

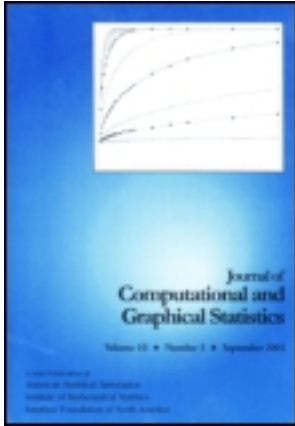
This article was downloaded by: [University of Arizona]

On: 28 May 2013, At: 13:32

Publisher: Taylor & Francis

Informa Ltd Registered in England and Wales Registered Number: 1072954

Registered office: Mortimer House, 37-41 Mortimer Street, London W1T 3JH, UK



Journal of Computational and Graphical Statistics

Publication details, including instructions for authors and subscription information:

<http://www.tandfonline.com/loi/ucgs20>

Intelligent Support for Exploratory Data Analysis

Robert St. Amant^a & Paul R. Cohen^b

^a Department of Computer Science, North Carolina State University, Box 7534, Raleigh, NC, 27695-7534, USA

^b Department of Computer Science, University of Massachusetts, Box 34610, Amherst, MA, 01003-4610, USA

Published online: 21 Feb 2012.

To cite this article: Robert St. Amant & Paul R. Cohen (1998): Intelligent Support for Exploratory Data Analysis, Journal of Computational and Graphical Statistics, 7:4, 545-558

To link to this article: <http://dx.doi.org/10.1080/10618600.1998.10474794>

PLEASE SCROLL DOWN FOR ARTICLE

Full terms and conditions of use: <http://www.tandfonline.com/page/terms-and-conditions>

This article may be used for research, teaching, and private study purposes. Any substantial or systematic reproduction, redistribution, reselling, loan, sub-licensing, systematic supply, or distribution in any form to anyone is expressly forbidden.

The publisher does not give any warranty express or implied or make any representation that the contents will be complete or accurate or up to date. The accuracy of any instructions, formulae, and drug doses should be independently verified with primary sources. The publisher shall not be liable for any loss, actions, claims, proceedings, demand, or costs or damages whatsoever or howsoever caused arising directly or indirectly in connection with or arising out of the use of this material.

Intelligent Support for Exploratory Data Analysis

Robert ST. AMANT and Paul R. COHEN

Exploratory data analysis (EDA) is as much a matter of strategy as of selecting specific statistical operations. We have developed a knowledge-based planning system, called AIDE, to help users with EDA. AIDE strikes a balance between conventional statistical packages, which need guidance for every step in the exploration, and autonomous systems, which leave the user entirely out of the decision-making process. AIDE's processing is based on artificial intelligence planning techniques, which give us a useful means of representing some types of statistical strategy. In this article we describe the design of AIDE and its behavior in exploring a small, complex data set.

Key Words: Artificial intelligence; Statistical strategy.

1. STRATEGIC ASPECTS OF EXPLORATION

Exploring data is as much a matter of strategy as of choosing specific statistical operations. In his introduction to exploratory data analysis, for example, Tukey begins by laying out guidelines for describing data: anything that makes a simpler description possible makes the description easier to handle; anything that looks below the previously described surface makes the description more effective (Tukey 1977, p. v). Surveys of techniques for exploration inevitably discuss such strategies for their use (Velleman and Hoaglin 1981; Hoaglin, Mosteller, and Tukey 1983; Hoaglin, Mosteller, and Tukey 1985). These strategies guide our selection and application of statistical operations and our interpretation of their results.

Modern statistical software gives us increasingly powerful tools for EDA, including interactive graphics, batch programming support, and an enormous array of statistical procedures. In one significant respect, however, the situation has not changed from the pencil-and-paper days. Even the most sophisticated modern systems rely on a clear, fixed division of labor: the user is responsible for decision-making, the system for computation. However effectively a system might fit linear models, perform cluster analyses, and carry out other computational tasks, it will generally never take the initiative to explore the

Robert St. Amant is Assistant Professor, Department of Computer Science, North Carolina State University, Box 7534, Raleigh, NC 27695-7534 (E-mail: stamant@csc.ncsu.edu). Paul R. Cohen is Professor, Department of Computer Science, University of Massachusetts, Box 34610, Amherst, MA 01003-4610 (E-mail: cohen@cs.umass.edu).

©1998 American Statistical Association, Institute of Mathematical Statistics,
and Interface Foundation of North America
Journal of Computational and Graphical Statistics, Volume 7, Number 4, Pages 545-558

data on its own. Lacking strategic knowledge, when faced even with the most obvious patterns in a relationship, or with only a single plausible choice among data analysis techniques, the system must wait for the user's commands.

Automated strategic reasoning could provide enormous benefits, greatly reducing the work load of statisticians and giving nonstatisticians easy access to expert advice. Over the past 15 years researchers have developed strategies for regression analysis (Gale 1986), regression-based data description (Lubinsky and Pregibon 1988), variable selection in clustering and discriminant analysis (Fowlkes, Gnanadesikan, and Kettnering 1987), MANOVA (Hand 1986), collinearity analysis (Oldford and Peter 1986), ANOVA in biomedical domains (Silvers, Herrmann, Godfrey, Roberts, and Cerys 1994), and a number of other areas (Gale, Hand, and Kelly 1993).

