

ASPECTS OF STATISTICAL DISCLOSURE CONTROL

A thesis submitted to The University of Manchester for the degree of
Doctor of Philosophy
in the Faculty of Humanities

2011

DUNCAN GEOFFREY SMITH

SCHOOL OF SOCIAL SCIENCES

CONTENTS

FIGURES.....	6
TABLES.....	8
ABSTRACT.....	9
DECLARATION.....	10
COPYRIGHT STATEMENT.....	11
ACKNOWLEDGEMENTS.....	12
THE AUTHOR.....	13
CHAPTER 1.....	14
Introduction.....	14
Basic formulation of the disclosure problem.....	15
Motivation.....	16
Inference.....	17
The Bayesian approach.....	18
Costs.....	21
Summary.....	21
Core chapters.....	23
CHAPTER 2.....	26
Background.....	26
Some basic SDC concepts.....	26
Microdata and aggregate data.....	26
Anonymization.....	27
Categorization.....	28
Linking and matching.....	28
Data divergence.....	29
Data utility.....	30
Sensitivity.....	30
Forms of disclosure.....	32
Identification and attribution.....	32
Assessing risk.....	33
Attribution risk.....	33
Identification risk.....	34
Sample data.....	35
Scenario analysis.....	37
Risk limitation.....	38
Suppression.....	39
Recoding.....	40
Cell Suppression.....	41
Sampling.....	44
Metadata suppression.....	45
Access restriction.....	46
Perturbation.....	48
Data swapping.....	48
Controlled tabular adjustment.....	48
Interval publication.....	50
Rounding.....	50
Barnardization.....	50

Controlled rounding.....	51
Random rounding.....	51
Inference.....	52
Decomposable graphical releases.....	54
The generalized shuttle algorithm.....	57
Bayesian methods.....	60
Summary.....	61
CHAPTER 3.....	64
A Measure of Disclosure Risk for Aggregate Data.....	64
Introduction.....	64
Disclosure Risk.....	64
Attribution risk from low population counts.....	68
Protection against attribution.....	68
A measure of attribution risk.....	72
Single rounded table.....	74
Single rounded table and rounded total.....	75
General table releases.....	79
A comparison of rounding schemes.....	79
Summary.....	81
CHAPTER 4.....	82
A Graphical Modelling Approach to Key Variable Mapping.....	82
Introduction.....	82
Key Variable Mapping System.....	83
Analysis.....	84
Aggregation graphs.....	85
Categorizations and harmonization.....	88
Harmonization graphs.....	91
Analysis.....	96
Summary.....	98
CHAPTER 5.....	100
An Evaluation of Strategies for Matching Population and Sample Units.....	100
Introduction.....	100
Existing measures.....	102
Simple attack strategies.....	104
Matching from population to sample.....	105
Matching from sample to population.....	105
Specific strategies.....	106
Strategy 1. An intruder matches from sample to population using all cells...106	
Strategy 2. An intruder matches from population to sample using all cells...106	
Strategy 3. An intruder matches from population to sample using all sampled cells.....106	
Strategy 4. An intruder matches from sample to population using all sample unique cells.....107	
Strategy 5. An intruder matches from population to sample using all sample unique cells.....107	
Strategy 6. An intruder matches from sample to population using all sample cells with count r107	
Strategy 7. An intruder matches from population to sample using all sample cells with count r108	
Dominant strategies.....	109

Search costs.....	110
Matching from population to sample.....	112
Matching from sample to population.....	113
Cost dominance.....	114
Efficiency.....	116
Sampling without replacement.....	116
Sampling with replacement.....	118
Simulation with real world data.....	118
Summary.....	120
CHAPTER 6.....	123
A Brief Introduction to Dynamic Junction Trees and Model Determination in	
Decomposable Graphical Models.....	123
Introduction.....	123
Bayesian networks.....	123
Markov graphs.....	125
Moralization.....	125
Triangulation and decomposable graphs.....	126
Junction trees.....	129
Dynamic Junction Trees.....	132
Model determination.....	134
Tree rotations.....	137
Markov Chain Monte Carlo Model Composition.....	141
Metropolis-Hastings.....	142
Simulated annealing.....	144
Inference using decomposable graphical models.....	146
Summary.....	147
CHAPTER 7.....	148
A Bayesian Strategy for Matching Population and Sample Units.....	148
Introduction.....	148
Search costs.....	151
Estimation of risks in the absence of population counts.....	153
A Bayesian matching strategy.....	153
The attack strategy.....	154
Results.....	155
Summary.....	161
CHAPTER 8.....	163
A Bayesian Alternative to the Special Uniques Detection Algorithm.....	163
Introduction.....	163
Special Uniques Detection Algorithm.....	164
Classification.....	166
Calibration with the DIS score.....	168
An alternative approach.....	169
Graphical models.....	169
Markov Chain Monte Carlo Model Composition.....	170
Simulation experiments.....	171
One percent sample.....	173
Five per cent sample.....	182
Sensitivity to table sparseness.....	186
1% sample with 16 variables.....	187
5% sample with 8 variables.....	190

Costs.....	194
Risk measures.....	196
Variable level.....	197
Category level.....	197
Discussion.....	198
CHAPTER 9.....	201
Summary.....	201
Motivation.....	201
Core chapters.....	201
Policy implications.....	207
Future work.....	208
Final comment.....	210
APPENDIX 1.....	212
KVMS Application Tutorial.....	212
Introduction.....	212
On start up.....	212
Data entry.....	215
Adding groups.....	223
Analysis.....	226
Saving outputs.....	228
APPENDIX 2.....	230
KVMS Algorithms.....	230
Introduction.....	230
Harmonization graph construction.....	230
Analysis.....	238
BIBLIOGRAPHY.....	241

FIGURES

Figure 1. Matching on key variables.....	29
Figure 2. Data release with a risky cell.....	40
Figure 3. A recoded data release.....	41
Figure 4. Ineffective cell suppression.....	42
Figure 5. A suppressed cell with suppressed marginal counts.....	43
Figure 6. A suppressed cell with preserved marginal counts.....	44
Figure 7. Flow diagram for a data release and analytical request Web service.....	47
Figure 8. Controlled tabular adjustment.....	49
Figure 9. A non-decomposable graph.....	55
Figure 10. A decomposable graph.....	55
Figure 11. A junction tree.....	56
Figure 12. An unreleased 2-way table with released margins.....	57
Figure 13. A dependency.....	58
Figure 14. The dependency graph for the tables in Figure 12.....	59
Figure 15. Number of beneficiaries by count and race.....	66
Figure 16. Number of beneficiaries by count and race.....	67
Figure 17. A 2-way cross-classification with margins.....	69
Figure 18. A conventionally rounded cross-classification with conventionally rounded margins.....	69
Figure 19. Trivial lower bounds.....	70
Figure 20. Trivial upper bounds.....	70
Figure 21. Non-trivial lower bounds.....	71
Figure 22. Non-trivial upper bounds.....	71
Figure 23. System of constraints for rounded detail counts and rounded total.....	78
Figure 24. An aggregation graph for Place of Residence.....	85
Figure 25. An aggregation graph for Age generated using a simple algorithm.....	87
Figure 26. The categorization $\{\{\text{England, Scotland, Wales}\}, \{\text{Northern Ireland}\}, \{\text{Other}\}\}$	89
Figure 27. The categorization $\{\{\text{England, Scotland, Wales, Northern Ireland}\}, \{\text{Other}\}\}$	90
Figure 28. The harmonization of the graphs in Figures 26 and 27, $\{\{\text{England, Scotland, Wales, Northern Ireland}\}, \{\text{Other}\}\}$	90
Figure 29. Harmonization graph for Age.....	95
Figure 30. Matching between sample and population on key variables.....	101
Figure 31. Sampling without replacement cost measure (with $r=3$ for strategies 6 and 7).....	119
Figure 32. Sampling with replacement cost measure (with $r=3$ for strategies 6 and 7).....	120
Figure 33. The Asia network.....	125
Figure 34. The Markov graph of the Asia network with moralization edges (E,B) and (T,L).....	126
Figure 35. A decomposition of the Asia network with triangulation edge (B,L).....	127
Figure 36. A junction tree for the Asia network.....	130
Figure 37. The junction graph for the Asia network.....	131
Figure 38. A dynamic junction tree for the Asia network.....	133
Figure 39. Schematic showing the subgraph induced by two adjacent cliques.....	137
Figure 40. Edge addition using a dynamic junction tree.....	140
Figure 41. Edge deletion using a dynamic junction tree.....	141

Figure 42. Matching between sample and population on key variables.....	149
Figure 43. The probability of a correct match for various strategies and metrics against sampling fraction.....	156
Figure 44. The probability of a correct match for 2 standard strategies and 2 Bayesian strategies against sampling fraction.....	160
Figure 45. The subset lattice for the 3 variables A,B and C.....	165
Figure 46. Predictive performance for DIS-IS with the 1% sample.....	174
Figure 47. Predictive performance for MC3 with the 1% sample.....	175
Figure 48. Predictive performance for simulated annealing with the 1% sample.....	176
Figure 49. Logit rank plot for the 1% sample.....	179
Figure 50. Best model found by simulated annealing for the 1% sample.....	180
Figure 51. Convergence of simulated annealing for the 1% sample.....	181
Figure 52. Predictive performance for DIS-IS with the 5% sample.....	182
Figure 53. Predictive performance for MC3 with the 5% sample.....	183
Figure 54. Predictive performance for simulated annealing with the 5% sample.....	184
Figure 55. Logit rank plot for the 5% sample.....	185
Figure 56. Best model found by simulated annealing for the 5% sample.....	186
Figure 57. Predictive performance for DIS-IS with the 1% sample and 16 variables.....	188
Figure 58. Predictive performance for MC3 with the 1% sample and 16 variables.....	189
Figure 59. Logit rank plot for the 1% sample with 16 variables.....	190
Figure 60. Predictive performance for DIS-IS with the 5% sample and 8 variables.....	191
Figure 61. Predictive performance for MC3 with the 5% sample and 8 variables.....	192
Figure 62. Logit rank plot for the 5% sample and 8 variables.....	193
Figure 63. DIS-IS running times (secs).....	194
Figure 64. MC3 running times (secs).....	194
Figure 65. Simulated annealing running times (secs).....	195
Figure 66. The Analysis tab.....	213
Figure 67. The Forms tab.....	213
Figure 68. The Variables tab.....	214
Figure 69. The shell.....	214
Figure 70. The Edit menu.....	216
Figure 71. The Form editor.....	216
Figure 72. The Variable Editor.....	217
Figure 73. The Aggregation Editor.....	218
Figure 74. The Aggregation Editor with selected categories.....	219
Figure 75. The edited aggregation graph.....	220
Figure 76. The Forms grid with 5 forms added.....	221
Figure 77. The Variables notebook page for Country of Residence.....	222
Figure 78. The Harmonizations notebook page for Country of Residence.....	223
Figure 79. Creating and adding a group.....	224
Figure 80. Selecting all nodes in a group by selecting the group.....	225
Figure 81. Adding a group via the shell.....	225
Figure 82. Confirmation that the new group has been added.....	226
Figure 83. Analysis output.....	227
Figure 84. Querying codes via the shell.....	228
Figure 85. Harmonization graph for Age.....	236

TABLES

Table 1. Simulation results showing, for 1200 randomly generated tables, the banded probabilities of producing a table containing at least one zero given subtraction of n randomly selected units from the population.....	80
Table 2. Joint distribution of population and table counts for the 1% sample.....	180
Table 3. Joint distribution of population and table counts for the 5% sample.....	185

ABSTRACT

Aspects of Statistical Disclosure Control

A thesis submitted to The University of Manchester for the degree of
Doctor of Philosophy
in the Faculty of Humanities

2011

Duncan Geoffrey Smith

This work concerns the evaluation of statistical disclosure control risk by adopting the position of the data intruder. The underlying assertion is that risk metrics should be based on the actual inferences that an intruder can make. Ideally metrics would also take into account how sensitive the inferences would be, but that is subjective. A parallel theme is that of the knowledgeable data intruder; an intruder who has the technical skills to maximally exploit the information contained in released data. This also raises the issue of computational costs and the benefits of using good algorithms.

A metric for attribution risk in tabular data is presented. It addresses the issue that most measures for tabular data are based on the risk of identification. The metric can also take into account assumed levels of intruder knowledge regarding the population, and it can be applied to both exact and perturbed collections of tables.

An improved implementation of the Key Variable Mapping System (Elliot, et al., 2010) is presented. The problem is more precisely defined in terms of categorical variables rather than responses to survey questions. This allows much more efficient algorithms to be developed, leading to significant performance increases.

The advantages and disadvantages of alternative matching strategies are investigated. Some are shown to dominate others. The costs of searching for a match are also considered, providing insight into how a knowledgeable intruder might tailor a strategy to balance the probability of a correct match and the time and effort required to find a match.

A novel approach to model determination in decomposable graphical models is described. It offers purely computational advantages over existing schemes, but allows data sets to be more thoroughly checked for disclosure risk.

It is shown that a Bayesian strategy for matching between a sample and a population offers much higher probabilities of a correct match than traditional strategies would suggest.

The Special Uniques Detection Algorithm (Elliot et al., 2002) (Manning et al., 2008), for identifying risky sample counts of 1, is compared against Bayesian (using Markov Chain Monte Carlo and simulated annealing) alternatives. It is shown that the alternatives are better at identifying risky sample uniques, and can do so with reduced computational costs.

DECLARATION

Some earlier work on adjusting trees to help with a specific inference problem was included in a Masters degree dissertation in 2000 (MSc in Applied Statistics, Sheffield Hallam University). This was the same inference problem addressed in Smith (2001). However, the dynamic junction trees presented in Chapter 6 are different data structures. Dynamic junction trees and their application to model determination in decomposable graphical models has not been previously submitted in support of an application for any degree or qualification, or published in any journal.

No other portion of the work referred to in the thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

COPYRIGHT STATEMENT

- i. The author of this thesis (including any appendices and/or schedules to this thesis) owns certain copyright or related rights in it (the “Copyright”) and s/he has given The University of Manchester certain rights to use such Copyright, including for administrative purposes.
- ii. Copies of this thesis, either in full or in extracts and whether in hard or electronic copy, may be made only in accordance with the Copyright, Designs and Patents Act 1988 (as amended) and regulations issued under it or, where appropriate, in accordance with licensing agreements which the University has from time to time. This page must form part of any such copies made.
- iii. The ownership of certain Copyright, patents, designs, trade marks and other intellectual property (the “Intellectual Property”) and any reproductions of copyright works in the thesis, for example graphs and tables (“Reproductions”), which may be described in this thesis, may not be owned by the author and may be owned by third parties. Such Intellectual Property and Reproductions cannot and must not be made available for use without the prior written permission of the owner(s) of the relevant Intellectual Property and/or Reproductions.
- iv. Further information on the conditions under which disclosure, publication and commercialization of this thesis, the Copyright and any Intellectual Property and/or Reproductions described in it may take place is available in the University IP Policy (see <http://www.campus.manchester.ac.uk/medialibrary/policies/intellectual-property.pdf>), in any relevant Thesis restriction declarations deposited in the University Library, The University Library’s regulations (see <http://www.manchester.ac.uk/library/aboutus/regulations>) and in The University’s policy on presentation of Theses .

ACKNOWLEDGEMENTS

The author would like to thank his supervisors. The author would also like to thank Graham Manwell for feedback on the technical and algorithmic aspects of the work. Credit is also due to those within the open source community whose efforts have enabled the whole of this work to be produced without the use of any proprietary software.

THE AUTHOR

The author possesses the following qualifications at Graduate Diploma level and above.

- Graduate Diploma in Foreign Languages (French)
- Graduate Diploma in Statistics
- MSc Building Heritage and Conservation
- MSc in Applied Statistics (Distinction)

The author has also passed the Chartered Institute of Building Professional Examinations and is a Chartered Statistician.

The author has been involved in research since 1997. Research topics have included Bayesian methods for contaminated land assessment, Bayesian methods for demining operations, statistical disclosure control, and discrete event simulation models for cardiovascular disease.

Most of the research has involved a significant amount of software development. Software has been mainly developed in Python, with some code written in R, C and Fortran. All software development has used open source technologies.

Throughout most of this time an interest in Bayesian methods and graphical models has been maintained. The author has carried out a large amount of mainly unpublished research on efficient belief propagation and model determination algorithms for Bayesian networks and decomposable graphical models. This has involved the development of new data structures as well as new algorithms. Some of this work is presented here for the first time.

CHAPTER 1

Introduction

There are many organizations that hold data on individuals or other organizations. In some cases the data are deliberately collected for planning, academic or marketing purposes. In other cases they are accumulated through the day-to-day operations of an organization. Data related to employees are held by almost every business.

In some cases data are collected for the explicit purpose of subsequent release. Data have been routinely collected on the British population every 10 years since 1801 (apart from 1941) through national censuses. Census data are made available to government departments and other organizations for planning and research purposes. Some census data are also released to the general public, but generally in very less detailed forms.

Survey data are collected by various organizations. They might be put to academic use if collected by scientists. They are more likely to be used for marketing purposes if collected by private organizations such as supermarkets. Scientists and private organizations are less likely to want to share the data they hold, although aspects of the data will often be published in scientific papers or business reports.

The availability of data has the potential to lead to scientific advancement and to better policy making. Some might not agree that all scientific advances are a good thing, and few would argue that government policies are wholly objective and evidence based. But it is the use that private organizations might make of data that suggests that there might be advantages and disadvantages to letting others possess one's personal data. They could be used to benefit the customer, or possibly more likely, to benefit the organization. In the case of supermarket loyalty cards the customer generally receives some small incentive to provide their personal information, and that information is usually supplied willingly. A greater issue is the release of personal information held in databases to individuals or organizations without the consent of the individual concerned.

Statistical disclosure control (SDC) is the practice of limiting the chances of released data enabling a 3rd party to infer confidential information about individuals or organizations through either logical and statistical means.

As a definition this is deliberately open as opinions will vary regarding what could reasonably be considered confidential. It raises questions regarding legality, trust, data utility and the sensitivity of the information contained in a potential data release. Clearly, the illegal sharing of data should be prevented. But even where the sharing is legal there should be some consideration of the trade off between the usefulness of the data and the potential impacts on both the targeted individuals through invasions of privacy and the data holder through a reduction in trust. Trust is very important for organizations, such as governments, who rely on public cooperation in the collection important data. A publicized statistical disclosure would damage that trust. This is where SDC comes in. It attempts to ensure that useful data can be shared for useful purposes without adversely impacting the privacy of the individuals concerned.

Basic formulation of the disclosure problem

In accordance with Duncan et al. (2011) a data holding organization will be referred to as a Data Stewardship Organization (DSO). 3rd parties seeking to discover information will be termed *data intruders*, and the individuals about whom the data intruders wish to discover information will be termed *targets*. Mechanisms by which a data intruder might recover information on targets will be termed *attack scenarios* (Elliot and Dale, 1999).

The basic premise is that a data intruder will attack the data in order to try to recover information on targets. The attack scenario describes both the type of intruder and the strategy employed by the intruder to recover information. For example, an intruder could be a member of the public who knows that a data record relating to a neighbour is contained in a published database. The intruder might attempt to identify the relevant record by matching against known information about the neighbour. A correct match based on a subset of the information in the record allows the remaining information in the record

to be inferred for the neighbour. An alternative scenario might involve a member of a company attempting to infer the salaries of other employees through payroll data.

The role of SDC is to limit disclosure. So it needs to consider alternative attack scenarios and quantify the chances and consequences of a successful attack. In order to help achieve this, measures of risk are often produced. A DSO can calculate the value of the risk measure for a potential data release and use it to inform the decision about whether to release the data. Risk measures can be as simple as basic rules of thumb, or involve detailed calculations to estimate the probability of a disclosure taking place under a given scenario.

Motivation

The motivation for this work is to evaluate methods for SDC that adopt the position of the data intruder. This is not a wholly original idea, and has been the basis of other approaches (Paass 1988) (Mokken et al., 1992) (Elliot and Dale, 1999). But adopting this position does suggest new approaches and bring into question existing approaches.

In part this work concerns what might be called the knowledgeable data intruder. That is, an intruder who understands inferential methods and who is prepared to spend some time and effort attempting to recover sensitive information about targets. The idea is to establish exactly what an intruder is capable of inferring from data, to produce measures of risk that truly reflect what an intruder might infer, and to assess whether existing measures can be improved. Improvement also includes improvements in algorithms for approaches that are time consuming. Some SDC methods are computationally demanding, limiting the size (in terms of the number of records or dimensionality) of data sets that can be risk assessed. This work is primarily concerned with inferences that an intruder might make using published *aggregate data*. These are data that have had detailed information such as names and addresses removed, and which might be published in the form of contingency tables. Nevertheless, alternative forms of data and associated risks and risk measures are discussed briefly in Chapter 2.

Inference

Statistical inference is inductive; sample data are used to make general inferences about populations. For instance, a randomized controlled trial (RCT) might be conducted to assess the efficacy of a new drug. Individuals with a given disease would be randomly allocated into a control group and a treatment group without being told which group they were in. The treatment group would be given the new drug, whilst the control group would be given a placebo. The trial would generally be double-blinded by making sure that those administering the drug / placebo would not be aware of which subjects were receiving the drug / placebo. The efficacy of the new drug would be assessed by comparing some pre-specified outcome measure across the two groups. Typically the observed treatment effect would serve as an estimate of the population effect and a statistical test would be used to assess the weight of evidence that the population effect is non-zero. Thus the sample data are used to make inferences about a population effect.

Inferences that are performed by data intruders are generally deductive; using data to make inferences about individuals rather than the general population. For example, assume released population data for a small level of geography contains a 16 year old widow¹ with a given medical history. An intruder might know a 16 year old widow who lives in the relevant geographical area and be able to match against the published record. As it is population data, a unique match implies a correct match (assuming the data are accurate and up to date). Thus the intruder can associate the medical history with the known 16 year old widow. Note that this is a purely logical, deductive inference.

However, if there was more than one possible match the intruder might not be able to make inferences using logic. There could be two 16 year old widows with identical medical histories (albeit unlikely), in which case logic could be used. The release of sample data would introduce the possibility of unsampled matching records, which might include the correct match. In such cases the intruder might be prepared to make assumptions or apply a probabilistic approach. Inferences would no longer be certain, but might still be of concern to a DSO if they could be made with a high enough degree of confidence.

¹ This is a standard example of a type of individual who is so rare that they might be matched against data records with a high degree of confidence, particularly given a small geographical area. Such an individual known to an intruder might be 'spontaneously recognized' in the data.

Assume the released data were actually sample data. Based on the rarity of 16 year old widows the intruder might infer that the 16 year old widow in the data set is almost certainly the known 16 year old widow. Thus the intruder could be highly confident that the known 16 year old widow has the medical history given in the data. The relevant question for the risk assessor is, “How confident?”. In this case the intruder can be at least as confident as he / she is in the correctness of the match.²

The above example demonstrates the trivial observation that intruder inferences are generally deductive. But it also shows that where wholly logical inferences are not possible the intruder's prior beliefs are relevant. The inference regarding the known 16 year old widow in the sample is a combination of prior belief and deductive inference. The prior belief might be based on personal experience or prejudice, or data analysis. In any case it is an inductive inference.

Whether uncertain inferences are an issue is dependent on context. Even an incorrect, but plausible claim that disclosure has occurred could be damaging to a national statistical agency. So it is clear that, where there is uncertainty, the risk of disclosure will need to be assessed in terms of a combination of inductive and deductive inferences. It is not something which can be addressed purely through logic or through traditional, frequentist statistical methods. It can, however, be addressed by Bayesian methods.

The Bayesian approach

Frequentist probabilities are based on proportions in the limit of repeated trials. If a fair coin is tossed repeatedly, then in the limit (as the number of trials tends to infinity) the proportion of heads will tend to 0.5. Thus the probability of a head resulting from a fair toss of a fair coin is 0.5. Frequentist probabilities are also referred to as objective probabilities.

But there is an alternative school of thought when it comes to probability. Bayesians view

2 Given a correct match it follows logically that the medical history pertains to the target. However, an incorrect match might have the 'correct' medical history. Thus the degree of confidence in the correctness of the match is only a lower bound on the degree of confidence that can be assigned to the 'matching' of the medical history with the target.

probabilities as degrees of belief. Thus there is no requirement for any notion of repeated trials and limiting proportions. Where such a notion exists frequentist and Bayesian probabilities coincide. But a Bayesian would be prepared to associate a probability measure with an event which has never occurred. An important distinction is that different individuals might hold different beliefs and therefore assign different probabilities to the same event. Thus Bayesian probabilities are sometimes termed subjective or personal probabilities.

An example of the distinction would be a coin toss. Given the assumptions of a fair toss and a fair coin a frequentist and a Bayesian would agree on the probability of a head being 0.5. If a fair coin was tossed and covered so that neither the frequentist nor the Bayesian could see it, then the frequentist would hold that the probability of a head was either 0 or 1. The outcome is no longer probabilistic; it is just unknown. Knowing that the coin had been tossed would not affect the belief of the Bayesian in the outcome, so the probability for the Bayesian would still be 0.5.

Belief revision in the Bayesian framework stems from a simple rule of probability known as Bayes rule:

$$P(H|D) = \frac{P(D|H)P(H)}{\sum_H P(D|H)P(H)}$$

The denominator term is equal to $P(D)$ and for continuous H the summation is replaced with integration. Derivation follows simply from equating the two alternative factorizations for $P(D,H)$ under dependence.

As a statement relating probabilities, Bayes rule is not problematic for frequentists. It is the interpretation of $P(H)$ as a prior belief rather than an objective marginal probability that is the issue. But in SDC it is sometimes important to understand how confident a data intruder might be in a particular hypothesis; for instance the hypothesis that a sensitive level of a variable holds for a given target. Assessing this can be achieved by combining reasonable prior beliefs with information that is contained in data.

Going back to the example of the 16 year old widow in the sample we might assume that the intruder's prior probability that the widow in the data is the known widow is 0.99. Assuming that the data are up to date and correct and that the given medical history only applies to 2% of 16 year old widows, then inference could proceed as follows.

The marginal probability that the medical history applies to the known widow is,

$$P(H) = \sum_M P(H|M) P(M)$$

$$P(H) = 1 \times 0.99 + 0.02 \times 0.01 = 0.9902 .$$

where M is a boolean variable for the correctness of the match.

Note that the probability that the history applies to the widow is slightly greater than 0.99 because of the possibility that it would apply even if the match were incorrect.

But we could take a more complex approach to estimate $P(M=True)$ based on the probability mass function for the population frequency for 16 year old widows, $P(F)$, where F is not less than the sample frequency.

$$P(M=True) = \sum_F \frac{1}{F} P(F)$$

$P(F)$ in turn will depend on the sampling fraction and the marginal probability of a population unit being a 16 year old widow. Thus a knowledgeable intruder might introduce more complexity and relevant information to attempt to generate more accurate posterior beliefs.

The point is that adopting the Bayesian view of probability allows the rules of probability to be used to generate the degrees of confidence that intruders might reasonably have in particular hypotheses. It also allows simplifying assumptions to be made and arbitrary sources of information to be incorporated. A naïve intruder might perform the sort of

calculation shown above, generating a fairly naïve posterior belief that the known widow possesses the published medical history with probability 0.9902. A less naïve intruder armed with more information might perform a more complex calculation and arrive at a different result. In either case it allows us to take an attack scenario, some data and to make inferences about the inferences that could be made by intruders. The external (to the data set) information held by the intruder and the methods and resources available to the intruder would form part of the attack scenario.

Costs

As an intruder adopts more complex approaches to making inferences computational costs can increase. Clearly this is also true for risk assessment methods based on discovering the inferences that would be possible to such a data intruder. So algorithms turn out to be very important. An intruder armed with better algorithms might be able to recover sensitive information from data sets that cannot be adequately assessed for risk.

A second aspect of cost is the cost of search. Many attack scenarios include matching data units (individuals or organizations which constitute 'records') and population units (individuals or organizations in the population). The time and effort required to find a match mitigates against some scenarios. But just as for computational costs, if the intruder can find a better way, there are potential problems for risk assessment.

Summary

Statistical disclosure control concerns the trade off between the benefits of making data available and the risks of allowing the disclosure of sensitive information about individuals or organizations. The basic premise is that of an intruder who will launch an attack on released data, and attempt to recover or infer sensitive information. In some cases purely logical inferences can be made. In other cases an intruder will only be able to make inferences with a certain degree of confidence. Depending on the type of DSO, and other factors such as the utility of the data and the inferences that can be made, either might be

of concern.

A computationally and mathematically literate intruder could conceivably launch attacks that would not be feasible for the average intruder. It is important to be able to limit the possibilities of disclosing information to such intruders, as well as protecting against disclosures under more basic attack scenarios. It could be the case that standard disclosure limitation methods and risk metrics do not adequately reflect the risks posed by the more knowledgeable intruder.

However, it is not simply the case that protecting data against a knowledgeable intruder will guarantee protection against a lay intruder. Perceptions of disclosure can be an issue, and a lay intruder might make a claim of disclosure where the knowledgeable intruder would realize that inferences could not be made with high degrees of confidence. A DSO might be concerned about such claims where they might have *prima facie* plausibility. Simply pointing out that the inference is incorrect could be problematic, as it might provide information that would lead to a disclosive inference; for instance, pointing out that a claimed match is incorrect when there are only two possible matches. A safer defence is to demonstrate the implausibility of the claim, which is most easily achieved by adopting the position of a knowledgeable data intruder and attempting to perform the same inference.

Another aspect of protection is the time and effort that would need to be employed by an intruder in order to effect a disclosure. A very costly attack might be unlikely to be launched. This might be computational time, which raises the issue of using high performance algorithms. It could also mean, for example, the time required to search a population for an individual who matches a given database record. These are issues that can change over time, with increased computational power and better algorithms, and with new data sources and attack scenarios. SDC approaches should take this into account. Once data are released they cannot be taken back, and the risks of disclosure might be expected to increase over time.

It seems reasonably clear that assessing the inferences that could be made by a knowledgeable data intruder is best achieved by placing oneself in the position of the data intruder. The gamekeeper should turn poacher (at least temporarily). This is the basic

position taken in this body of work. Some existing scenarios and risk assessment approaches are assessed, and new scenarios and assessment approaches are proposed. Some of the work focuses quite heavily on computational issues. This is important as it allows for better assessment of risks. The computational issues that face an intruder are equally as important for the risk assessor.

Core chapters

The core chapters are a collection of relatively self-contained contributions to SDC. However, excessive repetition is avoided by including material what would be common to Chapters 7 and 8 in Chapter 6. This also allows the data structures and algorithms to be explained in some detail. Without such detail the advantages of the precise implementation would not be appreciated, and an enormous amount of work would be hidden “under the hood”. In each chapter the work required a substantial amount of software development. The code is not included in appendices as it is being continually developed, and for reasons of space.

Chapter 2 provides a general review and introduction to the SDC methods and concepts that are relevant to the subsequent chapters. It describes different forms of data and different types of disclosure. This is a fairly gentle introduction to cover basic ground.

Chapter 3 describes a risk measure for tabular data which is based on a particular kind of attack scenario where an intruder has knowledge of the data units. This knowledge can be exploited to make inferences about unknown data units. It is shown how the measure can be calculated for simple forms of data release, and how it can be generalized to more complex forms of data release.

Chapter 4 describes novel graphical methods that can be used to assess possibilities for matching across the data sets within a data environment (set of data sets). It builds upon existing work carried out at the University of Manchester (Elliot et al., 2010). But it adds much in terms of mathematical definitions, the use of novel data structures and efficient algorithms.

Chapter 5 discusses strategies for matching sample units and population units. Various scenarios are discussed and probabilities of success are derived. Again this builds on existing work (Smith, 2006), but this work generalizes it and demonstrates that certain classes of attack scenario dominate others in terms of the probability of success. The various strategies are also assessed in terms of search cost and similar dominance relationships are shown to exist.

Chapter 6 covers some theory and algorithms relating to graphical models that are required for a detailed understanding of subsequent chapters. Specifically it describes the use of a non-standard type of graphical model to manage the Markov Chain Monte Carlo and simulated annealing model determination schemes used in Chapters 7 and 8. The graphical model has been described once before in the literature (Smith, 2001) but not in such detail, and not in the context of model determination. The computational benefits over existing approaches are described. Computational cost is an extremely important issue, which justifies a distinct chapter to cover the detailed implementation.

Chapter 7 develops a Bayesian matching strategy and compares success probabilities and costs with the non-Bayesian strategies described in Chapter 5. It is shown that the Bayesian approach offers much higher probabilities of a successful match than any of the non-Bayesian strategies.

Chapter 8 describes Bayesian alternatives to the special unique detection algorithm (SUDA) approach for assessing the risk from sample counts of one (Elliot et al., 2002) (Manning et al., 2008). Both Markov Chain Monte Carlo and simulated annealing approaches are described and compared against SUDA. They are both shown to offer improved estimates of risk. Simulated annealing is shown to be much quicker at generating risk measures and to produce more reliable results than Markov Chain Monte Carlo. Potential category level, variable level and file level risk metrics are described. Possible improvements and alternatives to the Markov Chain Monte Carlo approach are discussed.

Chapter 9 summarizes and justifies the work. Future work and policy implications are discussed.

Appendix 1 contains a tutorial for the software developed using the algorithms detailed in

Chapter 4.

Appendix 2 contains a detailed exposition and analysis of the algorithms contained in Chapter 4.

CHAPTER 2

Background

Data intruders and attack scenarios were discussed in Chapter 1. A data intruder could be anyone who has access to released data. An attack scenario is a description of how a disclosure might take place. It describes a type of intruder and how the intruder might discover or infer sensitive information about a target. In some cases an attack scenario might be detailed enough to describe the rules of a game and a strategy for winning the game. In this game-theoretic sense a strategy is a complete set of rules that will be followed by an intruder in an attempt to infer sensitive information.

The remainder of this chapter outlines concepts and theory that arise in the core chapters. It is not a comprehensive review of SDC. In particular, it omits much relating to microdata as the core chapters deal almost exclusively with aggregate data. See Duncan et al. (2011) for a comprehensive, and in parts, more detailed review of SDC.

Some basic SDC concepts

Before moving on to disclosure control methods and risk measures it would be sensible to cover some basic ideas relating to SDC. Some would be familiar to most readers, but some are specific to SDC. In effect, this section is a basic glossary, with some discussion, covering terms that will be met later in the document.

Microdata and aggregate data

In SDC there is usually a distinction made between microdata and aggregate data.

Microdata are in the form of a collection of individual records, whereas aggregate data are

often in the form of a contingency table containing a count for each possible combination of variable values. Aggregate data are also called tabular data. A microdata file and an aggregate table could contain exactly the same information.

A microdata file might contain personally identifiable information such as names and addresses. It might also contain variables on a continuous scale with variable values specified to high levels of precision. Thus, in a microdata file it would not be unusual to find many distinct or *unique* records, particularly if records contained a large number of variables.

In contrast, data that are provided in aggregate form generally do not include variables on a continuous scale or that have large numbers of possible values (such as names and addresses). Contingency tables are parsimonious representations of data sets covering a small number of categorical variables, and in SDC that is a reasonable description of the type of data contained in most aggregate tables.

Specific SDC methods will usually be predominantly associated with either microdata or aggregate data. That does not necessarily imply that they cannot be applied to data in the alternative form. For instance, it is a trivial exercise to convert aggregate data into microdata and generate risk measures designed for microdata.

Another form of data is magnitude data³, but it is not relevant to this work and will not be described in detail.

Anonymization

A common practice for DSOs such as national statistical agencies is to *anonymize* data before release. For most research purposes *formal identifiers* such as name and address are

3 Magnitude data is in the form of totals, e.g. turnover. The disclosure risk is that a published industry total would enable a large firm to put upper bounds on the turnover of competing firms by simply subtracting its own turnover from the industry total. The (n, k) rule (Willenborg and de Waal, 2001) specifies that if any n units contribute more than a proportion k of the total, then the data are insecure. There is also a $p\%$ rule, where data are considered to be insecure if they would allow any respondent values to be estimated within $p\%$ of their true value.

not required. So DSOs can be in the fortunate position that the data that would allow people to be easily identified are of least worth to the end user. Removal of obvious identifiers is usually termed *anonymization*, although the term de-identification is also sometimes used.

Categorization

Continuous variables can take an infinite number of values. In principle, each individual in a data set would have a distinct value for a continuous variable. In practice these values are not measured or recorded with infinite precision, so they are more like categorical variables with a very large number of levels. If recorded consistently across datasets they can offer the same possibilities for *record linkage* as formal identifiers. Therefore DSOs routinely *categorize* such variables. Categorization is the process of converting a continuous variable to a categorical scale. This would typically be done for variables such as age where 5 or 10 year age bands might be used.

Linking and matching

Linking data sources can provide much more detailed information on an individual than the individual data sources. Two records from distinct microdata files containing common variables could be used to construct a single record containing all the information in the individual records. Consider the case where the matching is done on a National Insurance number. One record might contain identifying variables such as name, address and birth date. The second record might contain information on income and employment status. Such matching on formal identifiers might happen for bona fide administrative or research purposes. An intruder will rarely have such an opportunity as the DSO will have removed them before releasing the data, so matching will be performed using a different set of variables. This set of variables is sometimes referred to as a *key* or as a set of *key variables*.



Figure 1. *Matching on key variables*

In some cases it might not be possible to establish that a match on the key variables is actually the correct match. There might be more than one match, and some might be more plausible than others given the resulting fuller records. But probabilistic record linkage (see for example Fellegi and Sunter, 1969) might enable a sufficiently high probability of a correct match that it is considered to be an issue by a DSO. Of course, it might be that all the possible matches contain the same values for a sensitive variable. In this case the inference regarding the sensitive variable can be made with certainty even though any given match is likely to be incorrect.

Data divergence

Data-world divergence (Elliot and Dale, 1999) is the degree to which recorded data are different from the correct data. Sometimes variable levels are wrongly coded, and sometimes people lie about personal information. So there is usually a degree of uncertainty over the correctness of data. Data-world divergence is generally ignored for the purposes of SDC.

Data-data divergence (Elliot and Dale, 1999) is the degree to which data contained in different data sets is inconsistent. Parallel divergence describes the situation where data-world divergence exists with respect to information, but the information is still consistent across different data sets. This might arise where an individual lies consistently.

Data-world divergence will tend to reduce the risks of identification and / or attribution (depending on which variables are affected). Data-data divergence reduces the chances of correctly matching records in distinct data sets if it affects key variables.

The goal in data collection is to have zero divergence, so it would be curious to rely on data divergence for security. Countering a claim of disclosure with the assertion that the data cannot be relied upon might be a little embarrassing for the data holder (even if it happens to be true).

Data utility

This is a fairly general term. It is clearly related to data-world divergence. In fact a high degree of divergence might imply negative utility, as inferences based on the data might be wholly unreliable and potentially misleading. It also depends on the way in which the data are to be used. Methods to limit the chances of disclosure often have a negative impact on data utility. They might be chosen so as to have minimal impact for certain use cases by preserving certain statistical qualities of the data. Data utility is not explicitly considered in the core chapters, but it is important to remember that the risk-utility trade off is at the heart of all decisions regarding the release of data. The trade off is explicitly considered in papers such as Duncan and Stokes (2004).

Sensitivity

An overriding issue in SDC is exactly what constitutes sensitive information. There are national and cultural differences, and even within a small defined population there will be significant variability in what people consider to be sensitive. Some people are happy, for example, to hand over much personal information when applying for a supermarket loyalty card. Others might question why lifestyle factors such as smoking status are required in order to even discuss taking out an Individual Savings Account (ISA).

As well as the issue of subjectivity it is clear that some levels of some variables are more sensitive than others. Very often it will be the more unusual levels of a variable that are considered to be most sensitive. Very high earners might be very keen that their earnings are kept confidential to avoid being labelled fat cats. Lower earners might be less

concerned about their earnings being known. The fact that somebody has a rare disease is likely to be much more sensitive than the fact that they do not.

Sensitivity can change over time. In the early 1960s homosexuality was a criminal offence in England and Wales. By the 1970s the threat of criminal prosecution was no longer present, yet many homosexuals would have considered their sexual orientation to be sensitive information. Today this is probably less so, although relatively few DSOs would consider sexual orientation not to be sensitive.

Sensitivity can change with age, and be different across different age groups. The sensitivity of a person's age can depend on age itself. The young and the old are often quite proud to be young, or to have lasted so long. Those in middle age are more likely to want to keep their age to themselves. The young are more used to sharing information with 'friends' that they have never actually met and with the world at large through social networking. The elderly are more likely to consider sensitive that which they considered sensitive in their youth.

In SDC it is common to classify variables as sensitive or not sensitive. This is clearly an over-simplification that is made for practical reasons. There is often also an implicit assumption of a monotone relationship between the amount of information an intruder might have and the total sensitivity of that information. This does not necessarily follow. Imagine that an intruder has discovered that a target has recently had an HIV test. That could be considered highly sensitive as the intruder might infer that the target is a member of a high risk group. Additional information regarding, say, sexual orientation might lead the intruder to believe that the target is an intravenous drug user. On the other hand, information that the target worked in a medical role that required regular HIV tests would drastically reduce the intruder's confidence that the target was in a high risk group. In this example it is the joint information that matters rather than a single level of a single variable. In fact the sensitivity really relates to the inferences that the information allows regarding sexuality or intravenous drug use, variables upon which the DSO might hold no direct information.

So sensitivity is subjective, varies over time and can change quite dramatically when national or cultural boundaries are crossed. Ideally sensitivity should be considered jointly,

as it is not necessarily the case that more information is more sensitive. Nevertheless, simplifying assumptions and generalizations are often made in order to produce workable SDC methods. In general, the specification of sensitivities must be left up to the DSOs. In many cases this simply involves the classification of variables as sensitive or not sensitive. The proper handling of sensitivity is a much under-researched area of SDC. All the methods discussed in the core chapters adopt the standard convention of simply classifying certain variables as target variables. Although alternatives are not discussed in the core chapters, a potential approach would be to take the joint information available to an intruder and generate posterior probability distributions over a set of variables containing sensitive categories. Model determination and inference approaches that would facilitate this are described in Chapter 6.

Forms of disclosure

Identification and attribution

Most SDC methods are primarily concerned with identification. Identification is the linking of a record in a data set with an individual within a population. It does not necessarily imply that the person performing the identification learns anything new about the identified individual. All the information within the data might be needed to perform the identification, leaving nothing left to infer. Alternatively, the inferred information might be already known to the intruder. An obvious example of disclosure-free identification would be somebody recognising their own record, *self-identification*. Yet some organizations might wish to limit the chances of even this possibility, as it could potentially lead to a publicized (yet spurious) claim of disclosure, or simply the perception that disclosure was a possibility.

Attribution is the association of a previously unknown variable level with an individual. So it is paradigmatically disclosure. In principle, attribution can take place without identification. A release of population data might show that all people living in a particular postcode area are unemployed, and thus allow the logical inference that any individual known to live in that area is unemployed.

It is clear from the two examples above that identification does not necessarily imply attribution, and attribution does not necessarily imply identification. However, in many scenarios they do occur together. Being able to identify an individual from the detailed information in a microdata record allows all the previously unknown information in the record to be inferred. The properties of identification and attribution are discussed in more detail in Chapter 3.

Assessing risk

In SDC we are faced with a number of dichotomies:

- Perceived risk v. actual risk
- Identification v. attribution
- Sample data v. population data
- Microdata v. aggregate data
- Logical inferences v. probabilistic inferences

The last pair are sometimes referred to as “exact” and “approximate” inferences.

So there are potentially many different configurations that could require alternative approaches to risk assessment. This is compounded by the fact that we might want to produce measures of risk for individual records, cells in aggregate data, variables or complete microdata files / aggregate tables. This work does not consider all the various possibilities, so consideration will be limited to those that are relevant to the core chapters.

Attribution risk

Attribution risk is discussed in some detail in Chapter 3, so will not be dealt with at length here. In essence attribution occurs when an intruder can establish that a target belongs to a

group which share some common characteristic. The group does not need to be of size 1. This was originally discussed in U.S. Department of Commerce (1978). Inference from the point of view of an intruder would typically involve a released data set that is known to contain a target. Conditioning on known variable levels would allow the intruder to associate the target with smaller groups within the data set. Any shared characteristic could be inferred for the target. That includes any variable levels, or combinations of variable levels, that apply to no individuals in the group. The crucial thing to grasp is that it is the missing combinations of variable levels that enable exact attribution in population data. In aggregate data these correspond to zero counts. Without zeros an intruder could condition on any set of variable levels and would still be left with a reduced population containing individuals with every possible combination of levels on the unconditioned variables.

In situations where probabilistic inferences might be of concern, then small and large frequencies would represent risks. If 95% of the individuals in the relevant group shared a characteristic, then an intruder might be 95% confident that the characteristic is possessed by the target. A knowledgeable intruder might also condition on known information about the intruder relating to variables outside the data set. This would not produce a reduced population as would conditioning on a variable within the dataset. But it would place the target within a notional subpopulation that might be believed to have different characteristics. This could revise the degree of confidence either up or down.

Chapter 3 addresses attribution risk and develops a risk measure for aggregate data. The measure can also be applied to microdata. It also allows for an assumed level of intruder knowledge regarding the other data units and can handle released data that have been distorted to limit disclosure risk.

Identification risk

A lot of the work on identification risk concentrates on unique records. Uniqueness in released population data implies a risk that an intruder might associate a record with a unique set of characteristics with a target sharing the same characteristics. Any other information in the corresponding record would apply to the target. That is, it might lead to

attribution, which is where the real disclosure takes place. It might be that there is no additional information in the record to infer, or that the information is not sensitive. Nevertheless, it could still give rise to a claim of disclosure which could be damaging to the DSO.

Benedetti and Franconi (1998) noted that the probability that a match from the population to a sample unit is correct is simply $1/F_j$ where F_j is the number of matches in the population. Thus a suitable record level risk measure might be the probability of a correct match, $1/F_j$, where F_j is the population frequency for the cell with index j in the aggregate table marginalized to the key variables. This implies that population uniques (with respect to the key variables) pose the greatest risk. However, it does not distinguish between different population uniques. These could be distinguished by considering the sensitivity of the information contained in the non-key variables, although the measure would then relate more to attribution risk. A more interesting problem is assessing identification risk in sample data.

Sample data

Releasing sample data essentially eliminates the possibility of exact attribution (attribution via logical inference). But the intruder can still make probabilistic inferences using the released data. Essentially the intruder can use the sample data to make inferences about the data generating process and make subsequent inferences regarding targets on the basis of fitted models. With small area geographic level aggregate data the intruder could pool the data for model determination purposes, and then combine the fitted model(s) with the data in a specific geographic area to make inductive inferences about specific targets in that area. A risk assessor could use the same approach to generate metrics based on posterior beliefs regarding sensitive inferences. This outline approach bears some similarity with methods employed in both Smith and Elliot (2003) and Smith and Elliot (2005). The measure proposed in Chapter 3 does not go this far, but does provide a metric that can be used with sample data.

Apart from the case of structural uniques, exact identification is not possible with sample

data. The following discussion applies to non-structural uniques. Identification risk might still be significant problem with sample data (see Chapter 7). Just as with exact identification, most of the focus is on uniques. But with sample data a match from the population against a sample unique might not be correct.

Uniques in released population data imply that a match will be correct, but in sample data the real relevance of uniqueness in the sample is in what it tells us about the population frequency. Firstly, the sample frequency f_j is a lower bound on F_j . Therefore $1/f_j$ provides an upper bound on the probability of a correct match. Only sample uniques can offer a probability of a correct match of greater than 0.5. Secondly, a sample unique might suggest that F_j is low, giving a higher probability of a correct match than a higher sample frequency. Uniques might be of particular concern to a DSO regarding approximate identification where the record contains an unusual combination of attribute levels. In this situation there is the issue of spontaneous recognition, as well as the intruder's prior belief producing a high posterior belief in a match being correct. The same advantages and disadvantages apply to the $1/F_j$ measure, although the risks will not generally be the same for all sample uniques.

File level measures for identification risk that have been proposed have been as simple as the proportion of population units which are population unique (Bethlehem et al., 1990) and the proportion of sample uniques which are also population unique (e.g. Fienberg and Makov, 1998). Other measures have been more closely linked to specific attack scenarios. For instance, an intruder might adopt the strategy of searching the population for a match against any sample unique. On the other hand, the intruder might select a particular sample unique for matching purposes. These strategies give rise to different probabilities of success. These probabilities can be used as risk metrics, and first appear in Skinner and Elliot (2002). These, and other strategies are investigated in Chapter 5. Some strategies are shown to dominate others in terms of the probability of an intruder achieving a correct match. The necessary search costs are also considered, as high search costs might render a scenario / strategy implausible.

When population counts are known to the DSO, then measures of the type discussed above can be calculated. But when a DSO is considering data from sample surveys the population frequencies are unknown. Estimating the $1/F_j$ from the data is one approach to generating a

suitable record level risk metric. This is the approach taken in Elamir and Skinner (2006) who used log-linear modelling, and in Forster and Webb (2007) who employed a Bayesian approach using Markov Chain Monte Carlo. Both these papers also discuss file level metrics.

The DIS formula (Elliot and Skinner, 2002) is a consistent estimator of the probability of a correct match given a unique match when matching from population to sample (that is, the strategy of searching the population for a match against any sample unique),

$$\hat{\theta} = \frac{\pi n_1}{\pi n_1 + 2(1 - \pi) n_2},$$

where π is the sampling fraction and n_1 and n_2 are the counts of 1 and 2 respectively in the sample table.

Other risk metrics are presented in Elliot et al. (2002) and Manning et al. (2008). Elliot (2003) proposes a metric that is hybrid of the metric in Elliot et al. (2002) and DIS. These are discussed in some detail in Chapter 8.

Chapter 7 shows that a Bayesian approach can offer very high probabilities of a correct match, even for small sampling fractions. This brings into question whether the above metrics are appropriate. Chapter 8 compares Bayesian approaches to identifying risky cells with those in Elliot et al. (2002), Elliot (2003) and Manning et al. (2008). Simulations show that the Bayesian methods offer improvements in both classification and execution times.

Scenario analysis

Scenario analysis (Elliot and Dale, 1999) is a method for assessing potential forms of attack. It assesses things like means, motive and opportunity in an effort to quantify the probability / plausibility of different forms of attack. Elliot and Dale (1999) also discuss the concepts of key and target variables. The set of key variables is essentially the

intersection of the set of variables contained in a data set and the set of variables known to an intruder with regard to a given target under a given attack scenario. The target variables are the ones classified as sensitive. They discuss how key variables might be identified by considering publicly available information (e.g. telephone directories), personal knowledge and organizational databases (commercial and governmental).

Elliot et al. (2010) describe a more formalized approach to the identification of key variables and the risks associated with record matching; data environment analysis (DEA). Data sets, such as those described in Elliot and Dale (1999), that are available to an intruder are identified. This constitutes the *data environment*. Possibilities for record matching across data sets within the environment are investigated via a spreadsheet application, the Key Variable Mapping System (KVMS). Chapter 4 describes the method in more detail. It also significantly extends the approach, developing new data structures and algorithms. More information on the software implementation is contained in Appendix 1.

The identification of attack scenarios and key / target variables is a very important part of disclosure risk limitation. The values of risk metrics often depend on which variables are classified as key variables and target variables. A good metric will relate directly to a particular form of attack. In the discussion of identification risk with sample data it was shown that there were a number of possible risk metrics stemming from different attack scenarios. They inform specific SDC risk limitation methods that generally fall into two categories; suppression and perturbation.

Risk limitation

There are different approaches for limiting risk in microdata and aggregate data, although in many cases the same methods can be used for both microdata and aggregate data. If necessary the aggregate data can be converted to a list of records, risk limitation applied, then a conversion back to aggregate form carried out for release. The equivalent might be achieved by the use of an alternative algorithm using the aggregate data. The following concentrates on methods for aggregate data as these are much more relevant to the core

chapters. Methods for microdata are discussed where they provide particularly good examples of the following types of risk limitation, or where the microdata algorithm is significantly easier to follow than the equivalent algorithm for aggregate data.

Suppression

Duncan et al. (2011) describe suppression as the “denial of data instances”. They discuss the deleting of records and variables in microdata, which they term *record suppression* and *attribute suppression*. Record suppression might be undertaken to remove unusual individuals who might be easily identified. Attribute suppression might be used to remove formal identifiers, key variables or sensitive variables. Thus, what is normally referred to as “anonymization” is actually a form of suppression. And each time a DSO decides to release aggregate data on a subset of the variables that they have information on, it is suppression of the unreleased variables.

