Evaluation of Edit and Imputation Using CMM

Ken Lees, Simon O'Keefe and Jim Austin

February 2003

Abstract

This report describes the evaluation of Correlation Matrix Memory (CMM) for edit and imputation in the EUREDIT project. CMM neural network methods have been studied at University of York for over 15 years, and provide fast, scalable, pattern matching functions for a range of applications. The Advanced Uncertain Reasoning Architecture (AURA) embodies CMM functionality with additional pre- and post-processing methods to support the construction of systems. The AURA class library developed at York is a software implementation of this architecture. In EUREDIT, further software has been developed to provide the additional functions necessary for edit and imputation, while retaining CMM as the core element. Five of the six main datasets available to the EUREDIT project were used in the CMM experiments at York. The CMM-based edit and imputation system developed at York has particular advantages of being very simple to use, fast and scaleable in operation, and offering generally good imputation performance. A new CMM-based error localisation method is introduced. Preliminary results show there is some potential benefit in this method, which is relatively fast, but further development is needed to refine error localisation performance. A major strength of the CMM-based methods is the relative ease of use and minimal skill requirements, particularly for imputation. The CMM based methods are generally very fast in computational terms, despite some weaknesses in the current prototype implementation of file and data handling components. In summary, the CMM-based system provides a fast, highly-automated technique for imputation and error localisation.

Table of Contents

1 Introduction

This chapter describes the results of applying Correlation Matrix Memory (CMM) neural network methods to data editing and imputation in the Euredit project. CMM uses high-performance pattern matching techniques to identify the K nearest-neighbours (K-NN) of a given record, allowing one of several alternative modes to be used in performing the final imputation step. For edit, essentially the same CMM method is used in an edit mode where information about the K-NN for each record is used to determine the "remoteness" (in Euclidean distance terms) of each record from it's neighbouring records. (The assumption is that remote records are more likely to be outliers, or to contain errors.)

This report is concerned primarily with experimental results only. Details of the steps involved in imputation and error-localisation using CMM are available in the Euredit report covering deliverables D4.4.1 and D5.4.1 (Lees et al, 2002). The CMM experiments discussed here were performed using a generic PC running the MS Windows 2000 operating system. The hardware specification comprises: AMD Athlon 1.2 MHz processor, L1 cache of 32 KB and L2 cache of 256 KB, and 512 MB of physical memory (RAM). The edit and imputation application software used was a combination of existing CMM library functions, existing extensions to the CMM software, and specific edit and imputation code developed during the Euredit project.

Of the six main datasets available to the Euredit project, five were used in the York experiments. Only the times series dataset provided by Quantaris was not addressed because the York implementation of E&I was unable to deal with this type of data (however, related work at York is applying CMM techniques to time-series data from a different perspective). The five datasets addressed were DLFS, ABI, SARS, EPE, and GSOEP. Of these, all Y2 versions were used for imputation; ABI and EPE Y3 versions only were used in error localisation (edit) experiments. Due to lack of time imputation was not attempted using Y3 versions of datasets.

The presentation of results here aims to provide a comparative view of the CMM based methods against some of the so-called "standard methods" evaluated in the Euredit project, which include DIS, GEIS, SOLAS, and CANCEIS/SCIA. There is no attempt here to compare CMM based methods against any of the other "non-standard methods" in Euredit. An overall comparison of all methods evaluated in the Euredit project is provided in the User Guide document (Deliverable D6.2 of Euredit).

2 CMM (Correlation Matrix Memory)

2.1 Method Description

CMM is a type of neural network that is "trained" to associate pairs of patterns. Most neural networks require many training cycles through training data, but CMM only requires a single pass through the training data to learn an association. Research at University of York is focused on a special binary version of CMM that uses only binary elements in the input and output patterns, and in the weights stored by the network. Binary CMM can be implemented very efficiently and the result is a very fast, scalable, pattern matching method that can deal with large data sets. Applications typically use the CMM as a kind of filter to remove patterns that do not match closely with the input pattern, so that a conventional (but possibly slow) algorithm can be applied to the relatively small number of remaining patterns. It is important to understand that CMM is trained to represent explicit features of the data, whereas most neural networks are trained to represent implicit features of some assumed model, which to some degree "explains" the data in a training set. This means that it is not necessary to provide a completely "clean" training set with the CMM method since no implicit model assumptions are made.

More formally, CMM can be viewed in terms of a correlation matrix *M*, comprising an array of binary elements, initially set to zero. The matrix is trained according to the values of binary input and output vectors, by computing an outer product between each input vector Q_i and output vector R_i. The result is bitwise-logically ORed with the existing matrix resulting in the following update equation:

$$M^k = M^{k-1} \cup \left(Q_i \times R_i^T\right)$$

Here, M_k is the updated correlation (or weights) matrix after K pairs of input-output patterns have been trained, and R^T is the transpose of column vector *R*. To prevent the memory becoming saturated, the input and output vectors are chosen to have a small, fixed bit density. To find the set of nearest-matching stored patterns, a recall operation is performed. The inner product of the unknown input pattern with the matrix *M* is computed, forming integer-valued elements in an output vector *G*:

$$G_i^T = I_i^T M^k$$

For the work described here, a threshold function known as Willshaw thresholding (Willshaw et al, 1969) is used. Willshaw thresholding operates by setting a '1' in the final binary output when the corresponding element in the output vector *G* contains a value greater than or equal to a threshold value θ determined as the number of bits set to '1' in the input pattern (I_i^T in the second equation) presented during recall, since this represents the maximum response of any element in the output vector *G*. As a result, elements of *G* with values below the threshold are set to zero, and all others are set to one.

In summary, the final (binary) output vector *R* is obtained using:

$$R^T = f_{thresh}\left(G^T\right)$$

where:

$$f_{thresh}\left(g_i\right) = \begin{Bmatrix} 1, \left(g \ \geq \theta\right) \\ 0 \end{Bmatrix}$$

and θ is a positive integer representing the number of bits set to '1' in the input pattern. Further details of both training and recalling with CMM can be found in (Austin, 1997).

The use of CMM for error localisation and imputation involves pre-processing the data to obtain a suitable binary pattern representation. The CMM is used in the "first stage" of processing to find a set of best matches for each data record. This best match set is then used in a "second stage" of processing to calculate Euclidean distances from the record to each match within this neighbourhood. This two-stage approach is used because CMM is highly efficient for finding the approximate neighbourhood set very rapidly, but it uses only an approximation to Euclidean distance. The second stage "conventional calculation" is then used to identify the correct subset of true nearest neighbours using a Euclidean metric.

For imputation, the best match set represents the empirical distribution in a local region in n-space, near to a record having missing values to be imputed. This set is used as the basis of a chosen imputation mode and currently five main modes are available: nearest neighbour, random neighbour, mean, weighted mean, and median. Each mode thus prescribes the method for selecting a suitable value for imputation from the local set. For error localisation, we compute a measure of Euclidean distance for each record to its Kth neighbour (for a suitable preset value of K). Essentially, the larger this distance is, the more likely it is that the record in question is an outlier. We refer to this process as the DKN ("Distance to Kth Neighbour") method.

Experiments were carried out to evaluate both imputation and error localisation (edit) performance. In principle it would have been possible to evaluate a version of the system which combines edit followed by imputation, but practical issues ruled out this possibility. The main problem here is due to the fact that edit rules are not implemented in the York system (because of the limited project timescale) and, in many cases, errors occur as constrained groups of missing values within a single record (e.g. one value is defined to be the sum of other values in the group). It does not seem reasonable to expect meaningful imputation of such variables without employing edit rules to take account of such constraints.

When used in Euredit for edit and imputation applications, CMM can be viewed as a highly flexible type of index system. For a given dataset, we first decide on the size and structure of the CMM according to the amount and types of data concerned. Next we train the CMM by setting certain bits to '1', representing the presence of a particular value or range of values in each record. In this case, the trained CMM represents a mapping from possible data values to each individual record in the data file that contains those values. At this point, we may take another (possibly previously unseen) data record and, using the CMM, determine which other records are similar to this query record. In particular, if this data record has some missing values, the CMM will identify matches using only the non-missing values.

The use of CMM for error localisation involves pre-processing the data to obtain a suitable binary pattern representation for the CMM. After storing the binary pattern representations for all data records in a CMM using "train mode", the CMM is used in "recall mode" to find a set of best matches for each record. This best match set is then used to calculate Euclidean distances from the record to each match within this neighbourhood, leading to a measure of Euclidean distance for each record to its Kth neighbour (for a suitable preset value of K). Essentially, the larger this distance is, the more likely it is that the record in question is an outlier. As mentioned above, we refer to this process as the DKN method.

Similarly, the use of CMM for data imputation involves pre-processing the data to obtain a suitable binary pattern representation for the CMM. After storing the binary pattern representations for all data records in a CMM using "train mode", the CMM is used in "recall mode" to find a set of best matches for each record. This best match set is then used to calculate Euclidean distances from the record to each match within this neighbourhood. From this point, the best match set may be analysed further in a number of ways to select a suitable value to be imputed in place of the missing item. In the experiments described here five main "modes" of selecting a suitable value are considered: nearest-neighbour, random neighbour, median, mean and weighted-mean (all defined over the local K-neighbourhood).

The procedure for CMM-based imputation and error-localisation is as follows:

Step	**Operation**
1	Configure the system with suitable pre-processing from external data to binary patterns
2	Train CMM using a binary pattern representation of every data record
3	For each record P:
3.1	Perform a recall from CMM using binary pattern representation of P as a query
3.2	Find the j best-matches, where j > K (since CMM matching can only approximate Euclidean-distance based matching)
3.3	Compute the K-NN subset from the j best-matches as follows:
3.3.1	Retrieve values from data file for every matching neighbour record
3.3.2	Find Euclidean distance from P to each matching neighbour record
3.3.3	Sort the matching neighbours according to distance to find true K-NN subset
	Then either *LOCALISE ERRORS*:
4	Store DKN – the distance from record P to it's Kth neighbour (value of K is preset – see below) in a table T

5 Determine a threshold (cut-off) distance based on the user-supplied parameter SD. SD is used to help determine the threshold so that a distance greater than the threshold indicates an error, while a distance equal to, or less than the threshold indicates an acceptable record. The error-status of each *value* in an error record is computed, based on its individual contribution to the DKN value. [NOTE: SD specifies a required number of standard deviations in the set of DKN values for the whole dataset, but is used only indirectly in setting the threshold.]

Or *IMPUTE*:

6 Impute a suitable replacement for each missing value in P, using the user-selected mode.

Implementation Issues

CMM forms a central part of AURA (Advanced Uncertain Reasoning Architecture). AURA is a family of techniques developed at York for the construction of high-speed pattern matching systems with a wide range of real-world applications. AURA is currently implemented as a C++ class library for a range of platforms, and implementations using conventional digital hardware have been demonstrated (Austin, Kennedy, 1998).

The AURA software library is now a mature and efficient implementation of CMM though the same is not true of the additional software produced during the project, which is necessarily of prototype quality only. In practice, this means that although fully functional, the system evaluated in the Euredit project does not necessarily use system resources as efficiently as one might expect from production-quality software, in terms of memory use and speed of computation. In addition the level of testing performed was necessarily limited to very basic functional tests. In summary, the core CMM routines are very efficient in speed terms but the overall system developed for Euredit also includes many prototype elements and so does not always reflect this same level of efficiency.

The current implementation of imputation for household variables in SARS assumes that records for a given household are always adjacent in the input data file. Household data also appears in the GSOEP dataset but this is arranged differently, in a form that is not recognised by the current system. It seems unreasonable to invent a new method for dealing with household variables to accommodate the variations in each dataset encountered.

Although the current system at York does not support edit rules directly, an indirect method of including edit rule failure information has been investigated. This currently assumes a particular file format for error data, which exploits the output of a software tool provided to the Euredit project by one of the partners (NAG). Basically, this tool provides a report for each of SARS, ABI and EPE datasets indicating the location of values that violate (so-called) hard edit rules (where there is no doubt that the value is in error). Unfortunately, there was insufficient time to complete this part of the system and so the results reported here do not use any externally derived hard error information.

CMM Background

AURA operation is different from most neural networks (though fundamental similarities remain). The AURA approach involves storing and comparing large numbers of features selected from the data. The core CMM component of AURA is based on a simple one-layer neural network that uses binary weights and Hebbian learning, with origins in the Learning Matrix (Steinbuch, 1961). Fundamentally, the memory associates an input pattern with an output pattern. For computational efficiency, a binary version of CMM is used that has binary weights and inputs. A threshold function known as Willshaw thresholding (Willshaw et al, 1969) is often used to select the best matches in the ‘raw’ CMM outputs. Details of both training and recalling with CMMs can be found in (Austin, 1996).

A CMM neural network (within the AURA framework) is used to find a smaller subset of records containing the K-NN, and then the K-NN rule may be applied to the subset. The subset found by a CMM always contains the required data, together with a few unwanted records (Turner et al, 1997). CMM can also be used to implement a K-nearest neighbour (K-NN) approach to finding outliers for data editing applications. Some related work concerning the implementation of a classifier based on K-NN using AURA technology was described in (Zhou et al, 1999). A detailed description of the implementation of a CMM K-NN classifier is given in (Zhou et al, 1999).

To simplify the mapping of continuous variables into binary patterns, continuous values are quantised using Robust Uniform Encoding (RUE), which is detailed in (Zhou et al, 1999). RUE is optimal in the sense that the number of continuous values assigned to each bin is approximately constant for a given training set.