Unfortunately, despite promising early results, statistical reasoning systems have seldom left the research laboratory, and no general-purpose commercial system can claim to reason strategically about statistical decisions (Gale, Hand, and Kelly 1993).

One reason for this situation is that developers of strategic statistical systems have traditionally concentrated on the question, "What part of the statistical reasoning process can be automated?" A more effective question, which we see asked in the areas of statistical user interfaces, programming environments, and visualization systems, is "How can human statistical reasoning be enhanced or complemented by automation?" Building an autonomous system to reason about EDA is only half a solution; the other half is the accommodation of the user's subject-matter knowledge and judgment (Lubinsky and Pregibon 1988). An accommodating system takes advantage of human knowledge to augment its own necessarily limited view of the world. A combination of autonomy and accommodation would let the human data analyst shift some of the routine or search-intensive aspects of an exploratory task to the system, without giving up the ability to review and guide the entire process.

Our goal is to give users a flexible measure of automated, strategic support for EDA, in the form of a system called AIDE. AIDE is a knowledge-based planning system that incrementally explores a data set, guided by user directives and its own evaluation of indications in the data. Its plan library contains a varied set of strategies for generating and interpreting indications in data, building appropriate descriptions of data, and combining results in a coherent whole. The system is mixed-initiative, autonomously pursuing high- and low-level goals while still allowing the user to inform or override its decisions.

2. AN EDA EXAMPLE

By EDA we mean techniques for fitting linear and higher-order functions to relationships, for composing and transforming variables with arithmetic functions, for separating relationships into partitions and clusters, for extracting features through statistical summaries, and the like. Simple exploratory results include histograms that describe discrete and continuous variables, schematic plots that give general characterizations of relationships, partitions of relationships that distinguish different modes of behavior, functional simplification of low-dimensionality relationships, and two-way tables such as contingency tables. EDA techniques are commonly distinguished from techniques for modeling

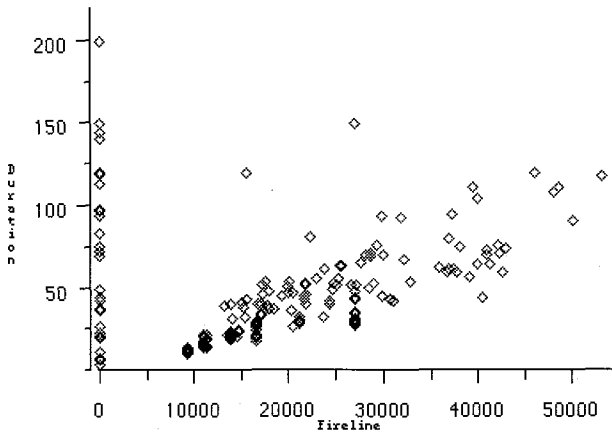


Figure 1. Patterns in Planner Fireline and Trial Duration.

and hypothesis confirmation; in EDA the emphasis lies on preliminary, informal probing of the data for underlying structure (Good 1983; Hoaglin, Mosteller, and Tukey 1985).

We build our description of AIDE around data from an experiment with PHOENIX, a simulation of forest fires and fire-fighting agents in Yellowstone National Park (Cohen, Greenberg, Hart, and Howe 1989). The experiment involved setting a fire at a fixed location and specified time, and observing the behavior of the fireboss and the bulldozers that put out the fire. Variability between trials is due to nonuniform terrain and elevation, a probabilistic model of fire spread, and the varying amounts of time agents take in executing primitive tasks. We collected 40 variables over the course of some 340 trials, including measurements of the wind speed, the outcome (success or failure), the type of plan used, and the number of times the fireboss failed in its attempts and needed to revise its plans. We were mainly interested in about ten of these variables, especially in the relationship between the time it takes the fireboss to put out a fire (Duration) and the amount of fireline built during the trial (Fireline). Figure 1 shows a scatterplot of these two variables.

The discontinuity in Fireline values is one obvious pattern. We call this discontinuity an *indication*, a suggestive characteristic of the data (Mosteller and Tukey 1977). Examining other variables we find that the zero values correspond to trials in which the outcome was Failure; the lack of data is an artifact of data collection. In the Success cases the correlation between Fireline and Duration is positive, as expected: doing more work takes longer. The relationship is approximately linear, with only a few outliers. The residuals of the fit show no indications of curvature or influential points that would render the linear description incorrect.

