To appear in *Learning from Data: Artificial Intelligence and Statistics*, D. Fisher and H. Lenz (Eds.), 1995.

Control Representation in an EDA Assistant

Robert St. Amant and Paul R. Cohen

Computer Science Technical Report 95-68

Experimental Knowledge Systems Laboratory
Computer Science Department, Box 34610
Lederle Graduate Research Center
University of Massachusetts
Amherst, MA 01003-4610

Abstract

To develop an initial understanding of complex data, one often begins with exploration. Exploratory data analysis (EDA) provides a set of statistical tools through which patterns in data may be extracted and examined in detail. The search space of EDA operations is enormous, too large to be explored directly in a data-driven manner. More abstract EDA procedures can be captured, however, by representations commonly used in AI planning systems. We describe an implemented planning representation for AIDE, an automated EDA assistant, with a focus on control issues.

This research was supported by ARPA/Rome Laboratory under contracts F30602-91-C-0076 and F30602-93-C-0100.

1 Understanding Complex Data

Data analysis plays a central role in our attempts to understand the behavior of complex systems. Exploratory studies are one part of the descriptive process; they provide an informal prelude to experiments, in which questions and procedures can be refined. Exploration is a kind of detective work: to make a formal presentation of a case, the researcher must follow subtle and potentially conflicting clues to a set of possible conclusions. Exploratory results give rise to confirmatory studies in a cycle of successively more refined exploration and confirmation [Cohen95, Hand86].

Exploratory data analysis (EDA) offers a wide range of statistical tools for the early stages of analysis [Tukey77]. Simple exploratory procedures generate histograms of discrete and continuous variables, scatter plots and box plots for bivariate relationships, partitions of relationships that distinguish different modes of behavior, simplified functional relationships, and two-way tables such as contingency tables. From a consideration of these partial descriptions of data, a more complete picture emerges.

Viewed as search, exploration is a difficult problem. The flexibility of exploratory operators entails a large branching factor and an unbounded search space. If an exploratory analysis were to be driven purely by successive features discovered in data, the task would be impossible: Is a partitioning or a functional transformation appropriate? With what parameters? When should one stop? Though difficult and painstaking, exploration is nevertheless manageable in human hands. Several characteristics of exploration make this possible: relatively few general principles guide exploratory procedures; difficult problems are often decomposed into smaller or simpler parts; exploration is constructive, often relying on partial results and incremental improvement to reach solutions. We can draw a natural analogy between exploration and planning.

We have designed an Assistant for Intelligent Data Exploration, AIDE, to assist humans in the early stages of data analysis [StAmant94a, StAmant94b]. In AIDE, data-directed mechanisms extract simple observations and suggestive indications from the data. Scripted EDA operations then act in goal-directed fashion to generate simpler, deeper, and more extensive descriptions of the data. Control plans guide the EDA operations, relying on intermediate results for their decisions. The system is mixed-initiative, capable of autonomously pursuing high and low level goals while still allowing the user to guide or override its decisions.

AIDE is currently a prototype under development. Though capable of the analysis we present here, AIDE has not yet been extensively tested.

2 The Problem

In *Exploratory Data Analysis* [Tukey77, p.v], John Tukey describes EDA in this way:

> A basic problem about any body of data is to make it more easily and effectively handleable by minds—our minds, her mind, his mind. To this general end:
>
> - anything that makes a simpler description possible makes the description more easily handleable.
> - anything that looks below the previously described surface makes the description more effective.
>
> So we shall always be glad (a) to simplify description and (b) to describe one layer deeper.

Our design revolves around an understanding of the EDA process as the iterative application of four classes of operations: description, simplification, deepening, and extension.

Descriptive operations produce conventional summaries of data: single-valued statistics such as means and medians, or structures such as histograms and fitted lines. Such results can be directly used or interpreted by the user.

Simplification operations transform data to facilitate description. A log transform that straightens a skewed relationship is one example. Irregularities such as outliers are more easily detected in linear relationships than in nonlinear ones; a change in density can often enhance patterns in data; many statistical operations, even ones as simple as Pearson's correlation coefficient, rely on linearity. A straightening operation simplifies in that it enhances our observation, manipulation, and evaluation—in a word, our description—of the data.

Deepening operations increase the detail, accuracy, and precision of descriptions, as a microscope enhances observation by trading a global perspective for local detail. An example is the common practice of examining the residuals of a linear fit. The examination itself is carried out by simplification and description operations. Residual examination can bring to light structure not captured by the line, such as clustering, unequal variance in residuals, or local deviations from linearity.

