HModel: An X Tool for Global Model Search

George Ostrouchov

Mathematical Sciences Section, Oak Ridge National Laboratory, Oak Ridge, Tennessee 37831-6367, U.S.A.

Abstract

HModel is a global model search tool that has X window graphics capability. The user defines the largest generalized polynomial model to consider, *HModel* generates the collection of all smaller hierarchical models that it contains and displays a graphical representation of the collection. Then, given a model optimality criterion threshold *HModel* finds all models that are at least as good. That is, *HModel* classifies all models into two groups: those at or below the threshold and those above the threshold. If the set of models at least as good as the threshold is small, finding this set requires fitting only a small fraction of all models. The selection of models to fit can be automatic or conveniently controlled by the user. A search string facility for selecting sets of models as well as other set operations are also available. All of these functions interact with the graphical representation of the collection of models. The current implementation and this paper focus on log-linear models for categorical data and the Akaike and Bayes information criteria although the underlying ideas are more general. The search can manipulate a collection of up to about 200,000 models on a high performance workstation. For example, six categorical variables give 140,035 possible models of order three or less.

1 Introduction

The global model search tool *HModel* finds the best models as defined by a given optimality criterion in a given collection of models. The collection of models can potentially be any family of generalized polynomial models. A broad range of optimality criteria is possible. Model search is restricted to models that are *hierarchical*.

A model term is a subset of model variables, so that for p variables there are 2^p possible model terms. The class of *hierarchical* models contains models in which higher-order terms are included only if lower-order marginal terms are included. For example, if a polynomial model contains the product term x_1x_2, it must also contain x_1, x_2, and the constant term in order to be hierarchical. Generalized polynomial models are of the form

$$y \doteq f(p(x)), \tag{1}$$

where y is the response variable, $f(\cdot)$ is a known function, $p(x)$ is a polynomial in x with unknown coefficients, and x is a vector of explanatory variables. In model selection among generalized polynomial models it is preferable to consider only those models that are hierarchical. The estimation space and thus model selection criteria are invariant to location and scale transformations of the explanatory variables if and only if the model

is hierarchical (Peixoto [0]). This is important because the selected models should be the same regardless of units that are used. Furthermore, the collection of hierarchical models is many orders of magnitude smaller than the collection of models constructed from all possible subsets of model terms and thus presents a more manageable search problem. For example, with six categorical variables we have roughly 18×10^{18} models that contain all possible subsets of terms and only 7.8×10^6 possible hierarchical models.

The present implementation of *HModel* and this paper focus on log-linear models for categorical data (see [0]). The available model optimality criteria are the Akaike and Bayes information criteria (see [0]). When the explanatory variables in (1) are 0-1 variables, as in the case of log-linear models for categorical data with maximum likelihood estimation, a hierarchical completion of a model (by inclusion of missing marginal terms) has the same estimation space as the original model.

Representation of the collection of models and the algorithm for generating this collection as well as the model search algorithms are based on Ostrouchov and Frome [0]. A brief summary is in § 2 but further details appear in [0]. The interface to X is discussed in § 3 and § 4 discusses some current work on extensions to *HModel* and its availability.

2 Representation and Algorithms

Given the largest model to be considered, which we call the root model, we generate all hierarchical models that it contains. Because a model is a set of model terms, the subset relation $\subset$ imposes a partial ordering on a set of hierarchical models. A partial ordering can be represented by a directed graph, where the nodes are the models and the edges give the ordering. We use the smallest directed graph that represents this partial ordering (containing the least number of edges), which is called the *cover graph* of the partial ordering. Any two models that are neighbors in this representation differ by exactly one model term.

Representation of a single hierarchical model by the set of its maximal terms or by the set of its minimal terms set to zero is discussed in [0]. We refer to these as the maximal and the dual representation, respectively. A smaller hierarchical neighbor of a model is obtained by deleting one of its maximal terms (see [0] for an illustration and more details). For this reason, each edge of the cover graph representation is associated with a model term. Maximal representation of a model at a node of the cover graph is then easily obtained from the edges that connect this node to its smaller neighbors. Similarly, model terms in the dual representation are exactly the terms that are added to obtain larger hierarchical models; thus the dual representation of a model is obtained from the edges that connect this node to its larger neighbors.