A closer look at the Success data, however, shows small clusters in the lower range of Fireline. The clusters also form a linear pattern, with a slope slightly less than that of the entire partition. Again examining other variables, we find a plausible explanation in the variable #Replans. The clustered values correspond to cases in which the fireboss succeeded in its first effort to put out the fire, in which it did not need to

Table 1. Median Duration by WindSpeed and PlanType for Clustered Data

PlanType =	WindSpeed =		
	Low	Medium	High
A	11.0	15.5	18.8
B	14.5	26.7	22.1
C	19.5	29.9	29.7

revise its initial firefighting plan. For the remaining values the fire temporarily burned out of control, which led to greater variance in effort. We can explain the separation of points into several distinct clusters with the variables WindSpeed and PlanType, which together strongly predict cluster membership. A table of medians summarizes these predictions with respect to Duration values, as shown in Table 1. All cells represent approximately equal numbers of observations. We find that Plan A is more conservative than B or C for all wind speeds, and that at a moderate wind speed, Plan B performs surprisingly poorly. An exploratory decomposition of the tabular data (Hoaglin, Mosteller, and Tukey 1991) shows no further patterns in the relationship.

In describing this process we have simplified a good deal. Some of the relationships, such as between #Replans and Fireline, were only discovered by searching through the data set to explain an unexpected pattern. In other cases, we needed to make decisions about how to fit lines, whether transformations were necessary, how to interpret clusters, whether specific indications were relevant, and so forth. These decisions drew on our knowledge about the domain and about data analysis procedures, but were also informed by patterns opportunistically discovered in the data. An automated assistant like AIDE can contribute to the above data analysis in a number of ways. For a regression analysis, for example, it can run various linearity heuristics and check for residual patterns. When clusters are present, it can explore different clustering criteria and search for predictive relationships. It can test different transformations of the data to eliminate patterns of skew or curvature. In general, while the system will contribute directly to interpretation, it can provide autonomous, data-driven search and evaluation of patterns, taking cues from user actions to direct its behavior.

These examples suggest some of the behavioral properties we would expect from an ideal strategic assistant for data analysis. First, the assistant performs its own heuristic evaluation of the data. It takes advantage of indications in the data as more detail is uncovered, to focus its exploration on promising areas. Second, the assistant does more than simply react to user commands. Though the familiar “submit command → view result” cycle is supported, the system anticipates the user’s actions with knowledge of common statistical practice. Third, the assistant does not go to the other extreme of being completely autonomous; to this end the assistant makes its decision process accessible to the user. The user can thus review the assistant’s decisions concerning statistical procedures, parameter settings, and variable selection, reevaluating and modifying decisions whenever necessary. These properties are central to the design of AIDE as a planning system.

3. PLANNING

AIDE treats EDA as a search problem, more specifically a planning problem. Planners formulate search in terms of states, goals, and sequences of actions to achieve goals. A planner solves a problem by constructing a step-by-step specification of actions that move from the initial conditions (the start state) through intermediate states to the desired conclusions (the goal state). The construction is complicated by constraints on the application of actions, uncertainty about the effects of actions, and unforeseen interactions between goals, among other difficulties. Planners rely on task decomposition to solve complex problems, using knowledge about states, actions, and goals to structure the search at different levels of abstraction (Russell and Norvig 1995).

Traditional AI planners construct plans from primitive actions, starting from scratch for each new problem. Script-based planning takes a different approach (Georgeff and Lansky 1986; Carber and Lesser 1993). If a planning system repeatedly encounters similar problems, it becomes wasteful to generate similar solutions from scratch each time. Instead, a script-based planner draws on a library of plans that apply to common cases. Rather than choosing an appropriate action at some point during the planning process, the planner can choose an existing plan in which many or all of the actions have already been determined. If the library has sufficient coverage, planning largely becomes a matter of choosing which plan to apply at which time.

AIDE follows this approach. It draws from a library of about 100 plans, mostly for analysis of univariate and bivariate relationships. Plan selection relies on a comparable number of indication definitions, rules, and other control constructs. In the next sections we discuss the actions that constitute basic plan primitives, the representation of AIDE plans, and the control of their execution.

3.1 PRIMITIVE OPERATIONS IN AIDE

AIDE represents data sets as relations. Variables correspond to relation attributes, observations to relation tuples. Bivariate and multivariate relationships are relations with two or more attributes. A variable may be considered either an attribute of a relation or a relation consisting of a single attribute. In contrast to usual relational database conventions, the domain of an attribute may contain other relations as well as atomic values.

For its data manipulation, in addition to conventional relational and statistical operations, AIDE relies on three types of higher-order operations: *reduction*, *transformation*, and *decomposition*. These constitute a language for manipulating exploratory structures, in the same sense that the relational algebra constitutes a language for manipulation of relations in a database.

A reduction takes a function and a relation as input and generates an atomic element as a result. Computing the mean of a sequence of numbers is an example of a reduction, as is the generation of letter values, or the correlation between two sequences of numbers. The functions by which we reduce variables are most often measures of location, scale, shape, and magnitude. Functions that compute location in AIDE include the mean, the median, the trimean, different levels of trimmed mean, and weighted averages. For

the other categories of functions we have similar ranges of possibilities. Reductions of bivariate relationships include measures of covariation (Pearson's r , Spearman's rank correlation, etc.) along with measures analogous to those we use for variables.

