Stichting Onderzoek Wereldvoedselvoorziening van de Vrije Universiteit

Centre for World Food Studies

Classification by crossing and polling for integrated processing of maps and surveys

An addendum to GRCP-software version 2

by

Michiel Keyzer and Saket Pande

Staff Working Paper **WP - 10 - 01** **March 2010**

Contents

Abstract

The paper presents numerical procedures to compute conditional frequencies on large scale maps and surveys comprising mostly qualitative data, much in the way commonly done for ballots but with sufficient generality and computational power to support simultaneous processing of a large number of questionnaire entries, for categorical as well as for real-valued data. The procedures evaluate the conditional distributions, and identify main conditioning variables. Next, once the relevant determinants have been selected including one categorical treatment variable, matching is applied to estimate the treatment effect of this variable, and given this effect to identify the most effective treatment. Specifically, when characterizing the conditional distributions we address the curse of dimensionality inherent in the crossing of a large number of qualitative answers by focusing on highest frequency outcomes and by applying sorting routines from database management in computation. In case real-valued explanatory variables appear jointly with categorical variables, we make use of kernel smoothing, which allows among others for the representation of spatial correlation, under a window size that maximizes the goodness of fit.. The appendix describes the fully GAMS-controlled operation of the software tool, a new component of SOW-VU's GRCP-package for grid level calculations, regression and classification. The tool also includesoptions for spatial interpolation, for projection of survey data on maps, and vice versa, as well as for calculations on recursive sequences of conditioning variables (Markov chains), so as to ease linkage of different surveys, construction of aggregate statistics at district level and navigation from one task to the next.

Version 2 extends version 1 (Keyzer and Pande, 2008) with facilities for matching and calculation of optimal treatment.

1. Introduction

In opinion polls, individuals are asked about their preferred candidate or party as well as about their personal situation as well as their motives and opinions. On the basis of this information, analysts can report on how given voters' characteristics such as age, sex, education and occupation, are distributed among candidates, and discuss changes in these distributions relative to earlier polls. Reporting will be on each characteristic separately or for two or three jointly. Combined these operations are known and will be referred to as polling.

More in-depth studies also indicate how characteristics *jointly* affect preference for a particular candidate or party, using statistical methods such as cluster analysis, factor analysis and logit and probit regression, and support-vector classification so as to identify major determinants. Countless findings were obtained in this way. Yet, it would seem that, between the partial, descriptive approach and the multivariate, regression-type approaches, the option of a descriptive analysis is being skipped that jointly looks at a large number of answers, aiming at comprehensiveness. This motivates the development of the classification tool reported on in the present paper.

Greater comprehensiveness is presumably important in this context, because the respondents answer many questions at the same time. Discussing averages and their distribution on a question-by-question basis, amounts to limiting attention to marginal probability distributions, while neglecting the interdependencies of the joint distribution. Conditioning on a few characteristics will only resolve this problem if the researcher has very good intuition and experience on which events and determinant variables to select, and which functional form to choose.

Proposed approach

Hence, we present an approach to compute conditional frequencies for a large number of variables from a database, typically a household survey or a geographical map. We propose to treat the full survey, or a large subset of it, as a joint empirical frequency distribution, from which conditional frequency distributions can be derived by partitioning the answers by say, S respondents indexed s into a vector of K^y dependent variables and a vector x of K^x independent variables, and the frequencies of y are taken conditional on x.

In a spatial setting, these operations will be referred to as zoning, and deal with the overlays of various geographical maps for distinct characteristics of a common area as a survey, treating map pixels for which observations are available as the, obviously georeferenced, respondents. Hence zoning amounts to interviewing pixels, or equivalently, allowing them to vote for the class they are considered to belong to. Spatial interdependence can be accounted for also, by characterizing each point on the map in terms of the conditions prevailing in the (eight) adjacent pixels as well as in the point itself. Zoning can, therefore, be interpreted as polling, applied to pixels, as opposed to households or individuals.

Clearly, the analysis can go beyond calculation of conditional frequencies in the sample. As the conditional frequencies are naturally interpreted as probability estimates, it becomes possible to compute the most probable characteristics associated to each x-value, which can be interpreted as winner of the election, as well as the runner up and so on. One may also calculate the edge of the winner over the runner up and other competitors, as a measure of stability.

These calculations in turn allow for further selection of subsets of variables, for instance stepwise selection to identify the subset of *y*-characteristics that yields the highest edge for this given combination *x*-characteristics, hence allowing for stepwise identification of the subset of characteristics *y* that discriminates best. Stepwise selection provides a bridge to common regression techniques that are less equipped than polling to deal with large number of characteristics in parallel, particularly for data are of categorical type. For example, ordered logit or probit analysis that can deal with categorical dependent variables on the left hand side but only accommodate very few categorical variables on de right-hand side, and even then without much flexibility as regards functional form specification.

Furthermore, once a conditional frequency distribution has been obtained, polling may be applied for interpolation at households or grid points where an *x*- value but no *y*-value is available as well as for prediction of *y* under changed *x*, computing at each pixel for which *x*-value is available the corresponding *y*-value with the highest probability, its runner up, its edge and so on.

Consequently, in case a conditional frequency distribution has been obtained from a household survey without georeferencing but the same *x*-values are also available in georeferenced form, possibly after downscaling of district data to pixel level, then zoning can be applied to this distribution as well. Hence, polling offers a tool to combine “people and pixels”.

As endogeneity issues in estimation of effects are unavoidable, “matching” for categorical variables is also proposed. Under matching a new data set is created by coupling to every treated individual, possibly synthetic, a non-treated one with a similar probability as that of treatment and a similar characteristics profile. On this, basis the treatment effect can be estimated, and after this, the most effective treatment identified.

Motivation

The application of polling to household analysis and zoning is envisaged to serve as a descriptive tool, to be used ahead of further specification of parametric and semiparametric regression and classification models, also for variable selection.

Furthermore, selection of determinant variables and specification of functional form become difficult when determinant variables are categorical (integer coded) only, as opposed to real-valued, which is the case for most answers in surveys, because no flexible forms are available to start from, which allow for gradual focusing on major components. Indeed, when explanatory factors are categorical and their combination values large in number relative to the number of observations and the number of real valued variables, it is no longer possible to follow the common dummy variable approach that allows for one equation per binary factor and let all coefficients on the real-valued determinants differ freely across equations. The practice in such situations is to treat the binary factors as dummies in specified structural forms, say, on the intercept or on selected real-valued variables, with the inevitable consequence that the functional specification becomes arbitrary, since the range of possible forms soon gets too wide for an exhaustive assessment.

Finally, there is a basic distinction when it comes to using the function in prediction between real-valued and categorical explanatory variables. For real-valued variables out of sample prediction is a natural step, since the functional form is defined on a real domain, and the belief that the

function holds in between points of observation is the starting point of the whole exercise. By contrast, for categorical explanatory variables, a combination of values not appearing in the data amounts to a completely new phenomenon. Bagging and bootstrapping may to some extent enable the modeler to assess whether within the set of observed combinations, estimation on the basis of some can be extrapolated to others but it definitely says nothing about extrapolation to new so far unobserved combinations of factors.

We conclude that there is little scope for identifying the correct functional form in the presence of more than two or three categorical variables and also that it is not meaningful to make predictions at out of sample values for categorical explanatory variables. This further justifies a procedure for stepwise selection as outlined above, since reducing the number of explanatory categorical variables will, unlike for real-valued variables, extend the domain of prediction.

We remark that in case some of the *x*-characteristics are continuous (real-valued) as opposed to categorical, probability mass becomes likelihood and probabilities are recovered only after integration over the continuum. For example, if a subset of *x*-characteristics refers to latitude and longitude, aggregation from pixel to district will yield frequencies at district level. In addition, conditional likelihood densities at pixel-level can be estimated in this case, by definition of proximity measures (e.g. kernels) through which spatial correlation can be accounted for and spatial inference conducted. Kernel smoothing or nearest neighbor interpolation can be used for this purpose, leading to conditional density maps at pixel-level.

One last point deserves attention, which motivates incorporation of matching approaches. It is the issue of endogeneity that can manifest itself for categorical just as for real-valued data. Consider a dependent variable, say, crop yield, a treatment variable, fertilizer application and a vector *z* of environmental conditions, soil quality and rainfall. The basic issue is to estimate correctly, how for given observed factors *z*, variation in *x* impacts on *y*. The main difficulty is due to the fact that unobserved factors ε, to the extent they correlate with variation in *x*, might explain the effect on yield, rather than variation in *x* itself. Hence, the main challenge is to disentangle the effect of the treatment *x* from that of unobserved ε. Clearly, this requires a priori information on the relationship between *x* and ε, which is of course problematic since ε is not observable. There are basically two types of approach available: experimental and non-experimental. Experimental approaches design the variation in *x* so as to break any correlation between *x* and ε, the non-experimental ones take *x* and ε as given but try to account for their relation explicitly, on an a priori basis. Since most survey data are commonly obtained in non-experimental settings, it is important to allow for statistical method that can break the relationship between *x* and ε also in a setting with categorical data. The matching techniques to be considered mimic the randomized sample by creating a sample in which every object has the same fifty percent probability of being treated or not. Finally, once the average treatment effect has been estimated, the next step is to look for the treatment with the highest payoff and compare it with its competitors.

Possible applications

Before proceeding to further description of the tool proper, some examples may clarify possible applications. These examples seek to deal simultaneously with all covariate observations of a questionnaire comprising say, 50-100 different questions/measurements, linked through geo-referencing to several maps and 30,000 observations. Each exercise starts with one, overarching question, such as: Who is my voter? Where do rural populations tend to settle? What is the best

cure to some illness? Indeed, finding a single question as unifying theme often is the key to coherence and success in empirical research. This question provides the theme to justify the proposal, and once the actual statistical investigation has been completed, to present the conclusions. Our tool seeks to help keeping the middle part, with statistical analysis, in tune with this theme, essentially by guiding the modeler from that question to practical numerical operations and empirical assessments.

1."Who is my voter?": What is the frequency distribution of votes (w) over political parties (y), by educational level, sex, marital status, occupation, and voting district (x)?

The calculations establish the dominant social profile of the voter for each party at an election. Total votes *m* (*m* for mass) cast on each party *y* are counted as real-valued numbers by summation over the weights *w* on individual observations (possibly dividing by a constant to obtain a mean). One-man-one vote corresponds to unitary weights. The "by"-defines the conditioning on *x*. Hence, *x*-classes consist of a crossing of the educational level (1=none, 2=primary, 3=secondary, 4=tertiary) , sex (1=male, 2=female), marital status (1=single, 2=married/couple, 3=divorced, widow(er)), occupation (1=unemployed, 2=worker/employee, 3=employer) , voting district (alphabetical order), possibly also the vote at earlier elections. To establish dominance, the number of votes on party *y* by class *x* is calculated, and from there the frequency distribution of the votes expressed by every class *x*, for the top *N* largest parties in each class.

Hence, *y* and *x* are integer coded vectors with character strings associated to their elements that make up the legend of the classification. Hence, the whole counting process boils down to obtaining array elements $m_{x,y}$, where *x* and *y* are integer coded vectors that actually define a long list of subscripts, and ranking the outcomes to establish dominance. This is straightforward in principle, and daily practice in marketing research among others, but there is a curse of dimensionality as a result of which looking up the *m*-value that corresponds to given *x* and *y* becomes far from straightforward as the dimensions of *x* and *y* grow larger. Yet, as the difficulty is not uncommon in, and actually the essence of database management, we can import practices from that field.

Moreover, the key point distinguishing these frequency calculations from regular clustering and regression are that high multidimensional (x,y) vectors (the questionnaire) can be processed, that the decision on conditioning as to what is *x* and what is *y* is perfectly free and interchangeable, and that while the variables are categorical, no measure of distance or proximity and no parametric form with dummy variables are required. This makes it possible to deal directly with the broad thematic question "what are the social and district characteristics of the people voting for me?", albeit, of course, that the number of observations in every *y*-cell corresponding to given *x* will be limited and often zero.
Furthermore, to present the data in a way that can be absorbed by the reader it is important that all outcomes (identity as well as frequency of observed votes and of votes for winning candidate and runner up) can readily be shown on geographic maps. We remark that the example given does not contain any real-valued variable *z* but if the votes can be geo-referenced through GPS, it becomes possible to differentiate the voting behavior socially, through an income or an age variable, or spatially within districts, or, alternatively, to conduct calculations at a higher, say, provincial level, with a continuous as opposed to discrete spatial differentiation of voting behavior across district boundaries within each province. A grid map rather than a district map is then needed to represent the outcomes spatially. However, as such a transition from coded (district) to mixed

coded and real-valued (province and co-ordinates) is seldom possible for qualitative variables, one is generally left with observations that cannot be compared across *x*.

At a more general level, this illustrates the tradeoff between the compartmentalization into discrete classes whose frequencies are readily calculated but often turn out to be zero, against the generalization through simultaneous treatment of observations in wider categories (here the province) with a continuous differentiation inside each. In fact, through the transition to the continuum, probabilities turn into densities that can only be evaluated under some parametrization, which for given number of observations inevitably rests on untestable beliefs. Compartmentalization is more transparent and more easily interpretable but it gives up any possibility of learning from qualitative similarities. Our frequency calculations are intended to help in this process of shedding in a stepwise manner the less essential variables and categories, on empirical rather than on a priori grounds, so as to zoom in on key relationships that may be parametrized subsequently by some other method.

2."What does vegetation predict about the soil quality?": What are the associations of land surface (w) between soil (y) and vegetation (x) characteristics?

As conducting soil tests at many locations may be costly, it would be practical if vegetation characteristics could be used to predict where the soil conditions are most favorable for crop production. Soil maps are data sets describing various soil properties such as texture, chemical composition and permeability, each by a given number of classes. Similarly, vegetations can be characterized on maps by plant variety, density, size and variability, also in qualitative terms and possibly jointly with climatic variables (temperature, wind, rainfall, usually real-valued). Calculation of the conditional frequencies, with surfaces now playing the role of votes, makes it possible to determine what an observation on vegetation can predict about soil type, expressed as a frequency distribution of soil types that could correspond to it.

Many other questions can be envisaged, for example "To which extent does application of soil classification system I match tally with the prediction of system II?". In this assessment it will be natural to allow for real-valued variables such as climate. Such variables typically impact on a neighborhood whose range will be application specific, making it necessary to determine how wide the "windows of prediction" can be, that is how fast information should decay over space. Specifically, this amounts to finding the window size leading to predictions that best fit the observations.

3. "How to live longest?": What are, for given social conditions (x), the lifestyles and health care strategies (y) that prove most conducive to longevity (m)?

We suppose that a longitudinal survey has recorded the age and cause of death of individuals, whose lifestyles and medical treatments were registered during their lifetime. These individuals can be classified by social condition (*x*), with a frequency distribution of longevity *w* under different (*y*) lifestyles (e.g. smoking or not), and health care strategies (e.g. frequencies of check ups).

On this kind of data set, health care researchers and insurance companies study the effectiveness of various cures in fighting given diseases for different groups of patients. Yet, their actual interest is in answering the broader question on longevity, a real valued variable taken to be dependent on qualitative interventions *y*, for given conditions *x*. Hence, rather than choosing the

class with maximal conditional frequency of occurrence we are now interested in finding the class producing maximal life expectancy.

Choosing mean longevity *m* by class as objective criterion (persons × years of life) to be maximized illustrates the use for decision support exercises where finding the most cost-effective strategy is the aim rather than predicting most probable outcomes. Of course, longevity could be weighted further, say, allowing for decay in value as the individual in the survey died at older age. Also, to the extent that expressing this value in monetary terms is deemed relevant and acceptable morally, costs could be subtracted, and the strategy with highest net benefit over total revenue selected.

Module of the GRCP package

The classification tool has been designed as an additional component of the GRCP-software developed at SOW-VU. The GRCP-software is a GAMS-controlled package for integrated data management and mapping, used to conduct Gridding, Regression and Classification exercises on large geographic and household-level data sets, such as GIS-maps and census-data, respectively (Keyzer, 2008; Norkin and Keyzer, 2009). The gridding part comprises arithmetic commands on basic data as well as a set of rule-based algorithms to conduct various types of computations on maps and surveys. A common gridding operation is to conduct constrained scaling, whereby district level data (*Y*) are being distributed over grid cells or census households in proportion with the known distribution of another variable (*x*), say, population, whose value at grid or household level is known, while maintaining specified bounds on the resulting values (*y*). The software also writes a SAS-program for display of maps with legends that is readily used for further display and processing of data.

By contrast, the regression techniques in GRCP conduct the distribution by estimating via support vector (SV-)regression (quadratic programming) algorithms, also known as kernel learning (see Schoelkopf and Smola, 2002). Combining map and district data with survey data, these techniques are used to estimate the relationship between data *x* and *y* in the survey, but with the extension relative to standard SV-regression that that the estimated grid-map or census weighted mean at district level matches the observed district mean via additional constraints in the program. Their intended field of application is similar to that of poverty mapping (Elbers et al., 2003) but the SV-approach in GRCP is different in that poverty mapping usually estimates a (parametric) relationship between *y* and *x* on the survey only, before applying it to the census or grid-map, whereas our technique imposes additional district information at the stage of estimation, which ensures that the eventual prediction will meet the district average. Another difference is that the regression is done with support-vector regression so as to allow for more flexibility and to account for interdependencies such as spatial autocorrelation. The classification facilities in GRCP apply similar SV-procedures to the case of limited dependent *y*-variables where they constitute the more flexible counterpart of standard logit and probit methods.

Despite their enhanced flexibility relative to pure parametric (i.e. not kernel based) estimation, all three approaches require the modeler to impose quite a few a priori restrictions ahead of any data analysis. For gridding, these are the gridding rules and for SV-regression and SV-classification the user has to keep the dimensionality of the *x*-vector limited to fit available quadratic programming software (usually to a number far below the number of observations), which as mentioned earlier forces an a priori selection of explanatory variables and functional forms

Computation

The additional GRCP-module described in this paper defines a two-step computational technique that can be repeated arbitrarily often in a sequence, to process a survey while dropping redundant (non-observed) combinations of codes. The first step applies the crossing procedure common in GIS-packages to the answers given by an individual (or grid cell). In principle, this amounts to entering the *j*-th answer as *j*-th character in a string (word). By using ASCII characters, we can accommodate 256 different answers to each question. Hence, along the sequence of crossing operations various questions may be compressed into a single answer: two binary questions lead to four combinations, and eight to the 256 answers, fully exploiting the capacity of a single byte 8-bit character. In all, we allow for two such questions for *x* and up to ten for *y*, leading to up to strings (y_s , x_s) of up to twelve characters describing the relevant characteristics of respondent *s*.

At the second step, the polling is implemented via a dedicated algorithm that relies on the fact that as never realized combinations can be dropped, the actual number of combinations will never exceed the number of observations. The algorithm invokes a freeware lexicographic ranking algorithm to order the character strings built up from the pairs (y_s , x_s). Once ordered, identical strings (votes) appear in blocks, and in the absence of real-valued variables *z*, the polling operation reduces to counting the number of votes in each, and eventually sorting these numbers in decreasing order. It is in general practical to conduct such pairs of steps in series, since this makes it possible to raise significantly the information processing capacity because each step drops all low frequency combinations, usually an overwhelming majority. Hence, all calculations are simple in principle.

In case real-valued variables *z* appear also, we propose to use an optimization method that relies on kernel smoothing (e.g. Haerdle, 1993), which unlike the SV-methods of kernel learning and logit/probit estimation only involves a simple scanning over averaging operations and is, therefore, more stable in the common situation that there are few but some observations in several classes.

