Missing Data Models as Meta-Data

Russell G. Almond and James Schimert
StatSci Division, MathSoft*

October 31, 1994

1 Introduction

Missing data are ubiquitous. The common practice is to throw away those parts of the data which have missing values. This practice not only wastes information, but may also lead to biased inferences unless the data are *missing completely at random*, i.e., the subjects with complete data are a simple random sample of all of the subjects in the data set. Examples where this assumption breaks down are numerous. In surveys, for example, wealthy people may be most likely to skip questions about income.

To go beyond simply omitting missing values, the computer requires an explicit model for the missing values, using information from either external sources or the completely observed data. These missing data models are *meta-data* about the variables. Given the missing data model, standard analysis procedures should adapt themselves to properly account for the missing data. This paper describes our prototype representation of missing data meta-data in the S-PLUS language.

First, this paper briefly describes two model based methods for analyzing missing data: multiple imputation and the Expectation—Maximation (EM) algorithm. Next, we review the current model for S-PLUS analysis in the framework described by Anglin and Oldford[1994]. We then extend that model to cover analysis with missing data and describe our progress in implementing this model.

2 Model Based Missing Data Methods

There have been numerous techniques proposed for handling missing data: imputation (Little and Rubin [1987], Rubin [1978, 1987]), the EM algorithm (Dempster, Laird and Rubin [1977]), Markov Chain Monte Carlo (MCMC, Smith and Roberts [1993]), data augmentation (Tanner and Wong [1987]), and various specialized techniques. All of these techniques require specifying a model, usually explicitly, for the missing values.

We will describe two of these methods—multiple imputation and the EM algorithm—in more detail because they exemplify two ways of how models and data are combined to handle missing values.

*Address for correspondence: StatSci, 1700 Westlake Ave N., Seattle, WA 98109 (almond@statsci.com).

2.1 Imputation and Multiple Imputation

One alternative to omitting rows of the data frame is to *impute*, that is estimate and fill in, values for the missing observations. Imputations are based on a model for the missing observations. Little and Rubin [1987] suggest a number of different methods for imputation. Heitjan and Little [1991] review several desirable properties for general-purpose imputations:

- Imputes of missing values should *condition* on the values of observed variables for that case.
- They should account for the *multivariate* nature of the non-response, i.e., values are missing on more than one variable, with a general pattern of missing data.
- Imputations should not distort marginal distributions and associations between observed and imputed variables. To achieve this, they should be *stochastic*, i.e., they should represent values from the predictive distribution, rather than just the means, of the missing variables.

For these reasons, many ad hoc imputation techniques are not sound. In particular, *univariate* imputation schemes, i.e., specifying a missing data model separately for each variable with missing values, should not be used except in special circumstances where the multivariate nature of non-response does not matter. One example of this is when the missing data fit a *monotone pattern* (Rubin [1987], Rubin and Little [1987]), so the imputation model factors into a series of univariate conditional models.

Multivariate imputation techniques include:

Hot Deck Select from an estimated distribution for each missing value. In applications, this may consist of drawing values from completely observed subjects deemed to be similar (by comparing on the observed values) to the case with missing values.

Iterative Simulation Straightforward imputation is achieved by first specifying a parametric probability model for the complete (missing and observed) data and a prior distribution for the parameters, and then simulating values from the conditional distribution of the missing data given the observed data. Complex patterns of missingness, however, make it difficult and often intractable to directly simulate these predictive distributions. However, recent advances in Bayesian computation, in particular iterative simulation schemes based on Markov chains (see e.g. Geman and Geman [1984], Gelfand and Smith [1990]), have made it possible to generate imputations under a variety of useful models for both continuous and categorical data, and a mixture of these (Schafer [1991, 1995]).

Each of these imputation methods can be described by a "missing data model", in particular an "imputation model". Although the interpretation differs for hot deck vs. iterative simulation imputation, in either case the imputation model refers to the information, besides the data itself, required to carry out imputation. This information can be contained in an object which is attached as a parameter to a data frame or given as an argument to a modeling function. We briefly describe the nature of this information for hot deck and simulation–based imputation in turn.

