Neural networks and predictive matching for flexible imputation

Fernando TUSELL*

Presented at the DataClean 2002 conference,
Jyväskylä (Finland),
30 May 2002.

Abstract

Let $\mathbf{X}$ be a $N \times (p+q)$ data matrix, with entries partly missing in the last q columns. A problem of practical relevance is that of imputing the missing values in such an incomplete data set. We propose to use (partially) predictive matching coupled with a flexible fit, such as provided by a neural network. The merits and demerits of the approach suggested are discussed.

Keywords: Missing data; multiple imputation; predictive matching; neural networks; binary trees.

1 Introduction

Let $\mathbf{X}$ be a $N \times (p+q)$ data matrix, with entries partly missing in the last q columns. This situation may arise, for instance, if $\mathbf{X}$ contains data collected in two surveys given to n_{obs} and n_{mis} subjects respectively ($N = n_{\mathrm{obs}} + n_{\mathrm{mis}}$), and complete data is available for the n_{obs} subjects interviewed in the first survey while the last q questions were not asked to the n_{mis} subjects interviewed in the second survey. The layout of observed and missing data is shown in Figure 1.

A problem of practical relevance is that of drawing inferences from such an incomplete data set, and a considerable body of literature exist on this issue. A landmark is the monograph [13], setting up a methodology and advocating the use of multiple imputation. [18] develops algorithms for imputation based on the EM algorithm and data augmentation.

The methods in [18] require the specification of a parametric model and a (possibly non-informative) prior on the parameters. There exist implementations as stand-alone programs and S-PLUS© functions dealing with the normal, multinomial and mixed cases. Some of these functions have been ported to the statistical package R. Alternatives include MICE (see [23],

*Facultad de CC.EE. y Empresariales, Universidad del País Vasco / Euskal Herriko Unibertsitatea, Avda. Lehendakari Aguirre, 83, 48015 BILBAO, Spain. e-mail: etptupaf@bs.ehu.es.

Figure 1: Regular pattern of missingness. The shaded area corresponds to observed data.

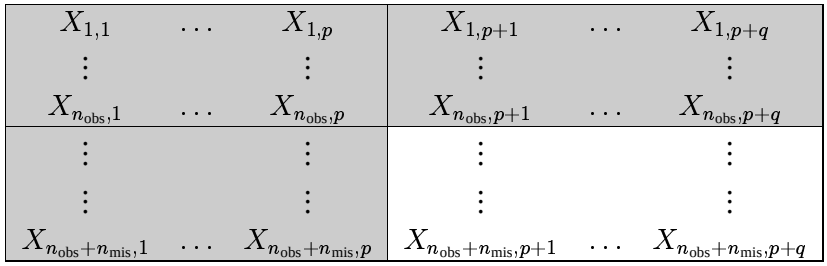
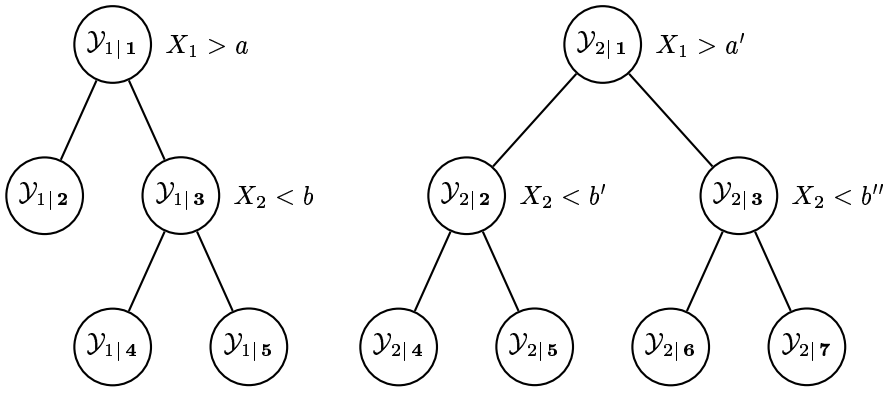


Figure 2: Trees $\mathcal{Y}_{1|\,\mathbf{X}}$ and $\mathcal{Y}_{2|\,\mathbf{X}}$. Next to each non terminal node is the condition used to split.



[22] for instance) or NIBAS (see [16]), which addresses the harder problem of statistical matching).

When the underlying models provide a good approximation to the mechanism generating the data, there is probably little motivation to go beyond those methods, which have a solid theoretical grounding. Nevertheless situations exist where a nonparametric approach might be adequate.

One possibility that has some appeal is the use of binary trees. It has a drawback, though: one is forced to discretize continuous responses. In Section 3 a method is proposed better suited to impute continuous variables. Its motivation, merits and demerits are better seen against the background of our previous work with trees ([3], [4], for instance; see also [14] and [15]). A summary account is therefore given in Section 2.