Essentially, suppression is the act of not releasing information, and that information can be quite arbitrary. It is not limited to records and variables. Statistical outputs might be suppressed in order to prevent certain inferences regarding the analysed data. The total earnings within an industry might be suppressed if there were only 2 companies involved, to prevent each discovering the earnings of its rival by simply subtracting its own earnings from the total.

However, it is not the case that suppression of data prevents intruders from making inferences about the suppressed data. So it is important to understand how well suppression actually protects sensitive data, whether or not it was the sensitive data that were suppressed.

Suppression contrasts with perturbation, which is the act of changing data before it is released. Although they seem to be distinct concepts there are SDC methods which employ both. For instance, sensitive data might be suppressed and then replaced by values generated using multiple imputation methods (see for example Reiter, 2005). The resulting *synthetic* data are then released. As the released data contain no missing values it would

probably be sensible to think of this approach as perturbation. That is the distinction made here. If data or information is not released, then it is suppressed. If it is released having been subjected to some form of distortion, then it is perturbed.

Recoding

Recoding is the practice of changing the level of detail on a variable. A continuous variable might be made into a categorical variable. For instance income might be given only in terms of 'low' and 'high' levels. A categorical variable might be collapsed into a smaller number of categories, say 5 year age bands being collapsed into 10 year bands. Bottom-coding and top-coding are terms used to describe recoding into 'less than' or 'greater than' categories. Recoding is sometimes called aggregation or *table redesign*. Certain cell values are suppressed, but aggregated values are published.

0	3	0	1	4
0	1	0	0	1
4	1	0	2	7
1	0	7	1	9
5	5	7	4	21

Figure 2. *Data release with a risky cell*

Figure 2 shows a possible data release. The count of 1 in the second row and second column might be considered to be too risky. For instance, the second level of the row variable might be unusual, enabling identification of the relevant individual for whom the second level of the column variable could then be inferred. Recoding could be used to eliminate this particular possibility.

0	4	0	1	5
4	1	0	2	7
1	0	7	1	9
5	5	7	4	21

Figure 3. *A recoded data release*

Recoding by aggregating the first two levels of the row variable disguises the presence and location of the risky *unique*. Figure 3 shows the recoded table. An alternative would have been to aggregate the second and third levels of the row variable. In fact there are often many choices that could be made. Some recodings will generally make more sense than others. For instance, if the row variable was on an ordinal scale it would be unusual to aggregate non-adjacent levels. Alternative recodings will have different effects on data utility.

Note that it is often the practice to also release smaller marginal tables, or simply the table total. This does not actually provide more information than contained in the detail table over all the variables, as the marginal tables can be derived by summing across variables. This is also true after recoding. But some disclosure control methods do not treat detail cell counts (counts in the full table) and marginal cell counts in such a consistent manner.

Cell Suppression

Masking is often carried out on aggregate data. Certain combinations of attribute values are considered to be too risky because of the potential for identification or attribution. Thus the value for that combination of values will be suppressed. In outputs the count will simply be replaced by a symbol such as an asterisk. Consider a contingency table for the variable 'marital status', 'age' and some sensitive third variable 'X'. If 'widow' is included as a level of marital status and 'age' contains a category 'under 18' then it is very likely that the

number of under 18 year old widows will be very low. If the table is for some relatively small level of geography, then any 18 year old widow is likely to be unique, relatively easily identified and disclosure on the sensitive variable X will take place. One option would be to remove X by summing across its categories, producing a table for only marital status and age. An alternative would be to only mask the counts that enable disclosive inferences to be made. This could be achieved by suppressing the counts for 'widow' and 'under 18' and each level of X. Great care must be taken in avoiding disclosure via masking. It is tempting to think that disclosure could be avoided by only suppressing the count for 'widow' and 'under 18' and, say, the single sensitive level of X. However, if the counts for 'widow' and 'under 18' and the other levels of the sensitive variable were all zero, then any 18 year old widow in the population would have to be associated with the suppressed value of X. Thus suppression on additional cells is required to make the data release safe.

When cell suppression is performed it is necessary to ensure that suppressed counts cannot be recovered by procedures as simple as subtracting the sum of all the published detail cell counts from a published table total. Thus complementary (secondary) cell suppressions might need to be performed. Finding a set of complementary suppressions which provides enough protection whilst minimising the impact on data quality is known as the *complementary cell suppression problem*.

0	3	0	1	4
0	*	0	0	1
4	1	0	2	7
1	0	7	1	9
5	5	7	4	21

Figure 4. *Ineffective cell suppression*

Figure 4 shows an attempt to mask the risky cell in the previous example. It is clearly ineffective given the published margins.

0	3	0	1	4
0	*	0	0	*
4	1	0	2	7
1	0	7	1	9
5	*	7	4	*

Figure 5. *A suppressed cell with suppressed marginal counts*

Suppression of all the relevant margins, including the table total, is enough to disguise the true value of the risky unique in the detail table. Of course, a known strategy of only suppressing risky unique cells would give the game away. So there are also issues regarding how secure an SDC method is given the possibility that the precise details might be leaked, or might be inferred from a large number of released tables.

An intruder can in principle place bounds on the detail cell counts given the information contained in marginal counts. In Figure 5 there is a lower bound of 0 implied by the nature of the data. There is no finite upper bound.

0	*	0	*	4
0	*	0	*	1
4	1	0	2	7
1	0	7	1	9
5	5	7	4	21

Figure 6. *A suppressed cell with preserved marginal counts*

Figure 6 shows one possibility, of several, for suppressing detail cells so that the risky cell is protected and marginal counts are preserved. But in this case the suppressed cell cannot take arbitrary non-negative integer values. It can only be a 0 or a 1 given the row total of 1. In fact, there are only 2 possible detail tables given the data release in Figure 6. Margins have been preserved, meaning univariate analyses can be performed with complete data. But the level of protection is lower. A little extra information such as a known, or even structural, 0 in cell (2,4) would allow the exact cell counts to be recovered.

Alternative cell suppression schemes will also have different impacts on data utility. Finding a scheme that provides adequate protection whilst minimising loss of data utility can be achieved using integer linear programming (ILP) methods. But the problem is NP hard and can be computationally demanding for larger problems (Fischetti and Salazar, 1999; Salazar, 2008).

Sampling

Yet another form of suppression is sampling. Rather than releasing population data a random sample is released; that is, the non-sampled units are suppressed. This can be effective at reducing the chances of disclosure taking place without significantly impacting on data quality. The samples of anonymized records (SARs) from the UK census can be very large, useful data sets. Matching is problematic because the correct matching record

for a target might not have been sampled. But even in this case it might be possible to match against a highly unusual set of attribute levels with some confidence. There might well be only one individual in the country who lives in London and has the job title 'mayor'. Although the samples of anonymized records do not contain the level of detail necessary to identify the Mayor of London, the example does serve to demonstrate the possible existence of structural ones in samples. Of course, with sampling the data utility is affected in predictable ways and standard analytical methods can be used with the released data. This is clearly not the case with methods such as cell suppression, where the analyst would need to handle the missing values.

Metadata suppression

It has already been pointed out that knowledge of a disclosure limitation policy can be useful to an intruder. It might help the intruder to infer narrow feasible ranges for suppressed counts, or even recover cell counts exactly. For instance, a very naïve policy of not releasing smaller margins and simply suppressing ones in the detail table would be pretty ineffective. Even without knowledge of the scheme it might be inferred from the fact that no released tables contained any 1s. The same is true of the perturbation methods that are discussed in the next section. Knowledge of the parameters of these schemes would aid an intruder in producing feasible ranges, or even credible intervals / posterior mass functions for the cell counts. Although suppression of metadata is an option for the DSO, it is inherently risky.

Kerckhoff's principle is the principle that a cryptosystem should be secure even if everything about the system (except the key) is public knowledge. This is a term that will not often be found in the SDC literature. An SDC analogy would be that any suppressed or perturbed released data set should be safe even if the data intruder were to find out the details of the protection scheme. Relying on keeping a scheme secret presents an enormous risk. Once data are released into the public domain they cannot be unreleased, and it would be catastrophic to find that large amounts of data that had been considered safe had suddenly become irretrievably compromised because of leaked metadata.

It might seem reasonable to keep schemes secret as it offers an extra layer of protection. This is the case with military grade cryptography systems. But in SDC we have to balance safety with data utility. There is an argument that if the data are safe under the assumption that an intruder knows everything about the system other than original data, then that information should be made available to analysts. *Security through obscurity* is an option for SDC and it is used. But its use is questionable. It is not only the possibility that the details of a protection scheme could be leaked that needs to be considered. We also need to consider the inferences that a determined intruder could make regarding the details of the scheme from examining a large number of released tables generated by the scheme. Security through obscurity is inherently risky.

Access restriction

The degree of data suppression might vary across different sections of the population. Data might be released to those where the utility of the data is greater or the risk of disclosure is lower. The risk-utility trade off can be considered in terms other than for a release to the general public.

A physical environment might be provided where approved users could analyse the data on a computer reserved for this purpose. Measures would be taken to ensure that raw data could not be copied and removed, although the analyst at the computer would be able to see the data.

Another option would be to set up a data server that would only release data that were deemed to be safe according to some criterion. Users with higher levels of trust would have access to more data. The cumulative data released to an individual would be recorded so that a data request could be judged based on the accumulated risk / utility.

An alternative would be to provide a set of tools for analysing the data, but without allowing the raw data to be seen by the analyst. The types of analysis might be restricted and the precision of outputs might be limited or even perturbed. Perturbation would have to avoid changing the substantive conclusions of the analysis. Standard data analysis

procedures such as residual model diagnostics might require some perturbation of the residuals.

Figure 7 shows flow diagrams for a notional web service allowing both data releases and analytical requests.

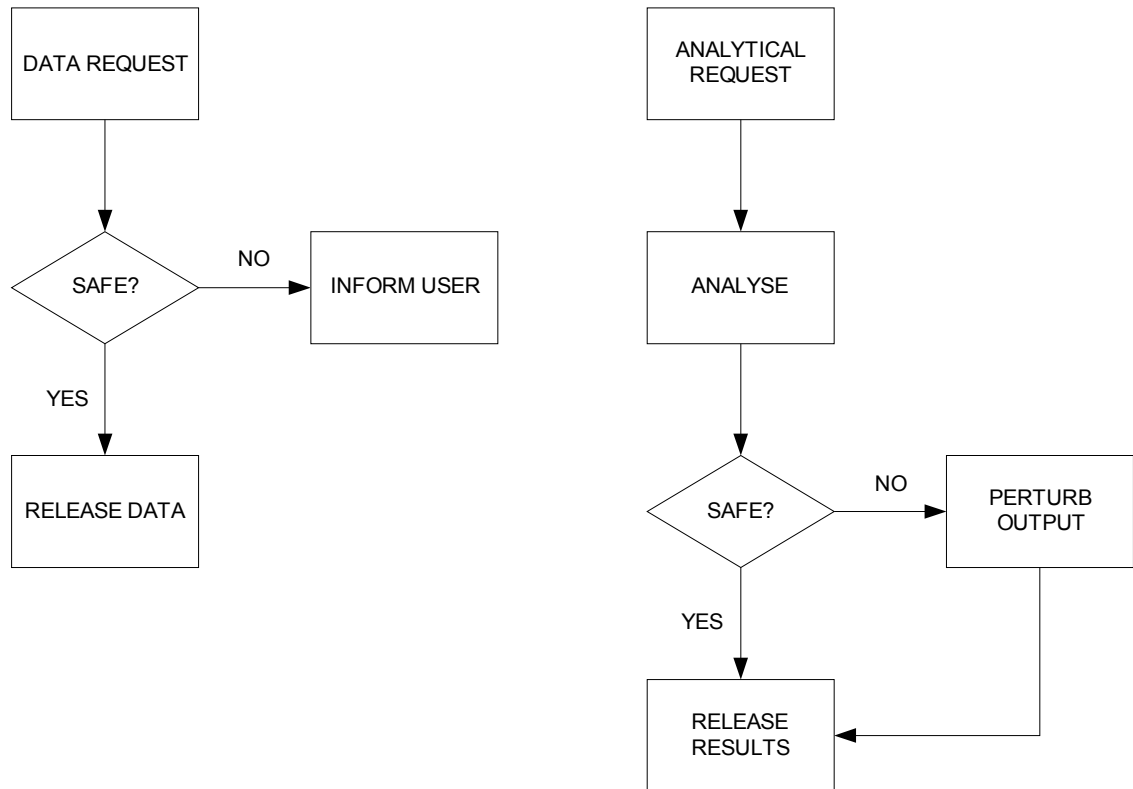


Figure 7. *Flow diagram for a data release and analytical request Web service*

Karr et al. (2002) develop a tool for releasing tabular data based on decomposable graphical releases (which are explained later in this chapter). Elliot et al. (2008) describes a similar tool that incorporates different levels of trust and attempts to minimize the possibility that a low utility data release will limit the possibilities for subsequent high utility releases. It is also based on the theory of decomposable graphs.

Perturbation

Perturbation involves making changes to the data to limit the risk of disclosure. Thus for aggregate data, cells would not simply be suppressed, their counts would be changed in order to add sufficient uncertainty over their true values. Perturbation techniques generally attempt to maintain certain statistical qualities of the data, so that standard methods of data analysis can be employed. Clearly, suppressed counts would entail imputation of the missing values or analytical methods that could handle missing values.

Data swapping

Data swapping is the process of swapping part of a record with another record (Griffin, 1989). So any attempts to match records are faced with the issue that the levels of variables used for matching might be incorrect, and any inferred levels of other variables might be incorrect even if the match is correct. Swapping will generally be limited to key or target variables. Clearly, with a simple data swapping scheme the univariate marginal distributions are preserved. Schemes might be implemented so as to preserve multivariate marginal distributions.

An analogous method for tabular data is to reduce some detail cell counts whilst increasing others so that certain marginal counts remain unchanged. This is known as *controlled tabular adjustment* (Cox et al., 2004). Again, there is the problem of having to make complementary changes to preserve margins.

Controlled tabular adjustment

In controlled tabular adjustment (CTA) each risky cell is changed to a nearby safe value. For example, this might entail making a risky 1 either a 0 or a 2. Minimal adjustments are made, so for any given cell there are a maximum of two values to which it can be changed. The precise nearby values for a cell will depend on context. After adjusting the risky cells to non-risky values other cells are changed so as to preserve certain margins.

The unsafe release in Figure 2 as an example, with the same risky cell, could be changed to the table in Figure 8.

0	4	0	0	4
0	0	0	1	1
4	1	0	2	7
1	0	7	1	9
5	5	7	4	21

Figure 8. *Controlled tabular adjustment*

The risky 1 has been changed to a 0. Three other cells have been changed in order to preserve the univariate margins. The changed cells are those that were suppressed in Figure 6. But again there are alternatives which would potentially have different impacts on data utility.

The table still has a unique detail cell with a unique margin, thus disclosure would appear to be possible by the same means as in the true table. However, the fourth level of the column variable might not be sensitive, the inference would be incorrect, and the knowledge that the table had been subjected to CTA would create uncertainty over such inferences in the mind of the intruder. The latter of these considerations is fundamental to SDC. The fact that an SDC method has been applied should create uncertainty over disclosive (deductive) inferences. A good SDC method will do this adequately whilst minimizing the impact on legitimate, statistical (inductive) inferences.

Optimizing CTA to find a perturbed table that provides adequate protection whilst minimizing the impact on data utility is again a computationally difficult problem involving integer linear programming methods.

Interval publication

Rather than simply masking a risky cell value, the value can be replaced by a feasible range of values. Fischetti and Salazar (2003) introduce the method, referring to it as a form of partial cell suppression. There is the usual issue of having to treat potentially non-risky cells in a similar fashion to avoid recovery of the risky cell values if margins are also published. Cell suppression is essentially interval publication with intervals $[0, \infty]$, so interval publication offers a wider range of solutions. Thus it is possible to find higher utility solutions for a required level of protection. The problem of finding optimal solutions is claimed to be simpler than the complementary cell suppression problem for masking (Salazar, 2008).

Rounding

Rounding is the process of taking detail cell counts and rounding them up or down to disguise their true value. Rounding is generally carried out to a close multiple of a chosen base, n . n is usually chosen to be odd in situations where rounding is to be carried out to the nearest multiple of n , in order to avoid ties.

Barnardization

Barnardization adjusts all non-zero cell counts in tabular data according to the following scheme. Zero counts are left unchanged.

$$P(j|i) = \begin{cases} p & \text{if } j=i \\ \frac{(1-p)}{2} & \text{if } j=i-1 \\ \frac{(1-p)}{2} & \text{if } j=i+1 \\ 0 & \text{otherwise} \end{cases}$$

where $P(j|i)$ is the conditional probability mass function for the published count j given the true count i . and p is the Barnardization parameter; the probability that a non-zero true count will be published (unperturbed).

Controlled rounding

Controlled (deterministic) rounding to base n implies the following conditional probability mass function for a rounded count j given a true count i ,

$$P(j|i) = \begin{cases} I\left(i \bmod n < \frac{n}{2}\right) & \text{if } j = i - i \bmod n \\ I\left(i \bmod n > \frac{n}{2}\right) & \text{if } j = i + n - i \bmod n \\ 0 & \text{otherwise} \end{cases}$$

where $I(\cdot)$ is the indicator function and $a \bmod b$ is the remainder when a is divided by b and takes precedence over other operators.

Random rounding

Random rounding uses a probabilistic scheme such as the scheme below for rounding counts up or down.

$$P(j|i) = \begin{cases} 1 - \frac{i \bmod n}{n} & \text{if } j = i - i \bmod n \\ \frac{i \bmod n}{n} & \text{if } j = i + n - i \bmod n \\ 0 & \text{otherwise} \end{cases}$$

In Barnardization, controlled rounding and random rounding the counts are rounded independently. In both Barnardization and random rounding the expected value of j is i . So the schemes are, in this particular sense, unbiased. Of course, this also implies that marginal totals will not generally be preserved. Publishing marginal totals creates dependencies between cell bounds and raises the possibility of using integer linear programming methods to attempt to recover detail cell counts. Even the publication of a similarly rounded grand total can enable the recovery of all detail cells counts in some cases. For instance, four published detail cell counts of 5 with a published grand total of 0 under random rounding to base 5 can only occur if all true detail counts are equal to 1.

Inference

It is clear that many suppression and perturbation schemes allow lower and upper bounds to be placed on cell counts in aggregate data. The release of marginal tables creates dependencies between bounds, and integer linear programming methods can be used to generate the tightest possible bounds for each published count. These calculations might not require full knowledge of the disclosure limitation scheme. For instance, the bounds on the true cell counts implied by Barnardization do not depend on the Barnardization parameter, p . Finding the tightest possible bounds might be enough for an intruder to recover sensitive information. As shown earlier, in degenerate cases the detail cell counts can be recovered exactly, removing all protection from the data. The tightest bounds dictate the logical inferences that can be made by an intruder. For many data releases the time required to generate the tightest feasible bounds will be enormous. This offers some degree of security. However, as computational power increases and better algorithms are discovered the risk that a given data release can be compromised will tend to increase. Where it is possible to generate the tightest bounds before release they can be checked for

safety. Thus efficient bounds algorithms are also of interest to the risk analyst.

Any scheme which allows bounds to be placed on cell counts also defines a set of feasible detail tables. Generating these tables or sampling from the set of feasible tables can be used to perform inference. A high proportion of feasible tables with a risky cell value equal to 1 might lead an intruder to believe that the true cell value is 1. However, there is often more information available to an intruder. Knowledge of a rounding scheme allows the probability of the released table for a given detail table to be generated. These probabilities can be used to weight the tables and perform Bayesian inference. This is also true for the data analyst, and is a valuable tool for analysing disclosure-protected data. The difference would be that the intruder would be making inferences about individual targets, rather than hypotheses relating to the larger population. Nevertheless, the approach is basically the same.

Generating all feasible tables can be achieved as set out in Dobra and Fienberg (2008). An essentially identical approach was used for the risk analysis of Office for National Statistics (ONS) data in Smith and Elliot (2003). In some cases the number of feasible tables is enormous, and there is no realistic prospect of being able to generate them all in reasonable time. In these cases sampling can be achieved via Markov Chain Monte Carlo methods (Forster et al., 1996; Diaconis and Sturmfels, 1998; Dobra, 2003).

So, for both the intruder and the risk analyst, there are two important aspects regarding aggregate data. There is the issue of bounds and the logical inferences that they allow; and there is the issue of the Bayesian inferences that can be made by either generating all the feasible tables implied by the constraints or by sampling from those tables. This is an area where good algorithms are important. The analysis conducted in Smith and Elliot (2003) initially adopted the approach of generating all feasible tables in order to produce the chosen risk metric. This was expensive given the large number of aggregate data releases that were being considered. But the development of a new algorithm tailored to the form of release allowed the metric to be generated without the need to generate all the feasible tables. Computational costs decreased by orders of magnitude and the analysis could be completed in minutes rather than days. This also demonstrated that a sampling approach that had also been tried was performing poorly in terms of identifying the higher risk tables. The sampling variability was such that it masked the real differences in risk.

The following sections cover some efficient methods for generating bounds along with some discussion of Bayesian methods, and their use in SDC. Inferential methods relating to identification risk are presented in Chapter 6 and the relevant core chapters..

Decomposable graphical releases

In many cases a set of marginal tables might be released rather than a full k -way table. For example, most aggregate output from the U.K. national census is in this form. Each released table might satisfy safety requirements individually. The question is whether the set of tables as a whole is safe. This can be addressed by calculating the implied bounds on the unreleased k -way table with dimensions equal to the union of the dimensions in the marginal tables.

The released marginal tables can be represented as an undirected graph, with a node for each variable and an edge between each pair of nodes that appear in a common margin. If each maximal pairwise connected subgraph in the graph corresponds to the variables in a released margin and the graph is decomposable, then bounds on the full k -way table can be calculated relatively efficiently. A graph is decomposable if it contains no unchorded cycles of length greater than 3. (This will be discussed in some detail in a subsequent chapter).

Assume margins with variable sets $\{A,B\}$, $\{A,D\}$, $\{B,C\}$, $\{B,E\}$, $\{D,E\}$, $\{F\}$ are released. The corresponding graph is shown in Figure 9.

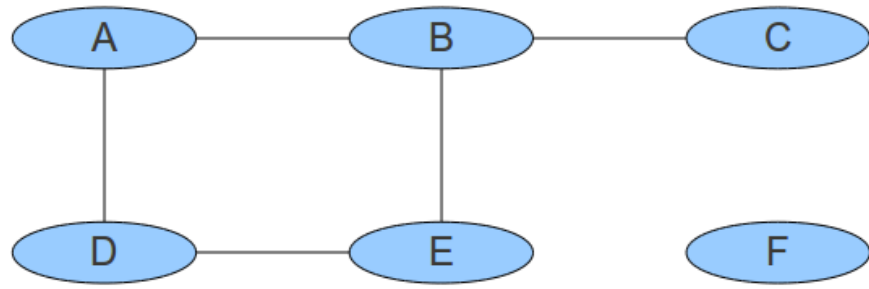


Figure 9. *A non-decomposable graph*

The maximal pairwise connected subgraphs correspond exactly to the released tables. But there exists an unchorded cycle of length 4, $[A, B, E, D, A]$. It is unchorded because the only edges connecting these variables are on the cycle.

Now consider the graph in Figure 10. This is a decomposable graph because the longest unchorded cycles are of length less than 4.

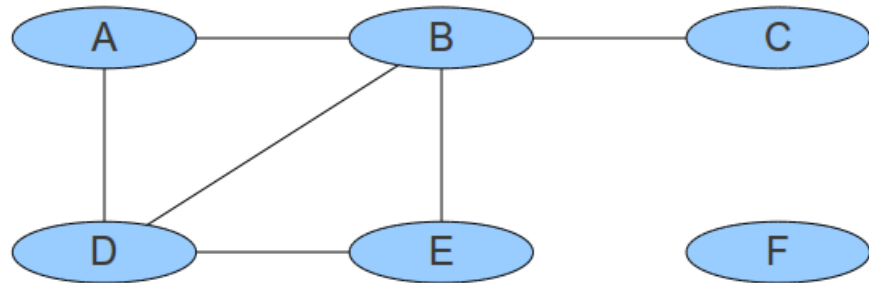


Figure 10. *A decomposable graph*

If this was the graph for a release $\{A, B\}, \{A, D\}, \{B, C\}, \{B, D\}, \{B, E\}, \{D, E\}, \{F\}$ we would have the issue that not all maximal pairwise connected subgraphs correspond to released margins. For instance, there is no released margin $\{A, B, D\}$. But if we had the release $\{A, B, D\}, \{B, C\}, \{B, D, E\}, \{F\}$ then we have a decomposable graphical release and Dobra and Fienberg (2001) show how the bounds on the full 6-way table can be efficiently

calculated.

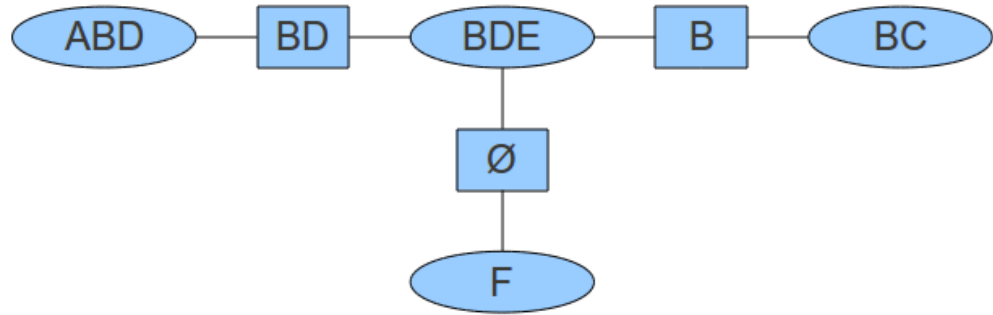


Figure 11. *A junction tree*

One approach to calculating bounds is to produce a junction tree from the graph. Algorithms for constructing junction trees appear in the literature (Lauritzen and Spiegelhalter, 1988). An algorithm is briefly outlined in a subsequent chapter. For most purposes it is sufficient to construct a forest of trees, one tree per connected component in the corresponding decomposable graph. For bounds calculations the number of components is relevant, so the choice has been taken to illustrate the calculations with a true tree. Hence the connection between nodes $\{B,D,E\}$ and $\{F\}$. Separators are associated with each tree edge. Thus the tree comprises a collection of nodes associated with a set of released tables, C , and a collection of separators that have an associated set of smaller marginal tables, S , which can be derived from the released tables. Each cell in each table in C and S is an aggregation of cells in the full k -way table. Thus each cell in the k -way table maps to a set of *relevant* cells in C and S . For the empty set each k -way cell maps to the scalar equal to the table sum.

For each cell in the k -way table its upper bound is the minimum of its relevant cells in C .

For each cell in the k -way table its lower bound is the sum of the relevant cells in C minus the sum of the relevant cells in S , or 0 if the calculated value is negative.

These results can be extended to decomposable graphical releases of perturbed tables where the bounds for the cell entries can be calculated from knowledge of the perturbation

method. When a release is not decomposable it is sometimes possible to break it down into irreducible components (Liemer, 1993), the solutions of which can be combined efficiently using the above approach. For instance, if the graph in Figure 10 had corresponded to the release $\{A,B\}$, $\{A,D\}$, $\{B,C\}$, $\{B,D\}$, $\{B,E\}$, $\{D,E\}$, $\{F\}$, then the bounds for the table with dimensions A,B,D,E could be solved using an alternative method. Then the solution could be combined with the BC and F , using the upper bounds for $\{A,B,D,E\}$ in the calculation of the upper bounds for the k -way table, and the lower bounds for $\{A,B,D,E\}$ in the calculation of the lower bounds for the k -way table.

The generalized shuttle algorithm

Consider a database consisting of a k -way contingency table. Release of the full k -way table might represent a risk of disclosure, generally from low counts. In this case it is common to release contingency tables for strict subsets of the variables in the full k -way table. The question from a statistical disclosure point of view is which sets of such marginal tables constitute safe releases. The marginal tables themselves might constitute a disclosure risk, or they might allow the generation of lower and upper bounds on the cell counts in the full table which represent a disclosure risk.

Buzzigoli and Giusti (1999) introduced the shuttle algorithm for generating bounds on k -way tables given a set of marginal tables. This has since been generalized (Dobra and Fienberg, 2001; Dobra and Fienberg, 2008).

Assume we have the following 2×3 table.

n1	n2	n3	n1+n2+n3
n4	n5	n6	n4+n5+n6
n1+n4	n2+n5	n3+n6	n1 + n2 + n3 + n4 + n5 + n6

Figure 12. *An unreleased 2-way table with released margins*

The two 1-way margins are known (and hence the total), but the individual counts are unknown. The bounds algorithm is based on iterating through a set of dependencies of the form shown in Figure 13.

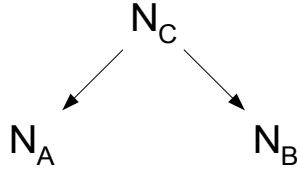


Figure 13. *A dependency*

The cell N_C represents a cell that might appear in a recoded table with N_A and N_B representing cells that would be aggregated in the recoding. It is convenient to denote the cells in terms of the sum over the detail cells for which they constitute an aggregation.

For example one dependency from the above table would be $(n_1+n_2) + (n_3) = (n_1+n_2+n_3)$. The first term on the left hand of this dependency can be expressed as a second dependency, $(n_1) + (n_2) = (n_1 + n_2)$. The set of all dependencies can be shown in a graph. Figure 14 shows the graph for the example table in Figure 12. The only issue with this representation is that it masks the fact that the dependencies it contains are of the one parent, two children form shown in Figure 13.

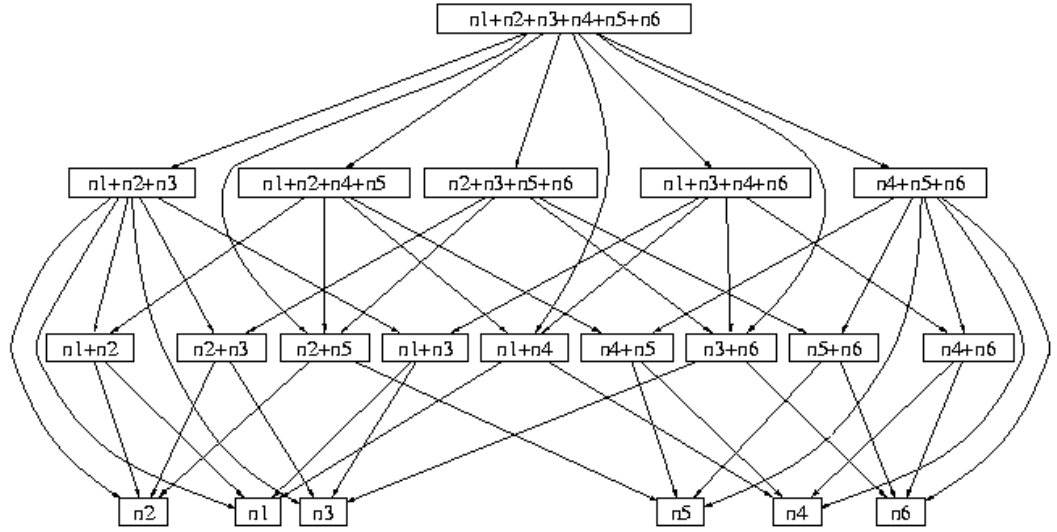


Figure 14. *The dependency graph for the tables in Figure 12*

It is clear that if we had such a collection of dependencies with known detail cell counts, then we could propagate the sums for all nodes in the above graph by iterating through the dependencies. In each case the parent sum would equal the sum of its children. If we had a dependency where the parent sum and one child sum was known, then the sum for the other child could be easily calculated. But each dependency can be used to update bounds in a similar way. The lower bound for the parent implied by the lower bounds for the children is their sum. Similarly the upper bound for the parent is cannot be greater than the sum of the upper bounds of its children.

The shuttle algorithm instantiates known lower and upper bounds for each cell in the graph. It then repeatedly iterates through the cell dependencies, tightening existing bounds based solely on the simple relationships relating the bounds within each dependency. It stops when a sweep through the dependencies does not tighten any bounds. It is not generally guaranteed to find the tightest possible bounds, something that would be of interest to a data intruder and a DSO wishing to guarantee confidentiality. It does find the tightest possible bounds for some types of release. For instance, it does find the tightest possible bounds for a decomposable graphical release of unperturbed tables. However, it is less efficient than the previously outlined propagation algorithm for decomposable releases. The algorithm does place bounds on all cells in all possible recoded tables, which

can be useful for assessing recoding options.

It is possible to extend the basic shuttle algorithm to generate the tightest possible bounds. The cell bounds for a cell of interest can be tightened and a propagation through the dependencies performed. If the new bounds are too tight the algorithm will meet an inconsistency; for example, a dependency where the upper bound is less than the sum of the lower bounds of its children. When a consistency is found the sequence of bound tightenings needs to be retracted before proceeding to the next trial tightening. It can be a costly exercise.

For larger problems the number of dependencies becomes very large. Yet it is an *anytime* algorithm. It can be stopped before completion and the current bounds examined. This can be still be useful for problems that are too large to solve completely.

The algorithm can also be used to generate all feasible detail tables. This requires recursively instantiating feasible detail cell counts and solving for the remaining cells. It is time consuming, but useful for small releases.

Bayesian methods

Bayesian methods can be used to investigate any quantity of interest, whether it is a hypothesis relating to the population, or whether it is an inference regarding a particular member of the population. Although the calculation of the tightest bounds on detail cell values provides some information regarding the possibilities for disclosure, it does not provide a complete picture. An intruder prepared to adopt a Bayesian approach might be able to generate high probability disclosive inferences even if the range of possibilities is quite wide.

The ability to generate all feasible tables given a set of (possibly disclosure-protected) margins allows exact Bayesian inference. Where this is computationally prohibitive there are Markov Chain Monte Carlo methods for sampling from the set of feasible tables. Knowledge of perturbation schemes allows likelihoods to be calculated, potentially

generating high posterior probabilities for a given cell count (Smith and Elliot, 2003) (Forster and Gill, 2008). Even naïve schemes (which make questionable assumptions to reduce computational cost) can generate accurate posterior mass functions (Smith and Elliot, 2005).

But the possibilities for Bayesian methods are wider than this. Bayesian methods could be used to launch the type of attack on released data that might be employed by an intruder. A posterior sufficiently close to 1 (or perhaps also 0) and / or sufficiently different to the prior, might be considered too risky.

Attack scenarios dictate the inferences that an intruder might wish to make. The Bayesian approach can handle various forms of suppression and perturbation and estimate the chances of disclosure occurring under a given scenario. It naturally provides meaningful risk metrics. In principle such metrics could be applied to perturbed data, enabling the effectiveness of disclosure protection to be assessed. For instance, the Bayesian alternatives to the SUDA approach could be adapted for use with perturbed data. The SUDA approach itself requires exact sample data.

Summary

This work is primarily concerned with the inferences that can be made about targets by a knowledgeable data intruder. Protection methods and risk metrics must reflect this. They must also reflect the costs to the intruder, in computational terms and other terms such as the cost of searching a population for a match. Thus both algorithms and attack scenarios are important aspects of SDC.

This chapter has covered some basic ground, explaining some fundamental concepts in SDC and summarising the risk assessment and risk limitation approaches that will be met, in some cases only tangentially, in the core chapters.

Utility and sensitivity are very important aspects of SDC and probably deserve much more attention than they have had in the literature. Due to the focus of this body of work they

also receive little attention here. But hopefully the preceding discussion (particularly regarding sensitivity) has demonstrated that they are often not handled particularly well in SDC.

Attribution risk is another under-researched area. The basic principles of conditioning on known information to revise beliefs is straightforward in principle, and certainly well-known to Bayesians. Exact attribution stems from logical inferences that might require the solving of systems of linear constraints. Some efficient approaches for bounds calculations for certain forms of data release have been outlined, although discussion of general methods for integer linear programming have been avoided. Approximate attribution might simply use the proportions that can be derived from conditional tables of counts. However, it could be a much more involved affair, specifying a prior (or priors) over unobserved variables, and using Markov Chain Monte Carlo methods to generate posterior beliefs.

Identification risk is intimately related to population frequencies. Released population data that have not been disclosure-protected provide these to the data intruder. In such cases exact identification might be possible. In other cases the intruder can use the sample data to inform attack strategies and to make inferences regarding the population counts, and hence the probability that a given match is correct. The risk assessor with access to the correct population frequencies could use these to generate risk metrics. However, for survey data the assessor would not have access to the population data and would have to adopt a role very similar to the data intruder. In this case adequate risk assessment requires that the assessor is able to generate inferences as reliably as the intruder. Treating sample uniques equally would be a poor approach to risk assessment, and several approaches that avoid this have been referenced. They will be revisited in Chapter 8.

Some limitation methods have been outlined. These involve suppression and / or perturbation. Suppression involves not releasing data. Perturbation involves distorting the data before release. In some cases, such as interval publication, it could be argued to be either suppression or perturbation. The main issues with disclosure limitation methods are their impact on data quality and the level of protection they provide. Some methods of perturbation attempt to minimize loss in data quality by being, in some sense, unbiased.

One approach to data quality would be to perform statistical inferences with the disclosure-

protected data and to compare them with the corresponding inferences using the unprotected data. Assessment of the adequacy of protection might involve attacking the data from the point of view of the intruder, generating deductive inferences regarding the data units under plausible attack scenarios. Thus data quality and the level of protection could both be assessed via Bayesian methods. Bayesian metrics can, in principle, be applied to both unprotected and protected data.

For aggregate data the main computational issues are the calculation of cell bounds (for exact inferences) and the generation of posterior beliefs (for approximate inferences). Some methods for cell bounds have been presented. An efficient Bayesian approach for modelling the processes that produce aggregate data is presented in Chapter 6. This might form part of an assessment of either attribution risk or identification risk.

CHAPTER 3

A Measure of Disclosure Risk for Aggregate Data

Introduction

Releases of population data can be used by data intruders to glean sensitive information about individuals in the population. Disclosure occurs when a data intruder makes reliable inferences (i.e., with a high degree of confidence) about one or more population units. Statistical agencies need to guard against disclosure in order to meet their legal obligations to safeguard respondent confidentiality and to maintain public trust. Lack of trust can result in individuals refusing to complete, for example, census forms or returning forms with false or missing information. Most statistical agencies are mainly concerned with the risk of an intruder identifying a population unit, although this is not a requirement for disclosure of information about the individual concerned.

The need for appropriate measures of disclosure risk has been well discussed. Many authors have indicated that such measures should as far as possible take a data intruder's perspective of the risk (see e.g. Paass, 1988; Mokken et al., 1992; Elliot and Dale, 1999). Although intruder-based measures have been established for identification risk (Skinner and Elliot, 2002), little progress has been made with generating appropriate risk metrics for the actual disclosure of information about members of the population in the absence of identification. This chapter describes the "subtraction - attribution probability" (SAP) method which attempts to fill this gap.

Disclosure Risk

Understanding of disclosure risk has evolved over the last twenty years and there is still no

unequivocal definition of the term. However, definitions of disclosure generally involve one or more of the following:

Identification – A one to one association between a data unit and a target.

Attribution – The association of one or more variable values with a target.

Here, a data unit is an individual or organization contained in microdata or tabulated data that is available to a data intruder; a target is an individual or organization about which a data intruder is trying to discover information.

In some cases it is possible for an intruder to perform identification or attribution with absolute certainty. In these cases the identification or attribution is termed exact. Otherwise, identification or attribution is termed approximate. Strictly speaking there will almost always be a degree of uncertainty regarding the correctness of the data, so all inferences are approximate. However, this source of uncertainty is generally ignored for disclosure risk assessment purposes, and this practice is followed here.

Samarati and Sweeney (1998) introduce k -anonymity. A dataset satisfies k -anonymity, for $k > 1$, if at least k records exist in the dataset for each observed combination of key variable levels. Thus no exact match can be made against any record on the key variables. It does not generally protect against exact attribution because records might contain common variable levels (Smith and Elliot, 2008; Domingo-Ferrer and Torra, 2008).

Machanavajjhala et al. (2006) introduce l -diversity. In its simplest form l -diversity requires that there must be at least l distinct values for each sensitive variable, for each combination of key variable levels. Thus it protects against exact positive attribution, but not negative attribution. Other forms of l -diversity are discussed in Domingo-Ferrer and Torra (2008).

Neither k -anonymity or l -diversity constitute measures of risk. They are criteria which are either satisfied, or not.

This paper addresses the risk of exact attribution. Previous papers have tended to concentrate on attribution stemming from identification. Fellegi (1972) considers

disclosure in terms of “sufficiently narrowly defined” populations, and goes on to state that such a population may “contain only one identifiable respondent or, at least, information can be deduced from the published estimates that can be related to a particular identifiable respondent”. He then goes on to illustrate how disclosure can occur from the conditioning on known information about a target, the conditional frequency table containing only the target individual. Clearly, if an intruder can achieve this by conditioning on a subset of the variables in the tabulation, then the levels of all the remaining variables can be discovered. If the levels of the discovered variables were previously unknown to the intruder, then disclosure has taken place. Fellegi also considers that conditioning to a (sub-)population of size two can result in similar disclosure if the intruder is the other member of the conditional population. A U.S. Department of Commerce report (U.S Department of Commerce , 1978) expands this idea by considering “coalitions” of individuals within a data set who might cooperate in order to discover new information about targeted individuals. The report also considers how disclosure can take place without the requirement for identification. Their examples are reproduced below.

In Figure 15 conditioning on a target being a resident of County B implies that the target is black. A risk of such exact disclosure exists if a marginal total (in dimension n-1) equals one of its detail cells (in dimension n). This contrasts with the example given by Fellegi (1972) which required also that the detail cell count be 1.

County	Race			Total
	White	Black	Other	
A	15	20	5	40
B	0	30	0	30

Figure 15. *Number of beneficiaries by count and race*

The U.S. Department of Commerce report contrasts this with the case when the sum of a proper subset of detail cells equals the total in the relevant margin (Figure 16). The report does not define the implication that a target in County B is either Black or Other as disclosure, because the subset of Black or Other is not as narrowly defined as possible. Similarly, the report authors do not consider exact inferences regarding age as disclosive

unless age is revealed to within a single year.

County	Race			Total
	White	Black	Other	
A	15	20	5	40
B	0	28	2	30

Figure 16. *Number of beneficiaries by count and race*

This distinction is fairly arbitrary as ethnicity can be broken down into more detailed classifications than those of the example, and any categorization of a continuous variable such as age will involve ranges that are not as narrowly defined as possible. One approach would be to associate sensitivities with any set / range of variable levels and consider disclosure to have taken place if the sensitivity of the discovered information exceeds some predefined threshold. However, data is often collected with an unqualified assurance of confidentiality, so that it is arguable that all data should be regarded as sufficiently sensitive to warrant protection. Therefore, for the purposes of this paper disclosure is considered to have taken place when an intruder, by whatever means, is able to condition to a population table which contains one or more zeros. This definition encompasses the two cases illustrated above, and the additional case where an intruder can infer that a particular combination of attribute levels does not apply to a target. For example, simply conditioning on a target being a member of the population in Figure 16 allows the intruder to infer that the target is not White and residing in County B (although either are individually possible). So in this strict sense, a risk of disclosure is present if a population table contains one or more zeros.

Skinner (1992) defines disclosure in the sense of Fellegi's (1972) example (requiring identification and attribution) as identification disclosure, whereas disclosure that does not require identification is defined as prediction disclosure. He considers approximate disclosure in sample tables and develops an argument that identification disclosure is a necessary and sufficient condition for prediction disclosure. This paper concerns only the risk of exact disclosure in population tables. Under these circumstances it is clear that identification is neither necessary nor sufficient for attribution (prediction disclosure).

Attribution risk from low population counts

Thus far in this paper the risks of attribution have only been considered in terms of conditioning on known information relating to some targeted individual. It is implicit that the intruder is also conditioning on a target being a member of the population. But conditioning on known variable levels (or known absence of variable levels) is not the only way an intruder might attempt to condition down to a smaller, more disclosive population. U.S. Department of Commerce (1978) describes the possibility of disclosure stemming from coalitions, the main questions arising regarding the likely size of coalition, and the distribution of the coalition within the population. However, the type of disclosure that can arise from coalitions does not require their existence. It is possible for an intruder to hold information on a number of population units, without their explicit cooperation. If they can be identified within the population, then their records can be removed from the data set, facilitating inferences regarding the residual sub-population. Removal of a unique clearly leads to the presence of a zero, and a risk of attribution. Of course, records of known individuals may be removed without identification, and partially known individuals might be ‘removed’ from the relevant margins, placing constraints on the counts in the full cross-classification of the residual population. In essence, an intruder can use arbitrary known information about the population units in order to try to facilitate attribution. Lower counts represent a greater risk of the recovery of zeros by subtraction of known individuals.

The above requires information that can be considered external to the data set in question, and as such might not be considered an overriding issue. However, any inferences regarding a population unit require such information. Both exact identification and exact attribution require external information; at the very least an intruder must be able to condition on a target being a member of the relevant population.

Protection against attribution

Statistical agencies tend to guard against disclosure by suppressing (withholding) data or disguising the true counts by deterministic or stochastic perturbations; (see Duncan et al.

(2001) for a review). For example, one deterministic method is conventional rounding. A suitable non-negative odd integer is chosen as base, and each count in the cross-classification is rounded to the closest multiple of the base. Figure 18 contains the conventionally rounded, to base 3, cross-classifications corresponding to the exact cross-classification in Figure 17.

		VAR2			
		D	E	F	
VAR1	A	1	3	0	4
	B	4	0	0	4
	C	3	2	0	5
		8	5	0	13

Figure 17. *A 2-way cross-classification with margins*

		VAR2			
		D	E	F	
VAR1	A	0	3	0	3
	B	3	0	0	3
	C	3	3	0	6
		9	6	0	12

Figure 18. *A conventionally rounded cross-classification with conventionally rounded margins*

An intruder (with knowledge of the rounding scheme) can easily generate bounds on the counts in the exact 2-way cross-classification, given the corresponding rounded cross-classification. Let these be termed *trivial* bounds as they are based solely on the rounding scheme.

		VAR2		
		D	E	F
VAR1	A	0	2	0
	B	2	0	0
	C	2	2	0

Figure 19. *Trivial lower bounds*

		VAR2		
		D	E	F
VAR1	A	1	4	1
	B	4	1	1
	C	4	4	1

Figure 20. *Trivial upper bounds*

Here the rounding has managed to disguise the exact value of all counts. But subtraction of a known individual in cell (A,D) would recover a zero.

It is not unusual for statistical agencies releasing perturbed cross-classifications to also release perturbed, or occasionally exact, marginal tables. The presence of marginal counts places a system of linear constraints on the counts in the full (in this case 2-way) cross-classification. Solving the system of constraints via integer linear programming methods can lead to tighter bounds than those derived solely from a full rounded cross-classification. Dobra (2002) develops a method for solving cell bounds given marginal cell counts. Although his algorithm is designed to deal with exact cross-classifications it is relatively easily extended for dealing with perturbed counts (Smith and Elliot, 2003). The release of all the rounded cross-classifications (including both 1-way margins and rounded total) in Figure 18 results in the following lower and upper bounds.

		VAR2		
		D	E	F
VAR1	A	0	2	0
	B	3	0	0
	C	3	2	0

Figure 21. *Non-trivial lower bounds*

		VAR2		
		D	E	F
VAR1	A	1	3	0
	B	4	1	0
	C	4	3	0

Figure 22. *Non-trivial upper bounds*

Three of the four zeros have been recovered. This stems from the fact that the trivial lower bounds for the VAR2 margin sum to 13, which is the trivial upper bound for the rounded total. Thus the total and VAR2 margin are recovered exactly. So the perturbation of the data has done little to remove the risk of attribution. Subtraction of known individuals could increase the risk still further.

A measure of attribution risk

A risk of attribution exists if, and only if, one or more zeros exist in some population cross-classification. The population cross-classification in question need not necessarily have been released. In fact, it is possible to construct examples where the exact counts in a 3-way cross-classification can be recovered from its three distinct 2-way margins. In any case, a set of population cross-classifications can be used to place bounds on any cross-classification from which they could be derived. It is enough to consider only the ‘base’ cross-classification with axes corresponding to the union of the variables in the released cross-classifications. Any cross-classifications over a superset of the variables in the base cross-classification contain (recovered) zeros if, and only if, the base cross-classification contains (recovered) zeros. Bounds on smaller margins can be solved, but again this is unnecessary, as any zero in a margin implies zeros in the full cross-classification.

Given the questionable distinction between inferences on the basis of the ‘narrowness of definition’ a measure is proposed based simply on the presence of zeros in the full population cross-classification. Sensitivities are not considered for the reason given earlier, although the methodology can be applied to conditional tables as easily as marginal tables, in which case risk could be assessed for given population units or population cells given an assumed set of *key* variables. It would also be useful to take into account the additional risk stemming from intruder knowledge of the population, and to be able to apply the measure to relatively arbitrary releases of exact and / or perturbed cross-classifications. Specifically, the chosen measure is ‘the probability of recovering one or more zeros in the full cross-classification given the subtraction of a random sample of n population units’. This is

termed the subtraction attribution probability (SAP). The parameter n is designed to encapsulate the intruder's knowledge of the population. The SAP measure can be generated for an assumed level of intruder knowledge, or across a range of values for n to investigate how risk varies with intruder knowledge.

Notwithstanding the previous discussion of the ‘narrowness of definition’, the presence of a few zeros in a base table with many cells might not be considered to be a major disclosure risk (although we would ideally consider both the probabilities of specific disclosures and their sensitivities in a full risk analysis). This suggests alternative measures such as ‘the probability of recovering k zeros in the full cross-classification given the subtraction of a random sample of n population units’ or ‘the expected number of recovered zeros in the full cross-classification given the subtraction of a random sample of n population units’. In fact the latter measure is more easily calculated than the chosen measure for some forms of data release. However, the measure adopted here was specifically designed for low dimensional tables at very small levels of geography where a single zero would constitute a significant risk.

Assume a base table of counts of arbitrary dimension with cell counts $c_i, i = 1$ to m . Assume that an arbitrary set of perturbed marginal tables is published, each perturbed using some independent rounding scheme (i.e. each cell is perturbed independently of the others). Then each published count, x , implies a pair of constraints of the form, $l \leq c$, $c \leq u$, where l and u are the trivial bounds implied by the rounding scheme and c is the total of some set of cells in the base table. Dependencies between bounds might imply that there exist tighter bounds than the trivial bounds. These can be found by integer linear programming methods. The recovery of a zero by subtraction of a known sample of the population occurs if, and only if, the sample implies that $s_i = c_i = u'_i$, where s_i is the corresponding known sample count and u' is the table of the tightest upper bounds on the base table implied by the set of all linear constraints.

The probability of recovering at least one zero for some assumed level of intruder knowledge, equivalent to a random sample of size n , is

$$SAP(n) = \frac{\sum_{s \in S} P(s|p) I(\sum_i s_i = n) I(0 \in u' - s)}{\sum_{s \in S} P(s|p) I(\sum_i s_i = n)},$$

where S is the set of all possible sample tables, p is the population table (known to the data holder), sampling is simple random sampling without replacement, subtraction of tables is pointwise, and $I(\cdot)$ is the indicator function.

For a data release comprising of a single rounded table there is a pair of constraints, $l_i \leq c_i$ and $c_i \leq u_i$ for each cell i . The mutual orthogonality of these pairs of constraints in R^n ensures that the trivial bounds are the tightest bounds. For a sample with corresponding sample counts, s_i , $i = 1$ to m , the SAP measure for a given sample size, n , can be calculated as follows.

Single rounded table

The marginal probability of recovering zeros in any set of cells with total x is simply the following Hypergeometric probability,

$$\frac{\binom{N-x}{n-x}}{\binom{N}{x}},$$

where N is the cross-classification total, $\sum c_i$.

Applying the inclusion / exclusion principle it is simple to derive an expression for the probability that at least one cell is zero given a random sample of n population units.

Let Z denote the set of all subsets of cell indices, equal to the union of the sets of n -subsets $Z(0), \dots, Z(m)$. i.e. $Z(0) = \emptyset$, $Z(1) = \{\{1\}, \dots, \{m\}\}$, $Z(2) = \{\{1,2\}, \{1,3\}, \dots, \{m-1, m\}\}$, \dots , $Z(m) = \{\{1, \dots, m\}\}$.

Let e.g. $c_1 + c_2$ be denoted by $c_{\{1,2\}}$.