For the hot deck, the model is implicit in the description of the procedure. There are several discrete choices involved in the choice of a hot deck method, e.g. Mahalanobis metric matching vs. matching on a set of predetermined factors, and there are also some control parameters that may be set in some versions of the matching algorithm, e.g. relative weights to be given to different variables, "similarities" among levels of a categorical variable, or choice of variables for which

exact matching is compelled. Another input to a hot deck could be a supplementary data set to be used as a donor pool.

In contrast to the "nonparametric" hot deck, in simulation–based imputation, we can specify the joint multivariate model for all of the data by formulas (this is meant in a specific sense made more precise below). In the S-PLUS software, we have employed the General Location Model with constraints (Krzanowski [1980,1982] and Little and Schluchter [1985]), which is the centerpiece of Schafer's [1991, 1995] work on missing data imputation. In fact, Joe Schafer's software forms an algorithmic "engine" around which we have created class objects and methods to implement the S-PLUS modeling paradigm. Once the joint distribution is specified (details follow below) and fitted, all of the conditionals may be obtained for imputation.

Single imputation takes the data frame with missing values and produces an effectively complete data frame (*i.e.*, has no missing values). One advantage of imputation is that, after imputation, the analyst can use familiar complete data methods, drawing on his experience with those methods. Different analysts might perform the imputation and analysis steps. Much of the development of the imputation methodology was sponsored by the Census Bureau. They assumed that the data supplier (the Census Bureau) would perform the imputations and their customers would used the imputed data for their own purposes. In general, a data supplier may have access to knowledge about the missing data mechanism and hence be better equipped to create completed data sets for public use by a varied audience. In this case, the analyst need not learn any new missing data techniques. Especially in this situation, it is desirable that statistical software handle the imputed data sets in a manner that does not require the analyst to act differently than with complete data sets.

The biggest drawback of applying usual inference procedures to a single imputation procedure is that this pretends that the missing data were known all along, and therefore ignores the additional variability in the model parameters induced by imputation. If we let μ represent our model parameters, and Y_{miss} the missing observations, then we have that:

$$Var(\mu) = E[Var(\mu|Y_{miss})] + Var(E[\mu|Y_{miss}]) \; .$$

Note that the first term is simply the variance of our parameter estimates reported by the complete data analysis. Single imputation implicitly assumes that the second term is zero.

Multiple imputation (Rubin [1978]) addresses this problem by taking a sample of possible values for the missing values, *i.e.*, imputing $M > 1$ times instead of just once. The between–imputation variance in the parameter estimates from the M resulting filled-in data frames supplies the correction term for the variance of the parameter estimates. Rubin claims a dramatic improvement for even modest values of M (*e.g.*, 2, 3 or 5).

Note that the result of multiple imputation is several effective complete data frames. In order to integrate multiple imputation with the rest of the analysis software, we must expand our analysis software (1) to accept data frames with additional multiple imputation data, and (2) to properly aggregate the results of the multiple complete data analyses.

2.2 EM Algorithm

The EM algorithm is a likelihood-based approach to handling missing data. Let $Y = (Y_{obs}, Y_{miss})$ be the *complete* data. We would like to maximize $l(\theta|Y)$, the log-likelihood of the complete data. However we cannot, because of the missing data. Instead, supposing that our best current estimate of the parameters is $\theta^{(t)}$, we create (E step) and maximize (M step) with respect to θ

$$Q(\theta|\theta^{(t)}) = \int l(\theta|Y) f(Y_{miss}|Y_{obs}, \theta^{(t)}) dY_{miss} \quad (1)$$

the average (over the missing values) of the complete log-likelihood, weighted by the distribution of the missing data (given the observed data and the current estimate of the parameter). This procedure is iterated until convergence—one of the optimality characteristics of the EM algorithm is that the likelihood increases at each iteration.