Figure 3: Partitions induced in $\mathcal{X}$ by trees $\mathcal{Y}_{1|\mathbf{X}}$ and $\mathcal{Y}_{2|\mathbf{X}}$.

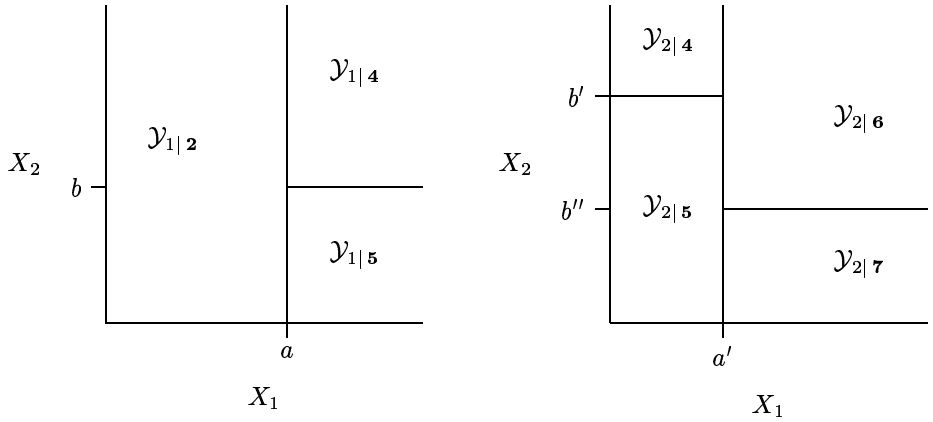
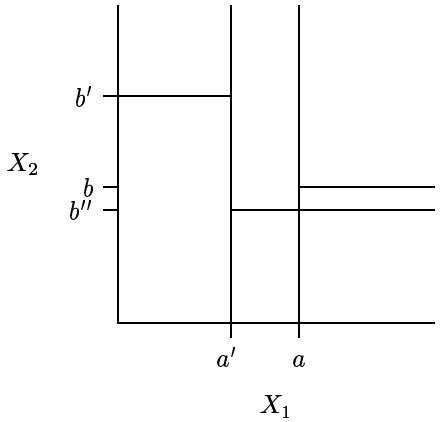


Figure 4: "Joint" partition induced in $\mathcal{X}$ by trees $\mathcal{Y}_{1|\mathbf{X}}$ and $\mathcal{Y}_{2|\mathbf{X}}$.



2 Multivariate imputation with binary trees

In order to impute a multivariate response, [4] proposed to use a collection of ordinary (scalar response) binary trees, one for each component of the response. They are built with the methodology described in [5] as implemented by [21].

We denote by $\mathcal{Y}_{x|\,\mathbf{z}}$ a tree "regressing" x on the predictors in $\mathbf{z}$ —the response x might just as well be qualitative, and the tree a classification rather than a regression tree. Assume we have a training sample as shown in Figure 1 Call $\mathbf{X}_{\text{obs}}$ the vector of the p variables fully observed (for all $n_{\text{obs}}+n_{\text{mis}}$ subjects) and $\mathbf{X}_{\text{mis}}$ the vector of the q variables incompletely observed (missing for the last n_{mis} subjects). We denote $\mathcal{X}$ the space spanned by the p components of $\mathbf{X}_{\text{obs}}$. The case where observations are missing irregularly can also be handled (using surrogate splits), but we will only deal with the regular case as described in the Introduction.

In [4], the following imputation strategy (the "forest climbing algorithm") is proposed:

1. Build trees $\mathcal{Y}_{X_{p+1}|\,\mathbf{X_{obs}}}, \ldots, \mathcal{Y}_{X_{p+q}|\,\mathbf{X_{obs}}}$ using the n_{obs} complete observations.
2. Drop each of the n_{mis} incomplete cases down the q trees constructed. Let case i fall in the terminal nodes labelled $(\ell_{i,1}, \ldots, \ell_{i,q})$ of (respectively) trees $\mathcal{Y}_{X_{p+1}|\,\mathbf{X_{obs}}}, \ldots, \mathcal{Y}_{X_{p+q}|\,\mathbf{X_{obs}}}$. Call $(\ell_{i,p+1} \cap \ldots \cap \ell_{i,p+q})$ the subset of the n_{obs} complete cases which also end in said leaves. If $(\ell_{i,p+1} \cap \ldots \cap \ell_{i,p+q}) \neq \emptyset$, impute the missing values of case i by those of one complete case which also ends in $(\ell_{i,p+1} \cap \ldots \cap \ell_{i,p+q})$. If multiple imputations are desired, sample k cases out of that intersection.
3. If $(\ell_{i,p+1} \cap \ldots \cap \ell_{i,p+q}) = \emptyset$, iteratively replace leaves by their ancestors ("climb the trees") until a non empty intersection is found. Then, proceed as in step 2 above.

