Modelling response models in software

D.G. Anglin* and R.W. Oldford†
Department of Statistics and Actuarial Science
University of Waterloo

Abstract

We describe our software design and implementation of a wide variety of response models, which model the values of a response variable as an interpretable function of explanatory variables. A distinguishing characteristic of our approach is the attention given to building software abstractions which closely mimic their statistical counterparts.

1 Introduction

One of the great workhorses of statistical science is the response model. This model supposes that there are $p+1$ variables $x_1, \ldots, x_p$ and y whose values are observed for each of n independent realizations. Interest lies in modelling the values of the *response* variable y as an interpretable function of the *explanatory* variables $x_1, \ldots, x_p$. The explanatory variables are taken to be fixed at their observed values – either because they are under our control and the response is observed after their setting, or because we choose to condition the response on the observed values of the explanatory variables. Once fitted to data, the model is an interpretable summary of the dependence of the response on the explanatory variables as well as a predictor of values of the response for any values of the explanatory variables.

The most widely used response model is the linear model, which expresses the response as a function of the explanatory variables and a random disturbance ϵ. This function is linear in its unknown parameters α and $\beta_1, \ldots, \beta_p$ and is typically written as

$$y = \alpha + \beta_1 x_1 + \cdots \beta_p x_p + \epsilon. \tag{1}$$

In its simplest form, the model takes each ϵ to be an independent realization from some distribution having zero mean and finite variance σ^2. When the distribution of ϵ is also assumed to be the Gaussian distribution, we will call this the *Gaussian linear model*.

Software for this model has long been the mainstay of statistical packages. The fitting algorithm is simply least-squares (or equivalently maximum likelihood when ϵ is Gaussian) and requires only data corresponding to the response and explanatory variables.

*Supported by a Postgraduate Scholarship from the NSERC.
†Supported by grants from the NSERC.

Summary results run the gamut from printed tabular output to interactive graphics for model criticism and exploration.

As computational resources have grown, so too has the response model. The relatively simple linear model has been generalized to response models which are at once more computationally intensive and more flexible. Even so they retain much of the interpretability of the linear model. We review some of these models in Section 2.

Good statistical modelling is a largely iterative process. Many models might be selected, examined, and discarded before the analyst settles on some hopefully small set of competing models worth reporting (if any). Statistical computing environments must be designed to support this process of data analysis and modelling.

In what follows we describe our software design and implementation of a wide variety of response models. A distinguishing characteristic of our approach is the attention given to building software abstractions which closely mimic statistical counterparts (see *eg.* [Oldford 87]). That is, we intend to build software models of response models. The response models we consider are reviewed in Section 2 and our software is described in Section 3. Section 4 relates a very short example of the use of the software. In Section 5 we describe our approach in relation to that of others. Section 6 contains some discussion of implementing other response models not covered in previous sections and of implementing statistical models in general.

2 Response Models

As described earlier, by a response model we mean one which designates some variables (typically one) as response variables and the remainder as fixed explanatory variables. We suppose further that values of all response variables are independent from one realization to the next. Of interest then, is inference about the conditional distribution of the response variables given the explanatory variables.

The linear model equation (1) is such a model. To emphasize this we rewrite the Gaussian linear model (1) as

$$\begin{aligned} y|x_1,\ldots,x_p &\sim N(\mu,\sigma^2) && (2)\\ \mu &= \alpha+\beta_1 x_1+\cdots\beta_p x_p. && (3) \end{aligned}$$

Here, $N(\mu,\sigma^2)$ denotes the Gaussian distribution with mean μ and variance σ^2. σ^2 is the unknown conditional variance of y and the explanatory variables enter the model only through the conditional mean μ of y.

An important natural generalization is to replace the Gaussian distribution by one that is more appropriate for the response. For example, if each realization of the response is a proportion of people who respond to some medical treatment, we may wish to use a Binomial distribution with mean proportion π. However, modelling the mean π as a linear combination of the explanatory variables may lead to estimated values of π outside of $[0,1]$. The logit, or log-odds, defined as $\log(\pi/(1-\pi)) = \alpha+\beta_1 x_1+\cdots\beta_p x_p$ is one possible alternative. This is the logistic regression model and is a special case of an extension due to Nelder and Wedderburn [NeldWedd 72] called a *generalized linear model* (see also [McCuNeld 89]).