An earlier smaller-scale project at University of York investigated the use of CMM methods for imputation under contract to Eurostat (contract reference 8223008/SUP-COM) and this is described in a Euredit report (Austin and O'Keefe, 1999).

Additional Developments in EUREDIT

A new method for error localisation termed DKN ("Distance to K^{th} Neighbour") was devised independently at York but it has since been discovered that similar methods have been described by other authors, for example (Byers et al, 1996) and particularly (Ramaswamy et al, 2000). The DKN method involves analysis of the *density* of data in the local neighbourhood of each point, by calculating a suitable distance metric from each point to its nearest neighbours, in particular noting the distance to the K^{th} neighbour. The rationale for this approach is based on the observation that points in a sparse neighbourhood will tend to have a relatively large distance to the Kth neighbour, so long as the choice of K is not too large. Using this approach it is possible to identify outlying data points in terms of their remoteness from 'normal' data points. The DKN method is still at the development stage, and details of how best to assign a probabilistic score to values considered potentially in error are still being investigated. Nevertheless, some results have been produced using CMM with the DKN method although these are probably sub-optimal.

The system currently provides five main imputation "modes" each of which operates by selecting a suitable value from the set of K-NN neighbourhood records, as determined by the CMM and subsequent Euclidean distance calculations.

- Nearest-neighbour mode simply replaces each missing value using the corresponding value copied from the neighbour that is nearest in terms of Euclidean distance.
- Random-neighbour mode simply replaces each missing value using the corresponding value copied from a neighbour selected at random in the K-NN neighbourhood.
- Mean mode replaces each missing value using the corresponding mean value computed over the K-NN neighbourhood.
- Weighted-mean mode replaces each missing value using the corresponding value of the Euclidean distance weighted-mean, computed over the K-NN neighbourhood. This gives greater weight to values belonging to closer neighbours.
- Median mode replaces each missing value using the corresponding median value computed over the K-NN neighbourhood.

A special method was devised to ensure household variables were imputed properly for the SARS dataset. These variables are imputed in such a way as to maintain the same value for each member of the household as follows:

- "parent" and "dependents" relationships are defined in the specification file
- store the first record in each household and impute any missing values
- additional records from same household inherit household values from the first record

The current implementation of this method requires that records for a given household are always adjacent in the input data file.

A modified calculation of distance was introduced as an option by the setting of a flag in the specification file. The modified calculation is appropriate when a mix of continuous and categorical variables is present in each data record, and treats all categorical variables as a single "pseudo variable" whose value ranges between 0.0 when every pair of categorical variables between the two records matches, and 1.0 when all categorical variables between the two records are different. Indications suggest that this modified distance gives a better ranking of near-neighbours than the standard distance calculation resulting in better overall system performance.

For all Euredit datasets, the CMM-based edit and imputation process is essentially the same. Where available, metadata is used to describe each variable in the dataset in terms of type and, in the case of categorical and ordinal types, the possible values allowed for that variable. When metadata is not available, another application (e.g. a spreadsheet) can be used to quickly examine the range of possible values for each variable. The description of each variable plus a small number of other dataset parameters and files names is stored (using a particular text format) in a specification file. The flags and values entered in this file control the whole edit or imputation process.

The DKN method used for error localisation can be illustrated as follows:

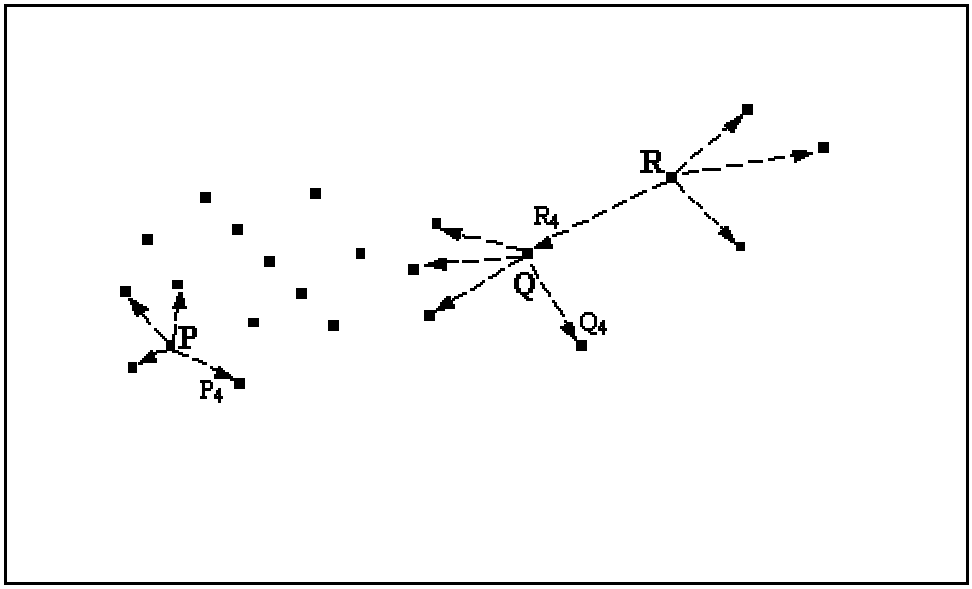


Figure 5.1 Example illustrating the DKN method of error localisation

The figure above shows a simple example of the DKN approach applied to a set of points in a 2D plane. In this example, the value of K is set at 4. In other words, we are interested in finding the distance to the 4th nearest neighbour of each point. For simplicity, we choose just three points P, Q, and R, to illustrate the approach.

Point P has its 4th nearest neighbour at distance P4, point Q has its 4th nearest neighbour at distance Q4, while point R has its 4th nearest neighbour at distance R4. Choosing a standard Euclidean metric would

result in the ranking R4 > Q4 > P4. The inference that could be drawn from this is that point R is more likely to be in a sparse region of the data space and, ultimately, more likely to be a remote or outlying data point. Although the choice of K is clearly important, in general, it is not critical. In this example, if K=3 had been chosen, the approach would not identify point R so clearly as an outlier, since the 3rd neighbour of point R is probably at a distance similar to that of the 3rd neighbour of many other points in the figure. However, choosing K=5 would (as with K=4) result in a decision that R is a more likely outlier candidate.

Setting up the system and usability issues

In working towards the objective of automatic edit and imputation, the York system has been developed as much as possible with the aim of minimising effort, skill, and knowledge on the part of an end-user, with all the important system parameters having suitable default values. This is seen as a particular strength of the York system.

The Euredit experiments at York use a combination of an existing CMM software library and some specially developed software, to provide a prototype imputation and error localisation application. This application is a simple command-line program. This command line program is controlled by a number of settings in a simple text file. Preparation for a new dataset involves using a simple Java-based GUI to set the values in the text file appropriate to the new dataset. The program analyses the specified input data file containing errors and missing values, including basic checks for range errors against the acceptable values defined in the text file. The program produces either an output file with missing values imputed (in imputation mode) or a file where detected error values have been replaced by missing values for imputation (in error localisation mode).

No particular data analysis is required, but certain basic dataset parameters must be set, as follows:

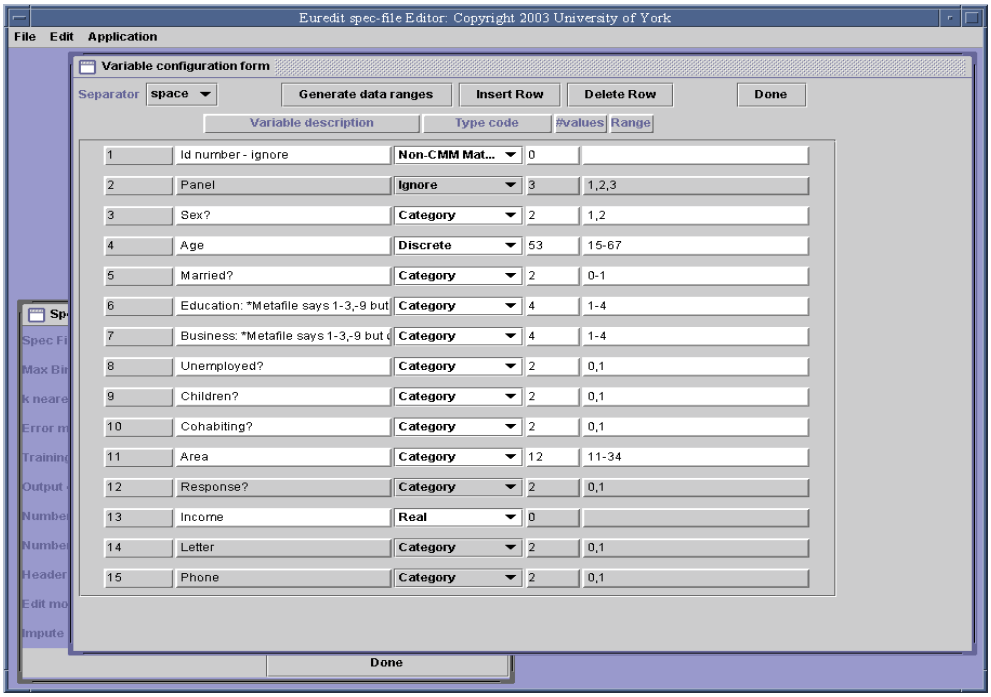
- choice of *either* error localisation or imputation processing (but not both in a single task)
- name of input data file containing errors and/or missing values
- name of output data file to contain imputed version of input file, or errors replaced by missing values
- total number of variables present in the dataset
- total number of records (i.e. physical text lines) in the dataset
- whether a header line is present in the data file
- identification of the type for each variable (continuous, discrete ordinal, categorical, text, or "ignore")

Some other values can be changed by the user (if required), but default values are determined by the system, which normally provide good performance. For imputation, a choice of donor and modelling modes is also available to the user, but normally only two of these modes are recommended for best results: weighted mean or median. The median mode is recommended when greater robustness against the effect of outliers is desired.

For error localisation, some judgement is needed when setting an additional parameter K. The value of K represents the number of neighbouring data points to be considered by the system when identifying errors. The best choice for K should be a value greater than the size of the largest error cluster and less than size of the smallest non-error cluster (assuming the data is clustered to some extent). Information about such cluster sizes is rarely available in practice, but fortunately the precise value is not critical. Generally the value of K will be small, and values of 10 and 20 have been found to give reasonable results with Euredit datasets.

These preparation steps can be achieved fairly quickly but the actual time will depend on features of the dataset (particularly the number of variables). In Euredit, the preparation time was typically found to be 30 to

40 minutes. It would naturally take more time for a new user on the first few attempts, but special knowledge is not required. The screenshot below shows the GUI used for setting the dataset parameters described above.



2.2 Evaluation

This section describes the experimental details and results for each dataset used. Comparative results are provided wherever possible, for the CMM experiments and the so-called "standard methods" evaluated in Euredit. The remainder of this section is organised by considering results for each dataset, in turn, followed by an assessment of the strengths and weaknesses of the CMM approach.

General note on timings: some tables show times for 'set up' and 'run time' although the basis on which these times is derived is not completely objective, particularly for the 'set up' phase, where there is no consensus (in the Euredit project) on what activities this should include. 'Set up' times for CMM always assume a "cold start", in other words given a new dataset, the time needed to prepare the text file which controls the CMM software for the imputation or error localisation process, using system default settings for most parameters (see previous section). It is emphasised that detailed or statistical analysis of the data is not necessary. For CMM all 'run time' figures are for complete end-to-end processing, including training of the neural network.

General note on settings: for all imputation experiments the value of K is selected automatically by the system, using an empirically determined heuristic equal to 0.6 of the square root of the number of records in the dataset, rounded to the nearest integer. Since the overall 'run time' increases according to the number of neighbours selected and the number of records requiring imputation, an upper limit of 50 was imposed on the value of K for the (larger) SARS dataset.

Dataset: ABI Y3

Technical Summary

Method:	*CMM (plus DKN method)*
Training data set:	*sec198(y3).csv*
Hardware used:	*MS Windows 2000, PC "generic" (unbranded), AMD Athlon 1.2 MHz, L1 cache: 32 KB; L2 cache 256 KB, Amount of system physical memory in MB: 512 MB*
Additional software used:	*none*
Test scope:	*Editing (error localisation) only.*

Edit Criteria

As described previously, outliers are identified in terms of the size of a metric (DKN), and larger values for DKN are considered more likely to indicate outlyingness. Error localisation criteria are specified in terms of two parameters: K and SD. K specifies the number of the neighbour whose distance is calculated in the CMM (DKN) procedure. SD serves the function of a threshold and is specified in terms of a number of standard deviations in the distance measures calculated for each record to it's Kth neighbour.

All variables were considered as potentially in error except for simple record identifiers. The error localisation approach used was the CMM (DKN) method as described in section 2.1. No logical checks were performed other than simple range checks where range information was available. No edit rules were used. Of the 6,233 records in the input file, experiment YA30004 rejected 695 records comprising a total of 9,633 rejected values. Experiment YA30008 rejected 712 records comprising a total of 10,128 rejected values.

Edit Training

Essentially, training for edit (error localisation) is exactly the same as for imputation. The difference in processing occurs later, when the trained network is used to identify similar records.

Results

Table 5.1 below summarises the experimental results reported in this section, where indicative timings are shown.