The idea is quite simple and better understood referring to the simple example in Figures 2 and 3. There, we assume $p = q = 2$ and therefore two trees for the responses X_3,X_4, both with predictors X_1,X_2. Figure 2 displays the two trees and Figure 3 the respective partitions induced in the predictor space. Take any tree $\mathcal{Y}_{X_k|\,\mathbf{X}_{\text{obs}}}$, $p < k \leq p + q$. The leaves of that tree are classes of a partition of the predictor space such that, within each class, knowledge of $\mathbf{X}_{\text{obs}}$ cannot help us in further refining our prediction of X_k (otherwise, the leave would have been split). It then makes sense that if subject i with unknown X_k ends in leave $\ell_{i,k}$ when dropped down the tree $\mathcal{Y}_{X_k|\,\mathbf{X}_{\text{obs}}}$, its X_k value be predicted by a function of the X_k values of subjects in the training sample which ended in the same leave. This function can be the mean, median or other summary statistic; or else we can sample from that leave if multiple imputations are desired.

Since we want to jointly impute all values in $\mathbf{X}_{\text{mis}}$ for the subject at hand, we would like to use complete cases in $(\ell_{i,p+1} \cap \ldots \cap \ell_{i,p+q})$, and this is exactly what the algorithm above does. The only additional caveat is that the relevant intersection might be empty —not one of the subjects in the training sample ended in exactly the same leaves than the subject to impute. If that is the case, the algorithm replaces nodes by their ancestors ("climbs the trees"),

until a non empty intersection is found. The order of climbing is governed by the deviance —we climb first the tree where the replacement of a node by its ancestor leads to the least possible increase in deviance. Inasmuch as the deviance is scale-dependent, this is only an *ad-hoc* device.

As a side issue, notice that the forest climbing algorithm was designed to *jointly* impute a vector $\mathbf{X}_{\text{mis}}$ with scalar response trees. There has been some work since on multivariate response trees (see [20]) that could be used instead.

3 An algorithm for continuous responses

The forest climbing algorithm described in the previous section has a number of desirable features and some drawbacks. Among the desirable features are the following:

1. It can adapt to any response surface and easily account for interactions.
2. Since imputation is done using one (or more) complete cases in the training sample, consistency of the imputed values is guaranteed, as in any *hot-deck* method. It is a "real donor" method in the terminology of [15].
3. The neighbourhoods —the intersections of leaves— from which we draw replacement cases are adaptive and their size and morphology are related to the responses. It is a predictive matching method.

The main drawback arises from the discrete approximation inherent to the use of binary trees: when the responses are continuous, their values are fitted by a piecewise constant function. It is natural then to explore alternatives which provide a smoother approximation of a map $R^p \longrightarrow R^q$, and several come to mind.

If we can assume an underlying model, the methods mentioned in the Introduction are of course clear favourites. In particular, data augmentation via the the S-PLUS package[1] `norm` could be used in the case of a $(p+q)$-multivariate normal distribution of predictors and responses.

If we want to stay in the realm of nonparametric or partially nonparametric procedures, the most direct generalisation of binary trees would be provided by MARS (see [6]). Other alternatives include the S-PLUS function `transcan` (in library the `Hmisc`, documented in [2], and ported also to R) or `aregImpute` (in the same library), or neural networks. The last provide about the most flexible and simple method to approximate a general $R^p \longrightarrow R^q$ mapping, and has been used in the following. The method we propose is sketched as Algorithm 1 in page 6 and briefly described in the following.

In step 2 we set two tunable parameters that we discuss below.

In step 3 we fit a neural network using sum of squared errors as a criterion to minimize. Data will have been normalized beforehand, which is also advisable to improve the convergence when training by back-propagation; see [10], Sec. 4.6. Scaled principal components can also replace the data.

[1] A port to R by Alvaro Novo is available at CRAN, `http://cran.r-project.org`.