A generalized linear model is determined by three relations:

$$\begin{aligned} y|\boldsymbol{x} &\sim f(\mu, \phi) && (4)\\ \eta &= \boldsymbol{x}^T\boldsymbol{\beta} && (5)\\ g(\mu) &= \eta && (6) \end{aligned}$$

for explanatory variables $\boldsymbol{x} = (1, x_1, \ldots, x_p)$ and parameters $\boldsymbol{\beta} = (\alpha, \beta_1, \ldots, \beta_p)$. Now the response is a random variable whose (conditional) distribution $f(\mu, \phi)$ is known to be a member of a restricted exponential family. Instead of σ^2, this more general setup has a dispersion parameter ϕ (which may be known; see [McCuNeld 89] for further detail). The *family* $f(\cdot)$ induces a function $V(\cdot)$ such that the variance of y is $\phi V(\mu)$. However, it is no longer the conditional mean μ which is a linear function but rather the function η. The two are related through a *link function* $g(\cdot)$. The class can be extended to so-called quasi-likelihood models by not specifying $f(\cdot)$ completely but by only asserting its mean and its variance (as a function of the mean).

In the binomial proportion example, y is a binomial proportion with conditional mean $\mu = \pi$ and link function $g(\mu) = \log(\mu/(1-\mu))$. This latter quantity is modelled by some linear combination of the explanatory variables, η.

We note that η and consequently μ are functions of the explanatory variables $\boldsymbol{x}$. To emphasize this point we will sometimes write these as $\eta(\boldsymbol{x})$ and $\mu(\boldsymbol{x})$. Typically all functions $f(\cdot)$, $\eta(\boldsymbol{x})$, and $g(\mu(\boldsymbol{x}))$ are specified; interest lies in the choice of values for $\boldsymbol{\beta}$.

A different way in which the linear model (1) has been generalized is to write y as a sum of smooth functions $s_j(\cdot)$ affected by random disturbance

$$y = \alpha + \sum_{j=1}^{p} s_j(x_i) + \epsilon. \qquad (7)$$

This model is called the *additive model* [HastTibs 90]. It retains the advantage of easy interpretation of the linear model, but adds the increased flexibility of some nonlinear impact of changes in the x_j's upon the value of y, with the unknown $s_j(\cdot)$'s now playing the role previously belonging to the β_j's.

The additive model and the generalized linear model are brought together in *generalized additive models* [HastTibs 90]. These use the additive term

$$\eta(\boldsymbol{x}) = \alpha + \sum_{j=1}^{p} s_j(x_j) \qquad (8)$$

of the additive model and otherwise have identical structure to the generalized linear model.

3 Implementation

Our software implementation of the wide class of response models consists of software components which mimic very closely the statistical concepts they represent. The object-oriented facilities of Common Lisp [Steele 90] are used to create distinct *classes* of objects which represent the statistical concepts discussed in Section 2. The strength of this

approach is that the statistical meaning and relationships between these different response models is preserved and enforced through the class structures and their accompanying functions. This clarifies the software for both the user and the developer.

As can be seen from the above discussion, response models are built from interpretable components. Consequently we use a variety of *model object* classes whose components mimic those of the response model. In particular, the class of a response model object is determined by its systematic component as represented by the model formula $y \sim \eta(\boldsymbol{x})$, by the specific functional forms of $\eta(\boldsymbol{x})$, by its stochastic component describing the nature of the random variation in y, and by the link function $g(\cdot)$ which joins them.