A transformation takes a function f and a relation R as input, and maps f over the tuples of R to generate a new relationship, S . For each tuple r_i over the specified attributes in R , S contains a corresponding tuple $s_i = f(r_i)$. Taking logs of a batch of numbers is a simple example of a transformation. Another commonly applied transformation in AIDE is the generation of residuals from a linear fit. In this case the transformation function is subtraction, the input a relationship containing y and \hat{y} . Many transformations, like this one, make patterns more easily visible. Power transformations, for example, can increase symmetry or separate densely clustered regions in data. Specialized, ad hoc transformations can be useful in a similar way; the reciprocal of a duration measurement, and ratios, sums, and differences of related measurements, are often meaningful.

A decomposition takes a relation R and a mapping function M as input. M is applied to individual tuples and returns a subset of $1, \dots, k$ for each tuple. The decomposition of R by M generates a new relation S , of cardinality k , with a single attribute whose values are new relations themselves. The contents of the i th relation in S are those tuples r_j in R such that $i \in M(r_j)$. Decompositions may partition a data set or generate overlapping subsets. Separating a data set into clusters is a decomposition, as is isolating outliers in a relationship. Decompositions in AIDE are mainly of two types. The first kind decomposes data into mutually exclusive subsets depending on a variety of criteria. We separate outliers from the main body of data, distinguish clusters, and separate partitions with different decompositions. The second kind of decomposition divides data into overlapping subsets, in order that a function may be applied within a limited "window" of the data. Smooths, kernel density estimators, and forms of robust regression rely on this form of decomposition.

Combinations of these higher-order operations offer surprising generality in representing common procedures. Consider a simple procedure for building a histogram for a discrete-valued variable x : divide the range of the variable into its unique values; break the variable into subsets, one subset per unique value; count the number of elements in each subset. In other words, we decompose the variable, then apply a transformation with an embedded reduction that maps each subset relationship to a single value. Now consider a procedure for building a contingency table for a relationship between x and some other discrete-valued variable y . We divide it into subsets, one corresponding to each unique combination of x and y values. We record the number of observations in each subset, associating each count with a different x, y combination. The resulting values are the cell counts for the contingency table. A contingency table can thus be seen as a two-dimensional analog of a histogram.

Such similarities between statistical procedures are not unusual. Constructing a table of median Duration values for the PHOENIX data set, for example, is similar to constructing a histogram or contingency table as above. A procedure for kernel density estimation has the same structure as the histogram procedure, but uses different decomposition and reduction functions. These combinations of operations constitute primitive actions in the planning representation, building blocks from which more complex plans are constructed.

```
(define-plan fit-relationship ()
  "Describe a relationship by fitting it. Evaluate the fit."
  :satisfies (generate-description :fit ?structure ?fit-operation ?fit-relationship)
  :features ((?structure (:dataset-type relationship)
                        (:cardinality 2))))
  :body      (:SEQUENCE
             (:SUBGOAL generate-fit
                       (generate-fit ?structure ?fit-operation ?fit-relationship))
             (:SUBGOAL evaluate-fit
                       (explore-by ?strategy ?activity ?model ?fit-relationship
                                   ?deepening-result))))
```

Figure 2. A Sample Plan.

3.2 PLANS AND GOALS IN AIDE

Above the level of primitive actions, AIDE supports two basic planning structures—goals and plans. AIDE begins with a single, top-level goal, for example, to describe a specific relationship. To satisfy this goal, AIDE searches through its library for plans and actions that provide potentially relevant descriptions of the given relationship. Actions generate a result directly, while plans establish more detailed subgoals. These subgoals, when satisfied, combine to produce a final descriptive result that completes the plan and thus satisfies the higher-level goal.

A goal has a name and a form containing variables and constants. Data values pass between goals and plans to satisfy them by unification. A plan has a name, a specification of a goal that the plan can potentially satisfy, constraints on variable bindings in the goal, and a body. The body of a plan is a schema of subgoals that must be satisfied for the plan to complete successfully. Control constructs in the schema allow sequencing, iteration, and conditionalizing of subgoals, as well as other domain-specific forms. A simple plan with subgoals is given in Figure 2.

In words, this plan computes a fit for a bivariate relationship and then examines the fit for possible deviations. The `:features` form constrains the plan's applicability. The first subgoal, `generate-fit`, may be satisfied by different subplans: AIDE's current plan library contains plans for a least-squares linear fit, different varieties of resistant line fits, and smoothing fits. The second subgoal, `evaluate-fit`, is matched in each case by plans that explore the residuals for patterns not captured by the fit. While not an example of deep statistical knowledge, this plan nevertheless captures a basic exploratory strategy, the separation of a description into structure and residuals. The `fit-relationship` plan makes explicit the dependence between procedures for fitting and procedures that "deepen" the fit. These latter procedures might confirm the adequacy of the description, or, for example, in the case of outliers in the residuals of a least-squares linear fit, can call for a revision of an earlier procedure. In these plans, the representation of explicit procedural knowledge lets the system (as well as the user) interpret results in the context in which they are generated.