Algorithm 1 – Predictive matching with a neural network

1: $\mathcal{C} \leftarrow \{\text{Complete cases}\}$; $\mathcal{I} \leftarrow \{\text{Incomplete cases}\}$
2: $p \leftarrow$ Tunable parameter ; $k_{\max} \leftarrow$ Tunable parameter
3: Train a neural network to approximate the mapping $\mathbf{X}_{\text{obs}} \longrightarrow \mathbf{X}_{\text{mis}}$ with the n_{obs} cases in $\mathcal{C}$. Use cross-validation to choose network topology and/or decay. Set $s \leftarrow \{\text{Number of weights in neural network}\}$ - q.
4: **for** each case $i \in \mathcal{I}$ **do**
5: Use the network to compute predicted values $\hat{\mathbf{X}}_{\text{mis}}^{(i)}$.
6: Find the set $N(i, k_{\max})$ containing the k_{max} nearest neighbours (= predictive matches) of $\hat{\mathbf{X}}_{\text{mis}}^{(i)}$ among $\hat{\mathbf{X}}_{\text{mis}}^{(j)}$ for $j \in \mathcal{C}$.
7: Let $\hat{\mathbf{X}}_{\text{mis}}^{(j_1)}, \ldots, \hat{\mathbf{X}}_{\text{mis}}^{(j_{k_{\max}})}$ be the predictive matches ordered by increasing distance to $\hat{\mathbf{X}}_{\text{mis}}^{(i)}$.
8: $\ell \leftarrow q + 1$
9: **repeat**
10: $\hat{\Sigma}_n = \sum_{l \le \ell} (\mathbf{X}_{\text{mis}}^{(j_l)} - \hat{\mathbf{X}}_{\text{mis}}^{(j_l)})(\mathbf{X}_{\text{mis}}^{(j_l)} - \hat{\mathbf{X}}_{\text{mis}}^{(j_l)})^T$
11: $\hat{\Sigma}_0 = \sum_{l \le \ell} (\mathbf{X}_{\text{mis}}^{(j_l)} - \hat{\mathbf{X}}_{\text{mis}}^{(i)})(\mathbf{X}_{\text{mis}}^{(j_l)} - \hat{\mathbf{X}}_{\text{mis}}^{(i)})^T$
12: $\ell \leftarrow \ell + 1$
13: **until** either $-2\log_e(|\hat{\Sigma}_n|/|\hat{\Sigma}_0|) > \chi^2_{s;1-p}$ or $k_{\max} \le \ell$
14: Impute case i from $\hat{\mathbf{X}}_{\text{mis}}^{(j_1)}, \ldots, \hat{\mathbf{X}}_{\text{mis}}^{(j_\ell)}$
15: **end for**

In fitting a feed-forward neural network, we must decide how many neurons to use, and in how many layers to arrange them. This can be done using cross-validation. (Same can be said of the choice of decay, to a certain extent a way to control the fit with a constant number of neurons: see [24], p. 301.)

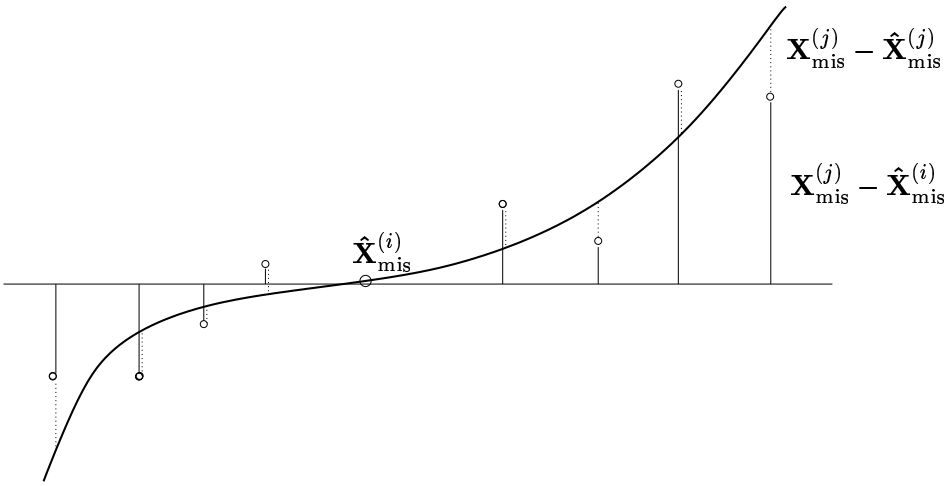
After the neural network has been trained, we generate predictions $\hat{\mathbf{X}}_{\text{mis}}^{(i)}$ for each incomplete case. A total of $k_{\max}$ near neighbours among predictions for the complete cases are selected (step 6).

The loop embracing steps 9–13 attempts to select a suitable neighbourhood. The idea, quite simple, is illustrated in Figure 5. For subsets of $q+1, q+2, \ldots$ predictive matches increasingly distant from $\hat{\mathbf{X}}_{\text{mis}}^{(i)}$ we compute $\hat{\Sigma}_0$ (the moment matrix of said complete cases about $\hat{\mathbf{X}}_{\text{mis}}^{(i)}$) and $\hat{\Sigma}_n$, a moment matrix measuring the dispersion of the near neighbours of $\hat{\mathbf{X}}_{\text{mis}}^{(i)}$ *about the respective fits as given by the neural network.* Therefore, $\hat{\Sigma}_n$ is (after suitable scaling) an estimate of the covariance matrix of the errors, while $\hat{\Sigma}_0$ is a rescaled estimate of the moment matrix about $\hat{\mathbf{X}}_{\text{mis}}^{(i)}$ of *both the signal and the error.*