The variables involved in a response model and the functional form of $\eta(\cdot)$ are represented by *formula objects*. This describes the systematic part of the model. The formula $\eta(\boldsymbol{x})$ is directly related to a function $\mu(\boldsymbol{x})$, where $\mu(\boldsymbol{x})$ describes the property of the response which we are modelling. Stochastic variation in y for a given $\boldsymbol{x}$ is described in terms of $\mu(\boldsymbol{x})$, though perhaps differently for different classes of response model. This random component of the model, corresponding to (4), is represented by a *family object*. Interpretation of the model formula amounts to specifying the relationship between the systematic component, described by $\eta(\boldsymbol{x})$, and the random component, described in terms of $\mu(\boldsymbol{x})$ [McCuNeld 89]. This relationship $g(\mu(\boldsymbol{x})) = \eta(\boldsymbol{x})$ is the link function (6), and is represented by *link objects*.

Using these various components we create a *hierarchy* of model objects. For example, we represent in software the fact that linear models are a special case of generalized linear models by making the class linear-model a *subclass* of the class generalized-linear-model. Subclasses *inherit* properties of their superclasses, since any particular instance of the subclass is also an instance of the superclass. For example, both generalized-linear-model and linear-model have a link component (6), but in the case of the linear model (3), the link $g(\cdot)$ is restricted to be the identity.

Parallel to the hierarchy of models will be *model fits* — as distinct from *models*. Model fits, represented by *fit objects*, result from fitting a model object to a *data object* using a certain *fitting procedure*.

3.1 Formula objects

Response models have the characteristic that one variable, the *response* y, is separable from the remaining variables, the *explanatory variables* $\boldsymbol{x} = (x_1, \ldots, x_p)$. We take to be common among statistical models the existence for each model of a *model structure* which identifies the variables in the model and at least to some extent describes the *systematic* relationship between them. The structure of the response model, in this sense, is provided by a *model formula* $y \sim \eta(\boldsymbol{x})$.

This structure is represented by class response-formula, which has slots identifying the variables involved in the model, identifying the response variable from amongst these, and specifying a function $\eta(\cdot)$ of the explanatory variables.

It is clear from Section 2 that different types of function $\eta(\cdot)$ will play an important role in specifying the structure of a response model. To represent the restrictions on the response-formula for the generalized additive model, we define the subclass additive-formula, instances of which enforce the requirement that $\eta(\boldsymbol{x})$ is of form (8). The further subclass linear-formula represents the more specific case (5) used in generalized linear models, linear

models, and Gaussian linear models.

3.2 Family objects

A class family-object represents in the abstract the *stochastic* component of the model. In order to facilitate dispatching to appropriate methods, specific distributions $f(\cdot)$ each have their own subclass of family-object with the following properties:

- The name of the family
- A variance function which produces the variance in this family for a single observation with mean μ
- A deviance function which produces the contribution to the deviance of an observation y with expected mean μ.

A top-level variable with the same name as the family for f is defined which is the sole instance of these subclasses. For example, for errors binomially distributed about the mean $\mu(\boldsymbol{x})$, there exists a subclass of family called binomial-family with name "Binomial", variance function $V(\mu) = \mu(1-\mu)$ and deviance function

$$D(\mu, y) = -2(y \log \mu + (1-y)\log(1-\mu)),$$

and a variable binomial-family bound to an instance of class binomial-family.

3.3 Link objects

Link objects follow the same strategy as family objects. Subclasses of link-object correspond to specific link functions $g(\cdot)$, have slots containing the name of the link, the link itself, and the inverse $g^{-1}(\cdot)$ of the link. The link and the link-inverse are represented by a class of function objects which stores information on derivatives and other function properties; in particular, first derivatives of the link and link-inverse, which are useful in fitting the model, are available. In the same way as for specific distributional families, there is defined a top-level variable bound to an instance of the specific subclass of link of the same name. For example, there is a link-object subclass called logit-link with name "Logit" and link function object $g(\mu) = \log(\mu/(1-\mu))$, and a variable logit-link bound to an instance of this subclass.

3.4 Model objects

The software representation of a particular response model is an object of class response-model, which possesses the slot structure. This slot represents the relationship between the explanatory variables and the response, and is restricted to contain an object which is a response-formula.