Table 5.1 Experiment reference information for ABI Y3 data

Experiment	Method	Setup (minutes)	Run Time (s)	Other (s)	Total Time (s)
YA30004	CMM DKN (1) (K=10, SD = – 0.1)	30	33	30	1863
YA30008	CMM DKN (2) (K=20, SD = – 0.1)	30	33	30	1863
IA30001	GEIS	120	22006		29206

Figures 5.2 to 5.8 show the performance for two versions of the CMM DKN method (with different values of the parameter K), and the GEIS method. Results are shown in terms of case-level, and alpha and beta measures for each of the variables of primary interest: TURNOVER, EMPTOTC, PURTOT, TAXTOT, ASSACQ, and ASSDISP.

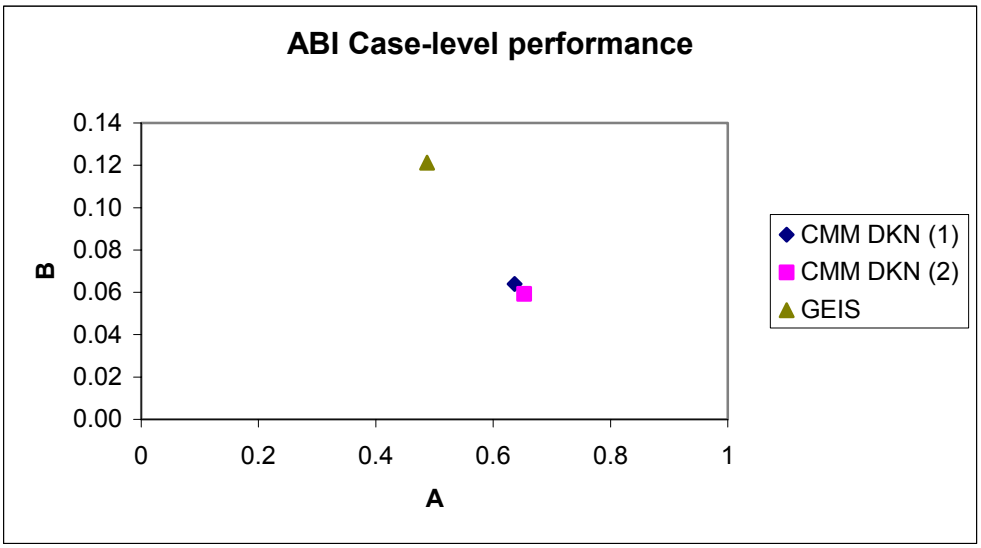


Figure 5.2 Case-level performance for ABI Y3 data

Figure 5.2 shows case-level performance, with GEIS slightly better in terms of measure A (failure to identify records with errors), and the two CMM experiments slightly better in terms of measure B (wrongly classifying correct records as having errors). Actual case-level values (rounded) are shown in Table 5.2 below.

Table 5.2 Case-level results for ABI Y3 data

	CMM DKN (1)	CMM DKN (2)	GEIS
G	0.021	0.021	0.022
A	0.873	0.871	0.487
B	0.106	0.109	0.121
C	0.320	0.322	0.224

CMM results for these two experiments are very similar in case level performance. All three methods are similar in terms of G, the Gini index, although just slightly better for the CMM experiments. The CMM experiments are better than GEIS in terms of B, whereas GEIS is better in terms of A and C.

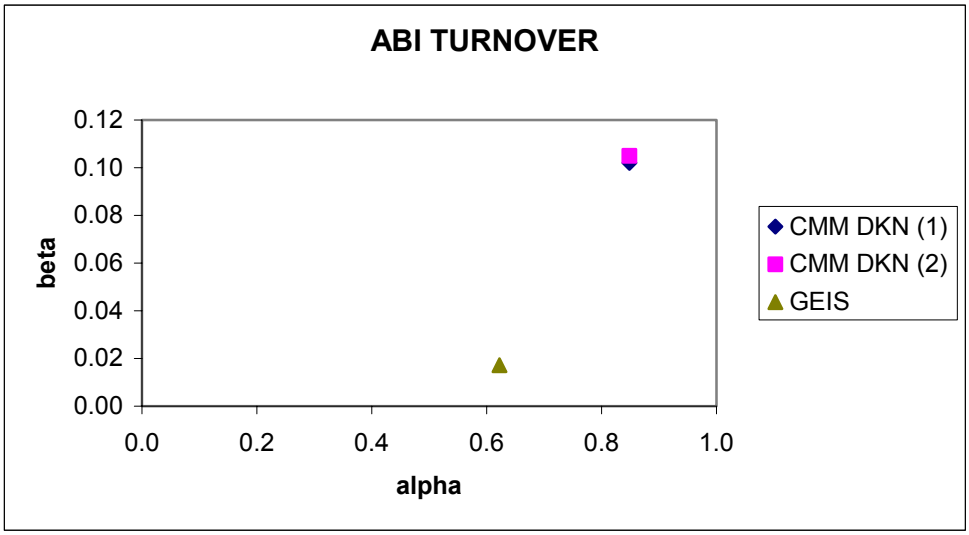


Figure 5.3 alpha-beta performance for ABI Y3 data: TURNOVER

In Figure 5.3 GEIS achieves rather smaller values for beta and slightly smaller alpha, compared with CMM. Table 5.3 confirms that GEIS performs better than CMM for this variable in terms of alpha, beta, and delta. However CMM seems better in terms of all the other performance measures in Table 5.3.

Table 5.3 Results for ABI Y3 data: TURNOVER

	CMM DKN (1)	CMM DKN (2)	GEIS
alpha	0.849	0.849	0.622
beta	0.102	0.105	0.017
delta	0.166	0.169	0.069
RAE	6.806	7.061	21.143
RRASE	0.178	0.301	11.043
RER	9,696	51,777	2,005,507
tj	8.63	8.43	157.41
AREm1	7.07	7.36	22.50
AREm2	1.8	7.1	10,480.0

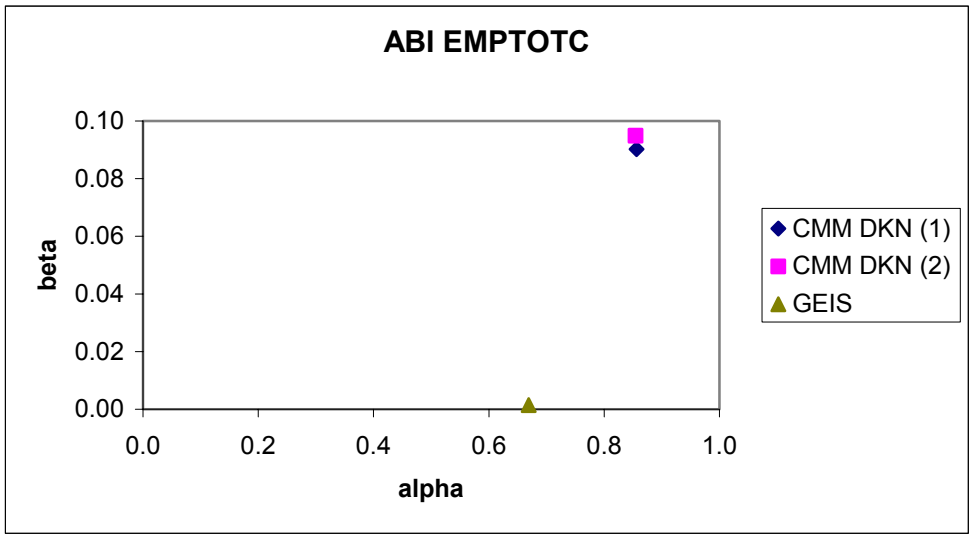


Figure 5.4 alpha-beta performance for ABI Y3 data: EMPTOTC

Figure 5.4 shows alpha-beta performance for variable EMPTOTC. The standard method GEIS has a beta score close to zero and is also a little better than CMM in terms of alpha. Table 5.4 gives more detail on the results. GEIS is better for alpha, beta and delta scores but the CMM experiments are better for all other measures.

Table 5.4 Results for ABI Y3 data: EMPTOTC

	CMM DKN (1)	CMM DKN (2)	GEIS
alpha	0.856	0.855	0.669
beta	0.090	0.095	0.001
delta	0.164	0.168	0.065
RAE	7.71	7.77	36.69
RRASE	0.20	0.24	21.00
RER	3,783	21,182	3,026,855
tj	8.14	8.13	294.42
AREm1	7.94	8.05	38.12
AREm2	3.17	5.00	43,572.02

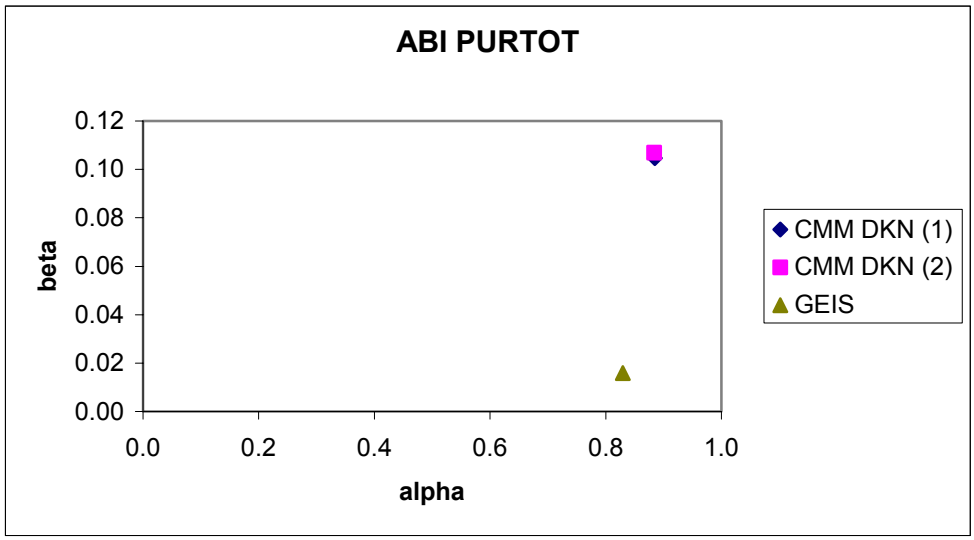


Figure 5.5 alpha-beta performance for ABI Y3 data: PURTOT

Figure 5.5 shows alpha-beta performance for variable PURTOT. The standard method GEIS has a better beta score than CMM experiments and is slightly better than CMM in terms of alpha. Table 5.5 gives more detail on these results. GEIS is better for alpha, beta and delta scores but the CMM experiments are better for all other measures.

Table 5.5 Results for ABI Y3 data: PURTOT

	CMM DKN (1)	CMM DKN (2)	GEIS
alpha	0.885	0.884	0.830
beta	0.105	0.107	0.016
delta	0.221	0.222	0.137
RAE	9.96	10.25	23.81
RRASE	0.25	0.37	9.21
RER	12,654	61,459	1,686,835
tj	7.39	7.41	171.68
AREm1	10.54	10.86	24.91
AREm2	4.40	10.35	6,724.03

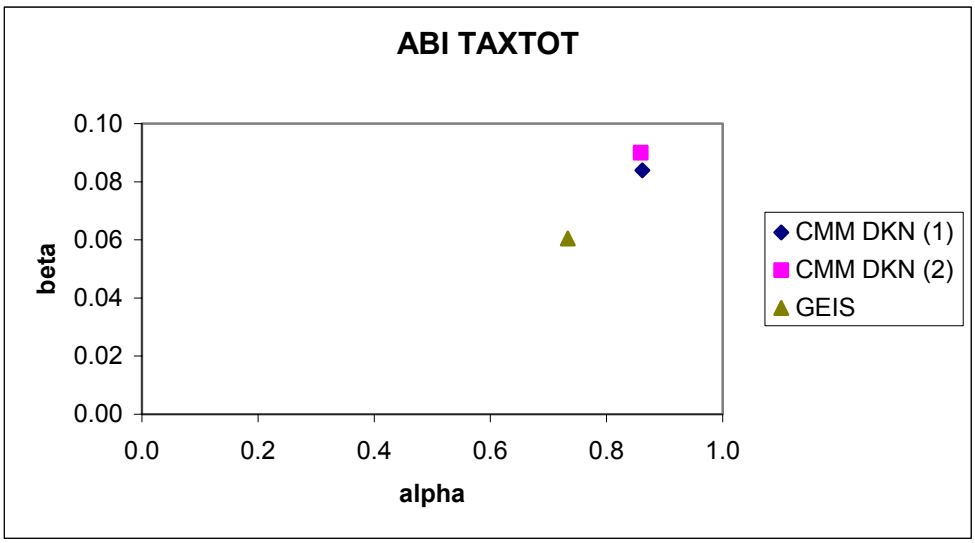


Figure 5.6 alpha-beta performance for ABI Y3 data: TAXTOT

Figure 5.6 shows alpha-beta performance for variable TAXTOT. The standard method GEIS is slightly better than the CMM results in both alpha and beta scores. Table 5.6 gives more detail on these results. GEIS is better for alpha, beta and delta scores but the CMM experiments are better for all other measures.