Then,

$$SAP(n) = \sum_{i=1}^m \left((-1)^{i-1} \sum_{z \in Z(i)} \frac{\binom{N-c_z}{n-c_z}}{\binom{N}{n}} \right)$$

In practice many of the terms in the above summation will be equal to zero. For exact tables $l_i = c_i = u_i$ for all i , and all cell counts represent some risk of recovering a zero, although for a given level of risk, n , it is only necessary to consider c_z such that $c_z \leq n$.

For rounded tables it is only necessary to consider c_z such that $c_z = \sum_{i \in z} u_i \leq n$.

Single rounded table and rounded total

In this case there is an additional pair of constraints, $l_t \leq \sum_i c_i$ and $\sum_i c_i \leq u_t$, where u_t denotes the trivial upper bound for the table total. There is also the obvious risk of subtraction where $\sum_i s_i = u_t$, and this only occurs when $\sum_i s_i = \sum_i c_i = u_t$. But this new constraint is not mutually orthogonal to the existing constraints, and the trivial upper bounds on the base table counts might not be the tightest possible bounds. With a rounded total and rounded sample the upper bound on a base table cell j is the minimum of the trivial upper bound and the upper bound implied by the bounds on the other base table cells and total,

$$u'_j = \min \left(u_j, u_t - \sum_{i \neq j} l'_i \right)$$

where $l'_i = \max(l_i, s_i)$.

Lemma

If there is any risk for the release without rounded total, then the release of the rounded total results in no increased risk.

Proof

Recovery of a zero by subtraction in the base table occurs if, and only if, $u'_j = c_j = s_j$ for some cell j . For this to occur due to a tightening of a trivial base table cell upper bound we require that for some cell j both $u'_j < u_j$ and,

$$c_j = u_j - \sum_{i \neq j} l'_i.$$

Let $u_i = \sum_i c_i + k$ where k is a non-negative integer. Then,

$$c_j = \sum_i c_i + k - \sum_{i \neq j} l'_i$$

$$c_j - l'_j = \sum_i (c_i - l'_i) + k.$$

This equality clearly does not hold for $k > 0$.

The equality does hold if, and only if, both $k=0$ and $\sum_{i \neq j} (c_i - l'_i) = 0$. The latter would imply that either one or more base counts were not rounded, or that $s_i = c_i \forall i \neq j$. But any non-zero risk for any sample without rounded total implies that $s_i = c_i = u_i$ for at least one i , where $i \neq j$ because $u'_j < u_j$.

The Lemma is proved for all independent rounding schemes that perturb all base table counts.

Corollary 1

If $u_i > \sum_i c_i$, then the risk with rounded table is exactly the same as the risk without rounded table.

Corollary 2

If a rounded table represents zero risk, then the addition of a rounded total represents a risk if, and only if, $u_i = \sum_i c_i$. This risk pertains only to knowledge of the full table, unless exactly one cell count, say c_j , is not equal to its trivial lower bound. In that case all tables such that $s_j = c_j$ represent a risk.

So if $s_i = u_i$ for any $s \in S$ or $u_i > \sum_i c_i$, then the algorithm for single rounded tables can be used.

Otherwise, the above results lead to the following algorithm.

1. Construct a list containing the trivial lower bounds for the rounded base table counts (i.e. based solely on the rounding scheme).
2. Construct a corresponding list of counts for the exact cross-classification.
3. Find the sum, S , of those counts in the list of lower bounds that are equal to the corresponding count in the list of exact counts.
4. For all n in the range 0 to $(T - S - 1)$ (where T is the exact cross-classification total) the SAP measure is zero.

5. For each n in the range $(T - S)$ to T the SAP measure equals $\frac{\binom{S}{n-T+S}}{\binom{T}{n}}$

Figure 23 shows the system of constraints for a 2 cell table for Gender with randomly

rounded (to base 5) counts of 5, and a similarly rounded total of 0. Assuming the underlying counts are 3 females and 1 male, then the observable samples are indicated by the crosses.

With only the orthogonal constraints implied by the detail cells there is no risk of recovering a zero. It is only because the lower bound for the table total equals the actual table total that the rounded total implies additional risk. Clearly if all 4 individuals are known, then the residual table contains zeroes, $SAP(4) = 1$. But just knowing the 3 females implies that there are no more females, because there are at most 4 individuals and at least 1 is male. The other possibility for $n = 3$ is knowing 2 females and 1 male. There is only 1 way to observe 3 females, and 3 ways to observe 2 females and 1 male. Hence $SAP(3) = 0.25$.

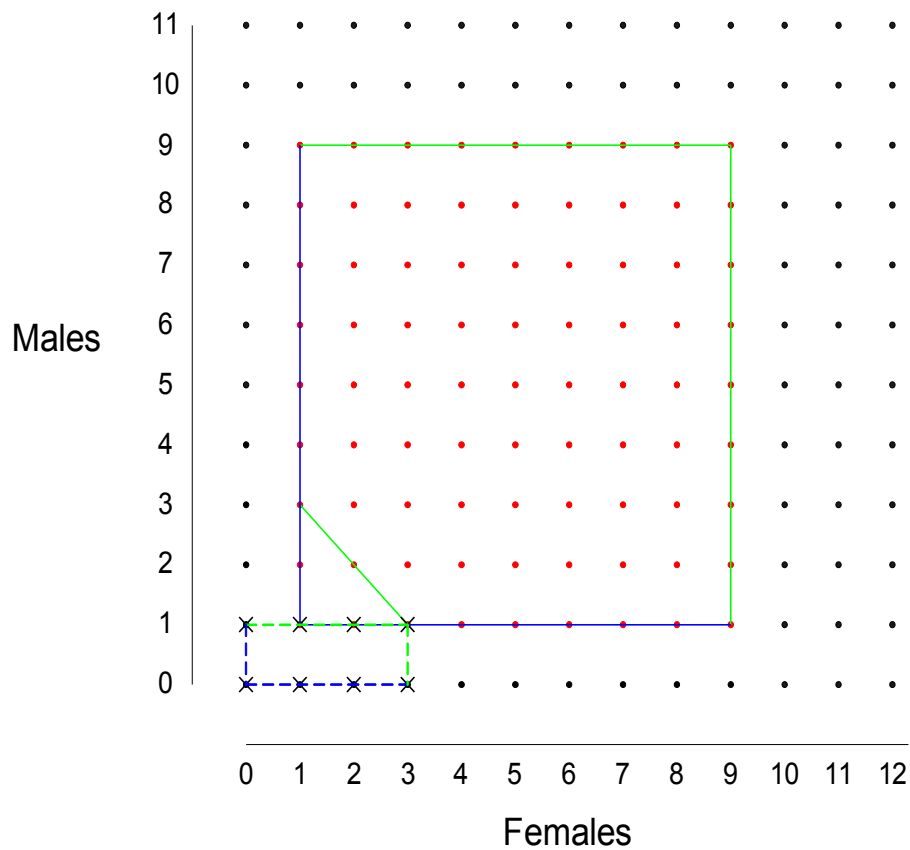


Figure 23. *System of constraints for rounded detail counts and rounded total*

General table releases

It is hoped that the existing results can be further generalized to provide efficient means for calculating SAP measures for more general table releases. The current approach is to use an extended version of Dobra's (2002) shuttle algorithm to solve the initial bounds problem and then recursively generate all tables with non-zero risk (Smith and Elliot, 2003). Randomly sampling tables is an alternative approach for generating approximate SAP measures.

A comparison of rounding schemes

An extensive SAP analysis has been conducted on the UK Neighbourhood Statistics (Smith and Elliot, 2003). The analysis is confidential and cannot be described here. Table 1 contains some results for an analysis of a set of 1200 randomly generated 2×6 cross-classifications. The cross-classification counts were generated from a Poisson distribution with mean 2. Each cross-classification was conventionally rounded to base 5, and the exact total was conventionally rounded (to base 5) to produce a rounded total. SAP measures for each cross-classification were generated for $n=0$ to 24. Table 1 contains the numbers of cross-classifications that had SAP scores in various ranges. SAP scores that were exactly 0 or 1 are contained in the second left and rightmost columns respectively.

For n in $\{0,1\}$ the SAP measure is necessarily 0 for all cross-classifications, due to the nature of the rounding scheme. For $n=2$ the SAP measure could be as high as 1, given a cross-classification total of 2.

The SAP measure for any individual cross-classification and value of n must be at least that for $n-1$, so there tends to be a migration of SAP measures from 0 to 1 as n is increased. For cross-classifications with no relevant cells, the SAP measure is zero for all n .

Table 1 demonstrates how the risk of recovering a potentially attribute-disclosive cross-classification tends to increase with greater intruder knowledge of the population. This also depends on the size of the cross-classifications, the distribution of counts and the rounding scheme. However, the pattern of results shown in Table 1 is reasonably close to that which have been found with real-world data sets. Analyses such as this can be used to help define threshold values for n for which a non-zero (or value greater than another threshold) SAP measure can be considered to constitute too great a risk for release. Similarly, analyses can be used to investigate the protection afforded by alternative perturbation schemes. Any comprehensive analysis of perturbation schemes would also consider the effect of perturbation on data quality.

	SAP=0	0-0.1	0.1-0.2	0.2-0.3	0.3-0.4	0.4-0.5	0.5-0.6	0.6-0.7	0.7-0.8	0.8-0.9	0.9-1	SAP=1
$n=0$	1200	0	0	0	0	0	0	0	0	0	0	0
1	1200	0	0	0	0	0	0	0	0	0	0	0
2	26	1173	1	0	0	0	0	0	0	0	0	0
3	26	1137	36	0	0	1	0	0	0	0	0	0
4	25	869	275	28	2	0	0	0	1	0	0	0
5	25	460	497	160	40	16	1	0	0	0	0	1
6	25	234	471	267	127	48	20	6	1	0	0	1
7	23	108	332	365	169	113	59	13	14	2	0	2
8	23	60	226	266	292	115	117	50	28	17	4	2
9	23	33	144	201	254	212	115	102	64	31	17	4
10	23	14	93	146	188	203	220	93	105	75	30	10
11	23	9	58	110	158	150	205	176	125	95	72	19
12	22	4	42	63	108	149	186	154	188	113	138	33
13	22	4	21	51	84	126	126	169	211	148	186	52
14	22	3	12	37	57	104	111	138	157	234	244	81
15	22	3	8	33	32	65	104	124	159	217	303	130
16	22	3	2	18	32	46	80	108	127	199	379	184
17	22	3	0	12	30	31	46	101	118	182	395	260
18	21	4	0	7	16	28	39	73	93	144	442	333
19	20	5	0	2	13	26	25	45	84	143	409	428
20	20	5	0	0	9	12	29	40	54	105	423	503
21	20	4	1	0	7	9	26	18	43	104	365	603
22	20	3	1	0	2	10	9	27	35	74	318	701
23	20	3	1	0	0	7	10	16	26	42	290	785
24	19	3	1	1	0	5	6	10	26	33	235	861

Table 1. *Simulation results showing, for 1200 randomly generated tables, the banded probabilities of producing a table containing at least one zero given subtraction of n randomly selected units from the population*

Summary

The SAP method provides an integrated approach for assessing attribute disclosure risk for any given release of cross-classifications. It incorporates the notion of intruder knowledge and allows the same metric to be produced for single released cross-classifications and multiple released cross-classifications, whether perturbed or unperturbed. Computational constraints mean that comprehensive analyses of large cross-classification releases can be time consuming. Although the computational burden can be ameliorated through sampling to derive approximate SAP measures, further work is needed on producing exact measures. Far more efficient algorithms have been found for certain special cases. These cases were chosen for no other reason than the fact that they are common forms of release from the Office for National Statistics; in fact, there are obvious extensions to other cases that are not detailed here. Nevertheless the SAP method provides a risk measure for attribute disclosure. Most existing risk measures only concern identification risk. Concentrating on identification risk, at the expense of attribution risk, raises the possibility of real disclosure occurring as a result of the release of data that are considered ‘safe’ by current risk measures.

It should be noted that actions taken as a result of risk assessment via methods such as SAP will tend to impact the desirable properties of schemes such as Barnardization and random rounding. These schemes are unbiased in the sense that the expected value of the change made to a cell is zero. If the release of a given perturbed table (or set of tables) depends on whether certain cells happened to be rounded in a particular direction, then a bias is introduced.

Alternatives to the chosen measure were briefly discussed, and these could be investigated in the future. The measure based on the expected number of recovered zeros offers some computational advantages for some forms of data release. Whether these advantages extend to the various forms of release discussed here has not yet been investigated. It would be interesting to see if the results that gave rise to an efficient algorithm for rounded single table and rounded total could be extended to produce similarly efficient algorithms for the aforementioned alternative measures.

CHAPTER 4

A Graphical Modelling Approach to Key Variable Mapping

Introduction

Disclosure risk, whether it is expressed in terms of identification or attribution, clearly depends on the amount of information available to an intruder. Elliot and Dale (1999) incorporate this information through the development of intrusion scenarios, where a scenario is a description of how an intruder might launch an attack and what information they might use. Variables which the intruder might find useful for matching purposes under a given scenario are termed *key variables*.

The general assumption is that an attack will involve a target dataset which contains identifying variables such as name and address, and attempts will be made to match against anonymized data sets. The data environment is a collection of datasets to which, it is assumed, the intruder has access. Data Environment Analysis (DEA) is carried out at the University of Manchester and involves the collection and cataloguing of forms and questionnaires that result in datasets that might be available to intruders. The Key Variable Mapping System (KVMS) is an approach for identifying matching possibilities across datasets within a data environment (Elliot et al., 2010). It is a formalized approach for identifying key variables.

The existing KVMS is implemented in a spreadsheet with Visual Basic macros. This places practical limitations on the computational methods that can be employed, and on performance for large collections of datasets. It also limits the extent to which the application can be separated into data access, business logic, and presentation layers. This paper describes a more principled approach to Key Variable Mapping implemented using alternative development tools. It concentrates on the differences between the existing system and the new system, with particular emphasis on the graph-based algorithms used

in the new system. These also allow consistency checking on data entry. Information relating to the implementation and usage is contained in Appendix 1. The software is mainly implemented in the Python programming language.

Key Variable Mapping System

An overview of KVMS is provided in Elliot et al. (2010). But for the purposes of this chapter several terms must be defined in detail. Terminology might not correspond exactly with that in Elliot et al. (2010) as this treatise will necessarily be more mathematical.

Categorization – A categorization is a set of mutually exclusive and exhaustive categories for a given variable.

Code – A code is a unique identifier for a given categorization of a given variable. It is constructed as for the original KVMS with a letter denoting the type of code, followed by an integer denoting the number of categories, a dot, and another integer to disambiguate the code from other categorizations with the same number of categories.

For instance, C3.002 would indicate a categorization for a given variable with 3 categories contained within the data environment. It would be distinct from at least one other categorization of the variable within the data environment that also contains 3 categories (i.e. C3.001). Leading zeros are used to accommodate up to 999 categorizations for the same variable with the same number of categories.

Harmonization – The unique maximal categorization that can be reached via aggregation of the existing categories of two or more categorizations. Here maximal is with respect to the number of categories. For example, {England, Scotland, Wales, Northern Ireland or Non-UK} and {England or Wales, Scotland, Northern Ireland, Non-UK} would harmonize to {England or Wales, Scotland, Northern Ireland or Non-UK} rather than {England or Wales or Scotland, Northern Ireland or Non-UK}.

Similarly, a maximal harmonization is a member of a set of harmonizations that contains a

maximal number of categories with respect to the other members of the set. Codes for harmonizations are distinguished from codes contained in the data environment by using the letter H instead of C in the code.

Analysis

KVMS analysis proceeds as follows; where $|\cdot|$ denotes set cardinality and $\lfloor \cdot \rfloor$ denotes the floor function.

- Select a target dataset and a set E of datasets to match against (the data environment)
- Select a parameter, $0 \leq p \leq 1$
- For each variable v in the target dataset
 - Let S be the set containing all the elements of E that contain the variable x
 - If $|S| < |E| \cdot p$ then the output for x is null
 - Otherwise, the output for x is a maximal harmonization over the set of harmonizations for x for all combinations of size $\lfloor |E| \cdot p \rfloor$ of the elements of S

So p defines a proportion of datasets that x can be matched against, above which x might be considered a key variable. The maximal harmonization that is generated provides an indication of the risk associated with matching against x .

At first glance it might appear that many harmonizations might need to be generated to produce the above output; one for each combination of S of size $\lfloor |E| \cdot p \rfloor$. However, although there might be many possible combinations of categorizations, they tend to give rise to a relatively low number of harmonizations. Searching through the possible harmonizations in a structured way and counting the number of elements of S which can produce each harmonization is a relatively efficient method for generating the above outputs, and is the method used in the original KVMS. The new KVMS uses a still more efficient approach which will be detailed later in this chapter.

Aggregation graphs

A categorization is easily defined, but combining categorizations to find harmonizations is not quite so straightforward. The existing KVMS requires the user to specify harmonization codes and the categorizations within the data environment that can be aggregated to those codes. Data must be entered into more than one sheet and it is possible to introduce inconsistencies. The new KVMS addresses this by making data entry easier and by requiring the user to specify the relationships between categories by constructing an *aggregation graph*.

An aggregation graph for a variable X is a directed acyclic graph $G(V, E)$ where each $v \in V$ is a category of X . The children of a node v are a set of mutually exclusive categories that aggregate exactly to v . The nodes of G without a parent (root nodes) are a categorization of the relevant variable. For convenience, let us say that an aggregation graph with these properties has the *aggregation graph property*. Figure 24 shows such a graph for a hypothetical variable *Place of Residence*.

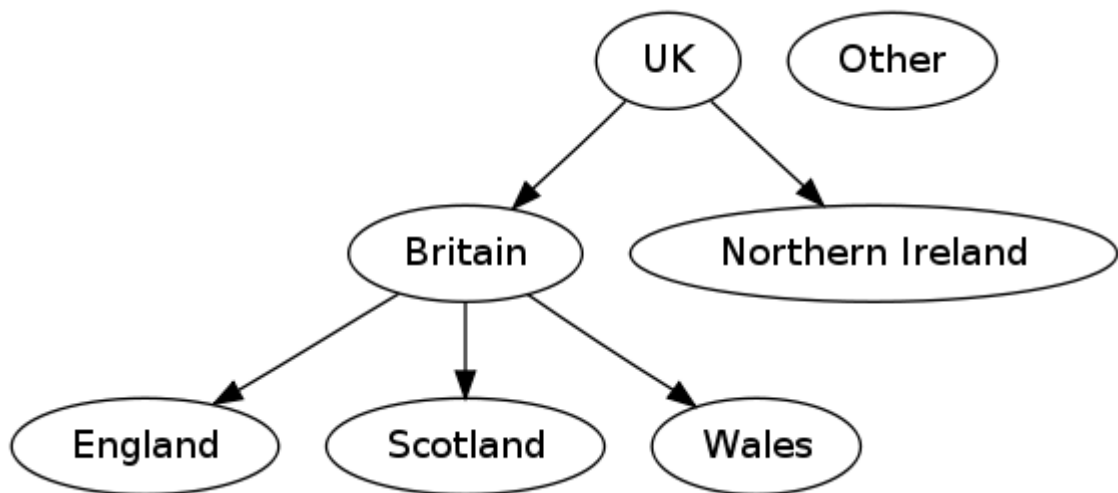


Figure 24. *An aggregation graph for Place of Residence*

This clearly has the aggregation graph property. Britain comprises of England, Scotland and Wales; the UK comprises of Britain and Northern Ireland; and the 'Other' category

ensures that the categories are exhaustive. UK is an ancestor of all categories except Other, so Other represents 'non UK'.

The aggregation graph property implies a simple way for checking that a categorization is valid, that is, comprises of a set of mutually exclusive and exhaustive categories.

A categorization C is valid with respect to an aggregation graph G if, and only if, all leaf nodes in G are reachable from exactly one node in C .

If any leaf node is not reachable, then the categorization is not exhaustive.

If a leaf node is reachable from more than one node in C , then the categories are not mutually exclusive.

An aggregation graph can be constructed incrementally. An initial categorization {England, Scotland, Wales, Other} would simply result in a graph with 4 nodes and no edges.

Accommodating a second categorization {UK, Other} would require England, Scotland and Wales to be reachable from UK and the addition of Northern Ireland to ensure that UK was the union of its children. It also implies that Other shifts its meaning from 'non Britain' to 'non UK'. Note that Britain and Northern Ireland appear in neither categorization. Britain might be added in the expectation that future categorizations might contain it. Northern Ireland had to be added to preserve the aggregation graph property. Note: in the new KVMS application consistency checks are performed. Once 'Other' has been added to a categorization with the meaning 'non UK' then it must continue to mean 'non UK'. So 'Other' would be a poor name for a category.

Wherever categories overlap, such as UK and Britain, extra categories must be added so that a valid graph can be constructed. This is most easily demonstrated with a variable on an interval scale, categorized into subintervals.

Assume Age is categorized as {'0-18', '18-infinity'} in one dataset and as {'0-16', '16-21', '21-infinity'} in a second dataset. Clearly '0-16' must be a descendant of '0-18', but one or more mutually exclusive categories that aggregate to '16-18' must also be descendants of '0-18'. For interval scaled variables it is possible to construct all necessary categories by

constructing a sorted list of the lower and upper bounds and constructing intervals from all adjacent pairs of bounds. These intervals become leaf nodes, and all categories that are not already leaf can be added as nodes and their children found by finding the relevant sequence of nodes from the list of leaf nodes. This produces a very shallow graph, and other algorithms can be used to produce more pleasingly laid out graphs with fewer edges. Although variables which are not on interval scales cannot have their overlaps generated automatically, the same principle can be used to construct their aggregation graphs. Nodes are produced for all category overlaps, the maximal set of mutually exclusive and exhaustive categories (maximal valid categorization for the set of added categories) is the set of leaf nodes, and other nodes can be added as parents of the relevant leaf nodes.

Figure 25 shows the aggregation graph for the simple Age example described above. The set of leaf nodes of an aggregation graph will always represent a valid categorization. Here that is {'0-16', '16-18', '18-21', '21-infinity'}. But it is clear that {'0-18', '18-infinity'} and {'0-16', '16-21', '21-infinity'} are also valid categorizations as, in both cases, each leaf node is reachable from exactly one node in the categorization. This is only to be expected, as these were the categorizations used to construct the graph. But assuming we added a new dataset to the data environment with categorization {'0-18', '18-21', '21-infinity'}. It is a trivial exercise (a simple algorithm) to check that this new categorization is also valid.

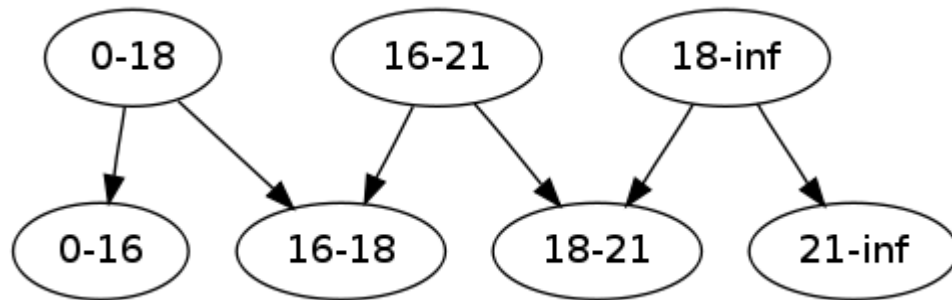


Figure 25. *An aggregation graph for Age generated using a simple algorithm*

In the new KVMS the user is responsible for creating and updating an aggregation graph for each variable within the data environment. An editor is provided which allows nodes and edges to be added and removed via a simple graphical interface. Restrictions are

placed upon the edits, so that an edited graph remains consistent with previous incarnations. For example, the user is not permitted to delete any existing nodes or edges, or add new edges from existing non-leaf nodes. Deleting nodes or edges would make consistency checking difficult. Given that deletion of existing nodes and edges is not permitted, then adding new edges from a non-leaf node would imply that the existing children do not aggregate to their parent. Editing of an aggregation graph is only required when a new categorization is encountered which contains categories that are not already present in the graph.

A pragmatic approach would be to work out an aggregation graph for a variable and a given set of datasets, then add it to the system once. Then specifying a categorization for a given dataset would simply be a case of selecting the appropriate nodes, rather than editing as described above before selecting the nodes. Constructing an aggregation graph incrementally is likely to represent more work.

Categorizations and harmonization

A valid categorization for a variable is a set of nodes in the relevant aggregation graph such that each leaf node can be reached from exactly one node in the categorization. Thus any categorization can be expressed as a partition of the set of leaf nodes in an aggregation graph. The categorization {Britain, Northern Ireland, Other} from Figure 24 can be represented as the partition, $\{\{\text{England, Scotland, Wales}\}, \{\text{Northern Ireland}\}, \{\text{Other}\}\}$. The categorization {UK, Other} from Figure 24 can be represented as the partition, $\{\{\text{England, Scotland, Wales, Northern Ireland}\}, \{\text{Other}\}\}$. In the new KVMS a categorization is specified via the aggregation graph editor described above. The categorization nodes are selected and it is checked that each leaf node is reachable from exactly one selected node.

If every element of a partition α is a subset of some element of a partition β , then α is termed a *refinement* of β .

If α is a refinement of β then each element of β is the union of one or more elements of α .

Thus categories in the categorization corresponding to α can be aggregated (using set union) to produce the categorization represented by β . If α is not a refinement of β then there is some element of α that is not a subset of some element of β and any aggregation of α (including α itself) will have some element that is not in β .

Thus the harmonization, H , of two categorizations, $C1$ and $C2$, can be expressed as set operations on their set partition representations. The harmonization is the maximal set partition for which $C1$ and $C2$ are refinements of H . Each leaf node, v , is contained in exactly one element of $C1$ and one element of $C2$, and must be contained in exactly one element of H . Thus the element in H which contains v must be a superset of the element in $C1$ containing v and the element of $C2$ containing v . If not, then $C1$ and $C2$ are not refinements of H . If this holds for all leaf nodes, then all elements of $C1$ and $C2$ are subsets of some element of H and they are therefore refinements of H .

So performing set unions on the elements of $C1$ and $C2$ which have non-empty intersection will produce their harmonization, H . The new KVMS implements these operations by mapping them to graph operations.

Each categorization is represented as an undirected graph (a categorization graph), such that the connected components are the elements of the set partition representation of the categorization. Thus $\{\{\text{England, Scotland, Wales}\}, \{\text{Northern Ireland}\}, \{\text{Other}\}\}$ might be represented as in Figure 26, and $\{\{\text{England, Scotland, Wales, Northern Ireland}\}, \{\text{Other}\}\}$ might be represented as in Figure 27.

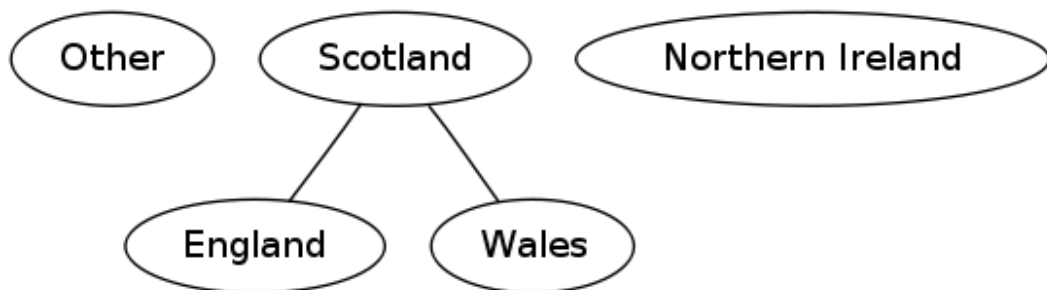


Figure 26. The categorization $\{\{\text{England, Scotland, Wales}\}, \{\text{Northern Ireland}\}, \{\text{Other}\}\}$

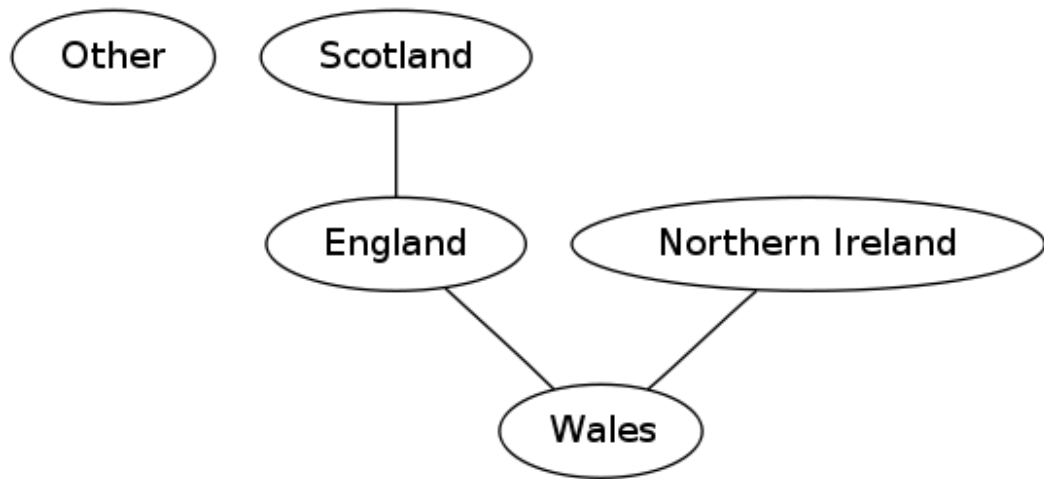


Figure 27. *The categorization $\{\{England, Scotland, Wales, Northern Ireland\}, \{Other\}\}$*

The graphs in Figures 26 and 27 contain the same nodes (as they are both valid categorizations of the same variable). The graph sum operation on such graphs constructs a new graph with the same nodes and adds all edges from the summed graphs.

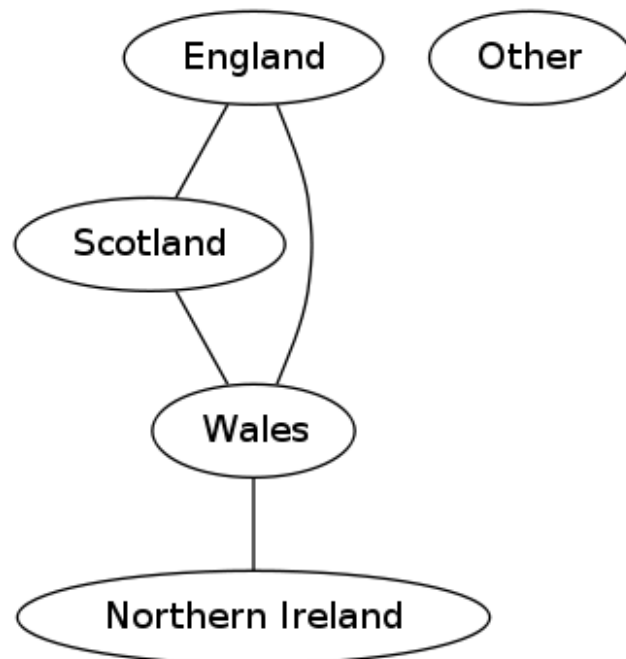


Figure 28. *The harmonization of the graphs in Figures 26 and 27, $\{\{England, Scotland, Wales, Northern Ireland\}, \{Other\}\}$*

Note that the harmonization of these categorizations is the same as the categorization shown in Figure 27. This is because the categorization in Figure 26 is a refinement of the categorization in Figure 27. Also note that the graph in Figure 28 is not the same as the graph in Figure 27. Yet they represent the same categorization because they contain the same connected components.

Connected components are identified via breadth first search. A node is selected and marked as visited. Then its nearest neighbours are marked as visited. Then their unvisited nearest neighbours are visited, and so on, until all the nodes in the component have been visited (and identified as belonging to the same connected component). Performing breadth first searches until all the graph nodes have been visited identifies all the connected components. The KVMS never changes these categorization graphs once they are created, and so equality can be tested on the basis of hash values generated on graph creation and which depend only on the connected components.

Harmonization graphs

The refinement relationship admits a partial ordering on the possible set partitions, and therefore on the set of categorizations for a given variable within a data environment. The set of categorizations for a given variable within a data environment and their possible harmonizations can be used to construct a directed acyclic graph $G(V, E)$ such that a path $v \rightarrow w$, $\{v, w\} \subset V$ implies that w is a refinement of v . The existing KVMS allows the user to manually construct such a graph, although the graph is not used for any purpose within the KVMS. The new KVMS constructs harmonization graphs automatically and imposes additional restrictions on their structure.

Let $H(\cdot)$ denote the function which returns the harmonization of its arguments.

If w is a refinement of v , then w is said to be finer than v , and v is said to be coarser than w . If w is finer than v , and v is not equal to w , then w is said to be strictly finer than v , and v is

said to be strictly coarser than w .

In order to be useful for looking up harmonizations the graph must possess the following two properties:

1. An edge (v,w) implies that w is strictly finer than v , and that no descendant of v is also strictly finer than w .
2. There exists a graph node for each $H(x_1, \dots, x_N)$ where $\{x_1, \dots, x_N\}$ is a subset of the set of categorizations, C , for the relevant variable contained in the data environment.

Thus there will be a node in the graph for each possible harmonization of the categorizations in C . Property 1 minimizes the number of edges in the graph and is a necessary property for the correctness of certain algorithms which follow.

The size of the powerset (set of subsets) of C increases exponentially with the size of C , so generating all possible harmonizations can be computationally expensive. In practice the number of distinct harmonizations is much smaller than the powerset of C , so an algorithm is developed that ensures that all the necessary harmonizations are contained in the graph without having to generate a harmonization for each member of the powerset of C .

The algorithm adds categorizations from C to a DAG G in an arbitrary order, ensuring that the graph structure is updated to produce a valid harmonization graph (with respect to the subset of the data environment containing the added categorizations) on each node addition. Within the new KVMS a harmonization graph $G(V, E)$ is valid if a path $v \rightarrow w$, $\{v, w\} \subset V$, exists if, and only if, w is strictly finer than v and no other descendant of v is also strictly finer than w ; and if it contains a node for the harmonization of each possible combination of the nodes in C .

For any given pair of distinct categorizations $\{v,w\}$ the refinement relationship can be tested thus:

$H(v,w) = v$ implies that w is strictly finer than v , and that v is strictly coarser than w .

Assume that there exists an existing harmonization graph G and we wish to add a new node u . Consider the subgraph of G induced by all the nodes in G that are coarser than u . The leaf nodes of this subgraph are exactly the nodes that must be parents of u under property 1. Now consider the subgraph of G induced by all the nodes of G that are finer than u . The root nodes of this graph are exactly the nodes that must be children of u under property 1.

Having identified its parents and children we can insert the node u into the graph and add the necessary edges. The insertion of u clearly does not remove any of the existing refinement relationships as any paths between nodes that existed before still exist. It does not add any paths between nodes that do not reflect valid refinement relationships because its children are refinements of u and u is a refinement of its parents. However, where edges (v,u) and (u,w) have been added and there already exists an edge (v,w) , then property 1 does not hold for (v,w) . But the refinement relationship between v and w is provided by the path $v \rightarrow u \rightarrow w$ as are all relationships that were implied by (v,w) . So the situation can be rectified by simply removing (v,w) .

The remaining issue is ensuring that property 2 holds. Assume that the equivalent property holds for the current graph with respect to the powerset of the set of already added members of C . Then ensuring that the graph contains a node $H(v,u)$ for each $v \in V$ will ensure that after addition of u the equivalent property will hold for the powerset of all the members of C added up to, and including, u . Thus after addition of all members of C property 2 will hold.

Assume a node u is added to the graph according to the above scheme and that the node v is an ancestor of u in the resulting graph. Then $H(v,u) = v$, and $H(v,u)$ already exists in G . Assume that v is a descendant of u in G . Then $H(v,u) = u$, and $H(v,u)$ already exists in G . If v is neither an ancestor nor a descendant of u there is no simple graphical approach to determining if $H(v,u)$ already exists in G or is to be added from C . So for each such node $H(v,u)$ is calculated, and if it is not either in G or a member of C it must be added to a set of nodes to be added to G .

The following pseudocode provides a very high level description of the algorithm. Comments are preceded by a #. A much more detailed version appears in Appendix 2,

along with a discussion of its computational complexity.

```
def harmonization_graph(categorizations):
    # create an empty harmonization graph
    G = empty directed graph
    # create a set to hold categorizations
    # and harmonizations to be added to G
    C = empty set
    for c in categorizations:
        add c to C
    while C is not empty:
        # get the next graph node, u, and add it to G
        pop u from C
        add u to G
        # find the parents and children of u
        # using the criteria described above
        parents = parents of u
        children = children of u
        # add necessary edges to G
        for each parent in parents:
            add the edge (parent, u) to G
        for each child in children:
            add the edge (u, child) to G
        # remove any edges from nodes in parents
        # to nodes in children (i.e. ensure property 1 holds)
        for parent in parents:
            for child in children:
                if the edge (parent, child) is in G:
                    remove edge (parent, child) from G
        # generate any new harmonizations required
        # to ensure that property 2 holds for the final graph
        for each node v in G:
            h = H(v, u)
            if h != v and h != u:
                # v is neither ancestor nor descendant of u
                if h is not in either G or C:
                    add h to C
    return G
```

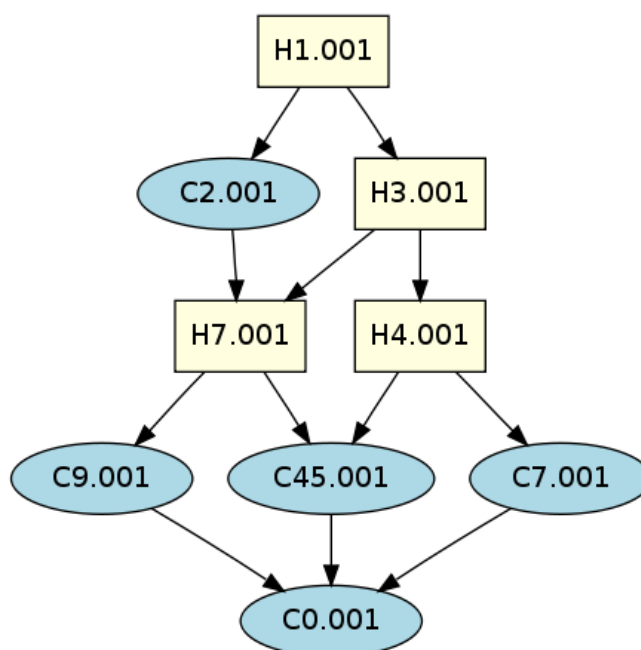


Figure 29. *Harmonization graph for Age*

Figure 29 is the harmonization graph for the variable Age from a data environment containing almost 200 datasets, the vast majority containing a variable for Age. The corresponding aggregation graph contains 103 nodes in total with 61 leaf nodes. Yet there are only 5 distinct codes for Age within the data environment, and only 4 additional harmonization codes required to construct the harmonization graph. Codes that appear within the data environment are shown as blue ellipses, to distinguish them from the generated harmonization codes which are shown as yellow rectangles. The two types of code are also distinguished by their leading letter. As previously indicated the convention for distinguishing different categorizations of the same type and with equal numbers of categories is to use the number of categories and a unique (zero padded) integer separated by a full stop. Thus C45.001 is a code, for a categorization of Age, which is found within the data environment and has 45 categories. If there was another categorization of the same type and with the same number of categories it would be assigned the identifier C45.002. The C0.001 category is a special category corresponding to datasets containing 'Date of Birth' or where there is no specified categorization. Forms where a respondent is asked to write in a value result in the same form of code. It is assumed that the user could provide an arbitrary level of detail, and that harmonization with any other code c will reproduce c .

Although the code seems to imply 0 categories, it actually means 0 specified categories. In reality it is an arbitrarily high number of categories, and has to be special cased. Note that any harmonization graph that contains an arbitrary precision node will have that node as a single leaf node. All harmonization graphs will have a single root node corresponding to a zero precision code (a single category).

Figure 29 shows that there are only 5 distinct categorizations for age across all the datasets within the data environment, and one of these allows the user to provide an arbitrary degree of precision. If trying to match on Age between two data sets with categorizations C9.001 and C45.001, then any potential match can be mapped to one of 7 categories. These categories are given by the unique leaf node in the graph intersection of their ancestral graphs, H7.001. Similarly, trying to match across C9.001, C45.001 and C7.001 will allow any matches (across all three datasets) to be mapped to one of three categories, given by H3.001. Whether these matching possibilities have significant implications will depend on how many datasets have these specific codes and the categories in their harmonization. The following analysis method attempts to exploit this information to produce an indication of the likelihood that a variable will be employed as a key variable.

Analysis

The analysis procedure in the existing KVMS was outlined in the introduction section. Essentially a prevalence parameter is specified and this is used to calculate an integer parameter, N . Then a maximal harmonization $h = H(c, x_1, \dots, x_N)$ is sought, where x_1, \dots, x_N are categorizations of the relevant variable from any N distinct datasets in the data environment and c is the categorization within the target dataset. Note: the categorizations might not be distinct.

Naively iterating through all the combinations of datasets of a given size that contain the relevant variable from the data environment can be costly. The existing KVMS uses a more sensible approach based on searching through the possible harmonizations, from largest to smallest, until it finds a harmonization (if any) that is coarser than the categorizations of the relevant variable from at least N datasets.

A more focussed search can be employed using the harmonization graph for the variable in question, constructed using all the data sets in the data environment. Here the data environment might be a subset of some larger data environment, perhaps conditioning the analysis on geographical location or some other dataset attribute.

Assume we have a valid harmonization graph, G , (note: a harmonization graph constructed from a superset of the data environment is still valid). The data environment can be scanned to produce a mapping of graph nodes (categorizations) to counts, so that finding the number of datasets with a given categorization of the variable is a simple lookup.

Once the mapping of categorizations to counts is performed the nodes of G are processed in order subject to the following constraint: each node must only be processed after its children have been processed. An appropriate ordering of the nodes can be generated by the well-known postorder depth first search algorithm which can be found in many introductory texts (e.g. Cormen et al., 2001).

The first step of processing a node is identifying its set of descendants. As the node's children have already been processed, this is simply a case of calculating the set union of the nodes' children and their sets of descendants. Once the descendants have been identified it is simple to calculate the number of datasets within the data environment that harmonize to the current node. The counts for the node and its descendants are simply summed. If this sum is greater than or equal to N , then the current node is a candidate solution. That is, the node is the harmonization of at least N categorizations for the relevant variable contained in the dataset. (Note that these are not necessarily distinct categorizations, which is why the mapping was required. The harmonization graph contains the distinct categorizations, but we also need to know how many times each occurs within the data environment.)

Once all the candidate solutions have been identified, then it is simply a matter of choosing one with a maximal number of categories. That is the analysis output for the variable in question. If no candidates are identified, then the result is null.

It should be noted that the cost of search can be reduced by exploiting the properties of

harmonization graphs. Only c and its ancestors can be candidate solutions as the solution is the harmonization of c and a set of other observed categorizations. Therefore only the subgraph induced by c and its ancestors need be searched. Also, a node in a harmonization graph contains fewer categories than any of its children. So the ancestors of a candidate solution are not solutions as they are not maximal. Pseudocode and a more detailed discussion of the algorithm are contained in Appendix 2.

Summary

A number of differences between the existing and new Key Variable Mapping Systems have been detailed. The most significant difference is the adoption of aggregation graphs to handle overlapping categories. In the original KVMS the user was expected to specify the harmonization codes and which categorizations harmonized to them. The new KVMS only requires that the user specify a categorization, and that the categorization is consistent with the corresponding aggregation graph. This is less error prone and guards against introducing inconsistencies. In fact, the system largely prevents inconsistencies by limiting how an existing aggregation graph can be edited. A general procedure for constructing aggregation graphs has been presented, and this provides an algorithm for constructing aggregation graphs automatically for a given set of categorizations if the variable is on a numeric interval scale.

It has been demonstrated that harmonization of categorizations can be viewed as a sequence of set operations on set partitions. This gives rise to a representation of a categorization as a set partition over the set of leaf nodes in an aggregation graph. This in turn gives rise to a graphical representation, where each leaf node in the aggregation graph has a corresponding node in an undirected graph, and each partition element is a connected component in the graph. This provides an attractive object model, where harmonization becomes a simple graph sum operation.

Elliot et. al. (2010) show how the relationships between categorizations and harmonizations can be shown in a directed graph. However, Elliot et. al. (2010) do not provide a formal definition of harmonization graphs and they are not constructed or used in

the existing KVMS. In this work a formal definition has been provided along with an algorithm for constructing a harmonization graph from a set of categorizations. All the relevant harmonizations are generated automatically. Furthermore, it has been shown how the use of such a graph can be used to generate the KVMS outputs in a particularly efficient manner.

Using the new KVMS the user is responsible for providing valid categorizations, editing an aggregation graph as needed. After that, the system takes care of everything else. This is more straightforward than the existing KVMS. Implementing the system in a high level programming language allowed the separation of data access, business logic, and presentation layers. Thus the new KVMS is more easily maintained and extended. It also offers significant performance gains. Analyses conducted on a data environment with nearly 200 datasets complete almost instantaneously, whereas there can be an appreciable delay when conducting analyses with the existing KVMS.

CHAPTER 5

An Evaluation of Strategies for Matching Population and Sample Units

Introduction

For disclosure control purposes data are often released as sample data, even if population data are available. Armed with sample data an intruder might attempt to make inferences about population units by matching against the sample, or vice versa. There are many strategies available to a data intruder, with varying probabilities of success and degrees of difficulty. The probabilities of a correct match for some strategies have been used as measures of risk (see e.g. Elamir and Skinner, 2006). This chapter examines a number of strategies, their probabilities of a correct match, their costs, and their appropriateness as risk measures.

In order to protect against disclosure it is usual to suppress (not publish) information regarding obvious identifiers such as name and address. Although a particular name and address does not necessarily correspond to a single individual in a population, it is generally felt that the publication of such information allows published records to be correctly matched with population units too easily. However, the suppression of obvious identifiers is not necessarily enough to prevent such matches being made with a high (personal) probability of correctness.

The variables that are contained within published records are often partitioned into *key* variables and *target* variables. It is assumed that an intruder will be able to condition on key variables and is intent on discovering the relevant values of the target variables. The appropriate partitioning of the variables will generally depend on the type of intruder and the resources available (Elliot and Dale, 1999).

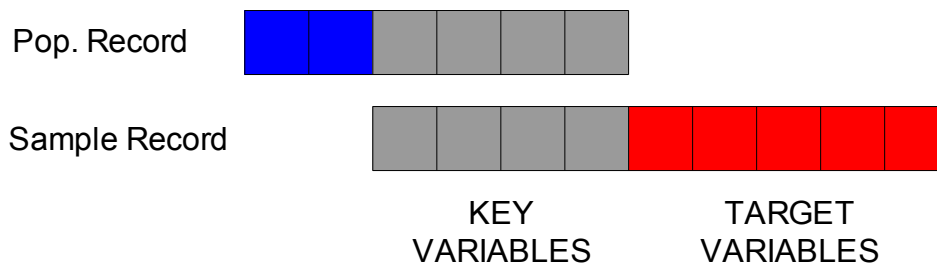


Figure 30. *Matching between sample and population on key variables*

The pairing of an individual in the population with a sample record with identical values on the key variables is known as a match. If the sample record is the record corresponding to the population unit, then the match is termed correct. A known correct match allows the levels of the target variables to be inferred for the population unit with certainty (assuming the data are accurate and up to date). With sample data it is not usually possible for an intruder to knowingly produce a correct match, although there are degenerate cases such as occupation = Prime Minister. But whatever personal probability of a correct match the intruder has, is a lower bound on the personal probability that the published values of the target variables hold for the population unit.

A number of risk measures have been proposed. These tend to be concerned with either population uniqueness (the general idea being that population counts of 1, on the key variables, will imply that a match against a sample unit will be correct) or the probability of generating a correct match.

Notation:

Assume we have a population table containing N individuals and a corresponding sample table containing n individuals.

X	set of key variables
$j = 1 \text{ to } J$	distinct values / equivalence classes / cells for X
$i = 1 \text{ to } N$	population units

X_i	value of X for unit i
N	population size
n	sample size
$[.]$	Iverson brackets
$F_j, j = 1 \text{ to } J$	population frequencies
$f_j, j = 1 \text{ to } J$	sample frequencies

Iverson brackets are used in the manner detailed in Knuth (1992), where

$$[P] = \begin{cases} 1 & \text{if } P \text{ is true} \\ 0 & \text{if } P \text{ is false} \end{cases}$$

and resulting zeros are “very strong”, meaning they annihilate anything by which they are multiplied. So,

$$\frac{x}{0}[P] = 0 \text{ if } P \text{ is false.}$$

Assume throughout the rest of this chapter that both N and n are greater than 0.

Existing measures

$Pr(PU)$ – the proportion of population units which are population unique (Bethlehem et al., 1990),

$$Pr(PU) = \frac{\sum_j [F_j = 1]}{N}.$$

This measure has been criticized for being too optimistic (Skinner and Elliot, 2002) as it assigns an element of risk to population units that have no possible match in the sample.

$Pr(PU|SU)$ – the proportion of sample uniques which are also population unique (e.g. Fienberg and Makov, 1998),

$$Pr(PU|SU) = \frac{\sum_j [f_j=1][F_j=1]}{\sum_j [f_j=1]} \left[\sum_j [f_j=1] > 0 \right].$$

However, this could also be criticized as being too optimistic as it is a lower bound on the probability that a proposed match against a sample unit is correct (which itself is only a lower bound on the probability that a target variable value applies to the relevant population unit).

The measure favoured by Skinner and Elliot is the probability of a correct match given a unique match, $Pr(CM|UM)$,

$$Pr(CM|UM) = \frac{\sum_j [f_j=1]}{\sum_j F_j [f_j=1]} \left[\sum_j [f_j=1] > 0 \right].$$

This measure implies a particular strategy being employed by the intruder. The intruder searches through the population until a match against *any* sample unique is found. Under the assumption that at each point in the search all undiscovered population units are equally likely to be the next one found, the above gives the probability of a correct match. In fact, this assumption of an unfocused search is used throughout this chapter. However, there are alternative strategies that give rise to other probabilities of a correct match, with various degrees of work required on the part of the intruder.

Skinner and Elliot (2002) also briefly present the following measure which stems from the strategy of searching through the population for a match against a specific, but randomly selected sample unique,

$$Pr(CM|SU) = \frac{\sum_j \frac{1}{F_j} [f_j=1]}{\sum_j [f_j=1]} \left[\sum_j [f_j=1] > 0 \right].$$

Simple attack strategies

There are a number of strategies an intruder might employ in order to identify an individual. For instance, an intruder might set out to find a member of the population and match against the sample; or, might take some sample unit and try to find a match in the population. An intruder might attempt to increase the probability of a correct match by only selecting targets with certain attributes. The probability of a correct match for a given strategy can be used as a measure of risk. The following demonstrates that some strategies dominate others, and that some obvious strategies are not as effective as they may, at first, seem.

The strategies discussed in this chapter do not exploit prior information that might be held by the intruder. Thus the joint distribution of the variables is irrelevant, and the equivalence classes can be thought of as an unordered list of categories indexed 1 to J .

A strategy is defined in terms of a pair of statements, the first being whether matching is attempted from population to sample, or vice versa, and the second, P , describing a condition under which units constitute potential targets. Matching from population to sample involves finding a population unit that satisfies P and matching against the sample. Matching from sample to population involves randomly selecting a sample unit that satisfies P and searching the population for a match.

A given strategy will imply marginal probabilities of choosing given units for matching, and a probability of a correct match for each unit. The probability of a correct match for a given strategy is given by,

$$Pr(cm) = \sum_i Pr(x_i) Pr(cm|x_i)$$

where x_i denotes the i th population unit and $Pr(x_i)$ is the probability of choosing x_i for matching.

As the units within an equivalence class are indistinguishable the resulting expressions generally involve summations over equivalence classes rather than population units.

Matching from population to sample

All potential targets are chosen from the population with equal probability. A target chosen from a non-sampled equivalence class cannot be matched against a sample unit. For any chosen unit from a sampled equivalence class j the probability of a correct match against a chosen sample unit is simply $1/F_j$. As there are exactly F_j units in the equivalence class, then the sum of probabilities for a non-empty sample cell is trivially 1. Thus,

$$Pr(cm) = \frac{\sum_j [P]}{\sum_j F_j [P]} \sum_j F_j [P] > 0 .$$

Matching from sample to population

All potential targets are chosen from the sample with equal probability. A target chosen from a sampled equivalence class j has a probability of a correct match against a chosen sample unit of $1/F_j$. There are f_j potential targets within the chosen equivalence class. Thus,

$$Pr(cm) = \frac{1}{\sum_j f_j [P]} \sum_j \frac{f_j}{F_j} [P] \left[\sum_j f_j [P] > 0 \right] .$$

Specific strategies

Strategy 1. An intruder matches from sample to population using all cells

Equation 1.
$$Pr(cm) = \frac{1}{n} \sum_j \frac{f_j}{F_j} [f_j \neq 0]$$

Strategy 2. An intruder matches from population to sample using all cells

Equation 2.
$$Pr(cm) = \frac{1}{N} \sum_j [f_j \neq 0]$$

or,

$$Pr(cm) = \frac{\text{no. of non-zero cells in sample table}}{N}.$$

However, it is reasonable to suspect that an intruder will at least continue searching for a population unit until there is a possible match.