We have to select the size of the neighbourhood (or number of neighbours) such that $E[\hat{\mathbf{X}}_{\text{mis}}^{(j)}] \approx E[\hat{\mathbf{X}}_{\text{mis}}^{(i)}]$. We pick those points for which $\hat{\Sigma}_0$ is not "much larger" than $\hat{\Sigma}_n$. As a criterion,

$$-2\log_e(|\hat{\Sigma}_n|/|\hat{\Sigma}_0|)$$

Figure 5: The thick line represents the neural network fit. $\hat{\mathbf{X}}_{\text{mis}}^{(i)}$ is the fit for case i. For predictive matches in the vicinity, the distances $\mathbf{X}_{\text{mis}}^{(j)} - \hat{\mathbf{X}}_{\text{mis}}^{(i)}$ (solid lines) and $\mathbf{X}_{\text{mis}}^{(j)} - \hat{\mathbf{X}}_{\text{mis}}^{(j)}$ (dotted lines) are shown. The respective sums of squares for varying number of neighbours are $\hat{\Sigma}_0$ and $\hat{\Sigma}_n$.



is used, and we check whether it is above a constant *c*. As a (very rough) rule of thumb, we have used in the examples below $c = \chi^2_{s;1-p}$ with $p = 0.05$ (the 0.95 quantile of a central χ^2_s distribution, with degrees of freedom s set to the number of parameters in the neural network minus q).

4 An example

To illustrate the method, an $R^2 \longrightarrow R^2$ function has been evaluated over a 51×51 grid and the results corrupted by additive normal noise. The variance of the of the noise was set to have a signal to noise ratio (SNR) of 10. A sample with 2601 observations was used to train a neural network with one layer of three hidden neurons and linear output units. All programming was done within the statistical and graphical package R (see [11]). We used the library `nnet` (see [24]) for the training of the neural network.

When the network was trained (with three hidden neurons, minimizing the sum of squared errors and with linear output units), a pattern $(X_1, X_2) = (0.2, 0.2)$ was presented and $k_{\max} = 60$ nearest neighbours selected. These are represented in the two top panels of Figure 6. Notice that at $(0.2, 0.2)$ the contour levels of $X_3 = f(X_1, X_2)$ and $X_4 = f(X_1, X_2)$ are oriented differently. Predictive matching tends to pick matches along a contour level; here, a compromise gives a nearly circular neighbourhood.

When we select the size of the neighbourhood as described before (loop repeat-until in steps 9–13 of Algorithm 1), with $p = 0.05$) a much smaller set of nearest neighbours was chosen (two bottom panels of Figure 6).

Figure 6: Contour levels of the "true" map $(X_1, X_2) \longrightarrow (X_3, X_4)$. Superimposed, $k_{\max} = 60$ predictive matches for point $(X_1, X_2) = (0.2, 0.2)$ (top panels) and those selected with $p = 0.05$ in (bottom panels). SNR=0.10. No locality in the input space enforced.