With descriptive, simplifying, and deepening capabilities, EDA builds descriptions of single variables, bivariate relationships, tables, partitions and clusters. Local descriptions can have nonlocal, often widespread implications, however; these are examined by *extension* operations. When clustering is observed in two different relationships, for example, an extension operation prompts a comparison and an attempt to consolidate the descriptions of the relationships.

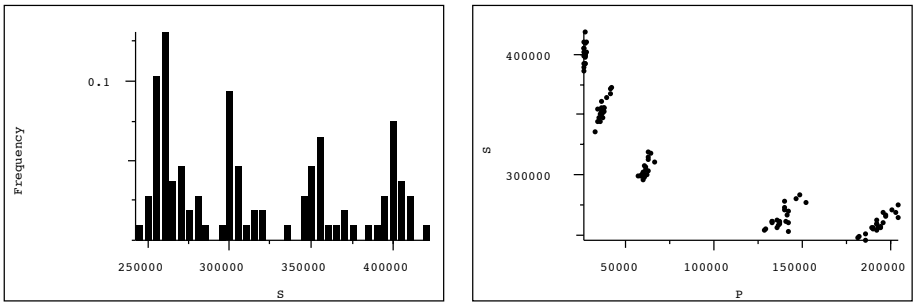
EDA operations give local detail about patterns in variables and relationships. If a dataset contains dozens or even hundreds of variables, however, it becomes difficult to determine where these operations should be applied. Modeling procedures serve to focus exploration. Such procedures include regression analysis, discriminant analysis, clustering, and causal modeling. In the case of regression, one explores to see whether there are nonlinear relationships between a regression variable and a regressor, or whether there are patterns in residuals [Daniel80]. In cluster analysis one searches for functional relationships that may distort cluster shapes [Chen74]. In causal modeling one searches for patterns that influence conditional independence [StAmant94b]. Modeling constrains the application of EDA operations and supplies context for interpretation of their results.

3 A Brief Example

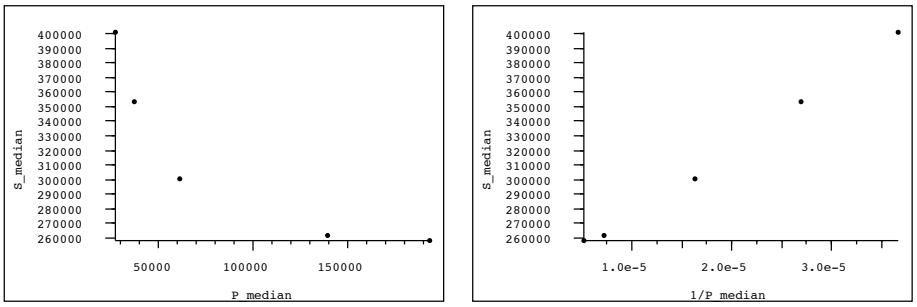
Much of our research deals with the behavior of AI planners in demanding simulation environments. One such system is TRANSSIM, a transportation planner/simulator [Oates94]. In TRANSSIM, ships travel between ports carrying cargo along assigned routes. Bottlenecks and other occurrences change the environment in unexpected ways. The planner must react dynamically to these changes by rescheduling dock and ship assignments and rerouting cargo through different intermediate ports.

An early experiment examined relationships between resource costs. We collected measurements of cumulative port cost (P), ship cost (S), and dock cost (D), the number of ships (N), and trial duration (TD). We were particularly interested in the relationship between the resource costs P and S over the duration of a trial. While we set N at different values, we fixed per-day port, ship, and dock costs. In this example, we assume that a modeling procedure (here a regression or causal modeling procedure) requires examination of the relationship $\langle P, S \rangle$.

We begin with summary statistics for each variable. For S, cumulative ship cost, we find that the median is about 310K, the interquartile range 95K, and there is a slight skew toward lower values. More significantly, when we examine a histogram of S (Figure 1a), there are four distinct clusters in the data. Our preliminary partial description of S comprises the statistics and our



(a) Histogram of S (b) $\langle P, S \rangle$



(c) $\langle P_{median}, S_{median} \rangle$ (d) $\langle 1/P_{median}, S_{median} \rangle$

Figure 1: Exploring ship and port costs

observations about the clustering. Our observations of P are comparable.