Strategy 3. An intruder matches from population to sample using all sampled cells

Equation 3.
$$Pr(cm) = \frac{\sum_j [f_j \neq 0]}{\sum_j F_j [f_j \neq 0]}$$

or,

$$Pr(cm) = \frac{1}{\text{mean size of sampled population cells}} .$$

Strategy 4. An intruder matches from sample to population using all sample unique cells

Equation 4.

$$Pr(cm) = \frac{\sum_j \frac{1}{F_j} [f_j = 1]}{\sum_j [f_j = 1]} \left[\sum_j [f_j = 1] > 0 \right]$$

Strategy 5. An intruder matches from population to sample using all sample unique cells

Equation 5.

$$Pr(cm) = \frac{\sum_j [f_j = 1]}{\sum_j F_j [f_j = 1]} \left[\sum_j [f_j = 1] > 0 \right]$$

It can be shown that the probability given by Equation 4 is equal to that for Equation 5. if all the relevant F_j (with corresponding $f_j = 1$) are equal. But it is worth generalizing to sample cells of a given size, r , first.

Strategy 6. An intruder matches from sample to population using all sample cells with count r

$$Pr(cm) = \frac{\sum_j \frac{r}{F_j} [f_j = r]}{\sum_j r [f_j = r]} \left[\sum_j [f_j = r] > 0 \right]$$

and as the r 's cancel,

Equation 6.

$$Pr(cm) = \frac{\sum_j \frac{1}{F_j} [f_j = r]}{\sum_j [f_j = r]} \left[\sum_j [f_j = r] > 0 \right],$$

or, if $\left[\sum_j [f_j = r] > 0 \right],$

$$Pr(cm) = \frac{1}{\text{harmonic mean size of sampled (size=r) population cells}}.$$

Strategy 7. An intruder matches from population to sample using all sample cells with count r

As for Strategy 2 the sum of probabilities for each sampled cell is equal to 1. But here averaging is over only those population units corresponding to sample cells of size r . So N in Equation 2 is replaced by the number of population units in equivalence classes with corresponding sample counts of r ,

Equation 7.

$$Pr(cm) = \frac{\sum_j [f_j = r]}{\sum_j F_j [f_j = r]} \left[\sum_j [f_j = r] > 0 \right],$$

or, if $\left[\sum_j [f_j = r] > 0 \right],$

$$Pr(cm) = \frac{1}{\text{mean size of sampled (size=r) population cells}}.$$

Dominant strategies

An obvious question is whether there exists pairs of strategies $\{A, B\}$ such that

$$Pr_A(cm) \geq Pr_B(cm) \quad \text{for all possible sets of population and sample frequencies.}$$

The finite form of Jensen's inequality (Jensen, 1906) for a continuous convex function $\phi(\cdot)$ in a given interval, and positive weights a_j is given in Equation 8.

Equation 8.
$$\frac{\sum_j a_j \phi(z_j)}{\sum_j a_j} \geq \phi\left(\frac{\sum_j a_j z_j}{\sum_j a_j}\right)$$

For $\phi(z) = \frac{1}{z}$,

$$\frac{\sum_j \frac{a_j}{z_j}}{\sum_j a_j} \geq \frac{\sum_j a_j}{\sum_j a_j z_j}$$

and,

$$\frac{\sum_j \frac{f_j}{F_j}}{\sum_j f_j} \geq \frac{\sum_j f_j}{\sum_j f_j F_j}$$

with equality when the F_j are equal. The above holds as long as all sampled cells are positive. Any attempt to match against a sample cell of zero has zero probability of success and we have equality.

Thus

$$\frac{1}{\sum_j f_j[P]} \sum_j \frac{f_j}{F_j} [P] \left[\sum_j f_j [P] > 0 \right] \geq \frac{\sum_j [P]}{\sum_j F_j [P]} \left[\sum_j F_j [P] > 0 \right]$$

holds if $f_j = r \forall j$ and the strategies are such that $\left[\sum_j f_j |P| > 0 \right] = 1$ if $\left[\sum_j F_j |P| > 0 \right] = 1$. This holds for strategies 6 and 7 as $|P| = [f_j = r]$ and holds trivially for strategies 4 and 5. Jensen's inequality implies that equality will exist when the F_j are all equal. Equality will also exist if $r = 0$ or if there are no sample cells equal to r , when both probabilities are 0. The dominance of strategy 4 ($\text{Pr}(\text{CM}|\text{SU})$) over $\text{Pr}(\text{PU}|\text{SU})$ is stated in Skinner and Elliot (2002).

Of the 7 listed strategies there is only one other dominance relationship. Strategy 3 dominates strategy 2 as the only difference between them is that strategy 3 avoids selecting targets that cannot be matched. The two strategies are equal when all the equivalence classes are sampled. Counterexamples can be found for all other pairs of listed strategies.

Another obvious question is whether strategies which choose target cells with low sample counts dominate similar strategies which choose target cells with higher sample counts. Strictly speaking the answer is no, because the probability of a correct match is zero if there are no sample cells with the desired count. Even when the probabilities for two distinct sample counts are both non-zero there is no dominance in the sense used here, although a Bayesian approach conditioning on the observed sample counts might generate dominance relationships in a different sense. A data intruder might well find that concentrating on low counts tends to yield better results.

Search costs

The measures discussed above have, wherever relevant, assumed that the intruder performs a completely unfocused search of the population. Essentially, it is as if the population units were contained in a list, with each unit equally likely to occupy any position, and with the intruder searching through the units in list order. Under this assumption it is possible to produce correct match probabilities for various strategies. However, it is also possible to calculate how costly those strategies would be under the same assumption.

Searching the sample data is assumed to take zero time, so the cost of a strategy is

expressed solely in terms of the population search. An obvious cost measure is the mean number of population units that would need to be searched before finding a potential match. The calculation is essentially a fairly standard urn problem.

Assume an urn contains a total of N balls with M matches and balls are drawn without replacement. Then the number of balls drawn, T , until the first match follows the following negative hypergeometric mass function,

$$Pr(T=t) = \frac{\binom{N-t}{M-1}}{\binom{N}{M}}.$$

The expectation is given by,

$$E(T) = \frac{N+1}{M+1}.$$

However, it might be reasonable to expect that the search would become harder over time. Although the order in which the population units are found might still be essentially random, finding new units might become progressively more difficult as the number of undiscovered units decreases. This corresponds to sampling with replacement.

Assume an urn contains a total of N balls with M matches and balls are drawn with replacement. Then the number of balls drawn, T , until the first match follows the following negative binomial mass function,

$$Pr(T=t) = \left(1 - \frac{M}{N}\right) \left(\frac{M}{N}\right)^{t-1}.$$

The expectation is given by,

$$E(T) = \frac{N}{M}.$$

A strategy which matches from population to sample will involve a single search for a candidate population unit. The number of candidates will depend on the strategy, the population distribution and which cells have been sampled. A strategy which attempts to match from sample to population will have a number of possible searches depending on the sample unit chosen. In this case the expectation can be expressed as an average over the population units, where some of the $Pr(x_i)$ will necessarily be zero because the unit is not sampled and others might be zero because of the strategy,

$$E(T) = \sum_i Pr(x_i) E(T_i).$$

Matching from population to sample

The probability of a correct match is reproduced below.

$$Pr(cm) = \frac{\sum_j [P]}{\sum_j F_j[P]} \left[\sum_j F_j[P] > 0 \right]$$

All eligible population units are selected with equal probability.

Without replacement,

$$E(T) = \frac{N+1}{1 + \sum_j F_j[P]} \left[\sum_j F_j[P] > 0 \right].$$

With replacement,

$$E(T) = \frac{N}{\sum_j F_j[P]} \left[\sum_j F_j[P] > 0 \right].$$

Matching from sample to population

The probability of a correct match is reproduced below.

$$Pr(cm) = \frac{1}{\sum_j f_j[P]} \sum_j \frac{f_j}{F_j} [P] \left[\sum_j f_j[P] > 0 \right]$$

All eligible sample units are selected with equal probability and the expected costs are the same for each member of the same equivalence class.

Without replacement,

$$E(T) = \frac{1}{\sum_j f_j[P]} \sum_j f_j \frac{N+1}{1+F_j} [P] \left[\sum_j f_j[P] > 0 \right]$$

and

$$E(T) = \frac{N+1}{\sum_j f_j[P]} \sum_j \frac{f_j}{1+F_j} [P] \left[\sum_j f_j[P] > 0 \right].$$

With replacement,

$$E(T) = \frac{1}{\sum_j f_j [P]} \sum_j f_j \frac{N}{F_j} [P] \left[\sum_j f_j [P] > 0 \right]$$

and

$$E(T) = \frac{N}{\sum_j f_j [P]} \sum_j \frac{f_j}{F_j} [P] \left[\sum_j f_j [P] > 0 \right].$$

Cost dominance

By Jensen's inequality,

$$\frac{\sum_j \frac{a_j}{z_j}}{\sum_j a_j} \geq \frac{\sum_j a_j}{\sum_j a_j z_j}$$

$$\frac{\sum_j \frac{f_j}{1+F_j}}{\sum_j f_j} \geq \frac{\sum_j f_j}{\sum_j f_j (1+F_j)}$$

$$\frac{N+1}{\sum_j f_j} \sum_j \frac{f_j}{1+F_j} \geq \frac{N+1}{1 + \sum_j f_j F_j / \sum_j f_j}$$

with equality when all the F_j are equal.

Thus, sampling without replacement,

$$\frac{N+1}{\sum_j f_j[P]} \sum_j \frac{f_j}{1+F_j} [P] \left[\sum_j f_j[P] > 0 \right] \geq \frac{N+1}{1+\sum_j F_j[P]} \left[\sum_j F_j[P] > 0 \right]$$

holds if the strategies are such that $\left[\sum_j f_j[P] > 0 \right] = 1$ if $\left[\sum_j F_j[P] > 0 \right] = 1$. Equality holds when there is only a single sampled equivalence class.

Sampling with replacement,

$$\frac{\sum_j \frac{f_j}{F_j}}{\sum_j f_j} \geq \frac{\sum_j f_j}{\sum_j f_j F_j}$$

$$\frac{N}{\sum_j f_j} \sum_j \frac{f_j}{F_j} \geq \frac{N}{\sum_j f_j F_j} \frac{1}{\sum_j f_j}$$

with equality when all the F_j are equal.

Thus, sampling with replacement,

$$\frac{N}{\sum_j f_j[P]} \sum_j \frac{f_j}{F_j} [P] \left[\sum_j f_j[P] > 0 \right] \geq \frac{N}{\sum_j F_j[P]} \left[\sum_j F_j[P] > 0 \right]$$

holds if the strategies are such that $\left[\sum_j f_j[P] > 0 \right] = 1$ if $\left[\sum_j F_j[P] > 0 \right] = 1$. Again, equality holds when there is only a single sampled equivalence class.

The criteria (implied by Jensen's inequality) for dominance in cost are not the same as for dominance in the probability of a correct match. If Jensen's inequality implies dominance

in the probability of a correct match, then it also implies dominance in cost (for both sampling with and without replacement). Yet the converse is not true.

Efficiency

In reality an intruder might not stick to a simple strategy. Some strategies might offer zero probability of success given the sample. So an intruder might intend to launch an attack by matching sample uniques against the population, then revert to sample 2s if the sample contains no uniques. Alternatively, the intruder might note that a sample contains few uniques, and decide to launch an attack on sample 1s and 2s to reduce the cost of the population search. The intruder has direct access to the sample counts, and can use that information to inform the chosen strategy. Of course, the probabilities of a correct match and the expected costs can be calculated exactly as for simple strategies. But it suggests that the intruder might consider the trade-off between the probability of a correct match and the search cost. Thus it is also worth considering the ratio of these two quantities. It seems reasonable to term this the *efficiency* of an attack strategy.

Sampling without replacement

Matching from population to sample,

$$Pr(cm) = \frac{\sum_j [P]}{\sum_j F_j[P]} \left[\sum_j F_j[P] > 0 \right]$$

$$E(T) = \frac{N+1}{1 + \sum_j F_j[P]} \left[\sum_j F_j[P] > 0 \right]$$

$$\frac{Pr(cm)}{E(T)} = \frac{\sum_j [P]}{\sum_j F_j[P]} \frac{1 + \sum_j F_j[P]}{N+1} \left[\sum_j F_j[P] > 0 \right].$$

Matching from sample to population,

$$E(T) = \frac{1}{\sum_j f_j[P]} \sum_j f_j \frac{N+1}{1+F_j} [P] \left[\sum_j f_j[P] > 0 \right]$$

$$\frac{Pr(cm)}{E(T)} = \frac{1}{N+1} \frac{\sum_j \frac{f_j}{F_j} [P]}{\sum_j \frac{f_j}{1+F_j} [P]} \left[\sum_j f_j[P] > 0 \right].$$

It can be shown that,

$$\frac{\sum_j [P]}{\sum_j F_j [P]} \frac{1 + \sum_j F_j [P]}{N+1} \left[\sum_j F_j [P] > 0 \right] \geq \frac{1}{N+1} \frac{\sum_j \frac{f_j}{F_j} [P]}{\sum_j \frac{f_j}{1+F_j} [P]} \left[\sum_j f_j [P] > 0 \right].$$

Simplifying and rearranging,

$$\sum_j \frac{f_j \sum [P]}{1+F_j} [P] \left(1 + \sum_j F_j [P] \right) \left[\sum_j F_j [P] > 0 \right] \geq \sum_j \frac{f_j}{F_j} [P] \sum_j F_j [P] \left[\sum_j f_j [P] > 0 \right]$$

and we now have the product of 2 terms on each side of the inequality, with the first and second terms on the left hand side being greater than the corresponding terms on the right hand side if $\sum_j [P] > 1$. Also, $\left[\sum_j f_j [P] > 0 \right]$ implies $\left[\sum_j F_j [P] > 0 \right]$ so a zero on the left hand side implies a zero on the right hand side.

Sampling with replacement

In this case the results are trivial and can simply be stated.

Matching from population to sample,

$$\frac{Pr(cm)}{E(T)} = \frac{\sum_j [P]}{N} .$$

Matching from sample to population,

$$\frac{Pr(cm)}{E(T)} = \frac{1}{N} .$$

It is clear that with both cost measures matching from a population to a sample dominates the reverse in terms of efficiency. In both cases equality occurs when $\sum_j [P] = 1$. There is a form of “no free lunch” theorem for matching from sample to population; all strategies are equally efficient.

Simulation with real world data

A sample of 10,000 was taken from the 1991 SAR (Office for National Statistics, 1993). This was used as a notional population. A cross-classification was produced over the variables AGE, SEX, DEPCHILD, ETHGROUP, OCCPATN. The counts in this table were treated as population counts. 40 3% samples (without replacement) were taken from this sample and probabilities of a correct match and expected search costs were generated.

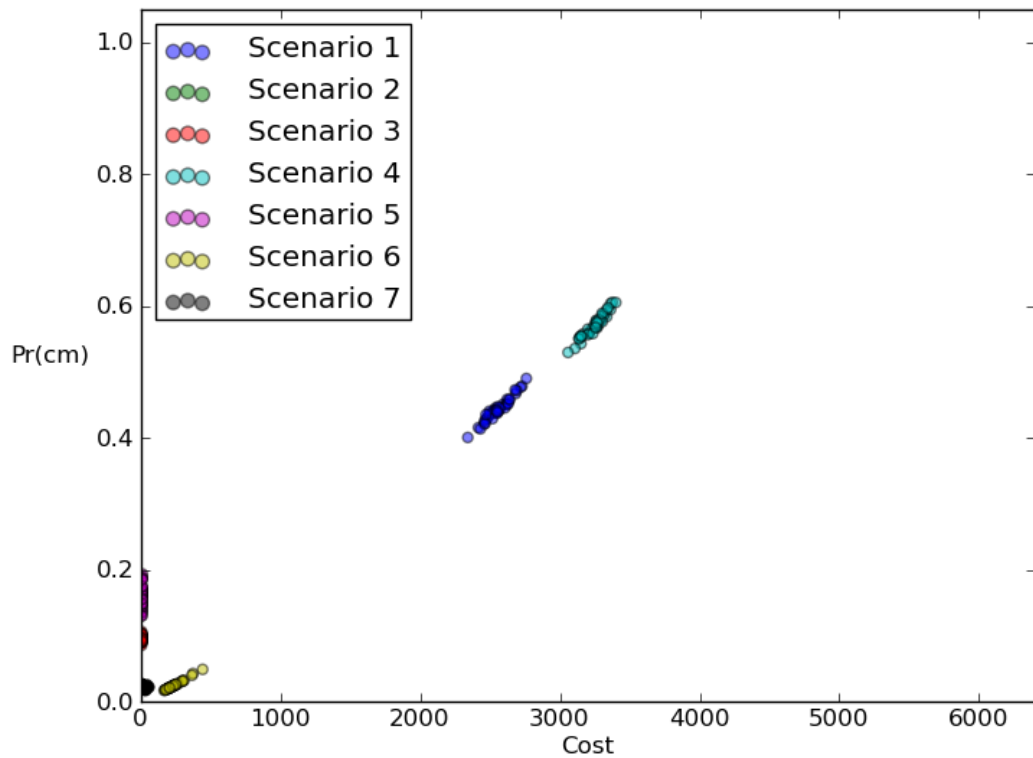


Figure 31. *Sampling without replacement cost measure (with $r=3$ for strategies 6 and 7)*

There were fairly low probabilities of a correct match for most strategies except strategies 1, 4 and 5. Strategy 4 (matching a randomly selected sample unique to the population) demonstrates the highest probability of a correct match (mean=0.571), but at greatest cost (mean=3250.0). Strategy 5 (matching from the population to any sample unique) is highly efficient, but with a fairly low probability of success (mean=0.159) might not be considered as problematical from a disclosure control point of view. This is the $\text{Pr}(\text{CM}|\text{UM})$ measure favoured by Skinner and Elliot. However, the expected cost is very low (mean=7.0), making it a far more plausible form of attack.

It is a little surprising that strategy 4 has such a high probability of a correct match (mean=0.444), with mean cost 2556.9. This is due to the sparseness of the population data, with a total of 10,000 population units spread over 564,000 combinations of categories. With 3,036 population uniques and very low population counts generally, many sample units will have very low corresponding population counts. The variables chosen were

designed to be a set of variables that an intruder might be able to use to identify individuals. A smaller set of variables would have led to larger population counts, lower probabilities of a correct match and lower costs.

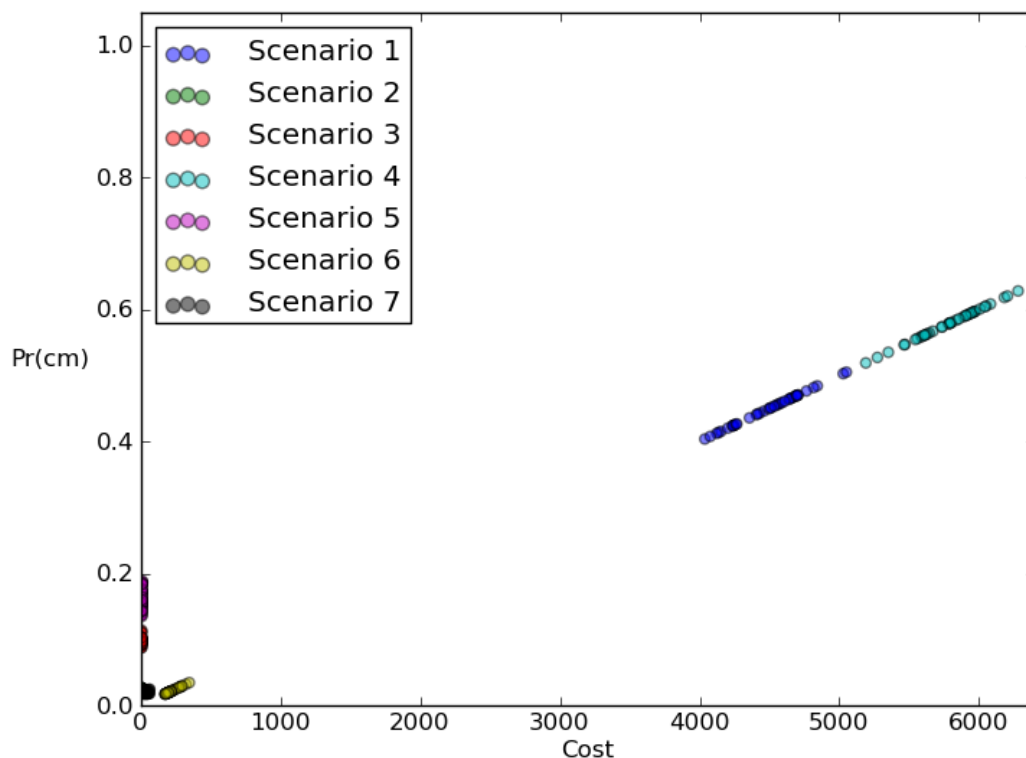


Figure 32. *Sampling with replacement cost measure (with $r=3$ for strategies 6 and 7)*

Using the cost measure based on sampling with replacement the costs tend to be much greater because of the type of search. But the overall picture is the same. All points for strategies 1, 4 and 6 (matching from sample to population) lie on the line $\text{Pr(cm)} = \text{cost} / 10,000$.

Summary

It has been shown that certain strategies for matching sample units with population units dominate others, in the sense that they offer higher probabilities of a correct match for any

population and sample counts. This does not necessarily make them more plausible risk measures because similar dominance relationships hold regarding search costs. Interestingly, there exists a general dominance relationship regarding efficiency, defined as the ratio of the probability of a correct match to the expected cost. Once a set of equivalence classes for consideration has been specified, then attempting to match from the population to the sample is at least as efficient as matching from the sample to the population. In fact for n_c equivalence classes it is n_c times as efficient. Of course, this implies a conditional “no free lunch theorem” for matching from the population to the sample given a specific number of equivalence classes. However, probabilities of success and costs could still vary widely. A strategy that includes a single sampled equivalence class that has a population count of 2 will have efficiency $1/N$ and $\Pr(\text{cm}) = 0.5$. A strategy that includes two sampled equivalence classes with population counts of 1 will have the same cost (due to the same number of matching population units) but be twice as efficient with $\Pr(\text{cm}) = 1$.

The $\Pr(\text{CM}|\text{UM})$ measure favoured by Skinner and Elliot (2002) was shown to be highly efficient. This was because it matches from population to sample and the data were sparse, containing many sample uniques. However, it is dominated (in terms of the probability of a correct match) by $\Pr(\text{CM}|\text{SU})$. This highlights the importance of considering how an intruder might search for a match.

Skinner and Elliot (2002) argue that $\Pr(\text{PU}|\text{SU})$ is too optimistic as it does not take into account the risk from non-unique population counts, and show that $\Pr(\text{CM}|\text{SU}) \geq \Pr(\text{PU}|\text{SU})$. They also propose that $\Pr(\text{CM}|\text{UM})$ is a more plausible strategy than $\Pr(\text{CM}|\text{SU})$ because “the intruder makes fuller use of the microdata information”. However, $\Pr(\text{CM}|\text{UM})$ does not dominate $\Pr(\text{PU}|\text{SU})$. (In simulation results that will be presented in Chapter 7, $\Pr(\text{PU}|\text{SU})$ is consistently greater than $\Pr(\text{CM}|\text{UM})$.) The fact that $\Pr(\text{PU}|\text{SU})$ does not take into account the risk from non-unique population counts does imply that an alternative measure which does might be a better metric. But that metric is $\Pr(\text{CM}|\text{SU})$. However, the search costs associated with $\Pr(\text{CM}|\text{SU})$ suggest that $\Pr(\text{CM}|\text{UM})$ might be a more plausible attack scenario.

All search costs have been calculated on the basis of sampling from the population. If an intruder could reduce the costs of search by getting access to an appropriate population

database, then $\Pr(\text{CM}|\text{SU})$ could become a realistic attack option. More focused searches could also reduce costs.

It has also been shown that where dominance relationships do not exist the relative merits of different attack strategies are highly dependent on the equivalence class structure of the population (and via sampling, the sample). When considering a sparse dataset strategy 1 offers a high probability of a correct match, whereas other simulations with larger population counts generally show that it has a lower probability of a correct match than $\Pr(\text{CM}|\text{UM})$ and at much higher cost.

A pragmatic approach to assessing risk would be to consider a number of scenarios so that the equivalence structure of a proposed data release can be taken into account. Careful consideration of the search strategies that could be employed would be needed. A focused search could make an apparently very costly form of attack plausible.

CHAPTER 6

A Brief Introduction to Dynamic Junction Trees and Model Determination in Decomposable Graphical Models

Introduction

Statistical models that can be represented as graphs are known as graphical models. A graph, $G=(V,E)$ is a vertex set V and an edge set E . Generally, each vertex / node represents a distinct variable or collection of variables, and each edge represents some relationship between its incident nodes. Edges may be directed from a source node to a target node, or be undirected. Some types of graph can have both undirected and directed edges, although all the types of graphical model discussed here will contain either only undirected or only directed edges. This chapter discusses a family of related graphical model types and outlines a novel approach for model determination for decomposable graphs. These models are useful for Bayesian inference (see for example Cowell et al., 1999), but have also been found useful for calculating Fréchet bounds in statistical disclosure control (Dobra and Fienberg, 2001).

Bayesian networks

A Bayesian network, $G=(V,E)$, is a directed acyclic graph where each node represents a variable, and each node is associated with a probability distribution, $p(v|pa(v))$, where $pa(v)$ is the parental set of v . Nodes with no parents are termed *marginal* nodes, because the distribution associated with such a node is marginal, rather than conditional. A node with no children is termed leaf, or *barren* (because it has no children).

The joint distribution over all variables is given by,

Equation 9.
$$p(V) = \prod_{v \in V} p(v | pa(v)) .$$

This equation can be used to perform inference, by conditioning on the relevant levels of any observed variables and marginalization to any queried variables (or sets of variables for joint posterior beliefs). *Normalization* of the resulting posterior distributions might also be required (introducing a constant multiplier so that the distribution sums to 1). In practice, the conditional independences encoded in the network usually allow these calculations to be performed in a piecemeal fashion, without the calculation of the full joint, $p(V)$.

Strictly speaking a Bayesian network should be a minimal I-map (Pearl, 1988). A graph G is an I-map of a distribution P if all the independences implied by G are satisfied by P . G is a minimal I-map of P if it is an I-map of P and no subgraph of G is also an I-map of P .

Figure 33 shows the well known Asia network (Lauritzen and Spiegelhalter, 1988) with the following variables:

A – recent trip to Asia

S – smoking

T – tuberculosis

L – lung cancer

B – bronchitis

E – tuberculosis or lung cancer (a superfluous variable that simplifies network structure)

X – X-ray result

D – dyspnoea (shortness of breath)

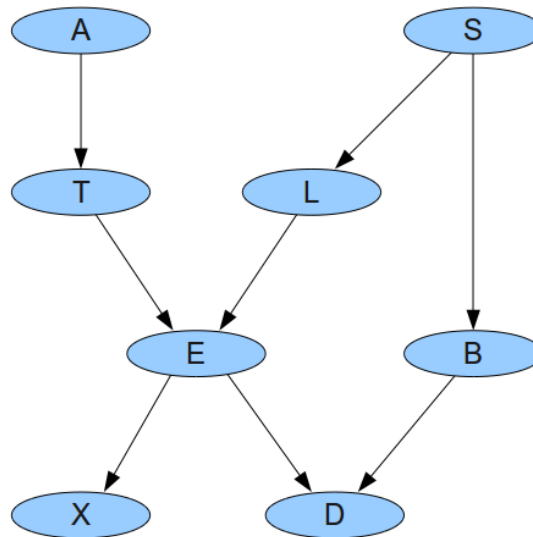


Figure 33. *The Asia network*

This network will serve as an example throughout this chapter.

Markov graphs

A Markov graph is a simple undirected graph that represents probabilistic dependences between variables. Markov graphs have a number of Markov properties; but all boil down to the basic property that if all paths from a set of nodes X to a set of nodes Y include a node from a set of nodes S , then the set X is independent of the set Y given the set S .

Moralization

A Bayesian network can be converted to a Markov graph via a process termed moralization. In moralization each node's parental set is made complete by the addition of undirected edges between all pairs of *unmarried* parents, and the directions on the Bayesian network's edges are removed (the edges are made undirected). Note that each probability distribution associated with the underlying Bayesian network has its variables

as a clique in the Markov graph. Here the word *clique* is used in its usual graph-theoretic sense, meaning a pairwise connected subgraph. A clique that is not a subset of a larger clique will be referred to as a maximal clique when maximality is relevant. (In most of the Bayesian network literature the word clique is used to refer only to maximal pairwise connected subgraphs.)

Figure 34 shows the Markov graph produced from the Asia network. Only nodes D and E have more than a single parent, and adding the edges (E,B) and (T,L) and dropping the directions on the edges completes the conversion to a Markov graph.

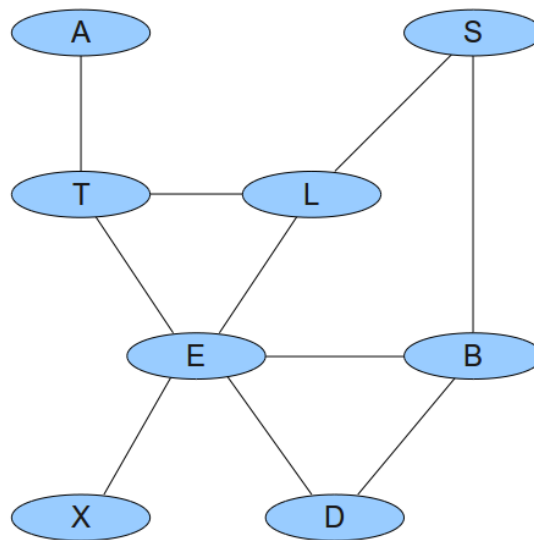


Figure 34. *The Markov graph of the Asia network with moralization edges (E,B) and (T,L)*

Triangulation and decomposable graphs

A triangulation of a Markov graph, $G=(V,E)$, is a set of graph edges T such that $G=(V,E \cup T)$ has no unchorded cycles with length greater than 3 (Rose, 1970). That is, for any cycle of length greater than 3 there exists an edge between non-consecutive cycle nodes. The edges of T are sometimes referred to as *fill-in* edges.

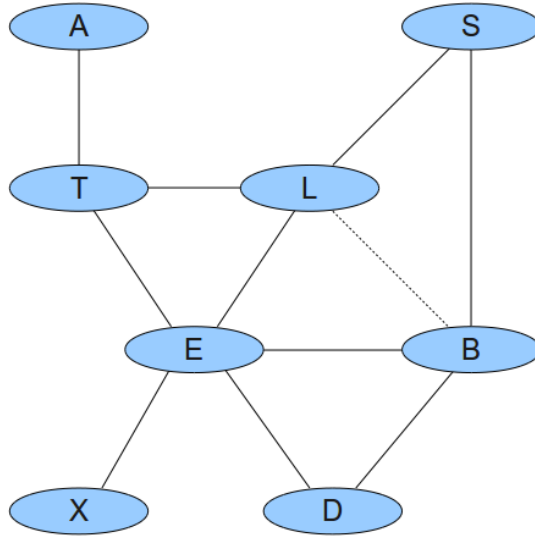


Figure 35. *A decomposition of the Asia network with triangulation edge (B,L)*

Figure 35 shows a triangulation of the Markov graph in Figure 34. The Markov graph only contains a single unchorded cycle of length greater than 3, the 4-cycle $[S,B,E,L,S]$. Adding the fill-in edge (B,L) triangulates the graph, leaving no unchorded cycles of length greater than 3.

It should be noted that performing inference according to Equation 9 involves conditioning on probability tables and marginalising across non-queried variables. Marginalization over a variable v can generally only take place after all probability tables containing v have been multiplied together elementwise. Multiplying all the tables containing a node v produces a generalized distribution with dimensions $\{v\} \cup adj(v)$ where $adj(v)$ is the set of neighbours of v . Marginalization over v leaves a generalized distribution with dimensions $adj(v)$ which might be combined with other distributions on the marginalization of subsequent variables. In terms of the graph, marginalization requires edges to be added such that $adj(v)$ is pairwise connected to reflect the multiplication operation, followed by the removal of v and its incident edges from the graph to reflect the marginalization over v . The resulting graph has node set $V - \{v\}$, and edge set determined by E and the removed edges incident to v and the edges added to make $adj(v)$ pairwise connected. Details can be found in e.g. Rose et al. (1976).

An *elimination ordering* α is an ordered set of the nodes of G . Elimination of the nodes according to any such ordering will eliminate all the nodes in the graph. But the union of the edges added during the elimination will constitute a triangulation of the initial graph G . Choosing to marginalize variables in different orders will tend to produce different triangulations with potentially widely ranging costs in terms of the required numbers of multiplications and summations. This approach to triangulation is sometimes known as the *elimination game*. However, other algorithms for triangulation exist which are expressed solely in terms of the structure of the G . There are many papers on the subject of graph triangulation algorithms, such as Rose et al. (1976), Ohtsuki et al. (1976) and Leimer (1993).

An elimination ordering of the variables in a graph which does not imply the addition of extra edges is known as a *perfect* elimination ordering. A graph is decomposable if, and only if, there exists a perfect elimination ordering of its nodes.

The triangulation in Figure 35 is consistent with an elimination ordering starting with S . In order to eliminate S all the tables containing S would need to be combined, that is $p(S)$, $p(B|S)$ and $p(L|S)$. This would produce a generalized table with variable set $\{B, L, S\}$. After marginalising across S we would be left with a generalized table (representing an induced dependency) with variable set $\{B, L\}$. Thus the edge (B, L) needs to be added so that the resulting graph is still a valid Markov graph for the remaining variables. Alternative elimination orderings, say one starting with D then B , would produce an alternative triangulation with the fill-in edge (E, S) .

The *elimination graph* resulting from the elimination of a set of nodes is independent of the order of elimination (Ohtsuki et al., 1976).

A triangulation, T , of a graph, G , is *minimal* if, and only if, there exists no triangulation of G that is a strict subset of T .

There do exist algorithms for finding minimal graph triangulations (Rose et al., 1976) (Ohtsuki et al., 1976) (Leimer, 1993), although an arbitrary minimal triangulation is not necessarily optimal in terms of the computational cost of inference.

The two triangulations $\{(B,L)\}$ and $\{(E,S)\}$ are clearly minimal. If all the variables were categorical we could work out the costs of inference (in terms of the numbers of multiplications and additions required to perform inference) to identify an optimal triangulation. Note that playing the elimination game with an ordering starting with E would produce a very poor triangulation, as B, D, E, L, T and X would need to be pairwise connected.

The union of a Markov graph with a valid triangulation can be referred to as a *decomposable* graph. Thus a decomposable graph can be defined as a graph $G=(V,E)$ which has no unchorded cycles with length greater than 3. It is more common to use the term *decomposable* for graphs which are decomposable, but are not the result of triangulating a Markov graph. They might also be referred to by other names such *monotone transitive* (Rose, 1970) or *rigid circuit graphs* (Dirac, 1961).

Junction trees

There is an alternative way of graphically representing a decomposable graph. A tree can be constructed with a node for each maximal clique in the graph. Edges are added between the cliques to construct a tree with the following property.

Running intersection property (Lauritzen and Spiegelhalter, 1988):

If a node is contained in two cliques, C_1 and C_2 , then it is contained in all cliques on the unique path between C_1 and C_2 .

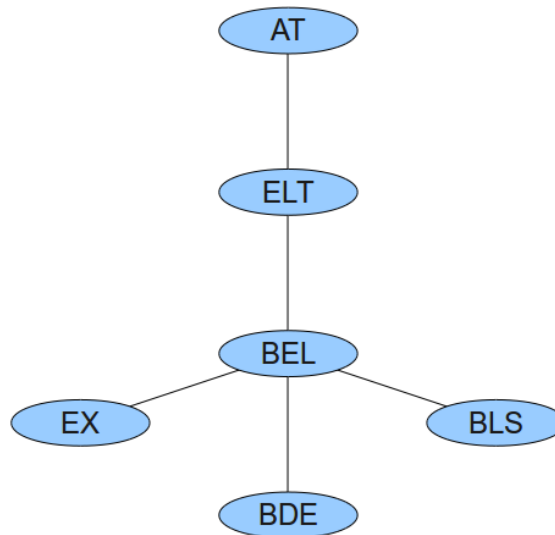


Figure 36. *A junction tree for the Asia network*

Figure 36 shows a junction tree corresponding to the triangulation $\{(B,L)\}$. Note that it possesses the running intersection property; each subtree induced by the cliques containing a given variable is connected. Note also that the clique $\{E,X\}$ could have been connected to $\{B,D,E\}$ or $\{E,L,T\}$. A given decomposable graph does not generally imply a unique junction tree.

Sometimes junction trees are shown with an extra *separator* node on each edge. A separator is the intersection of the nodes in the neighbouring cliques.

There are a number of algorithms for constructing junction trees. One approach is to add the maximal cliques to a graph and then add edges between all pairs of cliques have non-empty intersection. This produces a *junction graph*. A junction tree is a maximal weight spanning tree of the junction graph, where the edge weights are the cardinalities of the adjacent cliques' intersection (separator size).

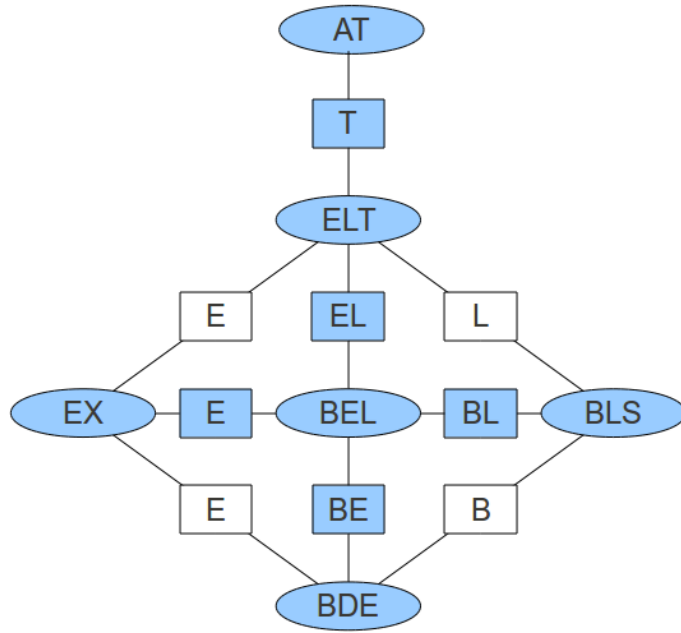


Figure 37. *The junction graph for the Asia network*

Figure 37 shows the junction graph for the Markov graph in Figure 34. Separators are shown and the spanning tree corresponding to Figure 36 is shown with blue separators. The edges with separators $\{L\}$ and $\{B\}$ cannot form part of a maximal weight spanning tree. Any one of the edges with $\{E\}$ separators could have been chosen.

Junction trees are usually used as a data structure for performing inference. Multiplication of distributions and marginalizations to generate posterior beliefs can be thought of in terms of passing messages through the tree to a clique that contains the queried variables. Querying joint distributions which do not appear in a single clique can be achieved by passing multiple messages (Jensen, 1996). Another approach is to change the structure of the tree itself, so that a suitable clique is created. This approach was presented in Smith (2001) and uses the variation on the usual junction tree structure detailed in the following section.

Dynamic Junction Trees

A dynamic junction tree is a rooted tree with labelled cliques (see Smith (2001) for an earlier but similar data structure). There is a node for each node in the underlying Bayesian network, thus the cliques are not necessarily maximal. Each node label corresponds to the elimination of a single variable and labels the root node of the subtree induced by the cliques that contain the label. Figure 38 shows a dynamic junction tree for the Asia network which is consistent with the triangulation in Figure 35. It contains all the cliques contained in Figure 36 but also some additional non-maximal cliques.

A dynamic junction tree can be constructed from an elimination ordering using the following algorithm. Note that $madj(v)$ is the monotone adjacency of v with respect to an elimination ordering. It is the set of nodes adjacent to v when v is eliminated as part of the elimination game.

```
makeDJT(ordering):
    clique_pool =  $\emptyset$ 
    tree = empty tree
    for v in ordering:
        clique =  $\{v\} \cup madj(v)$ 
        add clique to tree
        for other_clique in clique_pool:
            if v is a member of other_clique:
                add directed edge from clique to other_clique
                remove other_clique from clique_pool
        add clique to clique_pool
    return tree
```

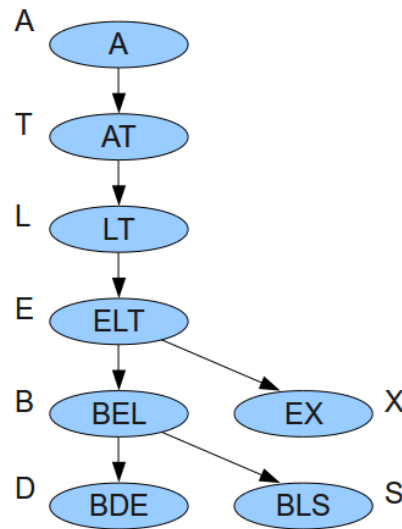


Figure 38. *A dynamic junction tree for the Asia network*

A dynamic junction tree possesses the running intersection property. However, a dynamic junction tree must possess an additional property which distinguishes it from similar trees such as elimination trees (Cowell et al., 1999) and factor trees (Bloemeke and Valtorta, 1998).

Dynamic junction tree property:

An edge between two cliques labelled v and w in a dynamic junction tree implies an edge (v,w) in the underlying decomposable graph.

In other words, the labels of a dynamic junction tree form a rooted spanning tree of the underlying decomposable graph. Unlike for clique trees, a given undirected graph and elimination ordering do imply a unique dynamic junction tree. However, distinct elimination orderings might result in the same dynamic junction tree, so a dynamic junction tree represents an equivalence class of elimination orderings.

The reasoning behind dynamic junction trees is that they can be used as a basis for managing changes to the underlying decomposable graph. For instance, the problem of finding an optimal graph triangulation can be viewed as the problem of finding a perfect elimination ordering that produces an optimal triangulation. The orderings can be searched by repeatedly swapping adjacent pairs of variables within an existing elimination ordering. This can be achieved via rotations of adjacent pairs of cliques in a dynamic junction tree. Details of the rotations have not previously been published and are given in the next section.

Inference using dynamic junction trees uses similar algorithms to those used in junction trees. But this will not be presented in detail as the reason they are being briefly outlined here is to illustrate their use in model determination.

Model determination

There are a number of approaches for model determination. One approach is to iteratively permute the structure of a Bayesian network to search for good models. Single edges can be added, removed or reversed in direction (e.g. Heckerman et al., 1995). More complex moves through the model space have been proposed (Grzegorzcyk and Husmeier, 2008). The problem is to ensure that permutations do not induce cycles, as Bayesian networks are acyclic.

A more common approach is to search for a good decomposable graphical model. In this case the standard moves are edge addition and edge deletion. It has been shown that the space of all decomposable graphical models can be traversed by adding or removing single edges (Frydenberg and Lauritzen, 1989). In this case the problem is ensuring that no edge addition or deletion induces a cycle of length greater than 3. Note that we are actually dealing with a forest of trees, and the starting point for search will often be the full independence model (no edges in the decomposable graph or the corresponding dynamic junction tree).

The basic rules for edge addition / deletion in decomposable graphs are:

An edge (v,w) can be added only if it is not already present, and v and w are either in adjacent maximal cliques or in distinct connected components.

An edge (v,w) can be deleted only if, and only if, it is present in exactly one maximal clique.

Most aspects of these rules can be checked quite efficiently with standard implementations of graphs (e.g. adjacency lists). The presence of an edge can be checked easily, and membership of adjacent maximal cliques implies that the neighbours of v and w have non-empty intersection. Membership of distinct connected components would generally require a search of, say, the component containing v to see if it also contained w . Breadth first search could be relatively expensive for large numbers of variables or if it had to be performed many times during the running of a model determination algorithm. There is no obvious efficient algorithm for checking that an edge is a member of exactly one maximal clique.

Giudici and Green (1999) introduced the idea of using a junction tree representation of the decomposable graph in order to reduce the cost of establishing whether edges could be added or deleted. This allowed membership of exactly one clique to be checked via a local search. The junction tree is updated to reflect edge additions and removals. This is where dynamic junction trees offer an advantage, as they are designed to be permuted.

In a dynamic junction tree:

Two nodes v and w are in distinct connected components if, and only if, the cliques labelled v and w are in components with different roots. Thus the dynamic junction forest can be used as a union-find data structure. Searches are conducted from the tree nodes with labels v and w along their ancestral paths, to identify their root node labels. v and w are in the same graph component if, and only if, they have identical root node labels. Search time is proportional to the depth of the relevant clique in its tree. The search can be shortened for cases where v and w are in the same tree by creating a set of visited cliques. If the path from the clique labelled w to its root contains a clique that is on the path from the clique labelled v to its root, then v and w are in the same component.

If the edge (v,w) is in the underlying decomposable graph, then either the clique labelled v contains w , or the clique labelled w contains v . This follows from the running intersection property and the fact that a clique labels the root node of the subtree of cliques containing the label. Assume that the edge is present and the clique labelled v contains w . None of the ancestors of the clique labelled v can contain $\{v,w\}$ (as they do not contain v). By the running intersection property the cliques containing $\{v,w\}$ form a subtree rooted at v . A depth first search of this subtree will identify whether $\{v,w\}$ is contained in more than one maximal clique. Initially the search attempts to identify a maximal clique containing $\{v,w\}$ (the clique labelled v might not be maximal). If at any point the subtree branches, then the edge is contained in more than one maximal clique. Once a maximal clique is found it is a simple question of checking its children to see if any of them contain $\{v,w\}$. If not, then the maximal clique is the only maximal clique containing $\{v,w\}$. In other words, if $\{v,w\}$ is contained in a single maximal clique, then the rooted subtree of the cliques containing nodes v and w has no branches and only one maximal clique. This is a very efficient check compared to, say, a breadth first search on the underlying decomposable graph / forest.

Consider the decomposable graph in Figure 35 and the corresponding dynamic junction tree in Figure 38. There are only 3 edges that would induce a cycle of length greater than 3 if removed. These are (B,E) , (B,L) and (E,L) . Each of these appears in more than one maximal clique. Take (B,E) for instance. Removal would induce the cycle $[B,D,E,L,B]$. Checking the cliques labelled B and E in the dynamic junction tree would show that the clique labelled B , C_B , is the root clique of the subtree induced by the cliques containing $\{B,E\}$. Checking the cardinalities of its children demonstrates that the C_B is maximal. (If not, then at least one of its children would have cardinality one greater than the cardinality of C_B . More than one such child would imply more than one maximal clique containing $\{B,E\}$.) Having identified it as a maximal clique and having not found any branches, then checking the members of its children would show that C_D contains $\{B,E\}$. Thus $\{B,E\}$ is contained in more than one maximal clique and removal of the edge (B,E) would induce a cycle of length greater than 3 in the underlying decomposable graph.

Tree rotations

On edge addition and deletion the dynamic junction tree must be updated to reflect the change. Dynamic junction trees are updated by tree rotations which are similar to the rotations used for balancing red black trees (see e.g. Sleator et al., 1988). A rotation reflects a local change in an elimination ordering.

Assume we want to locally change an elimination ordering by reversing the ordering of variables v and w . Let C_x denote the clique with label x . Assume also that there exists a dynamic junction tree with corresponding cliques C_v and C_w with a tree edge (C_w, C_v) . Firstly the orientation of the edge (C_w, C_v) is reversed. The edge from the parent of C_w is removed and an edge added from the parent to C_v . Children of C_w (other than C_v which is no longer a child of C_w) remain children of C_w . Children of C_v (other than C_w) remain children of C_v unless they contain w , in which cases the edge from C_v is removed and an edge added from C_w . These rules ensure that the running intersection and dynamic junction tree properties are maintained.

Having changed the structure the members of C_v and C_w must be updated. The rules are quite simple and are based on the result stated earlier regarding elimination graphs. The effects of changing the elimination ordering as above are local. The effect of eliminating all the variables labelling the descendants of C_v and C_w other than C_v and C_w produces a unique elimination graph. After the subsequent elimination of v and w we have a different unique elimination graph.

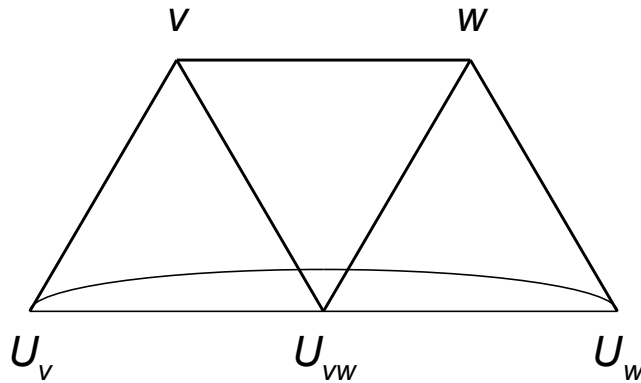


Figure 39. *Schematic showing the subgraph induced by two adjacent cliques*

Figure 39 shows a schematic of the subgraph induced by two adjacent cliques in a dynamic junction tree, C_v and C_w . U_v , U_w and U_{vw} represent the sets of nodes adjacent to only v , only w , and both v and w , respectively. One or more of these sets can be empty. The edge (v,w) is implied by the dynamic junction tree property. The pairwise connectedness of $U_v \cup U_w \cup U_{vw}$ is implied by the elimination of v and w in either order; any missing edges will be added as fill-ins.

Given the ordering v, w we have the addition of fill-ins between w and all members of U_v . The alternative ordering w, v adds fill-ins between v and all members of U_w . So the effect of locally changing the elimination ordering from v, w to w, v is to remove all edges between w and all members of U_v , and to add all edges between v and all members of U_w .

The first stage of a rotation is to identify the sets, U_v , U_w and U_{vw} . Nodes that are not adjacent to v are not members of C_v ,

$$U_w = C_w - C_v$$

Finding U_v and U_{vw} usually requires reference to the set of fill-ins. Thus we need both the dynamic junction tree and the underlying triangulated graph (or triangulation).

$C_v - \{v, w\}$ is the initial set of candidates for U_v . Any members of this set, that are also members of the children of C_w (other than C_v) or the children of clique C_v which also contain w as a member, must be in U_{vw} . Any other $x \in C_v - \{v, w\}$ could be a member of either U_v or U_{vw} . That is, given only the dynamic junction tree, the edge (x, w) could be a graph edge, or a fill-in edge added on the elimination of v . Access to the triangulation allows these edges to be correctly classified.

The new clique members are easily calculated,

$$C_w = U_w \cup U_{vw} \cup \{v, w\}$$

$$C_v = U_v \cup U_w \cup U_{vw} - \{w\}.$$

On rotating a clique C_v and its parent, C_w ,

1. If, and only if, $|C_v| = |C_w| + 1$ then no edges are added to the underlying graph.
2. If C_v has a child C_u such that $|C_u| = |C_v| + 1$ then no edges are removed from the underlying graph.

If both 1 and 2 hold, then the rotation is termed *neutral* with respect to the underlying graph as the triangulation is unchanged.

Neutral rotations can be performed very efficiently. In fact sequences of neutral rotations can be identified to make any clique the root of its dynamic junction tree, and to make any pair of cliques C_v and C_w adjacent if the edge (v,w) exists in the underlying graph [proofs omitted]. In essence, a neutral rotation is a move between distinct spanning trees (with respect to the clique labels) of the same underlying decomposable graph.

To add an edge (v,w) the required neutral rotations can be made so that, say, C_v is a child of a clique containing w . Then w is simply added to C_v . If C_v and C_w are in different components then, say, C_v can be made a root of its tree and then connected as a child to an arbitrary clique containing w , again adding w to C_v .

