International Statistical Review (1997), **65**, 2, 167–189, Printed in Mexico

Software for Learning and for Doing Statistics

Rolf Biehler

Institut für Didaktik der Mathematik (IDM), Universität Bielefeld, Postfach 10 01 31, D-33501 Bielefeld, Germany.
e-mail: rolf.biehler@post.uni-bielefeld.de

Summary

The community of statisticians and statistics educators should take responsibility for the evaluation and improvement of software quality from the perspective of education. The paper will develop a perspective, an ideal system of requirements to critically evaluate existing software and to produce future software more adequate both for learning and doing statistics in introductory courses. Different kinds of tools and microworlds are needed. After discussing general requirements for such programs, a prototypical ideal software system will be presented in detail. It will be illustrated how such a system could be used to construct learning environments and to support elementary data analysis with exploratory working style.

Key words: Statistics education; Statistical software design; Evaluation of statistical software; Exploratory data analysis; Simulation.

1 Computer Support for Introductory Statistics Education

What kinds of computer software do we need to support introductory statistics education? We will analyse this problem on two levels. In the first 4 parts of this paper, a general framework of discussion will be established. Basic requirements and problems for evaluating and designing software for learning and for doing statistics will be elaborated. In part 5, we will become much more specific. We will present detailed features of a prototypical software tool that would support learning and doing statistics more adequately. We will finish with exemplifying how such a prototype can be used to define learning environments that are useful for students.

Our notion of introductory statistics education is not restricted to college and university levels but also comprises the secondary level. If we browse through the literature, we can distinguish various types of software having different educational functions. Among these one can distinguish *tools, microworlds, resources* and *tutorial shells*.

We need tools or rather computing environments in the sense of Thisted (1986) if we want students to learn to practise statistics similar in style to that of current statisticians in their computer supported working environments.

We use microworlds as a notion that summarises interactive experiments, exploratory visualisations and simulations in which students have some limited opportunities to play with graphs, methods and parameters which may help them with conceptualising statistics. Typical examples are manipulating the shape of a distribution and observing how this affects numerical summaries, manipulating a fitted line and observing how this affects the quality of the fit, changing data and observing how this affects the value of a correlation coefficient, observing how initial distribution, number of repetitions and sample size affect the distribution of the mean (visualising the central limit theorem).

The notion resources comprises first of all data and meta-data: Interesting data sets with background

information (stories) and references that help with the interpretation. Several resources of this kind are available on the internet. Also, a hypertext glossary as a resource for statistical terms and methods could be part of it, as well as interactive movies with applications of statistics in multimedia environments (Moore, 1994).

We could consider broader roles of computers in introductory statistics education, i.e. taking over parts of roles of teachers and books. For instance, giving expositions and explanations, setting tasks for the students, analysing and evaluating student responses and providing feedback. Authoring systems like *Toolbook* or *Hypercard* are already used by some projects to program tutorial shells that provide an interface with other software and integrate the use of other tools and microworlds. In many educational programs, the distinction between tutorial shell and microworld can be made only as a conceptual one: they are in fact amalgamated—which is a disadvantage if someone would like to use the microworld but not the tutorial approach.

If we assume that teachers have an active role in interpreting and realising a curriculum or even that teachers want to explore new curriculum content or new sequences of content, this has consequences with regard to requirements of software support. Tools, microworlds, resources and tutorial shells should be adaptable—at least to a certain degree—by teachers to their specific audience and educational goals. Software has to have a *teacher interface*, a specific capability as a meta-tool or meta-medium. We will speak of *teachers'* meta-tool. We can summarise the components as in Figure 1.

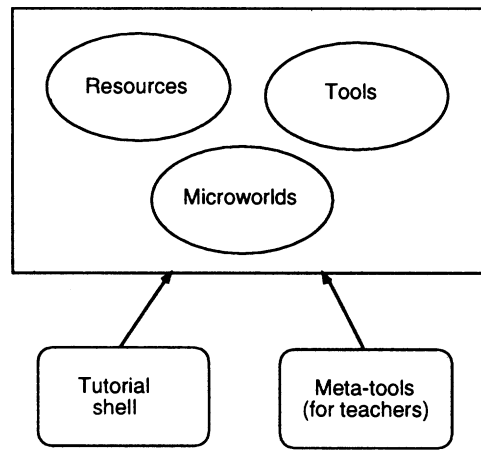

Figure 1. *Components of educational software*

These distinctions are partly conceptual and need not correspond to actual software components. For instance, there are programmable data analysis systems that can be used to construct microworlds within the system, namely through a limited set of functions and commands as in *Minitab* or *Splus*. Some programs even support the construction of tutorial shells within the system itself. An example is the programming of so-called SUCROS in the system *Survo 84c* (see e.g. Puranen, 1994). Other examples are the new interface features of *StatView* (since version 4.0) and of *Data Desk* 5.0 that can be used to construct limited tutorial shells, for instance for introducing the use of the program for certain statistical analyses.

How does a reasonable list of priorities for software support look like? This paper is based on the following rank order (with descending priority):

- student tools whose learning and use can be integrated into an introductory course
- resources of data and meta-data

- microworlds (a good but limited selection)
- further resources (electronic books, multimedia resources)
- tutorial shells

Teachers' meta-tool functions should be included at all levels.