The data structure representing the collection of models thus consists of two types of objects. A *node* that holds several attributes of a model (such as its deviance, number of parameters, and a few status indicators), and a pointer to an array of edges that define its neighbors as well as the model at the node. An *edge* holds its associated model term and an address of a node. Each model term is a subset of the variables whose presence or absence is indicated by a single bits. This data structure is generated very efficiently from a root model by an algorithm that is described in [0]. Our algorithm avoids generating any duplicate models so that no searches for duplicate models need to be performed.

Model optimality criteria that are based on information theory can be written in the same general form (see [0]). If we define $D_A = -2(L_A - L_0)$ as the deviance of model A, where L_A is the log likelihood for model A maximized over its p_A parameters and L_0 is the log likelihood of the saturated model maximized over its parameters. It is equivalent to say that model A is best if

$$D_A + \alpha p_A \tag{2}$$

is minimum, where the coefficient α may be a constant or a function of the number of observations.

HModel uses only a monotonicity property of a model optimality criterion and can thus admit a more general form than (2). Any function $c(D, p)$ that satisfies

$$c(D_1, p) \leq c(D_2, p), \text{ whenever } D_1 \leq D_2 \tag{3}$$

can be used as an optimality criterion. That is, the criterion only needs to be a non-decreasing function of D for a fixed p.

The model search algorithm uses (3) to compute bounds on the optimality criterion of models that have not been fit (their D is not known). A criterion where computing D is the most difficult part of computing $c(D, p)$ is best, because then very little time is spent computing these bounds. This is the case for the information criteria, because computing D requires fitting the model.

Given an optimality criterion threshold and the representation of models under consideration, we can begin to classify models above or below the threshold. For each model that is fit, we use (3) to classify some other models. *HModel* has a simple algorithm to classify all models. It simply fits the largest models (with most terms) that have not been classified and attempts to classify smaller models on the basis of each fit. This works rather well when the set of models below the threshold is small; only a fraction of all models requires fitting. This also suggests that if a good criterion threshold is not known, an iterative process with a low starting value for the threshold is appropriate. The threshold can be increased until a small number of models is selected.

The choice of models to fit is crucial in determining the total number of fits required to complete the search. For this reason, *HModel* allows control over the choice of models to fit that overrides the automatic choices.

3 Overview of the X Interface

X is a network-based graphics window system developed at MIT [0] that is freely available. Its advantages include its great popularity (it is by far the most common window environment now used on scientific workstations), portability, and its ability to create displays remotely over a network. The latter allows the user to harness computing resources (including parallel processors) over a network, while working in a familiar setting.

HModel first creates a window that is used to control most of its functions (shown in Fig. 1). The window contains a set of command buttons and toggle buttons across the top and text display areas below. The buttons invoke or control major functions of *HModel* and some request additional input from the user. The text areas below the buttons display relevant information at various stages of model selection and in response