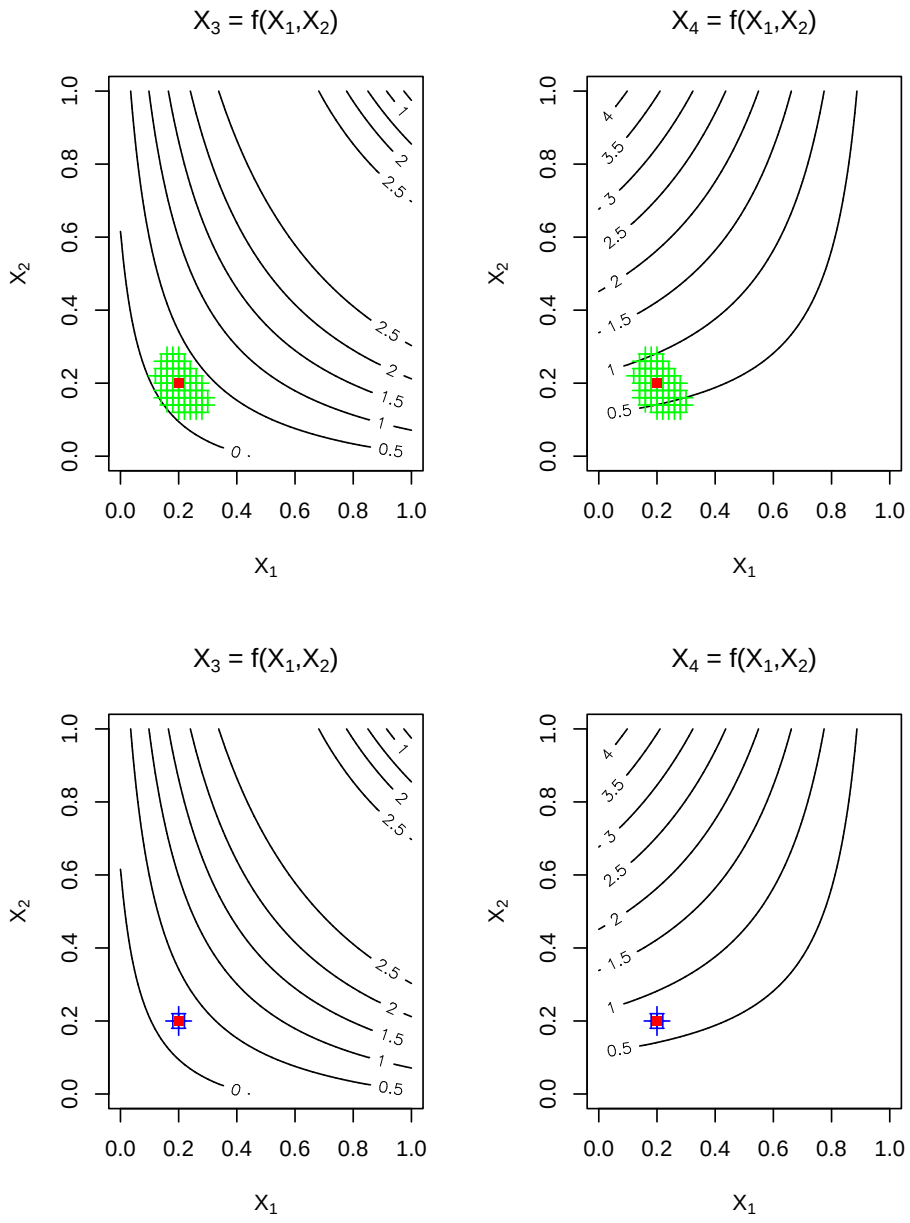


Figure 7 displays the behaviour of the method when presented with pattern $(X_1, X_2) = (0.9, 0.1)$. In the surrounding region, X_4 is quite flat while X_3 is not. The $k_{\max} = 60$ nearest neighbours selected are chosen along the contour level of X_3.

Since the match is performed using only the predicted values of the responses, it occurs in this case that a few "nearest neighbours" are selected which are quite distant in the (X_1, X_2) space (upper right corner of all four panels in Figure 7). This may be thought undesirable, and possible remedies are discussed below.

5 Discussion

A method has been presented that is well suited to the imputation of continuous variables (predictors may just as well be continuous, categorical or a mixture thereof). We have pointed already to its advantages: flexibility, near automatic character, absence from distributional assumptions, possibility to draw multiple imputations and therefore to assess the variability introduced in the imputation process. We now discuss the relationship of the method to similar ones that have been proposed in the literature, the shortcomings and possible ways to remedy them.

5.1 Related or similar work

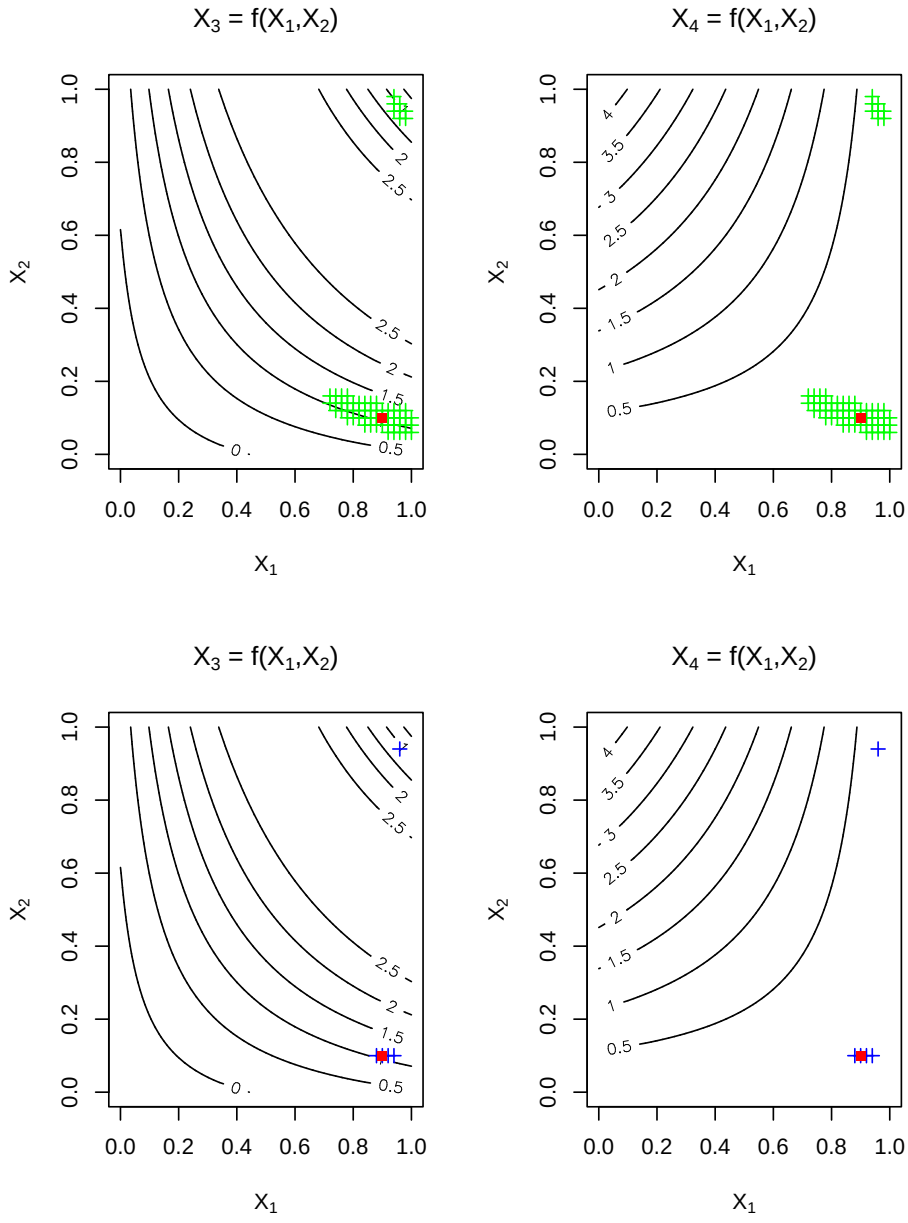
Nearest neighbour imputation has been extensively used. The use of trees to generate imputation cells has also been proposed (e.g. in [3], [4], or [14]). It has the appeal of defining subsets of donors whose "nearness" is partially or totally predictive: case i is close to case j if their respective predictions $\hat{\mathbf{X}}^{(i)}_{\text{mis}}$ and $\hat{\mathbf{X}}^{(j)}_{\text{mis}}$ are close, in the sense that they cannot be told apart using information in the predictor set. This happens with cases falling in the same leaf of a tree. We call this a "response induced topology" in the predictor space $\mathcal{X}$.