The planning representation lets us build data descriptions of different scope and at different levels of abstraction. In it we have implemented different kinds of resistant

lines, various box plot procedures, kernel density estimators, and smoothing procedures; we have found little difficulty in implementing other simple exploratory procedures as well. We have also developed more abstract sets of plans to implement forward selection algorithms for cluster and regression analysis, causal modeling algorithms, and other incremental, opportunistic modeling procedures.

Planning works for EDA by reducing the search space through abstraction, problem decomposition, and identification of useful combinations of operations (Kork 1987). These are identifying characteristics of a planning problem, and are central to strategic reasoning in EDA. Abstraction is ubiquitous in exploration. In fitting a line to a relationship, we implicitly decide that variance around the line, evidence of curvature, outlying values, and so forth may be abstracted away. Exploring patterns not captured by the description is then a matter of moving to a lower level of abstraction. Hierarchical problem decomposition plays a large part in exploration as well. We often partition relationships or examine clusters or outliers to gain a clearer picture of patterns. Much of exploration can be viewed as the incremental decomposition of data into simple descriptions, which are then combined into a more comprehensive result. Exploratory procedures also impose top-down structure on the exploration process. In other words, when we execute an exploratory operation we generally have a good notion of which operation to execute next. Furthermore, common procedures often fall into a few basic families that process data in similar ways. We can draw similar analogies between procedures for smoothing and for generating kernel density estimates, or between resistant line fitting and locally weighted regression curves. While sometimes novel procedures are constructed from scratch, variations on existing procedures are much more common.

3.3 CONTROL IN AIDE

Planning reduces the search problem of exploration, by considering a set of procedures rather than the much larger set of all possible combinations of primitive operations. We must still address some further issues: Which course of action (i.e., which plan) should be followed? Which data should be examined, which operations applied? More specifically, to which values should each plan's internal variables be bound? When should the choice of a specific plan or value of a plan variable be reconsidered? When should one course of action be abandoned for a different, more promising one?

In AIDE, these issues are addressed by three mechanisms. *Activation rules* determine which plans are active and which values a plan variable is bound to. *Focus points* manage plans as they execute, maintaining and evaluating different possible courses of action as more information becomes available. A *meta-planner* controls plan execution sequences, maintaining a broader perspective on the exploration process. These elements play complementary and interacting roles in exploration.

During exploration of the PHOENIX data set, a goal was established to describe the (Fireline, Duration) relationship. Several options arose, each represented by a different plan. These plans were activated by rules triggered by features and indications, ranging from judgments about linearity and curvature to tests for clustering and outliers.

Rules simply activate plans to satisfy goals; several may become active at once. A

plan focus point determines which plan is most appropriate. A plan focus point selects a single plan as its current focus but may change its selection depending on incrementally generated results. In the (Fireline, Duration) example, plans for the linear fit and generating clusters were both allowed to execute, in turn. The plan focus point managed the process by changing its focus appropriately.

A variable focus point is analogous to a plan focus point. Just as multiple plans can satisfy a goal, there may be more than one way to bind plan variables in executing a plan. In exploring a relationship with clusters, for example, it often happens that several decompositions may be plausible. A variable focus point maintains each of the different cluster mappings, to be examined individually. Its behavior is essentially the same as that of a plan focus point.

Together, rules and focus points determine which plans should be applied to satisfy a goal and which bindings are appropriate for plan variables. These control mechanisms act locally, taking advantage of planning context. For example, in a linear regression context, rules and focus points narrow the planner's consideration to a small set of relevant plans and actions. Sometimes, however, a result derived from a relationship in one plan may suggest a course of action in an entirely different area of exploration. Control must shift, not just from one plan to another in a single focus point, but from one focus point to another. This shift is managed by the meta-planner.

From the meta-planner's point of view, AIDE's expansion of the exploration search space is a branching tree. The root of each subtree represents the result of making a sequence of choices: plan P_1 rather than plan P_2 to satisfy goal G , binding variable v to b instead of a in plan P_1 , and so forth. Each branch point—that is, each node with more than one child—is a focus point. The meta-planner uses this branching tree to implement a form of opportunistic backtracking, which is sometimes called “refocusing” (Carber and Lesser 1993). By globally evaluating the results available at each focus point, the meta-planner can decide which branches of the exploration should be expanded and which should be pruned.

To summarize briefly, evaluation and preference rules decide which plans are relevant in trying to satisfy a goal. A focus point selects one plan from those possibilities and executes it; the focus point may switch dynamically between possible plans as knowledge about the problem changes. Meta plans decide which focus point is currently active; that is, which goal the system should be trying to satisfy at the current time. These mechanisms give flexible, opportunistic control over the exploration process.