When we turn to the relationship $\langle P, S \rangle$ (Figure 1b), we see a different pattern: the values fall into *five* clusters. The distinct separation in $\langle P, S \rangle$ values, as well as the observation that one of the modes in the histogram of S (Figure 1a) corresponds to a cluster twice as large as the others, leads us to return to our description of S. We establish an alternative possible description of S as containing five clusters, consistent with the clusters in $\langle P, S \rangle$.

Continuing with our analysis of $\langle P, S \rangle$, we see that values in the leftmost cluster (which we denote ps_a) can be fit by a straight line. In fact, this is true for all five clusters, though it is a different line in each case. Once we settle on the description of each cluster as a line, we can add the observation that the slopes of the lines decrease as the clusters move toward the right.

If we plot the central locations of the clusters ps_a through ps_e (i.e., the median coordinates P_{med} and S_{med} for each cluster), as shown in Figure 1c, we see that these five summary points can be fit by a smooth curve. Further exploration shows that the curve is of the form $S_{med} = 1/P_{med}$. When we perform this transformation (Figure 1d) it straightens the curve, leaving no clear pattern in the residuals.

Extending the analysis, we find that the five discrete values of the variable N, the number of ships, correspond to the clusters found in the relationship $\langle P, S \rangle$. Because we have experimental control over the number of ships but not the cumulative costs, we count an observation about the former measurements as an explanation of the latter. In this case, N explains the clustering of resource costs.

This description, though brief, should give the flavor of the analysis. To summarize, we begin with initial descriptions of the data, such as the observation of gaps between adjacent values. From these we generate indications [Mosteller77], or suggestive characteristics: the data fall into clusters.

Based on these indications we apply specific EDA procedures: we break the data down and analyze the clusters individually. These procedures may involve iterative refinement, as with the alternative descriptions of clusters in S. When possible, we simplify; instead of dealing with all data points, we work with a reduced dataset of just the median points of the clusters. We then follow two separate courses: we extend current results, here by considering other variables, and we deepen results, in this case by turning from a surface description of the clusters to an examination of the behavior of the data within each cluster. The result is a coherent, structured description of the data.

4 Data Manipulation

Datasets are the basic level of data representation. A dataset is an extension of the familiar relational table of attributes and values. Extensions capture implicit and explicit knowledge we have about the data. Datasets and attributes can be specialized as distinct types, the most common cases being relationships (or multivariate subsets of datasets) and variables (or univariate relationships). While there exist operations applicable only to variables, or to relationships, there is a large set of shared operations that make it desirable for all data to be handled in tabular form.

We manipulate a dataset in three distinct ways. The first kind of manipulation involves subdividing or combining elements of datasets. In order to construct fitted lines to each cluster in the TRANSSIM data, we partitioned the original data into five smaller datasets. We then examined each cluster independently, at a finer level of detail. We call an operation that breaks data into smaller but similar parts a *data decomposition.* A data decomposition generates a mapping from the elements of a dataset to membership in a set of new, derived datasets. Operations that bin data, exhaustively partition data, or nonexhaustively cluster data all decompose data in this way. The inverse operation, *data composition*, combines separate datasets. Data decompositions and compositions let us capture independent detail in subsets of data and recombine results.

The second kind of manipulation involves deriving new attributes for elements of the dataset. During the TRANSSIM analysis we performed a transformation of cluster medians to remove curvature. Operationally the procedure transformed a single attribute, P_{med}, into a new attribute, P', where $P' = 1/P$. We call such an operation an *attribute transformation.* A special type of transformation is attribute composition: consider the transformation from attributes port cost, ship cost, and dock cost to $S + P + D$, or total cost. Another type of transformation is attribute decomposition: to conclude that a given line is a good fit to a relationship, we need to generate the residuals by subtracting $\hat{P}$ from P—by decomposing the attribute into structure and residual.

Because dataset attributes may contain datasets as well, another useful transformation is the mapping of a statistic over the elements of an attribute. Our data decomposition of the $\langle P, S \rangle$ relationship generates five new datasets. More precisely, the result is a new dataset, PS_c, containing a single attribute "Clusters", whose value is these new datasets. We can now apply an attribute transformation to the "Clusters" attribute, to produce newa ttributes such as "x-median" and "y-median". We continue by fitting a line, a further attribute transformation. Functions for attribute transformation include arithmetic operations, exponentiation, and these higher level dataset operations.