Table 5.6 Results for ABI Y3 data: TAXTOT

	CMM DKN (1)	CMM DKN (2)	GEIS
alpha	0.862	0.859	0.733
beta	0.084	0.090	0.060
delta	0.176	0.181	0.141
RAE	19.150	19.269	28.751
RRASE	0.938	0.945	10.625
RER	150,546	150,546	1,894,894
tj	6.3	6.4	272.3
AREm1	20.3	20.5	32.6
AREm2	125	127	16,919

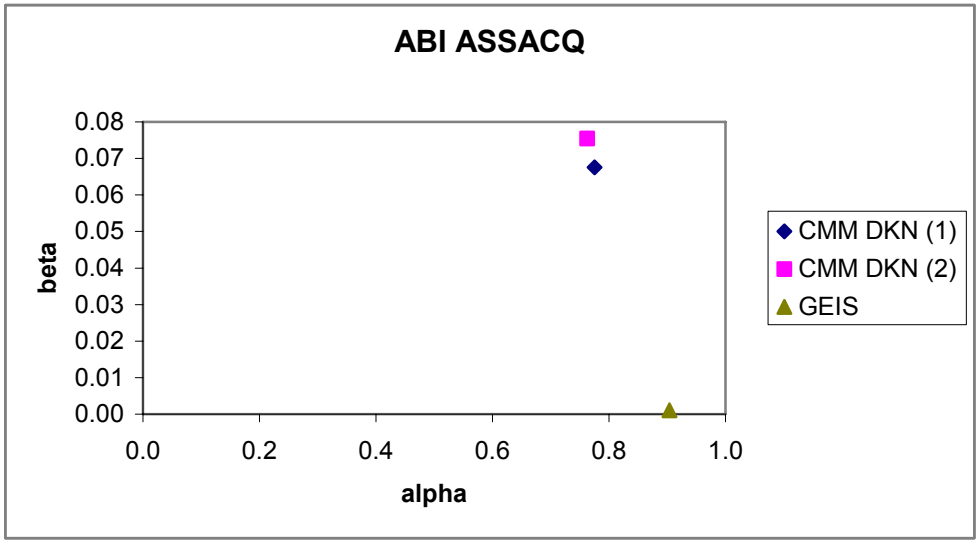


Figure 5.7 alpha-beta performance for ABI Y3 data: ASSACQ

Figure 5.7 shows alpha-beta performance for variable ASSACQ. The standard method GEIS is close to zero for beta, but the CMM results are slightly better in alpha scores. Table 5.7 gives more detail on these results. GEIS is better for beta and delta scores but the CMM experiments are better for alpha and all other measures.

Table 5.7 Results for ABI Y3 data: ASSACQ

	CMM DKN (1)	CMM DKN (2)	GEIS
alpha	0.776	0.763	0.904
beta	0.068	0.075	0.001
delta	0.102	0.110	0.069
RAE	10.322	10.360	39.60
RRASE	0.549	0.552	19.53
RER	20,876	20,876	5,374,471
tj	2.60	2.61	201.41
AREm1	10.97	11.07	40.32
AREm2	14.13	14.39	15,059.93

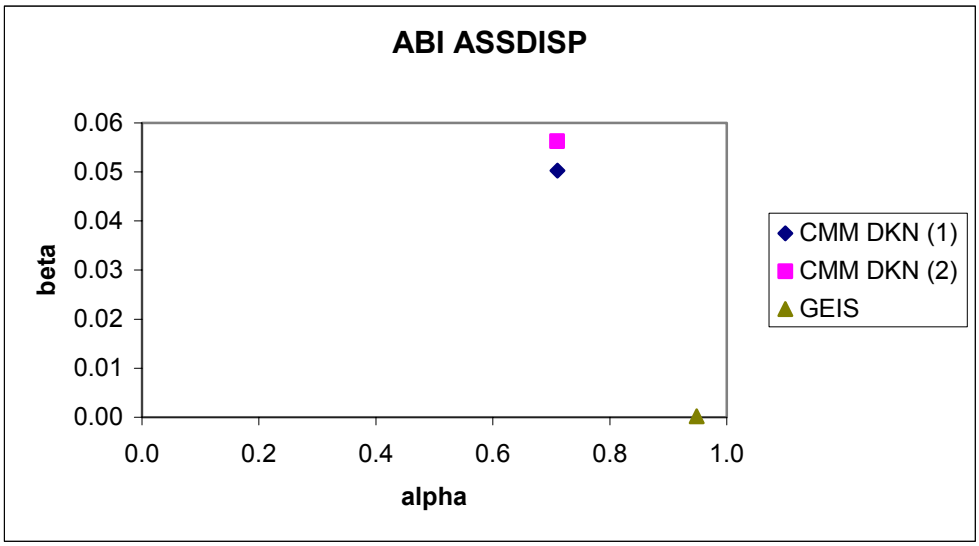


Figure 5.8 alpha-beta performance for ABI Y3 data: ASSDISP

Figure 5.8 shows alpha-beta performance for the variable ASSDISP. In this case, the CMM-based experiments perform better than GEIS in terms of the alpha score, although the GEIS score is virtually zero for beta and is somewhat better than the CMM score. Table 5.8 provides more detail, where it can be seen that for delta, GEIS also scores slightly better than the CMM experiments. For all other measures CMM experiments have much better scores.

Table 5.8 Results for ABI Y3 data: ASSDISP

	CMM DKN (1)	CMM DKN (2)	GEIS
alpha	0.710	0.710	0.949
beta	0.050	0.056	0.000
delta	0.070	0.077	0.057
RAE	17.19	24.44	105.27
RRASE	1.87	2.84	38.91
RER	213,458	259,744	1,063,677
tj	2.4	2.4	112.8
AREm1	18.30	26.35	106.51
AREm2	55.94	130.24	2,560.33

In summary the results for TURNOVER and EMPTOTC (Figures 5.3 and 5.4) show CMM is performing less well than the other methods for alpha, beta and delta scores, but better for the other criteria. CMM performs relatively better for ASSDISP, ASSACQ, also to a lesser extent for TAXTOT and PURTOT. The performance of the two CMM experiments is similar in most cases, tending to confirm a trend observed during development that the performance of the DKN method is relatively insensitive to minor changes in the parameter K (the number of neighbours).

Dataset: ABI Y2

Technical Summary

Method:	*CMM*
Training data set:	*sec198(y2).csv*
Hardware used:	*MS Windows 2000, PC "generic" (unbranded), AMD Athlon 1.2 MHz, L1 cache: 32 KB; L2 cache 256 KB, Amount of system physical memory in MB: 512 MB*
Additional software used:	*none*
Test scope:	*Imputation only*

Imputation

The CMM-based imputation method implemented in Euredit follows the simple strategy of attempting to impute all missing values found in a file. The results reported in this section are based on the Y2 version of the dataset, which contains missing values but not errors.

Of the five CMM imputation "modes" available, the two overall best performing experiments described here use median (shown as MDN in graphs) and weighted mean (shown as MDN in graphs). The median method of imputation determines the local median value by considering only the K neighbouring records of the record with a missing value, as described in section 2.1. This value is used as a donor value. This may be regarded as a donor method of imputation. The value of K is selected automatically by the system, as a function of the number of records in the dataset. The weighted mean method of imputation estimates a Euclidean distance-weighted mean value to be imputed by considering only the K neighbouring records of the record with a missing value, as described in section 2.1. This may be regarded as a modelling method of imputation, where the local Euclidean-distance weighted mean is computed over the local neighbourhood.

Results

Table 5.9 below shows reference details for experiments discussed in this section and compares times for the systems that produced these results.

Table 5.9: Experiment reference information for ABI Y2 data

Experiment	Method	Setup (minutes)	Run Time (s)	Other (s)	Total Time (s)
YA20005	CMM median	30	44		1844
YA20004	CMM weighted mean	30	48		1848
OA20001	DIS	120	60	60	7320
IA20001	GEIS	100	3060		9060

Table 5.10 Results for ABI Y2 data: TURNOVER

	CMM WMN	CMM MDN	GEIS (1)	DIS
Slope	1.14	1.21	1.12	0.40
t-val	3,944	85	3,281	-15,878
mse	3.48E+08	239.00E+08	0.16E+08	5.53E+08
R^2	0.99	0.73	1.00	0.98
dL1	256	666	161	1,114
dL2	5,185	27,307	3,339	47,507
dLinf	9,509	64,527	7,829	113,389
K-S	0.067	0.097	0.058	0.141
K-S_1	0.00025	0.00009	0.00029	0.00013
K-S_2	0	0.000001	0	0
m_1	103	594	62	860
m_2	2.55E+08	9.97E+08	2.00E+08	52.40E+08
MSE	19,005	19,047	19,010	20,149
AREm1	0.0018	0.0018	0.0018	0.0018
AREm2	0.014	0.014	0.014	0.014

Table 5.10 shows imputation results for the variable TURNOVER. The standard method GEIS appears to perform best overall for this variable, but is perhaps only slightly better than CMM weighted mean. Both CMM techniques seem to perform better than DIS.

Table 5.11 Results for ABI Y2 data: EMPTOTC

	CMM WMN	CMM MDN	GEIS (1)	DIS
Slope	0.81	1.13	1.00	0.89
t-val	-305.7	170.8	-1.4	-27.0
mse	169,144	165,673	78,667	255,096
R^2	0.963	0.965	0.982	0.966
dL1	18.6	16.8	16.0	26.1
dL2	120	68	47	67
dLinf	162	147	122	216
K-S	0.221	0.182	0.144	0.123
K-S_1	0.00064	0.00082	0.00024	0.00055
K-S_2	0.000019	0.000034	0.000007	0.000021
m_1	0.71	10.25	0.21	5.24
m_2	73,814	18,289	3,691	18,435
MSE	201.7	201.6	201.1	201.5
AREm1	0.0125	0.0125	0.0125	0.0125
AREm2	0.02	0.02	0.02	0.02

Table 5.11 shows imputation results for the variable EMPTOTC. Here again, the standard method GEIS appears to perform best overall for this variable. DIS and CMM median are fairly similar in performance. CMM weighted mean shows least good performance of the four experiments considered for EMPTOTC, but is not significantly different from the other results.

Table 5.12 Results for ABI Y2 data: PURTOT

	CMM WMN	CMM MDN	GEIS (1)	DIS
Slope	1.23	1.34		0.97
t-val	105	113		-259
mse	5.83E+08	9.22E+08		1.28E+08
R^2	0.913	0.903		0.978
dL1	158.8	182.4	5.7	70.5
dL2	4578	5636	32	1648
dLinf	9,338	11,117	199	3,648
K-S	0.115	0.116	0.028	0.082
K-S_1	0.00019	0.00023	0.00001	0.00021
K-S_2	0.000001	0.000003	0	0.000001
m_1	96.00	148.19	1.27	19.34
m_2	52,757,076	61,113,941	362,708	18,408,069
MSE	10,567	10,569	10,354	10,553
AREm1	0.012	0.012	0.012	0.012
AREm2	0.025	0.025	0.025	0.025

Table 5.12 shows imputation results for the variable PURTOT. Here, some key measures for the standard method GEIS are not reported by the evaluation system. However, for the measures actually reported, GEIS appears to perform best overall for this variable by a significant margin. DIS is generally slightly better than the CMM imputation methods. For CMM, the weighted mean experiment performs slightly better than the median.

Table 5.13 Results for ABI Y2 data: TAXTOT

	CMM WMN	CMM MDN	GEIS (1)	DIS
Slope	0.74	1.04	0.82	0.51
t-val	-243	9	-41	-1736
mse	39,180	22,010	20,826	28,766
R^2	0.908	0.936	0.946	0.900
dL1	4.8	4.1	5.3	8.0
dL2	28	27	21	97
dLinf	65	45	53	191
K-S	0.189	0.081	0.130	0.096
K-S_1	0.000569	0.00052	0.000419	0.000413
K-S_2	0.000033	0.000007	0.000019	0.000007
m_1	1.39	1.40	1.22	3.54
m_2	90	2,844	190	23,916
MSE	2.999	2.994	2.995	3.010
AREm1	0.0102	0.0102	0.0102	0.0102
AREm2	0.0198	0.0198	0.0198	0.0198

Table 5.13 shows imputation results for the variable TAXTOT. CMM median and GEIS tend to have best performance overall and are comparable. CMM weighted mean and DIS perform slightly worse but none is particularly bad and their performance is often complementary.

Table 5.14 Results for ABI Y2 data: ASSACQ

	CMM WMN	CMM MDN	GEIS (1)	DIS
Slope	1.91	13.63	1.25	1.25
t-val	12,900	28,168	22,560	11,988
mse	19,241,367	1.01E+09	3,068,957	41,250,330
R^2	0.995	0.812	0.999	0.988
dL1	206	157	46	66
dL2	6,152	8,050	1,685	2,008
dLinf	15,956	23,297	4,773	5,261
K-S	0.235	0.352	0.194	0.103
K-S_1	0.000214	0.000072	0.000163	0.000231
K-S_2	0.000001	0.000001	0	0
m_1	189.7	155.0	33.3	29.1
m_2	1.18E+08	68,412,153	24,563,481	31,254,264
MSE	43.90	57.23	44.53	44.25
AREm1	0.106	0.106	0.106	0.106
AREm2	0.286	0.286	0.286	0.286

Table 5.14 shows imputation results for the variable ASSACQ. Here GEIS performs best overall, but all methods perform well for this variable, and the variation in results does not seem large in most cases.