The existence of the edge (v,w) requires that either C_v contains w or C_w contains v . Assume that C_v contains w . C_v is made maximal by iteratively rotating it with children with cardinality one greater than that of C_v . If there are no such cliques then C_v is maximal. If there is more than one such clique, then the edge cannot be removed without inducing a cycle of length greater than 4 in the underlying decomposable graph. Once C_v is maximal, then w is not contained in any of C_v 's children. If it were, then $\{v,w\}$ would be a subset of more than one maximal clique and the removal of (v,w) would induce a cycle of length greater than 4. Thus C_v is leaf in the subtree induced by the cliques containing w , and w can be removed from C_v without breaking the running intersection property. If C_v only contains v , then it must be disconnected from its parent, becoming the root node of a new dynamic

junction tree. Otherwise, if C_v is a child of C_w it must be disconnected and connected to an ancestor of C_w to restore the dynamic junction tree property.

An alternative way of expressing when edges can be legitimately added or removed is in terms of whether sequences of neutral rotations can be found according to the above scheme for edge addition / deletion. In fact, neutral rotations are so efficient that the software implementation actually identifies and performs neutral rotations until it becomes clear that a necessary sequence of rotations cannot be performed.

Consider the decomposable graph in Figure 35 and the corresponding dynamic junction tree in Figure 38. Assume we wanted to add the edge (A,E). C_A has no parents and C_E 's parent does not contain A. Thus neutral rotations are required to make C_A a child of a clique containing E or C_E a child of a clique containing A. As C_A is an ancestor of C_E it is easier to make C_E a child of a clique containing A. The cardinality of C_E is one greater than its parent, thus their rotation will be neutral. This makes C_E a child of C_T . C_T contains A, so adding A to C_E completes the edge addition. The sequence of operations is shown in Figure 40.

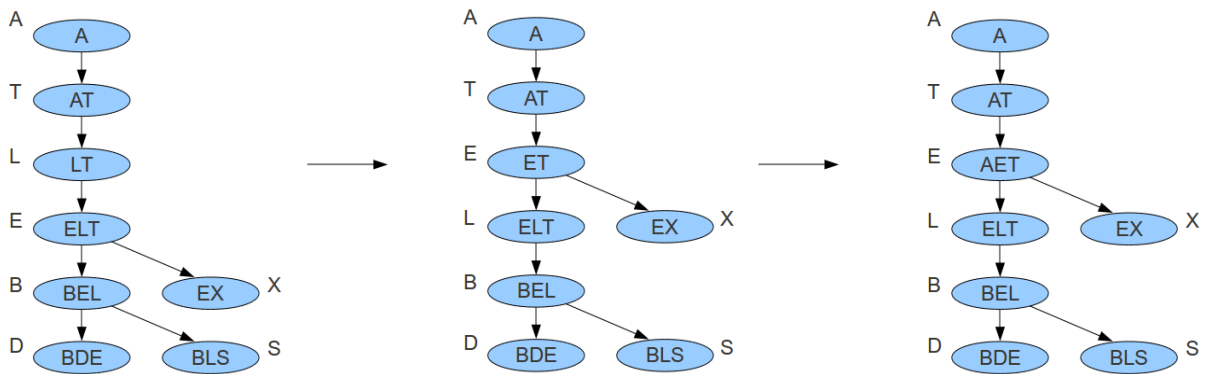


Figure 40. *Edge addition using a dynamic junction tree*

Now assume that we wanted to remove the edge (L,T). C_L contains T, so must be made maximal. This involves the same initial rotation as the previous example. This makes C_L maximal, allowing the edge (L,T) to be removed from the underlying decomposable graph by removing T from C_L . The sequence of operations is shown in Figure 41.

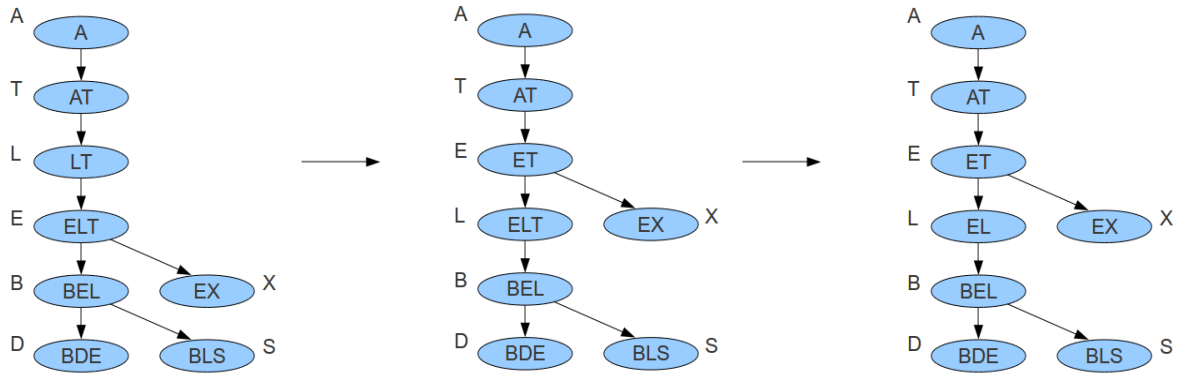


Figure 41. *Edge deletion using a dynamic junction tree*

There are slightly more complex cases where the best sequence of neutral rotations is less clear, such as adding an edge when the cliques with the relevant labels are in different branches. However, the algorithms are relatively straightforward. The main point is that adding and removing edges to traverse the space of decomposable graphs is much more efficient when based only on local structure.

Markov Chain Monte Carlo Model Composition

Bayesian model averaging (Hoeting et al., 1999) attempts to take into account model uncertainty by averaging inferences over a class of models. This requires a probability distribution over the model class. Given this distribution posterior distributions over any quantity of interest Δ can be derived as in Equation 10. Here the model class contains K models and conditioning on the data D used to estimate the models is shown explicitly.

Equation 10.

$$p(\Delta|D) = \sum_{k=1}^K p(\Delta|M_k, D) p(M_k|D)$$

Markov Chain Monte Carlo Model Composition (MCMCMC or MC³) was first described in Madigan and York (1995). A Markov chain is constructed with equilibrium distribution $p(M_k|D)$. This is usually achieved via the Metropolis-Hastings algorithm (Hastings, 1970).

Metropolis-Hastings

A proposal distribution is defined to give the probabilities of attempting a transition to a model N' given a current model N , $p(N'|N)$. A symmetric proposal distribution is often chosen in order to simplify the algorithm. In this case $p(N'|N) = p(N|N')$. The moves between models are made according to the following equation,

Equation 11.
$$p(\text{Accept } N') = \min\left(1, \frac{p(N|N')p(N'|D)}{p(N'|N)p(N|D)}\right).$$

Given a symmetric proposal distribution the ratio in Equation 11 reduces to a simple ratio of posterior model probabilities, otherwise known as a Bayes factor. For decomposable graphical models this is easily achieved by selecting a graph edge at random, and choosing it for deletion if it exists, or addition if it does not exist. Dawid and Lauritzen (1993) showed that the Bayes factor for decomposable graphical models differing by one edge can be expressed as a ratio involving only 4 terms, all of which can be computed locally. Assume that the edge (v,w) is being considered for addition / deletion and C_v is the clique to which w will be added / deleted on acceptance of the move. Then the Bayes factor is given by

Equation 12.
$$\text{Bayes factor} = \frac{p(C_v')p(C_v - \{v\})}{p(C_v)p(C_v' - \{v\})},$$

where C_v and C_v' will only differ by $\{w\}$.

For categorical models the individual terms can be calculated as in Equation 13,

Equation 13.
$$p(C) = \frac{\Gamma(\lambda)}{\Gamma(\lambda + n)} \prod_i \left(\frac{\Gamma(\lambda_i + n_i)}{\Gamma(\lambda_i)} \right),$$

where i indexes the counts n_i in a contingency table of the data over the variables in C . The λ_i are parameters derived from a Hyperdirichlet prior. The Hyperdirichlet prior can be viewed as a contingency table over all the variables in the model containing counts equivalent to prior information. Parameters for a given set of variables can be derived from this table via marginalization, just as the contingency table of data for a set of variables can be obtained by marginalization from a contingency table over all the model variables. Common choices for the Hyperdirichlet prior are to make all the λ_i equal for the single table (over all model variables) corresponding to the full dependence model. Of course, when all the Hyperdirichlet parameters are equal there is no need to maintain a contingency table for the Hyperdirichlet, as the parameters for smaller tables can simply be calculated. Prior specification then reduces to the specification of a single parameter, which may be supplied as the value for the λ_i for the single table over all variables, or their sum. The latter is referred to as the *total prior precision*.

The remaining terms in Equation 13 are given by,

$$\lambda = \sum_i \lambda_i$$

and

$$n = \sum_i n_i$$

and are clearly constants: the total prior precision and the table total respectively.

The scheme described above allows the sampling of the posterior distribution over the model space. It accounts for model uncertainty, but does not incorporate parameter uncertainty. Similar, but more complex, schemes can allow for that (Green, 1995). It is important to note that the sample is not independent as there are usually only a small fraction of possible models that can be moved to from a given model. A common approach to avoid the problem of having an autocorrelated sample is to thin the Markov chain by only including every k th visited state in the sample for some suitable integer k . A suitable number of iterations are performed to allow the algorithm to *burn in*. This is designed to allow the first recorded state to be drawn from the posterior distribution of interest, rather than simply returning the chosen starting state. A pragmatic choice of starting state for the problem under discussion is the full independence model. Finding suitable choices for the burn-in period and the thinning parameter k generally require some experimentation for a given problem.

Using the simple proposal scheme of selecting an edge of a pair of graph nodes at random and attempting to add / remove the edge (depending on whether it already exists) allows a Metropolis-Hastings algorithm to generate a posterior mass function over the space of decomposable graphical models. Inferences can then be averaged across this mass function in accordance with Equation 10.

Simulated annealing

Markov Chain Monte Carlo samples from the posterior distribution of a quantity of interest, in the case under discussion the space of decomposable model structures. It can be time consuming, and there can be problems with mixing. When samples are highly autocorrelated the chain is said to mix poorly. Ideally the sample would be independently and identically distributed, and therefore so would any characteristics of the sample points. Thus poor mixing can potentially be diagnosed through plots of marginal log likelihoods or the number of model edges over time. Such plots can indicate suitable parameters for burn-in and the thinning parameter.

In practice poor mixing is not always easily spotted. Chains can spend some time,

apparently mixing well, in certain regions of the model space whilst other similar regions remain unvisited. In cases of poor mixing or where computational costs are high, an alternative is to search for a single good model via simulated annealing (see e.g. Bertsimas and Tsitsiklis, 1993).

Given an existing MCMC scheme it is easily adapted to simulated annealing. The difference is in how the probability of a transition from one state to neighbouring state is generated. The goal is to end in a globally optimal state, rather than to sample from the posterior distribution.

Let the solution visited at time t be $x(t)$. Then

Equation 14.
$$P[x(t+1)=j|x(t)=i]=q_{ij} \exp\left[-\frac{1}{T(t)} \max\{0, J(j)-J(i)\}\right]$$

where q_{ij} is the probability of choosing j as the next candidate solution given the current solution i and where J is a real-valued cost function over the solution space.

$T(t)$ is the temperature at time t . T is a non-increasing function termed the *cooling schedule*. At high enough temperatures the costs have little influence on the transition probabilities which are close to 1. At low enough temperatures the probability of moving from a low cost solution to a higher cost solution are low. Thus the search will traverse the space quickly during the initial search, gradually moving to areas of lower cost regions, and eventually finding a low cost solution. A common choice is the exponential cooling schedule where $T(t) = T(t-1) \cdot z$, $0 < z < 1$ and z is typically chosen to be slightly less than 1. $T(0)$ and z are specified along with a finishing temperature at which point the algorithm will stop.

The MCMC scheme that has already been outlined can be used for simulate annealing. Neighbouring states are selected in the same way, but transition probabilities are generated according to Equation 14. A pragmatic cost function is the negative marginal log likelihood. Thus $J(j) - J(i)$ in Equation 14 is simply the log of the Bayes factor given in

Equation 12.

Inference using decomposable graphical models

Inference using Bayesian networks has been only very briefly outlined. It relies on the factorization over the joint distribution of all model variables given in Equation 9. Junction trees (or dynamic junction trees) can be used to manage the necessary mathematical operations. These trees are alternative representations of decomposable graphs. Given a decomposable graphical model, one approach would be to try to work back to a set of distributions for a member of the equivalence class of Bayesian networks admitted by the decomposable graphical model. Parameters could be estimated from the data used to fit the model. However, a decomposable graphical model admits a more general form of factorization over the joint distribution; the product over the maximal cliques of their joint distributions divided by the product over the joint distributions of the clique separators. For a dynamic junction tree a similar expression is,

Equation 15.

$$p(V) = \frac{\prod_{v \in V} p(C_v)}{\prod_{v \in V} p(C_v - \{v\})}$$

where $p(\emptyset)$ is defined as 1 (to handle the root cliques of dynamic junction trees).

It should be noted that terms in the numerator product will often cancel with terms in the denominator product. This results in the more usual expression in terms of maximal cliques and separators. Management of the mathematical operations can still be carried out using trees. But rather than having a conditional distribution associated with each clique, there are marginal distributions associated with both cliques and separators. Of course, the above factorization can be used directly, conditioning on known information and marginalising across non-queried variables. Performing marginalizations in a perfect elimination ordering

of the decomposable graph generally avoids much unnecessary computation. Again it must be ensured that all numerator and denominator terms involving a variable are combined (by multiplication or division) before marginalization across the variable. Using a junction tree (or dynamic junction tree) to manage the operations takes care of these issues. Belief propagation algorithms using trees are contained in several papers such as Lauritzen and Spiegelhalter (1988) and Bloemeke and Valtorta (1998).

Summary

A number of different, but related types of graphical model have been described. It has been shown how a Bayesian network can be converted to a Markov graph via moralization and dropping the directions on edges. It has been shown how a Markov graph can be converted into a decomposable graph through graph triangulation. Alternative representations of decomposable graphical models have been described.

A novel scheme for model determination in decomposable graphical models has been presented. The general methodology is standard, but the use of dynamic junction trees is not. It offers advantages over other schemes due to the efficiency of maintaining a structure that allows legal moves to be identified. Edges can be added and removed from a decomposable graph represented as a dynamic junction tree by performing simple tree rotations and set operations on the cliques. All the necessary rotations are of a *neutral* type, meaning no edges are added or removed. The actual edge addition / removal is simply the addition or removal of a single node from a single clique. The necessary rotations are very easy to identify and quick to perform.

There are similarities with an existing scheme proposed by Giudici and Green (1999). But the scheme detailed here is more efficient. For instance, it replaces the breadth first search implied by the existing scheme with simple depth first search of a rooted subtree. There is also the cunning use of dynamic junction trees as union-find data structures to identify nodes in distinct connected components, which requires that the trees are rooted.

CHAPTER 7

A Bayesian Strategy for Matching Population and Sample Units

Introduction

In order to protect against statistical disclosure data are rarely released as unperturbed population data. A common approach is to release sample data, uncertainty regarding the missing data making it difficult for data intruders to make precise inferences about the members of the population. However, there are a number of strategies that a data intruder can employ to attempt to match sample units and population units. These strategies have given rise to a number of risk measures, based on the probability that an intruder can correctly match population and sample units. This chapter considers a Bayesian approach to the matching problem, and whether existing risk measures adequately reflect the actual risk posed by an intruder with the knowledge to launch such an attack.

Although it is usual to remove obvious identifiers such as name and address from data before release, an intruder can still attempt to link sample data and population data through other information. The relevant information is commonly assumed to be contained within a set of variables contained in the sample, and potentially available to the intruder about members of the population. These variables are known as *key* variables.

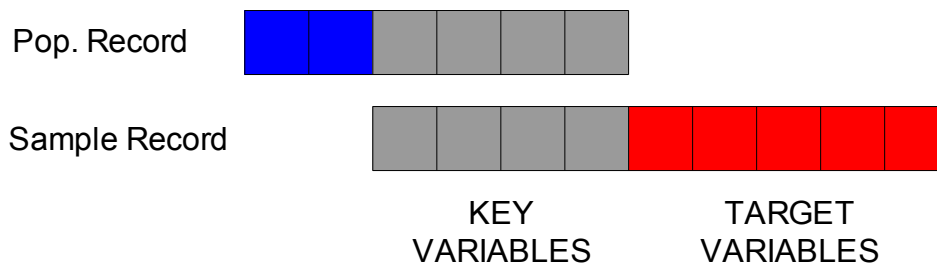


Figure 42. *Matching between sample and population on key variables*

Variables contained in the sample data which are considered sensitive are termed *target* variables. Correctly matching an identified population unit with a sample record allows the values of the target variables to be inferred for the population unit. Thus several strategies / risk measures are based on matching against population units that are unique with respect to a set of key variables.

Notation:

Assume we have a population table containing N individuals and a corresponding sample table containing n individuals.

X	set of key variables
$j = 1 \text{ to } J$	distinct values / equivalence classes / cells for X
$i = 1 \text{ to } N$	population units
X_i	value of X for unit i
N	population size
n	sample size
$[.]$	Iverson brackets
$F_j, j = 1 \text{ to } J$	population frequencies
$f_j, j = 1 \text{ to } J$	sample frequencies

Iverson brackets are used in the manner detailed in Knuth (1992), where

$$[P] = \begin{cases} 1 & \text{if } P \text{ is true} \\ 0 & \text{if } P \text{ is false} \end{cases}$$

and resulting zeros are “very strong”, meaning they annihilate anything by which they are multiplied. So,

$$\frac{x}{0}[P] = 0 \text{ if } P \text{ is false.}$$

Assume throughout this paper that both N and n are greater than 0.

A few measures are:

$\text{Pr}(\text{PU})$ – the proportion of population units which are population unique (e.g. Bethlehem et al., 1990),

$$\text{Pr}(\text{PU}) = \frac{\sum_j [F_j = 1]}{N}.$$

Skinner and Elliot (2002) criticize this measure for being too optimistic as it assigns an element of risk to population units that have no possible match in the sample.

$\text{Pr}(\text{PU}|\text{SU})$ – the proportion of sample uniques which are also population unique (e.g. Fienberg and Makov, 1998),

$$\text{Pr}(\text{PU}|\text{SU}) = \frac{\sum_j [f_j = 1][F_j = 1]}{\sum_j [f_j = 1]} \left[\sum_j [f_j = 1] > 0 \right].$$

This could also be criticized as being too optimistic as it is a lower bound on the probability that a proposed match is correct (which itself is only a lower bound on the probability that a target variable value applies to the relevant population unit).

Skinner and Elliot favour the probability of a correct match given a unique match, $Pr(CM|UM)$,

$$Pr(CM|UM) = \frac{\sum_j [f_j=1]}{\sum_j F_j [f_j=1]} \left[\sum_j [f_j=1] > 0 \right].$$

This measure implies a particular strategy being employed by the intruder. The intruder searches through the population until a match against *any* sample unique is found. Under the assumption that at each point in the search all undiscovered population units are equally likely to be the next one found, the above gives the probability of a correct match.

It can be shown that the strategy of randomly choosing a sample unique and searching the population for a match gives a higher probability of a correct match than the strategy underlying $Pr(CM|UM)$ except when all the sample uniques have been sampled from equal population frequencies. In that case the probability of a correct match is the same. This gives rise to the following measure, which is also discussed in Skinner and Elliot (2002).

$$Pr(CM|SU) = \frac{\sum_j \frac{1}{F_j} [f_j=1]}{\sum_j [f_j=1]} \left[\sum_j [f_j=1] > 0 \right]$$

Although it could be argued that $Pr(CM|SU)$ should be used in preference to $Pr(CM|UM)$ as a risk measure, this does not take into account the difference in search cost. The dominance of $Pr(CM|SU)$ over $Pr(CM|UM)$ is reflected in a similar dominance relationship for cost (see Chapter 5).

Search costs

The assumption of an unfocussed search that underlies the $Pr(CM|UM)$ measure can be used to generate expected costs for a given strategy. Assuming that searching a sample

takes negligible time, then the cost of a strategy can be expressed in terms of the expected number of population units that need to be inspected until a match is found. The search can be thought of as a standard urn problem where population units are randomly drawn until a suitable population unit is found. If the strategy is to identify an individual in the population to match against a sample, then the search is for any population member that satisfies the conditions specified by the strategy. For instance, the strategy might be to only consider population units that can be matched against sample uniques (equivalence classes with sample counts of 1). In fact, this is the precise strategy that gives rise to the $Pr(CM|UM)$ measure. Alternatively the search might involve identifying a sample unit and searching through the population until a match is found.

Even given the unfocussed search assumption it is possible to identify alternative measures of cost. For example, sampling with and without replacement give rise to different measures. The more realistic measure is probably the one derived from sampling with replacement, as this reflects the increasing difficulty of finding population units that have not been previously found as the proportion of undiscovered units decreases. This is the measure that will be used in this paper and is simply,

$$E(T) = \frac{N}{M}$$

where M is the number of candidate population units.

In the $Pr(CM|UM)$ strategy the number of candidates is the sum of the population frequencies with sampled cells equal to 1. In the $Pr(CM|SU)$ strategy there are a number of different costs depending on the sample unique chosen for matching. In this case the expected value of M is the mean of the population frequencies with sampled cells equal to 1. Thus the search cost is greater, but it is less likely that the matched population unit comes from a higher frequency population cell.

Estimation of risks in the absence of population counts

Calculating risks when both population and sample counts are known is trivial. But estimation of risks given the sample counts alone can be difficult. The DIS formula (Skinner and Elliot, 2002) is a good estimator of $Pr(CM|UM)$,

Equation 16.
$$\hat{\theta} = \frac{\pi n_1}{\pi n_1 + 2(1-\pi)n_2},$$

where π is the sampling fraction and n_1 and n_2 are the counts of 1 and 2 respectively in the sample table.

Similarly straightforward estimators for other strategies are still to be found.

A Bayesian matching strategy

A matching strategy involves a trade-off between the probability of success (a correct match) and the cost of search. The $Pr(CM|UM)$ and $Pr(CM|SU)$ strategies both attempt to exploit some information in the sample, namely the sample uniques. They try to exploit the information in slightly different ways. However, neither strategy attempts to exploit the information contained in the joint distribution of the variables. In effect, they treat a multi-dimensional sample table in the same way they would treat a one-way table for a single variable.

Simulations were carried out to investigate whether a Bayesian approach could significantly improve the chances of finding a correct match. Work by others such as Forster and Webb (2007) and Elamir and Skinner (2006) suggests that exploiting the dependence structure in the data leads to improved inferences.

A simple random sample of size 10,000 was taken from the 1991 SAR (Office for National Statistics, 1993) to serve as a notional population. It was assumed that an intruder would have access to information on the variables AGE, SEX, DEPCHILD, ETHGROUP,

OCCPATN. The cross-classification over these variables was sampled repeatedly with sample sizes from 1% to 99% in increments of 1% and each served as a notional sample release to be attacked.

The attack strategy

The first stage of the attack was to use the sample data to generate a posterior mass function over the space of decomposable graphical models. This was achieved using the Markov Chain Monte Carlo Model Composition (MCMCMC) scheme described in Chapter 6. The parameters (marginal probabilities for the probability tables implied by Equation 15) for each sampled model were estimated from the sample data using maximum likelihood estimation. Although there are MCMCMC approaches that can accommodate parameter uncertainty in addition to model uncertainty (e.g. Green, 1995), they were not investigated here.

For model fitting runs a Hyperdirichlet prior was used with total prior precision equal to 1. 10,000 burn in iterations were used, with 1,000,000 subsequent fitting iterations. The thinning parameter k was set to 100. This was felt to be perfectly adequate given that the model space is relatively small with only 5 variables.

The second stage of the attack was to generate the probability of a correct match for each sampled model.

For a given model each cell j had the probability of a correct match calculated as follows:

$$Pr(cm) = \sum_{k=0}^{\infty} \frac{Pr(F_j - f_j = k)}{\binom{k + f_j}{f_j}} [f_j > 0]$$

where,

$$F_j - f_j \sim \text{Poisson}(\lambda_j)$$

and,

$$\lambda_j = \frac{n}{\pi} (1 - \pi) p_j$$

where π is the sampling fraction, n is the sample size and p_j is the marginal probability for the cell indexed j . p_j is estimated from the model. These modelling assumptions follow Skinner and Holmes (1998).

These estimates were then averaged by summing using the posterior model probabilities as weights, producing a single model-averaged probability of a correct match.

$$Pr(cm) = \sum_i Pr(cm|M_i) Pr(M_i)$$

The posterior probabilities of a correct match can be used to inform the final stage of an attack in a number of ways. An obvious strategy would be to take the cell with the highest estimated probability of a correct match and to attempt a match with the population. This would involve a search through the population for a small number of matching units (a single unit if the population count was 1). In order to reduce the search cost, and make the attack scenario a little more realistic, it was decided to take the 10 cells with the highest correct match probabilities and match with the population.

The effectiveness of the strategy was assessed by calculating the true probability of a correct match by reference to the population counts. This probability was taken as an estimate of the probability of a correct match under the strategy (only one simulation was performed for each sampling fraction). Thus a probability of a correct match of 1 actually implies that the 10 highest ranked cells were all population unique.

Results

For all sample sizes the Bayesian approach produced a high estimated probability of a

correct match. In fact, only for the 1% sample did the Bayesian approach not generate an estimated probability of a correct match of 1.

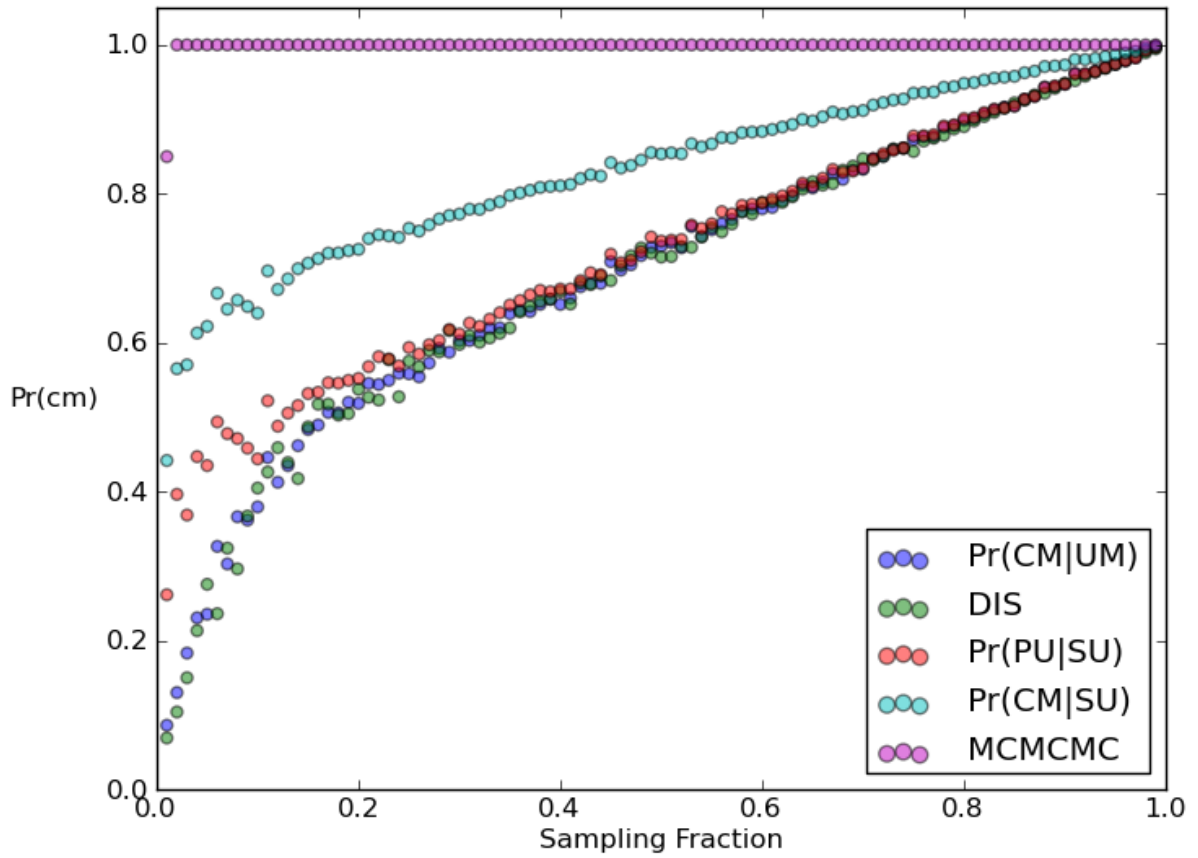


Figure 43. *The probability of a correct match for various strategies and metrics against sampling fraction*

It is clear that all strategies tend to have a higher probability of a correct match for larger sampling fractions. The results in Figure 43 clearly show the dominance of $Pr(CM|SU)$ over $Pr(CM|UM)$. They also demonstrate that the DIS measure is a good estimator for $Pr(CM|UM)$ over the whole range of sample sizes. The remarkable thing is how well the Bayesian approach fares compared to the other strategies. Ranking cells using the information contained in the joint distribution can significantly improve upon strategies that ignore this information.

The search cost for $Pr(CM|UM)$ can be shown to be,

$$E(T) = \frac{N}{\sum_j F_j [f_j = 1]} \left[\sum_j [f_j = 1] > 0 \right]$$

and the cost for $Pr(CM|SU)$,

$$E(T) = \frac{N}{\sum_j [f_j = 1]} \sum_j \frac{1}{F_j} [f_j = 1] \left[\sum_j [f_j = 1] > 0 \right]$$

(see Chapter 5).

In both cases the relevant cells are those that are sample unique. The differences in the probabilities of a correct match and costs are due to matching from the population to the sample and vice versa. Both these options are also available for the Bayesian strategy.

The costs for matching from population to sample, and from sample to population are, respectively,

$$E(T) = \frac{N}{\sum_j F_j [P]} \left[\sum_j F_j [P] > 0 \right]$$

$$E(T) = \frac{N}{\sum_j f_j [P]} \sum_j \frac{f_j}{F_j} [P] \left[\sum_j f_j [P] > 0 \right]$$

where $[P] = 1$ if the cell indexed j is chosen for matching, and 0 otherwise.

There are a number of possibilities for P under the Bayesian approach, as it generates a posterior probability for a correct match for each cell. It could exploit the estimated correct

match probabilities for the cells, or simply choose a given number of the highest ranked cells. The quite arbitrary strategy of taking the 10 highest ranked cells has shown that it can generate correct matches with far higher probabilities than $Pr(CM|SU)$ or $Pr(CM|UM)$ and at all sample sizes. An obvious question is whether it can offer better probabilities of a correct match at similar cost. We have the following relationships between $Pr(cm)$ and $E(T)$.

Matching from the population to the sample,

$$\frac{Pr(cm)}{E(T)} = \frac{\sum_j [P]}{N},$$

and matching from the sample to the population,

$$\frac{Pr(cm)}{E(T)} = \frac{1}{N}$$

(see Chapter 5).

So for $Pr(CM|SU)$ an equal cost implies an equal probability of a correct match. The Bayesian approach can clearly offer higher probabilities of a correct match, but only at greater cost. For $Pr(CM|UM)$ the Bayesian approach will offer higher probabilities of a correct match at equal cost if, and only if, it can identify a larger number of cells that produce equal search costs.

For matching from the sample to the population there is a form of “no free lunch theorem”. All such strategies are equally efficient (if we measure efficiency as the ratio of $Pr(cm)$ to $E(T)$).

For matching from the population to the sample, maximising $Pr(cm)$ for a given cost C involves finding a strategy that maximizes the number cells such that the sum of the corresponding cell population frequencies is equal to the constant, N/C . Thus an obvious strategy is to rank cells according to the expected cell population frequency, rather than the

probability of a correct match,

$$E(F_j|f_j) = f_j + \lambda_j.$$

Summing these in increasing order of value until the sum approximates N/C , where C is the desired cost, is a pragmatic way to attempt to maximize $Pr(cm)$ for a given (approximated) cost.

In order to investigate the possibilities for cost matching the analysis was repeated. $Pr(CM|UM)$ and $Pr(CM|SU)$ series were generated as before, along with series for two MC^3 strategies. Both MC^3 strategies ranked cells according to the expected cell population frequency as detailed above. MCMCMC1 attempted to match the cost of $Pr(CM|UM)$ by referring to the actual population counts. MCMCMC2 attempted to match the cost of $Pr(CM|UM)$ by referring to the expected population counts used to rank the cells. Thus MCMCMC1 was used to investigate whether it was possible to produce higher correct match probabilities, at the same cost, using a Bayesian strategy rather than by simply selecting all sample uniques for matching. MCMCMC2 was used to investigate how effectively costs could be matched using the pragmatic cost matching strategy outlined above.

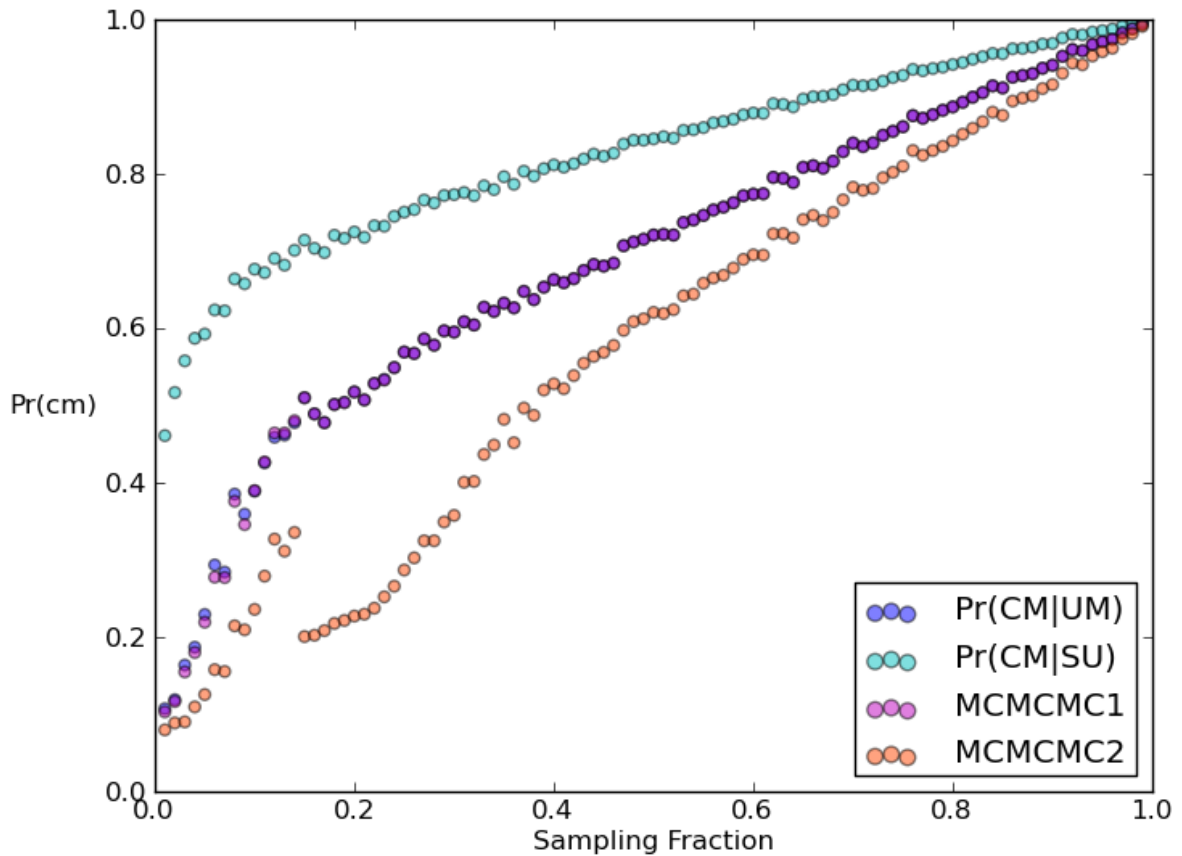


Figure 44. *The probability of a correct match for 2 standard strategies and 2 Bayesian strategies against sampling fraction*

The MCMCMC1 series in Figure 44 shows that it is difficult to produce higher probabilities of a correct match than $Pr(CM|UM)$ at equal cost. The probabilities are identical in most cases. To improve on $Pr(CM|UM)$ at equal cost it is necessary to identify non-unique sample cells that have lower population counts than some of the sample unique cells. The Bayesian approach does identify some candidate cells, but with mixed results, and only for sampling fractions below around 0.15. For higher sampling fractions it selects the sample unique cells exactly leading to the same inferences as $Pr(CM|UM)$.

The MCMCMC2 series is for the Bayesian approach that attempts to match a given cost using the estimated population counts. There is an apparent threshold at a sampling fraction of around 15%. Below this threshold $Pr(cm)$ increases rapidly with sampling

fraction. Above this threshold $Pr(cm)$ increases more slowly. $Pr(cm)$ falls quite dramatically at this threshold before steadily increasing. As the MCMCMC2 series uses exactly the same ranking of cells as MCMCMC1 it is clear that the expected population frequencies are being underestimated. This leads to the selection of a larger number of cells for matching, which reduces both the search cost and $Pr(cm)$. Above the threshold MCMCMC1 selects the sample uniques exactly which implies that MCMCMC2 is necessarily using sample non-uniques for matching. Comparison of the MCMCMC2 series with the $Pr(CM|UM)$ series demonstrates that this is also true for sample fractions below the threshold. However, the similarity of the MCMCMC1 and $Pr(CM|UM)$ series suggests that below the threshold there are sample non-uniques with similar population counts to those of sample uniques. This is not surprising for smaller sampling fractions and would explain the threshold effect evident in the MCMCMC2 series.

At a 20% sampling fraction MCMCMC2 considered 1,309 cells when MCMCMC1 only considered the 1,085 sample uniques. At this sampling fraction the largest expected population count for a sample unique was 1.445, and the lowest for any sample non-unique was fractionally above 2. Although there was still a large degree of uncertainty over the individual population counts (which were actually 6 and 2 respectively) the overall effect on $Pr(cm)$ is quite marked.

It should be noted that the notional population data are very sparse and 31.4% of the population records are population unique. Simulations performed with less sparse datasets do not demonstrate a threshold effect over the selected range of sampling fractions.

Summary

The differences between matching from population to sample and from sample to population have been highlighted. For a given set of cells the latter dominates the former in both $Pr(cm)$ and cost. If matching is restricted to cells that are sample unique, then the former strategy gives rise to the $Pr(CM|UM)$ measure favoured by Skinner and Elliot (2002). The latter strategy gives rise to a $Pr(CM|SU)$ measure. It is not proposed that the latter is a better measure than $Pr(CM|UM)$, as the cost underlying the strategy is much

greater. However, it would be a more appropriate measure if the data intruder had some efficient way of searching a population for a match.

A Bayesian approach has been described. This allows the cells to be ranked so that cells with equal sample counts can be distinguished. For example, choosing a small number of the highest ranked cells can produce probabilities of a correct match which far exceed those from strategies which only depend on the sizes of sample counts. In the simulations that have been carried out there has been a large tendency for the highest ranked cells to be sample unique.

For sparse population data and smaller sampling fractions there is evidence of a significant increase in the predicted population counts when one moves from sample 1s to sample 2s. A similar simulation using all the 1,116,181 records in the 1991 SAR (Office for National Statistics, 1993) resulted in the MCMCMC1 approach exactly identifying the sample unique cells for all sampling fractions.

A method of estimating the cost was presented, but it tends to overestimate the number of cells that should be considered. Nevertheless it provides an intruder with a strategy for trading off search costs and the probability of a correct match. This suggests that an appropriate risk measure should consider the search cost. An intruder who is prepared to spend more time searching a population can achieve higher probabilities of a correct match, either by considering fewer cells, or by matching from the sample to the population. In fact, it suggests that the Bayesian approach could be used to provide a measure of risk that can be tuned to detailed attack scenarios.

The Bayesian methodology is identical to that used in Smith (2006).

CHAPTER 8

A Bayesian Alternative to the Special Uniques Detection Algorithm

Introduction

The release of sample data provides an opportunity for matching against a population. Matches that have a high personal probability of a correct match are considered to be an issue for statistical disclosure control. A data intruder might infer sensitive levels of a variable for some identified population unit. Even where the inferences do not relate to sensitive information, a plausible claim of a correct match could reduce public confidence in the safety of a data release. Often sampling is used as a statistical disclosure measure, the non-sampled units creating uncertainty over the correctness of any matches. In this case the data stewardship organization (DSO) can assess risk by generating measures based on the released sample counts and known (to the DSO) population counts. But sometimes the population counts are not known to the DSO. This is the case when the released data result from a sample survey. The Special Uniques Detection Algorithm (SUDA) was developed at the University Of Manchester to try to address the issue of sample uniques and to categorize them according to risk. The approach is based on observations originally detailed in Elliot et al. (1998). This paper describes a Bayesian alternative to SUDA. The measures are evaluated by applying them to samples from a notional population, the 1991 SAR (Office for National Statistics, 1993).

Sample counts of zero offer no possibility for matching. In other cases, under many attack scenarios, the probability of a correct match for any sample unit and key variable set is $1/F$, where F is the population count with respect to the same variable set and matching variable levels (the number of possible matches). As the sample count provides a lower bound for F it is only the sample uniques that can provide correct match probabilities greater than 0.5.

Elliot et al. (1998) discuss 'special uniques' and 'random uniques'. Special uniques have an

unusual combination of categories, and can be recognized spontaneously. The standard example is the 16 year old widow in a small geographic region. Random uniques are, essentially, the remaining uniques. Clearly, there is not a clear dividing line between these two categories and what might be considered spontaneously recognizable by some, might not be by others. Nevertheless, attempting to rank sample uniques in order of their probability of population uniqueness is a worthwhile exercise. It helps to identify sample uniques which are more likely to give rise to a correct match, and thus pose a greater risk of statistical disclosure. With an effective ranking system, spontaneously recognizable uniques would be included amongst those with higher probabilities of population uniqueness. These could be considered to be of particularly high risk, because spontaneous recognition could lead to disclosure without the need for the sort of population search required by other attack scenarios (see Chapter 5). Spontaneous recognition could lead to an attack that would otherwise not take place.

Special Uniques Detection Algorithm

The general idea behind the SUDA approach is that a sample unique record over a set of variables is more likely to be population unique if it is also sample unique over subsets of the variables in the larger record. The simplest way to think of it is in terms of the detail cells in an n dimensional table of counts, where the n dimensions correspond to a set of key variables. Corresponding tables for subsets of the n variables can be produced by summing across the dimensions of the n -way table. Each member of the powerset (set containing all subsets) of the n variables implies a distinct marginal table, with the dimensionless table corresponding to the empty set being the scalar equal to the table total. Thus the tables form a subset lattice. Figure 45 shows a subset lattice for three variables. Each edge in the lattice corresponds to a marginalization over a single variable.

It is also clear that there is a transitive property in terms of sample counts. As marginalization involves summing of non-negative integers each marginal count must be at least as great as any of the counts that were summed to produce it. Thus a count of 1 in the 1-dimensional table over the set of variables $\{B\}$ must have been produced by summing a 1 and a number of 0s. Therefore the corresponding record must also be sample unique in

tables $\{A,B\}$, $\{B,C\}$ and $\{A,B,C\}$. This implies that the set of tables in which a given record is a sample unique can be specified in terms of the minimal tables in which the record is sample unique, where minimality is with respect to the set of table variables. Elliot et al. (2002) refer to these tables / sets of variables as minimal sample uniques (MSUs). The SUDA approach is to identify the MSUs for each sample unique in the n -dimensional detail table, then to produce risk measures based on this information. (It should be noted that the SUDA algorithms presented in Elliot et al. (2002) and Manning et al. (2008) are for microdata, but I explain the principles in terms of aggregate data for simplicity and convenience.)

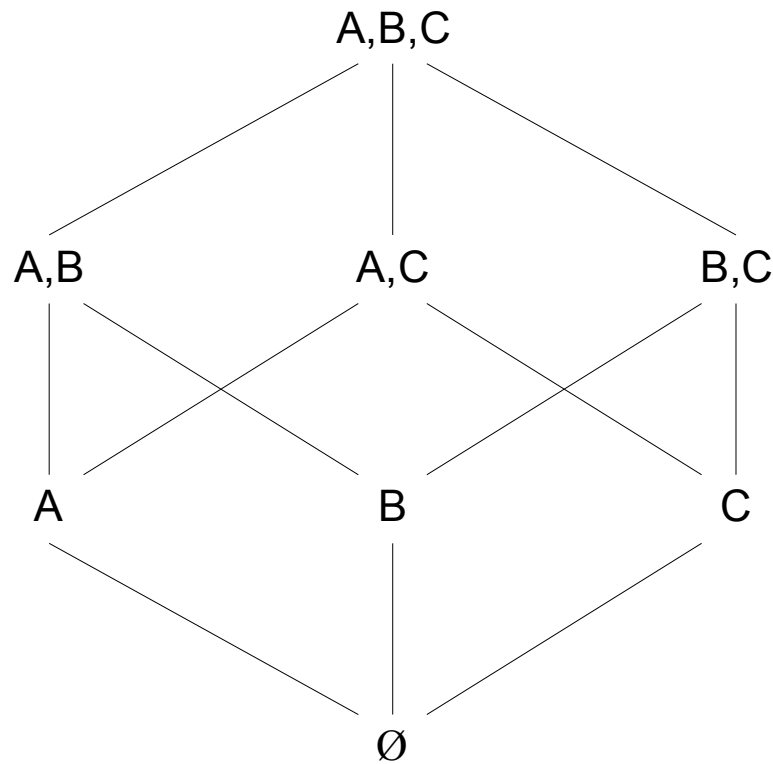


Figure 45. *The subset lattice for the 3 variables A,B and C*

The original algorithm appeared in Elliot et al. (2002). It is based on the well-known 'a priori' algorithm for association rule discovery (Agrawal and Srikant, 1994). Details of the SUDA algorithm are contained in Elliot et al. (2002). However, a brief exposition of a

simple search will provide a flavour of the algorithm without having to go into great detail. Note that this is not intended to be a precise exposition of the original SUDA algorithm, rather it is an exposition of a basic a priori search for MSUs.

Search commences at the bottom of the lattice. The empty set only represents a minimal unique if the sample size is 1. So search can generally be started at the univariate level. An ordering of the variables would be chosen; for convenience let us assume 3 key variables and the order A, B, C. The record set would be searched for records that were unique on {A}. If none were found search would progress to the variable set {A,B}. Assume that a record is found to be unique on {A,B}. This might be, for instance, our 16 year old widow. In order to establish whether this is a minimal unique it would have to be checked whether the record is unique on any of its strict subsets. This leaves only {B} to check. If the record is not unique on {B} then it is minimally unique on {A,B}. If it is unique on {B} then that is a minimal unique (as it is not unique on the empty set). Assume that {A,B} is found to be a minimal unique for the record in question. Therefore it cannot be unique on {B}, and uniqueness on {B,C} cannot be discounted. If it is found not to be unique on {B,C} then the search is over. If it is found to be unique on {B,C} a final check for uniqueness on {C} would be performed to decide which of {B,C} and {C} is minimally unique.

The SUDA algorithm is expensive for large sets of variables. A more efficient algorithm (SUDA2) has been developed more recently (Manning et al., 2008).

Classification

Elliot et al. (2002) contend that the 'riskiness' of a record depends on both the number and size of its minimal uniques. For each record a SUDA score is generated. Each minimal unique is assigned the following weight, equal to the number of its unique supersets,

$$\sum_{i=1}^{ATT-s} \binom{ATT-s}{i}$$

where ATT is the total number of variables and s is the size of the minimal unique.

This is more efficiently expressed (and computed) as,

$$2^{ATT-s} - 1 .$$

Each weight is then divided by the number of attribute sets of size s ,

$$\binom{ATT}{s} .$$

The resulting quantities are then multiplied by a suitably large power of 10 and truncated to integer values, before being summed to produce a score for the record. The scores are used to classify the records into various risk categories, higher scores suggesting higher levels of risk.

There is no theoretical basis provided for the above measure. It is largely based on simulation experiments and calibration.

However, this measure has since been largely superseded by an alternative measure, the intermediate SUDA metric (IS),

$$IS = (ATT - s)! .$$

This equals the number of distinct paths (of length $ATT-s$) from a minimal unique to the lattice node for the complete set of key variables. This gives rise to the proportion of lattice (POL) measure,

$$POL = \frac{(ATT - s)!}{ATT!}$$

which equals the number of distinct minimal paths from the MSU to the full set of key variables, divided by the number of distinct minimal paths from the empty set to the full set of key variables.

Calibration with the DIS score

The DIS formula (Elliot and Skinner, 2002) is a good estimator of $Pr(CM|UM)$, the probability of a correct match given a unique match when matching from population to sample (see Chapter 5),

$$\hat{\theta} = \frac{\pi n_1}{\pi n_1 + 2(1-\pi)n_2},$$

where π is the sampling fraction and n_1 and n_2 are the counts of 1 and 2 respectively in the sample table.

Elliot (2003) describes how the DIS score is used to calibrate SUDA outputs to produce estimated probabilities of a correct match. However, the current approach used by the commercial SUDA software uses the DIS score to calibrate the IS metric to the same ends.

To obtain an approximate matching probability from the IS metric for a sample unique with index j ,

LET D = The DIS matching probability for the file

LET U = the number of sample uniques in the file

LET K = the number of key variables

LET S = IS metric

LET Q = $1 + (8-K)/20$

Then,

$$DIS - IS = \left(1 + \left(\frac{\frac{1}{S_j^Q}}{\sum_i \frac{1}{S_j^Q}} \right) \times \left(\frac{U}{D} - U \right) \right)^{-1}.$$

Again, there is no theoretical justification for the approach. It is a result of experimentation.

So there are alternative methods for ranking sample uniques according to risk. The SUDA score and IS measure (or equivalently POL) will not generally produce identical rankings. The DIS-IS measure would produce the same rankings as IS or POL, but is designed to produce an estimate of the probability of a correct match given a unique match, $Pr(CM|UM)$.

An alternative approach

The SUDA approach is an automatic approach. It generates measures directly from the sample data, and from the sample data and sampling fraction in the case of DIS-IS. It requires no specification of assumptions or prior information. An alternative approach could specify a model form, fit the model from the data, then use the fitted model for inferences. Any such model would have to distinguish between different sample uniques. This suggests that the relationships between variables must be exploited. An increasingly more common method for modelling multi-way contingency tables is based on Markov Chain Monte Carlo methods and decomposable graphical models. This is the approach investigated here.

Graphical models

Dependencies between groups of variables can be represented as graphs. Graphs have a single node for each variable and graph edges denote relationships between variables. One of the simplest forms of graphical model is the *Markov graph* where conditional independence between variables v and w given a set of variables S is implied if the removal of the variables in S (and their incident edges) would result in v and w being members of distinct graph components.

A *decomposable graph* is a graph with no unchorded cycles of length greater than 3. A cycle is chorded if there exists an edge between any two nodes on the cycle that is not an edge on the cycle. Decomposable graphical models have the advantage that they admit a simple decomposition of the joint distribution over the graph nodes (variables). There are relatively simple methods for adding and removing single edges from a decomposable graph whilst maintaining decomposability. Thus the class of decomposable graphs allow a straightforward means of searching the model space for good models. Single edges can be added and removed whilst Bayes factors (ratios of posterior model probabilities) can be generated efficiently. Details of such schemes can be found in Giudici and Green (1999) or Chapter 6. The algorithmic details of the scheme employed in this chapter and appropriate references can be found in Chapter 6.

Markov Chain Monte Carlo Model Composition

Construction of a Markov chain sampler using decomposable graphical models is simple for the reasons stated above. Valid moves are easily identified and Bayes factors can be efficiently calculated. It is the efficient calculation of Bayes factors that leads to the class of decomposable graphical models being searched rather than the class of Markov graphs. Decomposable graphs are clearly a subset of the Markov graphs and are less rich in terms of specifying complex conditional independence relationships. Nevertheless, they are still a rich class of models and still imply independence of variables using the separation property described for Markov graphs above. One difference is that an edge does not imply a marginal dependence, as it could be an edge required for decomposability.

Markov Chain Monte Carlo (MCMC) can be used to generate a posterior mass function over the space of decomposable models. This is sometimes referred to as MCMCMC, MC³ or Markov Chain Monte Carlo Model Composition (Madigan and York, 1995). Inferences can then be averaged over this posterior mass function to provide model-averaged estimates (Hoeting et al., 1999).

MCMCMC offers an approach for modelling the joint mass function over the key variables

used in a SUDA analysis. The sample data are used to fit a posterior distribution over the space of decomposable models, and inferences regarding population counts are then averaged using the posterior distribution. The details of generating probabilities of a correct match under such a scheme can be found in Chapter 7. The scheme used here is identical.