Overview

The paper proceeds as follows. Section 2 describes the frequency estimation and maximum likelihood prediction when all explanatory variables are discrete. Section 3 allows for the mixed categorical-real valued explanatory variables, and shows that the kernel smoothing formulation proposed for this case can be used to represent spatial correlation. Section 4 discusses various aspects of implementation including spatial interpolation and projection of survey data on maps, and vice versa, calculations on Markov chains with a recursive sequence of conditioning variables and matching for categorical variables. The Appendix, referred to as Appendix D for compatibility with earlier GRCP-documentation, describes the software and its use. It refers to the classification by crossing and polling as “zoning” to emphasize its design for application in a geographic setting.

2. Frequency estimation and maximum likelihood prediction with discrete explanatory variables only

Let integer coded vector integer coded vector $g_s = (g_{1s},...,g_{js},...,g_{Js})$ the value of vector y_s, and $c_s = (c_{1s},...,c_{rs},...,c_{Rs})$ denote the value of x_s at *s*, $s = 1,...,S$, while scalar w_s attributes a weight to (measures the mass of) the observation; hence, assuming $w_s = 1$ leads to the counting of the number of observations (votes). Alternatively, non-unit weights can be used to measure, say, the spending by a consumer on a particular item. Thus, the triples $(y_s, x_s; w_s)$ define the empirical joint distribution of (y,x). Computations amount to evaluating conditional averages over this empirical distribution for given combinations of characteristics $q = (g,c)$, with *g* and *c* both vectors of the same dimensions as g_s and c_s, i.e. counting the number of observations whose values (g_s, c_s) coincide with (g,c). The mass of a class is now defined as:

$$m_{gc} = \frac{1}{n_{S_{GC}}} \sum_{s \in S^o_{gc}} w_s \,, \tag{2.1}$$

where $n_{S_{GC}}$ is the number of observations *s*, for which both the weight w_s is available and some some pair (g',c'), i.e. for which information is available on all elements of the triple $(y_s, x_s; w_s)$. Division by $n_{S_{GC}}$ is introduced only to express mass as sample average, so as to keep it bounded with number of observations rising to infinity.

Associated to this, we can also compute the conditional frequency:

$$P_{g|c} = \frac{m_{gc}}{\sum_{g \in G_c} m_{gc}}, \tag{2.2}$$

where G_c is the set of possible combinations of codes *g*, for given *c*. Formally, the polling facility computes a conditional frequency:

$$P_h(x_s) = Prob\{ y_s = q_h \mid x_s \},\ h = 1,...,H \,, \tag{2.3}$$

of the vector of (qualitative) characteristics $y_s = (y_{1s},...,y_{js},...,y_{Js})$ of observation $s \in S$ for given characteristic $x_s = (x_{1s},...,x_{rs},...,x_{Rs})$, adopting the value $q_h = (q_{h1},...,q_{hj},...,q_{hJ})$, $h = 1,...,H$ of discrete class values.

Clearly, the number of possible combinations appearing in the sample can be very large. Rather than computing frequencies for each, the program identifies the *N* classes with highest probability, while amalgamating the remaining classes under the $(N+1)$ category 'other'. It assigns rank $\ell = 1,...,N$ to these classes in decreasing order. Hence, ℓ is such that h_ℓ satisfies:

$$P_{h_\ell}(x_s) \geq P_{h_{\ell+1}}(x_s), \qquad \ell = 1,...,N \,. \tag{2.4}$$

We note that (2.3)-(2.4) can be interpreted as a purely non-parametric regression of *y* on *x*. Clearly, rank $\ell = 1$ predicts the "most likely" type y_s for given x_s. The difference $\left(P_{h_\ell}(x_s) - P_{h_{\ell+1}}(x_s)\right)$ measures the robustness of this estimate, which in turn dominates number three, and so on until the *N*-th estimate. Reporting on the edge of the winner over the runner up and over the *N*-th estimator can be informative, over the runner up because it indicates the extent to which the prediction of a class may change with new observations, and over the N-th estimate because it indicates the extent to which the prediction is possible at all (if the N-th predictor performs almost as well as the winner, prediction is difficult). Hence, the procedure can be used to train an expert system.

In addition, the estimation reports on the number (or mass) of observations of pairs (y_s, x_s), inducing the modeler to strike a balance between precision of prediction, obtained through a more differentiated classification schedule, and the mass available to corroborate individual predictions, which is higher for cruder classifications.

Yet, characterizing the actual statistical properties of the estimator requires assumptions on the data generating process of observations $(y_s, x_s; w_s)$. If these are iid, the conditional probabilities obey the multinomial distribution. For household samples, this could be a plausible assumption that would also make it possible to apply simulation techniques such as cross validation through bagging. However, the spatial context for which the facility is primarily designed suggests accounting for spatial dependencies as well. This aspect is considered in the next section

As discussed at some length in the introduction, because of its capacity to treat multiple dimensional *x* and *y* in one round, the polling approach is well equipped to address the "reverse regression" or thematic question: "What is the best set of explanatory variables to explain the given dependent variable?" and since the computing conditional frequencies is the sole aim, there is no problem of reverse causation arising in this context. Moreover, rather than purely asking for a prediction at any given *x* without much warning when this is purely speculative, polling will attribute zero probability to combinations that were not observed.

Finally, once the estimation has been conducted the frequency tables are readily used for maximum likelihood interpolation and prediction, by looking up the highest frequency are readily used for maximum likelihood interpolation and prediction, by looking up the highest frequency *y*-value associated with given *x*.

3. Allowing for both categorical and real-valued explanatory variables

So far, every respondent *s* received in return for the vote of mass w_s on pair *gc*, a mass equal to the class average m_{gc}. Hence, there was full conservation of mass within each *gc*. Indeed, equation (2.1) is naturally interpreted as a (weighted) voting scheme whereby each observation *r* sends its vote to the "box" of the class it belongs to. In return, it receives the mass distribution m_{gx_r} of *y* associated to its x_r, leading to the probability estimates

$$P_g^r = \frac{m_{gx_r}}{\sum_g m_{gx_r}}, \tag{3.1}$$

of what the code might have been. To introduce the alternative with categorical *x* and real valued *z*, we imagine a situation whereby voters communicate via a radio transmitter. In the setup of (3.1) the vote of every respondent in district *x* is communicated to a district-*x* ballot office, that after the closing of the ballots processes all votes and returns the results in the form of a probability distribution P_g^r, and the number (mass) of votes cast $M^r = \sum_g m_{gx_r}$, as well as the number of voters $n_{S_{gc}}$. At the same time, the information is communicated to a central office that consolidates the results.

Alternatively, one could imagine free communication, whereby every radio-transmitter can capture all messages from the own district and contains a device to process the votes at the level of the individual. The district office can now be dispensed of but the central office would still capture all messages,

Now in democratic elections, unitary weights w_s would refer to people with the right to vote and zero to the others. Clearly, a company vote by shareholders would attribute other weights. More importantly, in our case where individuals receive information on the votes of all others, it would be possible to make the strength of the signal dependent on the proximity of the sender, socially or geographically but expressed as a proximity function of real-valued variables (z_s, z_r).

This makes it possible to account for the fact that close neighbors may matter more for the own frequency distribution, and makes it possible to allow for spatial interdependence by letting votes decay with distance. To represent this proximity effect, kernel functions *k* offer a natural vehicle. For z_r denoting the real-valued "coordinates" of *r* (more generally a real-valued vector), every kernel function $k((y_s, x_s, z_s),(y_r, x_r, z_r))$ has the property that it provides the elements of a matrix $K = [k((y_s, x_s, z_s),(y_r, x_r, z_r))]$, called the Gram-matrix, which is positive semidefinite symmetric. We limit attention to a kernel function that has $k((y_s, x_s, z_s),(y_r, x_r, z_r)) = 0$ whenever $(y_s, x_s) \neq (y_r, x_r)$ and, moreover, is a mollifier density: $k((y, x, z),(y_r, x_r, z_r)) = \kappa_{yx}\psi_{yx}(\tilde{z} - z_r)$, where ψ_{yx} is a density (e.g. the normal density), and κ_{yx} a constant such that $\kappa_{yx}\psi_{yx}(0) = 1$. A kernel function measures the proximity of *r* from *s* and reaches its maximum at $z_r = z_s$. Hence, proximity between z_s and z_r is only positive within the same class: $(y_s, x_s) = (y_r, x_r)$, and the kernel function effectively varies with (z_s, z_r) only but

in a way that might be class-dependent. Practical applications of kernel smoothing and SV-regression often drop the condition that kernel values should be zero when measuring proximity of observations belonging to different classes but these require significant a priori work to quantify such similarities.

With a kernel smoothing specification, the vote reaching *r* when sent from *s* becomes subject to decay and equal to:

$$w_s^r = k((y_s, x_s, z_s),(y_r, x_r, z_r)). \tag{3.2}$$

reflecting the loss of mass. Therefore, total mass distribution is computed for all for all *g* occurring in x_r as:

$$m_g^r = \frac{1}{n_{S^o}} \sum_{s \in S^o_{gx_r}} k((y_s, x_s, z_s),(y_r, x_r, z_r)), \quad r \in S^o, \tag{3.3}$$

where S^o denotes the set all sites for which full (y_s, x_s, z_s) observations are available. We remark that since the kernel was taken to be a density, (3.3) has the form of the well known kernel smoothed estimator (mollifier, see Ermoliev and Norkin, 1997; Keyzer and van Wesenbeeck, 2005) for joint likelihood densities, with observation specific weights. Noteworthy is also that, unlike in logit and probit regression, the dependent variable *y* appears on the right hand side as explanatory variable. Indeed, (3.3) can also be interpreted as a common (but class-specific) kernel smoothing regression with (real-valued) dependent variable 1, whereby equation (3.3) generates m_g^r as a mean that can, since *k* is taken to be a mollifier density, also be written as:

$$m_g^r(z_r) = \frac{\kappa_{gx_r}}{n_{S^o}} \sum_{s \in S^o_{gx_r}} \psi_{gx_r}(z_s - z_r).$$

Therefore, with sample size $n_{S_{gx_r}}$ approaching infinity, one obtains – for an iid sample– the unbiased estimate as the average function:

$$m_g^r(z_r) = \kappa_{gx_r} \int \psi_{gx_r}(\tilde{z} - z_r) d\tilde{z} = E_{gx_r|z_r} w$$

At a practical level, the formulation (3.3) has the advantage over both SV-classification and logit/probit estimation that it frees the modeler of the cumbersome task of having to specify functional forms with dummy variables for each and every possible event. However, function (3.3) is computationally less simple than might seem at first, because the discrete values it could adopt have to be looked up in a table that in principle contains one value for every possible combination of subscripts and may, consequently, become intractably large, hence the need for a tailor made algorithm.

Associated to the mass calculation is the probability (actually the frequency) distribution of *g* occurring at *r*:

$$P_g^r = \frac{m_g^r}{\sum_g m_g^r}. \tag{3.4}$$

Moreover, the maximal likelihood classifier now differs across sites and can be obtained as:

$$g^r = arg\,max_g\, m_g^r, \tag{3.5}$$

with associated mass $m_{g^r}^r$. Similarly, a second-best class can be computed, and so on.

Finally, we mention the Nadaraya-Watson estimator that renormalizes the kernel value to avoid any loss in mass, within each class *gc*, i.e. replaces the denominator $n_{S_{gx_r}}$ in (3.3) by $\sum_{s\in S^o_{gx_r}} k((y_s,x_s,z_s),(y_r,x_r,z_r))$, leading to:

$$m_g^r = \frac{1}{\sum_g \sum_{s\in S^o_{gx_r}} k((y_s,x_s,z_s),(y_r,x_r,z_r))} \sum_{s\in S^o_{gx_r}} k((y_s,x_s,z_s),(y_r,x_r,z_r)), \quad r\in S^o, \tag{3.6}$$

which amounts to replacing the unweighted mean (2.1) by a kernel weighted mean.

Optimal window size

The kernel function may be parameterized further, for instance by variation of the window size θ that could be adjusted so as to maximize the overall mass of, say, the correctly predicted winning classes according to:

$$\theta^* = arg\,max_{\theta\in\Theta} \sum_r \tilde{m}^r_{g^r(\theta)}, \tag{3.7}$$

for correctly predicted mass based on mollifier (3.5) (or, alternatively, the Nadaraya-Watson form (3.6)):

$$\tilde{m}_g^r = \frac{1}{n_{S^o}} \sum_{s\in S^o_{gx_r}|y_s=g} k((y_s,x_s,z_s),(y_r,x_r,z_r)), \quad r\in S^o,$$

and $\Theta = [\underline{\theta},\overline{\theta}]$. In some applications, maximizing the overall mass difference between the correctly predicted winners and the runners-up may be more appropriate.

For window size θ going to infinity, the kernel density converges to uniform shape and calculates the same probabilities as in (2.1)-(2.4). This also is the case if there is no z entering the kernel function or if the *z*-values are constant. Therefore, we may conclude that (3.3) offers a complete generalization of the polling in section 2.

Maximum likelihood interpolation and prediction

Turning attention from estimation at points where full observations (y_s,x_s,z_s) are available to interpolation, where only (x_s,z_s) is given, we remark that in terms of the radio-transmission

metaphor, availability of an observation amounts to sending a signal (the own vote), while obtaining the conditional distribution at (x_r, z_r). Consequently, since the radio does not have to send before it can receive a signal, anyone with the same x_r and with not-too-distant z_r is in a position to obtain relevant information on distribution. Thus, (3.3) readily extends to the case of interpolation, with the mollifier-functions applied to (x_s, z_s)-values within the same census or map where no y_s observation is available, and to extrapolation at values outside the same census or map (here also the Nadaraya-Watson form could be used, and many other estimators):

$$m_g^r = \frac{1}{n_{S_{xzw}}} \sum_{s \in S^o_{gx_r}} k((y_s, x_r, z_s), (y_s, x_r, z_r)), \quad r \in S_{xzw} \subseteq S \tag{3.8}$$

where S_{xzw} is the subset of *S* with the points for which *x*, *z* and *w* are well defined (not missing). This computation can be followed by frequency calculations (3.4)-(3.5), subject, however, to a prediction error that is not reported. Actual interpolation or prediction of the classification itself will generally be so as to maximize the (probability) mass of correct choice:

$$g^r = arg\,max_g\, m_g^r. \tag{3.9}$$

Jointly, vote counting (3.8) and class ranking (3.9) constitute the basic framework for our computations.

For given window size, the statistical properties of the mollifier are well known. For iid sample observations *s* convergence of the mass (3.8) obeys standard laws of the mean. The main issue is, therefore, to arrive at an iid data generating process. Moreover, the properties of the estimator with endogenous window size would need to be studied.

Use in decision support

Calculation of conditional frequencies is a purely descriptive operation. In the context of elections from Example 1, it measures the percentage of voters from category *x* opting for a particular candidate or party. However, for interpolation and prediction we have re-interpreted these frequencies as conditional probabilities, and in (3.8) postulated maximum likelihood as decision rule. The classic justification of this rule is that being right in prediction provides higher benefits than being wrong.

However, there are settings, already mentioned in the health care example of the introduction on use in decision support, that go beyond this simple prediction of official scores. For example, we may consider the prediction problem by an undecided voter who seeks to find out what the own vote should be, as opposed to what others with the same characteristics would vote. Such a person will want to account for the extent to which the others liked the candidate they voted for, and discard, say, any votes made under political pressure. Such aspects can be included through the scalar mass *w*. Calculation of mass for each candidate followed by division through the total mass will obviously still yield a share but this will not be a frequency. Rather, it will become a value share acting as an expected utility weight in the person's decision for whom to vote, learning lessons from the choices made earlier by peers (in the same *x*-class) and supposing that their choice was based on individual preferences that the predictor considers relevant for the own

decision. This makes it possible to avoid any dichotomy between model estimation and not so much what others with the same characteristic have voted, as opposed to the common two-stage approach of estimating a model by maximum likelihood, and using it in policy simulation afterwards, under maximization of say, expected utility.

Similarly, in Example 3 of the introduction, we mentioned the longevity weigh as well as the net profit weight, and special allowance is to be made for the distinct property that profit can be negative. perform an expected profit maximizing choice that is based on peer experience. In a general decision support application, the reporting adjusts the labels of “mass”, “frequency”, into “profit” and “profitshare”, whereas for more biology oriented applications, we also provide an option with “fitness” and “frequency” as labels.

4. Endogeneity, and matching of categorical variables

Endogeneity

Consider a dependent variable, say, crop yield, a treatment variable, fertilizer application and a vector *z* of environmental conditions, soil quality and rainfall. The basic issue is to estimate correctly, how for given observed factors *z*, variation in *x* impacts on *y*. The main difficulty is due to the fact that unobserved factors ε, to the extent they correlate with variation in *x*, might explain the effect on yield, rather than variation in *x* itself. Hence, the main challenge is to disentangle the effect of the treatment *x* from that of unobserved ε. Clearly, this requires a priori information on the relationship between *x* and ε, which is of course problematic since ε is not observable.

There are basically two types of approach available: experimental and non-experimental. Experimental approaches design the variation in *x* so as to break any correlation between *x* and ε, the non-experimental ones take *x* and ε as given but try to account for their relation explicitly, on an a priori basis. In practice, this a priori information is diffuse and hard to test and, consequently, several correction procedures co-exist.

Experimental approaches

Experimental approaches also fall in two categories. The first has *x* varying fast relative to any unobserved change in external circumstances ε. If in every experiment, the treatment *x* is trembling fast relative to the external environment, then this external environment will be reflected in the intercept, and ε will only measure a weak kind of noise that can be taken to be independent of *x* and *q*. This essentially is the most common and naïve way of learning about the effect of one's own actions.

The second category applies when variation in treatment is slow, i.e. has a small number of realizations of x_s, for the same given z_s, then a single intercept cannot capture the effect of confounding unobserved variables. Randomization in this case seeks to break possible correlation between treatment x_s and unobserved conditions ε_s through random assignments. For example, in case of a trial plot on a farm, suppose that unknown conditions apply to the plot where the trial is conducted, not to the seed. Then, the random assignment will be of dosage *x* to plots with known conditions (say location) *q* and unknown conditions ε. If in addition, the seeds possess unknown properties, there will also be a random assignment of seeds to dosage-plot combinations, possibly in two stages: stage 1 sampling the seeds from the storage up to the total quantity required and the stage 2 sampling from this quantity to plots and dosages.

Non-experimental approach

In the non-experimental setting, there is no opportunity for the researcher to vary the dosage under fixed circumstances. There are two main approaches available in this field, instrumental variable regression and matching, with some mixtures (Heckman, 2005b).

Instrumental variable estimation postulates a structural equation where *y* depends on *x* and ε, and possibly also on a vector *q* independent of ε. It supposes that *x* "to a large extent" depends on an

observed vector *z* (relevance) that is independent from ε *(*validity*)*. If *x* depended on *z* with a perfect fit, this would imply its being fully dictated by it, leaving no room for any treatment decision. The IV/2SLS approach consequently expresses *x* in the structural equation as a linear function of *z,* a conditional expectation, as estimated in a first stage regression. While relevance can to some extent be tested, (with the limitation that the fit should neither be too good nor too bad), the independence assumption cannot be. Furthermore, since in a non-laboratory situation the object of treatment cannot be isolated from its environment, which consequently impacts on it directly as well as indirectly, it will often be difficult to exclude *z* from entering the structural equation as well, particularly once non-linear or semiparametric forms are being admitted.