Table 5.15 Results for ABI Y2 data: ASSDISP

	CMM WMN	CMM MDN	GEIS (1)	DIS
Slope	1.76			
t-val	26.51			
mse	520,682			
R^2	0.922			
dL1	30.68	5.21	67.70	5.42
dL2	318	165	3,998	128
dLinf	708	377	9,877	273
K-S	0.197	0.214	0.058	0.058
K-S_1	0.00130	0.00022	0.00001	0.00024
K-S_2	0.00003	0.000016	0	0.000001
m_1	13.6	5.1	64.0	2.8
m_2	206,406	29,440	17,231,916	61,616
MSE	3.1	3.1	8.6	3.1
AREm1	0.0047	0.0047	0.0047	0.0047
AREm2	0.024	0.024	0.024	0.024

Table 5.15 shows imputation results for the variable ASSDISP. The evaluation measures for three of the experiments have not been fully reported by the evaluation software. However, for the measures reported DIS and CMM median seem generally similar and are best overall. CMM weighted mean is better than GEIS for dL1, dL2, dLinf, m_1, m_2, and MSE measures, but GEIS is better for K-S, K-S_1, and K-S_2.

In summary, the standard method GEIS tends to perform better than the two CMM experiments for this dataset, although CMM is sometimes better than DIS. It is generally difficult to judge which methods are performing best, because they are often complimentary. The CMM methods seem easier and faster to set up, and provide faster processing of the data.

Dataset: EPE Y3

Technical Summary

Method:	*CMM (in "edit" mode using DKN method)*
Training data set:	*epe93na(y3).csv*
Hardware used:	*MS Windows 2000, PC "generic" (unbranded), AMD Athlon 1.2 MHz, L1 cache: 32 KB; L2 cache 256 KB, Amount of system physical memory: 512 MB*
Additional software used:	*none*
Test scope:	*Editing (error localisation) only.*

Edit Criteria

All variables were considered as potentially in error except for the first value in each record, which is the record identifier. The error localisation approach used was the CMM(DKN) method as described in section 2.1. Error localisation criteria are specified in terms of two parameters: K and SD. K specifies the number of the neighbour whose distance is calculated in the CMM(DKN) procedure. SD serves the function of a threshold and is specified in terms of a number of standard deviations in the distance measures calculated for each record to it's K^{th} neighbour. As described previously, outlier errors are identified in terms of the size of a metric (DKN), and larger values for DKN are considered more likely to indicate outlyingness. The method is capable of dealing with small as well as large outliers, since these are differentiated only in terms of degree of outlyingness by the method.

Results

EPE Y3 Data Edit (Error Localisation)

Table 5.16 below summarises the experimental results reported for this dataset. Unfortunately, comparative results for standard methods are not available for this dataset, so only CMM experiments are reported. Once again, indicative timings are shown, but the set up times are not completely objective, as there is no consensus on what activities this should include. For CMM, all run times are for complete end-to-end processing, including training of the neural network. Set up times for CMM always assume a "cold start". In other words given a new dataset, the time needed to prepare the text file which controls the error localisation process using system default settings for most parameters.

Table 5.16 Experiment reference information for EPE Y3 data

Experiment	Method	Setup (minutes)	Run Time (s)	Other (s)	Total Time (s)
YE30004	CMM DKN (1): DKN only (K=10, SD = – 0.1)	35	15	15	2130
YE30008	CMM DKN (2): DKN only (K=20, SD = – 0.1)	35	29	15	2144

Figures 5.9 to 5.13 show case-level and alpha-beta performance for two CMM DKN experiments with different values of the parameter K. Results are shown for each of the variables of primary interest: TOTINVTO, TOTEXPTO, SUBTOT, and RECTOT. Tables 5.17 to 5.21 provide more detailed results for these experiments.

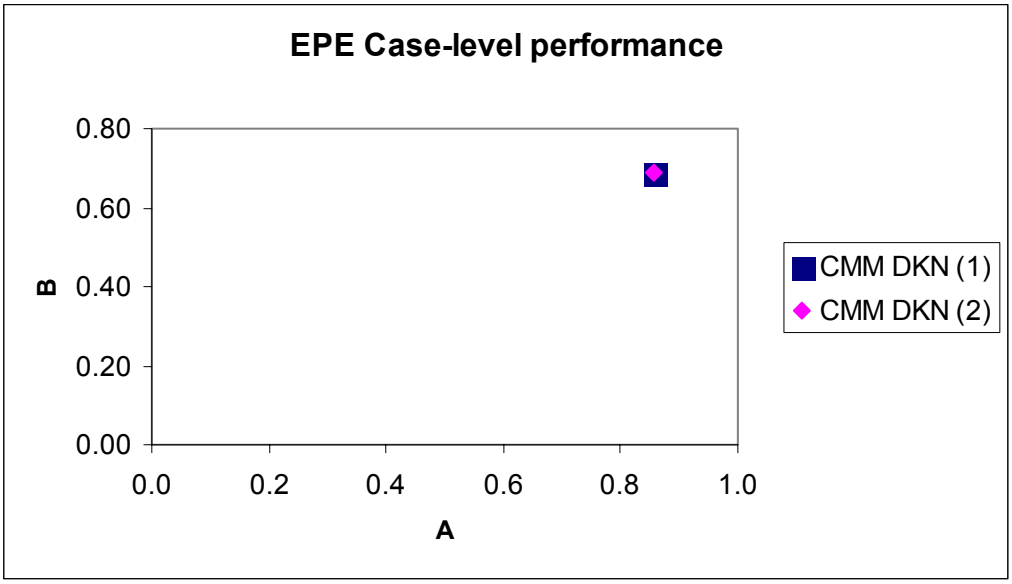


Figure 5.9 Case-level performance for EPE Y3 data

Figure 5.9 shows case-level performance. The two CMM experiments are virtually identical in case-level performance and the actual values (rounded) are shown in Table 5.17 below.

Table 5.17 Case-level results for EPE Y3 data

	CMM DKN (1)	CMM DKN (2)
G	0.017	0.017
A	0.855	0.855
B	0.687	0.687
C	0.793	0.793

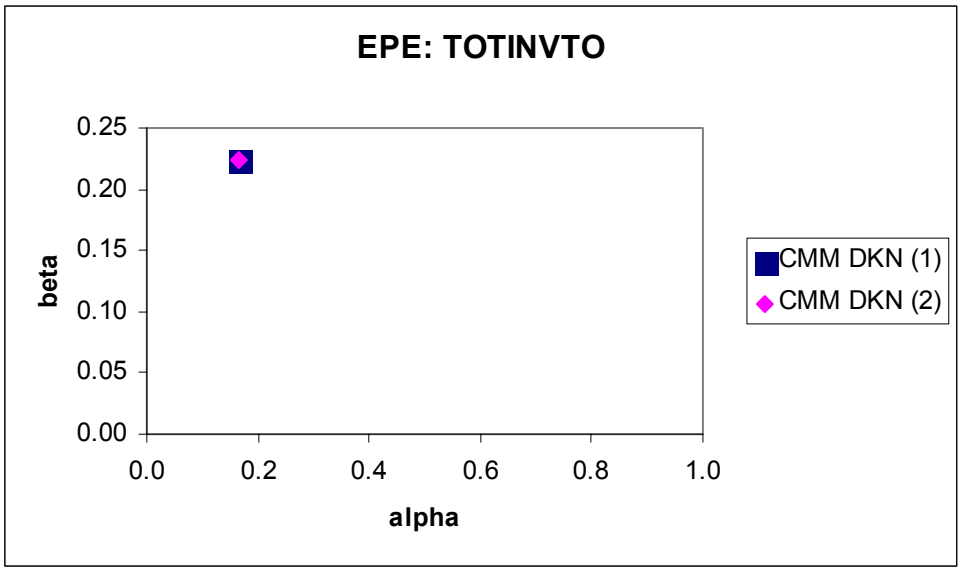


Figure 5.10 alpha-beta performance for EPE Y3 data: TOTINVTO

In Figure 5.10 the two CMM experiments have identical performance for the variable TOTINVTO, and this is confirmed in Table 5.18.

Table 5.18 Results for EPE Y3 data: TOTINVTO

	CMM DKN (1)	CMM DKN (2)
alpha	0.167	0.167
beta	0.223	0.224
delta	0.222	0.223
RAE	0.004	0.004
RRASE	0.003	0.003
RER	27	27
tj	1.12	1.12
AREm1	0.525	0.526
AREm2	0.869	0.869

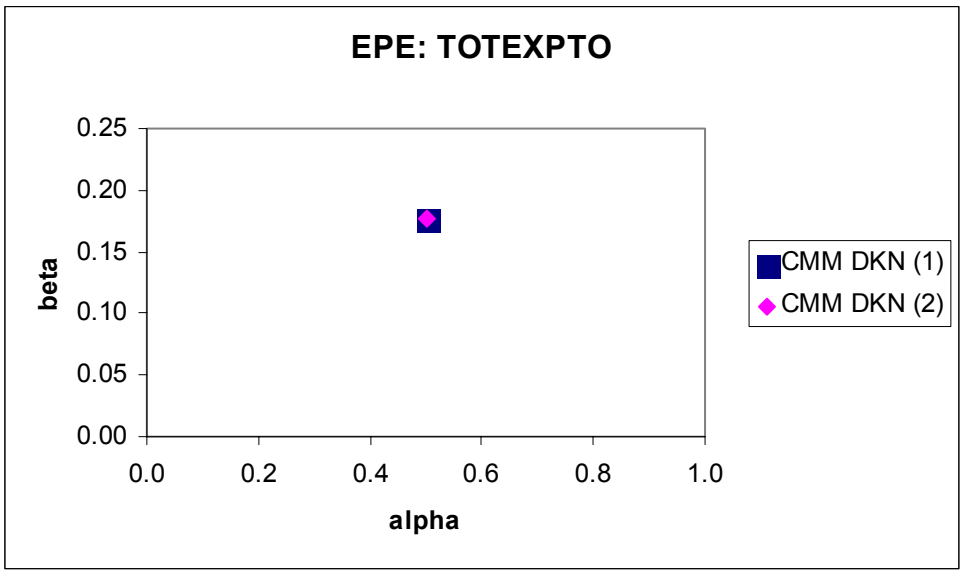


Figure 5.11 alpha-beta performance for EPE Y3 data: TOTEXPTO

In Figure 5.11 the two CMM experiments have identical performance for the variable TOTEXPTO, and this is confirmed in Table 5.19 with only slight differences in AREm1 and AREm2.

Table 5.19 Results for EPE Y3 data: TOTEXPTO

	CMM DKN (1)	CMM DKN (2)
alpha	0.5	0.5
beta	0.176	0.176
delta	0.182	0.182
RAE	-0.0047	-0.0047
RRASE	0.0012	0.0012
RER	3.545	3.545
tj	-1.062	-1.062
AREm1	0.324	0.335
AREm2	0.553	0.562

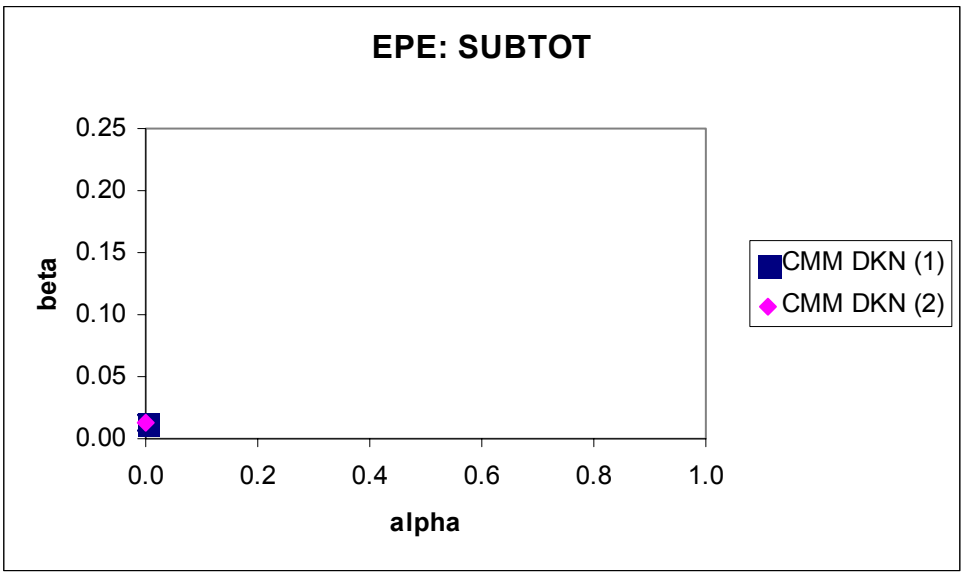


Figure 5.12 alpha-beta performance for EPE Y3 data: SUBTOT

In Figure 5.12 the two CMM experiments have identical performance for the variable SUBTOT, which is also confirmed in Table 5.20.

