TOWARDS A STRUCTURED DATA ANALYSIS ENVIRONMENT: A COGNITION-BASED DESIGN

FORREST W. YOUNG* AND JOHN B. SMITH†

Abstract. We present a structured data analysis environment designed to improve data analysts' productivity over present unstructured environments. The environment is based on three fundamental concepts: Cognitive Modes, Dataflow Diagrams, and Language Generation. We hypothesize that data analysts adopt different *cognitive modes* of behavior and thought for different stages of the data analysis process. Our environment has corresponding system modes that visually represent these cognitive modes.

One of the system modes enables users to construct *dataflow diagrams* by manipulating icons representing datasets, data analysis procedures, and data analysis results. In this mode the user structures the data analysis by drawing lines from dataset icons to procedure icons; the computer in turn performs the analysis defined by the dataflow diagram, displays result icons and draws lines from the procedure icons to the results icons to complete the diagram. Then, the icons and lines form a manipulable dataflow diagram which represents the flow of data from a dataset through an analysis procedure into results.

The dataflow diagram and its icons generate statements in the language of an underlying statistical system. The naive user is unaware of this. The more advanced user can modify the default language generated by the icons and diagrams. The sophisticated user can completely avoid the icons and diagrams and directly use the underlying language.

1. Introduction.

Scientists work with data. They analyze them, graph them and write about them. Because current computer environments are haphazardly organized and poorly integrated, they are poorly suited to these activities. We believe that scientific productivity would improve if analyzing, graphing and writing about data occurred in a structured environment designed to be consonant with the cognitive activities of the user.

We present an advanced visual interface for managing structured statistical analyses. The interface presents an environment which is structured, visual, and manipulable. The environment interfaces to an underlying statistical analysis system by generating statements in the statistical system's language. Note that we *do not* present a new statistical system. Rather, we present a new *interface* which can be used with an *already existing* statistical system. Thus, what we present does not require duplicating the great amount of effort already invested in the development of a statistical system.

The new interface we present is based on three fundamental concepts: Cognitive Modes, Dataflow Diagrams, and Language Generation. We introduce each of these concepts next.

The first cornerstone of our work, that of *cognitive modes*, has been discussed by Smith and Lansman [1988]. Their research shows that writers adopt different

*Psychometric Laboratory, CB-3270 Davie Hall, University of North Carolina, Chapel Hill NC 27599-3270.

†Computer Science Department, CB-3175 Sitterson Hall, University of North Carolina, Chapel Hill NC 27599-3175.

modes of behavior and thought for different stages of the writing process. The major premise of our work is that data analysts adopt analogous cognitive modes of behavior and thought for certain stages of the data analysis process. The interface we present represents these cognitive modes to the scientist as system modes which are structured windows on the screen of a display. One of the system modes is the dataflow mode and its associated dataflow diagrams. This mode allows the analyst to construct data analyses graphically. Another mode is the language mode which enables the analyst to construct the analysis alpha-numerically, by using the underlying data analysis language. Other modes permit the analyst to interact with the data analysis in other ways, including via forms and spreadsheets.

The second cornerstone of our work is the concept of a *dataflow diagram.* This concept, which is implemented in a structured and manipulable environment, has been discussed by Young [1988]. Users can control the statistical analysis by manipulating icons that represent datasets, data analysis processes, and data analysis results. This is done by placing icons for data or for analysis processes on the screen, and then drawing lines from data icons to analysis icons. When the analysis is performed, the system displays new icons representing results, which are connected by lines to the analysis process which produced the results. Both the user's and system's lines depict the flow of data from datasets through data analysis processes into results. Since an extended statistical analysis is an iterative process, we argue that the capability to easily and quickly construct new dataflow diagrams and to modify existing diagrams is particularly useful.

The third cornerstone of our work is *language generation*: The icons and dataflow diagrams generate statements in the language of an underlying statistical system. For the naive user (and for simple analyses performed by the more sophisticated user) the specifics of the underlying statistical system are completely irrelevant, other than the choice of analysis processes that are provided in the system's toolbox. In particular, analysts need not type statements in the language, nor even be aware that there is a language. For more advanced users (and uses), the default language generated by the icons and diagrams may be modified to more precisely implement the desired analysis. It is possible to completely avoid using the icons and diagrams and to only interact with the system via its language. Thus, those who are less visually oriented and are more comfortable with an alpha-numeric language may completely avoid the graphical interface and use only the language. Since the statements generated by the system can be saved for later execution, the system can also be used to perform repetitive, large, or time consuming analyses in "batch" or "background" style.

In the remainder of this paper, we first describe the system we are developing in general terms in order to give the reader a feel for its use and to place it in context with other statistical systems. Then we look more closely at fundamental concepts on which it is based. Finally, we return to the system to provide a comprehensive, albeit brief, description of the features it offers.

Currently, we are still in the early stages of developing the actual system. The pace of development will be determined by funding. We publish this description now in order to share with others what we believe is an unexplored perspective on the cognitive nature of data analysis and its impact on system design. We also hope to encourage further discussion of specific issues raised in this paper.

2. Data Analysis Environments.

Data analysis is not a simple, linear application of time-tested analytic procedures, particularly for scientific applications at the forefront of knowledge. Rather, the investigator spends much time analyzing and re-analyzing data; searching and re-searching for regularity in data; forming, testing and reforming hypotheses. In this section we discuss three types of environments for analyzing data, and we discuss their relative effectiveness in supporting the type of repeated, non-linear data analysis just mentioned.

2.1. Conventional Environments. Conventional data analysis software (i.e., SAS [1985], SPSS [1983], SYSTAT [1985], S [Becker & Chambers, 1984]) is based on the 1960's batch submission model, even though it is available on microcomputers. Such software is awkward and clumsy. The data analyst must prepare the data analysis "job" by typing statements that must satisfy strict syntactic rules, submit the job to the computer, and then wait for the analysis to be performed, often then to discover that a misplaced semicolon or a misspelled keyword prevents the analysis from being consummated.

Even if there are no syntactical difficulties, the data analyst's task is impeded by such conventional software. The data analysis "job" grows messier and messier as the analyst introduces new twists and wrinkles into the analysis. The code resulting from the repeated re-analysis of the data is messy and unstructured, making it difficult to keep track of what analyses have been done, what the results are, and how to interpret and communicate the results. At the end of the sequence, analysts often have trouble remembering exactly what analyses were done, why they were done, and in what form the results are stored. The stack of print-outs and lists of computer files created provides only the most awkward of memory aids. This style of interaction, based on obsolete software design [Blank, 1985], is slow, frustrating, and not as productive as it would be using more up-to-date software designs.

Even on today's microcomputers, many data analysis systems are still based on yesterday's batch submission model, despite the fact that microcomputers are highly interactive, dedicated to the investigator's sole use, and on the investigator's own desk. Of course, microcomputers dramatically reduce frustration and "turn-around" time from the days of carrying boxes of cards to a distant computer center. Still, even on a microcomputer the the batch submission style of interaction fails to provide the user with a strategic sense of the sequence of steps in the analysis.

2.2. Icon Environments. In the last few years statistical systems have been developed which are icon-based instead of language-based. One of the best examples of this type of system is DataDesk [Velleman & Velleman, 1988]. This type of system graphically represents datasets and their variables as icons. To perform an analysis, the user selects the desired dataset or set of variables by using a mouse to position a cursor on the appropriate icon and by then clicking the mouse's button. To then analyze the selected data, the user selects a menu item that specifies the desired analysis.

This type of icon-based environment is less awkward and clumsy to use than conventional language-based systems: the user does not have to prepare a data analysis "job" to be submitted to the computer for analysis; no statements need to be typed into the system in order to analyze the data; no semicolons or keywords are used, so no syntactical or grammatical errors can occur. This style of interaction, which is based on 1970's and '80's ideas of software design, is faster and less frustrating than the 1960's batch submission style, and seems to be more productive as well. However, the data analysis session still grows messier and messier as the analyst proceeds with the analysis. Now, instead of messy code resulting from repeated re-analysis of the data, a messy "desktop" results, with numerous icons and windows cluttering the screen. There is still no strategic sense of the sequence of steps taken during the analysis. Thus, with these systems we can arrive at the point of utter confusion more easily and quickly than ever before.

There is another major drawback of these systems: By getting rid of the awkward language of the 1960's software, the newer icon-based systems have thrown out the baby with the bathwater. One of the major strengths of the older software is that once the "job" is prepared it can be used over and over again, because the language can be saved and easily reused. While icon-based systems can save the current status of the analysis of a particular set of data so that the analysis can be taken up again at a later time, they cannot be used to create a particular series of analyses and then apply these analyses to a second wave of data. The icon-based systems are ideal for simple, one-shot data analyses, and they are wonderful for extensively exploring data, but they are weak or useless for complex analyses that must be repeatedly performed on wave after wave of data. The old, batch-style systems still excel at this last task.

2.3. A Structured Environment. Neither the language-based 1960's systems nor the icon-based 1980's systems present an environment to the analyst which is structured in a way which summarizes the series of analysis which have occurred. Our hypothesis is that a structured environment that presents a graphical model of the datasets, the data analysis processes, the data analysis results, and the flow of the overall analysis, will allow the data analyst to plan the overall data analysis more logically, revise the plan in line with partial results more effectively, and remember what was done, why it was done, and where the results are stored, more easily and accurately. We also hypothesize that a system which is both language-based and icon-based will further improve data analysis productivity, especially for naive users. Our ideas are related to those of by Borning [1986], Smith & Lansman [1988], and Stuetzle [1987], Oldfield and Peters [1986] and McDonald and Pedersen [1985, 1988], and have been discussed by Young [1988].

Perhaps the easiest way to communicate the nature of our interface, which we call MIDAS (the Mode-based Interface for Data Analysis and Statistics), is via an example. Keep in mind, however, that trying to present an example of an environment like MIDAS in a written document is a very frustrating and inexact process:

The MIDAS environment is dynamic, visual, and nonlinear, whereas writing about such an environment is static, alpha-numeric, and linear. The best we can do is to present a series of "still photographs" in place of the "movie" we would like to show.