The specific examples of response models which we described earlier have more structure yet, and these will be represented in software by subclasses of response-model. Using the objects we have just outlined, the hierarchy below response-model is straightforward to describe (see Figure 1). Class generalized-additive-model has slots structure (inherited

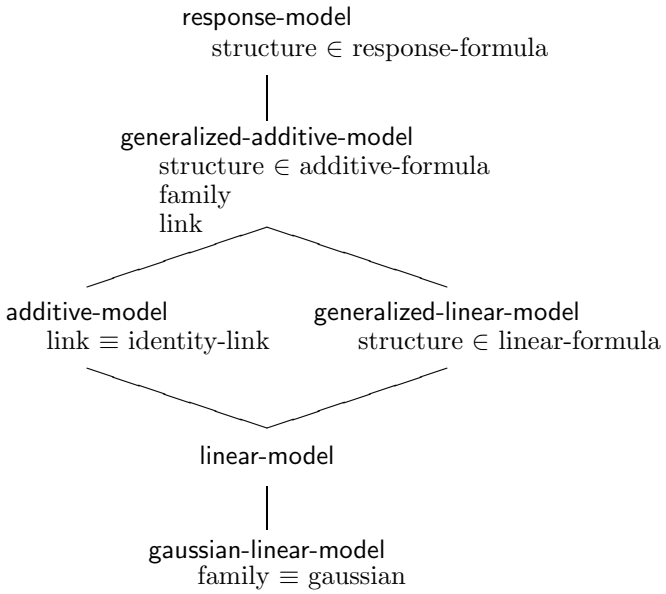

Figure 1: Class hierarchy of response models

from response-model), family, and link representing the systematic, the random, and the link components of the model. For generalized-additive-model the structure slot must contain an instance of class additive-formula. Slot family must have a specific family-object subclass instance as its value; similarly for slot link.

The characteristic feature of additive models as a subclass of generalized additive models is that the link function is the identity function. The link subclass identity-link represents link $g(\mu) = \mu$, and the link slot for class additive-model always has the value identity-link. An alternative specialization of generalized additive models is to further restrict the class of formulae. By restricting the contents of slot structure to instances of class linear-formula, we create the class generalized-linear-model.

The class which has both additive-model and generalized-linear-model as superclasses is the classical linear-model, since this class inherits both the property that the link is the identity, *and* the property that $\eta(\cdot)$ is linear. The further special case of a Gaussian family gives rise to the common gaussian-linear-model.

The class of a model object can be used by generic functions to dispatch to the most computationally efficient methods for a particular model class. For example, the fitting procedure for a gaussian-linear-model will be more computationally efficient than for more general models. Certainly we would write a method which applies to this special case for the generic function that performs the fit. This behavior is accentuated by careful creation of instances: if a program requests an instance of a linear-model which has the gaussian-family, the appropriate gaussian-linear-model is returned; similarly, a request for a generalized-linear-model instance with the identity-link will produce a linear-model instance.