2 Aiming at Improvements in the Universe of Existing Software

While software is only one aspect of statistical education, the quality of software can have a major impact on the quality of education. Thus, the community of statistics educators should take responsibility for the evaluation and improvement of software quality from the perspective of education. We should aim at developing a perspective, a guideline, an ideal system of requirements to critically evaluate existing software and to produce better software in the future. The identification of key elements of software that are likely to survive the next quantum leap of technological development and that are fundamental for introductory statistics should become an important research topic. Results should guide new "home grown" developments of educational programs and should influence the adaptation and elaboration of existing statistical systems toward systems more adequate for purposes of introducing and learning statistics.

When we look at the landscape of existing software, we can identify three basic problems that I will call the *complexity of tool problem*, the *closed microworld problem*, and the *variety problem*.

The tool problem becomes worse as the educational levels gets lower. Professional statistical systems are very complex and call for high cognitive entry cost. They are often not adequate for novices who need a tool that is designed from their bottom-up perspective of statistical novices and can develop in various ways into a full professional tool (not vice versa). A co-evolution of tool and user should be facilitated. This could mean two things: the adaptability and extensibility of the tool from the students' perspective and a conceptual structure of the tool which supports its piecewise appropriation. As a rule, current student versions of professional systems are no solution to this problem because they are technical reductions of the complete system. They provide, however, some relief with regard to the financial problems of buying professional systems.

The closed microworld problem refers to the problem that microworlds are of necessity constrained to enable students to concentrate on essential aspects of a learning situation and to make likely certain intended cognitive processes. Many educational programs contain microworlds that cannot be changed and adapted. They are therefore criticised or are simply not used by teachers because they will not fit their teaching-learning situations. A specific disadvantage can be that microworlds use artificial data or only a fixed real data set. They are not open for working with user selected data. Often, microworlds incorporate some limited tool functionality, for instance to explore a certain method's action on data sets. However, their support for flexible data analysis is limited. To satisfy the variety of demands in a course, one would need a collection of microworlds. A certain solution to this problem is to use *embedded microworlds*: A larger host system is used to construct, modify and enlarge a microworld and to continually extend activities from microworlds to working on problems using a tool. However, embedding often implies that microworlds will share features of the host system (for instance formal language interface). This may not be desirable and come into conflict with the microworld's aim to support a specific cognitive process, because the user also has to learn host system features and notation.

The variety problem is illustrated by the various collections of microworlds that exist. A more systematic and co-ordinated approach to microworlds is missing. The variety problem exists also with regard to tools. A good example is an undergraduate course of Snell (1992) where three tools were used, namely *StatView* as a data analysis tool, *Minitab* to do some simulation experiments and *TrueBasic* to design more complex microworlds and new methods. In general, coping with uncoordinated interfaces, notations, ideas and a large number of tools in one course would overtax

the average teacher and student.

Just as the history of statistical computing has lead us from collections of special purpose programs to integrated and adaptable tools for data analysis, the parallel idea of an integrated tool or an integrated system of co-ordinated tools for educational purposes seems fascinating. Adaptability (including extensibility) is a central requirement for data analysis systems to cope with the variety of needs and users. Current solutions of adaptability are far from satisfactory (Steinecker, 1990). We may even hope that adequately adaptable tools for doing statistics can also be co-used as a host system for defining and modifying microworlds for learning. As learning statistics should involve doing statistics (with computer support) in any case, a single adaptable tool may become all we need.

Obviously, there is a tension between solving the variety problem and the complexity of tool problem. We have to analyse to which extent the variety is just a result of uncoordinated and pragmatic developments and to which extent it reflects different task domains with different conceptual and operational structure. A coherent system of collaborating (modular) tools with well defined interfaces, minimal functional overlap, adequate data transfer and object linkage facilities between the components may be an alternative to one "super" host system with high complexity.

In statistics, for instance, it could be advantageous if we had a modelling and simulation tool co-ordinated with a data analysis tool instead of using *StatView* and *Minitab* as in the above quoted course with their large functional overlap and different philosophy and interface design. If a model is constructed with a modelling tool, and simulated data from the model have been generated, these data should be transferable to a data analysis tool. In addition, a hot linkage of the data to the model tool may support exploring changes in the results, when the model or the random seed is changed.

3 Kinds of Useful Tools

What kind of tools do statisticians need? Thisted (1986) distinguishes three types of functional environments "*Data analysis environment*", "*Monte Carlo workbench*" and "*Theoretical statistician's environment*". He specifies basic requirements for the first from the perspective of how to support certain statistical *strategies*, especially in Exploratory Data Analysis. Supporting the construction of models and doing simulations leads to requirements that are different from those for supporting the construction and analysis of new methods by a theoretical statistician. Existing software tools have different profiles of usefulness with regard to the above three (conceptual) environments (see Biehler & Rach, 1990a; Biehler & Rach, 1990b). My basic argument is that we need elements of these three environments for educational purposes, too. I will re-examine requirements for these environments from an educational point of view. Requirements of software are dependent on intended educational uses. However, a synthesising theoretical and empirical analysis of the most efficient and promising uses of computers is still lacking. We will start with globally describing assumptions concerning the direction in which introductory statistics education should move and improve. These have been discussed in more detail elsewhere (Biehler, 1992; Biehler, 1993; Gordon & Gordon, 1992; Thisted & Vellemann, 1992). We may also refer to Moore (1997).