For the complete exponential family of distributions, this algorithm consists of calculating the expected values of the sufficient statistics, then performing the usual maximization for complete data. This is close to the intuitive practice of iteratively imputing missing values, and then performing a complete data analysis. In fact, this intuitive idea underlies a variety of missing data algorithms including iterative simulation methods, the data augmentation methods of Tanner and Wong [1987], and sequential imputation (Kong, Liu, Wong [1991]). Multiple imputation simulates approximate draws from the predictive distribution f and thus approximates the averaging in (1). Note that, for the exponential family of distributions, when the log-likelihood is linear in the data, the E step of the EM algorithm is equivalent to imputing the data.

3 Complete Data Analysis

The goal of our research is to extend the S-PLUS modelling and analysis environment in a natural way to handle model–based missing data techniques. This section looks at the existing environment for complete data analysis. We start with a theoretical model proposed by Anglin and Oldford[1994] and then explain the existing S-PLUS environment in terms of that framework. Section 4 shows how we must extend that framework to include missing data methods.

3.1 The Anglin and Oldford Model

Anglin and Oldford[1994] describe a general class of model called the *response model*. This class is defined by the common characteristic that one variable is chosen as the *response* whose expected value (and variance) can be expressed as a function of a number of *explanatory variables*. The relationship between the explanatory and response variables is typically described in terms of a *formula* whose terms are functions of the response variables.

The simplest subclass of the response model is the gaussian linear model. Here the formula is linear in the explanatory variables and the response variables are assumed to be normally distributed about their expectations with a constant variance. More complex classes of models, such as the general additive model, can include a *link function* which represents a transformation of the formula to produce the expected response, a *family* for the distribution of the errors, *weights* to adjust for unequal variances and non-linear formulae. Both because the more general models can be thought of as variations on the linear model, and because the linear model encompasses a large number of useful models, we have chosen it as the focus for our prototype design and implementation.

Many important data analysis tasks involve comparing two models. For example, we may want to compare the fit of a model with and without a particular explanatory variable to understand the relationship (if any) between the explanatory variable and the response. Typically, this is done through the formula part of the model specification. The analyst adds and removes terms from the model searching for the best fit. Almond[1994] discusses some interaction metaphors which specifically support the model comparison task. The key idea of Anglin and Oldford[1994]

is that there are two parallel class hierarchies for models. The first is the model hierarchy described above; the second is the *fitted model* hierarchy.

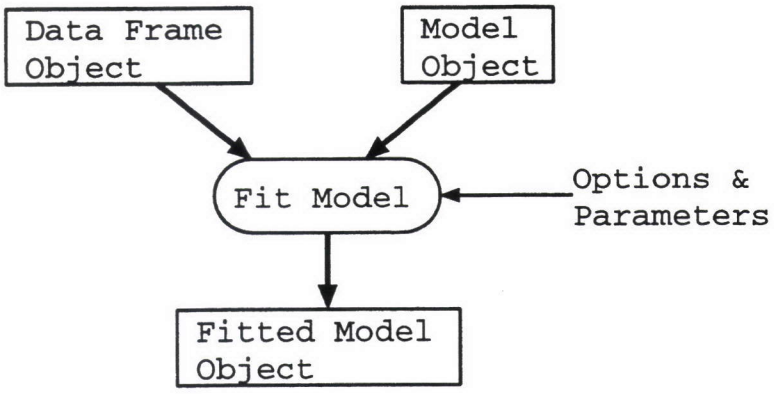

Figure 1: Fitting a model object to a data frame produces a fitted model object.

Each of these objects supports a different set of generic functions for many common analysis tasks. All of the object support some very general functions like `summary()`, `plot()`, and `print()`. Functions like `add1()` and `drop1()` manipulate the formula part of the model object. The fitting function takes a model object and a data object and produces a fitted model object. These fitted model objects include information needed for many important functions. Functions like `coef()` and `deviance()` return statistics from the fitting procedure; the `fitted()` and `residuals()` function return the predicted values for the response variable and the differences from the predictions which play an help detect and diagnose lack of fit. The `predict()` function predicts value for the response variable for new realizations of the response variables.