Table 5.20 Results for EPE Y3 data: SUBTOT

	CMM DKN (1)	CMM DKN (2)
alpha	0	0
beta	0.0125	0.0125
delta	0.0125	0.0125
RAE	0	0
RRASE	0	0
AREm1	0.890	0.890
AREm2	0.930	0.930

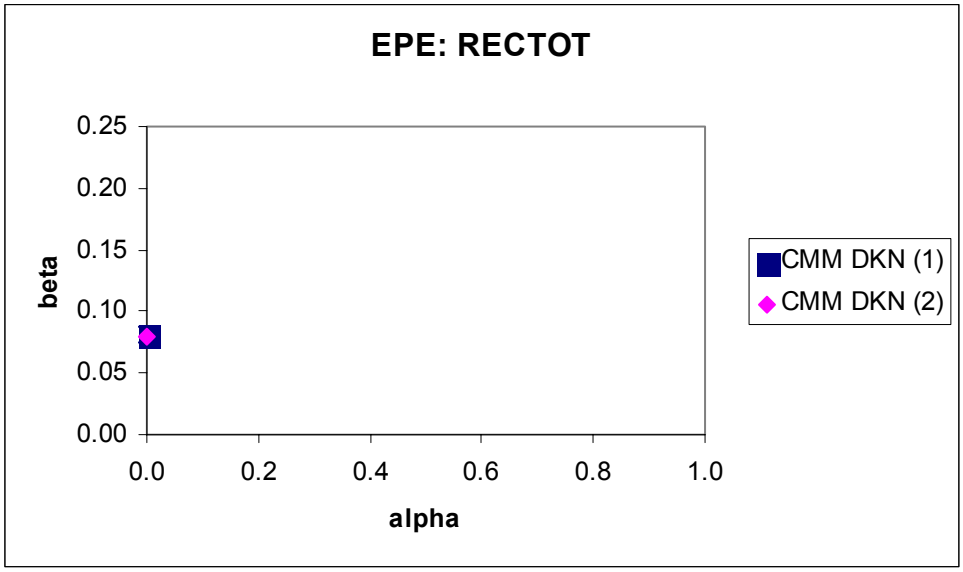


Figure 5.13 alpha-beta performance for EPE Y3 data: RECTOT

In Figure 5.13 the two CMM experiments have identical performance for the variable RECTOT, and this is confirmed in Table 5.21 apart from slight differences in AREm1 and AREm2.

Table 5.21 Results for EPE Y3 data: RECTOT

	CMM DKN (1)	CMM DKN (2)
alpha	0	0
beta	0.081	0.079
delta	0.081	0.079
RAE	0	0
RRASE	0	0
AREm1	0.6220	0.6218
AREm2	0.7192	0.7191

In summary, the performance of these two CMM experiments is virtually indistinguishable. This is of interest because it tends to confirm the result observed with the ABI Y3 experiments (and also observed during development) that the performance of the DKN method is relatively insensitive to minor changes in the parameter K (the number of neighbours). As with the ABI Y3 experiments, the two values of K chosen were 10 and 20. A greater range of experiments is needed to verify the extent to which this is a general result.

Dataset: EPE Y2

Technical Summary

Method:	*CMM*
Training data set:	*epe93na(y2).csv*
Hardware used:	*MS Windows 2000, PC "generic" (unbranded), AMD Athlon 1.2 MHz, L1 cache: 32 KB; L2 cache 256 KB, Amount of system physical memory in MB: 512 MB*
Additional software used:	*none*
Test scope:	*Imputation only*

Imputation

The CMM-based imputation method implemented in Euredit follows the simple strategy of attempting to impute all missing values found in a file. The results reported in this section are based on the Y2 version of the dataset, which contains missing values but not errors.

As with the ABI Y2 dataset, of the five CMM imputation "modes" available, the two experiments described here use median (shown as MDN in graphs) and weighted mean (shown as MDN in graphs). The median mode determines the local median value by considering only the K neighbouring records of the record with a missing value, as described in section 2.1. This value is used as a donor value, and this may be regarded as a donor method of imputation. The weighted mean method of imputation estimates a Euclidean distance-weighted mean value to be imputed by considering only the K neighbouring records of the record with a missing value, as described in section 2.1. This may be regarded as a modelling method of imputation, where the local Euclidean-distance weighted mean is computed over the local neighbourhood.

Results

Table 5.22 provides reference details and timings for the experiments considered.

Table 5.22 Experiment reference information for EPE Y2 data

Experiment	Method	Setup (minutes)	Run Time (s)	Other (s)	Total Time (s)
YE20004	CMM weighted mean	30	8		1808
YE20005	CMM median	30	16		1816
OE20001	DIS	180	60	60	10920

Table 5.23 shows imputation results for the variable TOTINVTO. CMM weighted mean appears to perform best overall for this variable, although none of the methods performs particularly well. Both CMM techniques seem to perform better than DIS except for m_1, m_2, and MSE.

Table 5.23 Results for EPE Y2 data: TOTINVTO

	CMM WMN	CMM MDN	DIS
Slope	1.25	3.86	0.26
t-val	4.25	21.48	-41.14
mse	996,388	956,804	1,106,955
R^2	0.117	0.162	0.040
dL1	100	102	127
dL2	289	300	324
dLinf	670	693	1271
K-S	0.450	0.597	0.646
K-S_1	0.017	0.022	0.012
K-S_2	0.0023	0.0049	0.0024
m_1	77.62	102.03	50.77
m_2	89,442	95,855	59,022
MSE	117	133	106
AREm1	0.096	0.096	0.096
AREm2	0.016	0.016	0.016

Table 5.24 below shows imputation results for the variable TOTEXPTO. The standard method DIS performs best overall for this variable, although the difference in performance between experiments is not large. CMM weighted mean is the slightly better of the two CMM methods tested here.

Table 5.24 Results for EPE Y2 data: TOTEXPTO

	CMM WMN	CMM MDN	DIS
Slope	1.01	3.91	0.84
t-val	1.04	64.08	-24.96
mse	603,250	626,239	206,832
R^2	0.460	0.396	0.884
dL1	50	65	39
dL2	239	283	118
dLinf	1,677	1,944	1,327
K-S	0.447	0.550	0.168
K-S_1	0.0047	0.0088	0.0037
K-S_2	0.0007	0.0018	0.0001
m_1	38.16	65.12	4.23
m_2	78,827	90,300	40,644
MSE	47.63	78.03	34.12
AREm1	0.186	0.186	0.186
AREm2	0.190	0.190	0.190

Table 5.25 below shows imputation results for the variable SUBTOT. The standard method DIS and CMM weighted mean seem to have identical performance for this variable in terms of the reported evaluation measures. Unfortunately, most measures for the CMM median method were not reported by the evaluation system, but this method performs better than the others for MSE, AREm1, and AREm2.

Table 5.25 Results for EPE Y2 data: SUBTOT

	CMM WMN	CMM MDN	DIS
dL1	1.44		1.44
dL2	2.08		2.08
dLinf	2.88		2.88
K-S	0.48		0.48
K-S_1	0.48		0.48
K-S_2	0.23		0.23
m_1	1.44		1.44
m_2	4.32		4.32
MSE	0.0102	0.0002	0.0102
AREm1	0.0010	0	0.0010
AREm2	0.0002	0	0.0002

Table 5.26 below shows imputation results for the variable RECTOT. The standard method DIS and CMM weighted mean again seem to have identical performance for this variable, in terms of the limited set of reported evaluation measures. The CMM median method performs less well than the others for most of the available results.

Table 5.26 Results for EPE Y2 data: RECTOT

	CMM WMN	CMM MDN	DIS
Slope		50	
t-val		29.5	
mse		14,088	
R^2		0.024	
dL1	16.82	22.59	16.82
dL2	32.66	53.77	32.66
dLinf	28.62	130.8976	28.62
K-S	1	0.97	1
K-S_1	0.083	0.056	0.083
K-S_2	0.010	0.018	0.010
m_1	16.82	22.59	16.82
m_2	1,067	2,894	1,067
MSE	0.027	0.306	0.027
AREm1	0.219	0.121	0.219
AREm2	0.106	0.046	0.106

In summary, it seems reasonable to judge that the CMM weighted mean experiment is slightly better than DIS in overall performance, and CMM weighted mean performs consistently better than the CMM median experiment with this dataset and for the variables considered here. It is interesting that the two different methods CMM and DIS produce identical results for the variables SUBTOT and RECTOT. The reason for this coincidence is not known.

Dataset: GSOEP

Technical Summary

Method:	*CMM*
Training data set:	*gsoep(m).csv*
Hardware used:	*MS Windows 2000, PC "generic" (unbranded), AMD Athlon 1.2 MHz, L1 cache: 32 KB; L2 cache 256 KB, Amount of system physical memory in MB: 512 MB*
Additional software used:	*none*
Test scope:	*Imputation only*

Results

Table 5.27 shows reference data for the two CMM experiments and the standard method DIS experiment discussed in this section.

Table 5.27: Experiment reference information for GSOEP Y2 data

Experiment	Method	Setup Time (minutes)	Run Time (s)	Other (s)	Total (s)
YG20001	CMM nearest neighbour	40	561		2961
YG20004	CMM weighted mean	40	613		3013
OG20001	DIS	240	120	60	14580

Tables 5.28 through to 5.33 show imputation results for personal income for years 1991 through 1996. CMM median generally provides best performance for all years for measures Slope, t-val, mse, R^2, dL1, and dL2. CMM weighted mean generally provides best performance for all years for measures dLinf, K-S, K-S_1, K-S_2, m_1, and m_2. Both CMM methods are generally better than DIS for personal income in each year.

Table 5.28 Results for GSOEP data: INCOME91

	CMM WMN	CMM MDN	DIS
Slope	0.86	0.91	0.84
t-val	-20.0	-12.4	-21.1
mse	401,832,598	309,713,976	458,790,920
R^2	0.50	0.59	0.44
dL1	12,525	11,195	13,172
dL2	20,796	17,771	22,322
dLinf	213,135	174,675	249,150
K-S	0.02908	0.061721	0.020772
K-S_1	0.002683	0.010572	0.002034
K-S_2	0.000028	0.00035	0.000015
m_1	19.176	506.163	170.446
m_2	20,092,967	135,309,251	32,806,816
MSE	209,811	229,270	212,806
AREm1	0.009186	0.009186	0.009186
AREm2	0.045946	0.045946	0.045946

Table 5.29 Results for GSOEP data: INCOME92

	CMM WMN	CMM MDN	DIS
Slope	0.90	0.93	0.85
t-val	-18.6	-12.8	-26.4
mse	381,285,970	325,343,326	543,518,083
R^2	0.58	0.63	0.46
dL1	11,585	10,501	13,164
dL2	19,980	18,259	25,080
dLinf	180,400	206,472	357,466
K-S	0.048	0.069	0.022
K-S_1	0.008	0.011	0.002
K-S_2	0.000188	0.000417	0.000014
m_1	40.229	501.568	178.928
m_2	66,930,385	113,146,705	212,321,990
MSE	216,289	239,480	240,515
AREm1	0.011627	0.011627	0.011627
AREm2	0.028087	0.028087	0.028087

Table 5.30 Results for GSOEP data: INCOME93

	CMM WMN	CMM MDN	DIS
Slope	0.90	0.93	0.63
t-val	-16.0	-11.3	-39.4
mse	420,464,830	375,216,220	940,592,280
R^2	0.59	0.63	0.20
dL1	11,882	10,557	20,716
dL2	20,624	19,322	37,505
dLinf	259,720	314,513	430,800
K-S	0.031	0.060	0.073
K-S_1	0.003	0.006	0.012
K-S_2	0.000048	0.000181	0.000488
m_1	210.24	329.23	5763.90
m_2	211,077,806	298,913,640	820,550,842
MSE	256,079	258,510	3,739,006
AREm1	0.009295	0.009295	0.009295
AREm2	0.035688	0.035688	0.035688

Table 5.31 Results for GSOEP data: INCOME94

	CMM WMN	CMM MDN	DIS
Slope	0.90	0.93	0.55
t-val	-17.7	-13.2	-48.3
mse	493,762,837	34,568,1827	1,123,170,417
R^2	0.53	0.64	0.13
dL1	11,932	10,705	23,405
dL2	23,222	18,953	47,698
dLinf	504,000	218,989	336,000
K-S	0.031	0.042	0.082
K-S_1	0.0024	0.0072	0.021
K-S_2	0.00003	0.00015	0.00069
m_1	56	295	7,140
m_2	98,909,186	51,693,814	1,903,221,243
MSE	300,855	298,196	5,274,514
AREm1	0.031	0.031	0.031
AREm2	0.092	0.092	0.092

Table 5.32 Results for GSOEP data: INCOME95

	CMM WMN	CMM MDN	DIS
Slope	0.93	0.96	0.63
t-val	-11.48	-7.16	-35.11
mse	1,193,306,905	1,062,851,544	1,979,718,005
R^2	0.39	0.45	0.11
dL1	13,152	11,632	23,824
dL2	34,516	32,499	51,365
dLinf	1,105,722	1,084,114	1,104,788
K-S	0.047	0.038	0.070
K-S_1	0.0015	0.0020	0.0038
K-S_2	0.00003	0.00004	0.00009
m_1	2,099	1,571	4,089
m_2	1,095,096,630	1,138,478,682	325,270,887
MSE	682,443	490,634	1,958,840
AREm1	0.0176	0.0176	0.0176
AREm2	0.121	0.121	0.121

Table 5.33 Results for GSOEP data: INCOME96

	CMM WMN	CMM MDN	DIS
Slope	0.89	0.93	0.65
t-val	-19.30	-14.70	-40.72
mse	616,937,641	536,884,912	1,244,737,645
R^2	0.560	0.611	0.192
dL1	13,542	12,097	22,641
dL2	24,882	23,026	40,047
dLinf	384,000	426,990	533,200
K-S	0.029	0.054	0.076
K-S_1	0.0024	0.0053	0.0062
K-S_2	0.00003	0.00012	0.00024
m_1	1,234	706	3,065
m_2	404,398,096	488,794,478	308,617,965
MSE	430,947	322,525	1,261,773
AREm1	0.0164	0.0164	0.0164
AREm2	0.0016	0.0016	0.0016

Tables 5.34 through to 5.39 show imputation results for household income for years 1991 through 1996. CMM median generally provides best performance for all years for measures Slope, t-val, mse, R^2, dL1, and dL2. CMM weighted mean generally provides best performance for all years for measures K-S, K-S_1, and K-S_2. Both CMM methods are generally better than DIS for household income in each year.