Alternatively, under matching a new data set is created by coupling to every treated individual a, possibly synthetic, non-treated one with a similar probability of treatment and a similar characteristics profile, and estimates the treatment effect on that basis. For categorical data, this is the relevant approach. Since most survey data are commonly obtained in non-experimental settings, it is important to allow for statistical method that can break the relationship between *x* and ε, also for categorical data. The matching techniques to be considered do this by pairing to every quadruple (π,q,x,ε) – with π referring to payoff relative to non-treatment, a scalar, *q* to a vector of categorical circumstances, $x=1$ to binary treatment – one observation $(\pi',q',x',\varepsilon')$ with $x'=0$ referring to non-treatment (a counterfactual), whose ε'-value is arguably very similar to ε, because q' (or some subvector of it) coincides with *q*. Hence, this mimics the randomized sample by creating a sample in which every object has the same fifty percent probability of being treated or not.

We remark that the matching literature often relates to real-valued *q*. It applies propensity score techniques whereby for classes with common characteristics, the probability of being treated is estimated parametrically as a function of *q* or a subset of *q*. Next, within these classes every treated observation is matched with a non-treated one, whose probability of being treated comes closest. Hence the difference in probability serves as distance measure. In the sections below we operate in a similar way but express the distance as its converse, the vicinity, which we estimate in various non-parametric ways.

Once the average treatment effect has been estimated, the next step is to look for the treatment with the highest payoff and compare it with its competitors.

5. Implementation

5.1 Performance criteria

Equation (3.8) applies the maximum probability mass as choice criterion for interpolation and prediction, based on training at the stage of estimation with a data set. This training is equivalent to choosing the classification that maximizes the mass of correctly classified observations relative to the (fixed) number of observations in total, i.e. maximizes the hit ratio. The equivalence holds because, as (3.8) determines the class that has highest mass of "votes", selection of any other candidate than the winner will go against the preference of the largest number of voters, i.e. reduce the hit ratio most. Hence, the hit ratio is the main performance criterion to print when reporting on goodness of fit. This is the hit ratio of the winners.

However, as mentioned in the introduction, to assess the robustness of this decision to follow these winners, it may be useful to learn about other criteria as well, such as the edge of the winner over the runner up. Results are given in total and by *x*-class.

In addition, the user may want to find out how important accurate prediction actually is. For example, if *y* does not vary at all within a class *x*, there is one class only, and the runner up is not even defined. More generally, if there is little variation, a naïve or even biased estimator will come close the winner that maximizes the hit ratio. Hence, we also report on the edge over the winner over the least performing prediction among the *N*-best performing, excluding any predictions referring to classes not present in the data set. In addition, this edge between best and *N*-th estimator can also be used to describe the nature of the conditioning of *y* on *x*. A high value (on the unit interval) implies that the *N*-th estimator performs poorly and suggests that the relation is strong and well established. A moderate edge may mean that there are a few competing classes, which the estimator cannot easily discriminate amongst. A low edge means that the conditioning does not lead to clear answers. Note also that small edge already means that leave-one-out and "leave-few-out" would yield the same maximum likelihood classification.

Comparing the edges for various conditionings is helpful in identifying the more robust relationships, in particular to find out whether *x* predicts *y* better than vice versa, as in the Granger causality tests (but without any claim that causality can be established in this manner). This may, for example, be helpful in the second example of the introduction, where it could be used to measure the extent at which two competing soil classifications are nested, i.e. one can more accurately predict the other than vice versa.

5.2 Stepwise classification

Since the classification tool is developed to assist at an intermediate stage of analysis in focusing on major events and in selecting major determinants, it should allow for a stepwise operation, with a variable composition of the *x* and *y* vectors, starting from a maximal number of characteristics (and hence of characters) and dropping the least discriminating ones at each step. This is implemented as follows.

For *x*, where we admit two elements at most, there would never be more than two possibilities, conducting regression with conditioning on x_1 only and with conditioning on x_2 only. Therefore, stepwise selection can be conducted by separate commands and does not gain from introducing any dedicated procedure. Relative to the regression conditioned on x_1 and x_2 jointly, the single

dimensional *x*-conditioning will yield a better coverage of *y* in that there will be fewer classes without observations, and more observations per class. Hence, the averages will tend to vary less with rising number of observations. Regarding the hit ratios, by construction the number of *y*-classes corresponding to every given *x* in the data will never become lower, making it more difficult to achieve correct prediction. At the same time, number of observations assigned to each *x* is not lower, neither is the mass. Whether the mean highest frequency and hence the hit ratio will become lower, very much depends on the extent to which the explanatory factor was adequate, and may, therefore, serve as a criterion to for selection of *x* characteristics, noting, however, that the edge over the runner up and other choices may have to be accounted for as well.

For *y*, the number of possible combinations may be very large, and some control is useful, to avoid the need to build new crossings from scratch every time. For this, there is a "clearing"-option, whereby the user can indicate from one computational round to the next whether any and if so which characteristics are to be disregarded and possibly replaced by another, while keeping the remaining ones in the string. Merely dropping a *y*-characteristic will, as for *x*, tend to reduce the number of unobserved cases and to raise the number of observations per class. Specifically, dropping *y*-characteristic *j* amounts to taking the marginal over that characteristic, i.e. to integrating it out. For steps $\ell = 1,...,L$ with characteristic j_ℓ dropped at step ℓ, this reads:

$$m^{\ell+1}_{(g_1^\ell,...,g_{j-1}^\ell,g_{j+1}^\ell,...,g_J^\ell)c} = \sum_{g_j^\ell \in G_j^\ell} m^{\ell}_{(g_1^\ell,...,g_{j-1}^\ell,g_j^\ell,g_{j+1}^\ell,...,g_J^\ell)c}, \qquad \ell = 1,...,L \qquad (4.1)$$

where G_j are the class values corresponding to characteristic *j*. The expression also shows that while the change in edges of the winner over others can be of any sign, the mass assigned to the correct class will never be lower.

To verify this monotonicity property, note that as the assessment of correct prediction is less refined (it is unchanged the remaining characteristics but cancelled for j_ℓ), wrong prediction will be less frequent, whereas the total mass associated to the given *x*-value remains the same. Hence, the hit ratio, measured as the fraction of mass that is correctly classified, will never drop. This is the discrete, dependent variable counterpart of the assured improvement in fit in regression when the number of explanatory variables is increased.

Conversely, insertion of an additional characteristic through crossing will never raise the hit ratio or the frequency of the most likely class, confirming that detail in *y*-classification needs to be reduced in a stepwise procedure until the main distinguishing factors remain (the parsimony-counterpart of the dropping of insignificant determinants in regression), or/and to be matched by adequate conditioning on *x*-side (the fit-counterpart of the search in regression for explanatory variables that give good r-square). These choices have to be made by the user, on the basis of the findings at each step.

Yet, despite these reservations an additional control is provided, whereby the program, by trial and error, looks for the class combinations with the highest edge among all possible combinations, for an array *(y,x,z)* with at most eight*(x, y)* elements, while maintaining maximal probability search for given *(x,z)*. Clearly, maximization of the edge is not as good a performance criterion as maximum likelihood but this search might help in finding good combinations.

Summing up, as the proposed approach – starting from a large number of crossings and gradually shedding some of the less discriminating factors until all cases of interest have an acceptable number of observations, and a reasonable hit ratio – is bound to remain a multi-criterion decision process weighing increase in hit ratio against the edge over competitors, the software leaves it to the user to specify the path along which variables are to be dropped, but also provides an admittedly theoretically questionable, automatic single criterion search for the best combination of characteristics.

5.3 Markov chain

Sequence of categorical variables $q \to x \to y$

Discarding the (real-valued) *z*-variables to begin with, we suppose that instead of the data set of triples $(y_s, x_s; w_s)$, we have $(y_s, x_s, q_s; w_s)$. Specific code values of variables *y* and *x* are denoted as before by *g* and *c*, and a given value of *q* is now denoted by *d*. We consider the computations that can be conducted with the commands for triples (here *Pr* refers to frequency calculation i.e. to a probability estimate):

(i) joint conditioning on *c* and *d*: $P_{g|cd} = Pr\{ y = g \mid x = c, q = d \}$;

(ii) single stage structural form for $g \mid c : P_{g|c} = Pr\{ y = g \mid x = c \}$;

(iii) reduced form $P_{g|d} = Pr\{ y = g \mid q = d \}$;

(iv) single stage structural form for $c \mid d : P_{c|d} = Pr\{ x = c \mid q = d \}$.

At this point data availability becomes important. We assume that the data are harmonized in that the subsets of points *s* with missing data are the same for *x*., *q* and *w* and discard any possibility of using different weights in evaluating (ii) and (iii). Then, we can use (iv) to evaluate directly the total effect of *d* on *g* along the Markov chain, according to:

$$P_{g|d} = \sum_c P_{g|c} P_{c|d} \ . \tag{4.2}$$

In terms of our voters' metaphor, every voter *s* of type *d* chooses in the data set one *x*-value as specified by $(y_s, x_s, q_s; w_s)$, and all votes are transmitted from *x* to their eventual *y*-destination *g*. Hence, if data are harmonized, there is no mass lost in evaluation of chains through reduced forms. Consequently, the procedure also applies to a chain of any length longer than two, making it possible to generate probability matrices with elements $P_{a|b}$ for each segment as well as for complete chains.

Yet, goodness of fit measures are needed to choose between the specifications (i), (ii), (iii) and (ii)-(iv) combined that compete in "explaining" the same *y*. Moreover, reverse options that treat some elements of *y* as explanatory may have to be considered as well.

Endogeneity

In this connection, the "endogeneity"-literature in econometrics (e.g. Heckman, 2005a) has devoted much attention to the fact that good performance of a regression of *y* on *x* might be

attributable to a common causative factor *q*, while Granger's causality analysis is on its part concerned with the comparison of *x* on *y* relative to *y* on *x*. Even though our current purely non-parametric setting with categorical variables does not provide any test to decide on such issues, its goodness of fit measures (hit ratios and edges) can be used to compare performances, and help choosing the best form.

At the same time, the frequency calculations have advantages. First, it circumvents a specification bias that occurs in parametric forms, due to correlation between independent variable *x* and the error in regression of *y* on *x* that tends to arise when *x* and *y* largely emanate from a common cause *q*. Instrumentalization amounts to regressing *x* on *q* and using the regressed value of *x* in the relationship with *y*, essentially to make it non-stochastic and hence uncorrelated with the error in regression of *y* on *x*. Since polling has no error in regression, this aspect becomes irrelevant.

Yet, instrumentalization remains relevant, as it may help establishing to which extent *x* can be varied, with *x* possibly referring to changes in policy rather than the policy itself The distribution of *x* at *q* could now be interpreted as indicative of the possibility to vary *x*, expressed as the frequency of various *x*-values observed in association to the now constant factor *q*. Alternatively, a reversed specification could be adopted, whereby the target outcome is treated as *x* and the policy as *y*.

It appears that applying instrumentalization in the conditional frequency framework does not suffer from the limitations common in parametric regression, where it works well only as long as the fit of *x* on *q* is very good, in which case it adds little and has minor effect on coefficients, whereas it becomes misleading when this relationship is poor, because it replaces the information on *x*, the actual variable of interest by some (usually linear) transformation of *q*, of equal dimension as *x*. Hence, 'the cure can be worse than the disease' Bound et al. (1993; see also 1995). The problem occurs because instrumentalization discards underlying information on the conditional distribution of *x*. Hence, it does not arise in (4.2), where the full distribution is transmitted and no information is lost as the eventual mass allocation across *g*-values is unaffected when the full chain of segments is replaced by the reduced form.

Missing data

The discussion has taken the data set to have been harmonized across segments, and assumed a common weight on all variables along the chain. We now consider the case where some elements of $(y_s, x_s, q_s; w_s)$ are missing, which is obviously essential since prediction (including interpolation) are all about filling data gaps. The hierarchy turns out to be from right to left. When either w_s is missing (i.e. has no measurement) or q_s is missing (i.e. the vector has missing data for at least one of its elements), the point has to be dropped both from prediction and from estimation. Predictions of *x* and *y* can be made for the remaining points on the basis of estimation on points where data are available for these. Therefore, a data harmonization command is provided (DATHARM) that drops observations when data are missing in a reference file.

To illustrate the treatment of missing data further and in preparation for the case with the real-valued *z*-variables, let us consider a single segment with weights available at all points, and suppose that *x* is a variable obtained through slicing from real-valued observations into a finite number of brackets. Our computations will now neglect all classes that have no observations on their bracket, and hence, for a fine slicing contain many classes with unobserved *x*, for which no

probability distribution will be available. This in a way addresses the common concern of making predictions on the effect of interventions that were never done before. Similarly, if *q* is the actual intervention, thought to affect *y* via *x*, it may happen that no observations are available on it for some q_s value, while (y_s, x_s) is observed. Then, the data harmonization eliminates all respondents *s* with this q_s, and through it avoid a prediction for the associated *x*. At the same time, this reiterates that slicing limits the opportunity for inference and hence for prediction, and justifies considering the real-valued z-variables themselves.

Sequence of categorical and real-valued variables $(q,u) \rightarrow (x,z) \rightarrow y)$

We now consider a two-stage process with *(y, x, z)* in stage 2 and *(x, q, u)* in stage 2, and after harmonization with respect to missing data, an observation is now represented by $(y_s, x_s, z_s, q_s, u_s; w_s)$. Returning to the radio transmission of the voter's metaphor, we may recall from section 3 that there was for the mollifier formulation (3.5) loss in mass within a class *(g, c)*, while the Nadaraya-Watson calculations avoided this.

However, extending the formulation with real-valued variables raises two issues. First, the eventual frequency distribution of y at point *s* now becomes sensitive to the dropping or keeping of intermediate segments, even with Nadaraya-Watson estimation whereby the total mass in *cg* is maintained. Second, one has to decide whether to treat the variables *z* as exogenous or as predicted from stage 1, in the usual approach of instrumented regression.

On the first score, irrespective of how *z* is arrived at, if the Nadaraya-Watson estimator is used, the sensitivity only occurs at the level of the grid cell or respondent, while at class level the estimation maintains fixed mass of the subset of observations $S^o_{gx_r}$:

$$m_g^r = \frac{1}{\sum_{s \in S^o_{gx_r}} k((y_s, x_r, z_s), (y_r, x_r, z_r))} \sum_{s \in S^o_{gx_r}} k((y_s, x_r, z_s), (y_r, x_r, z_r)), \quad r \in S_{xzw}, \qquad (4.3)$$

and hence a fixed frequency distribution within each class *gc*, implying that differences at the individual level average out within each class. This holds for every segment in the chain as well as for prediction of the probability distribution of y_s for given $(w_s, q_s, u_s, x_s, z_s)$, provided it has been harmonized with respect to missing data. We conclude that after due harmonization of data, the operations described in sections 2 and 3 can be conducted for Markov chains.

Regarding the second point, as Nadaraya-Watson regression generally gives good fit, the estimation will tend to be relatively insensitive to whether first stage regression is conducted or not. Yet, both options are to be provided for.

Finally, we remark that our discussion referred to vectors *x* and *z*. There is no necessity to treat all elements of these vectors in the same way. Clearly, if natural exogenous variables such as time or geographical co-ordinates belong to *z*, there is no point in instrumentalizing them. Specifically, the decision whether or not to instrumentalize some variable amounts to a choice between taking a conditional average or maintaining the empirical distribution.

5.4 Projection of survey data on geographic map, and vice versa

From survey to map

The frequency calculations process the data as they come, with correspondence to geographic entities registered on a geo-referenced file. This file, called locat.grd, is generated by the map-downloading and geo-referencing facility of GRCP (van den Boom and Pande, 2007). It has to be present but the frequency calculations will be conducted also in case the geo-refencing on it is fully uninformative (all entries with same latitude and longitude with a unit administrative code).

Clearly, meaningful geo-referencing is required when maps are to be produced. If the data supplied refer to a *grid map* already, the facility will use the locat.grd file for this purpose. Alternatively, if the files processed are *geo-referenced survey data*, the program needs two geo-referencing files. Locat.grd, is now the file used by the classification program processing the survey; it will in general contain multiple entries with the same co-ordinates, and many coordinates will be absent. The other geo-referenced file, locatm.grd is used for building of geographic maps (and incorporated in the SAS-mapping routines). It has one entry (and one only) for every grid cell, for which it points to co-ordinates and administrative units at three levels, referred to as county CN, province PV, region RE.

Next, to project the survey data on the map we must invoke an interpolation procedure of some kind. We must distinguish between classified data and real-valued data

Regarding classified data, as produced by the zoning or available otherwise, one possibility is to apply mollifier calculations. For this, we consider $\tilde{z}$, a vector with the number of cells on the grid map as dimension and with latitude and longitude co-ordinates as first and second entry. We define a kernel $\tilde{k}\left(\tilde{z}_r,\tilde{z}_{r'}\right)$ measuring proximity of *r* and r', both elements of R, the set of grid cells on the map. Next, we proceed in two stages. First, for S referring to the set of survey data on *y* (both direct and after interpolation), and S_r the subset of *S*:

(i) Compute for each r_o in the subset $R_o = \{ r \in R | S_r \neq \varnothing \}$ of sites with observations from the survey, the mass by *y*-class:

$$M_g^{r_o} = \sum_{s \in S_{r_o}} \sum_c m_{cg}, \qquad r_o \in R_o . \tag{4.3}$$

(ii) Determine for each r_o the class with highest mass:

$$g^{r_o} = arg\, max_g\, M_g^{r_o}, \qquad r_o \in R_o . \tag{4.4}$$

Second, we turn to interpolation so as to obtain values at sites in *R* other than R_o. For this, we apply a straightforward modification of (3.8), (3.9), but now for $\tilde{k}$ without reference to (y,x)-values.

Nearest-neighbor interpolation is the other option provided for projection of classified data. It directly assigns the class of the nearest neighboring cell with an observation available. Here, the

calculations proceed in four steps with (i) and (ii) as above, while the nearest neighbor calculations are:

(iii) For all sites *r* in *R* the nearest neighbor site r_o in R_o is determined, for $M^*_{r_o} = M^{r_o}_{g_{r_o}}$:

$$n_r = arg\, max_{r_o \in R_o} M^*_{r_o} \tilde{k}(\tilde{z}_{r_o}, \tilde{z}_r), \quad if\ max_{r_o \in R_o} M^*_{r_o} \tilde{k}(\tilde{z}_{r_o}, \tilde{z}_r) > 0 \text{ and undefined otherwise} \quad (4.5)$$

(iv) Assign this cell, if defined, to *r:*

$$g^r = g^{n_r}, \qquad r \in R - R_o \qquad (4.6)$$
$$M^*_r = M^*_{n_r}.$$

Hence, whereas the mollifier (3.8)-(3.9) chooses the *class* with the highest average mass "transmitted" to the destination *r*, nearest neighbor (4.5)-(4.6) chooses the class prevailing at the *cell* that transmits the highest mass.

Note that allowance is made for the maximal kernel value remaining zero when observations are too distant. In this case, no assignment is made and the grid cell remains blank. To control for this, the command that defines $\tilde{z}$ has a factor on the window size of the kernel, enabling the user to vary the degree of concentration. In the extreme case, when the factor is put at zero, only the original data will be projected on the map and a dot-plot will be made (the size of the dots can be controlled within the SAS-macro supplied to the user). Hence, the nearest neighbor option is flexible, and reduces to standard nearest neighbor if the weights are unity.

Step (iv) shows that the mapping in this case only generates two files, the classification *g*, and the mass *M*. These files have extension .gcdm/.hdrm and .grdm, respectively. They can be processed further in a separate, grid-oriented GRCP-application, with a directory structure of its own (after dropping the m-character in the extension using the projection from map to survey discussed next). The grid-based application should be kept separate because its vectors are of a dimension different from that of the survey files.