Directions of educational changes with distinct functions of computer use include:

(1) Students practice data analysis with exploratory working style. The kind of tool would not be so important if its major use were just executing some isolated commands in order to simply apply a single method learned theoretically before. But students should be empowered to do some "small scale" data analysis work that has the interactive, exploratory and open-end character of real data analysis in practice. Students should be empowered to combine exploratory and inferential methods, graphical and numerical methods as well as integrating data base activities, data transformations, and documentation of results into their work. For these purposes of supporting an *exploratory working style*, an integrated student tool similar to flexible professional tools is required. However, an improved balance between flexibility

and ease-of-use and ease-of-learning is necessary.

(2) Students participate in "research in statistics". They participate in constructing, analysing and comparing statistical methods. Those aspects of research that can be performed by computer based simulation studies or experiments on data sets promise to be more accessible to students than mathematical theory. For instance, students may explore sampling distributions of t-test statistics under several probabilistic assumptions. Students may re-discover and conceptualise the difference between median and mean as estimators of central tendency and so on. Generally, an exploration of alternatives to standard methods may become feasible. For instance, students may suggest and explore their own graphs, statistical summaries and tests, because custom-designed methods can be explored by simulation and resampling methods although a mathematical (analytical) theory does not exist. In other words, we consider a guided discovery approach to the *methods and concepts* of statistics, and not only with regard to applications of statistics. For that purpose, we require a software tool that has elements of a "Theoretical statistician's environment" and of a "Monte Carlo workbench".

(3) Students construct models for simple and multistage random experiments and use computer simulation to study them. This is an important part of learning elementary probability with many advantages over just working with ready made simulations (Biehler, 1991; Konold, 1991). Analysing statistical methods on the basis of one or more probability models, simulating experiments for demonstrating, experiencing and coping with situations of uncertainty are additional important pedagogical uses. From this perspective, a student tool for simulation and modelling, a "Monte Carlo workbench" is required.

(4) Teachers explore new curriculum content and new sequences of content. A software tool should support the educational exploration of various new emphases in the content of teaching. For instance, bootstrapping and permutation tests promise to provide a more "natural" introduction to inference, after students have learned exploratory data analysis. A software tool should be consistent with various alternatives or be adaptable to various purposes. The question of sequencing is particularly problematical when we start looking from the contents of general education in schools to more sophisticated levels, and not vice versa. The principle of EDA to start an analysis with simple methods and use more sophisticated methods afterwards, if necessary, could be reflected in curriculum construction. For instance, eye-fitted lines with residual analysis should precede the method of least squares. Or, the comparison of distributions by box plots precedes t-tests and ANOVA. But even if we would not aim at such shifts, the structure of an adequate tool should reflect the students' bottom-up perspectives and not only the top-down perspective where a user has to cope with a fully sophisticated system of statistical methods from the beginning.

Teachers should be enabled to adapt and modify material and software for their students. Ideally, a software has the character of a meta-tool and meta-medium. i.e. teachers are able to redesign a tool, to make it simpler or better adapted to the specific curriculum and students' competence.

4 Kinds of Useful Embedded Microworlds

A research program to overcome the closed microworld problem and to identify relevant microworlds would have to proceed on two levels: Firstly, we have to analyse and compare existing microworlds and their educational potential with the aim of identifying some general elements and common building blocks, and secondly a critical analysis of available host systems is necessary with

regard to their capability of adequately hosting the required educational microworlds.

The idea to co-use statistical data analysis systems for building educational microworlds is not new. Extensible systems with interactive and directly interpreted command languages offer good opportunities if they are designed with a view towards statistics. Systems like *Minitab, PC-ISP, Survo, Splus, xlispstat, APL*, spreadsheets and general mathematical systems like *mathematica, Maple* and *Mathcad* have been used and extended for these educational purposes. Each system offers its own type of microworld construction; they may be called macro, notebook, template and so on. Some also support the writing of electronic books or tutorial shells around the microworlds. We do not discuss educational use of these systems in this paper, but see Naeve (1991) who discusses the educational advantage of statistical language systems like *APL* if leading university students from performing experiments to writing their own is a basic aim.

In secondary education, but also in introductory statistics in higher education, using a command language system is problematical. We are convinced that a host system with a graphical user interface offers a more adequate basis.

An essential point is how easy it is to define microworlds, whether programming expertise is needed or whether we can do "programming by example". Kaput (1988) has underlined the centrality of this idea for improving educational software in general, see also Witten (1992) for the general background of this notion. Two trends in software maturation meet these requirements: the general effort of providing easier ways of customisation and the trend of statistical tools to support more open explorations.

The notion of programming by example can be explained in the context of a spreadsheet. Starting from a column of data, we can use a new cell for calculating the mean of these data and another cell for the median. We can also open a diagram with a graph of these data. The fact that all the new cells and diagrams are dependent on the original data shows up when we change a number in the original data set: We have not just calculated summaries or plotted data, but written "programs" for calculating mean, median and producing a specific visualisation. Another example would be to construct a complex statistical graph for a given data set where—hidden from the user—a code is generated that will produce a similar graph if the original data set is substituted.

Among the existing systems with graphical user interface, the software *Data Desk* offers quite interesting possibilities for defining educational microworlds, besides being a tool for data analysis. It supports the definition of nested "derived variables", and the linking of representations (data, graphs, results, equations of derived variables). A concrete analysis path that has resulted in a system of linked objects or windows can actually be used as a kind of macro or template: there is an easy possibility of changing data and substituting variables while maintaining all the defined relations. We consider this feature of *Data Desk* a good model for how the "programming by example" concept could be realised.