4. USER INTERACTION

AIDE can be used like any other menu-based statistical package, in which one examines the data, selects a statistical operation, examines the result, and repeats the cycle. A key distinction between AIDE and a conventional statistical interface, however, is that in addition to displaying data and results, AIDE supports the exchange of information about the decisions that give rise to the results. This is more than simply recording and replaying a history of user commands: the user is able to move forward and backward through a growing hierarchy of decisions, reviewing and modifying choices where nec-

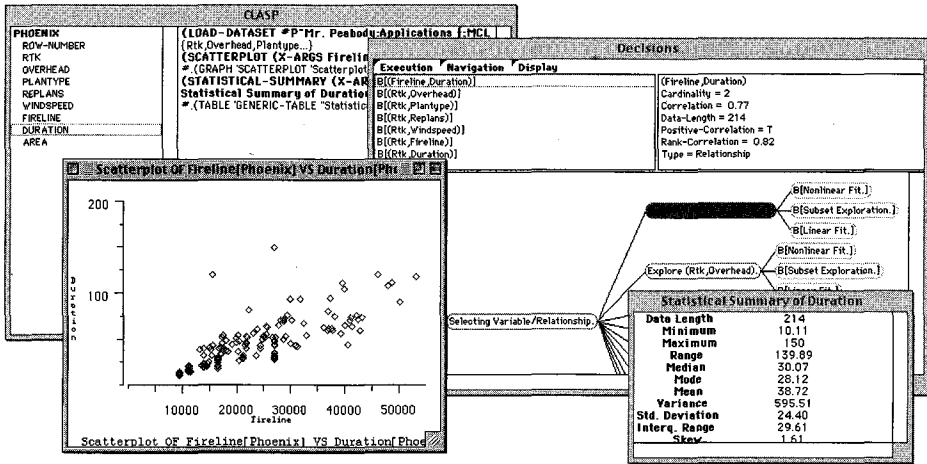


Figure 3. The AIDE Environment.

essary. AIDE itself has the same capabilities. This lets interaction with AIDE exhibit a flexible balance between autonomous action and accommodation of human knowledge.

The AIDE interface is shown in Figure 3. Menu choices let the user load a data set, compose variables into relationships, compute summary statistics, generate linear models, partition data, run statistical tests, and so forth. One window displays the variable, relationship, table, or other data structure under consideration. In a combined window, we see indications and derived descriptions for the data structure. When AIDE is used in its conventional, user-driven mode, such descriptive windows are opened and modified dynamically. The combined window also contains actions or choices that AIDE suggests the user take. Other windows are automatically created to document the choices AIDE is considering and the results AIDE has generated.

We can best see the interface at work by walking through a sequence of decisions. In our PHOENIX example, the first decision of the user is to focus on the relationship (Fireline, Duration). The user selects from a list of all data set variables to construct a bivariate relationship with Fireline as the x variable and Duration as the y variable. A scatterplot is immediately displayed, along with a textual description of relevant features and indications. For example, different indications of outliers are displayed, each generated by a different criterion. These indications are mouse-sensitive; clicking on one shows which points in the scatter plot can be considered outliers. Indications of clustering, curvature, skew, and so forth are similarly active.

The system immediately establishes an exploration goal for (Fireline, Duration), which results in a display of relevant actions, with the linear fit action marked as the system's suggestion. Each action is documented with set of choices that allow the user to see a justification for the action and the criteria by which the action was ranked among the other possibilities.

Later in the analysis the user requests a linear fit. AIDE generates a resistant line, which is superimposed on the (Fireline, Duration) relationship. AIDE presents in text form the parameters of the line along with active indications describing outliers and evidence

of clustering in the residuals. Several residual evaluation structures are automatically generated as part of fitting the line, to be examined further through AIDE's suggested actions.

These exchanges highlight the balance AIDE maintains between autonomy and accommodation. Because the selection of a variable or relationship often depends heavily on subject matter knowledge, AIDE makes suggestions rather than selecting data by itself. On the other hand, once a relationship has been selected and the user has decided on a general course of action, AIDE can take on more of the decision-making responsibility. In the last exchange, for example, the user accepts the suggestion of a linear fit, and AIDE pursues a linear fitting strategy which involves autonomous decisions about how to fit the line and how to examine the residuals (residual examination is part of a larger set of plans, based on a regression strategy developed by Gale and Pregibon 1986). Other examples of autonomous behavior on AIDE's part include searching for predictive relationships involving categorical variables when clusters are explored, generating a linear fit for a relationship if it has been transformed to remove curvature, "sharpening" a relationship if no clear pattern is initially apparent (Tukey and Tukey 1981), and performing various other data reductions and searches to bring out patterns. In some areas AIDE behaves much like a conventional statistical system but in other areas provides more autonomous assistance.

Maintaining this balance is harder than might appear at first glance. If the system can take actions without consulting the user, it can as a natural consequence drive the analysis into inappropriate areas, possibly losing the user in the process. Similarly, the user is not constrained to take only those actions the system finds reasonable; on the contrary, with better pattern-matching abilities and knowledge of what variable values actually mean, the user should be able to take the analysis in whichever direction appears appropriate. The system must nevertheless be able to follow in the user's path, ready to contribute to the process once it again reaches a familiar area.