We mention that, as for other GRCP-components, the program actually evaluating the map is a SAS-program automatically produced by the software that has access to all grid files and can easily be modified to produce other displays and to conduct further computations.

Turning to projection of real-valued data, we suppose that in case there is more than one observation at a point, the "household data" should be interpreted as comparable measurements (i.e. the observation vector is taken to be weighted already) and, therefore, take an arithmetic average over observation at each point to obtain the value on the grid. After that spatial interpolation takes place using, either the mollifier (now as Nadaraya-Watson estimator) or nearest neighbor.

Projection from geographic map to survey

The reverse operation of projection from map to survey is available also. It assigns the geographic code or numerical value at a location to all the survey observations, as an additional attribute. This creates files with extension .gcd/.hdr and .grd from those with extension .gcdm/.hdrm and .grdm, to the extent available, while keeping both.

Furthermore, recall that if all data refer to grid cells, the locat.grd and the locatm.grd will coincide. In this case (Datatype = 1) the operation can be used as a "move" device to drop the m-character, copying the .gcdm/.hdrm and .grdm files to .gcd/.hdr and .grd, and removing them afterwards, so as to maintain a fully GAMS-controlled file management.

Projection from survey to census

The regression and classification components of GRCP provide several options for prediction on a census (a full representation of the population) on the basis of a function estimated on a survey (a subset), subject to restrictions expressed a census-level. The zoning component can also be used for this, in two ways. One is to estimate conditional frequency $y|x$ on the survey, and use it for prediction on the census, in another application with a longer x-vector. This has the disadvantage that real-valued variables z cannot be used in the prediction, even though they can be accounted for in estimation. The other way is to work with the (longer) census vectors from the start, assigning "no data" to all observations not in the census, while known weights determine the importance of each. After estimation on the basis of these data, interpolation can fill all data gaps, with x as well as z as conditioning variables.

5.5. Non-parametric matching for categorical treatment

Matching specifies for every observation of an individual or object receiving a particular treatment, the performance of the corresponding non-treated. Through this it seeks to ensure that the sample is balanced in that for every category, every object has an equal probability of being treated or not. In case of discrete variables only, all non-treated objects all possess exactly the same characteristics and the non-treatment could equally well be represented as a weight. With respect to unobserved characteristics the idea would be that intra-class variation cancels out through averaging, whereas in between class variation can be addressed, to the extent known by subdivision of the sample into (sub-)classes. Limitation would be that there is no observable criterion for this subdivision and that a fine subdivision leads to subclasses with too few observations and also that the construct of the matching observation remains subjective, as it necessarily amounts to creating a counterfactual. Yet, as mentioned earlier, this creation is inherent in the inferring a treatment effect in a non-experimental setting.

Whereas the `ZONDIR` command considers the frequency of association of profile y, possibly including treatment characteristics, for given x, both categorical variables, and identifies profiles with the highest frequency, the `MATCH` command singles out the C-th (i.e. last) element of profile y, taken to correspond to the treatment, with a net payoff π, and estimates non-parametrically for each observation where treatment is given, the net payoff of a reference treatment, i.e. of a partner observation for which no treatment is given.

Next, the matching procedure identifies the treatment with largest positive effect, measured as difference between the payoffs, and shows the quantile distribution of this effect, the mean and the standard deviation, and same for worst effect. Clearly, to obtain the quantile distribution of a particular treatment, it suffices to conduct the exercise for this treatment only, by dropping other treatments from the selection (setting `INDSEL(K)=0`).

Programming steps

We consider a data set $\{x_s, y_s, q_s, z_s\}$ where x_s and y_s are vectors with categorical observations jointly describing the class, q_s is a categorical variable of single dimension referring to treatment, and z_s is a real-valued vector that enters the kernel measure. The programming steps are as follows:

1. Specify `KERNSET,` with 4 variables at most to measure vicinity. This specifies the kernel function $k(z_r, z_s)$.

2 Designate the *y*-value of non-treatment by assigning `INDSEL(K)=2`, to the corresponding entry.

3. Calculation: As there is no single-best matching technique, GRCP allows for choice so as to enable the use to assess the dependence of outcomes on the technique used. The estimator either considers all observations in the untreated group. Currently, options available include (i) nearest neighbor, (ii) mollifier and (iii) average.

(i) *Nearest neighbor* selects among the observations for non-treated, the payoff $w_{s'}$ of the nearest observation according to the kernel vicinity measure $k(z_r, z_s)$. Hence, for given observation *s*, it chooses according to kernel function $k(z_r, z_s)$, $r \in S_{x_s, y_s, \kappa_s}$ the nearest observation *r* among those with the same values x_s and y_s as *s*, but with $q_s = \kappa_s$, corresponding to non-treatment. In case no match can be found the observation *s* is being discarded.

$$\tilde{\pi}_s = \pi_{r^*}$$
$$r^* = arg\,min_{r \in S_{x_s, y_s, \kappa_C}} k(z_r, z_s).$$

(ii) *Mollifier on control group* performs a Nadaraya-Watson kernel smoothing estimation for given x_s, y_s and $q_s = \kappa_s$ referring to non-treatment:

$$w_s^r = \frac{k(z_r, z_s)}{\sum_{r' \in S_{x_s, y_s, \kappa_s}} k(z_r, z_s)},$$
$$\tilde{\pi}_s = \sum_{r \in S_{x_s, y_s, \kappa_s}} w_s^r \pi_r .$$

(iii) *Mollifier on all data* performs a Nadaraya-Watson kernel smoothing estimation for given x_s, y_s and all q_s, but it assumes that this treatment variable is discrete valued, and that it also appears as first element in the vector *z*, where it is set at zero treatment for z_r, i.e. $z_{1r} = 0$, which we denote as z_r^0

$$w_s^r = \frac{k(z_r^0, z_s)}{\sum_{r' \in S_{x_s, y_s}} k(z_r^0, z_s)}$$
$$\tilde{\pi}_s = \sum_{r \in S_{x_s, y_s}} w_s^r \pi_r .$$

In this case the size of the window in the kernel becomes important, since for a small window the weights will become the same as for the mollifier on the control group, but far more expensive computationally.

(iv) *Average* calculates the mean payoff:

$$\tilde{\pi}_s = \frac{\sum_{r \in S_{x_s, y_s, \kappa_s}} \pi_r}{\sum_{r \in S_{x_s, y_s, \kappa_s}} 1}.$$

(v) *Zeroref* assumes that $\tilde{\pi}_s = 0$ i.e. that π_s already measures the gain from treatment.

Next, the difference $\delta_s = \pi_s - \tilde{\pi}_s$ is calculated, whenever is defined, and summation for every $c = 1,...,C-1$ yields the class totals for the gains.

Finally, an exponential transformation of payoffs is provided for that takes all payoffs (i.e. not the net gains) to the power ρ, a parameter on the unit interval, with $\rho = 1$ corresponding to risk neutrality, while risk aversion is higher as the exponent gets closer to zero. Yet, optimal choice still is on the basis of the expected gain in this transformed payoff relative to the counterfactual with no treatment, as opposed to expected value of the payoff itself.

5.6 File management

Application. An application is characterized by the constant *S,* the number of respondents in the survey or cells on the map, which fixes the length of the vectors to be operated on. Hence, in case of multiple data sources, multiple surveys are supposed to have been established to a common set of respondents and geographic data sets to have been projected on the same grid map. Respondents may have missing data, represented by a common integer (see Appendix D for details). Each application obeys a standard directory structure, for a specification of which we refer to the GRCP-manual. Here we only mention what is specific to the present tool. Projection of district level data (region, province, county) on the grid or on the survey can be done in the GRIDDAT-facility.

Files extensions. There is now a need to distinguish various files under the same general label. Real valued files have extension .grd associated integer codes have extension .gcd and the headers with legends extension .hdr. Finally, the main result file with frequency distributions and report on goodness of fit comes with the extension .txt.

File locations. All files labels (name without extension) explicitly specified to be kept, are in the subdirectory DAT, except the .gcd and .hdr files, which are in DATC. Other files are in WKRUN.

6. Conclusion

Any regression technique might in principle be used to estimate relationship (3.8). The already available regression and classification components of the GRCP-package contain options for this but many other software tools could, obviously, be invoked as well. Those in GRCP rely on SV-regression, and may, therefore, outperform other procedures, such as spline regression, in estimation of flexible forms, because of the kernel function's capacity to adjust to any data set.

However, in the multiclass context of the present paper, regression will often prove unsuccessful, and lead to large numbers of insignificant coefficients, due to lack of sufficient data points within the same (y_s, x_s)-class. This is where the calculation of conditional frequencies, and in other decision contexts of related "fitness" criteria, can play its role. The suggested mode of operation is to proceed in a stepwise fashion so as to shed uninformative variables and categories at each step, and zoom in gradually on key variables and interdependencies. More advanced classification techniques can be invoked after that. This is in general necessary to arrive at more general results than can obtained through the classwise counting of votes that the polling technique relies on, and which consequently, cannot conduct any statistical inference across classes. Particularly when the number of classes of individual variables is large, the mass will tend to spread too thin over them to lead to significant results.

Regarding further research, the next step in developing the facility might be to account for asymmetries, referred to in the GIS-context as anisotropy, which would in terms of the metaphor imply that the loss of signal strength depends in addition to (kernel) proximity distance also on the direction of "winds" in the fields the message has to cross. This could be represented by extending the mollifier function with some parametric terms. Another possible extension, very natural in the context of polling would be to allow for bootstrapping and repeated sampling. Indeed, the mollifier expression is known to admit an adaptive representation as a moving average that could with appropriate smoothing allow for regular updates of the estimates. Finally, forms that allow for interpolation between classes should be looked into.

Appendix D: Zoning with Matching component of GRCP

D.1 Scope

The software refers to the polling operations as "zoning" to emphasize the GIS-context for which it mainly is intended, and to differentiate it from the already available classification component of GRCP.

A major consideration in design has been to address the high dimensionality inherent in operations on the product of classifications both in computation and in presentation of results. Regarding computation, the evaluation of expressions such as (2.1) and (2.2), and (3.8) is difficult due to the large size of the associated arrays *m* and *P*, of dimension $R + J$. These computational aspects are dealt with by use of sorting routines and dedicated (Fortran) program design and through options for stepwise classification. The presentation aspects are addressed by focusing on higher frequencies that select most important relationships, while information on significance is provided through indicators of how well the runner up and the N-th best perform, and by automated linkage to geographical maps that show the diversity, and by options for attribution of weights and for selection that should enable the user to pay sufficient attention to and focus on rare events. Specifically, the program generates:

(i) Three pairs of gridmaps[1] of *frequencies* of occurrence (extension .grd) with associated *codes* (extension .gcd for code file and .hdr for header file) at given x_s (possibly (x_s, z_s)) for (a) the observed y_s; (b) the most likely class-value g^s; and (c) the second most likely class value. Codes generated on .gcd files and documented on .hdr files are trimmed at every operation, in that their rank replaces their input value.

(ii) A text file (.txt) a GAMS-readable (.gms) file describing masses and conditional probabilities for the top *N* classes for every *x*. This is the main result file. We briefly outline the contents for the example discussed in the next sections, under each of the five headings that follow:

(1) *Overall Classification*
The file POP1 associated to *joint* has the mass after all crossings, the files CLASRAIN and CLASSOIL below it have the masses by constituent part of the crossing.

The 'code =' line that follows has the (newly defined) joint *y*-codes, followed by its overall frequency of occurrence, and the ranking of this frequency.

Next, follow lines with the codes and labels as supplied originally to the crossing; *frq* is the fraction of mass allocated to this entry.

Name of mass file	Top mass	Total mass
joint POP1	5111.8033	5777.6034
1 CLASRAIN	6587.3000	7555.4000
2 CLASSOIL	5821.7000	6585.8000

code = 1 frequency = .0001 rank = 40

[1] Here and in the sequel, grid points can be pixels on a map as well as respondents in a survey.

```
    1   1 SARI      Arid                  frq=   .0001
    2   3 SLSC      Slight                frq=   .0001
code =   2  frequency =   .0075 rank =   20
    1   1 SARI      Arid                  frq=   .0056
    2   4 MOCO      Moderate              frq=   .0054
```

(2) *Classification by x-class*

This is the key table with conditional frequencies that can also be used for prediction on other data sets with the same *x*-classes. It is as (1) but now by *x*-class and with additional information about how many *y*-classes were used in the top *N*, and how many would be required to achieve a given confidence level, i.e. cover a given fraction (here 95%) of the conditional distribution

```
top   5 frequencies     for 95% confidence   8 are needed
```

(3) *Overall fit*

After the tabulation of the conditional frequencies follows a report on the goodness of fit, with hit ratios (% of mass that correctly predicts *y* the class) for the winner, the runner up and the edge of the winner over the *N*-th best:

```
      Mass Hitrt1   Hitrt2  Edge1-N

   5777.60   .40     .20     .33
```

(4) *Frequency of occurrence of class in top N*

This part shows how often each constituent class of basic files CLASRAIN and CLASSOIL appeared in the top *N*.

CLASRAIN

1 SARI	.13	Arid
2 MARI	.48	Semi-Arid
3 SHUM	.32	Slightly Sub-Humid
4 MHUM	.03	Moderately Sub-Humid
5 VHUM	.03	Sub-Humid
6 XHUM	.01	Humid

CLASSOIL

2 NOCO	.18	No constraints
3 SLSC	.21	Slight
4 MOCO	.16	Moderate
5 MNCO	.24	Constrained
6 SECO	.10	Severe
7 VSCO	.08	Very severe
8 XSCO	.01	100% Severe
9 WATE	.03	Water

(5) *Hit ratios by x-class*

As (3) but now by *x*-class

.

The computed frequencies of occurrence $P_{h_\ell}(x_s)$ can be assigned for

(i) *estimation*: computed on a given data set of observed (x_s, y_s);

(ii) *interpolation*:, applied to sites with unobserved y_s for given x_s of the same geographical unit, based the conditional distribution evaluated under (i); in a survey context, this would rather be referred to as prediction.

(iii) *prediction*: applied to given x_s of a different geographical unit, also based on (i).

As a special case, if *x* is an administrative district zoning and *y* a one-dimensional vector with no more than *n* classes, the gridmap of ranks reduces to replication of the input map itself, illustrating the use of zoning as a simple GIS-facility for producing classified maps with legends, a frequency map by district, and tables of probabilities by district and class.

Since it seems practical to keep the present appendix self-contained, there is some repetition of the explanation given earlier in Appendix A for gridding. As for the gridding, the zoning software operates from a GAMS-platform, which is used to generate ASCII input files for Fortran programs that perform the zoning and generate Excel-files and SAS-command files for automatic preparation of geographical maps. GAMS is used as job control language to generate commands in a user-written program. Besides specifying the control options, the GAMS program serves to access data at district (non-grid) level, either directly via INCLUDE statements for files, or indirectly, via the restart option (GAMS [jobname] r=[filename]) from earlier calculations. Actual zoning calculations are done by the Fortran executable ZONDIST that in turn writes SAS-programs and Excel files for plotting of maps. Section 2.4 documents the major steps of such a program.

New users are advised to prepare such a program by using an existing application as template. Allowing for control via a GAMS-program has the advantage over a pure screen driven application that the user is more flexible in preparing the data and that the program offers a transparent documentation of the full application. A DOS-script is used to run the GAMS/Fortran/SAS suite (see ZONDAT.BAT as example).

SASZONE.SAS created in directory ..\WKRUN is an independent SAS-job (invoking application-specific macro's) that is readily modified to perform other preprogrammed SAS-functions, such as plotting of maps. The parameters concerning map plots can be specified via the GAMS job (see Zondat.gams as an example in D.5).

D.2 Zoning rules

The zoning processes a given grid file according to a specified rule. Currently, two rules common in GIS-operations are available: slicing and crossing. Repeated call of these routines builds up a character string for further processing in other routines. The present section introduces these routines, the control of which is implemented via GAMS-macros, as follows.

KERNM: *specify the kernel function for spatial interpolation on map.* This command is only relevant in case the data refer to geo-referenced survey observations with multiple observations at some sites and missing observations. Whenever needed, this command has to be given ahead of all others. The variables $\tilde{z}_i$ entering this kernel necessarily comprise latitude and longitude, at

most two other variables, defined as vectors on the grid map. A radial basis kernel density function is specified (a radial basis kernel is up to a constant the product of normal densities i.e.

$$\tilde{k}(\tilde{z}_r, \tilde{z}_{r'}) = \tilde{\kappa} \prod_i exp - \frac{1}{2} \frac{(\tilde{z}_{ir} - \tilde{z}_{ir'})^2}{(\tilde{\gamma}\theta_o \sigma_i)^2} \qquad r, r' \in R,$$

Factor $\tilde{\gamma}$ on default window size θ_o is entered to modulate the distance over which interpolation can take place. Projections on map are conducted as follows. Based on a survey weight file w_s, and a survey code file g_s associated with the various commands (CROSS, SLICE ZONDIR/ZONMOLL) the projection calculates the most probable code at each survey location r, and associated probability of this code. Next, it estimates these codes on the map by multiplying a site specific weight w_s^m, by the interpolated the probability weighted mass at each grid point, where the probability weight is computed by kernel smoothed maximum likelihood or by nearest neighbor. For ASLICE the procedure is different, as the output is real valued and kernel smoothed averaging is done. The command remains in effect until its next occurrence, where it can be redefined.

CROSS: crossing. The aim of this step is to build a composite code inserting an integer-valued map within range 0-255 as overlay to a vector, by appending the character-value of the integer to a string. The first of the pair contains an integer code within the range 0-255; the second an ASCII code. Specifically, for code *i* on the first map laying within this range, the operation stores an ASCII code corresponding to the character in the *n*-th position in the ASCII-list, and the *n*-th operation stores as *n*-th character from the left. Hence, starting from step 2 the operation concatenates the new ASCII code with the string built up so far, with up to 10 characters. Operations are only conducted on the items selected. Therefore, the procedure can also be used to select some codes for further analysis. The macro CROSS defines this task. If, however, the number of selected codes exceeds 256, program terminates stating fatal error. Thus the user has to ensure that this upper bound is obeyed.

CROSSOLD: crossing using file processed earlier. The macro operates exactly in the same way as CROSS but the user does not have to provide any further information on header labels and codes, since the file .hdr is already available.

SLICE: slicing. Given a real-valued grid map x_s and a list of *J* real-valued class upper bounds z_j ranked in increasing order, slicing creates an integer-valued code y_s, by assigning a code according to:

$$y_s = j \text{ if } x_s \in [z_j, z_{j+1}].$$

This code is subsequently appended to a character vector as for crossing.

ASLICE: anti-slicing. Given an integer-coded grid map j_s with a real-valued number z_j, associated to each code *j*, anti-slicing creates a real-valued grid map y_s, by assigning a real value z_j and multiplying it by a given weight w_s:

$$y_s = z_{j_s} w_s .$$

This makes it possible to perform simple transformations on the weights. The macro defines this task, which generates a file for further processing in SLICE, or in the GRIDDAT program, where more detailed calculations can be conducted, generating real valued grid files that can be processed further in zoning. Also, if projection from survey to map is required, the maps will apply the weights *w* in case the mollifier is used for projection, which amounts to kernel smoothed (Nadaraya-Watson) regression, e.g. to produce poverty maps of incomes or income shortfalls (the regression will apply smoothing everywhere, overwriting observations. In fact, even when no projection is actually needed, copying Locat.grd to Locatm.grd while specifying DATATYPE=2, instructs the ASLICE command to order such regressions on the real-valued maps in the data set, without any classification. This command is independent and does not set any parameters for following commands. Finally, we remark that since ASLICE will produce a vector with a small number of real-valued values only, the associated maps (that seek to maintain equal mass among color-classes) will tend to combine several original variable classes together in one color-class. To visualize an antisliced variable without such loss of resolution the user can print the underlying coded variable, possibly after editing its legend to include the real-valued numbers. As in CROSS, the program will terminate stating fatal error if the number of unique codes in input exceeds 256.