The third kind of manipulation is *reduction*, which is the type of function performed by summary statistics such as means and medians. Correlations and partial correlations are similarly reductions of bivariate and multivariate relationships.

Designing a system around such general structures and operations is not simply an attempt at conciseness. Rather, the design is geared specifically toward the kinds of operations appropriate for EDA. Consider the example of generating a histogram for a discrete variable N in a dataset T.

To build a histogram we divide a variable into bins and count the number of observations that fall into each bin. Using the operations describe above, we can implement a histogramming procedure as follows. First we apply a data decomposition to T. The result is a set of new datasets that form disjoint subsets of T, stored as an attribute "Data" in a new dataset, T_h. Our operations now work on T_h. An attribute transformation of "Data", based on the count statistic, gives the size of the bins. Another transformation using the mode statistic gives the value of N associated with each bin. These distinct values and their counts can then be displayed directly in histogram form.

This may seem an inordinate effort to produce such a basic structure, but consider a simple extension: the contingency table. To build a contingency table between N and C_{mem}, an attribute that records the membership of each data point in a specific cluster, we follow essentially the same procedure. This time the decomposition simultaneously bins N and C_{mem}. Decomposition operations are not limited to single variables; they can just as easily decompose two variables or an entire dataset. The result of the operation is a dataset, T_c, with a two-dimensional attribute containing the partitions, one for each unique value of N and C_{mem}. Again we transform by the count statistic. Calculating the N, C_{mem} values for each bin is equally straightforward. The desired contingency table data is the result. A simple variation of this procedure produces a box plot, by computing letter values rather than counts. More complex two-way tables can also be generated in a similar way, by calculating statistics for a third variable in each of the partitions.

Producing structures in this way has two abstract benefits: complex procedures can often be seen as natural extensions of existing procedures, and natural connections between conceptually similar structures become clear. While it is trivial to observe that a contingency table is a two-dimensional histogram, it is worth stressing that the two structures can be produced by very similar procedures, and that both procedures are different instances of a single canonical, divide-and-conquer EDA procedure. A data decomposition breaks a dataset into smaller parts; dataset-level attribute transformations compute a set of features of the reduced data; these features are explored both individually and in aggregate. The procedure is a simple, powerful example of generalization, as described in the conceptual clustering and constructive induction literature [Michalski83, Fisher86].

Planning offers a natural framework for representing EDA procedures. A planning representation can capture general procedures and intermediate goals, constructive derivation of results, and hierarchical problem decomposition—necessary elements for exploration. At the core of AIDE is a script-based planner with a library of partial plans, or scripts, which can be combined as required by the characteristics of the data.

```
(define-script variable-histogram
  :goal     (describe ?variable ?directives ?histogram)
  :vars     (variable directives (histogram :output))
  :features ((:content-type :categorical))
  :bindings ((histogram (partitions index count)))
  :grammar  (:SEQUENCE
             (script-partition variable 'identity
                               :output histogram
                               :names '(partition index))
             (script-transform histogram #'(lambda (partition)
                                             (script-reduce partition 'count))
                               :key (attributes partitions)
                               :name count)))
```

The scripting language is based on work in control planning for knowledge-based signal processing [Carver93]. In its simplest form, a script has a goal form with input and output variables, a set of bindings, and a grammar of actions to execute. Grammar constructs allow sequencing, iteration,

mapping, and conditionalizing of actions. A script becomes active when a goal is established at a higher level that matches its goal form; it succeeds when its own grammar of actions completes. A script to generate a histogram, slightly abbreviated, is shown above.

We have already mentioned variants of the histogramming procedure, which are implemented by similar scripts. Other scripts compute initial values for parameters of resistant lines, transformations of skewed relationships, and so forth. Each script produces a set of new or transformed structures, appropriately annotated and related to existing structures. As scripts execute, a growing hierarchy of datasets, relationships, and their attributes is generated.

5 Control Planning

The scripting representation lets AIDE search through a space of general procedures rather than primitive data manipulations. Guiding the search are *indications,* or suggestive characteristics of the data. Indications involve evaluation of a statistic or descriptive structure derived from the data. For example, evidence of clustering is an indication, as is the presence of outliers, and curvature in the residuals of a linear fit. Indications may be simple computations, such as testing the skew of a distribution against a preset threshold. A more complex indication involves constructing a nearest neighbors clustering for a relationship and checking for outlying distances between adjacent clusters. In each case the presence of an indication leads to the selection of scripts that can potentially explain its presence.