Figures 1 through 20 present mock-ups of a portion of the MIDAS screen, as it would appear during specific moments of a data analysis session. MIDAS has several visual modes, each of which can be displayed as a window on the screen. The modes, which are called *dataflow*, *structure*, *spreadsheet*, *language*, and *forms* modes, are illustrated in the Figures. These modes are explained in detail in the technical description section that appears later in this paper.

The figures are based on a real data analysis session: They represent a real series of actions taken by a real data analyst. Furthermore, this example is an unfinished data analysis. We use such an example because we think it represents a typical (i.e., messy) data analysis session. Data analysis, we emphasize, is not a simple linear process, even though the final result of a series of data analysis sessions is usually presented as such!

In the example, the data analyst knows that there is a file with data to be analyzed, but has only vague and unstructured ideas about the analyses to be done. At this stage, the MIDAS screen could look like that shown in Figure 1, where the rectangular icons with vertical columns represent data matrices (and their variables) and the rounded icons represent data analysis processes. Unbeknownst to the user these icons are associated with data analysis statements in the language of an underlying statistical system that is, at this point, hidden from the user.

Figure 2: The data analyst opens the data icon by first selecting it with the mouse and then choosing the appropriate item from a menu. The opened icon reveals a spreadsheet of data values, as shown in the figure. Since only a few data values have been entered into the data matrix, the spreadsheet editor would be used to enter the remaining values.

Figure 3: Once all the data have been entered and have been made ready for analysis, the data spreadsheet is closed, and a graphics structure editor is used to connect the data with the desired data analysis processes, as shown by the line connecting the data icon with the two process icons. The dataflow diagram is now ready to be enacted; when it is, the system will perform the specified analyses on the indicated data set. Note that the unconnected *plot* process icon is not involved in the analysis.

Figure 4: When the data analyst asks MIDAS to perform the analysis, it sends the statements that are associated with the dataflow diagram and its icons to the underlying statistical system. This system performs the analysis, returning the results to MIDAS in the form of new data, text or graphics files. These new files are represented to the data analyst on the screen by new icons such as those shown at the bottom of Figure 4. Note that the two small rectangular icons, seen for the first time in this figure, represent reports of data analysis results, whereas the two new dataset icons represent the original data as processed by the analysis procedures.

Figure 5: To view the results, the analyst opens the "report" results icon.

MIDAS then displays the contents, as shown.

Figure 6: The report suggests to the data analyst that a plot of a portion of the results might be informative. The analyst closes the report, and, having already obtained the plot procedure icon from the toolbox of data analysis procedures supplied with the underlying statistical system, connects the appropriate data icon ("Outstat" in the example) to the plot icon.

Figure 7: The analyst, wishing to review and possibly modify the plotting procedure's defaults, invokes the forms mode. Here the analyst is shown the current values of all options of the plotting process and can change any of the values with a forms editor.

Figure 8: The analyst invokes the language mode to obtain a view of the language currently generated by the plotting process. The analyst can edit this language and add any additional language that is desired. (The figure shows a sequence of pseudo-language statements for illustration. In an actual system, the language window would display statements in the language of the underlying statistical system.)

Figure 9: When the plotting process options are set as desired, the language and/or forms mode windows are closed and MIDAS is asked to perform the analysis. Since most of the analysis has already been performed (as described for Figure 4), MIDAS only has to create the graphic results of the plotting procedure. After the plot procedure creates the graphical results, MIDAS displays a results icon named "graphic" on the screen.

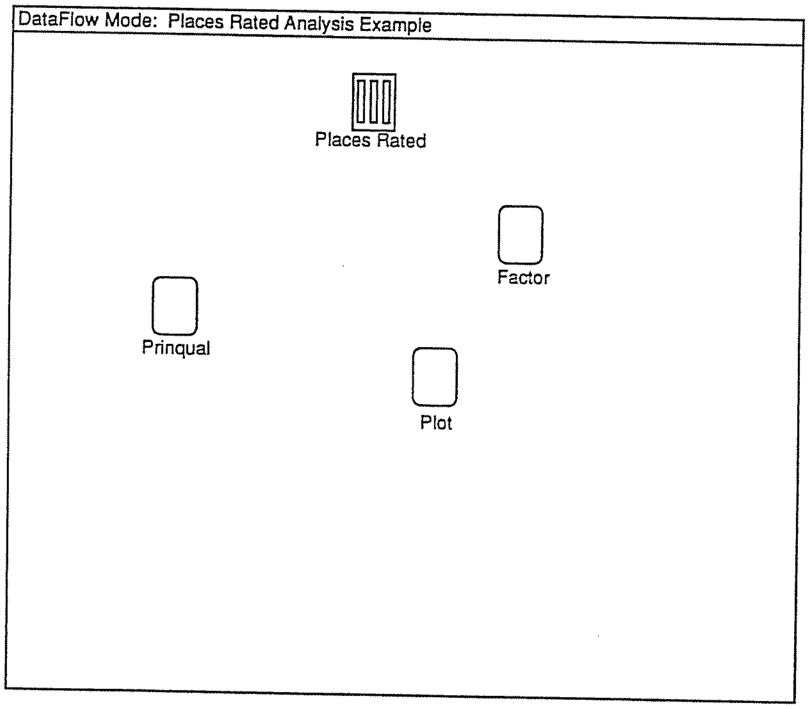
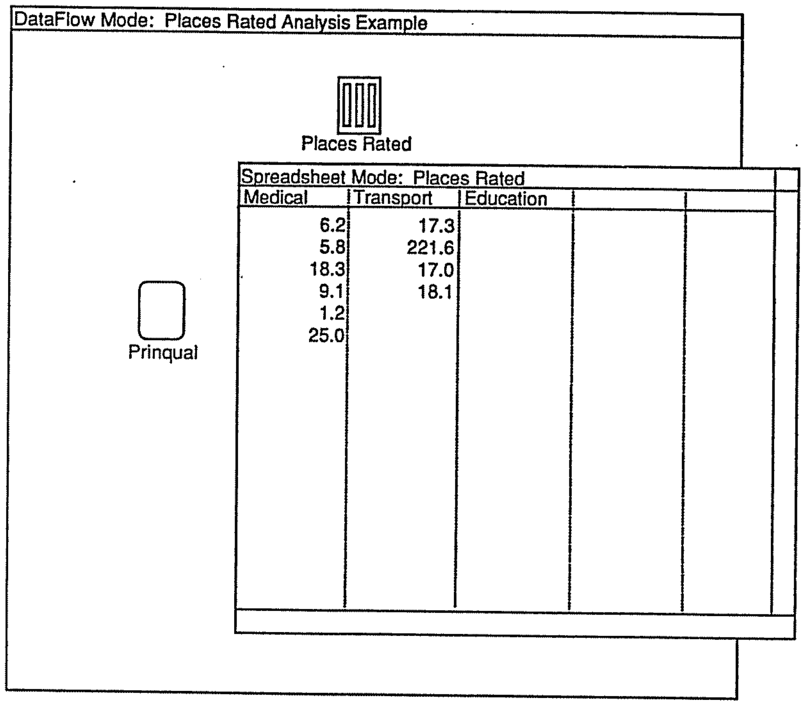


Figure 1.


DataFlow Mode: Places Rated Analysis Example
Places Rated
Prinqual
Spreadsheet Mode: Places Rated

Medical	Transport	Education		
6.2	17.3			
5.8	221.6			
18.3	17.0			
9.1	18.1			
1.2				
25.0				

Figure 2.

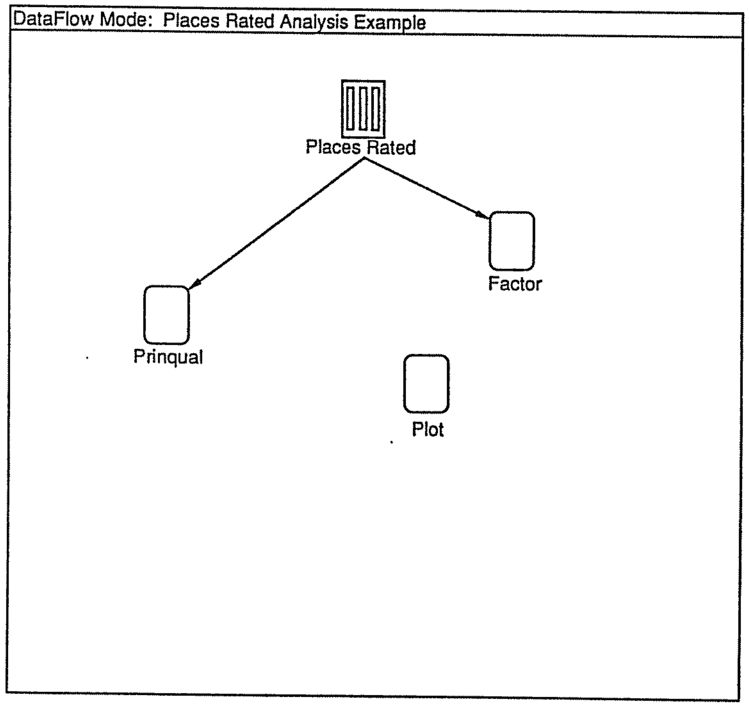


Figure 3.