The screen display in Figure 2 offers an example of a microworld for a set of variables referring to a data set on cars. We have drawn a regression line of *mpg* (miles per gallon) vs. *weight* of the cars (window 4). As the graph has curvature, we would like to determine experimentally a transformation of *mpg* from the system of transformations $x \to x^m$ for real parameters $m \neq 0$. We define a "slider" with parameter m (window 1) that can be continuously varied by moving the vertical line on the screen. We define a "derived variable" $f(mpg) = mpg^m$ (window 2). Window 3 shows a regression line of $f(mpg)$ vs. *weight*, windows 5 and 6 show two displays of residuals of that regression. If we change m in window 1, all the dependent windows will update. The dependence structure can be summarised as in the right part of Figure 2.

This microworld can be used for studying the effect of different data transformations. One could also use artificial data or a sample from a certain known model to study these relations. In practice, this microworld could be used for an interactive choice of a linearizing transformation—instead of using ready-made automatic methods for this purpose. As it is possible to save this environment and

to replace the above two variables by different ones, we have really defined a small new microworld for exploring the effects of data transformations. It would have also been possible to directly edit the formula in window 2. We note that formal-mathematical representations have re-entered at a reasonable location in a limited way. The concept of *slider* pushes the tool a step in the direction of a method construction tool where one can operate with general parameters. I have added the graph with the linkage structure to make the construction transparent. It may be considered a weakness of systems like *Data Desk* that the linkage structure is not explicitly documented as it is the case with explicit programming or if we had written the list of commands in an editor. An improvement would be if a list of commands or another representation of the linkage structure would be generated automatically.

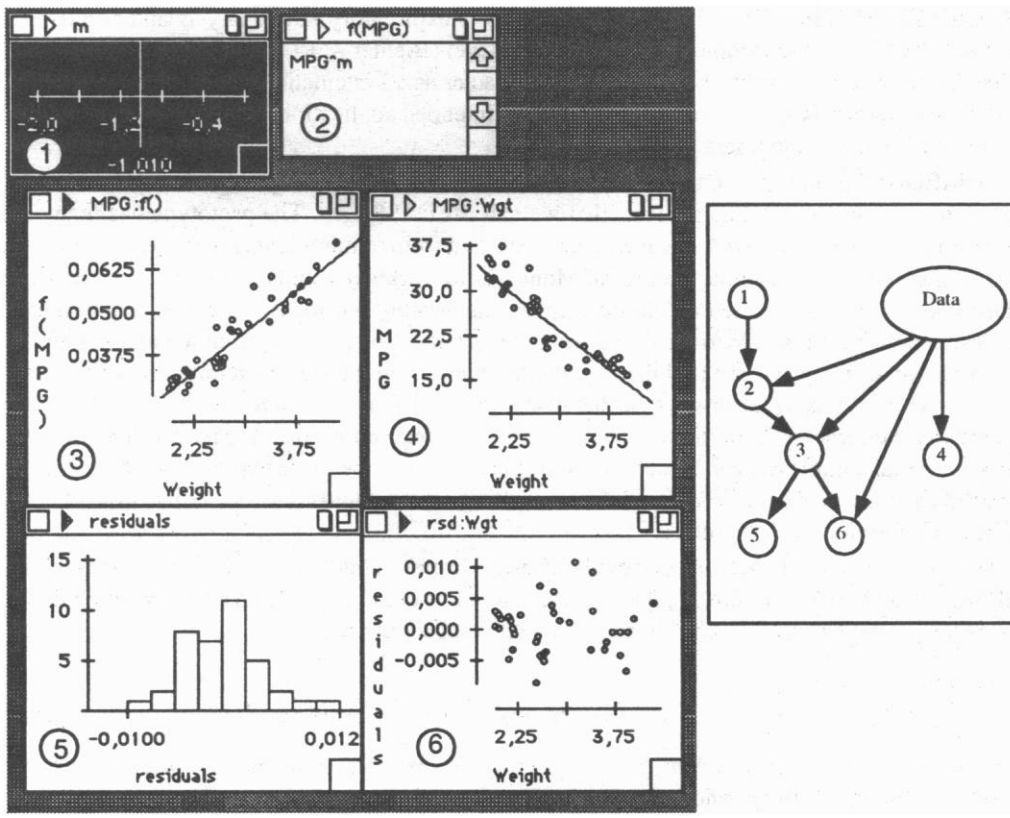

Figure 2. *An experimental environment in Data Desk 3.0; the linkage structure*

5 Requirements of Software Tools—Experiences from the MEDASS Project

After having unfolded the general scene in the preceding four parts, we will become much more specific now and discuss relevant features of our prototypical software tool in much detail and in a compact and structured manner. We will refer to other solutions in existing software if helpful. Readers who are concerned with designing, using and evaluating software may find the detailed requirements helpful for sharpening their critical awareness and for their practical work. Other readers may want to get an overall picture of the prototype and can selectively focus on some aspects

of the detailed exposition.