Similar ideas arise also in other contexts. For instance, the propensity score (see [17], [8]) has been proposed as a criterion for stratification and matching. [12] propose an approximate Bayesian bootstrap, sampling within classes homogeneous with respect to the propensity score of being missing (but see [1] for a critique).

Using predictive matching plus some sampling scheme has also been studied, for instance in [19]: they propose to use predictive matching followed by sampling residuals of near observations (LAD, "local residual draw"); we prefer taking a donor (or more) and use its values to impute $\mathbf{X}^{(i)}_{\text{mis}}$ *en block*, since this guarantees consistency. They also propose and adaptive method to choose the number of possible donors. They do not address the problem of non-locality in the $\mathcal{X}$ space of predictive matches.

In a post to the IMPUTE list, [9] proposed an strategy for imputation which also bears close resemblance to the method proposed in Section 3: rather than adjusting the size and orientation of the neighbourhood from which to draw donors, he suggested transforming suitably the predictors and

Figure 7: Contour levels of the "true" map $(X_1, X_2) \longrightarrow (X_3, X_4)$. Superimposed, $k_{\max} = 60$ predictive matches for point $(X_1, X_2) = (0.9, 0.1)$ (top panels) and those selected with $p = 0.05$ (bottom panels). SNR=0.10. No locality in the input space enforced, as shown by the few matches at the upper right corner of each panel.



responses and using neighbourhoods of fixed width ϵ (but no hints were given on the choice of ϵ).

5.2 Shortcomings and possible solutions

Non-locality of matches in $\mathcal{X}$. It has been mentioned before that predictive matching gives, on occasion, matches which are quite far in the $\mathcal{X}$ space of predictors from the case to be imputed. This is annoying. One possible solution is to look for closest matches in a different space. When imputing case i, rather than looking for complete cases j with small distance $d(i,j) = ||\hat{\mathbf{X}}_{\text{mis}}^{(i)} - \hat{\mathbf{X}}_{\text{mis}}^{(j)}||^2$ we can look for complete cases with small

$$d(i,j) = (1-\lambda)||\hat{\mathbf{X}}_{\text{mis}}^{(i)} - \hat{\mathbf{X}}_{\text{mis}}^{(j)}||^2 + \lambda||\mathbf{X}_{\text{obs}}{}^{(i)} - \mathbf{X}_{\text{obs}}{}^{(j)}||^2,$$

which forces the matches to be close in *both* $\mathcal{X}$ and the space of predictions. Fairly small values of λ suffice in the experiments we have done, but of course this is one more parameter to calibrate in the imputing process. Further, the problem remains that non-locality of the matches is not so easy to spot when we deal with highly dimensional spaces.

Variables of mixed type. The method presented is motivated by the poor approximation of continuous variables afforded by binary trees. Nonetheless, it is usually desirable to impute a vector $\mathbf{X}_{\text{mis}}$ not all of whose components are continuous variables. One has then to define a suitable distance, which can be done (see for instance [7]); but this hinders the use of off-the-shelf software to train the neural network involved.