Note that combining models and data to produce fitted models is a powerful idea. Cleveland [1993] extends this idea to produce general purpose plotting routines. These routines take a model object (a formula) and a data frame and produce a plot which should show how the response variable(s) behave as a function of the explanatory variables.

3.2 The Key Objects in the S-PLUS Implementation

Becker, Chambers and Wilks[1988] describe the New S environment for data analysis. New S is an interactive statistical programming language which supports statistical, graphical and mathematical analyses of data; S-PLUS is an extended version of New S commercially supported by MathSoft (StatSci). Although S-PLUS supports a wide variety of statistical techniques, its direct support for missing data methods is limited; the only general method provided is a function which omits all cases with missing values from a data frame.

An important feature of the S-PLUS language is that any object can have "attributes" as well as data (similar to the "properties" of Lisp symbols). For example, row and column variable names are implemented as attributes of the matrix object. For a long time, S-PLUS functions have used this feature to bundle the output of functions into compound objects which can be manipulated by other functions. Note that the class of an object (which S-PLUS uses to compute the dispatch for generic functions) is also implemented using attributes. Thus S-PLUS is weakly object-oriented: it supports both composite data objects and polymorphic functions, although it does not use class definition meta-objects or perform any type-checking or optimization.

The current version of S-PLUS (based on Chambers and Hastie[1992]) supports statistical

models as first class objects: objects which can be passed and manipulated by functions as if they were data. The basic kind of model object, the *formula* object, implements the Wilkinson and Rogers[1973] language for model specification. The S-PLUS implementation (Chambers and Hastie[1992]) extends the language, and uses it to fit a variety of models: least squares (regression), analysis of variance, generalized linear models, generalized additive models (including splines), local regression (loess) models, and tree based models.

Data Frames

One of the simplest conceptual models for data is the spreadsheet. In a data spreadsheet, rows of the table correspond to observations (*e.g.*, individuals in a survey) and the columns correspond to *variables*—measurements on the observations (*e.g.*, questions in a survey).

An S-PLUS *data frame* object is a representation of such a data spreadsheet. In one view it is a matrix of rows and columns of data; in the other view, it is a list of variables. Having a list of variables allows S-PLUS to store variable specific meta-data with each variable.

Currently, S-PLUS uses the variable specific meta-data to customize the behavior of certain procedures. For example, an important piece of meta-data is whether a variable is a continuous or categorical (a factor). The `lm()` function will adapt its behavior to the types of the explanatory variables in the model: if they are all factors it will perform an analysis of variance, if they are all continuous it will perform a regression. If an analyst has interest in specific contrasts between levels of a factor, he can specify them as variable specific meta-data and S-PLUS will adapt its behavior appropriately. Analogously, we propose using missing data models as meta-data to choose the appropriate algorithm. This use of variable specific meta-data is not unique to S-PLUS; Hand[1993] and Roth *et. al.*[1994] propose other uses.

Formulae and Models

Wilkinson and Roger[1973] introduced a language for specifying linear models. This language is a shorthand for complex model equations. Thus, to represent the model:

$$Y_i = \beta_0 + \beta_1 X_{1,i} + \beta_2 X_{2,i} + \epsilon_i,$$

we use the simpler form: $Y \sim X1 + X2$. The coefficients β_0, β_1 and β_2 are implicit as is the error term ϵ_1; the intercept, β_0, is implied by default. This allows for a much more compact notation and an easy way of describing the difference between models.

Formulas support four different types of explanatory variables: `factors`—variables whose values represent choices from a fixed set (*e.g.*, {true, false}, {red, blue, green}), `ordered factors`—variables whose values represent choices from an ordered set, numeric values—vectors or integers or real numbers, and matrices—a compact notation for several variables. Factors are treated slightly differently in formulas: the will expand to a number of coefficients (corresponding to the number of levels of the factor). The rules for handling formulas (described in Chambers and Hastie[1992]) take advantage of the natural difference between the way one would like to model factors and numeric vectors.