5.1 Background of the MEDASS Project

The discussion of requirements for software tools builds on the MEDASS project (Modelling in connection with Exploratory Data Analysis and Stochastic Simulation) that was motivated by the intention to improve statistics education at the secondary level by providing classroom material for computer-supported data analysis in several school subjects and also adequate software. Classroom materials for geography by Kohorst (1992), for social science and history by Portscheller (1992) and mathematics by Noll & Schmidt (1994) have been developed. Parallel and in co-ordination to this development work and related teaching experiments with existing software, we have elaborated a detailed specification of a tool intended to support exploratory data analysis and simulation at secondary level, and we assume, also beyond that level (Biehler & Rach, 1992). We call this result a "paper-and-pencil prototype" because of its character as a systematic model, not being just a list of partly contradictory general requirements. Also, intended applications and the background of that prototype have been elaborated (Biehler, 1992).

The MEDASS prototype is currently used as a basis for designing and programming an elementary student tool for statistical data analysis that we call *MEDASS light*. The prototype has many more advanced features than *MEDASS light* will have. Starting from requirements for an elementary student tool, we intended to incorporate features of Monte Carlo workbench and of a theoretical statistician's environment. We concentrated on elaborating and integrating our ideas of desired requirements in our design conception and learned a lot during the specification process about the amount and kind of complexity of the projected tool that we would obtain from the various requirements. Our model is intended to serve as a platform for further discussion and as a medium for specifying justifications for certain requirements. In particular, our assumptions concerning what may be intuitive for novices and what initial complexity can be demanded from novices will be made explicit. We think that some of our ideas are valuable as framework for evaluating existing tools and as elements of other tools that may be developed in the future.

In the following, I will describe and explain basic features of our "prototype", sometimes referring to different and better solutions in other software. Readers could always translate a wording like "the MEDASS prototype has . . ." into "a more adequate tool should have . . .".

5.2 System Architecture of the MEDASS Prototype: Result Objects and Their Linkage Structure

A fundamental design problem is how to balance flexibility and the resulting complexity of options against ease-of-use and ease-of-learning. We will describe some of our solutions below. Every basic system should at least provide a **multiple window environment**, where several graphs, data tables and results can selectively and simultaneously be displayed for purposes of comparison. Automatical resizing of plots in windows is important as well as possibilities for temporary closing of windows (iconizing). These features have particular advantages for supporting an incremental, multiple analyses working style and for pedagogical uses where comparisons of various results are important.

Workspace. In our prototype, an analysis is performed by selecting commands from the main menu that act on the selected variables of a data table according to their *roles* (see below). A result can be a *graph*, a *result table* or a *result form*. All these *result objects* will be saved and iconically represented in a so-called *Arbeitsmappe* (a metaphor for workspace) and displayed each in a window of its own on the screen. Providing such a workspace with additional commands for organising its elements (a small operating system) is helpful for dealing with multiple analyses' results. Some newer programs show how this feature can be extended: Result objects can be embedded in a kind

of notebook, where one can add comments to the results. Hot links between results and data make it easy to resume an analysis. Without workspace and notebook, users have to copy results to word processors to collect multiple results and write up reports.

Main system and satellite tools for more complex or specific problems. Figure 3 shows components of the MEDASS system. The various window types (objects) are displayed as well as the structure of the commands from the user perspective: a main menu, tool boxes associated with each window type and some "satellite tools", i.e. tools for defining and editing formulas, probability models, and subsets. The idea is to support the input of simple definitions by the menus and by direct manipulation as a first simple step for the novice. More complex ones that require some formal language input, and complex editing will be dealt with by the tools: an option for the developing user. The *Journal recorder/player* could be used to redisplay and analyse actions of the user: a valuable feature from a pedagogical point of view, that we however did not elaborate in our prototype.

Linking for experimenting. Result objects have hot links to the data table that support an update or redo possibility for the linked result when the data have changed. The linkage with the data table supports an easy exploration of how changes in the data, method or presentation (view) affect the results. Results can be conceptualised as

result = *f (data, method, parameters, view)* or

result = *f (chance, model, parameters, view)*. Results may have "hot" (dynamic) links with the "arguments" of the function *f*.

The variables and their roles belong to the data structure of the result objects. They can be displayed and changed in an associated information window. This feature supports a full documentation about the variables and their roles that went into an analysis. Moreover, variables and roles can be easily changed and substituted while keeping the statistical method "constant". In other words, result objects are the core of templates. The user has done some "programming by example", (s)he has constructed a macro or template whose components are contained in the Arbeitsmappe.

Linking extends the pedagogical and data analytical capability of a software by an order of magnitude, and this feature is increasingly offered by statistical tools. The linking with chance models however is offered much more rarely. However, there are some recent professional programs that implement linking in a way that the user cannot easily develop a cognitive model about what kind of linking is implemented and thus (s)he is continually surprised what happens on the screen. Such programs are hardly adequate for introductory statistics.

Numerical results are new data. Whereas the result form is conceived as more or less just showing numerical results (with commenting text that could be changed by the teacher), a result table is a table with numerical data. Variables of the result table can be selected for further analysis and display. This should be a basic feature of any software because complex statistical methods can now be composed from more elementary building blocks. Many current professional systems offer this possibility not generally but only for a very limited number of statistical methods. If we add a hot linking of tables to this feature, the user may produce a system of logically linked tables (and graphs). This extends the programming by example feature to a multi-step program. Spreadsheets are prototypes of infinitely many possibilities of linking cells in numerical tables. The thinking in terms of linked statistical "result objects" is more adapted to statistical thinking.