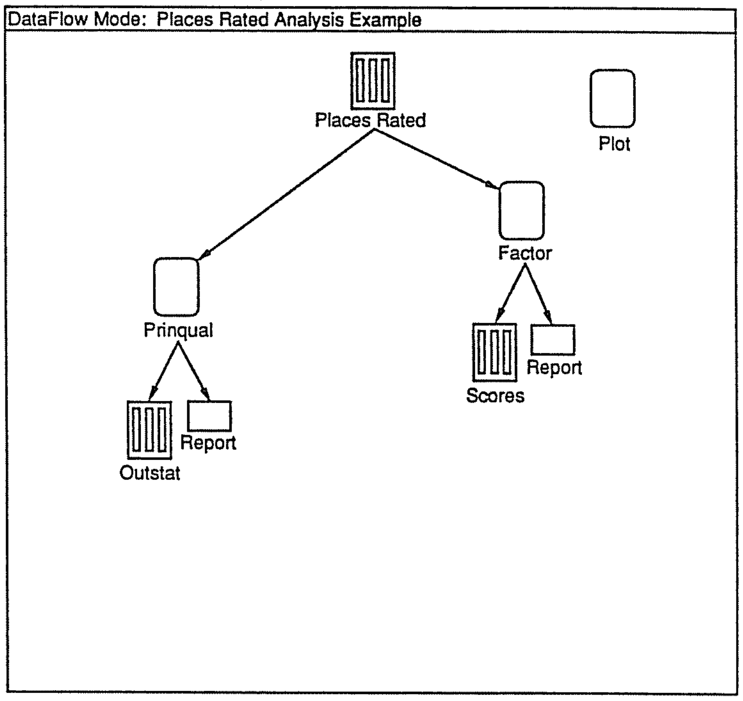
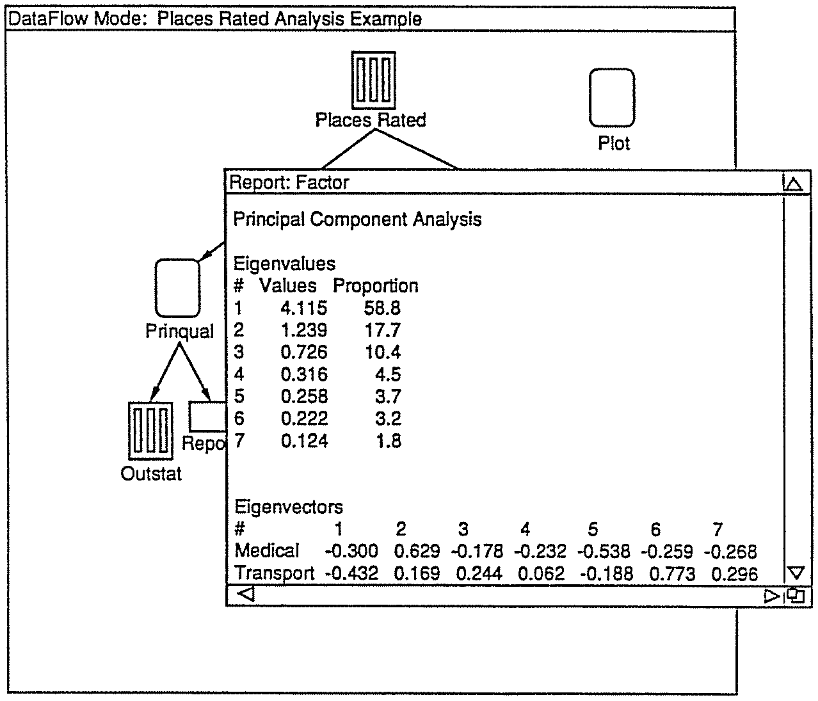


Figure 4.



```
Principal Component Analysis

Eigenvalues
#  Values  Proportion
1    4.115     58.8
2    1.239     17.7
3    0.726     10.4
4    0.316      4.5
5    0.258      3.7
6    0.222      3.2
7    0.124      1.8


Eigenvectors
#            1      2      3      4      5      6      7
Medical   -0.300  0.629 -0.178 -0.232 -0.538 -0.259 -0.268
Transport -0.432  0.169  0.244  0.062 -0.188  0.773  0.296
```

Figure 5.

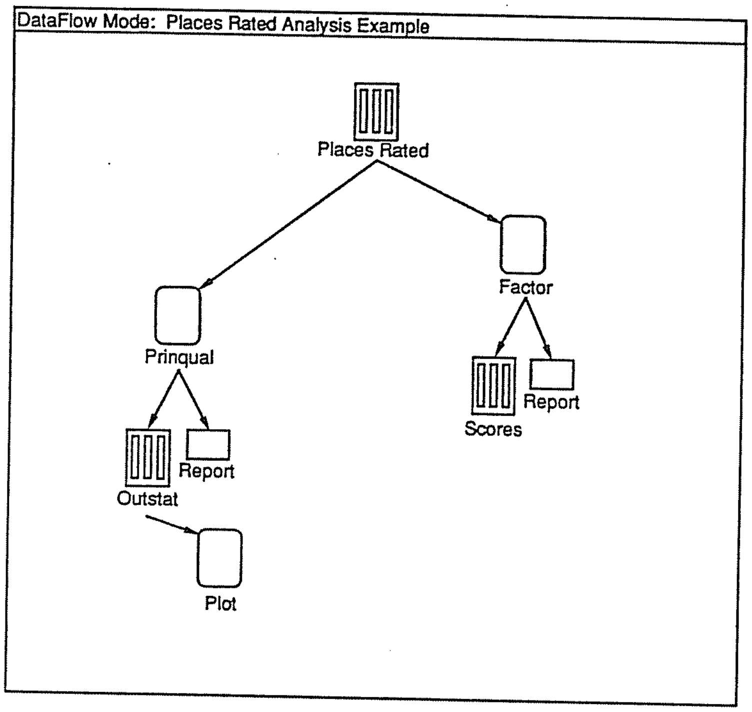
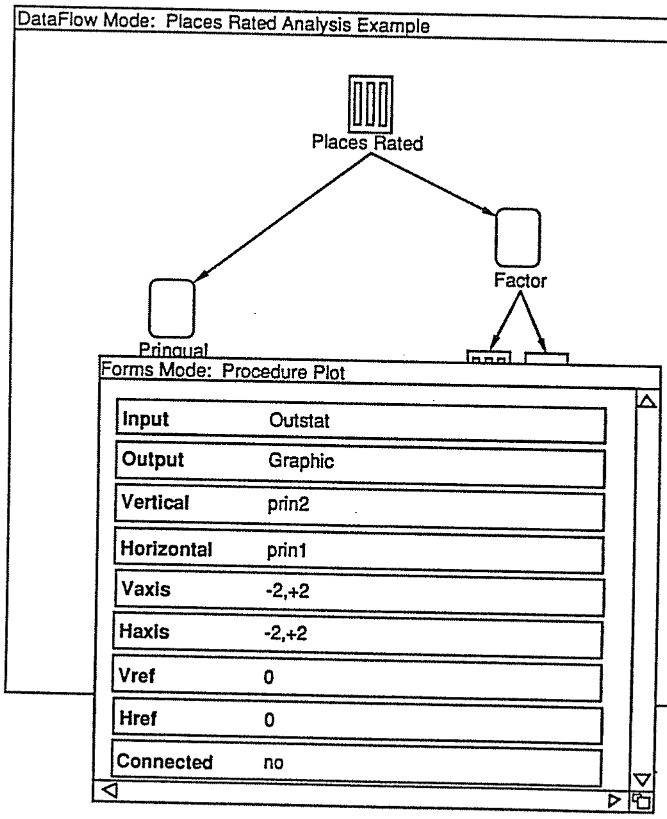
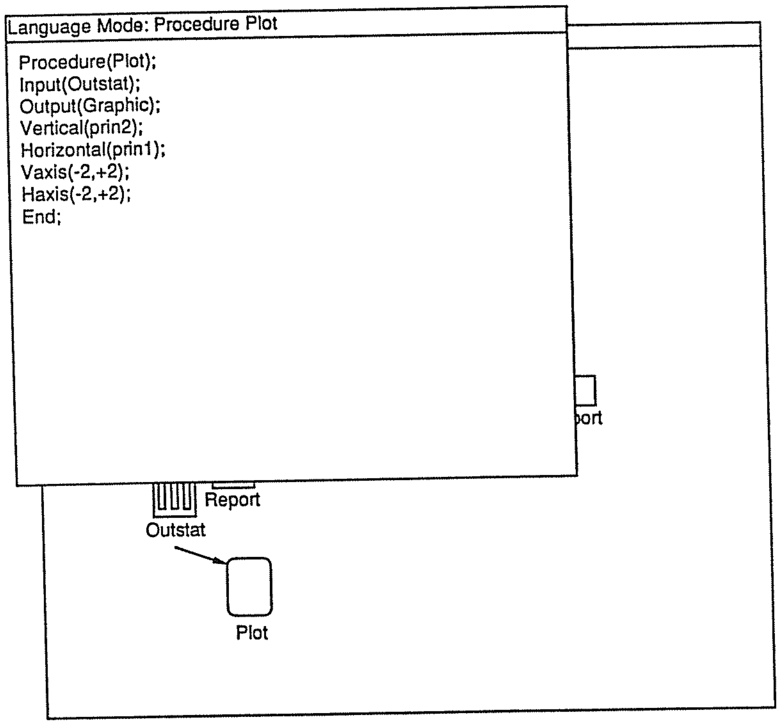


Figure 6.


DataFlow Mode: Places Rated Analysis Example
Places Rated
Factor
Prinqual
Forms Mode: Procedure Plot

Input	Outstat
Output	Graphic
Vertical	prin2
Horizontal	prin1
Vaxis	-2,+2
Haxis	-2,+2
Vref	0
Href	0
Connected	no

Figure 7.



```
Procedure(Plot);
Input(Outstat);
Output(Graphic);
Vertical(prin2);
Horizontal(prin1);
Vaxis(-2,+2);
Haxis(-2,+2);
End;
```

Figure 8.

