Psychological Measurement.* **28**: 211-238.

Merris, R. (1994). Laplacian Matrices of Graphs: A Survey, *Linear Alg. App.* **198,199**: 143-176

Mohar, B. (1991). The Laplacian Spectrum of Graphs. In Alavi, Chartrand, Ollermann and Schwenk (eds.). *Graph Theory Combinatorics and Applications*, Wiley, 871-898.

Mohar, B., Poljak, S. (1991). Eigenvalues and the Max-cut Problem, *Czech. Math. J.* **40**: 343-352.

Moody, J. (2001). Peer influence groups: identifying dense clusters in large networks, *Soicla Networks*, **23**: 261-283.

Motwani, R. and Prabhakar, R. (1996). *Randomized Algorithms*, Cambridge Univ. Press

Newman, M., Strogatz, S. and Watts, D. (2001) Random graphs with arbitrary degree distributions and their applications, *Physical Reviews E* **64**:026118

Noma, E. and Smith, D.R. (1985). Scaling sociomatrices by optimising and explicit function: correspondence analysis of binary single response sociomatrices. *Multivariate research.* **20**: 179-197.

Parlett, B. (1980). *The Symmetric Eigenvalue Problem*, Prentice-Hall.

Parlett, B., Simon, H. and Stringer, L. (1982). On Estimating the Largest Eigenvalue with the Lanczos Algorithm. *Mathematics of Computation.* **38** (157): 153-165.

Pattison, P. and Robins, G. (2000) Neighbourhood-based models for social networks, *preprint.* Presented at INSNA XX, Vancouver 2000.

Pattison, P., Robins, G., and Wasserman, S. (1999) Some computational notes on univariate and multivariate random graph models for social networks. *Preprint.*

Pothen, A., Simon, H, and Liou, K-P., (1990). Partitioning Sparse Matrices with Eigenvalues of Graphs. *SIAM J. Matrix Anal. App.*, **11** (3): 430-452.

Powers, D. (1988). Graph partitioning by eigenvectors. *Linear Algebra Appl.*, **101**: 121-133.

Press, W., Flannery, B., Teukolsky, S., Vetterling, W. (1986). *Numerical Recipes*, New York: Cambridge University Press

Richards, W.D. (1995). *NEGOPY 4.30 Manual and user's Guide*, School of Communication, SFU. http://www.sfu.ca/~richards/Pdf-ZipFiles/negman98.pdf

Richards, W.D. (1989). *A Manual for FATCAT* (2nd edition). School of Communication, SFU

Richards, W.D. (1988). *Private Communication*

Richards, W.D. (1988). "FATCAT ... for Thick Data", *Connections* (the Bulletin of the International Network for Social Network Analysis), Vol. **XI**, No. 3.

Richards, W.D. (1986) FATCAT: a different kind of network analysis program, http://www.sfu.ca/~richards/Pdf-ZipFiles/fatman2.pdf

Richards, WD. (1971). An Improved Conceptually-Based Method for the Analysis of Communication Networks in Large Complex Organizations. Presented to International Communication Association, Phoenix, Arizona.

Richards, W.D. and Rice, R. (1981). The NEGOPY Network Analysis Program, *Social Networks*, **3** (3): 215

Richards, W.D. & Seary, A.J. (2000). Eigen Analysis of Networks, *J. Social Structure.* http://www.heinz.cmu.edu/project/INSNA/joss/index1.html

Richards, W. D. & Seary, A.J. (1997) .Convergence analysis of communication networks, in *Organization Communication: Emerging perspectives V*, Ed. G. Barnett, Ablex.

Schwartz, J.E. (1977). An examination of CONCOR and related methods for blocking sociometric data, in Heise, D.R. (ed). *Sociological Methodology* 1977. 255-282. San Francisco: Jossey-Bass.

Seary, A.J., Richards, W.D., McKeown-Eyssen, G. & Baines, C., "Networks of Symptoms and Exposures", (2005) *Structure and Dynamics: eJournal of Anthropological and Related Sciences* (To appear) http://repositories.cdlib.org/imbs/socdyn/sdeas/

Seary, AJ and Richards, WD. (2003). Spectral methods for analyzing and visualizing networks: an introduction. In National Research Council, *Dynamic Social Network Modeling and Analysis: Workshop Summary and Papers*, (209-228). Eds. Ronald Breiger, Kathleen Carley and Phillipa Pattison, Division of Behavioral and Social Sciences and Education. Washington DC. The National Academics Press.

Seary, A.J. & Richards, W.D. (2000). Negative eigenvectors, long paths and p*, Paper presented at INSNA XX, Vancouver BC. http://www.sfu.ca/~richards/Pages/longpaths.pdf

Seary, A.J. and Richards W.D. (1999). Eigenvalue Bounds, presented to International Network for Social Network Analysis, Charleston, S.C. February, 18-21 http://www.sfu.ca/~richards/Pages/insna99x.pdf.

Seary, A.J. and Richards, W.D. (1998). Some Spectral Facts. Presented at INSNA XIX, Charleston. http://www.sfu.ca/~richards/Pages/specfact.pdf

Seary, A.J. & Richards, W.D. (1995). Partitioning Networks by Eigenvectors. Presented to European Network Conference, London. Published in Everett, M.G. and Rennolls, K. (eds). (1996). *Proceedings of the International Conference on Social Networks,* Volume 1: Methodology. 47-58.

Seary, A. (1999) PSPAR: Sparse Matrix Version of STAR. http://www.sfu.ca/~richards/pspar5k.zip

Seary, A.J. (1995) MultiNet for DOS, Presented at International Conference on Social Networks (Sunbelt XV), London, UK

Seary, AJ (1988). *Private communication.*

Senata, E. (1981). *Non-negative Matrices and Markov Chains*. Springer-Verlag.

Simon, H. (1984). Analysis of the symmetric Lancozos Algorithm with reorthogonalization methods, *Lin. Alg. And Appl.* **61**:101-131

Strauss, D. and Ikeda, M. (1990). Pseudolikelihood Estimation for Social Networks, *J. Am. Stat. Assoc.* **85**: 205-212.

Simon,H. (1991). Partitoning of Unstructured Problems for Parallel Processing. *Computing Systems in Eng.* **2** (2/3): 135-148.

Sinclair, A. (1993). *Algorithms for Random Generation and Counting*, Birkhauser

Valente,T.W. And Foreman, R.K. (1998) Integration and Radiality: measuring the extent of an individual's connectedness and reachability in a network, *Social Networks* **20**, 89-105

Walshaw, C. (2000). A multilevel algorithm for force-directed graph drawing, *Proc. Graph Drawing 2000,* Lecture Notes in Computer Science, **1984**: 171-182.

Walshaw, C. and Berzins, M. (1995). Dynamic Load-balancing for PDE Solvers on Adaptive Unstructured Meshes, *Concurrency: Practice and Experience.* **7** (1): 17-28.

Wang,Y.J. and Wong, G.Y. (1987) Stochastic Blockmodels for directed graphs, *J. Am. Stat. Assoc.*, **82**:9-19

Wasserman, S and Faust, K. (1994). *Social Network Analysis: Methods and Applications.* Cambridge: Cambridge University Press.

Wasserman, S. and Pattison, P. (1996). Logit Models and Logistic Regressions for Social Networks I: An Introduction to Markov Graphs and p*, *Psychometrika* **61**: 401-425.

West, D. B. (1996). *Introduction to graph theory*, Prentice Hall, 1996.

Wilson, T.P.(1982). Relational networks: an extension of sociometric concepts, *Social Networks.* **4** (2), 105-116.