Graphs as an interface to data. A result graph in the MEDASS prototype does not only reflect all changes in the data table, but can be also used in the other direction, if individual data points are represented in the graph: subgroups can be selected and this selection is represented in all other linked representations, and can be saved as a categorical variable in the data table. We speak of using graphs as an interface to data. We assume that this feature simplifies data base activities by novices as well as supporting new styles of elementary multivariate data analyses with linked representations.

Configurable menus and tools. A conceptually simple answer to the needs of adaptability that we favour in the MEDASS system is that menus are configurable. This feature is essential. It is

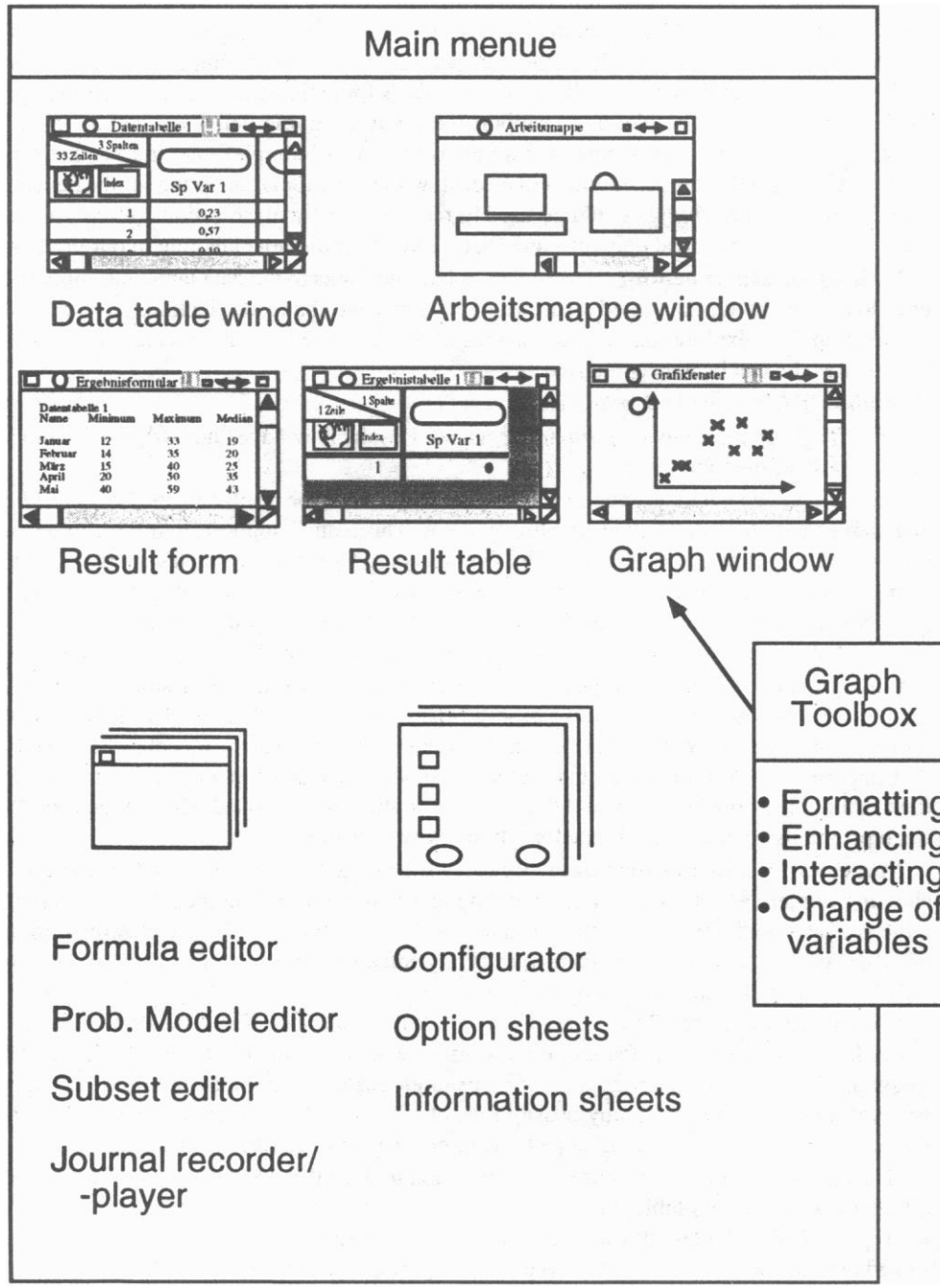

Figure 3. *Components of the MEDASS system*

becoming standard in many word processors and other tools but has not yet reached the realm of professional statistical software. Without this feature, the fully developed MEDASS tool could add to the complexity of tool problem that we discussed earlier. The design of menu structure has to bear in mind that reasonable reductions and reorganisations are possible.

5.3 Data Representation and Variable Selection: Roles and Visual Data Base

The basic object of the MEDASS software is a visually represented case-oriented *data table*, where columns represent variables (see Figure 4). We agree with other software that the table is a fundamental and intuitive data representation for novices. We are aware of its limitations, but we assume that a cleverly contrived system of possible "variable roles" mitigates this problem. Besides the data values that can be text or numbers, the variables have various *attributes* that are hidden in the table representation but can be accessed and changed through the information button. Attributes include variable type where we think that providing the types *quantitative, categorical,* and *name* is a minimum requirement.