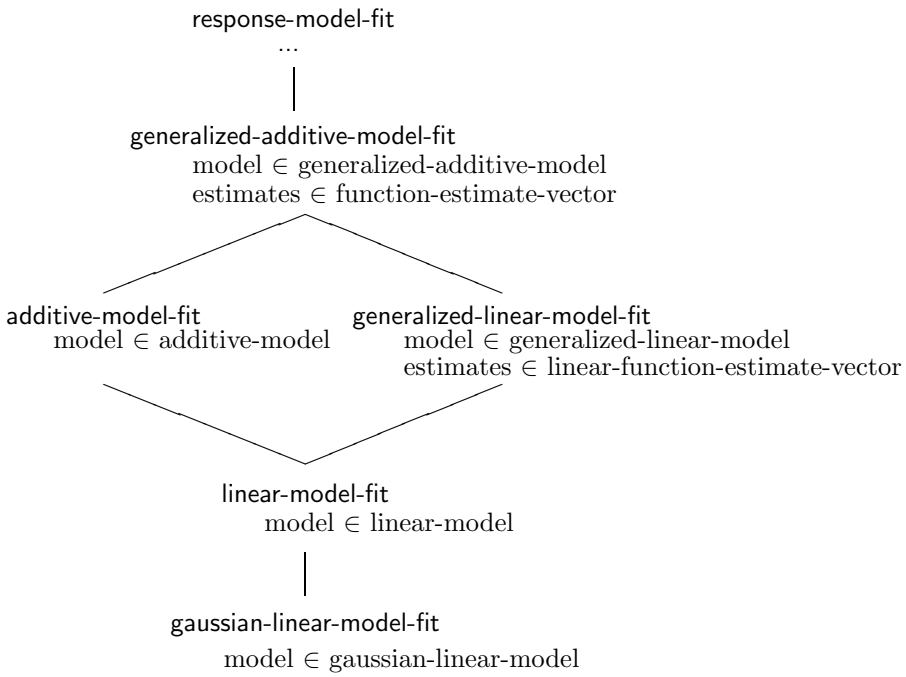

Figure 2: Class hierarchy of response model fits

3.5 Fit objects

The choice of a model class for a certain problem involves careful consideration on the part of the analyst of the type and structure of the data. Utilizing a suitable class is important to valid conclusions, and in many cases the data themselves may indicate that the selected model is inappropriate. Accordingly not only model summaries but also model assessment techniques are vital tools of the data analyst.

A major benefit of interactive statistical programming enviroments is they simplify, and can even encourage, the iterative data analysis procedure of choosing a model class, selecting and fitting a model from within that class, assessing the model, and when necessary, selecting a different model or even a different model class.

To this end, we define in parallel to our model hierarchy a hierarchy of *model fits*, which represent the fit of a model object to a *data* object by some *fitting procedure*. The fit object contains fundamental quantities important to interpretation and assessment of the fit, and there is collection of standard mathematical and graphical devices available for this purpose which accept fit objects as input.

Residuals, and quantities derived from them, are central players in model assessment. For the response models we've discussed above, there are a number of different residual quantities which have been used [McCuNeld 89, PierScha 86]. The generic function residuals is capable of providing any of these from a fitted model object.

Models are often summarized by numerical quantities such as the fitted coefficients and standard errors, and residual sums of squares. Some useful numerical summaries for a given model fit are provided by the summary generic function, which returns an object

of the appropriate subclass of summary-object.

There are also a variety of useful plots for model assessment within the models we've been discussing, and plots appropriate for a model under consideration can be produced from an object representing a fit of that model.

4 An example

Landwehr, Pregibon, and Shoemaker [LanPreSh 84] present a generalized linear model analysis of long-term survival of 306 breast cancer patients after surgery [Haberman 76]. The data consist of

- Survival, a binary variable which is 1 if the patient survived 5 or more years after surgery, 0 otherwise
- Age, the age of the patient at the time of surgery
- Year, year of the patient's surgery (minus 1900)
- Nodes, number of positive axillary nodes detected in the patient.

The response variable y_i is Survival for patient i, and we take $\boldsymbol{x}_i = (x_{1i}, x_{2i}, x_{3i})$ to be Age, Year, and Nodes for patient i, respectively. Since the response is binary, $\mu(\boldsymbol{x}) = E(y|\boldsymbol{x}) =$ probability that a patient with explanatory variable $\boldsymbol{x}$ will survive five or more years ($y = 1$). As a first step in analysis, we might consider a generalized linear model with

$$\eta(\boldsymbol{x}) = \alpha + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$$

with the logit link $g(\mu) = \log(\mu/(1-\mu))$. The appropriate family is binomial. Supposing that Cancer is a data structure to which our variables Survival, *etc.*, are meaningful, we can fit this as a generalized linear model by

```
(setf cancer-glm-1 (glm  "Survival  ~  Age + Year + Nodes"
                         Cancer
                         :family binomial-family
                         :link logit-link))
```

which returns a generalized-linear-model-fit object. Note that the keyboard expression of $\eta(\boldsymbol{x})$ omits the parameters. This is in keeping with the Wilkinson and Rogers model specification notation [WilkRoge 73]. The notation and its extensions (see [ChamHast 92]) are a convenient (and common) way to specify response models.