This representation restructures the search space of exploration into a more appropriate form. Unfortunately, the problem of choosing an appropriate search operation still applies, even at this higher level. The space remains too large to be searched with only data-driven operations.

Control planning addresses the problem [Carver93]. Control plans are a simple extension of data manipulation scripts. A control plan grammar contains subgoals rather than actions, and can implement more general procedures than those provided by scripts. Control plans guide the activation of data manipulation scripts and other control plans. The control plan given below is a generalization of the histogramming script.

```
(define-control-plan* describe-cluster-behavior
  :goal        (select-control ?relationship ?directives ?result)
  :vars        (relationship directives (result :output))
  :features    ((:dataset-type relationship)
                (:cardinality 2))
  :indications ((clustering-indication *))
  :grammar     (:SEQUENCE cluster-subgoal
                          extract-components-subgoal
                          (:MAP (substructure*)
                                describe-substructure-subgoal)
                          compose-descriptions-subgoal
                          explore-composition-subgoal))
```

This control plan is used in the TRANSSIM analysis to describe the data in terms of the locations of the clusters, rather than the data points themselves. It decomposes a dataset into separate components, describes each component, and aggregates the results into a single structure, to be explored in turn. A lower level plan supplies descriptions in the form of the location of each cluster. While the behavior of this plan is similar to that of a script, it can be used more generally to explore the locations of the clusters, their sizes, their functional behaviors, and so forth. The entire

process is an example of simplification, in which we work with an abstraction of the original data to facilitate a high level functional description.

The combination of control plans with data manipulation scripts further restructures the search space; focusing heuristics provide the final component. Actions that later turn out to be incorrect or irrelevant are inevitable in the analysis. This means that at points during the execution of a plan, an intermediate result can obviate some set of pending actions. When this happens, a focusing heuristic can abandon the current plan and select a new one based on the current state of knowledge.

More specifically, focusing allows context-sensitive information to influence the order in which subgoals are satisfied and the selection of appropriate parameter values. The most important role of focusing heuristics is to allow for user interaction in the analysis. We often find that information external to the data affects our decisions about the course of exploration. We may know that a relationship falls naturally into three clusters, rather than four, because of a known sampling bias, or that some variable value acts as a natural threshold between different behaviors in the data. Focusing heuristics take such user-supplied information into account where needed: in the decisions about which plans are applicable, and which input values are appropriate for their execution.

A broad view of the exploratory analysis, using the TRANSSIM example, is as follows. A modeling procedure generates a relationship $\langle P, S \rangle$ between port cost and ship cost. Exploration results are interpreted in the context of the model. The exploratory phase of the analysis begins with the generation of features and indications for the relationship: relatively high correlation, a hierarchical clustering with clear separation between individual groups, evidence of curvature, skew in both variables.

A focusing heuristic evaluates the relevant control plans, using information provided by the indications, and selects the plan describe-cluster-behavior. The rest are suspended, to be pursued when the first returns with more information. The selected control plan breaks down into three main subgoals: (a) the generation of clusters, (b) the exploration of each cluster, and (c) the exploration of the aggregated descriptive results of the clusters. Subgoal (a) can be satisfied by different plans: one that attends to clusters in the entire relationship, and others that detect clusters in the component variables. The bivariate clustering plan is selected, again by a focusing heuristic. This plan generates three candidate assignments of points to clusters; the heuristic threshold for the indication is not sensitive enough to make an unambiguous assignment. A single assignment is selected. As with plan selection, the remaining possibilities are suspended until later, if needed.

The selected cluster assignment gives a decomposition of the relationship into separate clusters. In control phase (b), each is explored in turn. A focusing heuristic, taking into account the context of the cluster description control plan, selects a data manipulation plan to determine the location of each cluster. Location is computed from the features of each cluster: from the mean, median, and other summary statistics. The values generated are returned and aggregated into a single relationship.

Subgoal (c) is then established to explore this new relationship. Based on indications for the reduced data, a new control plan is selected that will straighten the curved relationship and fit a line to the result. The linear fit is then explored in turn as a form of deepening.

Each decision point is controlled by a focusing heuristic. If a selected plan or variable value results in either failure or inadequate information, the focusing heuristic will be reactivated to make a different selection, to try to satisfy the goal in a different way. Focusing and refocusing give a needed flexibility to the exploration process. Sometimes a shallow exploration will be sufficient to show, for example, that there are no gross departures from linearity in a relationship; a focusing heuristic can evaluate indications in context and decide not to reactivate other pending plans. Sometimes a result can show that there are patterns deeper in the data; a focusing heuristic can

direct attention toward the patterns in an opportunistic way.