Within the scope of a formula, the normal operators are interpreted somewhat differently. For example, the + operator adds terms to the model and the - operator removes terms from the model (this is typically used only when describing the difference between two models or to explicitly remove the intercept). The : operator is used to add interaction terms into a model; the * operator

adds both the linear and the interaction terms and the / operator performs a special kind of nested interaction.

S-PLUS also implements fitted model objects. Model fitting functions, like `lm()`, `glm()` and `gam()` take a data frame and a formula as input and return a fitted model object as output. A wide variety of generic functions have specific methods for the model objects, including extraction functions for the inputs to the fitting routine call, extraction functions for statistics produced by the model, summary and diagnostic graphics functions and especially the `update()` function. This latter function allows the user to change the formula and refit the model. This is an important part of the model selection stage of analysis.

Note that although S-PLUS specifically supports data frame object and fitted model objects, it does not fully support model objects. The formula is only part of a model. For all classes of response model currently supported by S-PLUS, we could add attributes to the formula object to produce models which would sufficiently specify the intended fitting procedure.

Details of Model Fitting in S-PLUS

Our work aims to implement missing data procedures by combining models and data. The goal is to seamlessly integrate missing data handling procedures into the modeling functions so that, from a user's perspective, specifying a model in the presence of missing data is like modeling complete data.

In the case of imputation, the S-PLUS modeling paradigm needs to be implemented in at least two places:

- combining the imputation model and the original data frame (with missing values) to produce completed data.
- applying the usual modeling procedures to the imputed data frame.

In either case, we need to know the details of S-PLUS model fitting functions, which we now briefly examine. The `lm()` function (and other similar functions) proceeds in 4 steps: (1) `model.frame()`—This function prepares a reduced data frame for fitting the model by removing unneeded variables and then selecting a subset of cases, if a subsetting arguments is given. In the current S-PLUS implementation, it calls the `na.action` function if any rows contain missing values. (2) `model.matrix()`—This function prepares a matrix of explanatory variables based on the formula. It creates interaction terms and dummy variables (and contrasts) for factor variables. (3) The actual model fitting. (4) Collect the results and return the fitted model object.

4 Missing Data Analysis

The current S-PLUS implementation leaves precisely one place for adding missing data handling to the model fitting paradigm: within the model fitting code, the `model.frame()` function calls the `na.action` function which should return a data frame without missing values. Although this works adequately for procedures such as omitting the missing values or even for *single* imputation, it does not adequately serve the needs of more general procedures such as multiple imputation.

Figure 2 more properly captures the flow of the analysis using multiple imputation. First, the analyst imputes new values for the missing data using the missing data model. This step implicitly fits the parameters of the missing data model. Note however that the *primary* goal is to combine data and imputation model to produce other data sets, rather than a fitted model object. As in

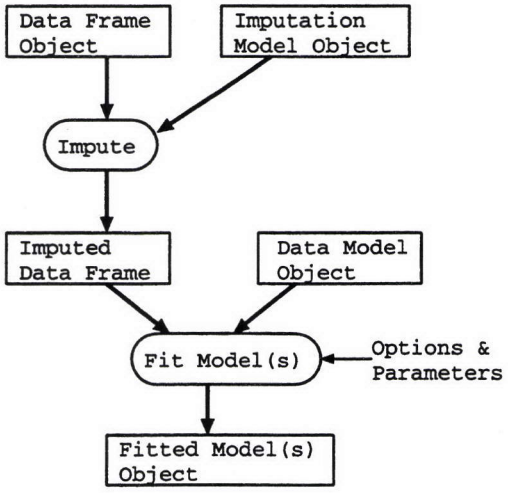

Figure 2: Expanding the model fitting model to include imputation.

any other model fitting procedure, this step typically involves an iterative process of exploratory data analysis, model identification, model fit, and diagnosis. These steps are, however, guided by diffent criteria than in the analysis stage. For example, since accurate prediction of missing values is more important than achieving a parsimonious model, variable selection may be of secondary importance (usually having an imputation model which is more general than the analysis model leads to sound inferences – although the opposite may not be true). After multiple imputation, one may graphically explore the completed data sets to informally assess how interesting features of the data are affected by missing–data uncertainty.