The above command structure hides much of the implementation. Closer examination would reveal that the function glm requests that the generic function model return a generalized-linear-model object, which object has a structure slot containing a linear-formula object appropriate to the string provided. In particular, for our example, the linear-formula object has slots identifying Survival as the response variable, and Age, Year, and Nodes as explanatory variables.

Once glm has a model-object in its possession, it passes this object, the data, and a fitting procedure (maximum likelihood, by default) to a generic function fit. Based on the

formula from the model, the fit procedure requests the data it needs by name from the data object, performs the fitting procedure, and returns the appropriate fit-object. This object is subsequently returned by glm. In our example, the returned object cancer-glm-1 is of class generalized-linear-model-fit, and can be queried for the estimates $\hat{\alpha}, \hat{\beta}_1, \hat{\beta}_2, \hat{\beta}_3$ of the parameters, for values $\hat{\eta}(x)$ and $\hat{\mu}(x)$, and for the total deviance of the fitted model, the latter obtained by summing $D(\hat{\mu}(x_i), y_i)$ over all patients i.

We can use fitted model objects like cancer-glm-1 as input to summaries and plots. If we do partial residual plots for cancer-glm-1, for example, we observe that the relationship between Nodes and the partial residuals for Nodes is quite nonlinear, suggesting a transformation may be appropriate. Exploration of Nodes and the other explanatory variables yields a more complex model involving a logarithmic transformation of Nodes ([LanPreSh 84]). We could alternatively have tried using generalized additive models via the function gam.

5 Related work

In traditional statistical systems models have had no explicit software representation. They are defined at each call for a fit. Sometimes there is an implicit current fitted model which can be examined and changed by adding and deleting terms from its structure (*eg.* GLIM [BakeNeld 78]). In other systems, many fitted models may exist simultaneously but the model is again defined implicitly as part of the fit (*eg.* [AbraRizz 88], [BecChaWi 88]). A notable early exception to this approach is the econometric modelling system TROLL ([TROLL 82]) which in fact predates the common interactive systems mentioned above. In TROLL, models are separate data structures involving many equations which relate variables (endogenous and exogenous distinguished), parameters, and random quantities. They can be fitted by various procedures and, because the random structure is specified, once fitted these models are used to simulate future outcomes. Our approach to model representation is closer in spirit to TROLL than to the more common statistical systems.

In distinguishing model fits from the models themselves, we depart from other authors. Consider the approach taken in the book *Statistical Models in S* (*SMS*) [ChamHast 92]. In *SMS*, there is a hierarchy of model fits corresponding to ours. A fit of the appropriate class is constructed and returned by a model-specific fitting procedure, such as glm, based on arguments specifying the formula, the data, and family and link objects. However, at no point is there created a model object, *per se*, and consequently there is no model hierarchy. This of course precludes the development of procedures which operate exclusively on models (*eg.* nesting operations, combination operations, comparison operators, *etc.*). The formulae of *SMS* have class formula, but this class specializes no further. By contrast a hierarchy of formula classes plays a major role in the definition of our model hierarchy. Section 6 below suggests extensions of formula to cover other statistical models.

Various flavours of object-oriented programming have been used to build software representations for statistical concepts (*eg.* see [Oldford 87], [Pedersen 91], [Tierney 91], [ChamHast 92], [Oldford 90], and the references therein). *SMS* is the first comprehensive treatment of software representations of statistical response models and has certainly influenced our work (particularly on summaries of model-fits). Perhaps the major distinction between our implementation and that in *SMS* or [Tierney 91] is a different interpretation

of object-oriented programming. In our implementation, all hierarchies of objects begin with the most general of objects, and grow downwards through progressively more specific classes. 'Downward' in this context refers to the direction of inheritance — from superclass 'down' to subclass. This is the classic approach described for example in SmallTalk-80 [GoldRobs 83]. In *SMS* and [Tierney 91], the hierachies are reversed: conceptually more general classes inherit from the more specific ones. In particular, generalized-linear-model is a subclass of linear-model. A strong argument against this nonstandard subtyping when designing a class hierarchy can be found in [HalbO'Br 87]. Because linear models are conceptually special cases of generalized linear models, we have chosen to have linear-model appear as a dependent of generalized-linear-model. Then any instance of a linear-model behaves exactly as a generalized-linear-model should the user wish it.