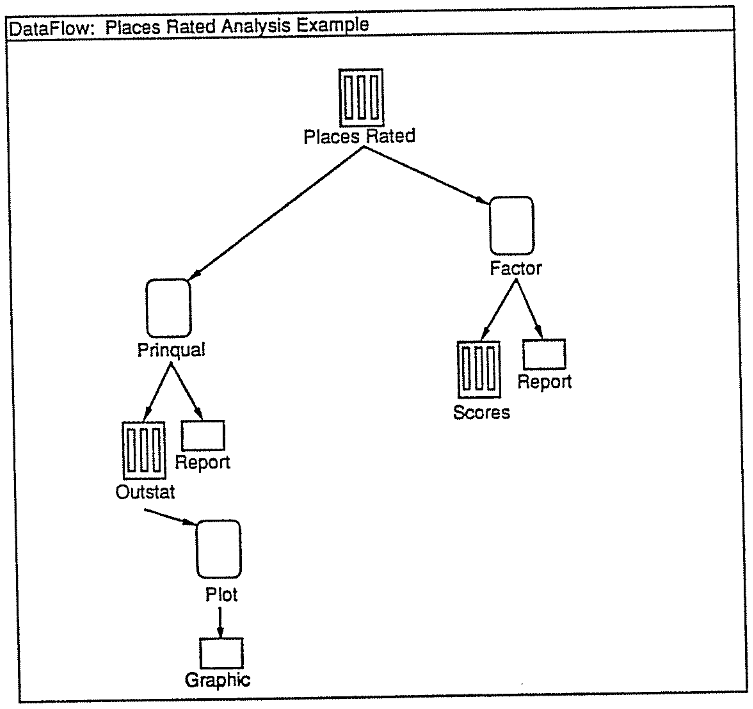


Figure 9.

Figure 10: The data analyst wishes to view the plot, so he/she opens it by clicking on the "graphic" icon. It opens to reveal the plot shown in this figure.

Figure 11: The data analyst now wishes to perform an additional analysis, but first wishes to "clean up the screen". Thus, a "macro" procedure is defined and named "factoring". This macro takes as input the places rated data and produces printed and graphical results of a prinqual analysis, as well as printed results and a dataset of scores from the factor process. We see this macro being defined in Figure 11: The dashed rectangle is drawn by the user around the portion of the dataflow diagram which is to become the macro.

Figure 12: Here, we see that the "factoring" macro process has been closed to hid unnecessary details.

Figure 13: The macro process is now being used in conjunction with further analyses. We also see a "language" process named "merge & sort". This language process has been created by the data analyst, who wishes to merge the factor scores resulting from the factoring macro with the raw data and then sort all observations into order according to their scores on the first principal component. This is done by creating a new language process icon representing the merge and sort process, opening the icon, and entering programming statements in the language of the underlying statistical system.

Figure 14: The (pseudo) statements are shown in the Language Mode. Portions of the language shown here are generated by the system, and portions entered by the analyst. The procedure statement, the two input statements, and the output and end statements have all been automatically generated by the system (on the basis of the dataflow diagram shown in the previous figure). The remaining statements have been entered by the analyst. When the analyst is finished, he/she closes the language mode. The language icon reappears in a form that suggests it is a process.

Figure 15: MIDAS is asked to perform the analysis, which creates the new data set shown in the figure. Note that the macro and language processes play exactly the same role here as system processes such as the Factor or Prinqual processes shown in earlier figures. The only difference is that macro and language processes are defined by the user, whereas system processes are defined by the system.

Figure 16: These new data are further analyzed with four additional data analysis processes.

Figure 17: At this point, the analyst wishes to obtain an overall view of the structure of the data analysis, so structure mode is invoked to obtain a display like that shown in this figure. Note that in this mode the analyst is not actually analyzing the data, but is trying to understand the structure of the analysis as it exists at this point in time. Thus, the structure mode presents a view of the analysis structure that is cleaner and more neatly organized than the view presented in dataflow mode.

Figure 18: The analyst, still in structure mode, wishes to see the flow of data that leads to the print process. By clicking on the print icon, the path from the original data that leads to this icon is highlighted. Note that the path is not a simple hierarchial path, as there are two sub-paths, which are joined by the merge

macro process.

Figure 19: The analyst now wishes to repeat the analysis, but on the logs of the original data instead of the data themselves. Thus, in this figure we see a new macro named "analysis" being defined which will next be used with a user-defined logs process.

Figure 20: The new macro is then readied to analyze the logs of the data.

These figures, then, represent how a typical "messy", unfinished data analysis could take place with the MIDAS system. Note that the structure of the data analysis is always available for the data analyst to see, thus helping keep track of the various analyses which have been performed. While this example is based on an exploratory data analysis where the user wishes to describe the data in order to generate hypotheses, MIDAS can also be used for confirmatory data analyses where the user has formed a prior hypothesis about the results of an experiment and simply wishes to test the hypothesis. In that case, the analysis is simpler: The analyst may only have to create a dataset, connect its icon to the appropriate analysis icon and request the analysis. Of course, if the default option values of the analysis process are not entirely appropriate, the forms or language modes would have to be used to set the options appropriately.

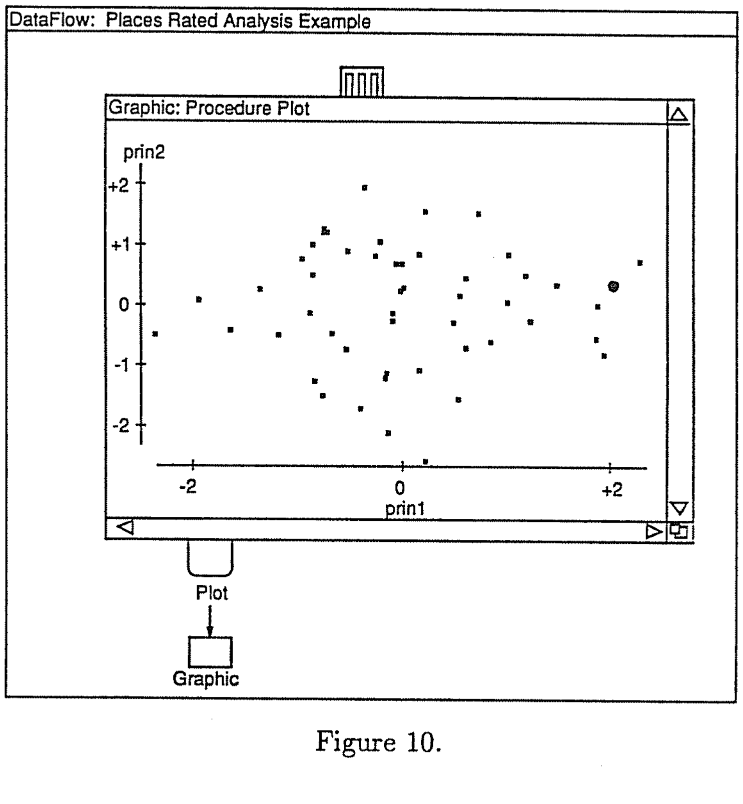


Figure 10.

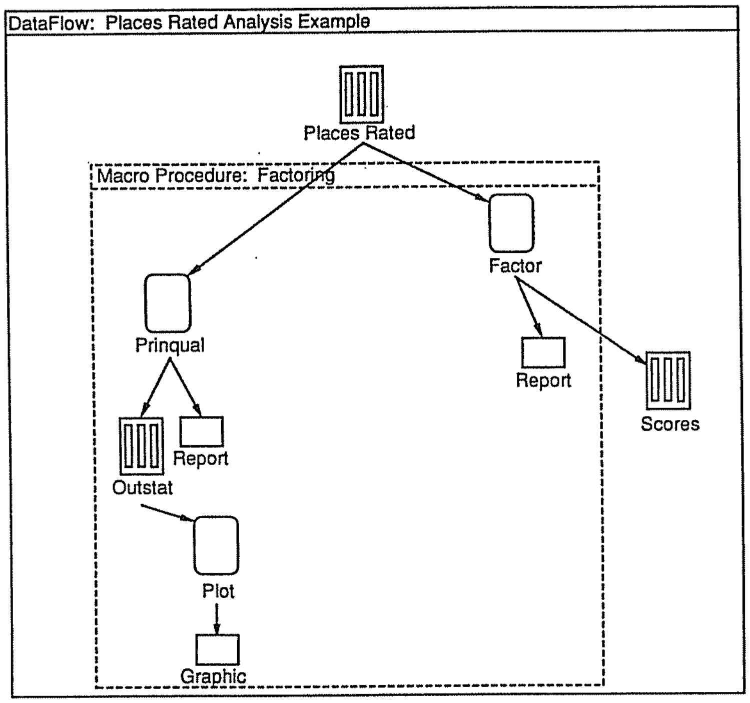

DataFlow: Places Rated Analysis Example
Places Rated
Macro Procedure: Factoring
Factor
Prinqual
Report
Scores
Outstat
Report
Plot
Graphic

Figure 11.

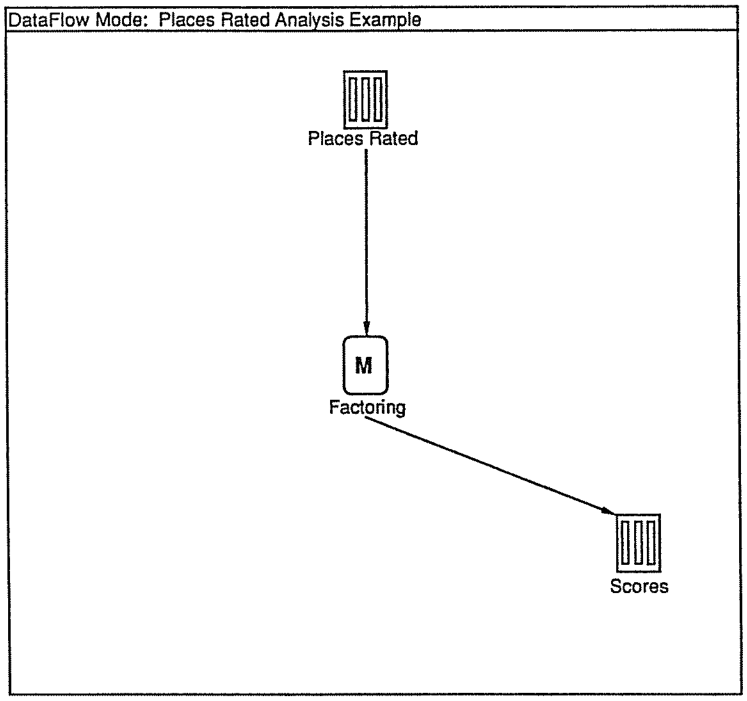


Figure 12.