In the second – complete data analysis – step, the analyst applies the model fitting procedure to the completed data, i.e., the imputed data frame. Note that different analysts might perform the imputation and analysis steps.

4.1 Missing Data Model

All missing data handling procedures require assumptions about the missing values. For example, omitting the rows of the data frame with missing observations assumes that the observations are *missing completely at random.* Other "non-parametric" techniques like hot-deck imputation assume that the missing observations are similar to the observed values for some variable in a *local neighborhood* (where the neighborhood is defined in terms of the completely observed variables) of the subject with missing values. However, defining the local neighborhoods requires making choices about the relative scaling of the variables. In all cases, we should study robustness and sensitivity to these assumptions: making the assumptions explicit helps us do this.

We use a separate model for the missing values and the response variables for a number of reasons. (1) The missing data model must provide a model for all column variables with missing observations, not just the response variable. (2) Often we want to include additional variables in the missing data model (proxies for the missing observations) which are not of direct interest the inference stage. (3) While parsimony is a concern when creating inferential models, it is not as great a concern when fitting missing value models. (4) We may want to separate the imputation and analysis steps.

An Imputation Model

One particularly useful class of missing data models is the General Location Model (Olkin and Tate [1961], also known as the conditional gaussian model (Lauritzen and Wermuth[1989]). This model can describe the joint distribution of variables that are either factors (i.e., discrete or categorical variables) or continuous variables. Following the usual terminology of categorical data analysis, the factors form a contingency table of *cells*. The General Location Model assumes that:

- the cell counts follow a multinomial distribution, and
- conditioned on the cells, the continuous variables follow a multivariate normal distribution. The cell means may vary across cells, but the cell covariances are assumed to be the same for all cells.

When there are just two cells, the General Location Model reduces to the familiar model for discriminant analysis.

The large number of parameters in this model make it difficult to fit to many datasets, especially those for which the number of factors is large. In this latter case, the contingency table (formed by the factors) may be too sparse to estimate either the probabilities for the cells, or the mean vectors within the cells. The analyst may impose either loglinear constraints on the cell probabilities and/or ANOVA-like constraints on the cell means. The loglinear constraints, as in hierarchical interaction models (HIM, Edwards[1990]), require that for every interaction term in the model, all of the lower order interaction terms involving the same variables and all of the corresponding main effects terms also be included in the model.

The constraints define the model, and therefore the imputation model under a General Location Model can be expressed by two formulas, one for each constraint. The ANOVA–like constraints on the cell means are easy to express using the syntax for linear models in S-PLUS, i.e. the Wilkinson and Rogers[1974] language. Because of the linear relationship between the log of the cell means and the factor variables, HIMs can be represented either with the Wilkinson and Rogers[1974] language, or the HIM model language developed by Edwards[1990].

Shafer[1991, 1995] has created software for multiple imputation based on the General Location Model (allowing constraints). Note that the General Location Model can specify the distribution for a mix of factors and continuous variables. Thus, we only need to specify one model for the entire data frame. Naturally, the variables may be all be continuous (multivariate normal model) or all factors. The nature of the variables and the formula will determine the appropriate algorithm.

4.2 Analysis with Multiple Imputation

In order to implement missing data analysis with multiple imputation, we must extend two of the objects in the original data analysis model (Figure 1): the data frame and the fitted model object. Figure 2 gives the resulting data analysis model.

The *imputed data frame* object differs from a standard data frame by the addition of two pieces of meta-data: the model under which the values were imputed, and the actual imputed values. Note that several of the generic functions must be extended to accommodate these new data frame objects. For example, plotting functions might plot a small cloud of imputed values in place of the missing observations. Of course, the most important extension is to the model fitting functions, which now must be generic and which dispatch on the type of data frame: complete data or with imputed values.