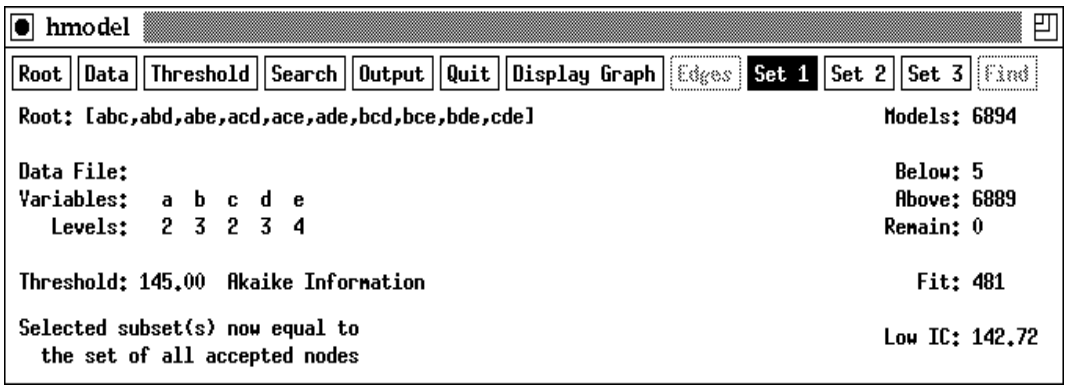

Figure 1: *The HModel window.*

to user actions. Fig. 1 shows a completed search session with 6,894 hierarchical models contained in $[abc, abd, abe, acd, ace, ade, bcd, bce, bde, cde]$ based on a data set used in [0]. Only 481 models required fitting to find the five best models out of the possible 6,894 models under consideration. The button functions are briefly described below.

Root Requests the root model and generates the cover graph.

Data Requests the file name containing the data and a description of the data.

Threshold Requests the optimality criterion threshold and type.

Search Starts the search for models that are at least as good as the criterion threshold and puts them into the currently selected set.

Output Writes model and criterion information for models in the currently selected set to a file for subsequent analysis by other software.

Quit Exits *HModel*.

Find Creates sets of models that satisfy a given search string. These sets can then be subsequently fit, output to a file, or used in other operations. The search string can contain explicit model terms (such as ab) or implicit model terms (such as ... representing any three-way interaction). It can also contain an * that matches any remaining terms.

Display Graph Pops up the cover graph window.

Set n If one or more of these are selected, any nodes that are in these sets are highlighted in the cover graph window. Any set operations are with respect to the selected sets.

Edges Displays the edges that extend from nodes in the currently selected sets.

The cover graph window, shown in Fig. 2, is optional and contains the cover graph representation of the collection of models generated from the root model. An automatic model search session does not need to display this window. This window provides more user control and feedback during the model search and allows some set operations on the models. Nodes with criterion values at or below the threshold are displayed in red (white on a monochrome display), and nodes above the threshold are displayed in blue (black on a monochrome display). Nodes whose position relative to the threshold is not known are displayed using both colors.