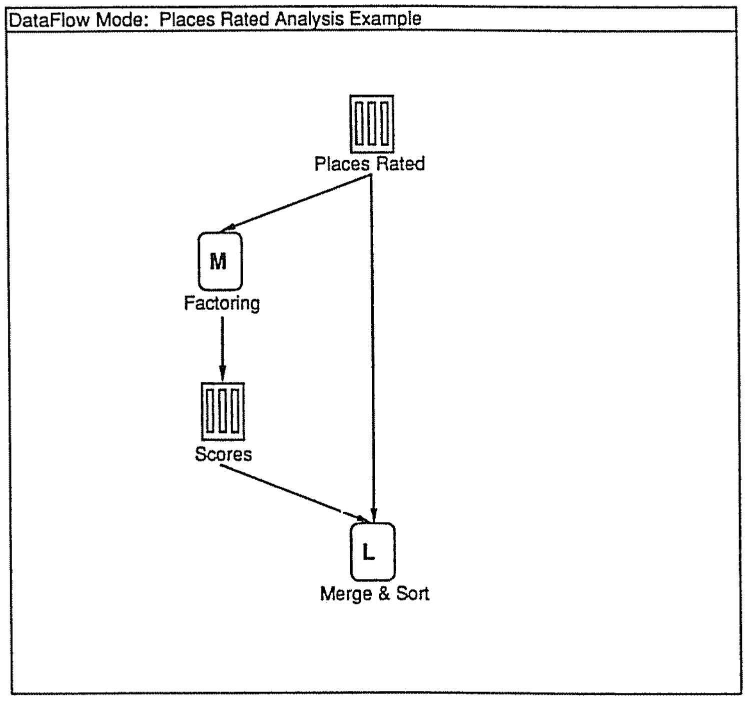
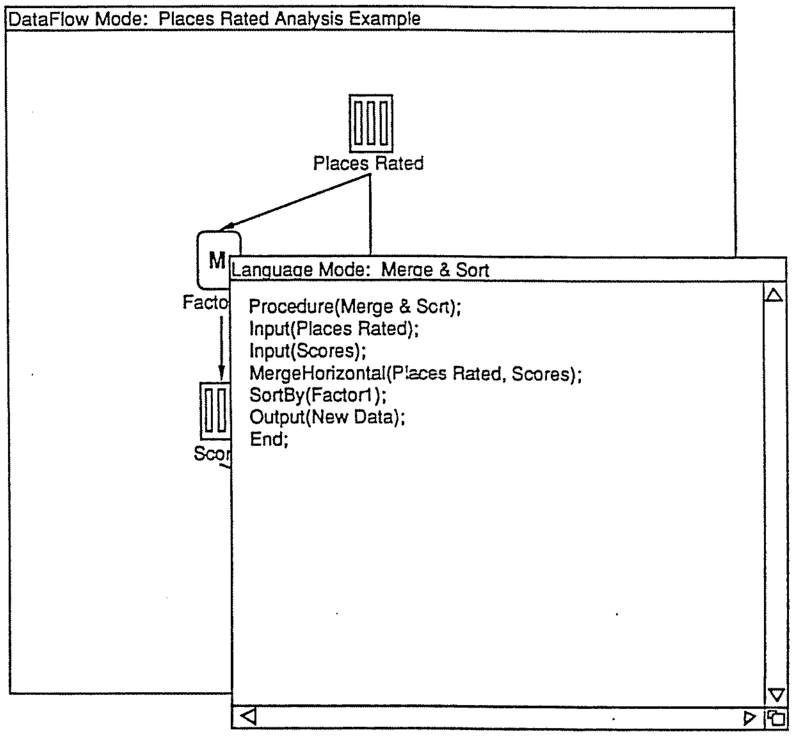


Figure 13.


DataFlow Mode: Places Rated Analysis Example
Places Rated
M
Facto
Scor
Language Mode: Merge & Sort

```
Procedure(Merge & Sort);
Input(Places Rated);
Input(Scores);
MergeHorizontal(Places Rated, Scores);
SortBy(Factor1);
Output(New Data);
End;
```

Figure 14.

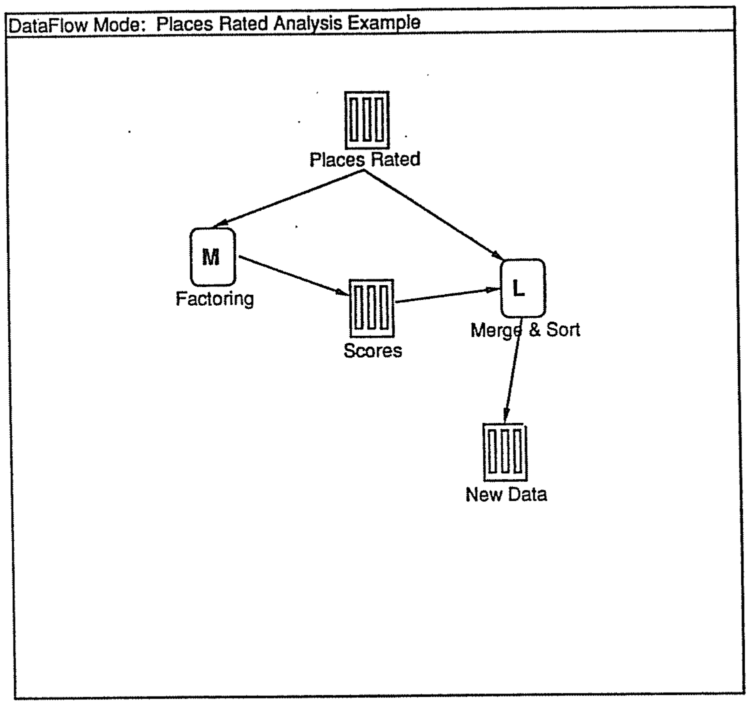


Figure 15.

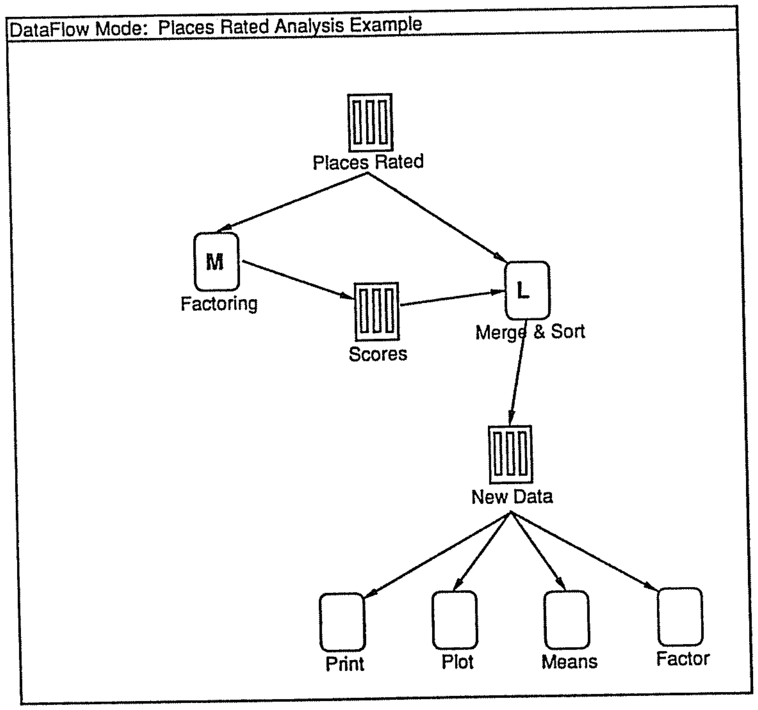


Figure 16.

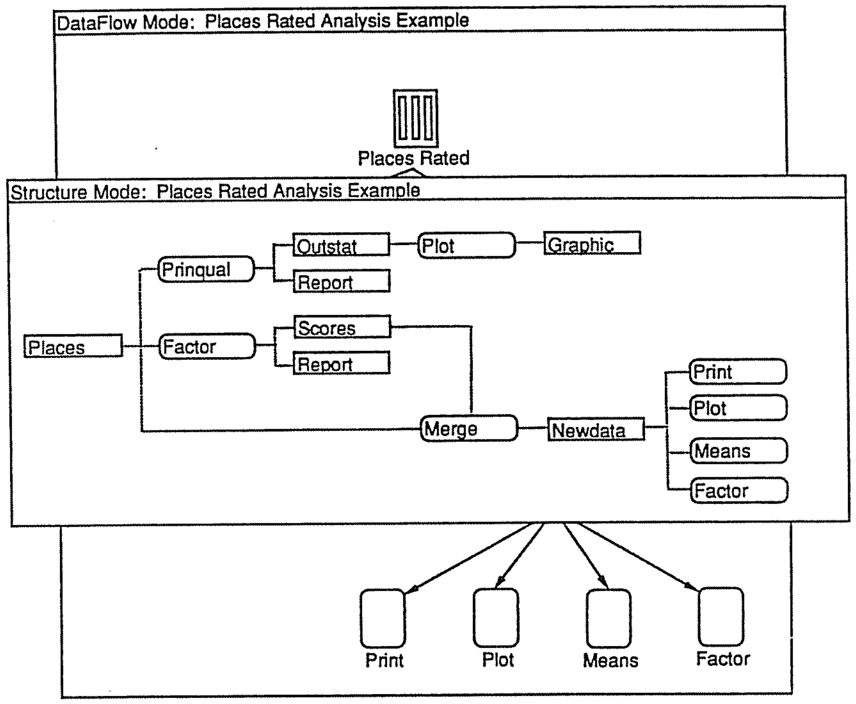


Figure 17.