RECODE: recoding. Given a coded grid map x_s, and integer-valued mapping *m*, by assigning a code according to: $y_s = m(x_s)$. The new file is available for further processing.

DATHARM: harmonize missing data and scale. The command is only required in case a Markov chain is considered (see section 4.3). It compares two (pairs of) input files and treats data as missing if they are missing in the .gcd or in the .grd vector of file FLIN (one of both may be unavailable) or in the .gcd or in the .grd vectors of FLREF (one of both may be missing); it writes results on FLIN as FLOUT as .gcd and .grd files, if the original was available, possibly overwriting FLIN), using NODATIN, NODATREF and NODATOUT as the values for missing data, respectively. For FLIN, the valued .grd vector is multiplied by the scalar FACT on output (so as to allow for rescaling of weights). The command can also be used to harmonize the missing data codes on various files. If districts or any other fixed geographic information is to be treated as missing, this has to be implemented by applying the command to the weight file, missing data for all cells in a district will inactivate the district. Other preparatory activities can take place using the GRIDDAT facility. This command is independent and does not set any parameters for following commands.

MAPTOSUR: projection from map to survey. This command performs assigns the geographic code or numerical value at a location to all the survey observations. Since all co-ordinates in the survey must necessarily appear on the map, and no visualization is required, there is no interpolation function here. All files available under the name provided, whatever their extension (.grdm, .gcdm or .hdrm) will be processed; there is no message issued if some of the files are missing.

ZONDIR: estimation of conditional probabilities and interpolation. The large number of combinations of codes that will often result from crossing and slicing makes it necessary to select the most common ones (top *N*); in addition the frequency of occurrence of a code is computed

and can be printed. After a sequence of SLICE and CROSS/CROSSOLD calls, the overlay file is processed via commands specified in the macro ZONDIR. The large number of combinations of codes that will often result makes it necessary to allow for selection of the most common ones; in addition, the frequency of occurrence of a code is computed and can be printed.. Hence processing is with the instructions:

i. Compute the (weighted by w_s) frequency of occurrence of the resulting code within a given x_s (say, an administrative unit and a soil characteristic).

ii. Create a grid file with top $TopN$ frequencies indicated; and print the labels of these "winners"; $TopN+1$ will be the residual class.

iii. Prepare file for creating vectors (maps) of the codes of the $TopN+1$ classes , at sites with observations y_s and x_s (estimation); and possibly also at sites with observations on x_s only (interpolation).

Calculations i.-iii. can be specified for: A: all map, R: region, PV: province, CN: county (as defined on file locat.grd); or <filename> given zone file of different name.

In addition, the zoning computes the average fractions within each layer that makes up the composite generated by the crossing. For this, every CROSS and SLICE command provide a weight v_{js}, and say a crossing by administrative district, of, say pasture land (intensive/extensive) and crop land (cereals/root crops) will yield occurrence four combinations but also the area shares under each. Yet, the probability of occurrence of a combination, say, extensive pastures-root crops, is not necessarily equal to this fraction, because of complementarities and other interdependencies.

Therefore, the facility separately computes the frequency of occurrence of the composite, and for each layer also the fractions making up the composite (here unlike for *w* only the frequency interpretation is admitted, as opposed to, say, profit rates). For land, these would be the fraction of intensive pasture versus the fraction of intensive pasture in total pasture, as well as the fraction of cereals versus the fraction of root crops in total crop land. For this, the user has to specify (not necessarily positive) weights w_s on the composite, jointly with the (non-negative) weights v_{js} on the separate layers of the crossing. By analogy to (2.1), (2.2) the program calculates:

$$\mu_b^j = \sum_{s \in S_{cg}^o | y_{js} = b} v_{js}, \qquad b \in B_{cg}^j, \tag{D.1}$$

where S_{cg} is as before the subset of cells *s* with composite characteristic *cg*, and B_{cg}^j the code associated with the *j*-th element of *y*. Associated to this, we also compute the conditional probability of finding code *b* in position *j* of *y* within composite *cg*:

$$P_{b|cg}^j = \frac{\mu_b^j}{\sum_{b \in B_{cg}^j} \mu_b^j}, \qquad b \in B_{cg}^j. \tag{D.2}$$

These probabilities are reported for information only, and play no role in class choice. Here as well as in ZONMOL, the user can analyze as many as 5000 distinct combinations (far exceeding the limitation on unique codes in CROSS or ASLICE). However, the files

resulting from these procedures should not be used in subsequent CROSS or ASLICE without appropriate selection (to limit the number of distinct codes up to a maximum of 256) as fatal error will occur.

APPLY: *applying the estimated probability map to another region.* The probabilities computed in a previous ZONDIR/ZONMOLL-call, are applied as a mapping to the x_s of another region, which, of course, requires x_s not to comprise any administrative district code, so as to predict the most probable typologies associated to the sites in that region where observations x_s are available. Just like ZONDIR, this command should be preceded by a sequence of CROSS/CROSSOLD and SLICE commands to supply the appropriate legends. The actual code or real values on the data files involved in these steps will not be used.

AGGREG: *computing aggregates.* The probabilities computed in a previous ZONDIR-call, are written as GAMS-readable file for further processing in GRIDDAT. Just like ZONDIR, this command should be preceded by a sequence of CROSS/CROSSOLD and SLICE commands to supply the text-labels. The actual code or real values on the data files involved in these steps will not be used.

In case real valued dependent variables appear in the zoning itself, as opposed to in the projection on the map (KERNM), the kernel has to be specified before the zoning with the mollifier.

KERNSET: *specify the kernel function for zoning.* This command should be executed ahead of ZONMOL. For at most four real valued variables, possibly comprising latitude and longitude, a radial basis kernel density function is specified (a radial basis kernel is up to a constant the product of normal densities i.e.

$$k_h(z_s, z_{s'}) = \kappa_{h_s h_{s'}} \prod_i exp - \frac{1}{2} \frac{(z_{is} - z_{is'})^2}{\sigma_i^2} \qquad s,s' \in S,$$

where *h* refers to the class as defined by (y,x) and $\kappa_{h_s h_{s'}} = 0$ unless$(y_s, x_s) = (y_{s'}, x_{s'})$. In addition, a lower and an upper bound of the window size θ are specified as well as the number of equidistant evaluations to be conducted to solve (3.8) approximately. Once specified the kernel definition remains in effect until a new call. The command also has a parameter to choose between mollifier (3.5) and Nadaraya-Watson (3.6). It also has one parameter specifying whether to write out the estimated observation-specific mass on a file, which makes it possible to use obtain a stage 1 instrumented estimate for the z-value in stage 2 (see section 4.3).

ZONMOL: kernel smoothing with both integer and real x. As ZONDIR but invoking the kernel function and applying calculations of (3.3)-(3.8). This option can be time-consuming if there are many grid cells per class, since all pairs of cells in each class have to be considered. ZONMOL is also to be used for instrumented estimation of *z* (see section 4.3), with the Nadaraya-Watson option activated, the dependent *z*-variable as weight, and the *u*-variables as determinants (see also KERNSET below). Similar to ZONDIR, user can analyze up to a maximum of 5000 distinct combinations. However, results files should be treated with caution when using as input files for subsequent analysis as fatal error may occur.

So far, the discussion was purely in terms of mass and frequencies. As mentioned in section 3, the tool is to serve applications beyond simple prediction. For this, it allows for 4 options (selected via the parameter CONTROLP), with the labels as indicated in the following table:

CONTROLP	Criterion	Ratio
1. Probability	Mass	Frequency
2. Net Profit	Expected Profit	Profit share
3. Viability	Fitness	Fraction

MATCH: Regarding the GAMS-commands, the sequence for matching starts with a `KERNSET` command. Next, the *y*-vector is built up via `CROSS`-commands as with `ZONDIR`. The treatment variable *q* should be the last crossing before the `MATCH`-command. Any crossing beyond it will be neglected. After this the user should select the matching method:

```
SET MATCHMETH
    /
    NEAREST     'nearest neighbor             '
    MOLLCON     'mollifier on control group   '
    MOLLALL     'mollifier on all observations'
    AVERAGE     'average on control group     '
    ZEROREF     'Payoff as gain from treatment'
    /
    ;

SET METHOD(MATCHMETH);
METHOD(MATCHMETH)=NO;
METHOD('NEAREST')=YES;
```

Finally, computations are initiated through the command:

```
$BATINCLUDE ..\LIBRARY\MATCH.gms PAYOFF X1 X2 FLOUT METHOD NODATA

* PAYOFF: a real valued file with payoff observations (input)
* X1 X2: x-variables, as in ZONDIR
* FLOUT: Root of label for output files
* METHOD: as selected
* NRQUANT: number of quantiles for best treatment in print
* RISKAV: risk aversion exponent
* NODATA: symbol for missing data
```

D.3 Grid files

The basic data for analysis whether obtained from grid maps of geographic data, or as survey files. Since most components of the software are applicable irrespective of the datatype, we refer to both files as grid files. In the GAMS job, the scalar DATATYPE specifies this: 1 for spatial grid files, two for survey files. Grid files need a geo-reference that is communicated via up to three files with standard names: (i) locat.grd, (ii) locatm.grd, and (iii) locat_subset.grd, to be available in the DAT folder (they need not be referenced in the REFER set described below). the grid cells (survey respondents) should appear in the same order in all grid files, All georeferencing files relate these entries to latitude-longitude, and three types of administrative

mappings at three levels districts (top), provinces (middle) and counties(bottom)). These appear as the first five columns, A sixth column indicates whether the grid cell is on mainland (=1) or not (=0).

When DATATYPE =1, the software assumes spatial analysis. In this case there is no need for the "locatm.grd" to be present in DAT folder, since the "locat.grd", which should be there, already provides all necessary georeferencing. The third file, locat_subset.grd, can be used to analyze only a subset of a map but wants to provide some background color for the remainder. To represent this software will first expect "locat_subset.grd" that contains a lat lon entry with its administrative mapping for each pixel of analysis for the subset to be studied. If "locat_subset.grd" is not present in the DAT folder, the software then looks for "locat.grd" assuming the analysis grids cover the entire spatial map. In addition, the user should ensure that when a "locat_subset.grd" is created and placed in the DAT folder, a sister "locat_subset.sas7bdat" containing the actual mapping administration (a SAS data file) should also be created and placed in "Makemap\SASdat\". If such a file is not present, maps will be produced assuming analysis grids cover the entire spatial map. Further the user should also ensure that irrespective of whether analysis grids are a subset of the entire map or not the SAS data set "locat.sas7bdat" should always be present in "Makemap\SASdat\". Finally, a label name for background color, when it is requested, should be assigned via the GAMS job (as explained below).

When DATATYPE =2, the software assumes survey analysis. In this case the software expects both "locat.grd" as well as "locatm.grd" to be present in DAT folder. For this data type option, "locat.grd" contains one lat lon entry with corresponding administrative mapping for each survey observation with order following that of respondents of the roster, while "locatm.grd" contains one lat lon entry for each pixel on the spatial map of the study area. If within this application, the user wants to display only a subset of resulting spatial maps with some background color, the software will first expect "locatm_subset.grd" that contains one lat lon entry with its administrative mapping for each pixel of the selected portion of the map. If "locatm_subset.grd" is not present in DAT folder, the software then looks for "locat.grd" assuming display of the results cover the entire map. The user should therefore ensure that when a background is needed a corresponding "locatm_subset.grd" be created and placed in the DAT folder. Further, a sister "locatm_subset.sas7bdat" (a SAS data file used for mapping purposes in Makemap) should also be created and placed in "Makemap\SASdat\". If such a file is not present, maps will be produced assuming that display covers the entire map. The user should also ensure that "locatm.grd" covering the entire spatial map and corresponding SAS data set "locat.sas7bdat" (and not "locatm.sas7bdat") should always be present "DAT" and "Makemap\SASdat\" folders respectively. Finally a label name for background color, when it is requested, should be assigned via the GAMS job (see below).

Next, we turn to the establishment of reference variables stored as grid files (resp. census files), defined in the set REFER, which obviously differs by application. Unlike in gridding, the list of files in the set REFER that appear below does not have to be comprehensive. It includes the files kept permanently in the subdirectories DATC, for integer valued maps and their headers (extensions .gcd and .hdr), and DAT for real-valued maps (extension .grd) as opposed to those in WKRUN (working space). While integer values maps (.gcd files with respective .hdr files) referred to in REFER set can be used by commands such as ASLICE and CROSS, real valued maps (.grd files) are used as input by SLICE.

```
SET REFER  'Reference variables            '
 /
 NOFIL     ' no file                   '
 F1        ' unit file                 '
 SOILS     ' soils                     '
 RAINFALL  ' rainfall                  '
 TEMPERAT  ' temperature               '
 TEMPZON   ' temperature zones         '
 SOILZON   ' soil zones                '
/;
```

The file labels may be up to 10 characters long, excluding the extension, and hence more differentiated than in GRIDDAT, where only the first 4 characters matter; the entries NOFIL and F1 are mandatory and should be in the first two positions of the set REFER.

A RECODE step can be used to transform original code files and their labels into the form required by a particular application.

Actual testing of availability of input files is done at time of execution only and non-availability stops the execution in ZONDIST.EXE.

D.4 The commands:

Program execution is controlled through a sequence of $BATINCLUDE-statements for SLICE and CROSS/CROSSOLD, completed by a single $BATINCLUDE for ZONDIR or ZONMOL followed by APPLY and/or AGGREG; the commands ASLICE, RECODE and DATHARM can be executed separately.

ZONMOL must be preceded by the command KERNSET. If projection on the map is required, KERNM should be entered as first command. Projection from survey to map is a final step after execution of other commands. It is activated by setting the parameter RULEM {0=no projection 1=mollifier 2=nearest neighbor), in case the data are of survey type (DATATYPE=2) and therefore in need of this mapping, since they do not cover the whole map already. Before these commands the rule for projection is to be specified using KERNM. The projection will be conducted for all operations, until a next KERNM command inactivates it. Finally, the MAPTOSUR command specifies the reverse projection. Processing follows the order of the commands. Files are available for subsequent ZONDIR commands, unless they have been overwritten explicitly. To check the sequence, the user can consult the file ZONINP.TXT in subdirectory WKINP. Here follows the list of commands (GAMS-subroutines to control execution of Fortran program) in alphabetical order.

subroutine AGGREG:

1. Name of file (input) to obtain the y|x association from
2. Name of x1-file (input) for which frequency and ranking calculations are to be conducted (A,M,R,PV,C or <filename>), where A = all; M = mainland code 1 (i.e. excluding water); R, PV or CN region, province or county level.
3. Name of second zone file (input) for which frequency and ranking calculations are to be conducted
4. *Fil10:* the filename (at most first ten characters will be used); four files will be generated, with suffixes .hdr (header),.gcd (integer grid file with codes),(real valued gridfile with frequencies).grd and .gms

(GAMS readable file with district aggregates).
5. Header for index set

subroutine APPLY:
1. Name of file (input) to obtain the y|x association from(it will be dealt with as if it was result from direct zoning).
2. Name of x1-file (input) for which frequency and ranking calculations are to be conducted (A,M,R,PV,C or <filename>), where A = all; M = mainland code 1 (i.e. excluding water); R, PV or C region, province or county level.
3. Name of second zone file (input) for which frequency and ranking calculations are to be conducted
4. *Fil10:* the filename (at most first ten characters will be used); four files will be generated, with suffixes .hdr (header),.gcd (integer grid file with codes),(real valued gridfile with frequencies).grd and .gms (GAMS readable file with district aggregates).
5. Header for index set
6. TOPN: computations are done for at most the TOPN-highest (by district) frequencies.
7. NLEV: maximal number of classes in SAS-maps (<9)
8. NODATA: code indicating non-availability of data
9. INTERPOL: 0/1 if interpolation is needed

subroutine ASLICE:
1. Real valued file with weights (w, no sign restriction), to be used for conversion(input file)
2. Integer valued file(input file)
3. Index set of class labels
4. Vector with codes in index set as appearing on the grid file
5. Vector with selection (0/1) from index set
6. Vector with conversion constants
7. Name of file with output values (real) as selected
8. NLEV: maximal number of classes in SAS-maps (<9)
9. NODATA: code indicating non-availability of data

subroutine CROSS:
1. Real valued file with weights (v≥0) to be used for fractions (input file)
2. Integer valued file (input file)
3. Index set of class labels
4. Vector with codes in index set to be assigned to the class
5. Vector with selection from index set
6. Name of output file with selected codes; if NOFIL is entered, no file is written
7. NLEV: maximal number of classes in SAS-maps (<9)
8. NODATA: missing data code on input file to be kept for output
9. CLEAR: release the overlay directly (=2) or
after the next ZONDIR/ZONMOL (=1), not at all (=0), create a file containing original codes with .ocd extension (=4)

subroutine CROSSOLD:
1. Real valued file with weights (v≥0) to be used for fractions (input file)
2. Integer valued file (input file)
3. Name of output file with selected codes; if NOFIL is entered, no file

is written
4. NLEV: maximal number of classes in SAS-maps (<9)
5. NODATA: missing data code on input file to be kept for output
6. CLEAR: release the overlay directly (=2) or
   after the next ZONDIR/ZONMOL (=1), not at all (=0)

subroutine DATHARM:
1. Name of input file(s) excluding the extension
2. NODATA: missing data code on input file
3. Name of reference file(s) excluding the extension
4. NODATA: missing data code for reference file
5. FACT: factor to multiply .grd of input file with
6. Name of adjusted input file(s) (output)
7. NODATA: missing data code on output

subroutine KERNM:
1. Index list input files
2. Rule: mollifier(1), nearest neighbor(2), or no projection (0)
3. Factor on window size.
4. Site specific weight file (input, F1 unit weight)

subroutine KERNSET:
1. Index set with file names.
2. THETALO: Lower bound on window size.
3. THETAUP: Upper bound on window size.
4. NTHETA: Number of evaluations at different window sizes.
5. EDGETH: Edge or absolute maximization for mollifier
6. MOLL: 1= unscaled, 2= Nadaraya Watson
7. FLOUT: save estimated mass on this file (not if NOFIL is specified)

subroutine MAPTOSUR:
1. Index set with input file names (.grdm extension)

subroutine RECODE:
1. Code file (input)
2. Index set of class labels
3. Vector with codes in index set as appearing on the grid file
4. Vector with selection (0/1) from index set
5. Code file (output)
6. Index set of class labels (input)
7. Vector with codes in index set as on the grid file (input)
8. Vector with selection (0/1) from index set
9. Mapping: parameter defined in index set %2 providing
   destination code in 6.

subroutine SLICE:
1. Real valued file with weights ($v \geq 0$) to be used for fractions (input file)
2. Integer valued file (input file)
3. Index set of class labels
4. Vector with codes in index set to be assigned to the class
5. Vector with selection from index set
6. Vector with upper bounds on classes
7. Name of output file with selected codes; if NOFIL is entered, no file

```
    is written
 8. NLEV: maximal number of classes in SAS-maps (<9)
 9. NODATA: code indicating non-availability of data
10. CLEAR: release the overlay directly (=2) or
    after the next ZONDIR/ZONMOL (=1), not at all (=0)
```

subroutine ZONDIR (or ZONMOL):

```
 1. Real valued file with weights (w, no sign restriction) for frequency
    calculations(input file)
 2. Name of x1-file (input) for which frequency and ranking calculations
    are to be conducted (A,M,R,PV,C or <filename>), where A = all; M =
    mainland code 1 (i.e. excluding water); R, PV or C region, province or
    county level.
 3. Name of second zone file (input) for which frequency and ranking
    calculations are to be conducted
 4. Fil10: the filename (at most first ten characters will be used); four
    files will be generated, with suffixes .hdr (header),.gcd (integer grid
    file with codes),(real valued gridfile with frequencies).grd and .gms
    (GAMS readable file with district aggregates).
 5. Header for index set
 6. TOPN: computations are done for at most the TOPN-highest (by district)
    frequencies.
 7. NLEV: maximal number of classes in SAS-maps (<9)
 8. NODATA: code indicating non-availability of data
```

Subroutine MATCH:

```
1. PAYOFF: a real valued file with payoff observations (input)
2. X1 X2: x-variables, as in ZONDIR
3. FLOUT: Root of label for output files
4. METHOD: as selected
5. NRQUANT: number of quantiles for best treatment in print
6. RISKAV: risk aversion exponent
7. NODATA: symbol for missing data
```

Any *.gcd* written by ZONDIST.EXE will also have a *.hdr* file corresponding to it. This *.hdr* file is an ASCII-file the labels of which can be modified by the user but the integer codes corresponding to these labels may, of course, not be altered, since they correspond to the codes appearing on the *.gcd* file. The index sets containing the actual labels for zoning must either be entered in the GAMS job or read directly from *.hdr* file that can be edited freely by the user. The latter is specified in the GAMS-command by referring to any user-specified index set with a single class.