Table 5.34 Results for GSOEP data: HHINCO91

	CMM WMN	CMM MDN	DIS
Slope	0.78	0.89	0.78
t-val	-17.54	-9.99	-14.47
mse	2,395,856,538	1,710,156,061	2,431,426,303
R^2	0.104	0.188	0.063
dL1	33,645	28,410	37,530
dL2	53,816	42,516	51,860
dLinf	538,180	266,015	595,045
K-S	0.042	0.070	0.101
K-S_1	0.0060	0.0194	0.0093
K-S_2	0.00008	0.00086	0.00043
m_1	671	665	5,167
m_2	655,739,319	411,409,244	521,576,263
MSE	603,008	521,736	3,122,290
AREm1	0.0037	0.0037	0.0037
AREm2	0.0112	0.0112	0.0112

Table 5.35 Results for GSOEP data: HHINCO92

	CMM WMN	CMM MDN	DIS
Slope	0.80	0.88	0.70
t-val	-18.10	-11.35	-23.96
mse	2,321,481,767	1,934,871,086	3,818,276,364
R^2	0.164	0.225	0.060
dL1	34,416	30,366	4,2865
dL2	50,519	44,832	70,954
dLinf	459,399	463,633	571,873
K-S	0.041	0.059	0.210
K-S_1	0.0038	0.0090	0.0222
K-S_2	0.00005	0.00033	0.00268
m_1	1,250	983	8,360
m_2	284,472,693	769,233,258	1,108,294,055
MSE	692,381	592,666	7,973,571
AREm1	0.0069	0.0069	0.0069
AREm2	0.0200	0.0200	0.0200

Table 5.36 Results for GSOEP data: HHINCO93

	CMM WMN	CMM MDN	DIS
Slope	0.85	0.94	0.73
t-val	-12.78	-5.43	-18.22
mse	2,618,549,410	2,246,139,320	3,779,470,656
R^2	0.189	0.241	0.053
dL1	33,982	30,136	43,860
dL2	51,824	47,300	64,873
dLinf	706,300	679,877	698,600
K-S	0.035	0.081	0.131
K-S_1	0.0046	0.0117	0.0101
K-S_2	0.00007	0.00054	0.00064
m_1	3,734	4,115	7,575
m_2	1,449,509,777	2,091,161,753	1,133,198,575
MSE	2,021,089	2,287,736	6,570,995
AREm1	0.0088	0.0088	0.0088
AREm2	0.0297	0.0297	0.0297

Table 5.37 Results for GSOEP data: HHINCO94

	CMM WMN	CMM MDN	DIS
Slope	0.830	0.930	0.664
t-val	-14.48	-5.97	-24.13
mse	2,470,340,040	1,979,833,601	3,763,918,205
R^2	0.206	0.287	0.055
dL1	33,472	29,282	45,332
dL2	51,138	44,580	69,104
dLinf	474,304	477,936	465,500
K-S	0.050	0.078	0.165
K-S_1	0.0065	0.0142	0.0209
K-S_2	0.00014	0.00061	0.00158
m_1	3,918	4,155	5,094
m_2	858,329,298	1,541,810,260	453,128,500
MSE	2,119,383	2,261,774	3,248,047
AREm1	0.0035	0.0035	0.0035
AREm2	0.0140	0.0140	0.0140

In summary, CMM median provides best performance overall in terms of measures Slope, t-val, mse, R^2, dL1, and dL2. Conversely, CMM weighted mean provides best performance overall for measures K-S, K-S_1, and K-S_2, and sometimes for dLinf. Both CMM experiments seem fairly consistent in outperforming the standard method DIS.

Table 5.38 Results for GSOEP data: HHINCO95

	CMM WMN	CMM MDN	DIS
Slope	0.837	0.921	0.612
t-val	-13.05	-6.87	-30.28
mse	3,917,676,497	3,389,727,967	5,246,882,384
R^2	0.139	0.201	0.044
dL1	35,708	31,430	48,774
dL2	63,494	58,184	79,750
dLinf	1,210,388	1,185,277	1,210,388
K-S	0.059	0.076	0.147
K-S_1	0.0034	0.0064	0.0092
K-S_2	0.00008	0.00024	0.00060
m_1	4,319	4,389	5,114
m_2	2,436,463,373	3,031,406,074	714,368,932
MSE	2,408,895	2,431,259	3,276,923
AREm1	0.0214	0.0214	0.0214
AREm2	0.1005	0.1005	0.1005

Table 5.39 Results for GSOEP data: HHINCO96

	CMM WMN	CMM MDN	DIS
Slope	0.834	0.935	0.617
t-val	-13.64	-5.37	-28.96
mse	2,548,703,778	2,104,008,911	3,938,185,505
R^2	0.252	0.325	0.066
dL1	34,696	31,287	47,496
dL2	51,667	45,888	70,178
dLinf	408,612	424,059	532,290
K-S	0.049	0.066	0.149
K-S_1	0.0064	0.0117	0.0142
K-S_2	0.00015	0.00047	0.00104
m_1	4,100	4,312	5,042
m_2	1,057,384,582	1,761,853,557	134,894,910
MSE	2,355,800	2,494,040	3,328,161
AREm1	0.0097	0.0097	0.0097
AREm2	0.0014	0.0014	0.0014

Dataset: DLFS

Technical Summary

Method:	*CMM*
Training data set:	*lfsn_dk2(miss).csv*
Hardware used:	*MS Windows 2000, PC "generic" (unbranded), AMD Athlon 1.2 MHz, L1 cache: 32 KB; L2 cache 256 KB, Amount of system physical memory in MB: 512 MB*
Additional software used:	*none*
Test scope:	*Imputation only*

Table 5.40 shows reference information for the five CMM experiments, two SOLAS experiments, and two DIS experiments discussed in this section.

Table 5.40: Experiment reference and times for imputation using DLFS Y2 data

Experiment	Method	Setup (minutes)	Run Time (s)	Other (s)	Total (s)
YL20002	CMM RND (random neighbour)	30	72		1872
YL20001	CMM NN (nearest neighbour)	30	73		1873
YL20003	CMM MN (mean)	30	82		1882
YL20004	CMM WMN (weighted mean)	30	82		1882
YL20005	CMM MDN (median)	30	82		1882
DL21600	SOLAS (1)	30	600	1200	3600
DL21700	SOLAS (2)	30	600	1200	3600
OL20001	DIS (1)	60	120	60	3780
OL20002	DIS (2)	60	120	60	3780

Table 5.41 shows comparative results for each experiment for the variable INCOME (the only variable to be imputed for this dataset).

Table 5.41 Results for DLFS data: INCOME

	CMM NN	CMM RND	CMM MN	CMM WMN	CMM MDN	DIS (1)	DIS (2)	SOLAS (1)	SOLAS (2)
Slope	0.832	0.773	0.886	0.886	0.946	0.807	0.834	0.743	0.846
t-val	-35.19	-41.41	-28.81	-28.77	-13.89	-40.94	-32.84	-44.65	-24.17
mse	9.57E+09	11.9E+09	6.65E+09	6.65E+09	6.68E+09	10.2E+09	9.66E+09	13.1E+09	9.3E+09
R^2	0.262	0.145	0.424	0.424	0.423	0.224	0.243	0.105	0.190
dL1	62,391	74,261	48,783	48,754	45,132	64,602	63,225	78,753	66,077
dL2	102,680	117,357	81,747	81,714	81,157	107,384	102,042	124,914	97,627
dLinf	850,380	884,379	833,999	834,028	834,249	870,720	869,105	965,724	830,290
K-S	0.060	0.085	0.149	0.149	0.135	0.082	0.058	0.080	0.212
K-S_1	0.0109	0.0136	0.0261	0.0261	0.0218	0.0141	0.0116	0.0137	0.0359
K-S_2	0.0004	0.0007	0.0019	0.0019	0.0011	0.0007	0.0004	0.0006	0.0038
m_1	7,290	10,728	11,516	11,499	1,132	11,251	6,315	11,552	15,112
m_2	8.88E+08	25.4E+08	30.5E+08	30.5E+08	69.0E+08	22.9E+08	5.43E+08	36.2E+08	35.8E+08
MSE	4.9E+06	9.4E+06	10.5E+06	10.5E+06	1.0E+06	10.2E+06	3.9E+06	10.7E+06	17.3E+06
AREm1	0.0468	0.0468	0.0468	0.0468	0.0468	0.0468	0.0468	0.3661	0.3661
AREm2	0.0733	0.0733	0.0733	0.0733	0.0733	0.0733	0.0733	0.3661	0.3661

In summary, for measures Slope, t-val, mse, R^2, dL1 and dL2 the three CMM experiments mean, weighted mean and median perform best. For measures K-S_1, K-S_2 the CMM nearest neighbour experiment performs best. The CMM median experiment seems to stand out in terms of generally good-to-best overall performance.

Dataset: SARS

Technical Summary

Method:	*CMM*
Training data set:	*newhholdm.csv*
Hardware used:	*MS Windows 2000, PC "generic" (unbranded), AMD Athlon 1.2 MHz, L1 cache: 32 KB; L2 cache 256 KB, Amount of system physical memory in MB: 512 MB*
Additional software used:	*none*
Test scope:	*Imputation only*

Table 5.42 shows reference information for the CMM, CANCEIS/SCIA, and DIS experiments discussed in this section.

Table 5.42: Reported times for imputation using SARS Y2 data

Experiment	Method	Setup (minutes)	Run Time (s)	Other (s)	Total Time (s)
YS20001	CMM weighted mean	30	43502		45302
IS20001	CANCEIS/SCIA	2160	11539		141139
OS20001	DIS	90	345600	60	351060

Tables 5.43 and 5.44 show comparative results for the two continuous variables in this dataset, AGE and HOURS.

For the variable AGE, Table 5.43 shows that CANCEIS/SCIA is generally performing best, although CMM weighted mean performance is almost as good. Performance for DIS on this variable is not as good.

Table 5.43 Results for SARS Y2 data: AGE

	CMM WMN	DIS	CANCEIS/ SCIA
Slope	0.989	0.846	0.996
t-val	-13.28	-216.48	-9.90
mse	94.21	225.94	39.00
R^2	0.820	0.591	0.926
dL1	6.94	11.26	3.67
dL2	9.72	17.45	6.250
dLinf	88	95	79
K-S	0.052	0.132	0.006
K-S_1	0.0204	0.0634	0.0024
K-S_2	0.00070	0.00580	0.00001
m_1	0.025	6.020	0.171
m_2	46	594	17
MSE	0.0011	0.2302	0.0013

For the variable HOURS, Table 5.44 below shows that CMM weighted mean is generally performing best. Performance for DIS and CANCEIS/SCIA on this variable is generally not as good.

Table 5.44 Results for SARS Y2 data: HOURS

	CMM WMN	DIS	CANCEIS/ SCIA
Slope	1.01	0.91	0.87
t-val	8.97	-34.19	-45.67
mse	179	600	572
R^2	0.210	0.066	0.007
dL1	8.61	16.54	16.68
dL2	13.40	24.83	24.52
dLinf	90	90	90
K-S	0.130	0.245	0.201
K-S_1	0.0404	0.1215	0.0876
K-S_2	0.0029	0.0258	0.0138
m_1	3.64	10.94	7.89
m_2	233	327	214
MSE	0.0163	0.138	0.079

Table 5.45 shows comparative results for the non-continuous variables in this dataset. The household variables are indicated by and asterisk (*) in the table, and these variables are treated differently in the CMM experiment during imputation so that these variables are constrained to share a common value for the household.

The CMM weighted mean experiment performs better than DIS and CANCEIS/SCIA for the (non-continuous) variables: BATH, INSIDEWC, DISTWORK, WORKPLCE, ECONPRIM, and ISCO2.

In summary, for the continuous variables the CMM weighted mean experiment performs as well or better than the two standard methods considered here. For six out of the nineteen categorical variables considered (BATH, INSIDEWC, DISTWORK, WORKPLCE, ECONPRIM, and ISCO2), the CMM weighted mean experiment performs better than the two standard methods. Considering the remainder, for five variables (CARS, HHSPTYPE, ROOMSNUM, TENURE, and LTILL) it is better than the CANCEIS/SCIA experiment, and for three variables (QUALEVEL, QUALSUB, and SEX) it is better than the DIS experiment. Considering just the categorical variables alone, the DIS experiment tends to perform better on the most variables. However, considering all variables, the CMM weighted mean experiment tends to perform better on the most variables.