Simulation experiments

Data from the 1991 2% Individual Sample of Anonymised Records (SAR) were treated as a notional population (Office for National Statistics, 1993). The population was taken to be all SAR records from the North West SAR region (the 10th level of REGIONP). This contains 126,251 individuals. The chosen key variables were AGE (age), ECONPRIM (primary economic position), ETHGROUP (ethnic group), LTILL (limiting long-term illness), MSTATUS (marital status), SEX (gender), BATH (bath / shower), CARS (number of cars), TENURE (tenure of household space), RESIDNTS (number of residents in household), DEPCCHILD (number of dependent children in household) and PENSINHH (number of residents of pensionable age in household). Both 1% and 5% samples were taken for the purposes of simulation.

Sample uniques were ranked according to SUDA scores and the IS metric. Probabilities of a correct match given a unique match were estimated using DIS-IS. These were compared with rankings and probabilities based on the above MCMCMC approach. Due to poor mixing of the Markov chain a simulated annealing (see e.g. Bertsimas and Tsitsiklis, 1993) approach was also tested.

MSUs were generated using an alternative algorithm to SUDA and SUDA2 recently developed by the author. A Fortran implementation of the algorithm has broadly similar performance to the C++ implementation of SUDA2 judging by the running times reported in Manning et al. (2008). However, a Python implementation was used so that the running times could be approximately matched with those of the MCMCMC and simulated annealing approaches, which are both implemented in Python.

The Bayesian approach used the Markov Chain Monte Carlo Model Composition (MCMCMC) scheme used in the attack strategy described in Chapter 7, but with the following parameters.

A Hyperdirichlet prior was used with total prior precision equal to 1. 1,000 burn in iterations were performed before 20,000 fitting iterations. The thinning parameter was 20. This is a short burn-in and a much smaller number of fitting iterations than would ideally be chosen. A more appropriate thinning parameter would have been at least 100. Some initial experimentation showed these to be reasonable given the constraint of trying to match the computational cost of the SUDA approach.

Unfortunately the mixing of the chain was extremely poor. Several runs were attempted with alternative priors, but resulted in similarly poor mixing. A hierarchical approach using a Gamma prior for the total prior precision was also tried in an attempt to improve the mixing (Tarantola, 2004). However, the hierarchical approach is significantly more expensive as it negates the speed ups that can be achieved by caching the results of intermediate calculations. It was abandoned on the basis of both computational cost and the fact that it did not appreciably improve mixing. An alternative approach is to start multiple chains from well dispersed starting models (Gelman and Rubin, 1992). This was not attempted as the mixing was so poor that many runs resulted in only a single model being visited during a run. Essentially it would have constituted an expensive way to generate a small number of plausible models, although suitable weights for averaging could have been easily derived from the models' marginal log likelihoods.

It does sometimes happen that with several variables the fitness landscapes are highly multi-modal. However, it is relatively unusual for mixing to be quite so bad. There is always the alternative of foregoing model averaging and using a single model instead. Simulated annealing allows a search to be carried out for a globally optimum model, the maximum a posteriori model. It was found that simulated annealing allowed good models to be found quickly. Multiple runs demonstrated that a variety of models were found, but they were very similar in terms of their edge sets. The Markov chain runs also tended to find models with similar edge sets, although they had greater variability than those resulting from the simulated annealing runs. Simulated annealing runs consistently found higher probability models than those visited in the MCMCMC runs.

The simulated annealing runs were also designed to find the best models possible within a similar time to that taken by SUDA analysis. The initial and finishing temperatures were 10^{14} and 0.01 respectively. The chosen cooling schedule was exponential with parameter 0.99.

Even a poorly mixing Markov chain will converge eventually, and a simulated annealing algorithm with appropriate parameters will find the optimal model with probability arbitrarily close to 1. However, for comparison purposes the various approaches were engineered to have comparable computational costs for the analysis based on the 12 key variables above for the North West region. The MCMCMC running times actually varied much more than the simulated annealing and SUDA running times across a number of trial runs. Once the various algorithms had been approximately calibrated to have similar costs a single run for each was performed for each of the two samples. The results of those runs are reported here.

One percent sample

Predictive performance was compared by generating probabilities of a correct match (given a unique match) and comparing against the correct probabilities derived from the population counts. These were plotted against each other on a \log_{10} scale to ensure a reasonable spread of data points. Only a random sample of 500 data points were plotted as this produced a clearer plot. Perfect predictive performance corresponds to a 45° line. Robust loess smooths (Cleveland, 1979) were also plotted.

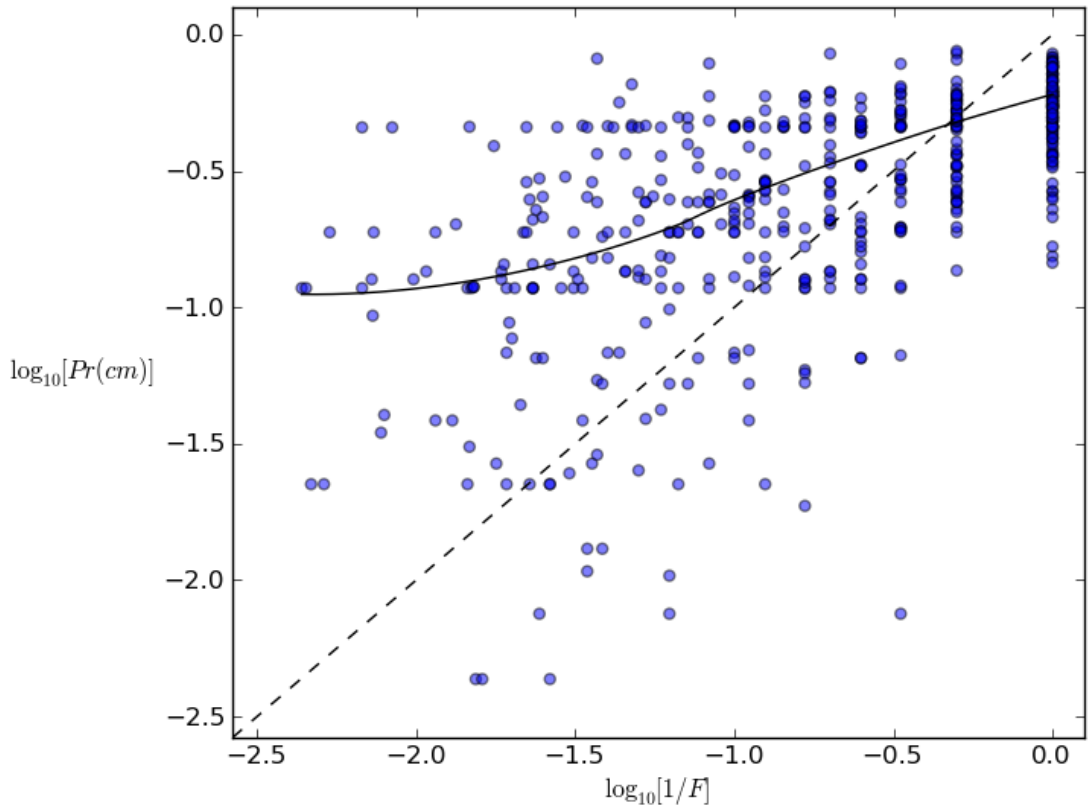


Figure 46. *Predictive performance for DIS-IS with the 1% sample*

The plot in Figure 46 certainly demonstrates a correlation between the true match probabilities and those predicted by DIS-IS. There appears to be a tendency for DIS-IS to overestimate the true probabilities except for smaller population counts, where the matching probabilities are underestimated. It is inevitable that any reasonable estimator will underestimate the true probabilities for population uniques ($\log_{10}[1/F]=0$) because the match probability is 1.

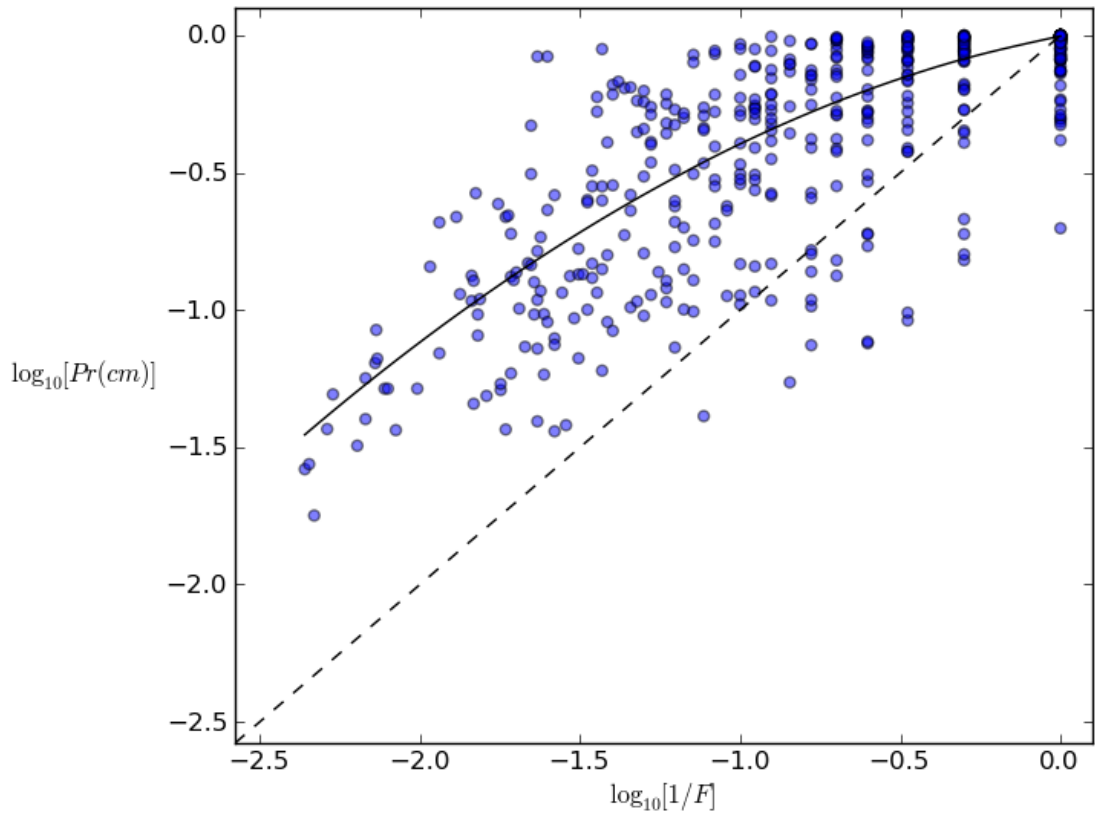


Figure 47. *Predictive performance for MC^3 with the 1% sample*

The MCMCMC run visited 2 models, but the most probable model ($p=0.995$) only differed from the less probable model by a single edge. The best models visited by the MCMCMC and SA (simulated annealing) runs had 11 and 14 edges respectively with 8 common edges. The most probable model overall was the 11 edge MCMCMC model. Figures 47 and 48 both demonstrate a clear tendency to overestimate the matching probabilities for population counts above 1. The MCMCMC and SA estimated match probabilities for population uniques are certainly much more accurate than the DIS-IS estimates, as suggested by the loess smooths. These results do not unequivocally demonstrate that the MCMCMC or SA approaches are better than the DIS-IS measure in distinguishing between sample uniques that are population unique and those that are not. The plots seem to demonstrate that MCMCMC and SA will often produce high probabilities of a correct match for population uniques, but they will also tend to overestimate the probabilities for population non-uniques.

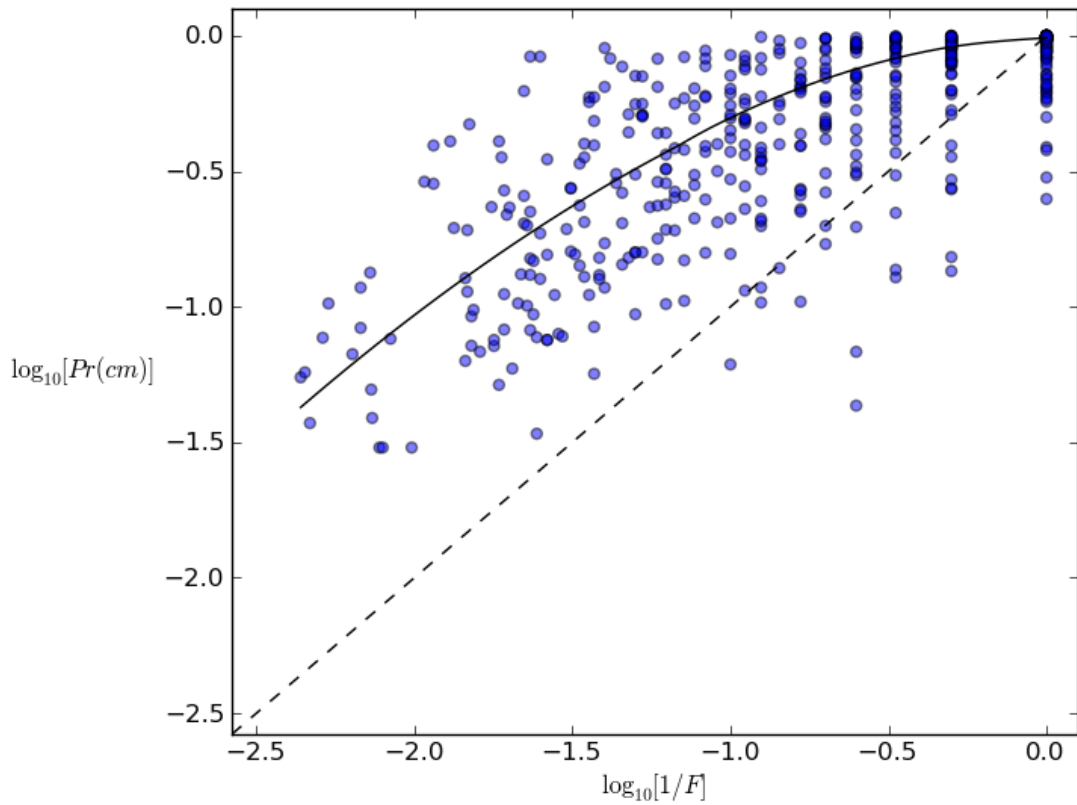


Figure 48. *Predictive performance for simulated annealing with the 1% sample*

A comparison of the relative merits of these measures should concentrate on how well they can rank the sample counts from low to high risk (of population uniqueness). In other words, they should be judged on how well they are able to discriminate population uniques and non-uniques. This depends only on how well they can order the sample uniques in terms of risk, rather than the actual values of the risk measures. Copas (1999) develops a method of comparing scores based on exactly this idea. He assumes that a score is related to the probability of an event through some monotone transformation of the score. But he assesses the effectiveness of a score independently of the accuracy of the monotone transformation. Thus he assesses the potential of a score to discriminate, rather than the accuracy of calibration. This allows DIS-IS to be compared with MCMCMC and SA independently of how well they are calibrated.

Copas (1999) plots the logit transformed ranks against the logit transformed probabilities associated with the ranks. This requires the correct probabilities to be known or estimable from data. For scores that take a finite number of values, such as IS, the true probability of population uniqueness can be estimated from the proportion of sample uniques with a given score that are also population unique. When scores do not have a finite number of possible values, or where the number of values is small in comparison to the available data, Copas (1999) suggests the fitting of a locally weighted logistic regression as detailed below.

For each of the considered measures we have data in the form (y_i, s_i) , $i = 1, \dots, n$, where y_i equals 1 if the population count equals 1 and 0 otherwise, s_i is the corresponding score, n is the number of uniques in the sample, and the i are ordered in increasing size of the score.

A suitable number, m , of plotting points is chosen,

$$t = \log\left(\frac{j}{m+1-j}\right) \quad j = 1, \dots, m.$$

Weights are calculated for each t ,

$$w_{t,i} = \exp\left\{-\frac{1}{2} \frac{(t - t_i)^2}{h_t^2}\right\}$$

where,

$$h_t = \frac{h}{4\psi(t)}$$

and

$$\psi(t) = \Psi(t)\{1 - \Psi(t)\}$$

is the logistic density function, and

$$\Psi(t) = \frac{\exp(t)}{1 + \exp(t)}$$

is the logistic distribution function, and

$$\Psi^{-1}(t) = \log\left(\frac{t}{1-t}\right)$$

is the logit function.

h is a smoothing parameter (the desired standard deviation of the smoothing window at the centre of the plot).

For each t , a_t and b_t are the solutions to the simultaneous non-linear equations

$$\sum w_{t,i} \left\{ y_i - \Psi(a_t + b_t t_i) \right\} t_i^l = 0$$

for $l = 0, 1$.

The logit rank plot is the plot of $a_t + b_t t$ against t .

The code for the plots was written in Python with the weighted regression being carried out using the statistical programming language R.

Properties of the logit rank plot are discussed in Copas (1999), particularly with reference to interpreting the slope and its advantages over the more often used receiver-operating characteristic curve (see e.g. Fawcett, 2006). For the current comparisons the basic interpretation is that better scores have greater slope. Figure 49 shows the logit rank plot for the SUDA score, IS, MCMCMC and SA. As the logit rank plot method is invariant to monotone transformations the IS curve applies equally to DIS-IS. MCMCMC and SA have greater slope than the other metrics. The SUDA score appears to be similar in performance to the IS metric. The curves suggest that if we were to take the n riskiest sample uniques according to the various scores, then we would generally expect to find a greater

proportion of population uniques for MCMCMC and SA, regardless of the value of n .

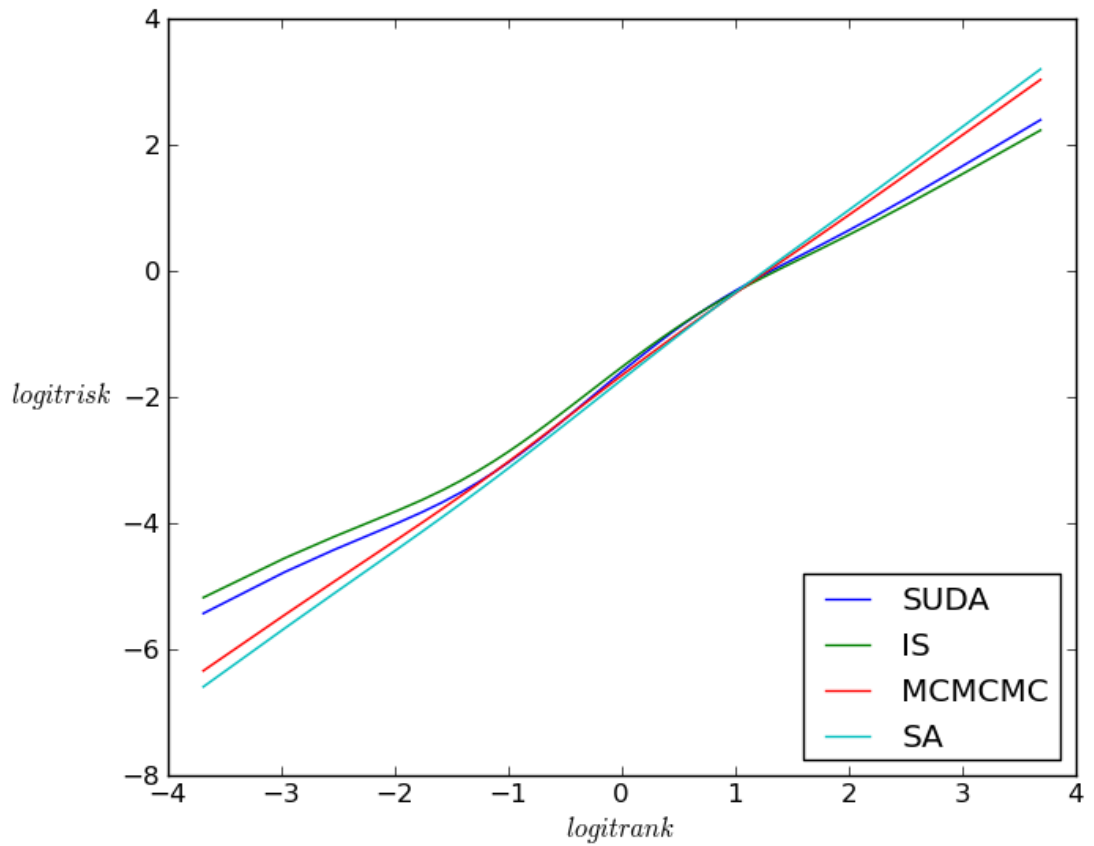


Figure 49. *Logit rank plot for the 1% sample*

The joint distribution of the population and sample counts is shown in Table 2. The table is very sparse, with 3,639,639,605 of the 3,639,680,000 population cells being zero. There are a total of 1019 sample uniques, 280 of which are population unique.

		Population counts									
		0	1	2	3	4	5	6-10	11-20	21+	Total
Sample counts	0	3639639605	26604	5327	2275	1256	822	1647	824	527	3639678887
	1		280	102	73	53	47	141	104	219	1019
	2			1	1	0	1	0	9	47	59
	3				0	0	0	0	0	21	21
	4					0	0	0	0	9	9
	5						0	0	0	4	4
	6-10							0	0	1	1
	11-20								0	0	0
	21+									0	0
	Total	3639639605	26884	5430	2349	1309	870	1788	937	828	3639680000

Table 2. Joint distribution of population and table counts for the 1% sample

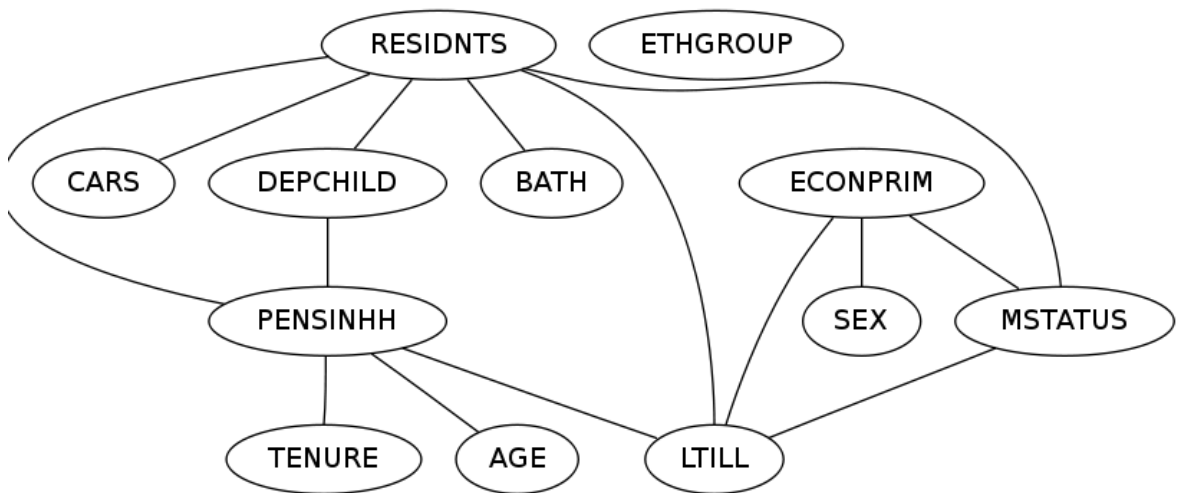


Figure 50. Best model found by simulated annealing for the 1% sample

It is always worth checking that the fitted model makes sense. The model found by the SA run is shown in Figure 50. It implies a number of marginal and conditional independence relationships. For instance, ETHGROUP is marginally independent of all other variables. CARS is independent of all other variables given RESIDNTS. There are several relationships that might be questioned, such as the independence of CARS and ECONPRIM given RESIDNTS. Nevertheless, the above model does lead to improved

inferences over the existing SUDA and IS measures.

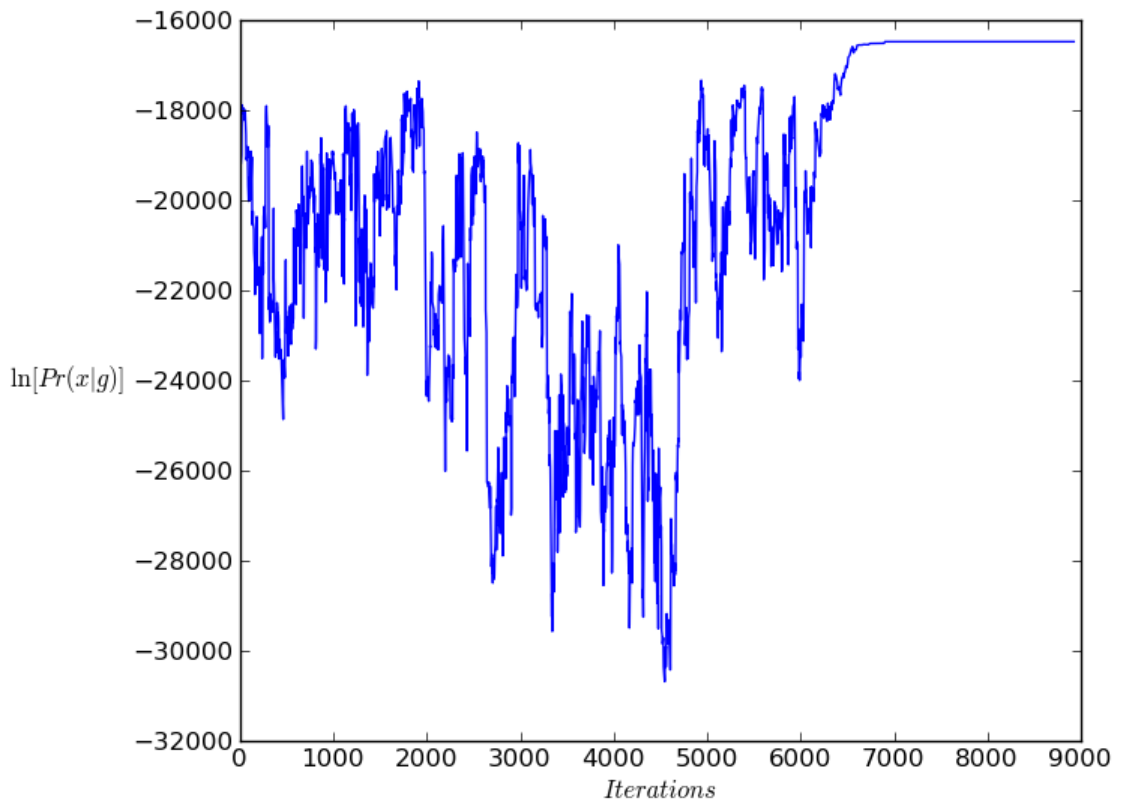


Figure 51. *Convergence of simulated annealing for the 1% sample*

It is also worth checking the convergence of the simulated annealing run. It should be characterized by a more or less random search at higher temperatures, gradually converging to a good solution at lower temperatures. Figure 51 shows the marginal log likelihoods for the visited models during the run. Models with higher marginal log likelihoods have higher posterior probabilities. The convergence appears to be reasonable, particularly given the constraint of finding a schedule that matches the SUDA approach in terms of running time.

There are similar diagnostics that could be produced for the MCMCMC run. However, they are not worth generating when the mixing is so awful. The simulated annealing run clearly visited many distinct models, whilst (after burn-in) the MCMCMC run visited only

two.

Five per cent sample

The results for the 5% sample are similar. There is evidence that all measures overestimate the true probabilities of a correct match for non-unique population counts. This is more so for the MCMCMC and SA approaches.

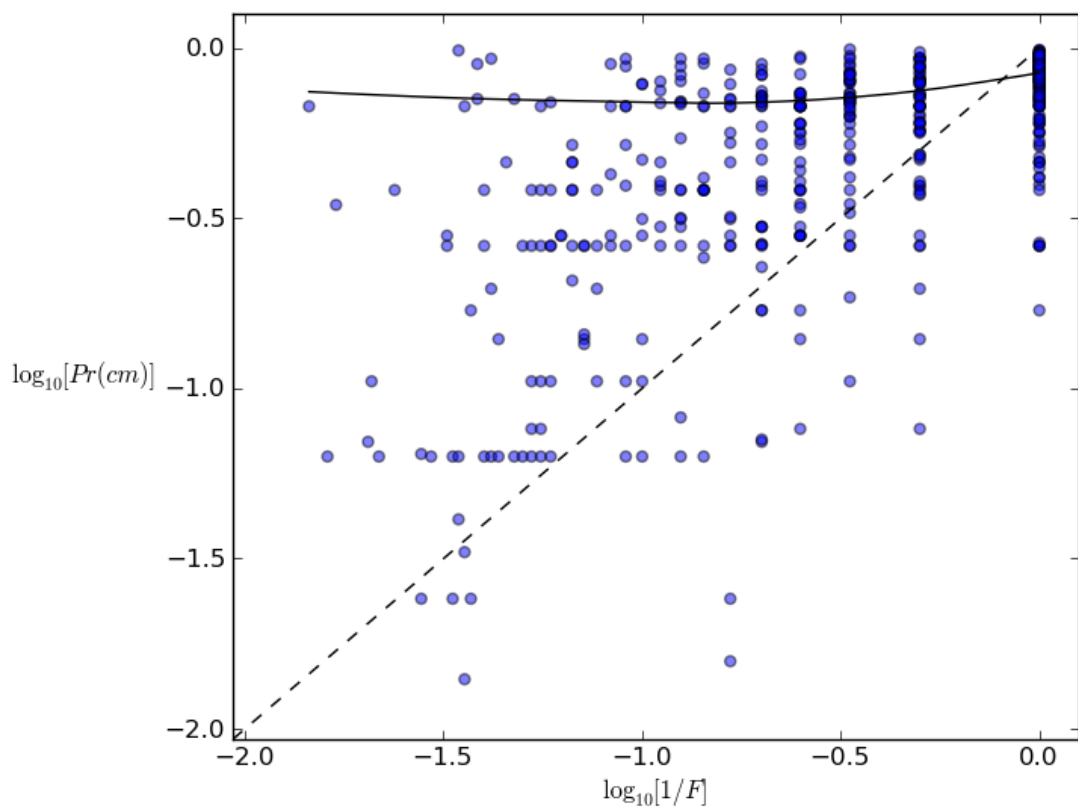


Figure 52. *Predictive performance for DIS-IS with the 5% sample*

Again, it appears that MCMCMC and SA produce better estimated probabilities of uniqueness for population uniques.

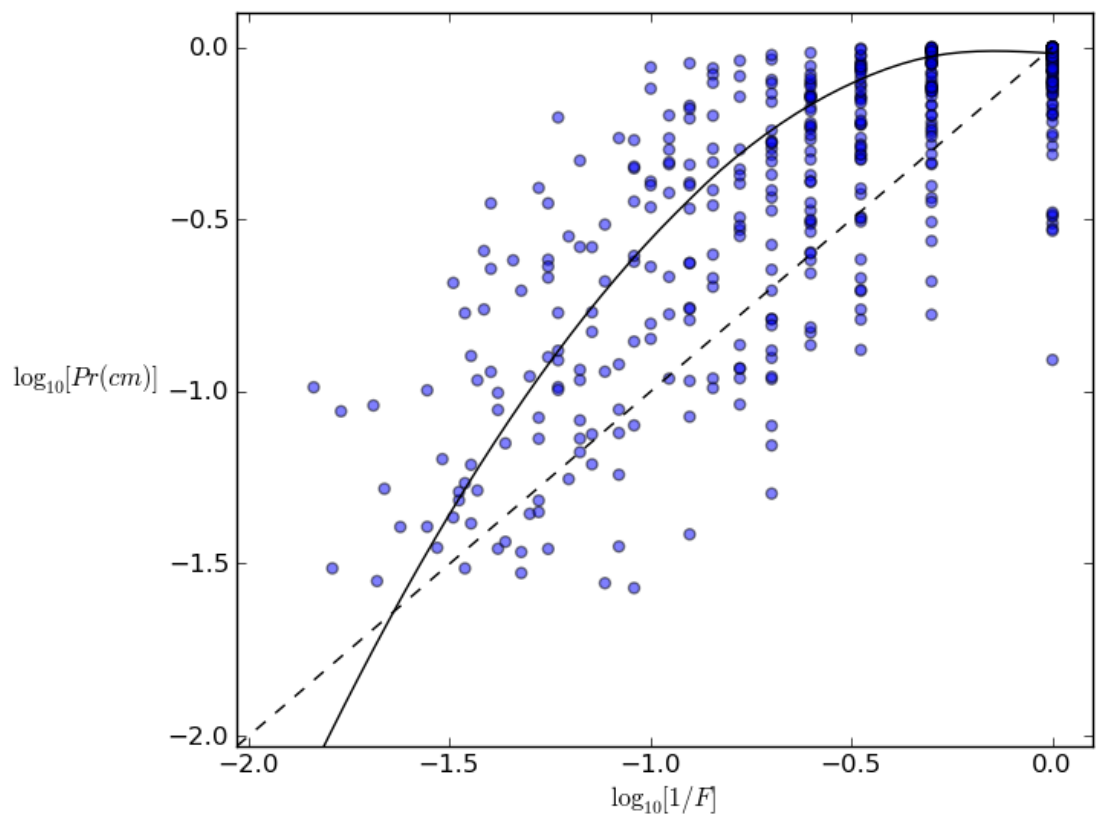


Figure 53. *Predictive performance for MC^3 with the 5% sample*

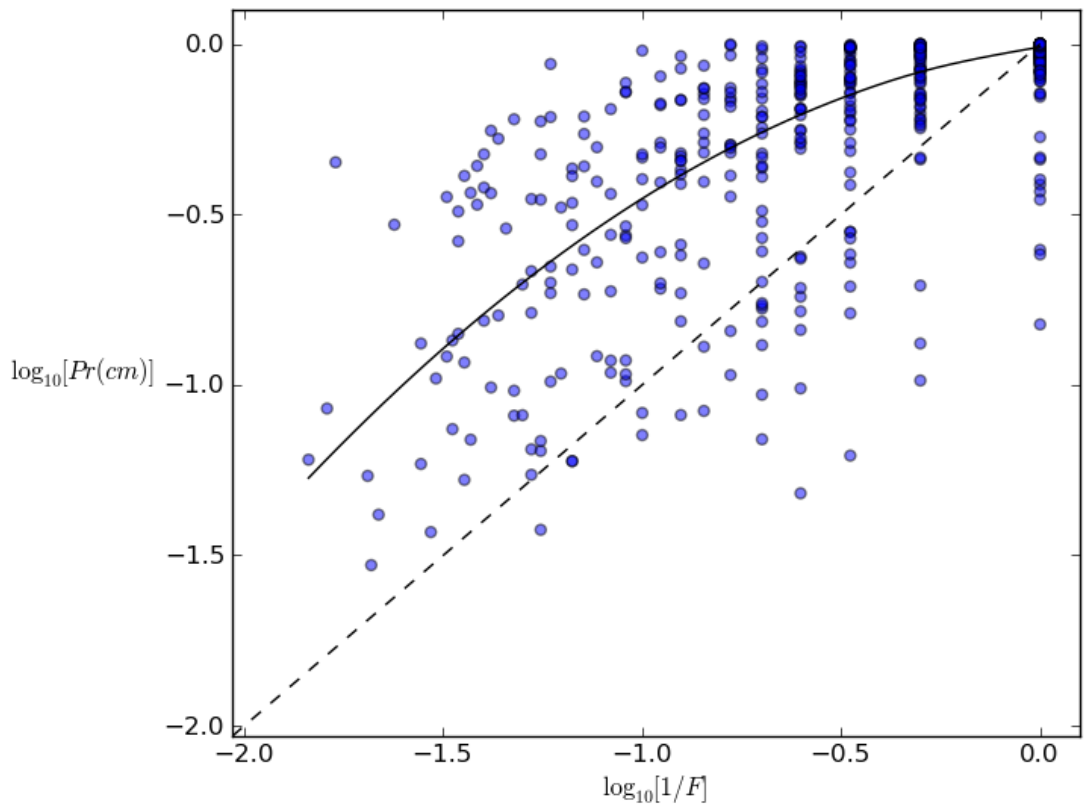


Figure 54. *Predictive performance for simulated annealing with the 5% sample*

The logit rank plot in Figure 55 demonstrates that MCMCMC and SA provide the better scores. In this instance the MCMCMC run visited four models after the burn-in period. The highest probability model ($p=0.920$) was more probable than the best model visited during the SA run. As for the 1% sample, MCMCMC and SA both outperform SUDA and IS.

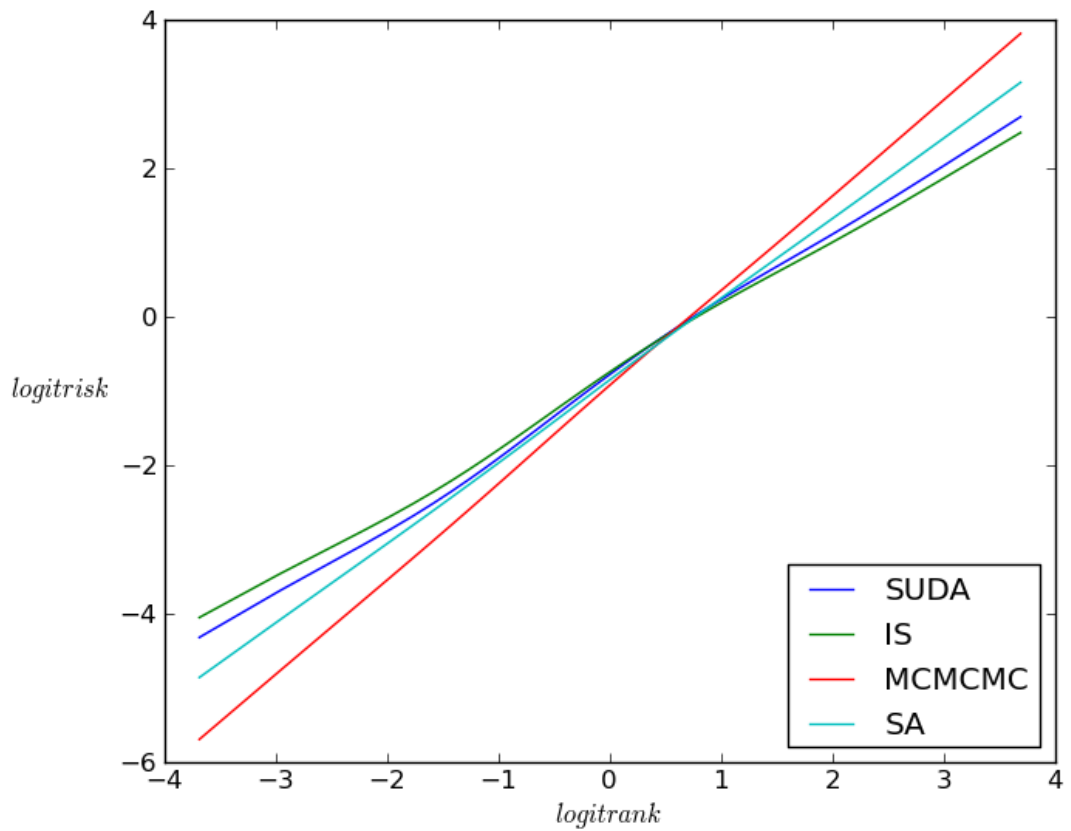


Figure 55. *Logit rank plot for the 5% sample*

		Population counts									
		0	1	2	3	4	5	6-10	11-20	21+	Total
Sample counts	0	3639639605	25567	4938	2016	1042	671	1195	447	125	3639675606
	1		1317	479	319	243	179	499	333	214	3583
	2			13	14	24	19	86	126	173	455
	3				0	0	1	8	27	122	158
	4					0	0	0	4	53	57
	5						0	0	0	40	40
	6-10							0	0	75	75
	11-20								0	25	25
	21+									1	1
	Total	3639639605	26884	5430	2349	1309	870	1788	937	828	3639680000

Table 3. *Joint distribution of population and table counts for the 5% sample*

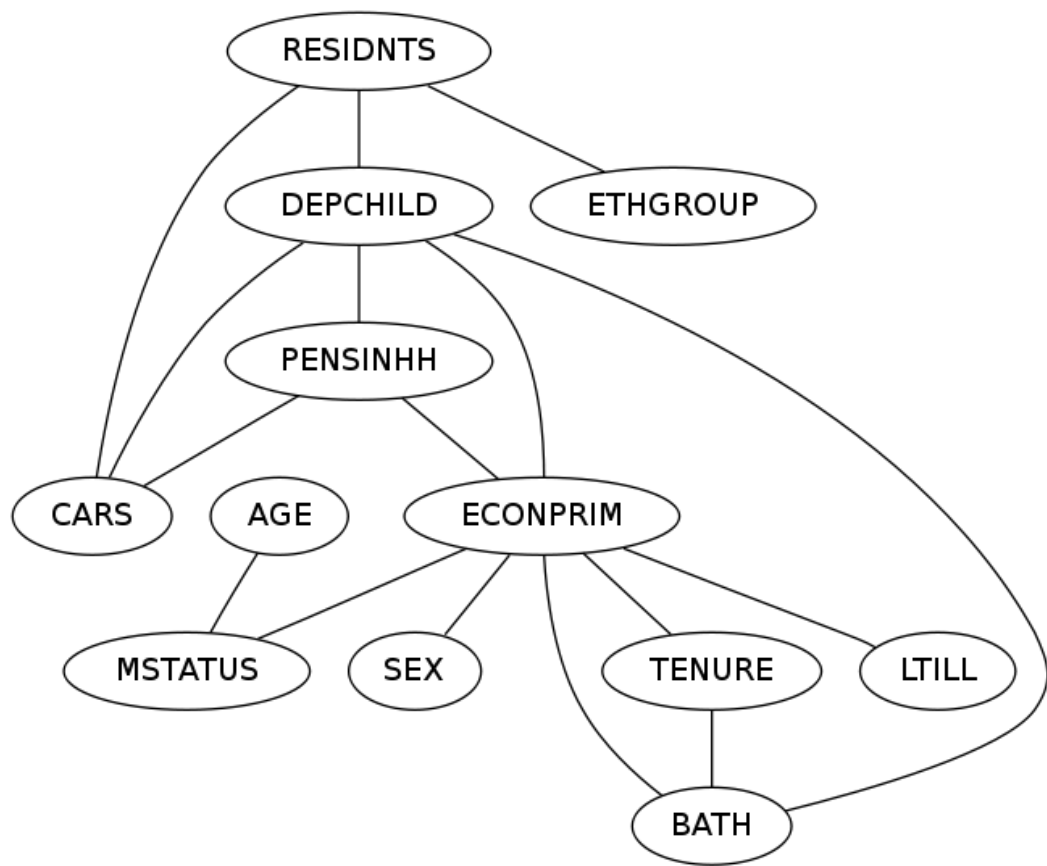


Figure 56. *Best model found by simulated annealing for the 5% sample*

It can be seen that the best model found for the 5% sample differs somewhat from the best model found for the 1% sample. There are no marginal independences and the conditional independences are largely different. Nevertheless it does lead to improved inferences.

Sensitivity to table sparseness

SUDA and DIS-IS have been calibrated against analyses involving around a dozen variables. Essentially, this is their 'sweet spot'. Conditioning on variables, such as the conditioning on REGIONP in the above analyses, will reduce the number of population units without reducing the number of cells (assuming REGIONP would not have been

included in the analysis otherwise). On the other hand, including more variables will increase the number of cells and tend to increase the number of sample uniques and the sparseness of the table. In order to investigate the possible effects the analysis was repeated with TENURE, RESIDNTS, DEPCHILD and PENSINHH removed. It was also repeated with a further 4 variables added; COBIRTH (country of birth), ECONSEC (secondary economic position), FAMTYPE (family type) and SOCLASS (social class). The same 1% and 5% data samples were used across all analyses.

Subsequent discussion will focus on the 16 variable analysis of the 1% sample and the 8 variable analysis of the 5% sample. These represent the most sparse and least sparse of all the analyses. Not all outputs are shown.

1% sample with 16 variables

The DIS-IS estimates of population uniqueness do not appear to be largely biased except for low population counts. This is not dissimilar to the previous results.

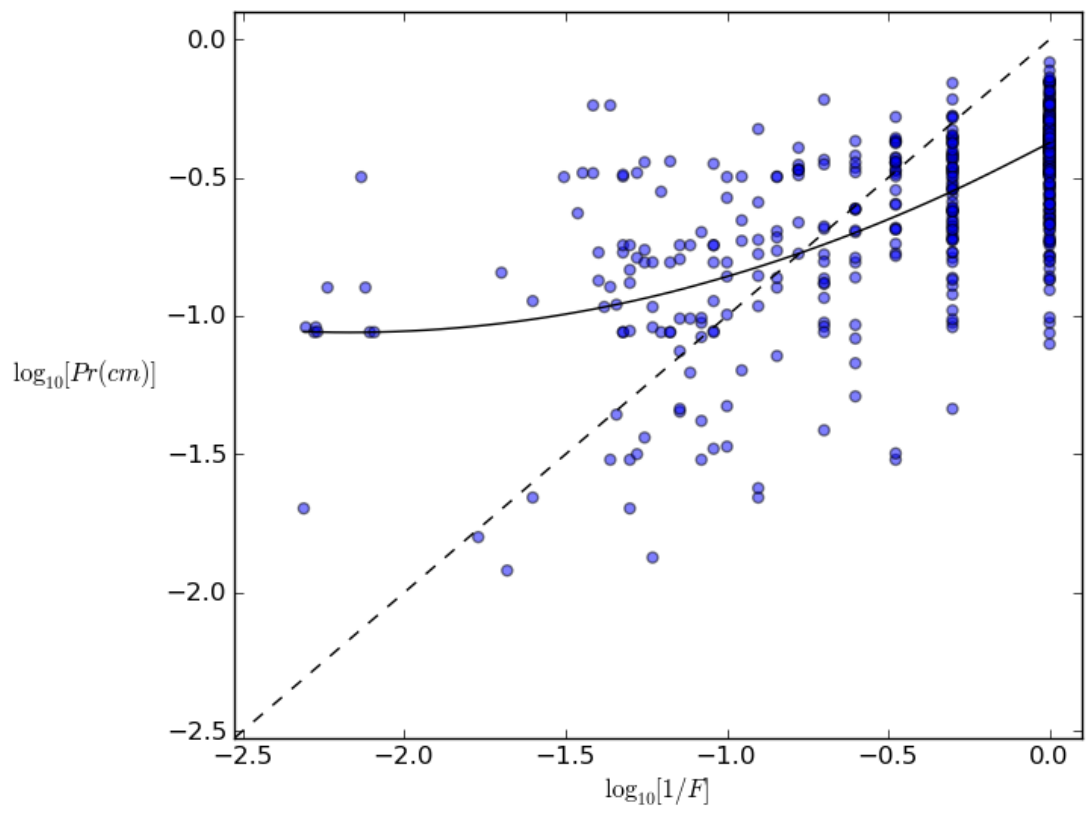


Figure 57. *Predictive performance for DIS-IS with the 1% sample and 16 variables*

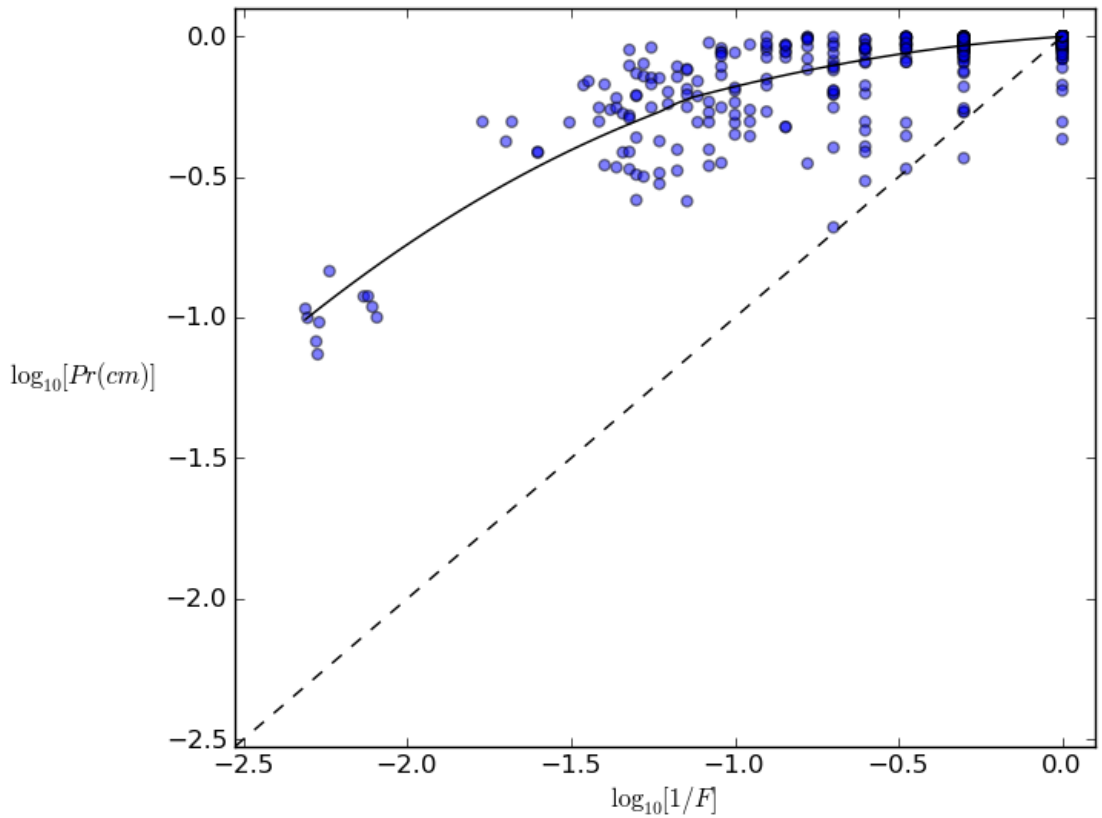


Figure 58. *Predictive performance for MC^3 with the 1% sample and 16 variables*

On the other hand the results for MCMCMC demonstrate a consistent overestimation of the true correct match probabilities (apart from population uniques). The plot for the simulated annealing run is not shown, but is very similar to that for MCMCMC, emphasising that the overestimation might not be due to chance. Additional simulated annealing runs with alternative cooling schedules demonstrate that this is not simply down to the relatively rapid cooling resulting from the fixed parameterization. It appears to be a more general result with sparse tabular data. However, the logit rank plot demonstrates that SA and MCMCMC show a marked advantage over SUDA and DIS-IS when it comes to distinguishing unique from non-unique population counts (given sample uniqueness). In this case MCMCMC and SA present almost identical results. The MCMCMC visited 8 distinct model (after burn-in), but SA found the highest probability model overall.

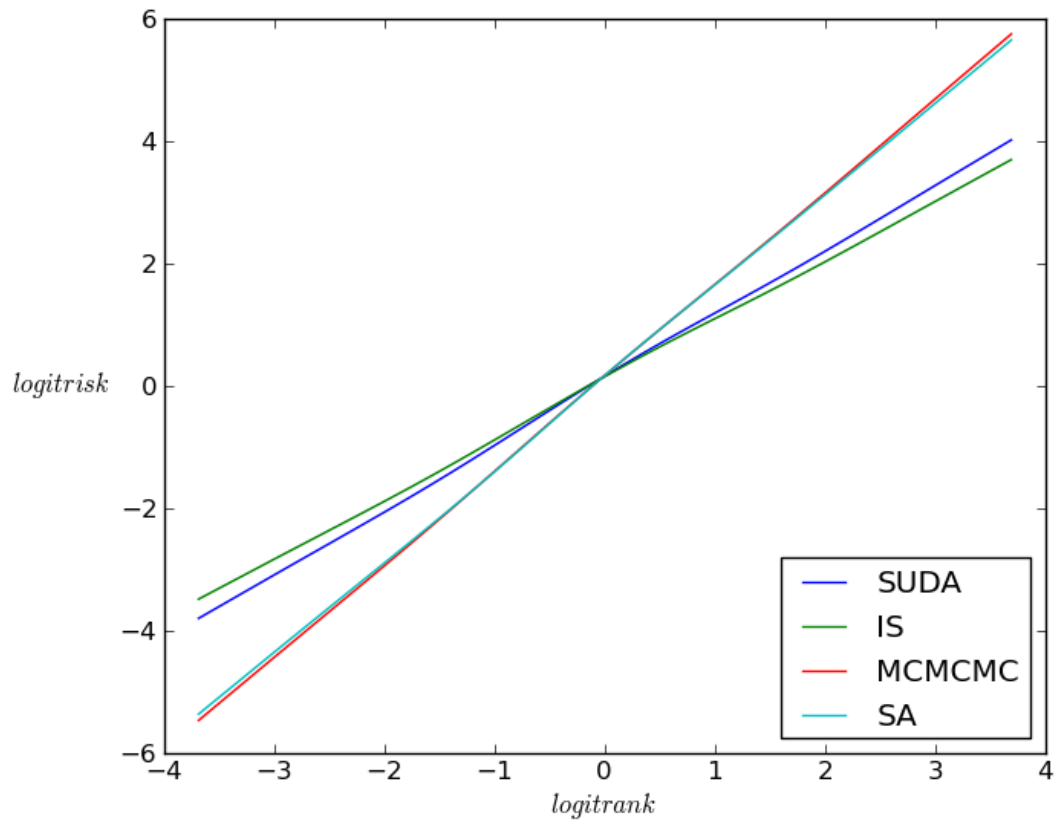


Figure 59. *Logit rank plot for the 1% sample with 16 variables*

5% sample with 8 variables

There is evidence of overestimation for both DIS-IS and SA. The SA results (not shown) are similar to the MCMCMC results.