The modelling functions, such as `lm()` must be extended as follows: (1) The function `model.frame()` which produces the reduced data sets must operate properly on imputed value data frames. Next, for each of the M completed data sets, the model fitting function must (2) Calculate the data matrix (`model.matrix()`) and (3) fit the model using the complete data method. Finally, the model fitting function must gather the results of the fitting procedures into an aggregate model object.

Several of the generic functions which operate on the fitted model objects must also be extended to properly summarize the multiple fits returned by the multiple imputation methods. In particular, the variance calculations for coefficients and predictions need to be adjusted for the uncertainty about the model fit caused by the missing observations.

Note that once the analyst has performed the imputations, the modelling functions use the meta–data associated with the imputed data object to automatically adjust the behavior of complete data procedures. From the user's perspective, modelling with missing data is not much different from modelling with complete data.

4.3 Analysis with the EM Algorithm

Other missing data methods would interact differently with the modelling process. For example, the EM algorithm assumes a model for the *complete data*; the result of running EM is a set of parameter estimates. There is no preliminary step, as in imputation, of assuming an imputation model to produce completed data sets, which are then analyzed using analysis models. Therefore, changes needed to modeling functions such as `lm()` are minor. One can create an `na.keep()` missing data handler which simply keeps the original missing data symbols, i.e., `NA`s. Then the existence of `NA`s in the data frame causes `lm()` to choose an EM algorithm (rather than using a particular matrix decomposition method). The meta–data for an EM algorithm is therefore the same as the meta–data used to fit the analyst's model.

5 Conclusions and Future Directions

Working with explicit model objects has a distinct advantage: the computer can select the appropriate algorithm for fitting the model to the data. Associating missing data models with a data frame should have similar advantages: the computer can select the appropriate algorithm for handling missing values.

We have chosen to explore two missing data methods—multiple imputation and the EM algorithm— because they typify two different ways of specifying a missing data model.

In multiple imputation, there are two distinct steps: (1) impute (i.e. estimate and fill in missing data) and (2) analyze the resulting complete data set(s) using the usual complete data methods. This requires an explicit *imputation model* which, combined with data, produces an *imputed data* object. The nature of this object determines that a method for multiply imputed data sets is chosen in the subsequent complete data analysis.

In contrast, the model for the EM algorithm is the same as the analysis model, therefore the modeling software need only be changed by adding another algorithm which is invoked if there are missing values in the data object.

Extending the modelling functions to handle imputed values gets around the *missing completely at random* assumption. However, it still contains the milder *missing at random* assumption which says that missing values are a random sample of all values within subclasses defined by observed data. Models without this assumption are difficult to fit because we would need to actually find

the true values of the missing observations to fit them. On the other hand, there are some relatively simple alternative, such as adding an inflation factor for missing values (*e.g.*, missing values are approximately 10% higher than the corresponding non-missing values). Using models such as these, we can at least study the sensitivity of our conclusions to our assumptions about the missing data mechanism.

6 Acknowledgements

The authors acknowledge support under NIH SBIR grant R43 CA65147-01, whose proposed research was originally motivated through work with Gary Donaldson under a Fred Hutchinson service agreement. Trevillore Raghunathan, Joseph Schafer, and Alan Zaslavski gave insight and references to the literature in many stimulating exchanges. We are building our S-PLUS software around the routines (written in Fortran and S-PLUS) developed by Joe Schafer for the General Location Model.

References

Anglin, D.G. and R.W. Oldford[1994]. "Modelling Response models in Software." in P.Cheeseman and R.W.Oldford (eds.) *Selecting Models from Data: Artificial INtelligence and Statistics IV*, Springer-Verlag, 413–424.

Becker, Richard A., Chambers, John M., and Wilks, Allan R. [1988]. *The New S Language: a programming environment for data analysis and graphics.* Wadsworth and Brooks/Cole, Pacific Grove, California.