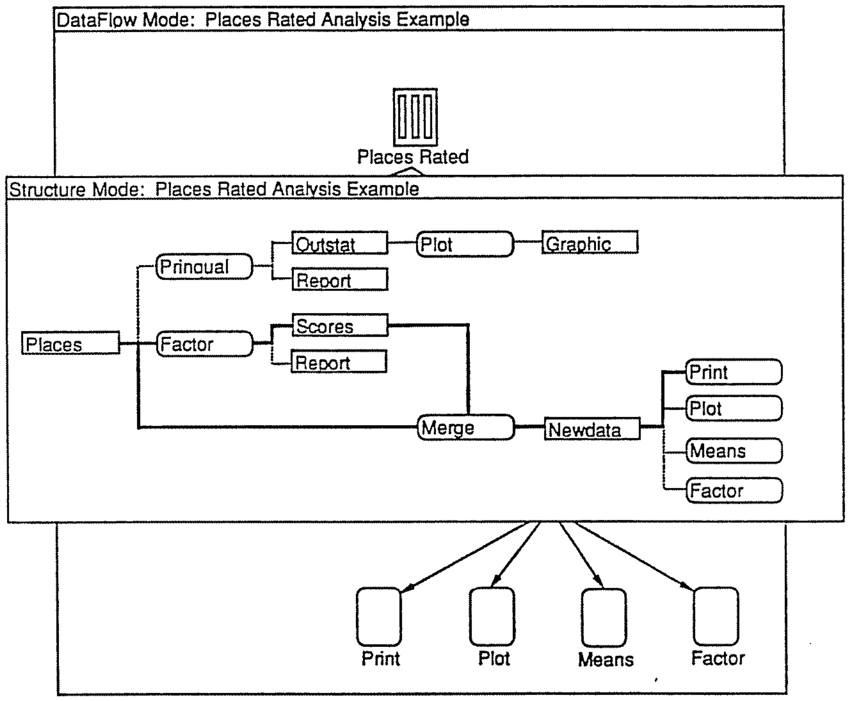


Figure 18.

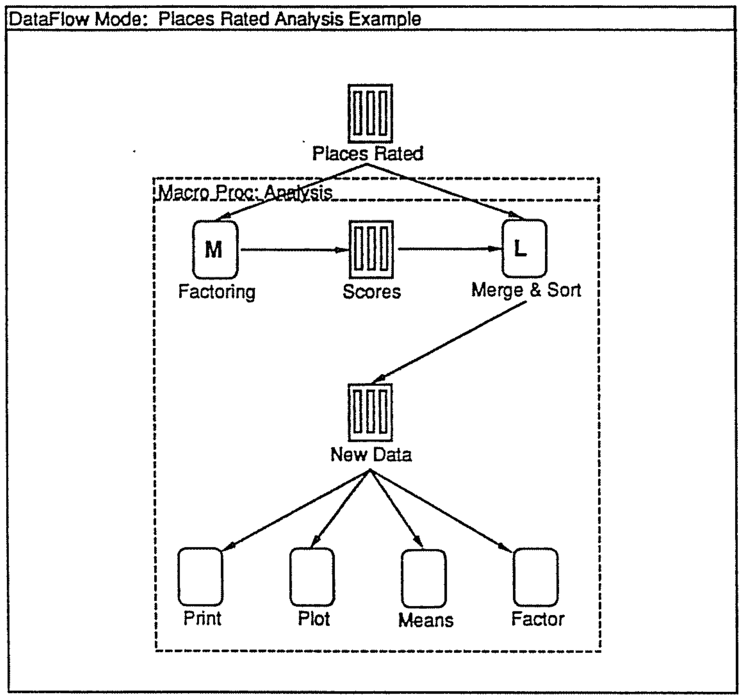


Figure 19.

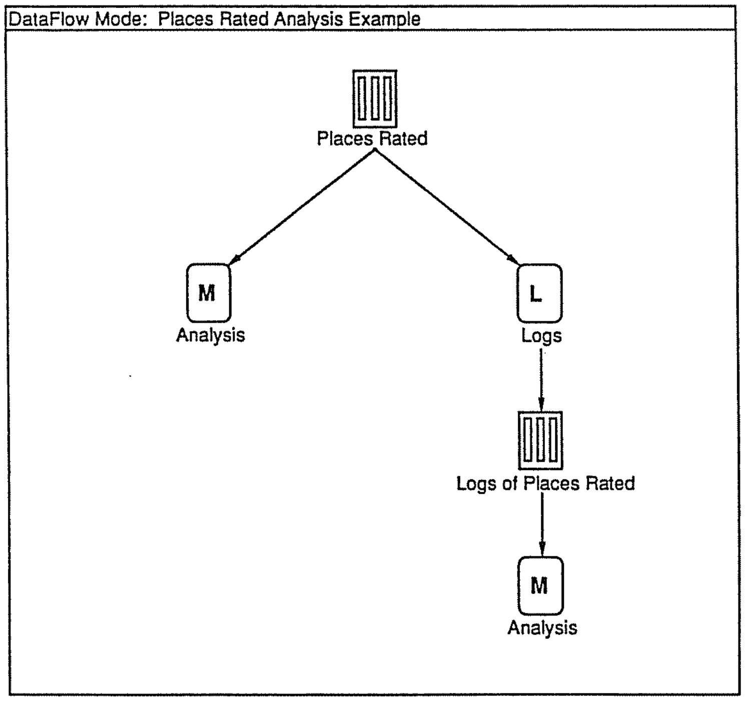


Figure 20.

3. Fundamental Concepts.

Principles of software design based on Cognitive Psychology are at the heart of MIDAS's design. We have searched the literature and have found no previous work in which the design of a data analysis system has been based on cognitive principles (see Lubinsky, Young and Frigge [1990] for a continuation of these ideas). However, principles borrowed from the work of Smith and Lansman [1988], who designed and developed a writing environment using cognitive principles, can be adopted for data analysis systems.

Many aspects of data analysis are analogous to writing. Both are open-ended problem-solving tasks. Both involve building large abstract structures: in the one case, the analytic interpretation of data; in the other, the organizational plan for the document. And, both are iterative processes of conceptual refinement. Thus, a major premise of our work is that the cognitive principles resulting from research in writing can be used to guide the development of a data analysis environment. The major cognitive principle which we adopt is that of Cognitive Modes, a principle which we turn to now.

3.1. Cognitive Modes. Synthesizing concepts from cognitive psychology, reading comprehension, and composition theory, Smith & Lansman [1988] have suggested that writing (and we believe statistics, data analysis, computer programming, and other open-ended intellectual activities) draws on a number of different cognitive modes. They view a cognitive mode as a way of thinking that is engaged in for a particular purpose, that is more or less constrained relative to other modes, which emphasizes certain cognitive processes, and which uses these processes to create certain forms of (intermediate) cognitive products. Thus, a cognitive mode is a conjunction of *goal(s), constraints, processes*, and *products*.

In this section, we review Smith and Lansman's discussion of two of their seven cognitive modes for writing, presenting their illustration of the differences between their exploratory and organizational modes of thinking, interleaved with our suggestions for analogous data analysis modes. We also propose a cognitive mode which we believe to be central to the scientific process, the confirmatory cognitive mode.

Smith and Lansman argue that many writers engage in an early *exploratory mode* of thinking in which the *goal* is to externalize ideas, to consider various possibilities, and to gain a general sense of the material available to be included in the document. In that mode, they argue, *constraints* are loosened, relative to other modes, to encourage creativity and alternative perspectives. The *processes* that are favored are memory recall, associative thinking, categorizing, and noting basic subordinate and superordinate relations. They argue that as a consequence, the intellectual *products* produced by these processes tend to be concrete representations of ideas, clusters of related concepts, and small structures that are often represented graphically.

We argue that the exploratory mode engaged in by writers is very similar to an exploratory mode engaged in by data analysts and underlies the entire branch of data analysis known as *exploratory data analysis* [Tukey, 1977]. As Tukey stated, the purpose of this data analysis "mode" is to "see what the data seem to say".

By its very nature, during exploratory data analysis constraints are loosened to encourage creativity and alternative perspectives. The goal here is to generate as many ideas as possible and to gain a general sense of which analyses can be sustained by the data. The "explorer" attempts to use whatever means possible to generate hypotheses, which will be tested later.

Our notion of the dataflow diagram (described in the next section) supports the kind of exploratory behavior and thinking typical in early stages of data analysis while the data analyst is trying to understand what the data seem to say. The diagram enables analysts to generate as many ideas as possible (by placing icons on the screen) and to gain a sense of what analyses make sense (by connecting icons together and observing results). Thus, MIDAS has a dataflow system mode to support data exploration and hypothesis generation.

We hypothesize that data analysts employ an additional cognitive mode which is in marked contrast to the exploratory mode, and which is not employed by writers: The *Confirmatory* mode. This mode of cognition, which is at the very heart of the scientific process, has a single *goal*: The confirmation or disconfirmation of a scientific hypothesis. In a statistical system, this goal is realized by a statistical test of the hypothesis. The *constraints* on the analyst's behavior and thought are exceedingly tight in comparison to other modes: He or she has formed a specific question to ask of the data, and a specific way to ask the question. The cognitive *processes* include deductive reasoning and focusing on the testing of explicit hypotheses. The cognitive *product* is the knowledge that results from the confirmation or disconfirmation of the hypothesis.

Our notion of language-generating icons is designed to support confirmatory data analysis. Thus, MIDAS has a language system mode to assist analysts as they work in this cognitive mode to confirm hypotheses. In this system mode, the analyst can directly enter statements in the underlying statistical language which will test the hypotheses he/she has concerning his data. He/she can also repeatedly test the same hypotheses on new batches of data by saving the statements for later use. Note that these behaviors, and the kind of thinking that goes with them, are very different than those involved in exploratory data analysis. Thus, we believe that the overall productivity of the analyst will be improved by having the two very different dataflow and language system mode.

Thus, both writers and data analysts explore their data. In addition, data analysts also look more closely at some results in order to confirm or disconfirm specific hypotheses, an activity and a type of cognition that is not performed by writers. We now look at another mode – organization – which is shared by both writers and data analysts.

Smith and Lansman hypothesize that writers often engage in an *organizational* cognitive mode that is quite different from the exploratory mode. Here the writer's *goal* is to organize ideas which have already been generated and written down – to take the set of ideas generated in exploratory mode and form them into a coherent whole. In this mode, *constraints* are tightened, relative to the exploratory mode, to encourage coherence and consistency. Thus, the thinking *processes* tend to involve logic, emphasizing subordinate/superordinate relations among ideas, with the *product* being a single large structure which is the organization plan for the final written piece.