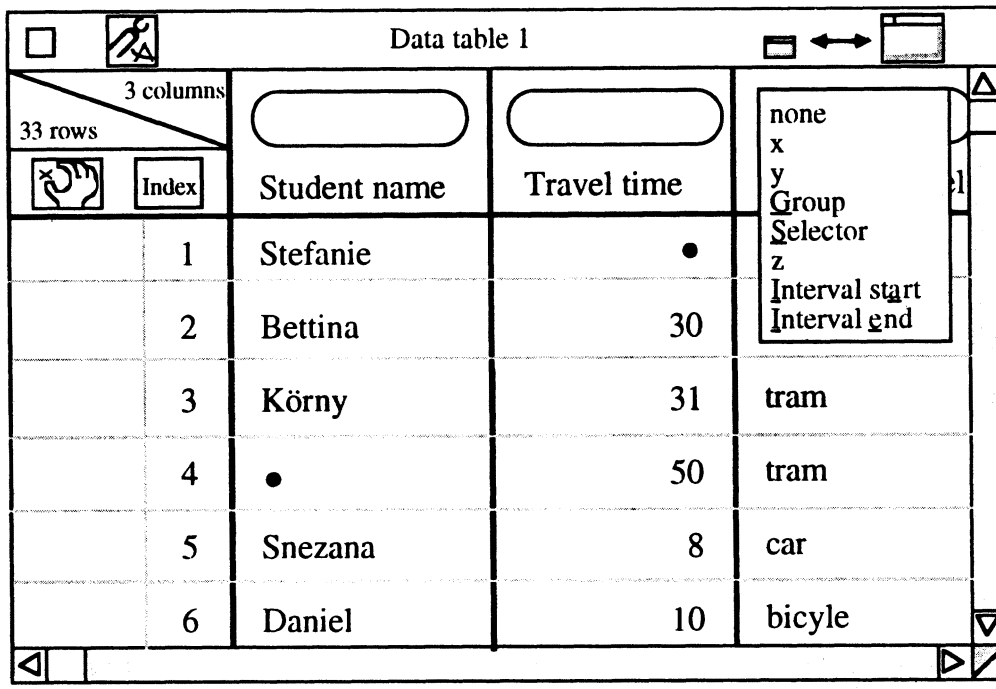

Figure 4. *MEDASS data table with open button for selecting the role of a variable*

Variable based statistical spreadsheet. With a second attribute, we distinguish *standard variables* from *derived variables*, which are defined by formula. This is fairly common. The possibility of using chains of derived variables provides some algebraic and numerical capacity for user-defined methods and derived probability distributions. This concept of a *statistical* spreadsheet where the set of variables with data can be extended by further variables, which may be dependent on the data or on a probability distribution or formula, seems to be intuitive for novices. Its handling may become a problem with large data sets and many derived variables.

Derived numerical results and variables can also be put into other linked result tables to overcome the limits of only one data table. Conceptions like *Data Desk's* systems (relations) of variables and "sliders" (see part 4) provide more flexibility. As they are supported by intuitive visual representations and metaphors, they also have advantages for novices. We have assumed that such further extensions of our data structures would be too complex for novices.

Random variables. We introduce a third type of variable: *random variables* are columns of a data table that are defined by probability distributions from which the column is drawn as a sample. Updating such variables is equivalent to a new random trial. The random variables are the elementary building blocks of random experiments in our system. The model editor tool can be used to define models for single and multistage random experiments using elementary building blocks. Standard discrete and continuous distributions, arbitrary user defined distributions for modelling and pedagogical purposes and sampling from data columns to support bootstrapping are available. The fact that the definition of a random variable can be hidden from the user can be exploited to have the user do a statistical analysis on simulated data and compare the results to the previously hidden "true" model.

Visual data base. A major feature for simplifying data base selections are visual representations of cases which permit graphical selection. We provide data base functions by means of selecting rows "by hand" in the data table and we use *case preserving* statistical graphs dynamically linked to the data, such as scatterplots. The *subset editor tool* can be used for defining logical conditions for selecting rows. These features are part of the software *MEDASS light*. A further step ahead would be the possibility to save and edit (conditions for) subsets. Comparison of methods on various subsets has pedagogical and data analytical uses. The representation of cases by rows in a data table, however, may already be too abstract for younger students. Because of this reason, cases are additionally represented by icons in the educational software tool *Table Top*. The icons can be grouped, reordered and so on.

Generally, a record-based structure from which several table "views" (with different possible definitions of "cases") can be created is more adequate for many complex data. The planned educational software tool by Finzer (1993) aims at providing this more complex structure in a way that is more intuitive for novices.

Role concept. In the MEDASS data table, the user can select variables for analysis by assigning roles to them, as in several other programs. The assumption is that the role concept is intuitive for novices and helps very much to reduce complexity. For instance, most commands are defined for various role selections. A standard scatterplot will be produced if one x-role and one y-role is selected. Several selected x- or y-roles produce a juxtaposed or superimposed scatterplot. If we further select a variable with role g (grouping variable) then the categories of this variable are shown in the scatterplot by different symbols or colours. The user selects roles for certain variables and then can try out several methods of analysis or display on this selected set. By this feature, the role concept supports pedagogical and data analytical experimentation and ease of variable selection. A very differentiated and complex role concept, however, can make it difficult for users to construct a cognitive model of the system behaviour. A system that requires the method selection first and then prompts for variables may be safer for users that tend to select variable roles without much thinking.