AIDE's plans alleviate this problem somewhat. They attempt to implement common statistical practice, relying implicitly on the user's knowledge of what actions are appropriate in a given situation. Thus, if the system takes the initiative by executing a sequence of actions without consultation, the result will rarely be completely incomprehensible to the user. In the worst case, the user will think, "Why did the system choose that action in this situation?" while in the best case the response will be, "That's exactly what I would have done." The examination of residuals is a simple example of the best case: the exploration process continues in an expected direction. To handle cases in which AIDE's actions are more subtle, a set of navigation tools lets the user follow its decision-making and make changes when appropriate.

The navigation interface, visible in the lower half of one of the windows in Figure 3, serves to locate the user in the "landscape" of the exploration process. This landscape can be displayed in several different forms.

- The command history view displays a trace of the analysis: each operation, the data to which it was applied, and the result generated.
- The overview view presents a complete graphical model of the data set, with variables represented as nodes and relationships as arcs, marked appropriately

to show which have been explored and the types of descriptions that have been established for them.

- The data view displays relationships and derivations (e.g., transformations, partitions) in a tree form.
- The plan view displays the plan structure as it has been elaborated to the current point. Each focus point is presented, with documentation about the decision it manages. By browsing through this graph, the user can review past decisions. The user can also cause AIDE to return to an earlier decision to resume at that point.

The importance of a navigation facility is reflected by research in mixed-initiative planning. James Allen has identified strong similarities between mixed-initiative planning and dialogs that problem-solving agents participate in (Allen et al. 1995; Ferguson, Allen, and Miller 1996). Allen et al. (1995) identified three distinguishing characteristics of the approach: flexible, opportunistic control of initiative; the ability to change focus of attention; and mechanisms for maintaining shared, implicit knowledge.

AIDE's control of initiative changes with context. That is, whether a decision is presented to the user or is settled internally depends on situation-dependent factors. For example, in most cases, if AIDE determines that only one plan is able to satisfy a given goal (i.e., all others rendered inactive by evaluation rules), then this decision point will not be presented to the user. At any other point in the exploration, the user is free to take the initiative. When the user changes focus of attention, AIDE follows suit. For example, the user may decide to abandon exploration of one relationship for exploration of another, simply by choosing `Select variable/relationship` from the menu bar and constructing the new relationship. AIDE interprets this as a shift in focus of attention, and runs the appropriate meta-level actions to match the shift. AIDE also makes limited decisions on its own to change focus. For example, after fitting a line to a relationship and generating residuals, AIDE presents a set of residual examination and other plans to the user. An `Okay` gesture, which usually indicates that the top-ranked plan for the current decision should be activated, instead causes AIDE to refocus on other plans for exploring the relationship. Finally, part of the shared context between the user and AIDE is implicit in the plans stored in the library. These plans are intended to implement common statistical procedures, so that the user, when encountering a focus point, will find that it involves a plausible and perhaps even familiar decision. Navigation mechanisms further support the sharing of context. The user can view the data under consideration, results constructed, commands carried out, and planning structures leading to the current state.

5. RELATED WORK

AIDE builds on work in the areas of statistical expert systems, statistical interfaces, and planning. The study of statistical strategies, or formal descriptions of the actions and decisions involved in applying statistical tools to a problem (Hand 1986), has led to automated systems in several areas. AIDE's plans are based mainly on strategies formulated by earlier researchers, as described earlier (Gale 1986; Lubinsky and Pregibon 1988; Fowlkes, Gnanadesikan, and Kettenring 1987; Hand 1986; Oldford and Peter 1986;

Silvers, Herrmann, Godfrey, Roberts, and Cerys 1994; Gale, Hand, and Kelly 1993).

Work in statistical user interfaces has also influenced AIDE's development, especially the systems DINDE (Oldford and Peter 1986; Oldford and Peter 1988) and KENS (Hand 1987). DINDE is a statistical, object-oriented computational environment. It models a statistical analysis explicitly as a collection of acyclic directed graphs, in which each node represents a step in the analysis. Data analysis can be treated as a kind of navigation through these graphs. KENS is also a navigational system, but one less directly tied to a specific analysis. KENS is designed for flexible access to an information space of statistical knowledge, to be retrieved and applied as the user sees fit. AIDE follows the navigational approach of these systems and adds semi-autonomous decisions on the part of the system.

Mixed-initiative systems have recently become interesting to researchers in AI planning. Current systems sometimes lean toward a mostly autonomous, planning-with-consultation approach, and sometimes toward a user-driven planning environment supplied with appropriate tools. This same distinction between user-driven and mostly autonomous collaboration can be seen in statistical expert systems. REX is an autonomous regression system with consultation: it surrenders control when reaching difficult points in its analysis. ESTES, a system for time series analysis, is comparable to a tool-based environment, in which the user picks the appropriate operation, which then executes autonomously. AIDE, by relying on explicit similarities between planning and EDA, attempts to balance problem-solving initiative in a more flexible way.

In summary, we have described the task of EDA and how it can be cast as a planning problem. We have shown how AIDE contributes to EDA: in providing a representation well-suited to procedural statistical knowledge, and in allowing flexible, mixed-initiative interaction with the user. We have described a representative problem in the domain of EDA, and explained AIDE's role its solution. Our evaluation of AIDE, describe elsewhere (St. Amant and Cohen 1998), gives evidence that our approach can be effective in some areas of collaborative, human-computer data analysis.