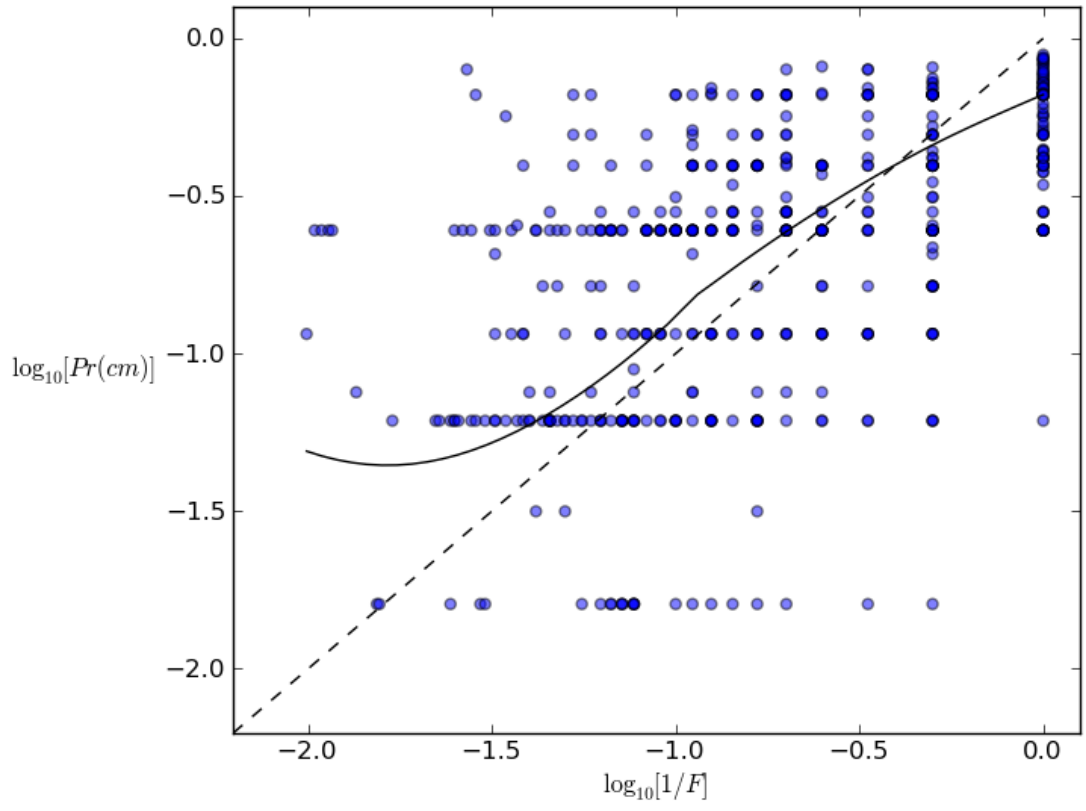


Figure 60. *Predictive performance for DIS-IS with the 5% sample and 8 variables*

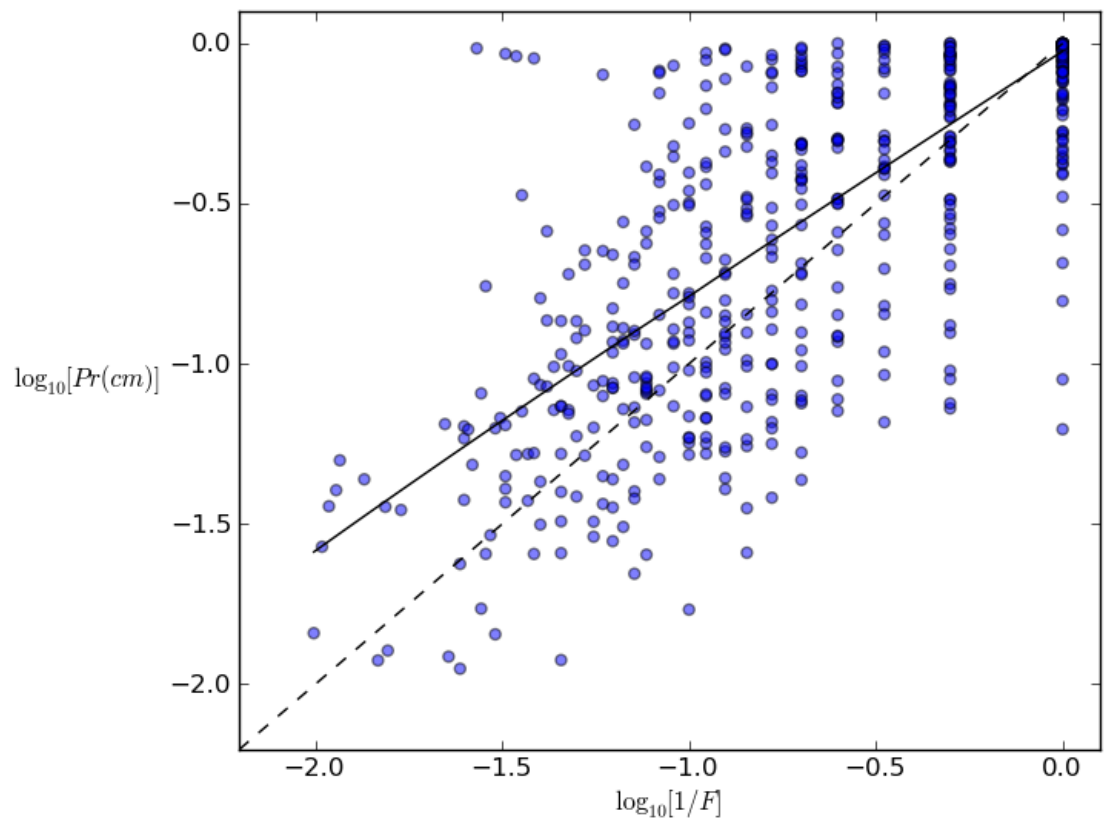


Figure 61. *Predictive performance for MC^3 with the 5% sample and 8 variables*

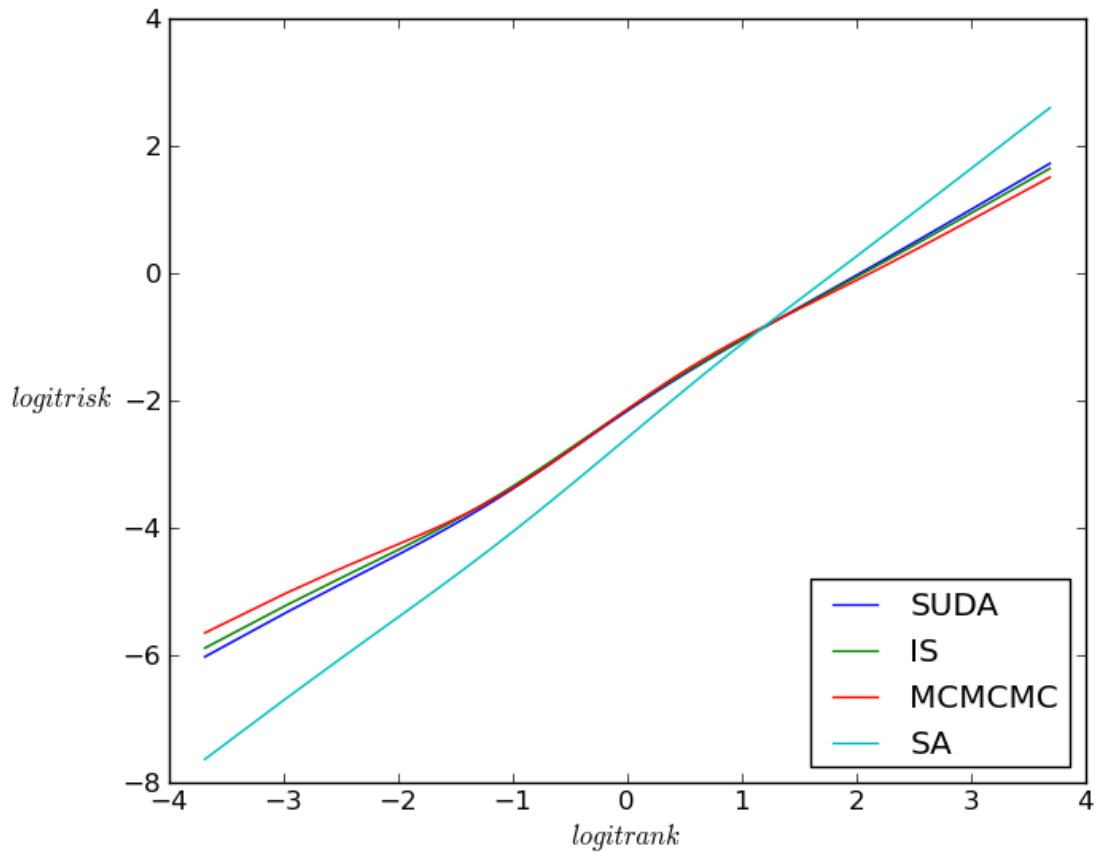


Figure 62. *Logit rank plot for the 5% sample and 8 variables*

The logit rank plot demonstrates that SA provides the better score, and MCMCMC performs worse than SUDA or DIS-IS. In this instance the MCMCMC run only visited two distinct models after the burn-in period, and SA produced the highest probability overall.

SA seems to be more reliable than MCMCMC, which suffers from mixing problems. Longer MCMCMC runs would produce more consistent results, and different cooling schedules might produce better results for SA. But the parameters were fixed during the 12 variable 1% sample run. This allows a comparison of how the various approaches might scale in computational terms. It also reflects the fact that users of the existing SUDA software might not feel comfortable performing diagnostics and tuning parameters in order to generate reliable results. It also gives an unfair advantage to the SUDA approach, making this a more stringent test of the MCMCMC and SA alternatives.

Costs

Computational costs will tend to vary with problem size for all the methods. For sparse tables there will tend to be more sample uniques to investigate. Sparse tables will also tend to have larger MSUs, which can be a good or a bad thing depending on the precise algorithm used to find them. The running times for the various algorithms are shown in Figures 63, 64 and 65. The MCMCMC and SA algorithms were calibrated to have similar cost to DIS-IS for the 12 variable 1% sample runs. However, the running times (MCMCMC in particular) were quite variable.

	Number of Variables			
Sample proportion		8	12	16
	1%	0.54	8.63	138.20
	5%	1.01	21.66	519.50

Figure 63. *DIS-IS running times (secs)*

	Number of Variables			
Sample proportion		8	12	16
	1%	9.62	10.66	34.62
	5%	16.72	48.90	50.76

Figure 64. *MC³ running times (secs)*

	Number of Variables			
		8	12	16
Sample proportion	1%	5.40	11.03	21.01
	5%	9.75	34.04	76.17

Figure 65. *Simulated annealing running times (secs)*

The DIS-IS results show increasing computational cost with the number of variables. Costs rise quite quickly as the tables become sparser. The effect of sample proportion is less clear, although the 5% sample runs carried out here took longer than the 1% sample runs. As population units are sampled unique sample counts appear, but a sample unique disappears when a further unit is drawn from the same equivalence class. In any case, the algorithm used here differs from both SUDA and SUDA2, so any general inferences about how the approach scales would have to be tentative.

The MCMCMC algorithm used the same parameters for all runs. Thus it was not affected by problem size as it would have been if appropriate parameters were used for each run. Costs do increase with the number of variables, but this is largely due to the increased number of sample uniques. There is no obvious reason why the model fitting should take significantly longer for larger problems. Much of the measured cost is taken up by generating the posterior probabilities from the discovered models. One potential issue is that there are more models to visit with larger numbers of variables, and previously cached calculations can be reused less frequently. More experimentation and code profiling will enable these issues to be investigated.

The SA costs are far less variable than the MCMCMC costs. They generally compare very favourably with the other approaches for larger problem sizes.

Risk measures

The SUDA / DIS-IS software generates record level risk measures for each sample unique. This is clearly the case for the MCMCMC and SA approaches. However, the commercial SUDA software also generates measures at the file, variable and category levels.

The basic building block of these measures is the IS metric. For each sample unique the IS metric is calculated for each MSU, the sum of these metrics constituting a record level risk measure. (This might be divided by the factorial of the number of variables to produce the POL measure.)

The file level risk measure is the sum of the measures for the sample uniques. In other words, the sum of the IS metrics for all the MSUs for all the sample uniques.

The variable level risk measure is calculated in the same way as the file level measure, except only the MSUs that contain the variable concerned are summed.

The category level risk measure is calculated in the same way as the variable level measure, except only the sample uniques that have the given variable at the given category level are summed.

The variable level and category level measures are generally divided by the file level measure to produce percentage contributions to the file level measure.

The above measures are not based on any underlying theory, or are intended to have anything other than the obvious interpretation stemming from their construction. It does indicate that users might want measures associated with files, variables and variable categories. There are certainly sensible file level measures that can be generated. In Chapter 5 a number of measures were discussed, some of which were based on similar outputs. It is simple to generate estimated probabilities of correctly matching from sample to population, or population to sample, given MCMCMC or SA results. A candidate for a file level risk measure would be the probability of a correct match given a unique match, $\Pr(\text{CM}|\text{UM})$. This is easily estimated.

Of more interest is how meaningful measures could be generated at the variable and category levels.

Variable level

A possible approach would be to assess the change in risk produced by removing a variable from a data release. A variable could be removed, an analysis repeated, and the change in the file level measure reported. If $\Pr(\text{CM}|\text{UM})$ was the file level measure in use, then the ratio of the measure given all variables and the measure given a reduced set of variables would provide a relative risk of including the variables in the set difference. As in the simulations above, it would be sensible to use the same sample. Both the MCMCMC and SA approaches generate a joint mass function over all the variables. The joint mass function over a subset of variables can be generated by marginalization. This is accommodated in standard Bayesian network inference algorithms (see Chapter 6). So there is no need to repeat analyses. Variable level relative risk measures can be generated from the already fitted model(s) considering those sample uniques that survive marginalization.

Category level

A variable level measure can be generated by marginalization. But marginalization simply involves collapsing across all the categories of a variable. Exactly the same approach could be used but only collapsing over a subset of categories. Thus the first two categories of a variable could be collapsed into a single category. Some sample uniques would disappear, but the inferences for those that remained would be different, leading to a different file level measure. Again, a relative risk measure could be produced. The issue here is which categories to collapse and how to generate a single measure for a single category.

The worth of being able to investigate the effect on risk of recoding a variable by collapsing categories is clear. That is exactly the type of measure that a statistical agency might consider for disclosure control. A possibility for generating a measure for a single

category would be to collapse the category with each other category and to average the resulting relative risk measures. Thus a total of $n \cdot (n-1)/2$ relative risk measures would need to be generated in order to generate single scores for each of n variable categories. For variables that are categorized interval scaled variables it would be sensible to only consider neighbouring categories, which would also reduce cost.

An alternative approach would be to simply not include any sample uniques that had the relevant variable at the given category level. A file level measure could be generated for the remaining sample uniques, which would be equivalent to the risk if the variable category was simply suppressed. A relative risk associated with the inclusion of the category would be the ratio of the file risk measure for full analysis to the file risk measure associated with the suppression of the category. This measure could be generated much more efficiently than the previously suggested measure based on recoding rather than suppression.

Another possibility would be to provide a desired reduction in risk and to carry out a search for recodings / marginalizations that reduced the risk sufficiently whilst minimising loss of utility.

Discussion

The SUDA and DIS-IS approaches have been used in statistical disclosure control for several years. They are surprisingly effective methods for identifying sample uniques that are likely to be population unique. However, evidence has been presented that approaches based on fitting models via Markov Chain Monte Carlo methods and simulated annealing offer improvements in terms of the accuracy of classification. Although all approaches tend to overestimate $\Pr(\text{CM}|\text{UM})$ for non-unique population counts, MCMCMC and SA approaches are better at distinguishing population uniques from non-uniques. MCMCMC results can be a little unreliable due to poor mixing of the chains. There is some evidence that the SUDA score distinguishes population uniques better than the IS metric, although it is not terribly evident from the simulations reported here. The experiments used a fixed parameterization for both MCMCMC and SA, based on trying to match computational

costs with DIS-IS for a particular problem. They would be expected to perform better if parameters were allowed to depend on relevant factors such as problem size and table sparseness.

It appears that MCMCMC and SA also scale better. SUDA and DIS-IS require a search for MSUs which can become very costly. The search can be limited to MSUs below a given size; and this option is available in the commercial software. However, this is likely to mitigate the problem, rather than to solve it. Sparser tables will tend to have more sample uniques, inevitably pushing up computational costs. Although this affects the MCMCMC and SA methods too, it is through the generation of estimated probabilities rather than through the model fitting.

There is an issue with mixing for MCMCMC. This seems to be worse for sparser tables. The analyses in Smith (2006) and in Chapter 7 used a lower number of variables and there were no particular issues with mixing. Alternative schemes might allow better mixing. This would allow the potential advantages of model averaging to be properly assessed. Nevertheless, even locally good models found by MCMCMC often perform well compared to SUDA and DIS-IS.

It should be noted that Forster and Webb (2007) adopted an alternative approach when estimating different risk measures that would normally require access to population data for calculation. They used a decomposable graphical modelling approach, but used a scheme that branches out from the full independence model to find a number of local optima. They then generated weights for model averaging, and produced model averaged estimates. This scheme was first presented in Madigan and Raftery (1994). This is a fairly pragmatic and efficient approach and would be worth comparing against SA and potential alternative MCMCMC schemes (with better mixing).

As the MCMCMC and SA approaches generate full probability models they can be used to generate risk metrics for smaller marginal and conditional tables. This offers the potential to generate meaningful file, variable and category level risk metrics. In particular, it has been shown how these metrics could reflect alternative options available to the DSO. That is, metrics could be based on potential recoding and suppression schemes. This raises the question of whether these metrics could be generalized to other options available to DSOs.

In principle the answer is yes. Smith and Elliot (2003) and Forster et al. (2008) both assessed rounding schemes using a similar Bayesian approach. The difference would be that model fitting runs would need to be repeated.

Strictly speaking the variable and category level measures suggested could also involve refitting. They assume that refitting after suppression of a variable or category will produce posteriors that are marginal or conditional distributions of the model fitted to the complete table. This is not necessarily the case. But refitting would introduce model uncertainty that could render the measures less reliable. A relative risk different to 1 might be explained by the variability in posterior distributions that would be experienced across multiple fits on the same data.

The fact that MCMCMC and SA produce full probability models also increases the options for the metrics used. DIS-IS is restricted to $\Pr(\text{CM}|\text{UM})$. Yet arbitrary types of measure can be generated from full probability models.

CHAPTER 9

Summary

Motivation

The main assertion underlying this body of work is that any person or organization wanting to assess disclosure risk should put themselves in the position of the data intruder. Then, risk should be assessed in terms of what an intruder can reasonably infer from the data that are available. This is not, in itself, a novel argument, but there is a tendency – in both research and practice - to focus on simple data-centric measures that do not reflect true risks. Also, some existing approaches do not fully take into account the inferences that could be made by a knowledgeable intruder. In some instances this is because they use modelling approaches which can be improved or because they use inefficient algorithms leading to long running times and / or unnecessary approximations.

The core of the thesis is several relatively self-contained chapters demonstrating these issues and offering new or improved approaches.

Core chapters

U.S. Department of Commerce (1978) discussed the risk of attribution and the mechanism by which it occurs over 30 years ago. Much work in SDC since has concentrated on the risk of identification. Disclosure implies making previously unknown inferences about individuals. This is attribution, and attribution can occur without identification. Exact attribution requires that the intruder knows that a target is a unit in a relevant data release, but does not require a correct match against a published record. Knowing that an individual is male and on a low income would allow the inference that he is unemployed if all males on low income within the population were known to be unemployed from a data release. It

is the fact that there are no employed males on low income that makes the inference possible. An exact match against a population unit would clearly allow the same inference, and also inferences regarding any other published attributes. This raises a second point. Identification is not necessary for attribution, but neither is it sufficient. If all the information in a released record is required to generate a correct match, then there is no remaining information to infer about the target. Identification occurs, but there is no attribution. A DSO might still be concerned that an individual might be identified, even if it is self-identification, because of the possible perception of disclosure. It is important to distinguish real disclosures of information from perceptions of disclosure.

The SAP measure was developed with this in mind. Much emphasis had been placed on cell counts of 1 in released tabular data. Little attention has been given to the presence of the zeros which actually allow attribution to take place. A number of different measures could have been chosen. For released population data containing zeros there is a clear risk, and simple measures such as the proportion of zero count cells could be used. At the time the method was developed the data releases that were being considered were perturbed releases of population data at small levels of geography. Thus there was also the possibility that the intruder might know some individuals within the population and be able to exploit this information. So the chosen SAP measure was the probability of being able to recover one or more zeros after removal of k randomly selected individuals from the population. It was assumed that an intruder would know the perturbation scheme, would have some local knowledge, and would be able to exploit this by recovering zeros in a table for unknown individuals. The measure was quite easily calculated for simple releases, and an efficient approach was developed for the specific forms of release that were under consideration. Generalizations were also considered. The main issue that was being addressed was the lack of a suitable measure for the risk of attribution. It was felt that attribution risk was more relevant than identification risk for data at low levels of geography where an intruder might have substantial information on the local population. A neighbour discovering previously unknown information about a target would have had more impact than self-identification.

The chapter on the new Key Variable Mapping System (KVMS) showed how an existing piece of SDC software could be improved. Spreadsheets are often awkward for a user, as

there is inadequate separation of concerns. Data, application logic and user interface are lumped together. A user can make arbitrary changes to the spreadsheet and consistency checking data is difficult. Usability is severely compromised by having to navigate around a large number of distinct worksheets to accomplish relatively simple objectives. Appendix 1 contains a KVMS tutorial which demonstrates the data entry system, designed to prevent users from entering inconsistent information. It demonstrates functionality that was not described in the KVMS chapter. However, the more interesting developments related to a more concrete, mathematical specification of the objectives of Key Variable Mapping and the data structures and algorithms it led to. This led to significant performance improvements.

KVMS (Elliot et al., 2010) concerns the possibilities for matching that are available across data sets contained within a data environment. The environment stems from a consideration of intrusion scenarios. The data sets generally stem from questionnaires and it is the possible answers to questions that can be matched on. It is not always the case that a question corresponds to a variable and that the possible responses are mutually exclusive and exhaustive. Sometimes it is necessary to consider all the possible responses before the meanings become clear. For instance, 'terraced house' would not include 'end terrace' if that was a distinct option. The decision was taken that the new KVMS would deal with variables with mutually exclusive and exhaustive categories. Thus the user would have to sort out ambiguities such as the precise meaning of 'end terrace'. This, and the consistency checking, helps to ensure that categories for a variable within the KVMS have a clear and distinct meaning. Once it is clear that the KVMS is dealing with variables and valid categorizations, then the possibilities for matching can be specified mathematically. This enables the development of improved data structures and algorithms.

Aggregation graphs are a simple idea, but they allow valid categorizations to be specified easily and checked automatically (assuming the existing graph is correct). Any edits to the graph (to accommodate new categories) can be checked for validity (consistency with existing categorizations). Harmonization can be defined precisely, allowing a very efficient means of harmonising categorizations by generating set partitions using the aggregation graph and using simple graph sum operations on graph representations of the set partitions. An efficient algorithm for constructing harmonization graphs allows these structures to be used for generating analysis outputs. Although the analysis is the same as in the original

KVMS, the harmonization graph could be used to answer other queries regarding matching possibilities. This offers the potential to extend the functionality of the system. The performance improvements will allow much larger data environments to be assessed in reasonable time.

The chapter on evaluating strategies for matching sample and population units builds upon work presented at the International Conference of the Royal Statistical Society, 2006. It is an attempt to find out what matching strategies might be most useful to an intruder. It looked at the specific strategies underlying some commonly used risk measures and considered more general forms of strategy and their probability of success. One of the main results was that matching from sample to population dominates matching from population to sample, although there are degenerate cases where the probability of a correct match is the same for both strategies. It was also found that similar dominance relationships were evident for the costs of search, and that these held whether search was costed in terms of sampling with or without replacement. In fact, for the perhaps more realistic costing with replacement measure, the ratio of the probability of a correct match to the search cost (efficiency) was constant for matching from sample to population, equal to the reciprocal of the population size. It was shown that simple strategies could be constructed with widely varying probabilities of success, and therefore costs. In contrast to the “no free lunch” of matching from sample to population, matching from population to sample could offer e.g. improved probabilities of success for no additional cost. This would involve attempting to find a larger number of equivalence classes without increasing the sum of their population counts.

A Bayesian approach was shown to offer very high probabilities of a correct match for anything but very small sampling fractions. Essentially the Bayesian approach allows equivalence classes to be ranked so that those with lower expected population counts can be included in a strategy. The efficiency of the strategy is fixed for matching from sample to population, but simulations showed that matching from population to sample could be made much more efficient without significantly impacting the probability of a correct match. The Bayesian approach also offers a means of trying to maximize the probability of a correct match for a given cost. Trying to match the cost of simply selecting the sample unique equivalence classes tended to produce strategies with lower probabilities of a

correct match (at lower costs). This was due to the Bayesian approach tending to underestimate the population counts and thus including sample non-unique cells. Nevertheless, this issue could have been identified if the expected counts had been examined, as there tended to be a noticeable difference in the expected counts for sample uniques and sample 2s. But the important point is that the Bayesian approach allows probabilities of a correct match and costs to be traded off. An intruder using this approach can tailor the attack strategy. Moreover, it highlights the importance of how efficiently the intruder can search the population. An intruder who can search the population efficiently can generate correct matches with very high degrees of confidence.

The chapter describing graphical models and the Metropolis-Hastings and simulated annealing schemes is needed to understand the details of the Bayesian approach. Other approaches are also based on the idea of adding and removing single edges from decomposable graphical models. But there is one area where the schemes differ, and that can make a large difference to computational costs. A move to a proposed model must be made with probability zero if the proposed model is not decomposable. There are a number of algorithms for testing for decomposability contained in the references. But performing any of these on each iteration of a Metropolis-Hastings run would be expensive. Giudici and Green (1999) realized this and used a junction tree to enable checks that only relied on local structure. Checks were performed much more efficiently than using global checks with existing algorithms. But junction trees were never designed to be dynamic structures and were not optimized for this purpose. Although dynamic junction trees were not designed with model determination in mind (e.g. Smith, 2001) they were always designed to be dynamic analogues of junction trees. They were actually initially designed to facilitate finding good triangulations for Markov graphs. Nevertheless, they are perfectly suited to decomposability checks given the addition or removal of an edge. No empirical evidence of their superiority is provided as this would require a comparable implementation using standard junction trees. However, comparison of the algorithms required suffices here. Establishing whether two nodes are in the same connected component and whether two nodes are present in a single clique are both steps that imply breadth first searches in a junction tree. In dynamic junction trees (or forests) the former is achieved by iterating through parental nodes to find the root(s), and the second is achieved by a simple depth first traversal of a subtree. Although a separate union-find data structure

could be used alongside a junction tree to eliminate the former breadth first search, it would involve having to update two data structures rather than one. There are also means of improving performance in identifying whether an edge is a member of exactly one maximal clique by maintaining a mapping of variables to the maximal cliques of the junction tree that contain them. But the dynamic junction tree representation encapsulates all these data structures in one. The relevance of this to SDC is the performance that it offers. A naïve implementation using one of the standard algorithms that could be used to check for decomposability would be far less efficient. A testament to this is the fact that Giudici and Green found it very much worthwhile finding something better. A testament to the performance of dynamic junction trees is that for Markov Chain Monte Carlo Model Composition the performance bottleneck is the calculation of gamma functions, even though they are coded in Fortran and results are cached for reuse. Efficiency means being able to risk-check more data potential data releases in a given time, or being able to check the same releases more thoroughly.

The final core chapter compared DIS-IS with an alternative Bayesian approach. DIS-IS is an approach used to generate probabilities of a correct match for sample uniques when the population counts are unknown. It was shown via simulations that Bayesian approaches using Markov Chain Monte Carlo and simulated annealing offer both increased performance and better results than SUDA and DIS-IS. The Markov Chain Monte Carlo approach was almost identical to the Metropolis-Hastings scheme used for estimating correct match probabilities. Although simulations showed that it tended to perform better than DIS-IS the mixing of the Markov chain tended to be poor. Thus a simulated annealing approach was also investigated. DIS-IS had lower running costs than MCMCMC and SA for small numbers of variables. But for larger problems where cost is more significant SA outperformed DIS-IS and MCMCMC by a significant factor. Better estimation at lower computational costs is a worthwhile achievement in itself. However, MCMCMC and SA approaches produce full probability models. This allows the generation of arbitrary risk measures, rather than just $\Pr(\text{CM}|\text{UM})$. In fact Forster and Webb (2007) have used a similar MCMCMC scheme to estimate other risk measures in the absence of population counts. This also allows more meaningful measures to be generated for variables and variable categories. Most of the measures generated by the SUDA software (the commercial implementation of DIS-IS) have no simple interpretation in terms of disclosure

risk. The MCMCMC and SA approaches allow measures to be produced that have direct interpretations in terms of relative risks. What is more, they can be generated for various recoding, suppression and rounding schemes. These are distinct advantages over the existing DIS-IS scheme.

Policy implications

The work on attribution risk in tabular data revisits concerns first raised in U.S. Department of Commerce (1978). It emphasizes that SDC is not only about identification risk, and that it is important to put oneself in the position of an intruder when assessing risk. The original motivation for the measure stemmed from exactly that type of consideration. The method was originally used to risk assess proposed releases for the UK Office for National Statistics (Smith and Elliot, 2003). Published Office for National Statistics guidance on disclosure control will now often be found to refer to the risk from table zeros and scenarios where intruders use known information to make inferences about residual populations⁴. Available software would increase the impact on policy, allowing risk measures to be generated. Nevertheless, concern over the issue of zeros does appear to be more prevalent than it once was.

The new KVMS has been implemented in software with a user friendly interface. It is not of production quality, and some desirable features such as automatically generating aggregation graphs for variables on a numeric ordinal scale have not been implemented. Nevertheless, it does provide distinct advantages over the original spreadsheet model. There is potential for further development. It is being tested at the University of Manchester and is being used to construct a data environment from the same forms used for the original KVMS. It is a useful tool that could be made more useful with some extra work. Available production quality software (whether free or commercial) would increase its impact on policy.

The work on matching between samples and populations certainly provides some insight into how a knowledgeable intruder might launch an attack. Using the Bayesian approach

4 e.g. www.statistics.gov.uk/about/data/disclosure/downloads/GSS-Microdata-Policy.pdf

an intruder can trade off the probability of a correct match against search costs, rather than sticking to one of the standard scenarios where both probabilities of success and costs are fixed (although unknown to the intruder) by the population and sample data. Risk assessment could also exploit the trade-off rather than sticking to standard attack scenarios. In particular the work drives home the potential issue of easily searched population data. This might require a specific type of intruder. Nevertheless, it is possible to generate correct matches with extremely high degrees of confidence. The only thing discouraging such an attack is the cost of searching the population for a potential match. Potential search strategies deserve more attention.

The MCMCMC and simulated annealing alternatives to DIS-IS need to be implemented in software in order to have much impact on policy. They offer better ranking (in terms of riskiness) of sample uniques, more easily interpreted risk measures, and the potential to deal with more variables through improved scaling of the computational costs. But without available software these advantages would only be slowly realized. DIS-IS is used by national statistical agencies. There is no reason to think they would not appreciate something of a similar nature with large performance improvements.

There are perhaps two main implications for the detailed implementation work using dynamic junction trees. Firstly, it directly improves the performance of the Bayesian matching approach and both the alternatives to DIS-IS. Secondly, it has the potential to produce new schemes which avoid the issue of poor mixing when dealing with more than half a dozen or so variables. It has much wider implications than record matching and the alternatives to DIS-IS. This requires extra work. There are other advantages to dynamic junction trees, but they are less directly relevant to SDC.

Future work

The new KVMS was designed to generate the same analysis outputs as the original KVMS. But now that harmonization graphs are used as the basis of generating these outputs it opens the way to alternative analyses. At the moment a target data set is specified; and a single threshold parameter, p , is specified; and the output is an arbitrary

maximal harmonization, h_v , for each variable, v , in the target data set such that a fraction p of the forms that contain v harmonize to h_v . The new KVMS would be able to generate plots of maximal harmonization sizes against p . It was shown that a harmonization graph for a superset of a data environment was also valid for the data environment. Thus all data sets could be entered into a super data environment and analyses performed on subsets of forms. This is already implemented in the new KVMS via the specification of groups. But it suggests that a large single super-environment could be maintained to ensure consistency of variable naming across environments and to eliminate duplication of work specifying aggregation graphs. In such cases it might be worthwhile saving the harmonization graphs. At the moment they are constructed from saved aggregation graphs and form data when the application is opened. This works well enough for a data environment with about 200 forms.

A harmonization graph has a number of properties that could be useful for generating other outputs. For instance, the graph induced by the nodes in $\{\{v\} \cup \text{anc}(v)\} \cap \{\{w\} \cup \text{anc}(w)\}$, where $\text{anc}(x)$ denotes the ancestors of x , has a single leaf node which is the harmonization of v and w . In fact, this can be extended to more than two nodes. So once a harmonization graph has been constructed for a variable the harmonization for any set of categorizations can be looked up relatively efficiently. Although the alternative graph sum operation on the categorizations is hardly inefficient, having all harmonization relationships between all sets of categorizations encoded in a graph suggests that many useful outputs might be generated via simple graph traversal algorithms. The current analysis output is just one example.

The dominance relationships for matching between samples and populations were derived for any population and any possible sample from the population. So they hold even for degenerate samples. If we were to take into account sampling distributions other dominance relationships might exist. For instance, strategies based on equivalence classes with low (but non-zero) sample counts do not dominate strategies based on equivalence classes with higher sample counts. Yet it is intuitive that they will generally produce higher correct match probabilities. The work could be extended to investigate this.

It has been shown that the MCMCMC scheme is problematic for some data sets. It performed well for the matching simulations, but badly for the DIS-IS simulations where

the chains mixed very poorly. It seems to perform well for small sets of variables, but for more than about half a dozen variables mixing can become poor. There are possibilities for improving this. An obvious potential solution would be to generate candidate moves on the basis of adding or removing more than a single edge. It is easy to construct a valid chain, but generating Bayes factors would involve more computation, as moves would be less localized. An alternative is to add or remove more than one edge but retain the local updating of Bayes factors. A solution using dynamic junction trees (or forests) has been implemented for inhomogeneous Markov chains. It has not yet been tested. A solution for homogeneous Markov chains has been developed, but involves an implicit uniform prior over the space of dynamic junction forests rather than over the space of decomposable graphical models. This might still produce good results, generating a useful posterior distribution. This could be tested. But a little extra work might allow the proposal distribution to be adjusted to allow the same implicit uniform prior over the space of decomposable graphical models that is used in other schemes. Thus the comparison with existing schemes would be more meaningful. One option is to try alternative schemes for generating plausible models and applying model averaging to those. The scheme used in Forster and Webb (2007) and due to Madigan and Raftery (1994) might be useful for this. Finding a good scheme would allow model averaging to be compared against simulated annealing. It is also likely that schemes such as Madigan and Raftery's will be significantly more efficient than MCMCMC, and perhaps even quicker than simulated annealing. Simulated annealing is a global search algorithm specifically designed to avoid local optima and therefore does not suffer from the mixing issue. The disadvantage is that it finds a single model, foregoing the potential benefits of model averaging. It does, however, provide good solutions in reasonable time, and will be useful for comparing with the MCMCMC approaches listed above.

Final comment

It is difficult to argue with the position that SDC should focus on the inferences that can be made by a data intruder. Some of those inferences would constitute disclosure. But in order to properly assess risk the assessor should have access to the best performing tools available. In essence there is a kind of arms race. A knowledgeable intruder could do much

better in, say, matching between sample data and a population than would be suggested by some standard attack scenarios and risk measures. It is important to know what can be achieved by an intruder today, and to bear in mind what might be achievable in the future with increased computational resources or better algorithms.

A lot of risk assessment is carried out by statistical agencies that might not always possess the expertise to implement the type of methods discussed here. So it is important that software tools be made available to them. In essence, they need to be armed. This work has shown that there is potential for better tools using better models and better algorithms. In order to have a significant impact on policy they need to be made available in software.

APPENDIX 1

KVMS Application Tutorial

Introduction

Data environment analysis coupled with the Key Variable Mapping System (KVMS) is a methodology designed to identify sets of key variables within a data environment. It is discussed in Elliot et al. (2010). The method was originally implemented in a spreadsheet. This tutorial describes an implementation of KVMS written in Python and available for Windows, Linux and Mac (currently untested on Mac). The screenshots show the application running on Ubuntu (Linux). It contains several extensions to the spreadsheet version, simplifies data entry, and contains many checks to avoid the entry of inconsistent data.

On start up

On starting the application for the first time 4 new directories are created in the application directory. These are *Forms*, *Aggregations*, *Groups* and *Backup*. The first 3 are used for application data whilst the 4th is used to store all application data from the previous session. Files in *Backup* are overwritten each time the application is used (when it is closed), so the directory should not be used for important backups. It simply provides a 'last known good configuration' set of files that can be copied back to the data directories if the application cannot parse the existing data files on opening. Although data entry is designed to avoid inconsistencies in the data it is possible to create inconsistencies if the data files are edited manually or the data environment is edited via the shell.

The interface contains a notebook with 5 tabs.

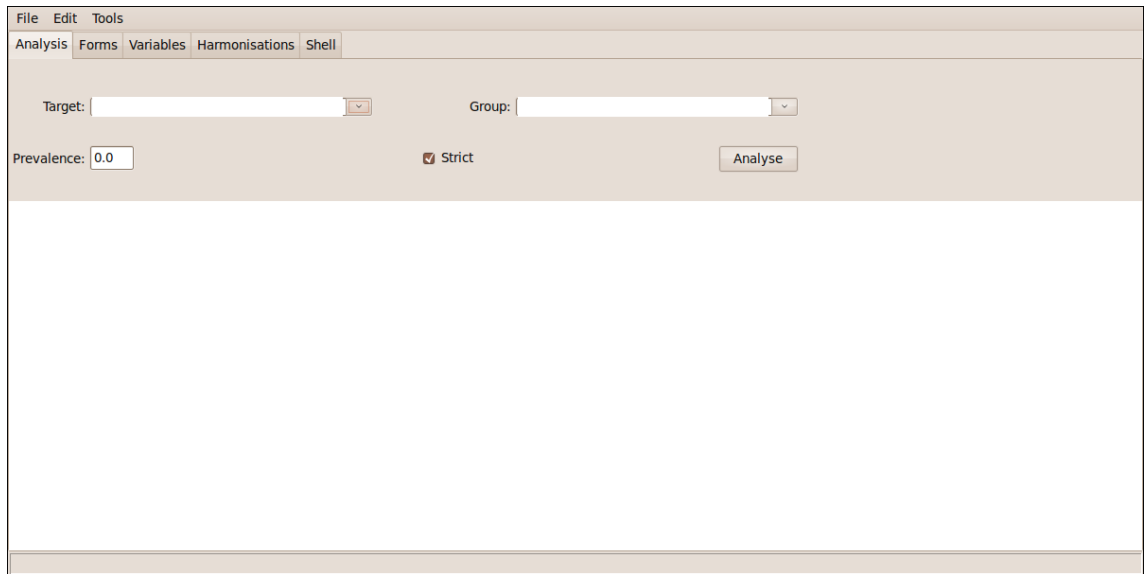


Figure 66. *The Analysis tab*

The Analysis tab contains a panel used for performing analyses.

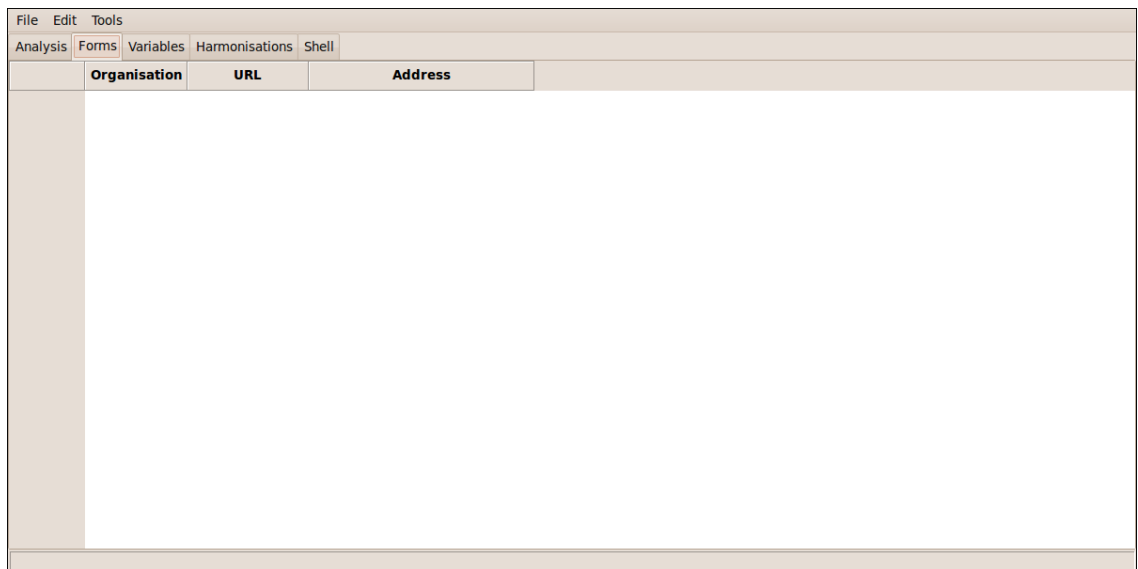


Figure 67. *The Forms tab*

The Forms tab contains a grid for displaying form data. Initially there is no data to display and it only contains 3 empty columns for form metadata. The column labels are specified

in *config.cfg* which can be manually edited to specify alternative column labels.



Figure 68. *The Variables tab*

The Variables and Harmonizations tabs contain notebooks which have tabs for each variable. Initially these notebooks have zero tabs.

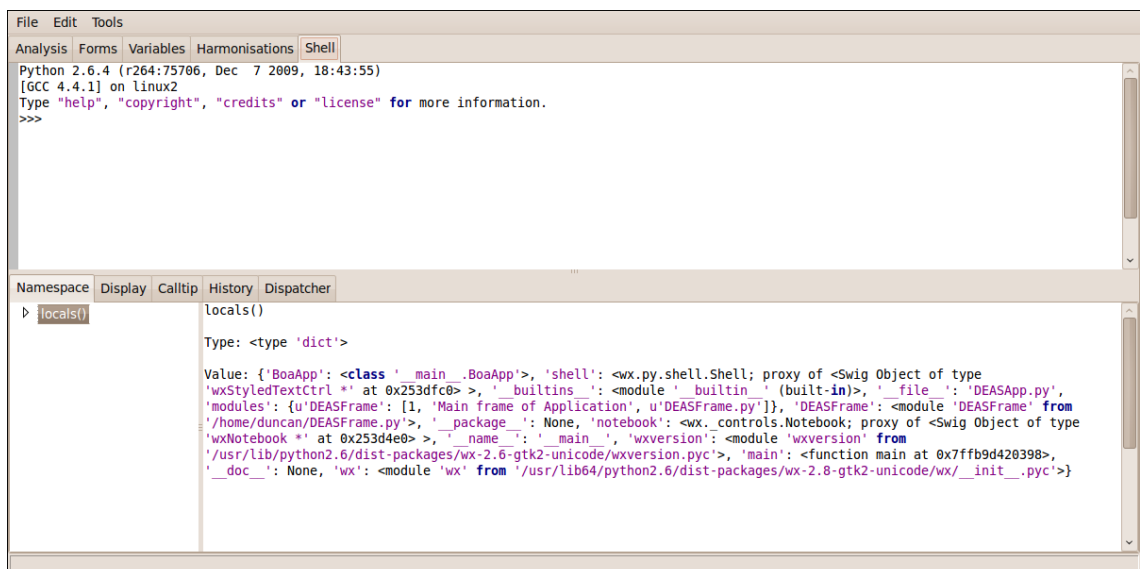


Figure 69. *The shell*

The 5th tab contains a Python shell. This can be used to interact with the data environment directly, although only certain types of interaction will be reflected in the user interface. For instance, it is perfectly possible to add new forms via the shell, but the changes would not be reflected in the user interface until the application was restarted.

On starting, all the application data is loaded into a namespace. User actions (via the user interface or the shell) can edit the namespace. On closing the application the data in the namespace are dumped to the data directories after moving any existing data files to the Backup directory. So saving and opening data is automatic.

Data entry

The *Edit* menu should be used for data entry. Attempting to enter data directly via the data grids will have no effect on the data environment, and will only result in incorrect data being displayed. In the future these might be made non-editable.

The Edit menu contains *Forms* and *Groups* menu items. The Forms menu item is used for all data entry other than groups. The general procedure is to create a form, add variables to the form, and specifying the categorizations for the variables.

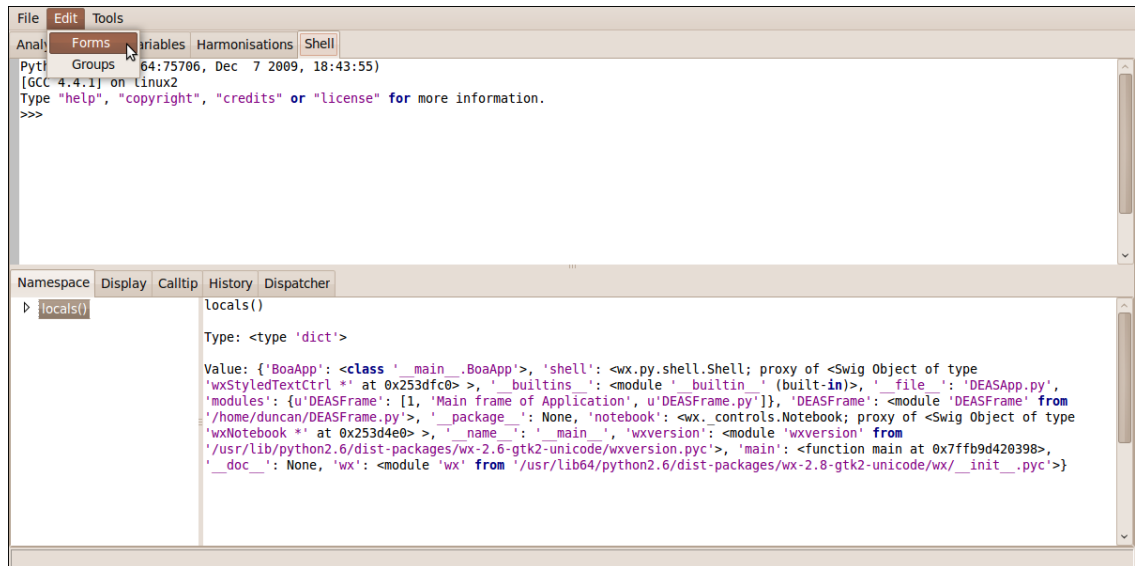


Figure 70. *The Edit menu*

The screenshot shows the 'Form editor' dialog box. It contains four text input fields: 'Name' (with a dropdown arrow) containing 'Survey X', 'Organisation' containing 'Pollsters R US', 'URL' containing 'www.pollsters.org', and 'Address' containing '123, Railway Cuttings, East Cheam'. To the right of these fields are four buttons: 'Variables', 'Metadata', 'Clear', and 'Delete'. At the bottom right are two buttons: 'Close' and 'Add'.

Figure 71. *The Form editor*

A form name must be entered into the Form Editor. Metadata are optional. The editor provides text controls for the names specified in the configuration file. Other name, value pairs can be entered via a grid that can be accessed via the *Metadata* button. The *Clear* button will clear the controls ready for entry of a new form. The *Delete* button will delete an existing form from the data environment.

Although initially a form name must be entered manually, once forms have been added they can be selected by name from the combobox. When a form is selected in this way it becomes the 'working form' and can be edited before saving. In fact, a copy of the existing form is created which can be freely edited. The changes are only committed to the data environment once the add button is clicked. The changes are not committed to file until the application is closed. To work with a blank form after selecting an existing form via the combobox the editor must be cleared by clicking on the *Clear* button.

To recap, the editor initially has a blank working form, selecting a form via the combobox selects a copy of an existing form as working form, and clearing the editor creates a new blank working form. The editor does not close when a form is added. It must be explicitly closed once the user has finished with data entry.

To add or edit the variables associated with a form a Variable Editor is invoked via the *Variables* button.

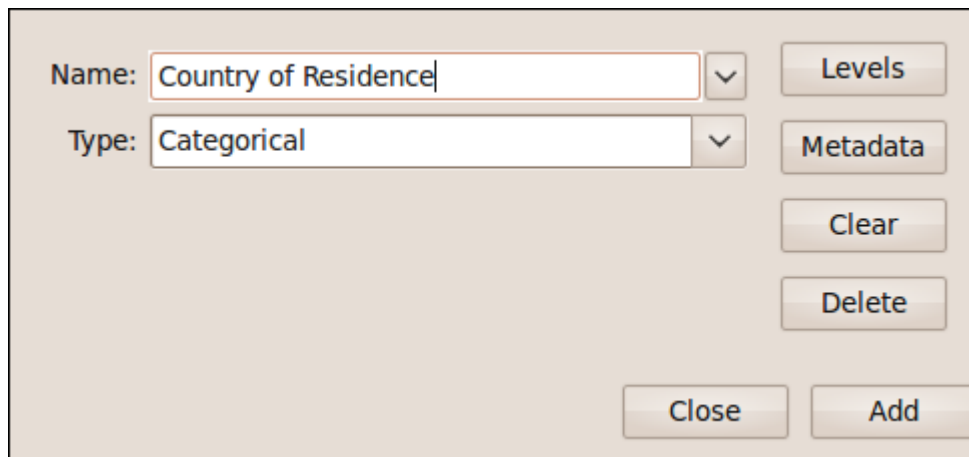
The image shows a 'Variable Editor' dialog box with a light beige background. It contains two rows of labels and text boxes. The first row is labeled 'Name:' and contains a text box with 'Country of Residence' and a small downward arrow. The second row is labeled 'Type:' and contains a text box with 'Categorical' and a small downward arrow. To the right of these text boxes are four buttons: 'Levels', 'Metadata', 'Clear', and 'Delete', arranged vertically. At the bottom right of the dialog are two buttons: 'Close' and 'Add'.

Figure 72. *The Variable Editor*

The Variable Editor is similar to the form editor. Initially it contains a blank working variable. But a copy of an existing variable can be made the working variable via selection

from the combobox. The combobox separates variables already contained in the form from other variables contained in other forms so that the user can identify the variables already added to a form. The buttons operate similarly to those in the Form editor. Clearing the editor makes a new blank variable the working variable. Deleting a variable removes it from the working form. Essentially, the form is the data environment for the variable. No changes (other than variable deletion) are made to the form until the *Add* button is clicked. Note: no change is made to the underlying application data environment until the form is also added. To edit a variable in a given form the user would have to open a copy of the form, select and edit the relevant variable, then add the variable to the form (overwriting the existing variable instance), and then add the form to the environment (overwriting the existing form).

The second combobox should be ignored in the current version. Only 'Categorical' can be selected, but in the future it might be possible to select an interval scale for suitable variables, allowing automation of the creation of aggregation graphs. An aggregation graph is a directed acyclic graph with nodes for variable levels and the properties that a parent is equal to the union of its children and the children are mutually exclusive. Thus Britain could be a parent of England, Scotland and Wales. Aggregation graphs contain information regarding the relationships between different categorizations. Each variable has a single aggregation graph.

The categorization for a variable is specified via an Aggregation Editor which is invoked by clicking the *Levels* button. This is the 3rd and final level of data entry.

	Name
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	

Figure 73. *The Aggregation Editor*

The Aggregation Editor consists of a canvas and a grid. These are used to specify the categories for a variable and the relationships between different categories in different variable instances. The category names are entered into the grid, whilst an aggregation graph can be created / edited via the canvas:

- Right click canvas to add a node
- Single left click to highlight a node or edge
- Highlight a node and hold shift key down whilst left clicking on another node to add an edge from the highlighted node to the clicked node
- Hit delete key to delete a highlighted node or edge
- Double click a node to toggle selection status (unselected nodes are white, whilst selected nodes are orange)

After creating / editing an aggregation graph the relevant variable levels are selected on the canvas.