6 Related Work

This work draws on a number of different sources. The clearest relationship is to early work in developing concepts of statistical strategy, or the formal descriptions of actions and decisions involved in applying statistical tools to a problem [Hand86]. The REX system, for example, implemented a strategy for linear regression [Gale86]. Comparable complex strategies were developed by others for collinearity analysis [Oldford86]. The goals of AIDE bear a resemblance to those of TESS [Lubinsky88], which supports analysis by accommodating user knowledge of context in a search good descriptions of data. AIDE extends this work by supplying an appropriate taxonomy for EDA operations and using this taxonomy to guide search at different levels of abstraction.

Central to AIDE is the opportunistic, incremental approach to discovery described by Tukey, Mosteller, and other advocates of EDA. There are some obvious difficulties: maintaining consistency in an incrementally growing set of descriptions can be difficult; in many cases local techniques can miss simple global patterns; local techniques can lead to a plethora of spurious results. Nevertheless many such problems can be alleviated in a system does not always act autonomously, focuses on both data and on goals of the analysis, and pursues exploration paths with the aid of external knowledge and the context supplied by its own actions.

7 Acknowledgments

This research is supported by ARPA/Rome Laboratory under contracts F30602-91-C-0076 and F30602-93-C-0010. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes not withstanding any copyright notation hereon.

References

[Chen74] H. J. Chen, R. Gnanadesikan, and J. R. Kettenring. Statistical methods for grouping corporations. *Sankhya, Series B*, 36:1–28, 1974.

[Carver93] Norman Carver and Victor Lesser. A planner for the control of problem solving systems. *IEEE Transactions on Systems, Man, and Cybernetics, special issue on Planning, Scheduling, and Control*, 23(6), November 1993.

[Cohen95] Paul R. Cohen. *Empirical Methods in Artificial Intelligence.* MIT Press, 1995. In press.

[Daniel80] Cuthbert Daniel and Fred S. Wood. *Fitting Equations to Data.* Wiley, 1980.

[Fisher86] Douglas Fisher and Pat Langley. Conceptual clustering and its relation to numerical taxonomy. In William Gale, editor, *Artificial Intelligence and Statistics.* Addison-Wesley, 1986.

[Gale86] W. A. Gale. REX review. In W. A. Gale, editor, *Artificial Intelligence and Statistics.* Addison-Wesley, 1986.

[Hand86] D.J. Hand. Patterns in statistical strategy. In W.A. Gale, editor, *Artificial Intelligence and Statistics*, pages 355–387. Addison-Wesley, 1986.

[Lubinsky88] David Lubinsky and Daryl Pregibon. Data analysis as search. *Journal of Econometrics*, 38:247–268, 1988.

[Michalski83] R. S. Michalski and R. E. Stepp. Learning from observation: Conceptual clustering. In R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, editors, *Machine Learning: An artificial intelligence approach.* Morgan Kaufmann, 1983.

[Mosteller77] Frederick Mosteller and John W. Tukey. *Data Analysis and Regression.* Addison-Wesley, 1977.

[Oates94] Tim Oates and Paul R. Cohen. Toward a plan steering agent: Experiments with schedule maintenance. In Kristian Hammond, editor, *Second International Conference on Artificial Intelligence Planning Systems*, pages 134–139. AAAI Press, 1994.

[Oldford86] R. Wayne Oldford and Stephen C. Peters. Implementation and study of statistical strategy. In W.A. Gale, editor, *Artificial Intelligence and Statistics*, pages 335–349. Addison-Wesley, 1986.

[StAmant94a] Robert St. Amant and Paul R. Cohen. A planning representation for automated exploratory data analysis. In D. H. Fisher and Wray Buntine, editors, *Knowledge-Based Artificial Intelligence Systems in Aerospace and Industry Proc. SPIE 2244*, 1994.

[StAmant94b] Robert St. Amant and Paul R. Cohen. Toward the integration of exploration and modeling in a planning framework. In *Proceedings of the AAAI-94 Workshop in Knowledge Discovery in Databases*, 1994.

[Tukey77] John W. Tukey. *Exploratory Data Analysis.* Addison-Wesley, 1977.