[Received February 1996. Revised May 1998.]

REFERENCES

- Allen, J. F., Schubert, L. K., Ferguson, G., Heeman, P., Hwang, C. H., Kato, T., Light, M., Martin, N. G., Miller, B. W., Poesio, M., and Traum, D. R. (1995), "The TRAINS Project: A Case Study in Building a Conversational Planning Agent," *Journal of Experimental and Theoretical Artificial Intelligence*, 7, 7-48.
- Carver, N., and Lesser, V. (1993), "A Planner for the Control of Problem Solving Systems," *IEEE Transactions on Systems, Man, and Cybernetics, special issue on Planning, Scheduling, and Control*, 23, 1519-1536.
- Cohen, P. R., Greenberg, M. L., Hart, D. M., and Howe, A. E. (1989), "Trial by Fire: Understanding the Design Requirements for Agents in Complex Environments," *AI Magazine*, 10, 32-48.
- Ferguson, G., Allen, J., and Miller, B. (1996), "TRAINS-95: Towards a Mixed-Initiative Planning Assistant," in *Proceedings of the Third International Conference on Artificial Intelligence Planning Systems*, ed. B. Drabble, Menlo Park, CA: AAAI Press, pp. 70-77.
- Fowlkes, E. B., Gnanadesikan, R., and Kettenring, J. R. (1987), "Variable Selection in Clustering and Other Contexts," in *Design, Data, and Analysis: By Some Friends of Cuthbert Daniel*, ed. C. L. Mallows, New York: Wiley, pp. 13-34.

- Gale, W. A. (1986), "REX Review," in *Artificial Intelligence and Statistics*, ed. W. A. Gale, Reading, MA: Addison-Wesley, pp. 173–227.
- Gale, W. A., Hand, D. J., and Kelly, A. E. (1993), "Statistical Applications of Artificial Intelligence," in *Computational Statistics*, vol. 9 of *Handbook of Statistics*, ed. C. R. Rao, Amsterdam, New York: North-Holland, pp. 535–576.
- Georgeff, M. P. and Lansky, A. L. (1986), "Procedural Knowledge," in *Proceedings of the IEEE Special Issue on Knowledge Representation*, 74, pp. 1383–1398.
- Good, I. J. (1983), "The Philosophy of Exploratory Data Analysis," *Philosophy of Science*, 50, 283–295.
- Hand, D.J. (1986), "Patterns in Statistical Strategy," in *Artificial Intelligence and Statistics*, ed. W.A. Gale, Reading, MA: Addison-Wesley, pp. 355–387.
- (1987), "A Statistical Knowledge Enhancement System," *Journal of the Royal Statistical Society, Series A*, 150, 334–345.
- Hoaglin, D. C., Mosteller, F., and Tukey, J. W. (1983), *Understanding Robust and Exploratory Data Analysis*, New York: Wiley.
- (1985), *Exploring Data Tables, Trends, and Shapes*, New York: Wiley.
- (1991), *Fundamentals of Exploratory Analysis of Variance*, New York: Wiley.
- Korf, R. E. (1987), "Planning as Search: A Quantitative Approach," *Artificial Intelligence*, 33, 65–88.
- Lubinsky, D., and Pregibon, D. (1988), "Data Analysis as Search," *Journal of Econometrics*, 38, 247–268.
- Mosteller, F., and Tukey, J. W. (1977), *Data Analysis and Regression*, Reading, MA: Addison-Wesley.
- Oldford, R. W., and Peters, S. C. (1986), "Implementation and Study of Statistical Strategy," in *Artificial Intelligence and Statistics*, ed. W.A. Gale, Reading, MA: Addison-Wesley, pp. 335–349.
- (1988), "DINDE: Towards More Sophisticated Software Environments for Statistics," *SIAM Journal of Scientific and Statistical Computing*, 9, 191–211.
- Russell, S., and Norvig, P. (1995), *Artificial Intelligence: A Modern Approach*, Upper Saddle River, NJ: Prentice Hall.
- Silvers, A., Herrmann, N., Godfrey, K., Roberts, B., and Cerys, D. (1994), "A Prototype Statistical Advisory System for Biomedical Researchers I: Overview," *Computational Statistics and Data Analysis*, 18, 341–355.
- St. Amant, R., and Cohen, P. R. (1998), "Interaction With a Mixed-Initiative System for Exploratory Data Analysis," *Knowledge-Based Systems*, 10, 265–273.
- Tukey, J. W. (1977), *Exploratory Data Analysis*, Reading, MA: Addison-Wesley.
- Tukey, P. A., and Tukey, J. W. (1981), "Graphical Display of Data Sets in Three or More Dimensions," in *Interpreting Multivariate Data*, ed. Vic Barnett, New York: Wiley pp. 189–275.
- Velleman, P. F., and Hoaglin, D. C. (1981), *Applications, Basics, and Computing of Exploratory Data Analysis*, Boston, MA: Duxbury Press.