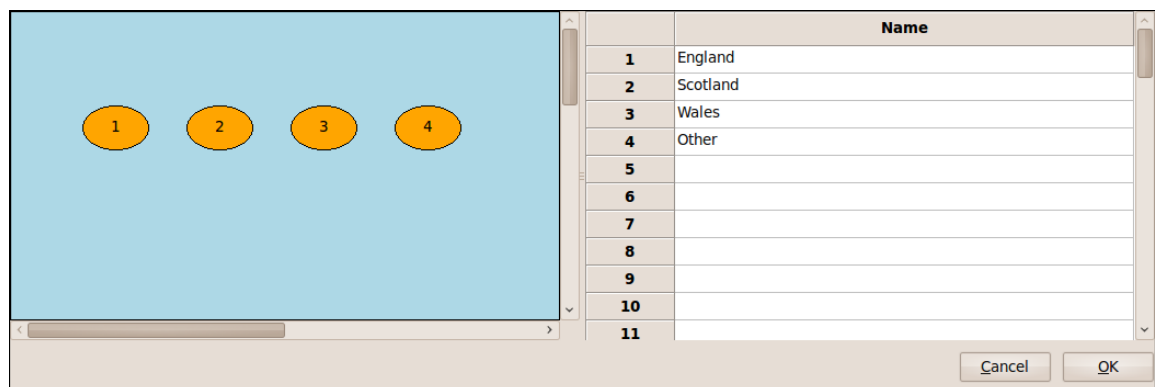


Figure 74. *The Aggregation Editor with selected categories*

In Figure 74 the categorization of Country of Residence is England, Scotland, Wales and Other. Clicking the OK button adds the categories to the working variable and closes the editor. Any changes to the aggregation graph are also saved (but again, only saved to the

underlying data environment when the form is added). If none of the nodes is selected the variable contains no levels in its categorization. This is valid, and represents a form variable where the respondent is invited to enter any value they want.

After adding the variable to the working form and adding the form to the data environment the user interface is automatically updated.

A second form was added by loading the existing form and variable in the editors, editing the aggregation graph, specifying an alternative categorization, and changing the form name before adding to the data environment. Figure 75 shows the edited aggregation graph.

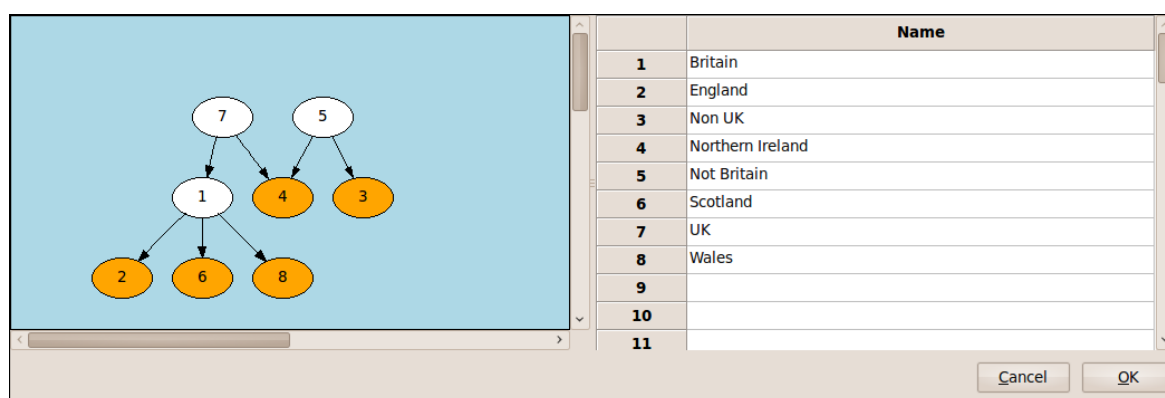


Figure 75. *The edited aggregation graph*

The 'Other' category was renamed to 'Not Britain' and 'Northern Ireland', 'UK', 'Non UK' and 'Britain' categories were added. The categorization of this second form for Country of Residence is England, Scotland, Wales, Northern Ireland and Non UK.

The editor only allows valid changes to be made to an existing aggregation graph. Thus it is important that the first time an aggregation graph is specified the categories are mutually exclusive and exhaustive. If the Other category had not been specified originally, then creating the new categorization would not have been possible. Note that a valid

categorization is a set of nodes such that all the leaf nodes (nodes with no children) are reachable (along directed paths) from the nodes in the categorization, and each leaf node is reachable from exactly one node in the categorization. Note that this is still true, in Figure 75, for the categorization in the first form; England, Scotland, Wales and Other (renamed to Not Britain). Renaming of categories is performed globally, renaming the category in any instances of the variable contained in the data environment.

The following figures show the user interface after adding several more forms containing various categorizations of Country of Residence.

File Edit Tools						
Analysis Forms Variables Harmonisations Shell						
	Organisation	URL	Address	Form Type	Country of Residence	
Survey V	Pollsters R US	www.pollsters.org	123, Railway Cuttings, East Cheam	On-line	X	
Survey W	Pollsters R US	www.pollsters.org	123, Railway Cuttings, East Cheam	On-line	X	
Survey X	Pollsters R US	www.pollsters.org	123, Railway Cuttings, East Cheam	Paper	X	
Survey Y	Pollsters R US	www.pollsters.org	123, Railway Cuttings, East Cheam	Paper	X	
Survey Z	Pollsters R US	www.pollsters.org	123, Railway Cuttings, East Cheam	On-line	X	

Figure 76. *The Forms grid with 5 forms added*

When a form is added to the environment a row is added to the forms grid and new metadata and variable columns are added as needed. The reverse happens when forms are deleted. Figure 76 shows that form metadata on *Form Type* have been added.

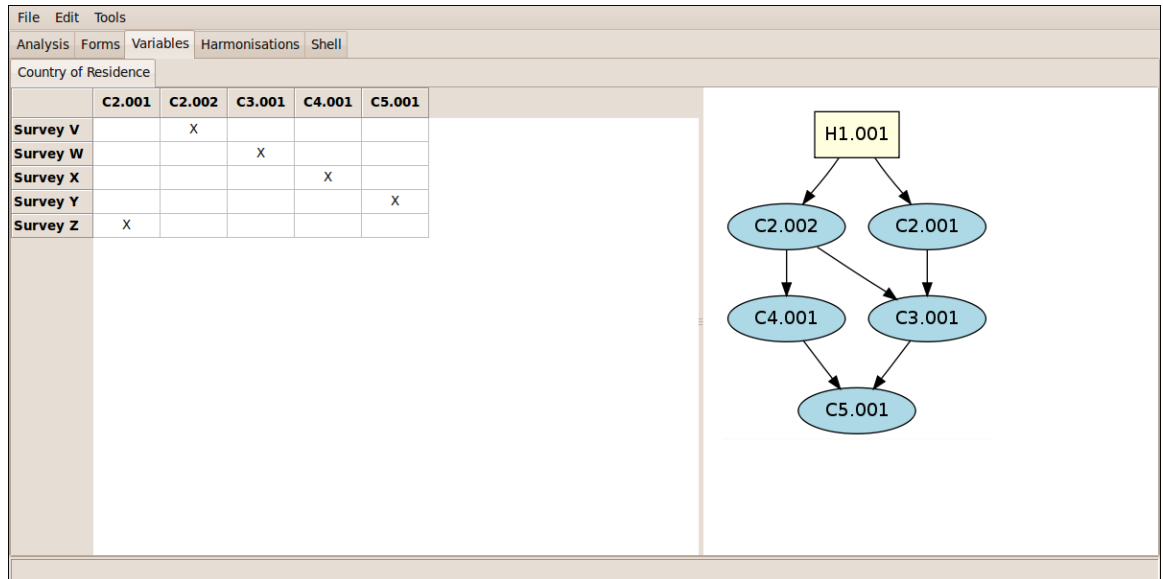


Figure 77. *The Variables notebook page for Country of Residence*

When a variable is added to the data environment a new tab is added to the notebook under the Variables tab. This contains a grid and a canvas. The grid contains information about variable codes (corresponding to categorizations). The codes are of the form CX.Y, where X is the number of categories and Y is a positive integer (less than 1000) which distinguishes between codes with the same number of categories. The canvas contains a harmonization graph where an edge from a code v to a code w implies that w can be aggregated to form the categorization v (and therefore any of v 's ancestors). This graph is unique and the ancestors of a node represent exactly the other categorizations within the data environment into which the node can be transformed by aggregation.

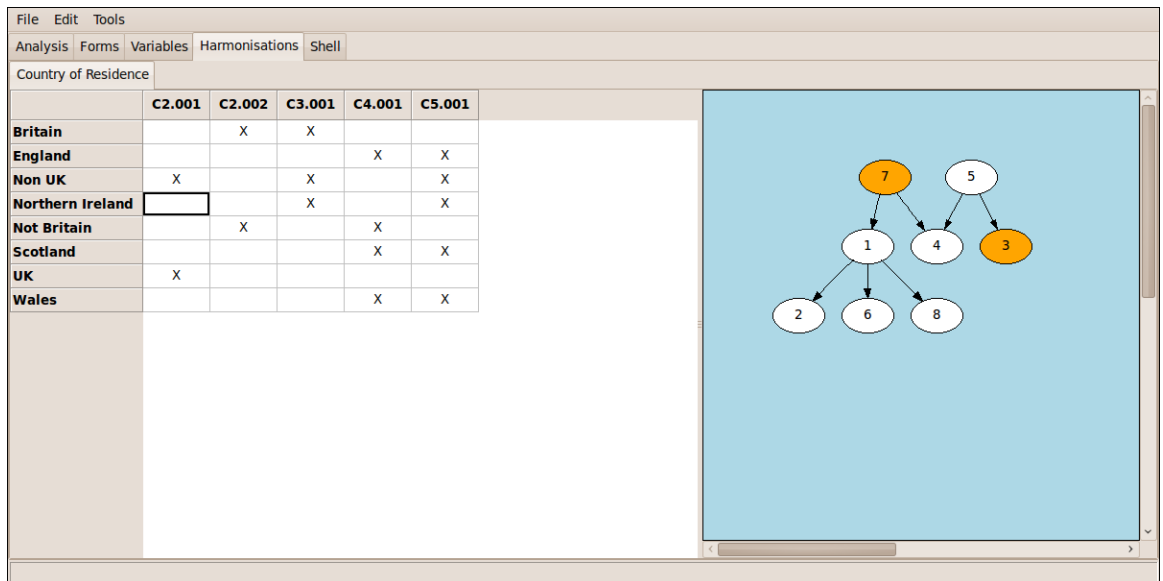


Figure 78. *The Harmonizations notebook page for Country of Residence*

When a variable is added to the data environment a new tab is also added to the notebook under the Harmonizations tab. This also contains a grid and a canvas. The grid provides details of the categorizations corresponding to the codes. The canvas contains the aggregation graph corresponding to the variable. If a column cell is selected the nodes corresponding to the categorization are selected in the aggregation graph. Editing the graph on this canvas makes no difference to the underlying data environment. To do that the user would have to go back into the data entry system.

When a variable is deleted from the data environment (i.e. all forms) the corresponding tabs are removed from the notebooks under the Variables and Harmonizations tabs.

Adding groups

Groups can be added / edited via the *Groups* menu item under the *Edit* menu. The group must be given a unique name. Forms can be selected individually or selected as a group by selecting an existing group. Selecting or deselecting a group selects or deselects all forms

within the group. Thus, for example, the set difference of two groups can be selected by selecting both groups and then deselecting one of them. Groups can be deleted from the data environment by selecting the group(s) and clicking the *Delete* button.

The figure consists of two vertically stacked screenshots of a software dialog box for creating a group. Both screenshots have a light beige background and a thin black border.

Top Screenshot:

- Name:** A text input field at the top left, currently empty.
- Empty Text Area:** A large rectangular text area below the name field, currently empty.
- Delete Button:** A button labeled "Delete" located to the right of the empty text area.
- Survey List:** A list of five items, each with an unchecked checkbox and a label:
 - ☐ Survey V
 - ☐ Survey W
 - ☐ Survey X
 - ☐ Survey Y
 - ☐ Survey Z
- Close and Add Buttons:** Two buttons at the bottom right, labeled "Close" and "Add".

Bottom Screenshot:

- Name:** The text input field now contains the text "Group1".
- Empty Text Area:** The large rectangular text area remains empty.
- Delete Button:** The "Delete" button is still present to the right of the text area.
- Survey List:** The list of surveys is the same, but the checkboxes for "Survey X", "Survey Y", and "Survey Z" are now checked. The "Survey Z" row is highlighted with a dark brown background.
 - ☐ Survey V
 - ☐ Survey W
 - ☒ Survey X
 - ☒ Survey Y
 - ☒ Survey Z
- Close and Add Buttons:** The "Close" and "Add" buttons are at the bottom right. A mouse cursor is pointing at the "Add" button.

Figure 79. *Creating and adding a group*

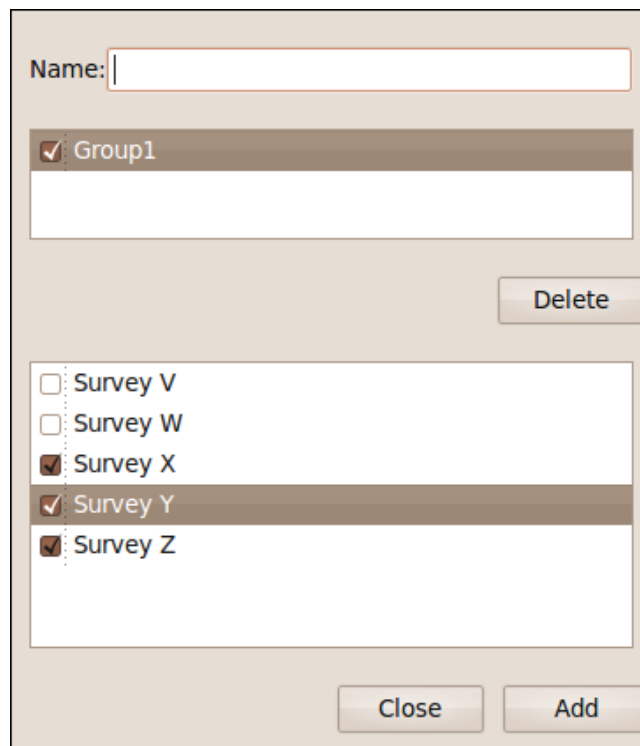


Figure 80. *Selecting all nodes in a group by selecting the group*

```
File Edit Tools
Analysis Forms Variables Harmonisations Shell
Python 2.6.4 (r264:75706, Dec 7 2009, 18:43:55)
[GCC 4.4.1] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import DEASenv as env
>>> env.groups
{'Group1': set([u'Survey Z', u'Survey X', u'Survey Y'])}
>>> env.add_group('On-line', [name for name in env.forms if env.forms[name].metadata['Form Type'] == 'On-line'])
>>> env.groups
{'Group1': set([u'Survey Z', u'Survey X', u'Survey Y']), 'On-line': [u'Survey Z', u'Survey V', u'Survey W']}
>>>
```

```
Namespace Display Calltip History Dispatcher
▸ locals()
locals()
Type: <type 'dict'>
Value: {'BoaApp': <class '._main_.BoaApp'>, 'pp': <bound method Display.setItem of <wx.py.crust.Display; proxy of <Swig Object of type 'wxStyledTextCtrl *' at 0x3492c60>>, 'shell': <wx.py.shell.Shell; proxy of <Swig Object of type 'wxStyledTextCtrl *' at 0x33ef450>>, 'name': u'Survey W', 'builtins': <module '__builtin__' (built-in)>, '__file__': 'DEASApp.py', 'modules': {u'DEASFrame': [1, 'Main frame of Application', u'DEASFrame.py']}, 'DEASFrame': <module 'DEASFrame' from '/home/duncan/DEASFrame.pyc'>, '__package__': None, 'filling': <wx.py.filling.Filling; proxy of <Swig Object of type 'wxSplitterWindow *' at 0x346f270>>, 'notebook': <wx.controls.Notebook; proxy of <Swig Object of type 'wxNotebook *' at 0x33ee970>>, 'env': <module 'DEASenv' from '/home/duncan/DEASenv.pyc'>, '__name__': '__main__', 'wxversion': <module 'wxversion' from '/usr/lib/python2.6/dist-packages/wx-2.6-gtk2-unicode/wxversion.pyc'>, 'main': <function main at 0x7f4981618398>, '__doc__': None, 'wx': <module 'wx' from '/usr/lib64/python2.6/dist-packages/wx-2.8-gtk2-unicode/wx/_init_.pyc'>}
```

Figure 81. *Adding a group via the shell*

A more flexible way to create groups is to use the shell. The data environment can be imported and groups added via Python. The environment contains a global dictionary mapping group names to lists of form names. This dictionary is named 'groups' and can be manipulated directly. There is also a function 'add_group' that takes a name and a list of form names as parameters. Figure 81 shows how to create a new group on the basis of metadata values.

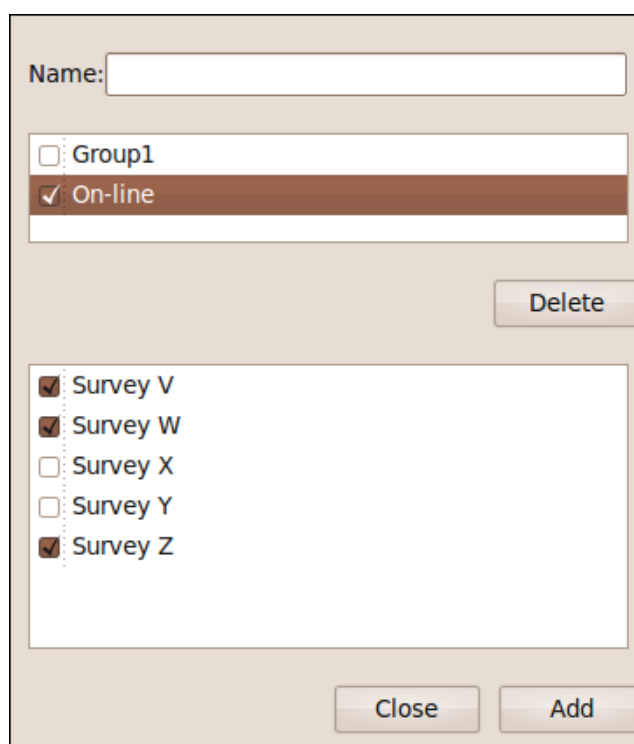


Figure 82. *Confirmation that the new group has been added*

Analysis

The analysis tab contains a panel that allows the selection of a target form, a group / single form and a prevalence. If no group / form is selected then a group comprising all the forms is used. The results are displayed in a text control.

The Strict checkbox allows the consistency constraints on the analysis to be relaxed. Although it is difficult to enter inconsistent categorizations for a variable, it is not impossible. After all, the user has direct access to the data environment via the shell. In normal use the application will check for inconsistent categorizations and print a relevant message in the output, rather than to try to compute a result. If the checkbox is unchecked, then the consistency check is skipped and a result is computed. However, the user should not place too much trust in such outputs.

Variable	Coding
Country of Residence	C4.001

Figure 83. *Analysis output*

The codes can be looked up in the notebook under the Harmonizations tab if they are existing categorizations. If they are not, then they will be of the form HX.Y and can be looked up via the shell, as can any such codes shown in the harmonizations graphs in the notebook under the Variables tab. For instance, the harmonization code in Figure 77 can be looked up as shown in Figure 84 (although the code is obvious as it only contains one category).



Figure 84. *Querying codes via the shell*

Code H1.001 is one category containing all the lowest level categories (leaf nodes in the aggregation graph). In contrast, code C3.001 contains the three categories {Northern Ireland}, {England, Scotland and Wales} and {Non UK}. The function 'get_frozen' takes a variable name and a code as arguments. It is slightly curiously named, as it returns the representation of a categorization as a frozen set of frozen sets.

The queries shown in Figure 84 require a little more knowledge of the data environment (and a little knowledge of Python). The environment contains a 'forms' dictionary mapping form names to form objects. A form object has a 'metadata' attribute which is bound to a dictionary which maps names to values.

Saving outputs

Under the *File* menu there is a *Save As...* menu item. This allows the saving to file of various outputs. If the analysis tab is open it allows the saving of the results to a text file. Under the Variables and Harmonizations tabs it allows the saving of the harmonization graph or aggregation graph of whatever notebook tab is open.

Under the *Tools* menu there is an *Aggregation Editor* menu item. This launches an independent application which is similar to the editor in Figure 75. The difference is that there are fewer restrictions placed on how the graph can be edited. This should only really be used when the user wants to create a suitable aggregation graph for a number of forms in one attempt, rather than incrementally building it up as forms are added. However, it is possible to introduce errors which are prevented by the incremental approach, so it must be used carefully (if at all). Inconsistent data can prevent the KVMS application from opening. In such cases the data files must be edited manually to restore consistency. All files are in a plain text (JSON) format, so this is feasible. But it is best avoided. It should be noted that any files created in the Aggregation Editor should initially be saved to a location other than the Aggregations directory if the KVMS application is open. Otherwise they will be overwritten when the KVMS application is closed if there are aggregation graphs for the same variables in the data environment. The Aggregation Editor is not closed when the KVMS application is closed, and can be launched from the application directory without opening the KVMS application.

APPENDIX 2

KVMS Algorithms

Introduction

Chapter 4 includes two algorithms which will be discussed in more detail here. The first is an algorithm for constructing a harmonization graph from a set of distinct categorizations (of a single variable).

The second is an algorithm for finding a harmonization, $h=H(c, x_1, \dots, x_N)$, of a categorization c and a subset of N other categorizations for a variable within a data environment such that h is maximal with respect to the set of subsets of size N .

Both these algorithms will be discussed in turn, with emphasis on their computational complexity and potential optimizations. Computational complexity will generally be discussed in terms of amortized costs (see Tarjan, 1985). Analyses are relatively informal.

Harmonization graph construction

The basic algorithm presented in Chapter 4 is reproduced below.

```
def harmonization_graph(categorizations):  
    # create an empty harmonization graph  
    G = empty directed graph  
    # create a set to hold categorizations  
    # and harmonizations to be added to G  
    C = empty set  
    for c in categorizations:  
        add c to C  
    while C is not empty:
```

```

# get the next graph node, u, and add it to G
pop u from C
add u to G
# find the parents and children of u
# using the criteria described above
parents = parents of u
children = children of u
# add necessary edges to G
for each parent in parents:
    add the edge (parent, u) to G
for each child in children:
    add the edge (u, child) to G
# remove any edges from nodes in parents
# to nodes in children (i.e. ensure property 1 holds)
for parent in parents:
    for child in children:
        if the edge (parent, child) is in G:
            remove edge (parent, child) from G
# generate any new harmonizations required
# to ensure that property 2 holds for the final graph
for each node v in G:
    h = H(v, u)
    if h != v and h != u:
        # v is neither ancestor nor descendant of u
        if h is not in either G or C:
            add h to C

return G

```

What it does not describe in detail is the algorithm used for finding the parents and children of u in G . As explained in Chapter 4 the parents of u are the leaf nodes in the subgraph of G induced by the nodes that are coarser than u . The children of u are the root nodes in the subgraph of G induced by the nodes that are finer than u . The nodes in these subgraphs can be identified easily using the refinement relationship: $H(v, w) = v$ implies that w is finer than v , and that v is coarser than w . In cases where $H(v, w)$ is not in the set $\{v, w\}$ then v is neither finer nor coarser than w . In the basic algorithm when a node u is added it is the nodes that are neither finer nor coarser than u that give rise to harmonization codes. So calculating $H(v, u)$ for each node v in G will allow the two subgraphs to be identified as well as those nodes that might give rise to new harmonizations.

The remaining issue is the identification of the leaf nodes (parents of u) and root nodes (children of u) of the induced subgraphs described above. Processing the nodes in an appropriate ordering can help with this. A preorder depth first search of a graph visits a node before visiting its children. If searches are performed from each root node in G , then

all nodes in G will be visited.

```
def preorder(G, v):
    visit(v)
    for child in the children of v in G:
        if child has not been visited:
            preorder(child)
```

Slightly expanding the above pseudocode demonstrates how the leaf nodes of the subgraph induced by the nodes coarser than u can be identified. The argument 'ancestors' is a reference to a set, initially empty, that will contain the visited ancestors of u . The argument 'parents' is a reference to a set, initially empty, that will contain the parents of u after the traversal is complete.

```
def preorder(G, u, v, ancestors, parents):
    visit(v)
    h = H(v, u)
    if h == v:
        # v is a candidate parent (will be an ancestor of u)
        add v to ancestors
        is_parent = True
        for child in the children of v in G:
            if child has not been visited:
                preorder(G, u, child, ancestors, parents)
            if c is in ancestors:
                # v is not a parent
                is_parent = False
        if is_parent == True:
            add v to parents
```

If a node v is a candidate parent (coarser than u) it is added to 'ancestors'. Once a child of v has been visited it will be in 'ancestors' if, and only if, it is a candidate parent. If all the children of v have been visited and none of them are members of 'ancestors', then v must be a parent of u . Note that the search proceeds (recursively) until nodes that are not an ancestors are visited. Such nodes act as sentinels, stopping the search.

The preorder algorithm allow the identification of the parents of u . The sentinels that are encountered are either finer than u , or are nodes that might give rise to new harmonizations. An obvious strategy to identify the children of u and potential new harmonizations is to extend the search to the remaining graph nodes. A node that is finer than u is a candidate child of u (a descendant of u after the addition of u to G). Such nodes can be added to a set

of candidate children. Any child of a candidate child must be finer than u and cannot be a child of u . Such nodes can be added to a set of non-children, and after the search is completed the children of u is the set difference of the candidate children and the non-children.

So the parents and the children can be identified via a depth first search. It simply requires different processing of the nodes depending on the result of $H(v, u)$. In fact, if a reference to the set of variables to be added to G is passed to 'preorder' the result of $H(v, u)$ can be added from within the function. Adding the required arguments and the set of visited nodes needed to manage the traversal we end up with the following:

```
def preorder(G, u, v, ancestors,
            parents, non_children,
            candidate_children, visited, C):
    add v to visited
    h = H(v, u)
    if h == v:
        # v is a candidate parent (will be an ancestor of u)
        add v to ancestors
        is_parent = True
        for child in the children of v in G:
            if child is not in visited:
                preorder(G, u, child, ancestors,
                        parents, non_children,
                        candidate_children, visited, C)
            if c is in ancestors:
                # v is not a parent
                is_parent = False
        if is_parent == True:
            add v to parents
    elif h == u:
        # v is a candidate child (will be an descendant of u)
        add v to candidate_children
        for child in the children of v in G:
            add child to non_children
            if child is not in visited:
                preorder(G, u, child, ancestors,
                        parents, non_children,
                        candidate_children, visited, C)
    else:
        # if h is not in G it must be added later
        if h is not in G:
            add h to C
        for child in the children of v in G:
            if child is not in visited:
                preorder(G, u, child, ancestors,
                        parents, non_children,
                        candidate_children, visited, C)
```

Some unneeded checks for set membership have been removed, to reflect the fact that adding an element that is already in a set has no effect on the set. Now the complete, detailed algorithm for constructing a harmonization graph can be presented.

```
def harmonization_graph(categorizations):
    # create a harmonization graph
    G = empty directed graph
    C = empty set
    for c in categorizations:
        add c to C # eliminating duplicates if present
    while C is not empty:
        # get the next categorization/harmonization, u
        pop u from C
        # set up the sets required for the preorder traversal
        ancestors = empty set
        parents = empty set
        non_children = empty set
        candidate_children = empty set
        visited = empty set
        # perform the preorder traversal
        for v in the root nodes of G:
            preorder(G, u, child, ancestors,
                    parents, non_children,
                    candidate_children, visited, C)
        # get the children of u (using set difference)
        children = candidate_children - non_children
        # add u and the necessary edges to G
        add u to G
        for parent in parents:
            add the edge (parent, u) to G
        for child in children:
            add the edge (u, child) to G
        # remove any edge from parents to children
        for parent in parents:
            for child in children:
                if the edge (parent, child) is in G:
                    remove the edge (parent, child) from G
```

A detailed complexity analysis of the algorithm is far from easy. Initializing C takes time proportional to the number of categorizations. This would dominate the computational cost in the degenerate case that all the categorizations were identical. In other cases the cost would be dominated by the preorder traversals. For most problems the cost of the traversals would be dominated by the generation of harmonizations. The input is a sequence of categorizations and these have many parameters as they are represented as

undirected graphs. The actual cost is likely to be a complex function of these parameters. A worst case analysis would have to identify the worst case inputs, which would be a complex issue in itself. However, an approximate cost can be generated if some simplifying assumptions are made.

Firstly it is assumed that the categorizations are distinct. Thus the cost is dominated by the 'while' loop.

Although there is some housekeeping required after the preorder traversal, the cost depends largely on the number of parents and children generated by the traversal. Adding and removing graph nodes and edges is generally fairly efficient. Thus the traversals will tend to dominate.

The cost of a preorder traversal is $O(|V| + |E|)$ where V and E are the node and edge sets of the graph and $|\cdot|$ denotes set cardinality. Each node is visited and the number of children (which are in essence visited within the 'for' loops) in the graph is equal to the size of the edge set. Here the cost is generally going to be dominated by the processing of the nodes, which is going to be dominated by the generation of harmonizations. Harmonization is a graph sum operation on undirected graphs that contain a node for each possible leaf node in an aggregation graph.

Assume there are k leaf nodes in the relevant aggregation graph. A pairwise connected undirected graph with k nodes has $k(k-1)/2$ edges. But a categorization with only a single level can be represented as a connected graph with only $k-1$ edges. Categorizations with more categories can be represented by graphs with fewer edges. Assuming the most parsimonious representation of a categorization the graph sum operation is going to require the addition of k nodes and at most $2(k-1)$ edges, which is $O(k)$.

Initially the graph will be empty. On each iteration of the 'while' loop the number of nodes in the graph will increase by 1. Thus the total number of nodes processed in the preorder traversals will be $0+1+\dots+(|V|-1)$, where V is the set of nodes in the completed harmonization graph. This equals $|V|(|V|-1)/2$, or $(|V|^2 - |V|)/2$.

Thus we have an approximate average complexity of,

$$O(k|V|^2)$$

where k is the number of root nodes in the relevant aggregation graph and $|V|$ is the number of nodes in the constructed harmonization graph. However, $|V|$ is the size of the solution rather than the size of the problem. It is possible for n (non arbitrary precision) categorizations to produce a distinct harmonization code for each combination of categorizations of size greater than 1, leading to a harmonization graph with 2^n-1 nodes. Thus the complexity in terms of the input size would be approximately,

$$O(k2^{2n}).$$

This is a sub-exponential running time. But as the number of harmonizations generated is typically far fewer than this, the average case complexity is likely to be more quadratic in nature. It is also worth noting that in this 'worst case' scenario the non-leaf nodes have exactly 2 children, meaning that the number of edges is bounded by $2n$. The cost of processing nodes would still dominate that of processing the edges.

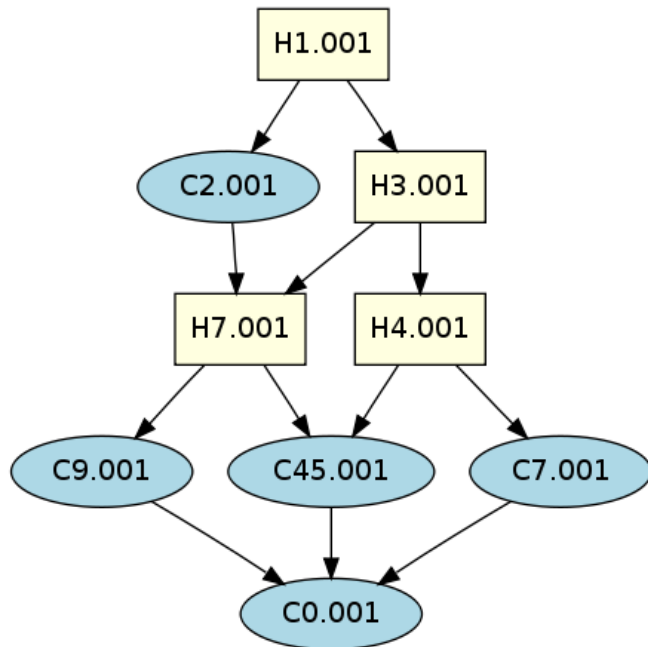


Figure 85. *Harmonization graph for Age*

The example from Chapter 4 is reproduced in Figure 85. The data environment contained almost 200 categorizations for age, but only 5 were distinct. One was an arbitrary precision category which does not give rise to new categorizations when harmonized with other categorizations. The other 4 categorizations only generated 4 harmonization codes. The aggregation graph contained 61 leaf nodes. This is an untypically large figure due to some categories being 1 year age ranges. So the generation of this graph would have required 81 harmonizations, each harmonization involving the graph sum of a pair of undirected graphs with 61 nodes.

It should be noted that a harmonization graph need only be constructed incrementally, adding a new categorization for a variable as it is added to the data environment. When viewed in this context the cost of adding a new categorization, c , to an existing graph G with node set V is approximately,

$$O(k(|V|+|X|/2))$$

where X is the set of new harmonizations generated on the addition of c .

The KVMS generates harmonization graphs using the algorithm described above because it simplifies the software and does not represent a noticeable computational burden. Adding categorizations to existing harmonization graphs remains an alternative. This is exactly what the algorithm above does, but repeatedly, for each categorization within a collection of distinct categorizations.

Perhaps the most questionable assumption above relates to the cost of harmonization. A parsimonious graph representation results in linear time. The worst case is quadratic. This emphasizes the importance of parsimony. It will be worthwhile investigating approaches to maintain a minimum number of edges for generated harmonizations. A minimum number of edges for a category is achieved when the relevant connected component is a tree. In other words, the graph should be a forest. Well-known algorithms such as that due to Kruskal (1956) can produce such a forest from a graph, $G = (V, E)$, in $O(|E|\log_2(|V|))$ time. As the number of edges for the current problem is bounded by $2V$ this is $O(|V|\log_2(|V|))$, or using the notation above, $O(k\log_2(k))$.

Analysis

The output of an analysis (for a given variable) is a categorization or harmonization that is the harmonization of a target node c and N other (not necessarily distinct) categorizations, and that is maximal with respect to the number of categories it contains. An algorithm for this is described in Chapter 4. Here it will be provided in the form of pseudocode and its computational complexity discussed.

The first step of the algorithm is to scan the data environment to count the number of times each categorization appears. The time for this is clearly linear in the number of datasets. Once this is done we can associate a count with each observed categorization. (The observed categorizations in Figure 85 are shown as blue ovals.)

In the previous section the algorithm for generating a harmonization graph used preorder depth first traversals of the graph. The algorithm in this section uses a postorder depth first traversal of the graph. The difference is quite simple. A preorder traversal visits a node before visiting its children. A postorder traversal visits a node's children before visiting the node. As before, visiting a node involves some processing of information as well as marking the node as visited.

```
def postorder(G, v):
    for child in the children of v in G:
        if child has not been visited:
            postorder(child)
    visit(v)
```

The postorder graph traversal propagates the descendant sets to each node in the harmonization graph. (They are actually stored in a mapping of nodes to descendant sets.) Finding the descendants of a node, v , is simply a case of generating the union of v 's children and its children's descendant sets. Once the descendants are found it is simple to sum the associated counts for v 's descendants and v itself to produce a count of the maximal number of (not necessarily distinct) observed categorizations that harmonize to v . If this is at least N , and $H(v, c) = v$, then v is a candidate solution. A reference is kept to the

visited candidate that is maximal with respect to the number of categories it contains. After traversing the graph the referenced candidate is the solution.

Note that the pseudocode is based on Python, which does not have variables in the same sense as languages such as Fortran. Nevertheless the code has been written as if it does have variables. (Python programmers will know how to adjust the code accordingly, and those who do not know Python need not be distracted by this note.)

```
def postorder(G, c, v, visited, counts, descendants, N, res):
    for child in the children of v in G:
        if child is not in visited:
            postorder(G, c, child, visited,
                      counts, descendants, N, res)
    add v to visited
    # process v
    # get the descendant set and store in descendants
    descendants[v] = empty set
    for child in the children of v in G:
        add child to descendants
        # add the descendants of child
        descendants[v] = descendants[v] | descendants[child]
    # check if the solution is candidate
    # check the  $H(v, c) == v$  criterion first
    if  $H(v, c) == v$ :
        # get the total number of datasets
        num_datasets = counts[v]
        for d in descendants[v]:
            num_datasets = num_datasets + counts[d]
        # check the 'at least N' criterion
        if num_datasets >= N:
            # it is a candidate solution
            if res is None:
                res = v
            elif v has more categories than res:
                # it is the best solution found so far
                res = v

def maximal_harmonization(G, c, counts, N):
    visited = empty set
    counts = empty mapping
    descendants = empty set
    res = None # the default solution if no candidates are found
    for v in the root nodes of G:
        postorder(G, c, v, visited, counts, descendants, N, res)
    return res
```

It is clear that each node is visited once. On visiting a node a harmonization is performed and a descendant set is generated. In some cases the number of datasets is generated and this requires a number of lookups that is proportional to the size of relevant descendant set. The average size of the descendant set is bounded by $|V|/2$ where V is the set of nodes in the harmonization graph.

Assume there are k leaf nodes in the relevant aggregation graph. Following the same argument as in the previous section the approximate cost of harmonization is k . Thus we have an approximate average complexity of,

$$O(k|V|^2).$$

BIBLIOGRAPHY

- Agrawal, R. and Srikant, R. (1994) "Fast algorithms for mining association rules." Proc. 20th Int. Conf. on Very Large Databases (VLDB 1994, Santiago de Chile), pp.487-499 Morgan Kaufmann, San Mateo, CA, USA
- Benedetti, R. and Franconi, L. (1998) "Statistical and technical solutions for controlled data dissemination." In Pre-proc. New Techniques and Technologies for Statistics, Vol. 1, pp.225–232
- Berge, C. (1967) "Some classes of perfect graphs." In Graph Theory and Theoretical Physics (F. Harary, Ed.), pp.155-166, Academic Press, New York
- Bertsimas and Tsitsiklis (1993) Simulated annealing. Statistical Science, Vol. 8 No. 1, pp.10-15
- Bethlehem, J. G., Keller, W. J. and Pannekoek, J. (1990) Disclosure control of microdata. J. Am. Statist. Ass., 85, pp.38–45
- Bloemeke, M. and Valtorta, M. (1998). "A hybrid algorithm to compute marginal and joint beliefs in Bayesian networks and its complexity." In Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (G.F. Cooper and M. Serafin, Eds.), pp.16-23, San Francisco, California, Morgan Kaufmann
- Buzzigoli, L. and Giusti, A. (1999). "An algorithm to calculate the lower and upper bounds of the elements of an array given its marginals." In Statistical Data Protection (SDP'98) Proceedings, pp.131–147. Eurostat, Luxembourg
- Cleveland, W.S. (1979). Robust locally weighted regression and smoothing scatterplots. Journal of the American Statistical Association 74 (368): pp.829–836
- Copas, J. (1999) The effectiveness of risk scores: the logit rank plot. Appl. Statist. 48 Part 2 pp.165-183
- Cormen, T.H., Leiserson, C.E., Rivest, R.L. and Stein, C. (2001) Introduction to Algorithms, Second Edition. MIT Press and McGraw-Hill
- Cowell, R.G., Dawid, P., Lauritzen, S.L. and Spiegelhalter, D.J. (1999) Probabilistic Networks and Expert Systems. Springer-Verlag
- Dawid A.P. and Lauritzen S.L. (1993) Hyper Markov laws in the statistical analysis of decomposable graphical models. The Annals of Statistics, 21, pp.1272–317

- Diaconis, P. and Sturmfels, B. (1998). Algebraic algorithms for sampling from conditional distributions. *The Annals of Statistics*, 26, pp.363–397
- Dirac, G.A. (1961) On rigid circuit graphs, *Abh. Math. Sere. Univ. Hamburg*, 25, pp.71-76
- Dobra, A. (2002) Statistical Tools for Disclosure Limitation in Multi-Way Contingency Tables. Ph.D. Thesis, Department of Statistics, Carnegie Mellon University
- Dobra, A. (2003) Markov bases for decomposable graphical models. *Bernoulli*, Volume 9, Number 6, pp.1093-1108
- Dobra, A. and Fienberg, S. E. (2001). Bounds for cell entries in contingency tables induced by fixed marginal totals with applications to disclosure limitation . *Statistical Journal of the United Nations ECE* 18, pp.363–371 IOS Press
- Dobra, A. and Fienberg , S.E. (2008) The generalized shuttle algorithm. Working Paper no. 83, Center for Statistics and the Social Sciences , University of Washington
- Domingo-Ferrer, J. and Torra, V. (2008) A critique of *k*-anonymity and some of its enhancements. In *Proceedings of ARES/PSAI*, Los Alamitos, CA:IEEE Computer Society, pp.990-993
- Duncan, G.T., Elliot, M.J., Salazar, J.J. (2011) *Statistical Confidentiality*. New York, Springer
- Duncan, G.T., Fienberg, S.E., Krishnan, R., Padman, R. and Roehrig, S.F. (2001) “Disclosure limitation and information loss for tabular data.” In *Confidentiality and Data Access: Theory and Practical Application for Statistical Agencies* (Doyle, P., Lane, J.I., Theeuwes, J.M. and Zayatz, L.M., Eds). pp.135-183
- Duncan, G.T., Stokes, S.L. (2004) Disclosure risk vs. data utility: the R-U confidentiality map as applied to topcoding. *Chance* 17(3), pp.16–20
- Elamir, E. and Skinner, C.J. (2006) Record level measures of disclosure risk for survey microdata. *J. Official Stat.* 22, pp.525–539
- Elliot, M.J. (2000) DIS: A new approach to the measurement of statistical disclosure risk. *International Journal of Risk Management*, 2(4) pp.39-48
- Elliot, M.J. (2003) “Using DIS to Calibrate the Output of SUDA.” In *Proceedings of UNECE international conference on statistical confidentiality*
- Elliot, M.J. and Dale, A. (1999) Scenarios of attack: the data intruder’s perspective on statistical disclosure risk . *Netherlands Official Stat.* 14, pp.6–10
- Elliot, M., Lomax, S., Mackey, E. and Purdam, K. (2010) “Data environment analysis and the key variable mapping system.” In *PSD 2010, LNCS 6344*, (J. Domingo-Ferrer and E. Magkos, Eds.), pp.138–147

- Elliot, M.J., Manning, A. and Ford, R. (2002) A computational algorithm for handling the special uniques problem. *International Journal of Uncertainty, Fuzziness and Knowledge Based Systems* 5(10) pp.493-509
- Elliot, M., Purdam, K. and Smith, D. (2008) Statistical disclosure control architectures for patient records in biomedical information systems. *Journal of Biomedical Informatics*, 41, pp.58–64
- Elliot, M.J. and Skinner, C. (2002) A measure of disclosure risk for microdata. *Journal of the Royal Statistical Society Series B* 64(4) pp.855-867
- Elliot, M., Skinner, C. and Dale, A. (1998) Special uniques, random uniques and sticky populations: some counterintuitive effects of geographical detail on disclosure risk. *Research in Official Statistics* 2, pp.53-67
- Fawcett, T. (2006) An introduction to ROC analysis. *Pattern Recognition Letters*, 27, pp.861–874
- Fellegi, I.P. (1972) On the question of statistical confidentiality, *Journal of the American Statistical Association*, Vol. 67, No. 337, pp.7-18
- Fellegi, I. and Sunter, A. (1969). A theory for record linkage. *Journal of the American Statistical Association* 64 (328) pp.1183–1210
- Fienberg, S.E. and Makov, U.E. (1998) Confidentiality, uniqueness and disclosure limitation for categorical data. *J. Off. Statist.*, 14, pp.385–397
- Fischetti, M. and Salazar, J.J. (1999) Models and algorithms for the 2-dimensional cell suppression problem in statistical disclosure control. *Math. Program.* 84, pp.283–312
- Fischetti, M. and Salazar, J.J. (2003) Partial cell suppression: a new methodology for statistical disclosure control. *Stat. Comput.* 13, pp.13–21
- Forster, J.J. and Gill, R.C. (2008) “Bayesian assessment of rounding-based disclosure control.” In *Privacy in Statistical Databases, UNESCO Chair in Data Privacy International Conference, PSD 2008*, pp.50-63, Berlin, Germany, Springer
- Forster, J.J., McDonald, J.W., and Smith, P.W.F. (1996) Monte Carlo exact conditional tests for log-linear and logistic models. *Journal of the Royal Statistical Society*, 58, pp.445–453
- Forster, J.J. and Webb, E.L. (2007) Bayesian disclosure risk assessment: predicting small frequencies in contingency tables. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 56, (5), pp.551-570
- Franconi, L. and Polettini, S. (2004) “Individual risk estimation in μ -ARGUS: a review.” In *Privacy in Statistical Databases, Springer Lecture Notes in Computer Science 3050*, (J. Domingo-Ferrer and V. Torra, Eds.), pp.262– 272, Berlin
- Frydenberg, M. and Lauritzen, S.L. (1989) Decomposition of maximum likelihood in mixed graphical interaction models. *Biometrika*, 76, 3, pp.539-55

- Gelman, A. and Rubin, D.B. (1992) Inferences from iterative simulation using multiple sequences (with discussion). *Statistical Science* 7 pp.457–511
- Giudici P, Green PJ (1999) Decomposable graphical Gaussian model determination. *Biometrika*, 86, pp.785–801
- Green, P.J (1995) Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika* 82 (4) pp.711-732
- Grzegorzcyk, M. and Husmeier, D. (2008) Improving the structure MCMC sampler for Bayesian networks by introducing a new edge reversal move . *Machine Learning*, 71, pp.265–305
- Hastings, W.K. (1970) Monte Carlo sampling methods using Markov chains and their applications. *Biometrika* 57 (1) pp.97-109
- Heckerman, D., Geiger, D. and Chickering, D.M. (1995) Learning Bayesian networks: the combination of knowledge and statistical data . *Machine Learning*, 20, pp.197-243
- Hoeting, J.A., Madigan, D., Raftery, A.E. and Volinsky, C.T. (1999) Bayesian model averaging: a tutorial. *Statistical Science*, 14(4), pp.382-417
- Jensen, F.V. (1996) *An Introduction to Bayesian Networks*. Springer-Verlag. New York
- Jensen, J.L.W.V. (1906) Sur les fonctions convexes et les inégalités entre les valeurs moyennes. *Acta Mathematica*, 30(1), pp.175-193
- Karr, A.F., Dobra, A., Sanil, A.P. and Fienberg, S.E. (2002) Software systems for tabular data releases. *International Journal on Uncertainty Fuzziness and Knowledge-based Systems*, 10(5), pp.529-544
- Knuth, D. (1992) Two notes on notation. *American Mathematical Monthly*, Volume 99, Number 5, pp. 403-422
- Kruskal, J.B. (1956) On the shortest spanning subtree of a graph and the traveling salesman problem. In *Proceedings of the American Mathematical Society*, Vol. 7, No. 1, pp.48-50
- Lauritzen, S.L. and Spiegelhalter, D.J. (1988) Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society B*, 50, No. 2., pp.157-224
- Leimer, H.G. (1993) Optimal decomposition by clique separators, *Discrete Mathematics* 113 pp.99–123
- Machanavajjhala, A., Gehrke, J., Kiefer, D. and Venkitasubramanian, M. (2006) L-diversity: privacy beyond k-anonymity. In *Proceedings of the IEEE ICDE 2006*.
- Madigan, D. and Raftery, A.E. (1994) Model selection and accounting for model

uncertainty in graphical models using Occam's window. *J. Am. Statist. Ass.*, 89, pp.1535–1546

Madigan, D. and York, J. (1995) Bayesian graphical models for discrete data. *International Statistical Review*, 63 pp.215–232

Manning, A.M., Haglin, D.J. and Keane, J.A. (2008) A recursive search algorithm for statistical disclosure assessment. *Data Mining and Knowledge Discovery*, Vol. 16, No. 2, pp.165-196

Mokken, R.J., Kooiman, P., Pannekoek, J. and Willenborg, L.C.R.J. (1992) Disclosure risks for microdata. *Statistica Neerlandica*, Vol. 46. pp.49-67

Office for National Statistics (1993) 1991 Great Britain Sample of Anonymised Records, Individual File [computer file] distributed by the Cathie Marsh Centre for Census and Survey Research, University of Manchester

Ohtsuki, T., Cheung, L.K. and Fujisawa, T. (1976) Minimal triangulation of a graph and optimal pivoting ordering in a sparse matrix. *J. Math. Anal. Appl.*, 54, pp. 622–633

Paass, G. (1988) Disclosure risk and disclosure avoidance for microdata. *Journal of Business and Economic Statistics*, 6(4), pp.487-500

Pearl, J. (1988) *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Representation and Reasoning Series (2nd printing ed.). San Francisco, California, Morgan Kaufmann

Reiter, J.P. (2005) Releasing multiply imputed, synthetic public use microdata: an illustration and empirical study. *J. R. Statist. Soc. A*, 168, Part 1, pp.185–205

Rose, D.J. (1970) Triangulated graphs and the elimination process. *J. Math. Anal. Appl.*, 32, pp.597-609

Rose, D.J. (1973) "A graph-theoretic study of the numerical solution of sparse positive definite systems of linear equations." In *Graph Theory and Computing* (Ronald C. Read, Ed.), pp.183-217, Academic Press, New York and London

Rose, D.J., Tarjan, R.E. and Lueker, G.S. (1976). Algorithmic aspects of vertex elimination on graphs. *SIAM Journal of Computing*, 5, pp.266-283

Salazar, J.J. (2008) Statistical confidentiality: optimization techniques to protect tables. *Comput. Oper. Res.* 35, pp.1638–1651

Samarati, P. and Sweeney, L. (1998) Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. Technical report, SRI International.

Skinner, C.J. (1992) On identification disclosure and prediction disclosure for microdata. *Statistica Neerlandica*, Vol. 46, No. 1, pp.21-32

- Skinner, C.J. and Elliot, M.J. (2002) A measure of disclosure risk for microdata. *J. R. Statist. Soc. B*, 64, Part 4, pp.855–867
- Skinner, C.J. and Holmes, D.J. (1998) Estimating the re-identification risk per record in microdata. *Journal of Official Statistics*, 14, pp.361-372
- Sleator, D.D., Tarjan, R.E. and Thurston W.P. (1988) Rotation distance, triangulations, and hyperbolic geometry . *Journal of the American Mathematical Society*, Vol. 1, No. 3, pp.647-81
- Smith, D. (2001) The efficient propagation of arbitrary subsets of beliefs in discrete-valued Bayesian belief networks. Eighth International Workshop on Artificial Intelligence and Statistics, Key West
- Smith, D. (2006) An evaluation of strategies for matching population and sample units. International Conference of the Royal Statistical Society, Queen's University, Belfast
- Smith, D. and Elliot, M.J. (2003) An Investigation of the Disclosure Risk Associated with the Proposed Neighbourhood Statistics. Report for the Office of National Statistics
- Smith, D. and Elliot, M J. (2005). An experiment in naive Bayesian record linkage. 55th Session of the International Statistical Institute, Sydney, Australia
- Smith, D. and Elliot, M J. (2008) A measure of disclosure risk for tables of counts. *Transactions in Data Privacy*. 1(1), pp.34-52
- Tarantola, C. (2004) MCMC model determination for discrete graphical models . *Statistical Modelling*, 4, pp.1–23
- Tarjan, R.E. (1985) Amortized computational complexity. *SIAM Journal on Algebraic and Discrete Methods*, Vol. 6, No. 2, pp.306-318
- Tarjan, R.E. and Yannakakis, M. (1984) Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs. *SIAM J. Comput.*, 13, pp.566-579
- Willenborg, L.C.R. and de Waal, T. (2001) Elements of Statistical Disclosure Control. *Lecture Notes in Statistics*, Vol. 155, Springer, New York
- U.S Department of Commerce (1978) Report on Statistical Disclosure and Disclosure Avoidance Techniques. Statistical Policy Working Paper 2, Washington
- The 1991 SARs are provided through the Census Microdata Unit, at the Cathie Marsh Centre for Census and Survey Research (University of Manchester), with the support of the ESRC/JISC/DENI. All tables containing Census data, and the results of analysis, are reproduced with the permission of the Controller of Her Majesty's Stationery Office (Crown Copyright).