Irregular patterns of missingness. One of the beauties of the algorithms presented in [18] is the flexibility with which they cope with irregular patterns of missingness, through clever use of the sweep operator. We are discussing here the simplest monotone pattern of missingness, but it would be desirable to generalize to irregular patterns. It is not clear how to do that, short of training many different networks, which seems awkward and out of the question.

Acknowledgements. I have talked many times with M.J. Bárcena, whose dissertation on survey merging methods introduced me to the subject This research supported through grants PB98-0149 (CICYT) and 9/UPV 00038.321-13631/2001. Availability of open source software, notably R and library `nnet` made the work of prototyping much easier than it would have been otherwise.

References

[1] P.D. Allison. Multiple imputation for missing data: a cautionary tale. Technical report, Available at `http://www.ssc.upenn.edu/ allison/MultInt99.pdf`, 1999.

[2] C.F. Alzola and F.E. Harrell. An introduction to S-Plus and the Hmisc and Design libraries. Technical report, Online publication, available at `http://hesweb1.med.virginia.edu/biostat/s/index.html`, 1999.

[3] M.J. Bárcena and F. Tusell. Enlace de encuestas: una propuesta metodológica y aplicación a la Encuesta de Presupuestos de Tiempo. *Qüestiió*, 23:297–320, 1999.

[4] M.J. Bárcena and F. Tusell. Tree-based algorithms for missing-data imputation. In J.G. Bethlehem and P.G.M. van der Heijden, editors, *COMPSTAT'2000. Proceedings in Computational Statistics*, Heidelberg, 2000. Physica-Verlag.

[5] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone. *Classification and Regression Trees.* Wadsworth, 1984.

[6] J. Friedman. Multivariate adaptive regression splines. *Annals of Statistics*, 19:1–41, 1991.

[7] J.C. Gower. A general coefficient of similarity and some of its properties. *Biometrics*, 27:857–872, 1971.

[8] Xing Sam Gu and P.R. Rosenbaum. Comparison of multivariate matching methods: Structures, distances, and algorithms. *Journal of Computational and Graphical Statistics*, 2(4):405–420, 1993.

[9] F. Harrell. A predictive mean matching multiple imputation strategy. Posted to the IMPUTE list on Nov, 8, 2001.

[10] S. Haykin. *Neural Networks. A comprehensive Foundation*. Prentice Hall, second edition, 1998.

[11] R. Ihaka and R. Gentleman. R: a language for data analysis and graphics. *J. of Comp. and Graphical Stats.*, 5:299–314, 1996.

[12] P. Lavori, R. Dawson, and D. Shera. A multiple imputation strategy for clinical trails with truncation of patient data. *Statistics in Medicine*, 14:1913–1925, 1995.

[13] R.J.A. Little and D.B. Rubin. *Statistical Analysis with Missing Data.* John Wiley and Sons, New York, 1987.

[14] D.M. Mesa, P. Tsai, and R.L. Chambers. Using tree based models for missing data imputation: and evaluation using UK Census data. Technical report, Department of Social Statistics, University of Southampton, 2000.

[15] P. Piela and S. Laaksonen. Automatic interaction detection for imputation - Tests with the WAID software package. Statistics Finland, 2000.

[16] S. Räessler. Alternative approaches to statistical matching. In J. Bethlehem and S. van Buuren, editors, *Missing values. Proccedings of a Symposium on incomplete data.*, pages 85–104, 2000.

[17] P.R. Rosenbaum and D.B. Rubin. Reducing bias in observational studies using subclassification on the propensity score. *Journal of the American Statistical Association*, 79:516–524, 1984.

[18] J.L. Schafer. *Analysis of Incomplete Multivariate Data*. Chapman and Hall, London, 1997.

[19] N. Schenker and J.M.G. Taylor. Partially parametric techniques for multiple imputation. *Computational Statistics and Data Analysis*, 22:425–446, 1996.

[20] R. Siciliano and F. Mola. Multivariate data analysis and modeling through classification and regression trees. *Computational Statistics and Data Analysis*, 32:285–301, 2000.

[21] T.M. Therneau and E.J. Atkinson. An introduction to recursive partitioning unsing the RPART routines. Technical report, Mayo Foundation, 1997.

[22] S. van Buuren and C.G.M. Oudshoorn. Multivariate imputation by chained equations. MICE v1.0 user's manual. Technical report, TNO Prevention and Health, 2000.

[23] S. van Buuren and K. Oudshoorn. Flexible multiple imputation by MICE. Technical Report PG/VGZ/99.054, TNO Prevention and Health, Public Health, POB 2215, 2301 CE Leiden, October 1999. Available from `http://www.multiple-imputation.com`.

[24] W.N. Venables and B.D. Ripley. *Modern Applied Statistics with* S-PLUS. Springer-Verlag, New York, third edition, 1999.