In case of projection from survey to map, files are generated with extensions *.hdrm*, *.gcdm* and *.grdm*, with vectors of length equal to the number of cells in the map, as opposed to the files without *.hdr*, *.gcd* and *.grd* that have length equal to the number of household respondents.

Subroutine MATCH needs particular attention. Matching produces several files that project an optimal intervention on individual sample points, even for those where no payoff or treatment measurement is available. The results below are calculated for each (x,y) separately, and are referred to as a class. The larger category x is referred to as x-class.

An optimal choice is made both for the class and the x-class. At points where no observation is available on y while x is observed, the optimal choice for the x-class can be made, not for the

class itself. Hence, the command also provides for prediction/interpolation. The incidence of wrong choices being reported on in the .txt file, and projected back to survey level through .gcd and .grd files that contain, as in ZONDIR, one vector each, for codes and real values, respectively. The calculated payoffs can be compared with the observed ones that are entered as input file. A set of output files from MATCH command are in the following (when output file name specified in the MATCH command is FLOUT)

.gcd and .hdr

```
FLOUT                  most frequent treatment in this class
FLOUT1                 best treatment in this class
FLOUT2                 runner up in this class
FLOUT3                 best treatment in this x-class
```

.grd

```
FLOUT                  average payoff of the treatment in this class
FLOUT1                 payoff of best treatment in this class
FLOUT2                 payoff of runner up in this class
FLOUT3                 %cases best payoff < observed payoff in this x-class
FLOUT4                 %cases best payoff is optimal in this x-class
FLOUTg                 payoff difference of observation from its match
FLOUTs                 standard deviation of best treatment in its class
```

.txt

```
FLOUT                   report file
```

D.5 Example

Here follows an example of such a sequence. The practical way of using the facility is to start with the sample program ZONDAT.GMS provided in the subdirectory SRC, and modify it to fit the problem at hand. This program contains the full GAMS-program of the example in this section.

Set definitions for administrative levels (CN = districts, PV = provinces, and R = regions) and correspondence between administrative level (between CN to PV and PV to R) are required by ZONDAT.GMS. These can be defined as separate .gms files, stored in "../DECLARATIONS" folder and included in ZONDAT.GMS as follows: the first three include statements include CN, PV and R set definitions while the last file includes correspondence from CN to PV and from PV to R):

```
$INCLUDE '..\Declarations\CN.gms'
$INCLUDE '..\Declarations\PV.gms'
$INCLUDE '..\Declarations\R.gms'
$INCLUDE '..\Declarations\CN_PV_R.gms'
```

These set definitions and the correspondences can be written into ZONINP.txt via the following:

```
PUT '* Number of countries  R ' / CARD(R):5:0 /;
PUT '* Number of provinces  PV ' / CARD(PV):5:0 /;
PUT '* Number of districts  CN ' / CARD(CN):5:0 ;

PUT / '* country codes R and rank' ;
LOOP(R,
```

```
     PUT / @2 R.TL:<3:0, @19, ORD(R):2:0, @23, R.TE(R) ;
     );
PUT / '* province codes PV and rank' ;
LOOP(PV,
     PUT / PV.TL:<4:0, @19, ORD(PV):2:0, @23, PV.TE(PV) ;
     );
PARAMETERS
  POPCN(CN)    "population by county"
  ;

POPCN(CN) = 1;

PUT / '* county codes and rank' ;
LOOP(CN,
     PUT  /  CN.TL:14:0,@15,  ORD(CN):6:0,  @23,  CN.TE(CN):20:0,  "  ",
POPCN(CN):8:2;
     );
```

In addition, the list of files to be kept in the subdirectory DAT as opposed to WKRUN is written, as specified by the index set REFER:

```
SET REFER  'Reference variables             '
 /
 NOFIL     ' no file                    '
 F1        ' unit file                  '
 SOILS     ' soils                      '
 RAINFALL  ' rainfall                   '
 TEMPERAT  ' temperature                '
 TEMPZON   ' temperature zones          '
 SOILZON   ' soil zones                 '
/;
```

These filenames are then written into ZONINP.txt (for later reference by the Fortran code) via the following:

```
PUT / '* number of files in REFER ';
PUT / CARD(REFER):3:0;

PUT / '* files kept in subdirectory DAT as opposed to WKRUN' ;
LOOP(REFER,
    PUT / REFER.TL:14:0,@15, REFER.TE(REFER):20:0;
    );
```

A file path should also be specified for various files that the Fortran may utilize. The root directory is specified via the string being put into FADDRESS.TXT. For relative root directory, “..\” should be put.

```
FILE ZONDAT /..\WKOUT\ZONDAT.log/;

 FILE FADDR /..\WKINP\FADDRESS.TXT/;
 PUT FADDR / "..\";
 PUTCLOSE FADDR;
```

ZONHEAD.GMS contains file path specification for ZONINP, and SETX1, as well as file relevant parameters of ZONINP.txt and ZONDAT.log. All file path names can be made relative.

```
$INCLUDE '..\SRC\zonhead.gms' ;
```

The options to control various visual parameters of the maps via map macros should now be defined. For further details of the specification, see van den Boom and Pande (2007)

The set ‘COLORS’ contains a list of color combinations that are available map plots. The names of the set elements are standard and should be from the list of color schemes specified in van den Boom and Pande (2007). When chosen ‘DEFAULT’, a version of ‘GREEN_RED’ is implicitly specified.

```
   COLORS      'SAS colors'
     /
      GREEN_RED               ‘from green to red’
      DEFAULT                 ‘use default’
      RED_BLUE                ‘from red to blue’
      GREEN_RED_BLUE          ‘from green to blue via red’
     /
```

Additional colors can be added to the set and used via subset selection and only one element can be selected. For example, the following selects GREEN_RED as the color scheme.

```
COLORS(' GREEN_RED ') = YES;
COLORS('DEFAULT') = NO;
COLORS(' RED_BLUE') = NO;
COLORS(' GREEN_RED_BLUE ') = NO;
```

The set ‘BACKGROUND’ contains possible label names of the background pixels, if any background coloring is requested by the user. Background specification requires that the output maps generated by any of the jobs of GRCP are subset of some bigger map. Correspondingly, it will require another locat file called “locat_subset.grd” in the DAT directory as well its SAS data set counterpart, named as “locat_subset.sas7bdat,” in the SASDAT directory of ‘MAKEMAP’ folder. ‘BACKGROUND’ specification then paints the map difference between the two (the bigger map and a subset of it) by a standard color and is named by the selected element of the set in the legend of the map. Similar to the selection of color scheme, only one element can be selected.

```
 BACKGROUND  'Background label'
    /
    NONE                    'none     '
    unpopulated             'unpopulated'
    SWEET                   'sweet'
    /
```

The set “Anno_sel” contains the kind of annotates that a user want to show on requested maps. These element names are standard and created within map macros facility (van den Boom and Pande, 2007). Here subsets can be selected in a manner similar to the selection of color schemes but with possibility of multiple selection. The order in which any subset appear on the map, in case of when annotates overlap, follow the order in which the elements of this set is specified.

```
Anno_sel  'Annotate selection option in makemap'
   /
   anno_CN                        ' put district annotates'
   anno_PV                        ' put provincial annotates'
   anno_R                         ' put regional  annotates'
   anno_ML                        ' put mainland annotates'
   none                           ' put no annotates'
    /
```

The following two sets “Adm_sel” and “Selection” allow sub-map plots by CN, PV, R, or ML codes. The elements of “Adm_sel” are standard, and only one element can be selected in a manner similar to color scheme selection.

```
Adm_sel   'Administrative selection option in makemap'
/
CN_sel         'select districts'
PV_sel         'select provinces'
R_sel            'select regions'
ML_sel         'select mainland'
None             'select none'
/
```

The elements of the selection set identify which units with in the administrative definition selected above are to be selected. It should therefore be ensured that the elements in the selection set belong to the selected administrative definition. Each element of this set can have a maximum of 14 characters that can be used to specify either a single identifier or a range of identifiers. First two characters are not considered part of numeric identification. In defining a range, follow first two characters by the lower limit, then a “-” and then the upper limit. All the elements in the set are then considered for sub-map selection, including those in the range (if included as one of the set elements).

```
Selection   'Admin number selection option in makemap'
/
sl200             'unit 200'
sl201-400         'a collection of 200 units from 201 to 400'
/
```

The following two sets control the kind of plotting procedure and the kind of plotting device used by map macros. Option “gplot” within set “Plotproc” should be used when a plot of two variables on the same map is requested. For further information on these options, refer to (van den Boom and Pande, 2007).

```
Plotproc   " which plotting procedure to use in the makemap"
/
gplot       'procedure gplot'
none        'none'
/

Device    " which device to use"
/
gif_win2     'gif_win2'
win          'win'
none         'none'
```

```
    /
```

In order to demonstrate the selection of certain administrative units in creating maps, the following will select CN (district) values that are 200, and from 201 to 400. However, the user should ensure that the values defined in set ‘Selection’ are valid values available for CN (that there are districts with values 200, and 201-400).

```
Adm_sel('CN_sel') = YES ;
Adm_sel('PV_sel')= NO ;
Adm_sel('R_sel')= NO ;
Adm_sel('ML_sel')= NO ;
Adm_sel('none')= NO ;
```

The following options, defined as scalars, specify 1) NLEV: number of levels for the legend in a requested map, 2) ENLARGE: whether a sub-map after requesting selection via the set “Adm_sel” and “Selection” should resize to the map area, 3) lsopt: factor on line thickness of requested annotates on a map, 4) POSITION: where should the legend appear, 5) dpi: the resolution of the map, 6) DECIM: the precision of the numbers appearing in the legend, 7) scale_gr: scaling of the map within the plotting area, and 8) scale_px: factor on the height of the pixels in the maps. In order to invoke default values of all these options except NLEV, -1 should be assigned. For option NLEV default value of 8 levels is assigned whenever NLEV is assigned a value less than equal to 0. For further details on these options and its’ various possible values, readers are referred to (van den Boom and Pande, 2007).

```
SCALAR NLEV, ENLARGE, lsopt, POSITION, dpi, DECIM, scale_gr, scale_px ;


NLEV = - 1 ;
ENLARGE = -1 ;
lsopt = 0.1 ;
DECIM = -1 ;
POSITION = -1 ;
dpi = -1 ;
scale_gr = -1 ;
scale_px = -1;
```

Next, the data type must be specified:

```
SCALAR DATATYP ' Data type: 1=gridmap 2=survey';

DATATYP = 1;
```

If DATATYP=2, the program expects either NLEV=0 (no maps), or if NLEV is positive, a pair georeferencing files to be available: locat.grd for zoning calculations over the survey, and locat2.grd, for mapping of the survey to a grid map, after spatial interpolation.

We also specify index sets and parameters of the zoning itself.

```
SET CLTEMP 'Temperature classification'
          /
    CLD    ' Cold     '
    MOD    ' Moderate '
    HOT    ' Hot      '
```

```
           /;
```

The selection and coding are communicated via parameters. The code number should be the one appearing in the data set

```
PARAMETER CODTEMP (CLTEMP) ' Codes for temperature classification '
             /
    CLD      2
    MID      3
    HOT     10
             /;
```

Selection is done by the entry itself.

```
PARAMETER SELTEMP (CLTEMP) ' Selection for temperature classification '
             /
    CLD      1
    MID      0
    HOT      1
             /;
```

The upper bounds are also communicated via a parameter

```
PARAMETER CLS(CLTEMP) ' Bounds for temperature classification '
             /
    CLD      5.
    MID     16.
    HOT     50.
             /;

SET CLSOIL 'Soil classification'
    /
    SND   ' Sand     '
    CLY   ' Clay     '
    RCK   ' Rock     '
    /;

PARAMETER CODSOIL (CLSOIL) ' Codes for soil classification '
             /
    SND      7
    CLY      3
    RCK      2
             /;

PARAMETER SELSOIL (CLSOIL) ' Selection for soil classification '
             /
    SND      1
    CLY      0
    RCK      1
             /;
```

Also, some control parameters have to be set. The local ones are communicated directly as parameter in a BATINCLUDE; the global ones that now follow are set in the main program and keep their value until they are set at a different value.

The number of top combinations that the user wants the program to write in a txt file when executing ZONDIR should be specified via NTOP scalar.

```
SCALAR NTOP 'Top N';
 NTOP = 5;
```

The number of levels that can be displayed in various maps should be specified by NLEV. Another similar scalar NLEV0 with value 0 can also be defined and used in case the user wants a routine not to produce any output maps.

```
SCALAR NLEV 'Number of levels in SAS-printing';
 NLEV = 10;
SCALAR NLEV0 'No SAS maps';
 NLEV0 = 0;
```

A numeric value for representing no data value should also be defined via a scalar called NODATA.

```
SCALAR NODATA   ' entry for nodata';
 NODATA = -9999;
```

If the user wants to interpolate over missing values during any procedure, scalar INTERPOL should have value 1.

```
*SCALAR INTERPOL ' interpolate 0/1 ';
INTERPOL = 0;

* Print control (global i.e. remains active until new assignment)

  CONTROLP = 1;
```

To control operations ZONDIR or ZONMOL, scalar CONFID specifies the confidence level in the sense that classes are identified in ascending order of their mass upto this fraction of total mass.

```
* Confidence level as: ' number of classes needed to reach …' (global)
  CONFID = .95;
```

The KERNSET-command is needed in preparation of the operation ZONMOL. As parameter is has a set with 4 elements pointing to filenames. Reserved names are VOID1-VOID4 (unit values for unused elements), LAT and LON (latitude and longitude as available from LOCAT file). If the filename appears in REFER it will be expected to be available on DAT otherwise on WKRUN.

```
SET ZVAR ' File names and description real valued variables'
            /
    LAT          'Latitude           '
    LON          'Longitude          '
    VOID1        'void               '
    VOID2        'void               '
           /;
```

The KERNSET routine that prepares for ZONMOL needs to be preceded by three scalars that define the lower, and upper bounds of the multiplication factors on a Gaussian reference bandwidth as well as the number of iterations requested by the user to reach some agreeable window size. However, the user should be careful in defining THETALO as well as NTHETA because high THETALO or NTHETA may become computationally expensive.

```
* Specifying kernel window size
  THETALO = .1;
  THETAUP = 2.;
  NTHETA = 3;
```

In order to specify mollification for the KERNSET subroutine, the scalar MOLL should be appropriately assigned. MOLL = 1 indicates that mollifier weights should be assigned on grid cells (survey files). These computed weights override any weight file specification provided in the GAMS job. The Nadaraya Watson option differs from mollifier in that it normalizes the weights to let them sum to unity. In either case, in view of the high computational requirements of mollification , users are recommended to use MOLL = 0 when fast computation is required.

```
* Mollifier (1= mollifier, 2= Nadaraya Watson)
  MOLL = 2;
```

The user should, via the parameter EDGETH indicate whether maximization should be of absolute mass of correct maximum likelihood predictions (EDGETH = 0) or of their edge over second best (EDGETH = 1):

```
  EDGETH = 0;
```

A second edge parameter, unrelated to the first, controls the stepwise selection. For each given combination of variables, it performs a maximum probability estimation but it in addition selects the combination of (y,x)-elements that maximizes the edge value: EDGESL = 0 keep all; EDGESL = 1 select from all possible combinations; EDGESL = 2, select from combinations as supplied by the user, on file WKINP\Edgesl.txt ; this file can be obtained by editing WKINP\Edgesl.txt as produced under a run with EDGESL=1:

```
   EDGESL = 0;
```

The KERNM-command (counterpart of KERNSET used for mapping and only needed if this command is used) defines the rules for projection from survey to map. The set must have 4 elements.Specify the index set for KERNM-command (counterpart of KERNSET used form mapping and only needed if this command is used). The set must have 4 elements. Reserved names are VOIDM1-VOIDM4 (unit values for unused elements), LATM and LONM (latitude and longitude (available from LOCATM file). Since this command is needed when survey point observations are spatially interpolated, specifying CNM or PVM or RM (but never to specify more than one at a time) will yield spatial interpolations that are constrained by CN or PV or R boundaries, respectively. If filename appears in REFER it will be expected to be available on DAT otherwise on WKRUN.

```
SET ZVARM ' File names and description real valued variables'
            /
    LATM         'Latitude           '
    LONM         'Longitude          '
    CNM          'CN constrained     '
    VOIDM2       'void               '
```

```
            /;
```

Corresponding to survey interpolation onto a map (when using KERNM routine), the scalars RULEM and THETAM are also needed to specify the kind of interpolation and the multiplication factor on the Gaussian reference bandwidth respectively (the latter only used when mollifier is used as the interpolation technique). However, care should be taken in defining THETAM because with high THETAM computation may be slow. Therefore, are therefore recommended to use RULEM = 0 in case they desire quicker results.

```
* Specify rule and kernel window size for mapping on grid
  SCALAR RULEM,THETAM;

  THETAM = 1.;
  RULEM = 1;  {0=no projection 1=mollifier 2=nearest neighbor)
```

Finally, in order to name the operations that are realized by invoking routines ZONDIR, ZONMOL, APPLY or AGGREG, the following sets should be named. Every operation should have a set of its own,

```
* Result set: only the label and name matter (up to 30 characters
* for long label
SET RESIND 'Where do most farmers live?'
    /
    NORES
    /;

SET RESIND2 'Where do most farmers live?'
    /
    NORES2
    /;
```

Now commands (here $ is used as shorthand for `$BATINCLUDE ..\LIBRARY\` in the actual GAMS-program, where each command must be entered on a single line) can be entered.

```
* KERNM: defines the kernel functions for mapping on grid
* %1 Index list input files
* %2 Rule (0=no projection, 1=mollifier, 2=nearest neighbor)
* %3 Window size factor (e.g. 1.=default, 0.= dot plot)

$KERNM.gms ZVARM RULEM THETAM
```