Chambers, John M. and Hastie, Trevor J. [1992]. *Statistical Models in* S . Wadsworth & Brooks/Cole, Pacific Grove, CA.

Cleveland, William S.[1993]. *Visualizing Data.* Hobart Press.

Dempster, A.P. Laird, N.M. and Rubin, D.B.[1977]. "Maximum likelihood from incomplete data via the EM algorithm." *JRSS, Series B*, (**39**), pp 1–38.

Edwards, David[1990]. "Hierarchical Interaction Models." *Journal of the Royal Statistical Society, Series B*, **52** pp. 3–20

Gelfand, A. E. and Smith, A. F. M.. "Sampling–Based Approaches to Calculating Marginal Densities", *Journal of the American Statistical Association*, (**85**), No. 410.

Geman, S. and Geman, D. [1984]. "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (**6**), 721–741.

Hand, David J. [1993]. "Measurement scales as metadata", in Hand, D.J. (ed.) *Artificial Intelligence Frontiers in Statistics: AI and Statistics III.* Chapman and Hall, 54–64.

Heitjan, D. F. and Little, R. J. A. [1991]. "Multiple Imputation for the Fatal Accident Reporting System", *Aplied Statistics*, (**40**), No. 1, 13–29.

Krzanowski, W. J. [1980]. "Mixtures of continuous and categorical variables in discriminant analysis", *Biometrics*, (**36**), 493–499.

Krzanowski, W. J. [1982]. "Mixtures of continuous and categorical variables in discriminant analysis: a hypothesis–testing approach", *Biometrics*, (**38**), 991–1002.

Kong, A., Liu, J., Wong, W.H. [1991]. "Sequential Imputations and Bayesian Missing Data Problems." *Technical Report No. 321*, University of Chicago.

Little, R. J. A. and Schlucter, M. D. [1985]. "Maximum Likelihood Estimation for Mixed Continuous and Categorical Data with Missing Values", *Biometrika*, (**74**), 497–512.

Lauritzen, Steffen L. and N. Wermuth [1989]. "Graphical models for associations between variables, some of which are qualitative and some quantitative." *Annals of Statistics*, **17**, 31–57. Corrections on 1916.

Little, Roderick J.A., Rubin, Donald B. [1987]. *Statistical Analysis with Missing Data.* John Wiley and Sons.

Olkin, I. and Tate, R. F. [1961]. "Multivariate correlation models with mixed discrete and continuous variables", *Ann. Math. Statist.*. **32**, 448–465.

Roth, S.F., J. Kolojejchick, J. Mattis and J. Goldstien[1994]. "Interactive Graphic Design Using Automatic Presentation Knowledge." In *Human Factors in Computing Systems: CHI '94 Conference proceedings*, ACM Press, 112–117.

Rubin, D.B. [1978]. "Multiple Imputations in sample surveys: A phenomenological Bayesian approach to nonresponse." *Proceedings of the Survey Research Methods Section*, American Statistical Association, pp 20–24.

Rubin, D.B. [1987]. *Multiple Imputation for Nonresponse in Surveys*, John Wiley and Sons.

Schafer, J. L. [1991]. Harvard Ph.D. Thesis: *Algorithms for Multiple Imputation and Posterior Simulation from Incomplete Multivariate Data with Ignorable Nonresponse*

Schafer, J. L. [1995]. *Analysis of Incomplete Multivariate Data by Simulation*, Chapman and Hall.

Smith, A.F.M. and Roberts, G.O. [1993]. "Bayesian Computation via the Gibbs Sampler and Related Markov Chain Monte Carlo Methods." *J. R. Statist. Soc. B (1993)*, (**55**), No. 1.

Tanner, M. A. and Wong, W. H.[1987]. "The calculation of posterior distributions by data augmentation (with discussion)." *JASA*, (**52**), 528–550.

Wilkinson, G. N. and Rogers, C. E. [1973]., "Symbolic Description of Factorial Models for Analysis of Variance." *Applied Statistics*, **22**, 392–399.