6 Other statistical models

We think of a statistical model as a relationship between variables that involves some *stochastic* (or *random*) component(s). To capture this intentionally vague description, we define a class called model-object to be the top of our model class hierarchy. The class response-model previously discussed is a direct subclass of model-object.

The model-object class has a single slot called structure whose contents represent the known relationship between the variables. Often the defining characteristic of more specific model classes will be a more restricted class for the contents of structure. In particular, the class response-model is a subclass of model-object for which the slot structure is a formula that separates the response variable from the explanatory variables (*i.e.* a response-formula). But for other models structure might be something more complex: a graph representing the joint distribution of the variables (*eg.* [Whittake 89]); a system of differential equations for structural equations models (*eg.* [TROLL 82]); and so on.

Other statistical models fit naturally into this framework. The class of tree-based models (*eg.* [BrFrOlSt 84]), for example, would be a subclass of response-model. Other examples include non-linear regression [BateWatt 88], locally-weighted regression [ClevDevl 88], and multivariate adaptive regression splines [Friedman 91]. Implementation of subclasses of model-object which are not response models remains to be explored.

7 Concluding remarks

Constructing software models which match the corresponding statistical models in organization makes the software easier to understand. In particular, a coherent hierarchy of model classes naturally representing their conceptual counterparts can be used easily by other code from a variety of levels. While an interface at the level of the glm command of Section 4 is possible even when the software is not structured this way, a less superficial implementation encourages evolution of more sophisticated interaction [OldfPete 88]. One can write clear and simple procedures which operate on objects in the way in which we usually think of them. The class of individual objects in this environment in large part defines what operations may be performed on them.

Formal software representations of abstract concepts like data, model, estimation procedure, and fit permit their manipulation in novel ways. Some examples illustrate the

point. A TROLL model can be estimated, decomposed, linearized, simulated, and bootstrapped [TROLL 82]. Bates and Chambers [BateCham 87] also describe a bootstrap resampling procedure in this context: given a model instance, an estimation procedure, and a set of sample data sets, the bootstrap function estimates the model for each data set in the set of sample data sets. In *SMS* there are tools for model searching which use formula objects to bound the search.

With others we are developing a statistical system based on matching software constructs to statistical analysis concepts. We anticipate that the opportunity for further research in this area is substantial.

References

[AbraRizz 88] Abrahams, D.M. and F. Rizzardi (1988) *BLSS: the Berkeley interactive statistical system*. W. W. Norton & Company, New York, NY.

[BakeNeld 78] Baker, R.J. and J.A. Nelder (1978) *The GLIM System, Release 3, Generalized Linear Interactive Modeling*. Numerical Algorithms Group, Oxford.

[BateCham 87] Bates, D.M. and J.M. Chambers (1987) *Statistical Models as Data Structures* AT&T Bell Labs Statistical Research Report. 6 pages.

[BateWatt 88] Bates, D.M. and D.G. Watts (1988) *Nonlinear Regression Analysis and its Applications*. John Wiley & Sons, New York, NY.

[BecChaWi 88] Becker, R. A., Chambers, J. M., and Wilks, A. R. (1988) *The New S Language: A Programming Environment for Data Analysis and Graphics*. Wadsworth & Brooks/Cole, Pacific Grove, CA.

[BrFrOlSt 84] Breiman, L., Friedman, J.H., Olshen, R., and C.J. Stone (1984) *Classification and Regression Trees* Wadsworth International Group, Belmont, CA.