We argue that data analysts employ a similar organizational mode, and that the goals, constraints, processes and products that apply to this mode of thinking for data analysis are similar to those of the analogous mode for writing. As data analysis ideas are generated in the exploratory mode, and/or evaluated in the confirmatory mode, they must be organized so that they don't end up in a confused jumble. When the final hypotheses have been generated and/or tested, they, too, must to be organized into a logical and coherent whole for communication and presentation to others. Thus, the *goal* of a data analyst's organizational cognitive mode is to structure the results of the analysis into a coherent whole that can be easily understood. In order to do this, *constraints* are tightened (relative to the exploratory mode) or loosened (relative to the confirmatory mode) for clarity, logical cognitive *processes* are employed, and a single coherent structure is *produced* to encompass the present state of the data analysis.

Smith & Lansman argue that while different modes represent different ways of thinking, they are not independent. The cognitive products created in one mode often become the raw material that is worked on by the cognitive processes in another. For example, they note that a small hierarchical relation created during exploration might be incorporated into the larger structure being built during organization. Thus, intermediate products tend to flow between modes.

We argue that exactly the same observation is appropriate to data analysis: While exploring data, the data analyst comes across something that the data seem to say. This is represented by a small specific insight about the data. This relation or pattern then becomes the focus of a confirmatory data analysis based on a different portion of data, and is incorporated into the larger interpretive structure being built by the data analyst. It is common that the hypothesis generated during exploration (the product of the exploratory mode) is then tested during confirmation (serves as input to the confirmatory mode process). Thus, in data analysis as in writing, conceptual products tend to flow between modes. No matter whether the data analysis is predominately exploratory or confirmatory, the actual process of data analysis involves many steps in which the basic steps are created (the exploratory mode), intermixed with steps in which the analysis is attempted and revised or corrected (organization mode).

Smith and his colleagues argue that when the person's cognitive modes of thinking are paralleled by the machine's software modes, the human-computer interaction is more natural and more productive. Thus, in designing their writing environment they have attempted to incorporate their theoretical perspective into the architecture of their system, resulting in a multimodal system. As can be inferred from previous paragraphs, MIDAS is also multimodal. MIDAS's five system modes are discussed in the technical description section that appears later in this paper.

3.2. Dataflow Diagrams. A particularly important premise for the writing environment work is that the interface should be highly visual in design and that it should be controlled largely through direct manipulation of spatial representation. Of even greater importance is the premise that the visual representation should be structured. In data analysis, the premises are similar: We believe that the interface should present a highly visual, directly manipulable, structured environment. The dataflow diagram is such an environment.

We believe that these design philosophies, while bearing on human/computer interface issues, go deeper. Our colleague, Marcy Lansman, has observed that cognitive psychologists typically study how people represent the external world *internally*, whereas in studying writing she and her colleagues have reversed the problem, asking how people *externally* represent their internal thoughts. We believe the same situation exists for data analysis. At the starting point, the writer or data analyst has in long-term memory a loosely connected set of ideas that are relevant to the topic at hand. Although some of these ideas are related, they may not be clearly thought through, and are certainly not organized systematically. To externalize these ideas, the writer or data analyst must express them clearly and organize them into a coherent structure.

A major obstacle facing writers and data analysts is that they cannot hold all their ideas in short-term memory at one time. While they are organizing one set of ideas, another set slips out of consciousness. The problem with conventional writing and data analysis systems is that they are not very useful in helping the user *visualize* either the steps or the overall organization of the problem. In order to combine a set of writing or data analysis ideas effectively, it is helpful to *see* the ideas and the relationships among them. Visual images are useful in allowing people to see how ideas are related. Introspections of creative thinkers from many fields suggest that innovative discoveries (i.e., hypotheses) often begin with visual images [Shepard, 1978]. Scientific thinkers as diverse as Einstein and Darwin claim that visual imagery played an essential part in their creative thought processes. Yet conventional writing and data analysis systems do not encourage their users to use spatial representation to create and integrate their ideas.

The writing and data analysis environments enable users to "see" the structure of the steps in their work just as they would see a tangible visual stimulus. Therefore, it is important that the representational format presented by the computer be compatible with the representational format of visual perception. In his theory of visual perception, Palmer [1977] has argued that people represent perceptual information about both the parts of a stimulus and the emergent properties that result from the combination of the parts. For example, a person's internal representation of a face contains not only information about the features of the face, but also information about how those features are combined to produce emergent properties, such as "internal strength" or "slyness". In a similar vein, Baggett & Ehrenfreucht [1985] have shown that when people are asked to assemble a physical structure, parts of the structure are assembled first, then these parts are combined into a whole.

Smith and Lansman draw the direct analogy between the physical structures that people use to represent physical objects and the cognitive structures they use to represent their written ideas. These authors suggest the analogy may apply to a wider range of complex cognitive activities. We specifically extend the analogy to hypothesize that the dataflow diagram is a visual structure that corresponds to one's cognitive structure of a data analysis.

More specifically, we argue that a completed data analysis is analogous to a physical structure: The components of the structure are the individual data analysis steps. If this analogy is appropriate, then the creation of a completed data analysis is like the assembly of a physical structure: when the data analysis is structured it occurs more efficiently. A helpful data analysis tool would enable data analysts to create representations of their work that show not only the parts but the global inter-relationships between the parts. In a data analysis, the parts are the datasets being analyzed, the data analysis processes that are being applied to the datasets, and the results of the analyses. The overall global structure of a data analysis is the exact flow of data from the original datasets through intermediate data analysis processes, and into the representation of the data in written, tabular, or graphical form. The dataflow diagram embodies these ideas. Consequently, MIDAS provides the user with tools to create icons which represent the parts of the data analysis, tools to create the dataflow diagram that represents the data analysis structure, and tools to edit the structure and its parts.

3.3. Hidden Language. One of the three cornerstones of our work is the concept of hidden language: When the user creates a data analysis in dataflow mode, the actions taken by the analyst to create the diagram and its icons have, unbeknownst to the analyst, also generated statements in the language of a statistical system. When the analyst requests that the analysis be performed, these statements, which are hidden below the interface, are submitted to the underlying statistical system. This system performs the analysis specified by the submitted statements, and returns results. These results are then represented by MIDAS as icons which the user, in turn, can open to view, as shown in the example.

Every icon, whether for a dataset or for a data analysis process, has associated with it statements in the underlying language, as is shown in Figures 8 and 14. Each specific process icon has specific default language. These defaults specify details of the analysis which the system designer thinks are most commonly used or are most often appropriate. When the user draws a line connecting a dataset icon to a process icon, the appropriate language is generated to indicate that the particular dataset is serving as input to the specified procedure.

MIDAS is, thus, *both* icon based *and* language based. The analyst can use MIDAS by exclusively interacting with the icons, by exclusively "typing" statements in the underlying language, or by combining icon manipulation with statement typing. We believe that such a hybrid system has several advantages.

The first advantage is that a system which presents modes for both graphical and alpha-numeric interaction will be seen as comfortable and easy to use by a larger number of analysts than systems which have only one of these modes. It is well known that some people are more visually oriented, while others are more language oriented. We hypothesize that analysts who are more visual will prefer to remain in the dataflow mode, whereas those who are more language oriented will prefer to use the language mode. We anticipate that these preferences will also relate to whether the analyst is more comfortable with algebra or geometry.

The second advantage is that the system is appropriate for both naive and sophisticated users, and can help the naive user become more sophisticated. The naive user can use the system without needing any knowledge of the underlying statistical language. He or she doesn't even need to know there is such a system involved. As this user gains experience with MIDAS, he or she can gradually begin to learn the syntax and grammer of the statistical language generated by the icons and, if desired, can begin to directly use the language. The more sophisticated user, on the other hand, can completely avoid the iconic interface and can perform the desired analysis by entering statements via the keyboard.

A third closely-related advantage of a statistical system based on both icons and language is that it can be used for both teaching and research. Such a system has the simplicity and intuitiveness of an icon system that is needed for teaching statistics, and has the power and flexibility of a language system that is needed for the analysis of the complex data that often result from research. Thus, students can be taught data analysis and statistics with a system which is sufficiently intuitive, and can then grow with this system as they become proficient researchers.

A fourth advantage of a statistical system based on both icons and language is that it is suited for both exploratory and confirmatory data analysis. At one extreme, icon manipulation is probably more suited to exploring data than to confirming hypotheses. With icons it is very easy to perform many "what-if" data analyses to look for patterns that may reside in the data. This would seem to be much simpler and more error-free than repeatedly entering sets of statements or making changes in already existing language. At the other extreme, when the analyst knows exactly what analysis is to be done and exactly which hypotheses are to be tested, it may be easier, and less error prone, to directly enter the analysis (or at least portions of it) via the keyboard in statements of the data analysis language.

A fifth advantage of a mixed icon/language statistical system is that it can be used easily and effectively for both simple, one-shot analyses and for complex analyses performed repeatedly on new samples of data. Icon-based systems are ideal for simple analyses which are to be performed once and where it is possible to get the analysis "right" on the first try (or at least in a few tries). Language-based systems, on the other hand, are useful for large and complex analyses which must be performed again and again as new samples of data are obtained. Such systems have the ability to save the data analysis "job" once it is created, so that it can be recalled and used again when the next batch of data is obtained. The hybrid system would seem to be ideal, however, because the original large and complex job could be prepared by using icons (and fine tuned via the language, if need be), and then the underlying language generated by the icons could be saved for reuse with later batches of data.

4. System Description.

A technical description of MIDAS's main system characteristics is presented in this section. The discussion focuses on the system's modes, its dataflow diagrams, and its language generating icons.

4.1. Dataflow Mode.

This provides an environment tailored to the exploratory cognitive mode of the data analyst. In dataflow mode, the data analyst creates data analyses and explores data by using icons and dataflow diagrams (explained below). He or she can also perform confirmatory data analyses in dataflow mode, although users may prefer to use the language mode (described below). Figures 1, 3, 4, 6, 9-13, 15, 16, 19 and 20 of the example focus on activity in dataflow mode.