The following SLICE command generates a map in the “Makemap\Pict” folder (since NLEV >0). The resulting map figure generated will be named after the output file name with a title “Result from slicing”, while the legend of the map will follow the name specified for the set CLASPOP (the set that defines the names of various classes). For DATATYPE = 2 (i.e. data of type survey), SLICE also uses the weight matrix generated by the KERNM operation above to display the winning class among neighboring points for which survey observation(s) are available. Winner is the class that has most influence on a pixel. Hence, this operation conducts a spatial interpolation between survey data points.

```
* SLICE: converts from real to coded file on the basis of class bounds
* %1 v weights
* %2 Real valued file   (input file)
```

```
* %3 Index set of class labels
* %4 Vector with codes in index set as appearing on the grid file
* %5 Vector with selection (0/1) from index set
* %6 Parameter that specifies the bounds
* %7 Name of output file; if NOFIL is entered, no file is written
* %8 NLEV number of levels in SAS-printing
* %9 NODATA code for 'no data'
* %10 CLEAR whether to drop the information from the character string
*    0=keep/1=drop after next ZONDIR or ZONMOL/2=drop immediately

$BATINCLUDE ..\LIBRARY\SLICE.gms POP1 POP1 CLASPOP CODPOP SELPOP BNDPOP
POPCL NLEV NODATA "2"
```

The following SLICE doesn't produce any map, as the number of levels that are input into the subroutine is NLEV0 (which is assigned 0).

```
* second slice
$BATINCLUDE ..\LIBRARY\SLICE.gms LGP1 RFL1 CLASRAIN CODRAIN SELRAIN RAINBND
RAINFIL NLEV0 NODATA "0"
```

Now turning to ASLICE below, a map named SLICEFIL.gif will be produced (since NLEV>0) with a title "Result from Antislicing", and a legend title named after the title of the set CLASRAIN.

```
* ASLICE: converts from code to real valued map (Antislice)
* %1 Real valued file   (input file)
* %2 Index set of class labels
* %3 Vector with codes in index set as appearing on the grid file
* %4 Vector with selection (0/1) from index set
* %5 Parameter that specifies the bounds
* %6 Name of output file
* %7 NLEV number of levels in SAS-printing
* %8 NODATA code for 'no data'

$ASLICE.gms RAINFIL CLASRAIN CODRAIN SELRAIN RAINBND SLICEFIL NLEV NODATA



* DATHARM: harmonize data availability by dropping all observations with
* any missing; multiply by factor
* %1 name of input file
* %2 NODATA: missing data code for input file
* %3 name of reference file (input)
* %4 NODATA: missing data code for reference file
* %5 FACT: factor to multiply .grd of input file with
* %6 name of adjusted input file (output, none if NOFIL is entered)
* %7 NODATA: missing data code on output

$DATHARM SCC1 NODATA CEO1 NODATA SCC1 NODATA FACT
```

The following CROSS routine also generates a map. Since CROSS is generally used to update the classes selected in a variable for further analysis, the resulting figure is aptly titled "Result from selection".

```
* CROSS: crosses by extending the y-part of the character string
```

```
* %1 weights
* %2 name of coded file (input)
* %3 Index set of class labels
* %4 Vector with codes in index set as appearing on the grid file
* %5 Vector with selection (0/1) from index set
* %6 Name of output file if NOFIL is entered, no file is written
* %7 NLEV number of levels in SAS-printing
* %8 NODATA code for 'no data'
* %9 CLEAR whether to drop the information from the character string
*    0=keep/1=drop after next ZONDIR or ZONMOL/2=drop immediately

$CROSS.gms CEO1 SCC1 CLASSOIL CODSOIL SELSOIL CROSSFIL NLEV NODATA "0"
```

The following ZONDIR produces a total of 6 maps or 3 pairs of maps in case DATATYPE = 1 (data are of type gridmap) holds. The first pair of maps, named ZOUT1.gif and ZOUT1c.gif, reports on observed *y*-class combinations at each pixel of the map and its frequency of occurrence corresponding to the *x* observed for that pixel, hence their titles “Classification” and “Frequency of observed classes”, respectively. The legend of these maps will now be titled after the set title of RESIND defined above, which also names the operation that this ZONDIR is part of. Other two pairs of maps that will be generated are {ZOUT11.gif, ZOUT11c.gif} and {ZOUT12.gif, ZOUT12c.gif} that report on most frequent *y*-classes and their observed frequencies; and second most frequent classes and their frequencies, respectively. While the legend title of all these maps remain the same and are named “Where do most farmers live?”, the map titles are {“Most frequent class”, “Highest class frequency”}, and {“Second most frequent class”, “Second highest class frequency”}, respectively. In case DATATYPE = 2 (data are of type survey), ZONDIR only outputs two maps: ZOUT1.gif (titled “most frequent class”), and ZOUT1c.gif (titled “weight of influence”). Unlike the case of DATATYPE = 1, these maps only report the most frequent *y* class in ZOUT1, while ZOUT1c reports on the weight of influence those winning *y*-classes have in their neighborhood, and require KERNM as preparatory step.

```
* ZONDIR: direct zoning (for qualitative variables only, no z-variables)
* %1 weights
* %2 Parameter 1 indicating for which district frequency and ranking
*    calculations should be conducted (A/M/R/PV/CN or <filename>)
* %3 Name of second zone file (input) for which frequency and ranking
*    calculations are to be conducted
* %4 name output files (excl extensions)
* %5 Header
* %6 TOPN
* %7 NLEV of levels in printing
* %8 NODATA code for 'no data'
* %9 INTERPOL interpolate

$ZONDIR.gms POP1 CN POPCL ZOUT1 RESIND TOPN NLEV NODATA INTERPOL
```

The following APPLY routine applies the most frequent *y*-classes as computed in the previous ZONDIR operation (and stored in ZOUT1.txt) and corresponding to *x*-variables CN and POPCL. An output map called ZFLOUT2 is created with a map title “Prediction” and a legend title corresponding to the set label of RESIND.

```
* APPLY: Applies mapping to new x-conditioning
* %1 File (.txt) to obtain the y|x association from
* %2 Parameter 1 indicating for which district frequency and ranking
*    calculations should be conducted (A/M/R/PV/C or <filename>)
```

```
* %3 Name of second zone file (input) for which frequency and ranking
*    calculations are to be conducted
* %4 name output files (excl extensions)
* %5 Header for results
* %6 NLEV of levels in printing
* %7 NODATA code for 'no data'

$APPLY.gms ZOUT1 CN POPCL ZFLOUT2 RESIND NLEV NODATA

* AGGREG: aggregates to district file (level indicated)
* %1 File (.txt, .hdr input) to obtain the y|x association from
* %2 File (.gms, output)
* %3 Parameter indicating for which district frequency and ranking
*    calculations should be conducted (A/M/R/PV/C or <filename>)
* %4 Name of second zone file (input) for which frequency and ranking
*    calculations are to be conducted
* %5 Header for results

$AGGREG.gms ZOUT1 CN POPCL ZFLOUT2 RESIND


* MAPTOSUR: assigns map values to survey data
* %1 Name of input file on map   (input file)

$MAPTOSUR.gms SCC1


* KERNSET: defines the kernel functions and the mollifier operations
* %1 Name of z1-file (input) for which frequency and ranking calculations
*    are to be conducted (F1, LAT, LON or <filename>), where F1 = unit
*    values ; LAT = latitude; LON = longitude.
* %2 THETALO: Lower bound on window size.
* %3 THETAUP: Upper bound on window size.
* %4 NTHETA: Number of evaluations at different window sizes.
* %5 EDGETH: Edge (0=no 1=yes)
* %6 MOLL: 1=mollifier 2=Nadaraya Watson
* %7 FLOUT: output file for estimated weights (mass) (NOFIL = no file
*    written)

$KERNSET.gms ZVAR THETALO THETAUP NTHETA EDGETH MOLL FLOUT
```

With prior invocation of KERNSET above, ZONMOL produces mollified maps that are similar in content to those produced by ZONDIR.

```
* ZONMOL: kernel smoothed zoning with y x and z (real-valued) variables
* %1 W-weights
* %2 Parameter 1 indicating for which district frequency and ranking
*      calculations should be conducted (A/M/R/PV/C or <filename>)
* %3 Name of second zone file (input) for which frequency and ranking
*    calculations are to be conducted
* %4 name output files (excl extensions)
* %5 Header
* %6 TOPN
* %7 NLEV of levels in printing
* %8 NODATA code for 'no data'
* %9 INTERPOL interpolate (0/1)
```

```
$ZONMOL.gms POP1 CN POPCL ZOUT2 RESIND2 TOPN NLEV NODATA INTERPOL
```

The following lines lead to a matching operation. The treatment variable is assumed to be the one used in the last CROSS operation (which is `HCTPXASc` (poverty head count)). Note also that matching also requires the user to specify which class values of the treatment variable is “non-treatment” or control value. This is done for the treatment variable in this example by specifying a value of 2 in . The other classes of the treatment variable are then (varying degree of) treatment values. The additional y variables are built up via prior SLICES and CROSSES.

First we show the last CROSS with corresponding class definitions for clearer discourse:

```
*---------------------------------------------------------------------
*HCTP 'Head count poor’

SET CLASHCTP 'Head count poor'
             /
    HCLO   'low poverty  '
    HCME   'medium poverty '
    HCMH   'medium-high poverty '
    HCHI   'High poverty   '
             /;

PARAMETER CODHCTP (CLASHCTP) 'Head count classification '
             /
    HCLO    1
    HCME    2
    HCMH    3
    HCHI    4
           /;

PARAMETER SELHCTP (CLASHCTP) 'Selection for HEADCOUNT classification '
             /
    HCLO    1
    HCME    2
    HCMH    1
    HCHI    1
           /;

PARAMETER BNDHCTP (CLASHCTP) 'Bounds for HEADCOUNT classification '
             /
    HCLO         20.
    HCME         35.
    HCMH         60.
    HCHI         100.
            /;
*---------------------------------------------------------------------

$BATINCLUDE  ..\LIBRARY\CROSS.gms  F1  HCTPCL  CLASHCTP  CODHCTP  SELHCTP
HCTPXASc NLEV NODATA "0"
```

Now follow matching-specific definitions:

```
SET MATCHMETH
/
NEAREST     'nearest neighbor                '
MOLLCON     'Nadaraya Watson on control group'
MOLLALL     'Nadaraya Watson on all'
AVERAGE     'Average treatment               '
ZEROREF     'Payoff as gain from treatment   '
/
;

SET METHOD(MATCHMETH);
METHOD(MATCHMETH)=NO;
METHOD('MOLLCON')=YES;

* Risk aversion exponent
SCALAR RISKAV;
RISKAV = 1.;

{4. Matching
The y-vector is built up via CROSS-steps as with ZONDIR. The treatment
variable should be in the last crossing before the MATCH command. Any
crossing beyond it will be neglected in calculation.
Next, the matching is initiated through the command:
}

SET RESIND6 'matching '
    /
    NORES6
    /;
```

Finally the MATCH command, where the payoff file is defined by ASCO (Arsenic contamination) and the x variable files are PV (provinces) and POPDCL (population density classes):

```
$BATINCLUDE  ..\LIBRARY\MATCH.gms  ASCO  PV  POPDCL  ZOUT4  RESIND6  METHOD
NRQUANT RISKAV NODATA
```

The command above then generates the following set of files:

.gcd and .hdr files:

```
ZOUT4                 most frequent treatment in this class
ZOUT41                best treatment in this class
ZOUT42                runner up in this class
ZOUT43                best treatment in this x-class
```

.grd files:

```
ZOUT4                 average payoff of the treatment in this class
ZOUT41                payoff of best treatment in this class
ZOUT42                payoff of runner up in this class
ZOUT43                %cases best payoff < observed payoff in this x-class
ZOUT44                %cases best payoff is optimal in this x-class
ZOUT4g                payoff difference of observation from its match
ZOUT4s                standard deviation of best treatment in its class
```

.txt file:

`ZOUT4` report file

Finally, the closing line of the GAMS program.

```
* closing line
PUT ZONINP / "#";
```

Grid files are stored under DAT or WKRUN depending on whether they appear in REFER; code and header files in DATC or WKRUN; result and debug files appear under WKOUT. Input generated by GAMS is kept in WKINP.

This completes a zoning operation. Several of such operations can be conducted in a single GAMS-job.

Packing

An example is now presented for Bangladesh that demonstrates how different Arsenic concentration levels vary with different land use categories and different elevation levels. By conditioning such a map on population density, one can also identify the most (or second most) likely combination for observed population densities. Given three variables with multiple classes, combinations on such maps may become intractable, making it difficult to isolate combinations of interest. The exercise, therefore, calls for a stepwise procedure that retains tractability.

For this, we consider an example of step-wise "packing", whereby a sequence of ZONDIR/ZONMOL commands is used to cross a number of files with categorical data. The intent is to create a "packing" of these three variables conditioned on province and population density maps (ppl/sq.km.) and isolate a subset of observed category combinations of these three variables of interest. All the maps are at 30 arc seconds with 841 rows and 601 columns with approximately 174310 active cells (i.e those cells that have values). They are obtained as real valued map, and the "SLICE"-command is used to classify them into categories. Such categorization also makes a case for robust instrumentalization (see Section 4.3). For Arsenic concentration the classification is:

Range(µg/l**)**	**Class label**	**Class characteristics**
0. – 10.	Safe	Safe for human consumption
10. - 50.	Low	Low harm
50. - 100.	Medium	Medium harm
100. - 300.	High	High harm
300. - 500.	Very high	Very high harm

Table1. Typical class definition used in SLICE

"SLICE" operation yields the map operation shown in Figure 1. Already classified maps can be used as input maps via the CROSS command. The other two categorical maps are also shown in Figure 2.

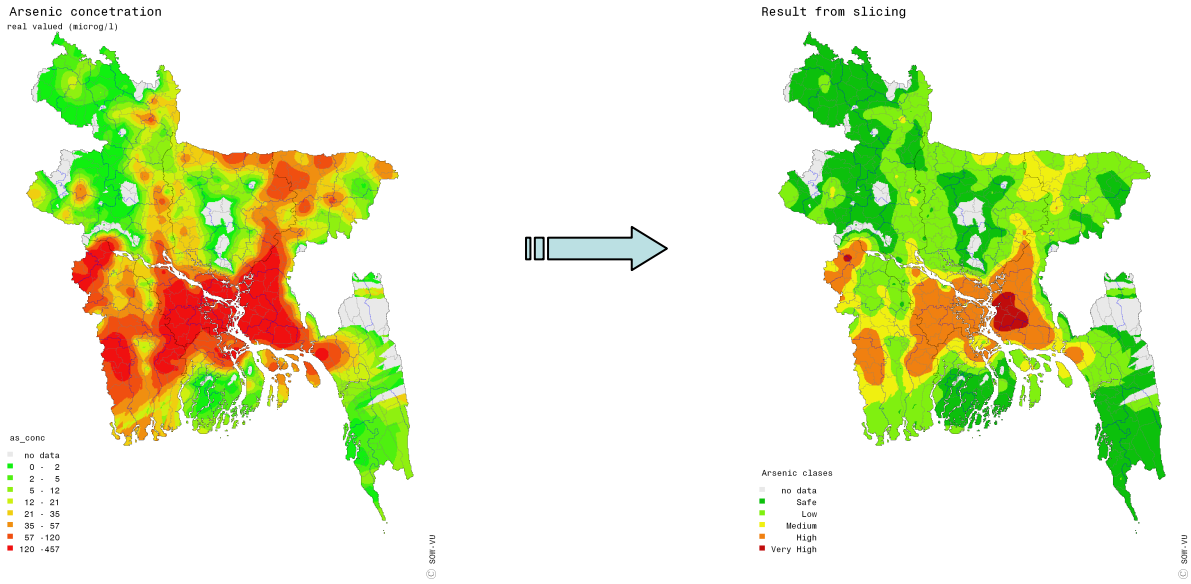

Figure 1. An Example of "SLICING" a real valued map to a categorical map: Arsenic concentration.

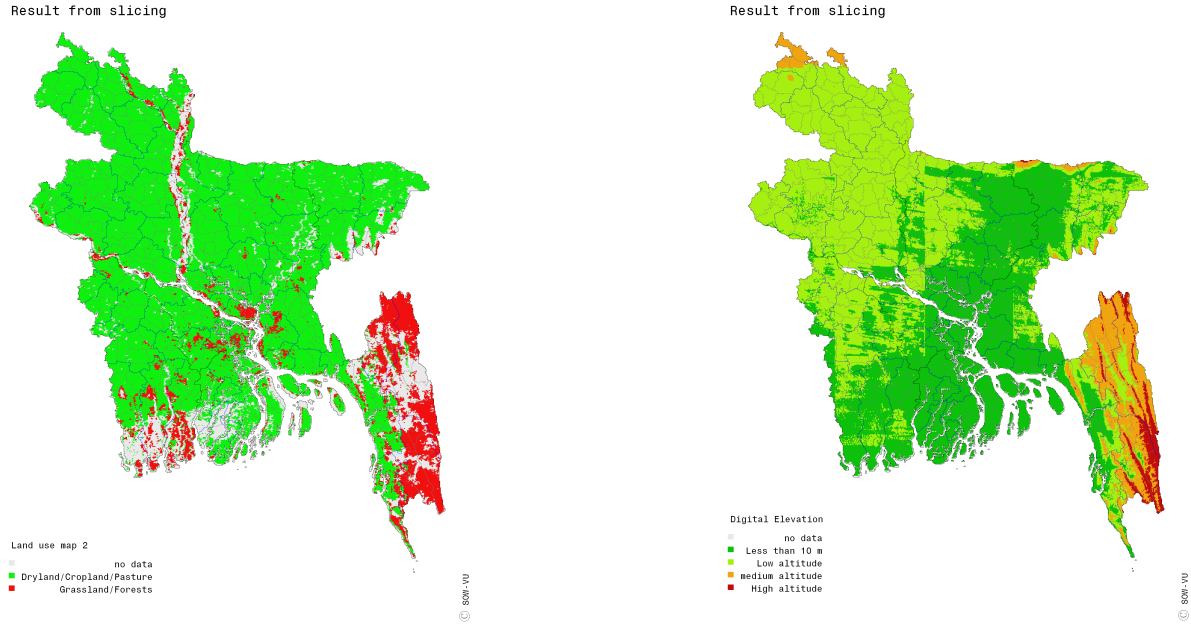

Figure 2. Categorical variable maps to be packed, along with Arsenic concentration category map shown in Figure 1.

With categorical maps available, "packing" is performed in a sequential manner. As a final result it produces *y*=classified maps of{Arsenic concentration, land use and elevation} to be "packed" conditioned on *x*=classified maps of{province and population density}.

First, a pair of elevation and land use categorical maps, y1, is considered with conditioning of categorical maps as *x*. "CROSS" or "SLICE" with drop option (=0) should be operated on variable maps of interest before invoking "ZONDIR" at any level of "packing". Figure 3 shows the output categorical map generated by the first ZONDIR. We remark that the labels of various observed combinations are long already and that subsequent ZONDIR operation will make them

even longer. Therefore, some relabeling will usually be needed eventually, to keep the headings of maps and tables interpretable.