Table 5.45 Results for SARS Y2 non-continuous variables

		CMM WMN	DIS	CANCEIS/ SCIA
BATH *	W	18	105	1175
	D	0.0005	0.0080	0.0399
	Eps	0	0	0.03
CENHEAT *	W	1062	34	365
	D	0.38	0.44	0.46
	Eps	0.370	0.434	0.456
INSIDEWC *	W	23	82	990
	D	0.001	0.007	0.041
	Eps	0	0	0.03
CARS *	W	924	316	1671
	D	0.49	0.56	0.61
	Eps	0.482	0.558	0.608
HHSPTYPE *	W	1,730	966	5,499
	D	0.589	0.708	0.731
	Eps	0.581	0.702	0.725
ROOMSNUM *	W	1,559	932	5,721
	D	0.701	0.786	0.808
	Eps	0.695	0.781	0.803
TENURE *	W	4,940	1,758	4,980
	D	0.545	0.624	0.671
	Eps	0.537	0.616	0.664
COBIRTH	W	6,883	48	698
	D	0.422	0.370	0.211
	Eps	0.415	0.362	0.202
DISTWORK	W	571	3,203	1,724
	D	0.683	0.829	0.819
	Eps	0.668	0.822	0.811
LTILL	W	355	288	1,577
	D	0.128	0.176	0.136
	Eps	0.118	0.167	0.126
MSTATUS	W	3,705	920	40
	D	0.290	0.321	0.160
	Eps	0	0	0
QUALNUM	W	1017.6	9.6	285.9
	D	0.140	0.171	0.049
	Eps	0.129	0.161	0.037
QUALEVEL	W	337	2,494	121
	D	0.534	0.903	0.524
	Eps	0.503	0.891	0.500
QUALSUB	W	1,474	2,535	579
	D	0.929	0.985	0.944
	Eps	0.919	0.981	0.936
RELAT	W	2,708	1,287	42
	D	0.276	0.355	0.051
	Eps	0.267	0.345	0.040
SEX	W	17.23	654.96	0.41
	D	0.262	0.331	0.230
	Eps	0.253	0.322	0.220
WORKPLCE	W	172	2,939	939
	D	0.183	0.463	0.251
	Eps	0.167	0.450	0.236
ECONPRIM	W	965	1,949	1,465
	D	0.372	0.704	0.493
	Eps	0.358	0.695	0.481
ISCO2	W	2,216	5,842	3,663
	D	0.857	0.914	0.899
	Eps	0.850	0.910	0.895

Strengths and weaknesses of this method

A major strength of the CMM-based methods is the relative ease of use and minimal skill requirements. This is particularly true for the imputation process. With adequate provision of metadata the method is highly automated, so that the system will automatically set almost all parameters and use defaults where appropriate. At present, a little more judgement is required to use the current error localisation process, but this method is still new and further refinement is expected to improve this aspect.

The CMM based methods are generally very fast in computational terms, despite some weaknesses in the current prototype implementation of file and data handling components. These prototype components are quite separate from the AURA CMM library (which is now mature and efficient), and are currently reducing run time performance especially for larger datasets where AURA and CMM methods would otherwise excel. Nevertheless, for smaller datasets these inefficiencies in the edit and imputation prototype are less evident, and the York system is typically significantly faster than the "standard methods" in these cases.

Table 5.46 shows a comparison of the run time and set up time performance for CMM-based experiments and the "standard method" experiments DIS, GEIS, SOLAS, and CANCEIS/SCIA (see page 3, and notes on timing on page 10) for the same datasets. Speedup factors for basic runtime performance range between 0.2 and 667. A more practical measure of speedup is also shown which combines run time and set up time, giving CMM speedup factors which range from 1.9 to 15.7 times faster than the standard methods. Relatively slow runtimes (and hence speedup factors) for the GSOEP and SARS datasets are attributed to inefficient file and text handling issues in the prototype edit and imputation system, as described above.

Table 5.46 Performance speedup of CMM compared with standard methods

Speedup Factor (runtime)	Speedup Factor (runtime + setup)	Dataset	Total Records	Total Variables	Missing Values	Compared with Method
0.2	4.9	GSOEP	5,383	195	20,390	DIS
0.3	3.1	SARS Y2	492,472	31	648,881	CANCEIS/SCIA
1.4	4	ABI Y2	6,233	33	2,765	DIS
1.7	2	DLFS	15,579	15	4,175	DIS
7.5	6	EPE Y2	1,039	70	2,728	DIS
8	1.9	DLFS	15,579	15	4,175	SOLAS
8	7.75	SARS Y2	492,472	31	648,881	DIS
70	4.9	ABI Y2	6,233	33	2,765	GEIS
667	15.7	ABI Y3	6,233	33	N/A	GEIS

Table5.46 also demonstrates how the highly-automated features of the York system also contribute to significantly shorter times for the whole procedure (including preparation time). With a few additional refinements the York system could provide a very effective tool for imputation, requiring:

- minimal skill/training,
- modest system requirements,
- simple set-up,
- parameter setting/tuning is unnecessary,
- high-speed/throughput, and
- generally good performance against Euredit criteria.

The method provides reasonable imputation performance in most situations. However at present, CMM-based imputation is sometimes less effective in situations where significant constraints are imposed by edit rules. For example, performance with the ABI Y2 dataset is generally better for the GEIS "standard method" than the CMM-based imputation. This is possibly because GEIS ensures that each proposed imputation satisfies all edit rules whereas CMM-based imputation does not take any account of edit rules in the prototype version, although range limits and specific category value restrictions are implemented. The CMM-based method does not implement mechanisms to preserve constraints at present, and imputed values are always derived from the properties of similar matching records (units). The method seems more successful in those situations where variables to be imputed are not excessively constrained by rules (other than basic range or category value checks. Thus performance appears better for datasets like DLFS and GSOEP.

3 Conclusion

3.1 Discussion of results

For error localisation using the ABI Y3 dataset, indications are that the current CMM (DKN) approach performs less well compared with the results available for the GEIS method, though it should be noted that GEIS uses edit rules to identify some errors, while CMM does not use any rules (at present).

For imputation using the ABI Y2 dataset, performance is generally better than DIS and slightly worse than GEIS when compared with these standard methods. Overall, there is not much difference between any of the methods compared here. CMM performance tends to be better for variables EMPTOTC, TAXTOT, and ASSDISP, and less good for TURNOVER, PURTOT, and ASSACQ.

For error localisation using the EPE Y3 dataset, no results for standard methods were available for comparison. The two CMM experiments performed produced almost identical results, tending to confirm earlier indications that the DKN method seems relatively insensitive to minor variations in the value of the parameter K (the selected number of neighbours).

For imputation using the EPE Y2 dataset, CMM weighted mean experiment is slightly better than the standard method DIS in overall performance, whereas the CMM median method is generally worse than DIS. CMM weighted mean performance was best or equal best for three out of four key variables, and second best for the fourth.

For imputation using the GSOEP dataset, the CMM methods perform very well. CMM median provides best performance overall in terms of measures Slope, t-val, mse, R^2, dL1, and dL2. CMM weighted mean provides best performance overall for measures K-S, K-S_1, and K-S_2, and sometimes for dLinf. Both CMM methods outperform the standard method DIS in most cases. The good performance of CMM methods here is a little surprising because the longitudinal structure of the data was not considered at all, and the CMM method was used in a highly automated way, using parameter settings provided by the system. Overall, the distance-weighted mean mode of imputation gave best results here, which is expected with such a dataset composed largely of continuous values. The nearest-neighbour mode also performed quite well and seems to perform better than distance-weighted mean in achieving small Kolmogorov-Smirnov distance measures.

For imputation using the DLFS dataset, CMM methods perform well overall, and perform better than other methods tested for the particular measures slope and t-val when using the median imputation mode. For the Kolmogorov-Smirnov measures, the nearest-neighbour imputation mode is very good, and is better than the other CMM modes. The CMM median experiment seems to stand out in terms of generally good-to-best overall performance.

For imputation using the SARS Y2 dataset, CMM performs as well or better than the two standard methods considered for the continuous variables AGE and HOURS. Considering just the categorical variables, the standard method DIS tends to perform better on the greater number of variables. However, considering all variables, the CMM weighted mean experiment tends to perform better on the greater number of variables. The reduced performance with categorical variables can be accounted for in the way the York system is currently designed to use only one method of imputation during the imputation task for a given file (e.g. mean, or median). This is rather inflexible for some imputation applications, but a simple modification enable the most appropriate method of imputation to be used depending on variable type. For categorical variables, a new imputation method would be added to perform imputation of the mode value, and this is expected to give much better performance with categorical variables, at the same time maintaining the current good performance for continuous variables.

The discussion of results above seems to suggest that different CMM imputation modes are often best suited to different types of data. As a rule-of-thumb, the distance-weighted mean mode often gives the best performance of the current CMM imputation modes, but occasionally the median mode can give better performance (as with the DLSF dataset), and it seems reasonable to speculate that the known bias present in that dataset is better modelled using the median imputation mode. Categorical variables are not imputed in an optimal way at present, and the best compromise is currently provided by the distance-weighted mean mode of imputation, which provides a fair approximation to the modal value. Future work should provide a different imputation "mode" for each variable type in a dataset, in particular, categorical variables should be imputed by calculating the modal value from the set of neighbours values considered.

The EPE dataset was known in advance to contain a large number of zero entries, and for some methods it as anticipated that this would cause some difficulties. Although the CMM methods did not perform especially well with the EPE dataset, there is no particular reason to suppose that this due to the high number of zero values present. Assuming these values are not erroneous in some way, the CMM methods will simply treat these as legitimate possible values when determining a donor value.

CMM imputation methods were not attempted with any of the Y3 datasets primarily due to lack of time but other factors affected this decision, notably some doubts raised about the comparability of Y3 imputation results by other partners. However, in principle it would not require significant resources to perform the experiments using Y3 versions of datasets, and it is expected that the CMM methods in general would perform well in these conditions.

3.2 Weaknesses in the editing/evaluation procedure(s) considered

Considering the new error localisation method CMM (DKN), the results show there is some potential benefit in this method, which is relatively fast. However, further development is needed to investigate more carefully the parameter space of the method in relation to error localisation performance.

For imputation, the CMM based methods with different imputation modes seem to offer good performance in many situations, but are sometimes less effective when variables are related via constraints (such as totals and their constituent values). To address this issue it would be necessary to extend the current York system to allow rules expressing such constraints to be incorporated during processing. This was considered beyond the scope of the Euredit project due to the considerable effort required. Current provision for categorical variables is inadequate, with the best solution provided at present being the distance-weighted mean mode. A relatively simple modification to the York system could allow categorical values to be imputed with the modal value of the local neighbourhood, which should yield a good improvement in performance for those variables.

For both edit and imputation, the absence of a mechanism to incorporate the results of applying edit rule checks limits the performance of the system in some situations.

For the relatively large SARS dataset some problems were encountered with memory usage, which contributed to lengthy processing times. These problems can be largely attributed to inefficient file and data handling in the York prototype edit and imputation system, since the underlying core CMM implementation (in the AURA library) is known to be very efficient and is designed specifically for high-performance in large data environments.

3.3 Areas for further study

A number of areas for further study and improvement have been identified:

- investigate the parameter space of the method in relation to error localisation performance in more detail;
- modification to the York system to allow categorical values to be imputed with the value of the mode of the local neighbourhood;
- review some of the implementation features of the current edit and imputation system at York to improve speed and so better exploit the high-speed capabilities of the CMM software (AURA);
- investigate the feasibility of incorporating a general constraint and "edit rule" processing feature.

The extent to which these areas may be addressed will, of course, be limited beyond the end of the Euredit project.

4 Glossary of Terms

CMM – Correlation Matrix Memory is a type of (artificial) neural network.

AURA – Advanced Uncertain Reasoning Architecture is a set of methods that supports the use of CMM-based systems in a range of applications. The AURA software class library is an implementation of AURA using C++.

Imputation "mode" – This refers to one of the five currently available methods that may be selected for the final imputation step. The currently available modes are: Nearest-neighbour (NN), Random-neighbour (RND), Mean (MN), Weighted-mean (WMN), and Median (MDN). See section 2.1 for details.

DKN – "Distance to K^{th} Neighbour" is the CMM-based method for error localisation used in Euredit. See section 2.1 for details.

5 Bibliography

Austin, J., (1997) AURA, A distributed associative memory for high-speed symbolic reasoning. In: Sun, R. and Alexandre F. (eds.) **Connectionist-Symbolic Integration**, Lawrence Erlbaum Associates, Inc.

Austin, J., and Lees, K., (2000) A Search Engine Based on Neural Correlation Matrix Memories, Neurocomputing 35, 55 – 72, November 2000, Elsevier Science.

Austin, J. and O'Keefe, S., (1999) Neural Networks for the Estimation of Missing Data, Euredit Report T009.01.

Byers, S. and Raftery A.E., (1996) Nearest neighbour clutter removal for estimating features in spatial point processes, TR 305, Department of Statistics, University of Washington, Seattle.

Lees, K., O'Keefe, S., and Austin, J., (2002) Euredit D4.4.1 and D5.4.1: Application of CMM techniques to data editing and imputation, Report covering Deliverables D4.4.1 and D5.4.1 of the Euredit project.

Ramaswamy, S., Rastogi, R., and Shim, K. (2000) Efficient algorithms for mining outliers from large data sets, (SIGMOD 2000 Proceedings), SIGMOD Record, 29, 2, 427-438.

Steinbuch, K. (1961), Die lernmatrix, Kybernetic, 1, pp. 36-45.

Willshaw, D.J., Buneman, O.P., and Longuet-Higgins, H.C., (1969) Non-Holographic Associative Memory, Nature 222, 960-962.

Zhou, P., Austin, J. and Kennedy, J. (1999) A high performance K-NN classifier using a binary correlation matrix memory. Advances in Neural Information Processing Systems Vol. 11.