4.1.1. Icons.

In dataflow mode, the system lets the user represent a data analysis step by allowing the data analyst to create a small *icon* (*node* in graph theory terminology) on the screen. The icons, which are described in detail below, represent either datasets or data analysis processes. The analyst creates the icon simply by using a mouse to select the desired icon from a toolbox of possible icons and then pointing with the mouse to the place in the dataflow mode window where it is to be placed. The placement of an icon in dataflow mode causes a corresponding icon to appear in the proper part of the structure displayed in structure mode. An instance of the language mode is also created, with the data analysis language implied by the icon being automatically generated and displayed (if opted) in the language mode. In addition, dataset icons have associated forms and spreadsheet modes for entering data. The specific icons include the following:

Dataset Icons represent data. The data may come from a file external to MIDAS, or may be entered at the keyboard via the spreadsheet or forms editor. A dataset icon placed in dataflow mode creates a new instance of the language. forms, and spreadsheet modes.

Process Icons represent data analysis processes. There are data analysis process icons such as REGRESSION, CLUSTER, and FACTOR that represent data analysis processes such as Multiple Regression, Cluster Analysis, and Common Factor Analysis. A process icon placed in dataflow mode creates a new instance of the language and forms modes.

Language Icons represent the underlying data analysis language. This icon looks slightly different from other process icons, and creates a new instance of only the language mode. In the icon's language mode the sophisticated user can define his/her own data analysis processes by programming in the data analysis language. An entire data analysis can consist of just one language icon and the language in its language mode. A language process icon named "merge sort" has been created in Figure 13, and statements are being entered into it via language mode in Figure 14.

Macro Icons can be defined by the user graphically via a graphics editor as shown in Figures 11 and 19. A graphically defined macro consists of a portion of an already existing dataflow diagram which is enclosed in a box and named. It can then be closed and saved in the toolbox for later use. When closed it is represented by an icon that is similar to a process icon, as shown in Figures 12 and 20. It can be copied for use elsewhere in the system, as has been done in Figure 20.

4.1.2. Dataflow Diagrams. In addition to representing the steps of the analysis in the dataflow mode window, the overall structure – dataflow – of the analysis is also represented in this window. The data analyst constructs dataflow diagrams like those shown in the figures by locating the mouse's cursor on an icon, holding the mouse button down while dragging the cursor to another icon, and letting up on the mouse button while on the second icon. This causes data to flow from one icon to the other. The user proceeds through the data analysis by placing icons in the dataflow mode's window. As described above, these icons represent datasets and data analysis processes, and have associated with them modes that provide various views of the data and of the analysis processes. As the analysis proceeds, the user and MIDAS together construct a dataflow diagram. This diagram is displayed in unorganized form in dataflow mode, and in structured form in structure mode. The diagram connects together and organizes the data icons and analysis icons to represent the flow of data from data icons through process icons. In addition to the icons for datasets and analysis processes, the data flow diagram also consists of directed arrows connecting the icons. The arrows indicate the direction of flow of data from datasets through analysis processes, into new datasets.

4.2. Language Mode. This provides an environment tailored to the confirmatory cognitive mode of the data analyst, and to the sophisticated data analyst who is oriented more algebraically than geometrically. Language mode provides a standard program editor which allows the user to directly enter statements in the underlying data analysis language, and which allows the knowledgeable data analyst to expand the underlying data analysis language that is automatically generated by dataflow mode. The professional user can use language mode to gain full access to the statistical system underlying the interface, and can in fact prepare an entire analysis in language mode. On the other hand, the novice user need only deal with dataflow mode and its icons, and need not be concerned with what statistical system is actually performing the analysis, nor with the syntax for that system. Figure 8 and 14 present examples of language mode.

4.3. Structure Mode. This provides a system environment tailored to the organizational cognitive mode. Figure 17 displays the kind of view in structure mode, while Figure 18 demonstrates the tool that highlights the analysis stream leading to a specified step. The primary goal of this cognitive mode is to present the structure of the data analysis as coherently as possible. Although data analysts can construct coherent structures in dataflow mode, we elect to support the creation of data analyses and the organization of data analyses in separate system modes because the two cognitive activities are quite different. In creation, constraints are lowered to emphasize flexibility; in organization, constraints are tightened to emphasize coherence and consistency.

4.4. Forms Mode. This reveals a form for entering data into datasets and for setting data analysis process parameters. This mode is tailored to the naive user who does not wish to use the language or spreadsheet modes. The mode is associated with all icons, including newly created (empty) or already existing dataset icons, as well as process icons. If the mode is associated with a newly created (empty) dataset, then the mode presents a form for entering variable names and characteristics. If the mode is associated with an existing dataset, then the form contains variable names and has blanks into which variable values can be entered. If the mode is associated with a data analysis process, then the form presents the parameters of the process and the current (default) parameter values. These parameter values can be changed by the user. Figure 7 displays what might be seen in forms mode.

4.5. Spreadsheet Mode. This mode is tailored for the data analyst who wishes to enter or edit data. With a new (empty) dataset, this mode reveals an empty spreadsheet, into which data can be entered from the keyboard. With an already existing dataset, this mode reveals the data and makes it available for editing with a spreadsheet editor. If variable names have been assigned, they are shown here as column labels. Other characteristics or attributes of the variables can also be revealed. Figure 2 presents an example of the spreadsheet mode.

5. Summary.

We have presented MIDAS, a structured software environment for data analysis. MIDAS is designed to improve the data analyst's productivity over unstructured environments, because it is designed to correspond with the cognitive processes we believe are used during data analysis.

MIDAS is consistent in both design and theory with the similarly structured writing environment discussed by Smith and Lansman [1988] and with the dynamic statistical graphics methods of Young, et al. [1988]. A structured environment which integrates writing, data analysis, and dynamic graphics is one in which scientists can analyze their data, view their data with dynamic graphical techniques, and write reports about their results. Such an environment seamlessly integrates all of these activities so that is is inherently straightforward to include the analyses and their results, both tabular and graphic, in the written reports.

The design of MIDAS is based on a theory of cognitive modes. The theory predicts that scientific productivity will improve when scientists shift from unstructured environments for writing, analyzing, and graphing data to structured environments for these activities. The theory is testable: Indeed, we anticipate using MIDAS, once it is completed, as a testbed for the theory. Smith and Lansman's writing environment includes tools for generating protocols of a writer's session, and tools for inferring the cognitive modes of the writer during the session. These same tools will be available in MIDAS, permitting us to investigate the cognitive processes of data analysts as they use MIDAS to analyze their data. We will be able to fully explore the general patterns and strategies of data analysts as they move from early exploratory data analysis, to hypothesis-testing, through refinement of their interpretations, and into writing reports of findings. We will also be able to empirically confirm or disconfirm the major hypothesis of our work: Data analysis is more productive in a structured environment than in an unstructured environment.

We hope to implement and evaluate MIDAS within the next two or three years. In the meantime, we hope to encourage further discussion about the complex cognitive activities that underlie data analysis and about how we can best exploit the capabilities now available in advanced computer workstations to support those activities.

REFERENCES

BAGGETT, P. & EHRENFREUCHT, A., *Conceptualizing assembly tasks*, Technical Report # 139. Boulder, CO: Institute of Cognitive Science, The University of Colorado. (1985).

BECKER, R.A. & CHAMBERS, J.M., S: An interactive environment of data analysis and graphics, Belmont, CA: Wadsworth (1984).

BLANK, G., *The obsolescence of the SAS System*, Proceedings: SAS Users Group, (1985), pp. 279–284.

BORNING, A., *Defining Constraints Graphically*, CHI'89 Proceedings (1986), pp. 137–143.

LUBINSKY, D.J., YOUNG, F.W., & FRIGGE, M.L., Representing and Using Data Analysis Structures, ASA 1990 Proc. Sec. Computational Statistics.

McDONALD, J.A. & PEDERSEN, J., *Computing Environments for Data Analysis I: Introduction & II Hardware.*, SIAM J. Sci. Stat. Computing (1985), pp. 1004–1021.

McDONALD, J.A. & PEDERSEN, J., *Computing Environments for Data Analysis III: Programing Environments*, SIAM J. Sci. Stat. Computing (1988), pp. 380–399.

OLDFIELD, R.W. & PETERS, S.C., *Object-oriented Data Representations for Statistical Data Analysis*, Proceedings, Compstat-86 (1986).

PALMER, S.E., *Hiararchial structure in perceptual representation*, Cognitive psychology, 9 (1977), pp. 441–475.

SAS INSTITUTE INC., SAS User's Guide: Basics, Version 5 Edition, Cary, NC: SAS Institute Inc. (1985).

SHEPARD, R.N., *Externalization of mental images and the act of creation*, In Randhawa, B.S. & Coffman, W.E. (Eds); Visual Leárning, Thinking, and Communication, New York, NY: Academic Press (1978), pp. 133–189.

SMITH, J.B. & LANSMAN, M., *A Cognitive Basis for a Computer Writing Environment*, In Britton, B.K. & Glynn, S.M. (Eds.); Computer writing aids: Theory, Research and Practice (1988).

SPSS INC., SPSS^X User's Guide, New York, NY: McGraw Hill (1983).

STUETZLE, W., *Plot Windows*, J. Amer. Statistical Association, 82 (1987), pp. 466–475.

TUKEY, J.W., Exploratory Data Analysis, Reading, MA: Addison-Wesley (1977).

VELLEMAN, P.F. & VELLEMAN, A.Y., Data Desk Professional, Northbrook, IL: Odesta Corp. (1988).

WOODS, W.A., *Transition network grammars for natural language analysis*, Communications of the ACM, 13 (1970), pp. 591–606.

YOUNG, F.W., *A Modern Interface to Data Analysis Systems*, Unpublished Manuscript, L.L. Thurstone Psychometric Laboratory, University of North Carolina (1988).

YOUNG, F.W., KENT, D.P. & KUHFELD, W.F., *Dynamic Graphics for Exploring Multivariate Data*, In: Cleveland, W.S. and McGill, M. (Eds.); Dynamic Graphics for Statistics, Belmont, CA: Wadsworth, (1988).