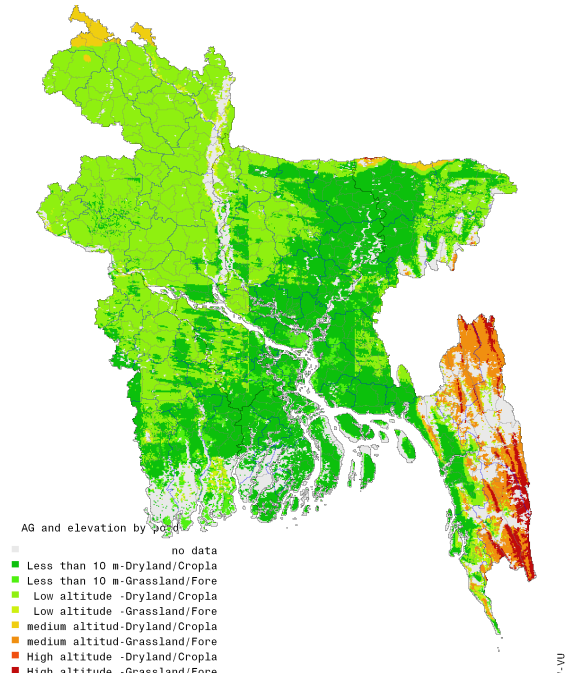

Figure 3. Elevation-Landuse "packed" map. It shows various combination categories that exist throughout the study area.

This relabeling can be effectuated by copying the .gcd and .hdr files that correspond to combination codes appearing in Figure 3 from '..\WKRUN' folder to the '..\DATC' folder. It is then renamed and used as another categorical map in the next invocation of ZONDIR. This implies that the user should now include this filename in the "REFER' set of 'ZONDAT.gms' as well. Upon relabeling the observed combination categories in the .hdr file (now in DATC folder) as shown in Table 2 (only the first 3 of the 8 combinations are shown), the relabeled Figure 3 is shown in Figure 4. Note that labels that are replaced in .hdr file are not the labels that appear in the map above, but are program generated. These labels appear under the heading '* Index list' of the .hdr file. However, their order of appearance follows the order in which the labels appear in the legend of the map above.

Labels in .hdr file	**Original Labels**	**New labels**
DLOW_AF1_	Less than 10 m Dryland/Cropland/Pasture	LEAG
DRLO_AF1_	Less than 10 m Grassland/Forests	10AG
DMED_AF2_ 0	Low altitude Dryland/Cropland/Pasture	MEGF

Table2. Relabeling .hdr file in DATC folder, for elevation-landuse "pack", from program generated labels to user-specific ones. This relabeled .hdr file will be used in another ZONDIR operation to achieve next level of packing.

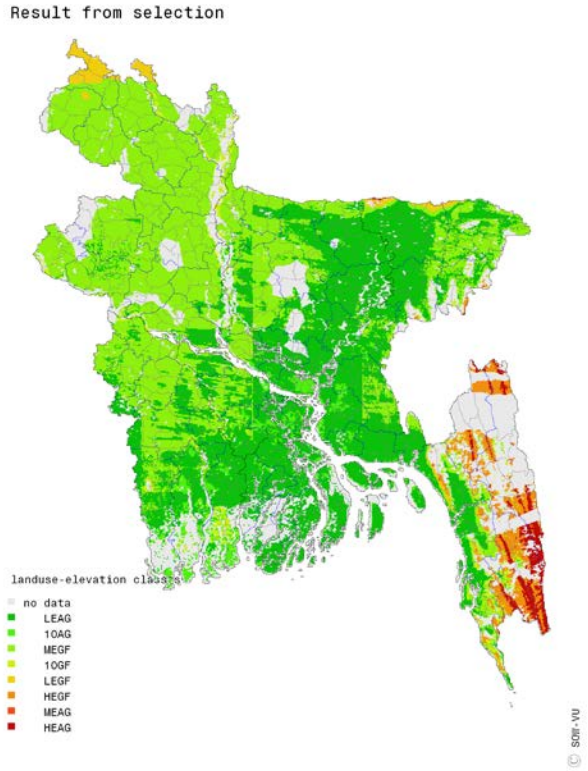

Figure 4. Display of relabeled elevation-landuse "pack". The figure is the same as Figure 3 except that the the legend has been relabeled via changes made in the .hdr file in the DATC folder.

Figure 4 has been created by using "CROSS" procedure of the renamed .gcd file with another .gcd file containing one. Such a "CROSS" operation can be used to demonstrate or check the changes made in .hdr in '..\DATC' folder or as a precursor to a "ZONDIR" to invoke changes made in the .hdr file in DATC folder.

The map of Arsenic concentration can now be packed on top of the combination map of elevation and landuse (produced by the previous "ZONDIR" operation and now available as a gcd file in DATC folder) using another invocation of "ZONDIR", with the same conditioning variables. However such a packing leads to 35 distinct observed combinations. This may be cumbersome as a user may only want to use a selected few of these combinations for display as well as further analysis. Suppose that we want to display only those combinations that correspond to high or very high arsenic concentration classes. Selection of certain classes can be achieved by making appropriate changes under the heading "* Selection" of the .hdr file, by keeping 1 for classes that need to be selected and 0 for the others, a subset is obtained.

Now by (1) renaming the .gcd and .hdr output files of "ZONDIR" that correspond to observed combinations, (2) relabeling certain classes that are of interest, (3) placing them in '..\DATC' folder, (4) selecting those classes that are of interest via changes in .hdr file and (5) including the renamed file name in 'REFER' set of "Zondat.gms"; we operate a "CROSS" similar to the one that generated Figure 4. The resulting combination map appears in Figure 5a. Note that the .gcd and .hdr output files that will now be created as a result of this "CROSS" will only contain those combinations that have been selected, while those classes that have been dropped will now appear as "missing".

Similar result can also be obtained by reordering the classes in the .hdr file (placed in DATC folder) to let the classes that a user wants to display appear first, which leads to (1) later classes will be given a label (called "other classes"), and (2) the order in which these classes appear (except "other classes') on the map will follow the order specified in the .hdr file. Having

reordered the hdr file, “CROSS” operation yields Figure 5b. Note that the classes that were not selected in Figure 5 now appear as “other classes” in Figure 5b. Specifically, .hdr file contains 3 kinds of specifications for observed combinations. The first, appearing under the heading “ * Index list” is where labels for observed combinations is defined that appear in the map. The second, appearing under the heading “* Codes” is the location where codes for various observed combinations are specified that appear in corresponding gcd file. Finally, the third location, appearing under “* Selection” is where the selection should be done. Care should be taken to reorder all three specifications in the same manner. Codes appearing in the gcd file should always remain untouched.

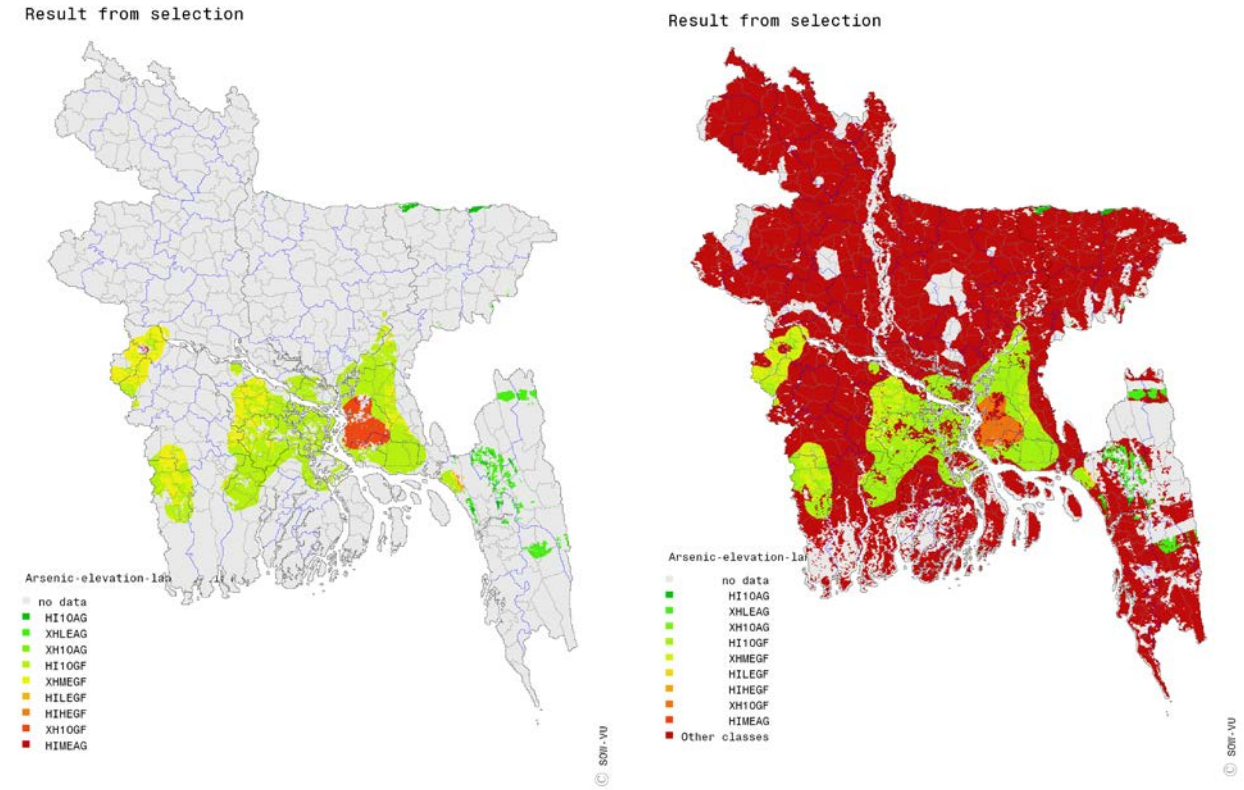

Figure5. Output maps from “CROSS” operations on outputs of “packing” processes. a) Example of selecting specific category combinations obtained at one level of “packing” that will be used at the next level. b) Example of selecting all for the next combinations but displaying only specific categories by reordering the entries in .hdr file in DATC folder. Both figures display almost the same level of information but one carries on only a part of available combinations from one level of “packing” to the next (i.e. Figure 5a) while the other carries on all the available category combinations.

With each step of packing process, an analysis file (output file with extension.txt, say ZOUT.txt) is created. This analysis file reports on marginal as well as conditional frequency of occurrence of various y combinations (“packed” combinations) “upto” that step of packing. However, the combinations that correspond to the non-selected categories of contributing variables in previous steps (via prior CROSSs) are excluded from this analysis file. Thus, the analysis files of another packing step with underlying data of Figure 5a as a base file will be different from when the underlying data of Figure 5b is a base file.

In general, the ZOUT file has a header followed by three sections that comment on overall frequencies, conditional frequencies and the hit ratios of the most likely combinations. The header “Frequency calculations via ..” contains: (a) number of data points, (b) number of data points that has observations for conditioning variables, (c) maximum number of classes for each conditioning value that will be stored, (d) total number of observed categories for “each” of the

conditioning variables, (e) distinct number of y-classes that are observed in the data set, (f) total number of observed "combination" categories of conditioning variables, and (g) total number of variables that are active in the analysis. The header is followed by a section "(1) Overall Classification" which provides (a) a summary of how much mass is explained by the maximum allowed number of combinations in comparison to the total mass, (b) overall (not conditioned) frequency of occurrences of all observed y combinations in the data set along with the ranks of such combinations (by frequency of occurrence). The next section on conditional frequencies "(2) Frequencies by x-class" provides for "each" conditioning value of x (a) the categories of variables constituting that x-combination, (b) total mass of observed categories for "each" element of the y-variable for that x-combination, and (c) y-variable "combination" values and their frequencies, upto user-specified maximum number of top combinations, that appear for that x-combination. It is followed by section on "Report on Goodness of Fit" that includes (a) "(3) Overall fit" that reports on total mass of observation (which is equal to total number of observation if weight files used in analysis is always a unit file), and the fraction of mass explained by the winning and runner-up y-combinations, (b) "(4) Frequency of occurrence of class in top 5" that reports on the "overall" frequency of occurrence of various categories that appear in user-specified top N categories for "each" of the y-variables, and finally (c) "(5) Hit ratios by x-class" reports on mass, and the hit-ratio of the winner y-combination (conditioned on individual observed x-combinations) as well as the hit-ratio of the runner-up y-combination (conditioned on the same x-combination) with the difference in the frequencies, called Edge1-N, between the two.

Continuing in such a fashion, one may pack multiple variables. Based on what changes have been made in the .hdr file, either partial or all the combinations can be taken forward to the next level of packing. This helps in not only reducing the number of combinations under consideration by user intervention but also to geographically focus onto areas that have category combinations of interest (as shown in Figure 5a). Finally, Figure 6 summarizes the operations.

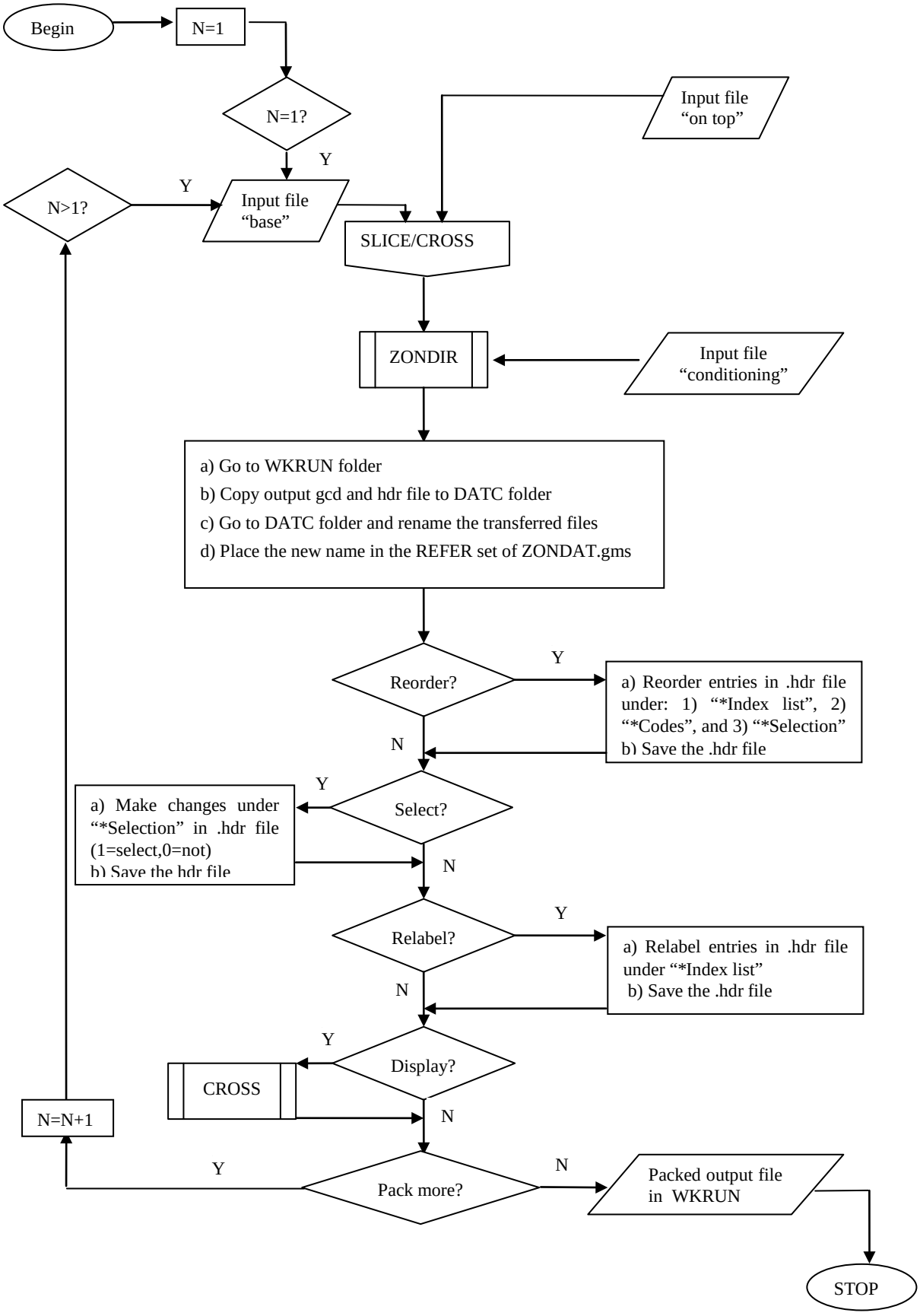

Figure 6. Flowchart overview of the "packing" process.

Alphabetic list of commands

AGGREG
APPLY
ASLICE
CROSS
CROSSOLD
DATHARM
KERNM
KERNSET
MAPTOSUR
MATCH
RECODE
SLICE
ZONDIR
ZONMOL

References

Boom, G.J.M., van den, and S. Pande (2007) 'User manual for the SAS-facility to plot maps'. Amsterdam: Centre for World Food Studies

Bound, J., D. Jaeger, and R. Baker (1993) 'The cure can be worse than the disease: a cautionary tale regarding instrumental variables'. Technical Working Paper 137. Cambridge, MA: National Bureau of Economic Research.

Bound, J., D. Jaeger, and R. Baker (1995) 'Problems with Instrumental Variables estimation when the correlation between the instruments and the endogenous regressors is weak', JASA, 90: 443-50.

Elbers, C., J.O. Lanjouw, P. Lanjouw (2003) 'Micro-level estimation of poverty and inequality', *Econometrica*, Vol. 71, 355-364.

Ermoliev, Yu.M., V.I. Norkin (1997), **'**Stochastic generalized gradient method for nonsmooth and nonconvex problems**'**, *European Journal of Operational Research*, 101:230-244.

Haerdle, W. (1993) *Smoothing techniques, with implementation in S.* Berlin: Springer.

Heckman, J. (2005a) *Evaluating Economic Policy*. Princeton: Princeton University Press.

Heckman, J. (2005b) 'Scientific Model of Causality', *Sociological Methodology*, 35(1), 1-98.

Keyzer, M.A. (2008) Rule-based and support vector (SV-) regression/classification algorithms for joint processing of census, map, survey and district data. Amsterdam, Centre for World Food Studies, Working Paper WP-05-1D.

Keyzer, M.A., and C.F.A. van Wesenbeeck (2005) 'Equilibrium selection in games: the mollifier method'**,** *Journal of Mathematical Economics* 41: 285-301.

Norkin, V.I. and M.A. Keyzer (2009) 'On convergence of kernel learning estimators', *SIAM Journal on Optimization* 20 (3):1205-1223.

Schoelkopf, B., and A. Smola (2002) *Learning with kernels: support vector machines, regularization, optimization and beyond.* Cambridge, Ma.: MIT Press.

Vapnik, V.M. (1998) *Statistical Learning Theory*. New York: Wiley.

The Centre for World Food Studies (Dutch acronym SOW-VU) is a research institute related to the Department of Economics and Econometrics of the Vrije Universiteit Amsterdam. It was established in 1977 and engages in quantitative analyses to support national and international policy formulation in the areas of food, agriculture and development cooperation.

SOW-VU's research is directed towards the theoretical and empirical assessment of the mechanisms which determine food production, food consumption and nutritional status. Its main activities concern the design and application of regional and national models which put special emphasis on the food and agricultural sector. An analysis of the behaviour and options of socio-economic groups, including their response to price and investment policies and to externally induced changes, can contribute to the evaluation of alternative development strategies.

SOW-VU emphasizes the need to collaborate with local researchers and policy makers and to increase their planning capacity.

SOW-VU's research record consists of a series of staff working papers (for mainly internal use), research memoranda (refereed) and research reports (refereed, prepared through team work).

Centre for World Food Studies
SOW-VU
De Boelelaan 1105
1081 HV Amsterdam
The Netherlands

Telephone (31) 20 – 598 9321
Telefax (31) 20 – 598 9325
Email pm@sow.vu.nl
www http://www.sow.vu.nl/