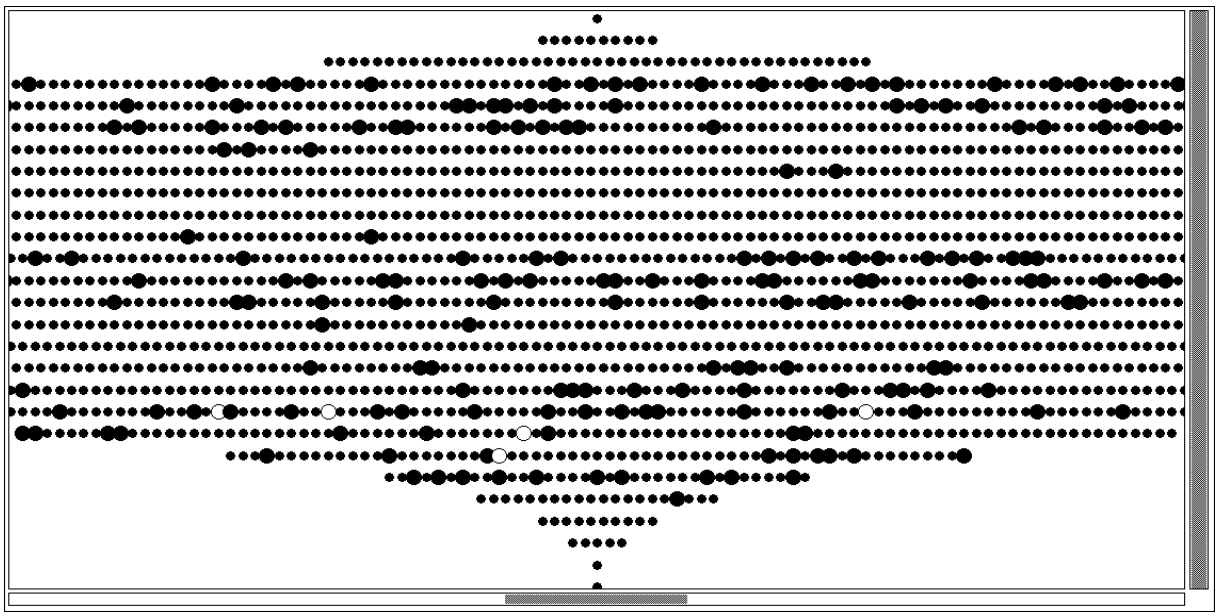

Figure 2: *The cover graph window corresponding to session in Fig. 1. Only nodes (the dots) without the edges are shown. Each level contains models with the same number of terms starting with the null model at the bottom and finishing with the root model at the top. Large nodes are those that were fit in order to find the five best models (white nodes). Scroll bars indicate what portion of the complete cover graph is visible.*

Many keyboard and mouse button operations are available in the cover graph window. These include:

- Display a model and its deviance, number of parameters, and criterion value.
- Fit a single model and try to classify more models.
- Fit all models of a given size that are not classified and use this information in classifying smaller models.
- Fit all models in a set.
- Create a set from all models that have been fit.
- Create a set from all models currently below the threshold.
- Add or delete a model in a set.
- Clear a set.
- Find the intersection or the union model of the models in a set.
- Adjust the space between nodes.
- Adjust displayed area of cover graph with scroll bars.
- Display available keyboard commands.

4 Availability of *HModel* and Current Work

HModel is available from *statlib* (a system for distribution of statistical software located at Carnegie Mellon University) by electronic mail. The source code and documentation can be obtained by sending the message *send hmodel from general* to *statlib@lib.stat.cmu.edu*. If you have access to ftp, you can obtain the same file faster by anonymous ftp (username

statlib) to *lib.stat.cmu.edu* and getting *hmodel* in the directory *general*.

Current work on *HModel* focuses on extensions beyond log-linear models in categorical data and on improving the interactive performance through the use of parallel processing by distributing the model fitting workload across several processors. A collection of workstations connected over a network can be used as a single parallel machine. We use PVM (Parallel Virtual Machine) [0], a software tool designed for adapting applications to run in parallel on a network of heterogeneous processors.

We encourage users or potential users of *HModel* to send their comments, bug reports and suggestions to *ost@msr.epm.ornl.gov*.

5 Acknowledgments

The author acknowledges the work of Linda Kelleher and Eric Sedlacek for programming much of the X interface.

The author's work was supported by the Applied Mathematical Sciences subprogram of the Office of Energy Research, U. S. Department of Energy under contract DE-AC05-84OR21400 with Martin Marietta Energy Systems, Inc.

References

[1] A. C. Atkinson. Likelihood ratios, posterior odds and information criteria. *Journal of Econometrics*, 16:15–20, 1981.

[2] David Edwards and Tomáš Havránek. A fast procedure for model search in multidimensional contingency tables. *Biometrika*, 72:339–351, 1985.

[3] Stephen E. Fienberg. *The analysis of cross-classified categorical data*. The MIT Press, Cambridge, MA, 1977.

[4] G. A. Geist and V. S. Sunderam. Network-based concurrent computing on the PVM system. *Concurrency: Practice & Experience*, 4(3):in press, 1992.

[5] Jim Gettys, Ron Newman, and Robert W. Scheifler. *Xlib – C language interface*. Massachusetts Institute of Technology, Cambridge, MA, 1987.

[6] George Ostrouchov and Edward L. Frome. A model search procedure for hierarchical models. *Computational Statistics & Data Analysis*, 15:285–296, 1993.

[7] Julio L. Peixoto. A property of well-formulated polynomial regression models. *Amer. Statist.*, 44:26–30, 1990.