[ChamHast 92] Chambers, J.M. and T.J. Hastie (1992) *Statistical Models in S*. Wadsworth & BrooksCole, Pacific Grove, CA.

[ClevDevl 88] Cleveland, W.S. and S.J. Devlin (1988) Locally-weighted Regression: An Approach to Regression Analysis by Local Fitting. *JASA* **83**, 596-610.

[Friedman 91] Friedman, J.H. (1991) "Multivariate Adaptive Regression Splines", *Ann. Stat.* **19**, 1-141.

[GoldRobs 83] Goldberg, A. and Robson, D. (1983) *Smalltalk-80. The Language and Its Implementation*, Addison-Wesley, Reading, MA.

[Haberman 76] Haberman, S.J. (1976) "Generalized Residuals for Log-Linear Models", *Proc. 9th Intl. Biometrics Conf., Boston*, 104-122.

[HalbO'Br 87] Halbert, D. C. and O'Brien, P. D. (1987) "Using Types and Inheritance in Object-Oriented Programming," *IEEE Software*, Sept. 1987, 71-79.

[HastTibs 90] Hastie, T.J. and R.J. Tibshirani (1990) *Generalized Additive Models*. Chapman and Hall, London.

[HurlOldf 89] Hurley, C.B. and R.W. Oldford (1989) "A Software Model for Statistical Graphics," Technical Report STAT-89-13 (University of Waterloo, Department of Statistics and Actuarial Science, Waterloo, ON). Also appears in: *Statistical Computing and Graphics*, A. Buja and P.A. Tukey, eds., 77-94, Institute for Mathematics and its Applications, University of Minnesota (1991).

[LanPreSh 84] Landwehr, J.M., Pregibon, D., and A.C. Shoemaker (1984) "Graphical Methods for Assessing Logistic Regression Models", *JASA* **79**, 61-71.

[McCuNeld 89] McCullagh, P. and J.A. Nelder (1989) *Generalized Linear Models* (Second Edition). Chapman and Hall, London.

[NeldWedd 72] Nelder, J.A. and R.W.M. Wedderburn (1972) "Generalized Linear Models" *JRSS (A)* **135**, 370-384.

[Oldford 87] Oldford, R. W. (1987) "Abstract Statistical Computing," *Bulletin of the International Statistical Institute: Proceedings of the* 46^{th} *session*, **52**, Book 4, 387-398.

[Oldford 90] Oldford, R. W. (1990) "Software Abstraction of Elements of Statistical Strategy," *Annals of Mathematics and Artificial Intelligence*, **2**, 291-308.

[OldfPete 88] Oldford, R.W. and S.C. Peters (1988) "DINDE: Towards more sophisticated software environments for statistics," *SIAM Journal on Scientific and Statistical Computing*, **9**, 191-211.

[Pedersen 91] Pedersen, J. (1991) "Situations, Summaries, and Model Objects," in *Statistical Computing and Graphics*, A. Buja and P.A. Tukey, eds., 139-185, Institute for Mathematics and its Applications, University of Minnesota (1991).

[PierScha 86] Pierce, D.A. and D.W. Schafer (1986) "Residuals in Generalized Linear Models" *JASA* **81**, 977-986.

[Steele 90] Steele, G. (1990) *Common LISP: The Language* (Second Edition). Digital Press.

[Tierney 91] Tierney, L. (1991) "Generalized Linear Models in Lisp Stat". *Technical Report No. 557*, School of Statistics, University of Minnesota.

[TROLL 82] *TROLL Documentation*. Technical Report from the Center for Computational Research in Economics and Management Science, MIT, Cambridge, Massacusetts.

[Whittake 89] Whittaker, J. (1989) *Graphical Models in Applied Multivariate Statistics*. John Wiley & Sons, Chichester, England.

[WilkRoge 73] Wilkinson, G.N. and C.E. Rogers (1973) "Symbolic Description of Factorial Models for Analysis of Variance" *Applied Statistics* **22**, 392-399.