Analysis by group as a general feature. The selection of a variable with g-role (grouping variable) produces a multiple analysis by group. The fact that all analyses can easily be done "by group" or without grouping reflects the assumption that "analysis by group" should be a fundamental operation of data analysis from the beginning. Analysis by group ranges from just calculating sums to producing multiple graphs. Many existing tools offer grouping only in connection with some methods but not as a general option.

Dummy roles for additional flexibility. The MEDASS prototype provides the possibility of assigning further roles like z, ia (interval start) and ie (interval end). This adds flexibility and helps

to overcome the limitations of case-oriented data tables. For instance the case oriented data table will be interpreted as a contingency table by adequately assigning these roles (z can be interpreted as declaring a column as a frequency column).

Excluding cases and subsets. Cases can be excluded from an analysis by simply clicking in the first column of the respective row in the data table or using the subset editor tool. We can also select a variable with a *s*-role (selective variable). Only those cases in which the *s*-role variable is different from zero are included in a subsequent analysis. Thus defined filters can be saved as derived or standard variables with values 0 or 1 in the data table itself. This integration of filtering into the role concept is practical, systematic and elegant, but requires that novices consider subsets as defined by indicator variables. An alternative for novices could be that several subset definitions or filters could be "attached" to a data table somehow with the possibility of further editing (like in *StatView* 4.0).

We will use one short example to illustrate the use of the role concept. Let a raw data table contain for each day of a year (i.e. 365 rows) the average daily temperature in three cities. A fourth column contains the month to which the day belongs. Let us look at the following derived table that contains median temperatures grouped by month.

Month	*city 1*	*city 2*	*city 3*
January	1,3	2,5	-0,5
February	3,7	2,8	1,5
....			

Figure 5. *Median monthly temperatures in 3 cities in degree Celsius*

How can such a table be produced in the MEDASS system? We have to think that we are interested in the variables *city 1, city 2, city 3* grouped by *month*. We select the first three as x_1, x_2, and x_3 variables and *month* as a grouping *g*-Variable. Then we decide which method to use, for instance numerical summaries of the distribution, box plots, histograms. For generating the above table, the command *median* from the menu *numerical summaries* was selected. In a next step, the data analysis tool can be applied to this result table, for instance for constructing a visualisation of the whole table or of parts of it.

5.4 Structure of Statistical Methods: Generic Commands and Option Sheets

A fundamental design question is how to cope with the complexities of statistical methods. We assume that the following features support a fluent working style and hide complexities from the novice.

Draft results, option sheets, coherent tool boxes. The selection of a command from the main menu immediately produces a "draft result", which can be afterwards improved and changed by means of a structured *tool box* that is "at hand" in any result window. The initial defaults are chosen by the system designers. However, an *option window* can be opened for every method, where default method parameters and view parameters can be changed. What is different in MEDASS from these fairly common features in recent programs for exploratory analyses? The option window can also be used to vary method parameters, while keeping data and view constant: a good pedagogical facility for exploring effects of method variation on results. Moreover, quick draft results have pedagogical disadvantages if students should learn to consciously choose options before applying a method. Therefore, we can select parameters in the option window and assign them the status "to be prompted".

Generic command names. A further way of reducing complexity is summarising slightly different methods under a common command name, i.e. using generic methods that accept and automatically

adapt to different numbers, roles and types of variables.

For instance, our scatterplot command would produce slightly different plots if both variables are quantitative or if one of them is categorical. Figure 6 shows examples for the various combinations categorical/quantitative (1), categorical/categorical (2), categorical/name (3), and quantitative/name (4). In (4) the (name) axis tool has been used for sorting the y-axis according to the variable travel *time*. We assume that stressing this generalisation of the Cartesian co-ordinate system beyond numerical variables has pedagogical advantages. It underlines a basic display principle for two variables.

Categorical and other axes. Categorical variables are graphically represented by what we call *categorical axes*. The tool box has a tool for changing scales. If it acts on a categorical axis, other options become available than in the case of a quantitative variable: categories can be rearranged "by hand" or according to criteria referring to other variables. With the opportunity of rearranging and permuting the order of the 2 categories on each axis "by hand", we might discover new relationships and possible classifications. In fact this would be a way of implementing Bertin's (1977) graphical method. We consider these manipulations of categorical axes (and similar of *name axes*) an elementary but very powerful operation. These features are seldomly implemented. Although spreadsheets have implemented some kind of categorical or name axis types, easy manipulation in our sense is not possible.

Multiple graphs. Only one command is used to produce multiple graphs by selecting several variables with x or y roles. Our conception of multiple graphs is threefold: the single graphs can be *juxtaposed* (all with separate but co-ordinated axes and scales, perhaps in different windows), *superimposed* (with shared axes) or *integrated* in a new display format (not always possible). Examples for integrated displays are back to back bar charts and multiple box plots. Integrated displays can be very poor, for instance, multiple barcharts are worse for comparison of grouped frequency distributions than the juxtaposed multiple graphs. Nevertheless, it is easy to switch between these three formats. This easy switching is not only of practical advantage, but we also assume that it helps novices with understanding the decomposition and recomposition of multiple graphs.

Modern multiple window systems support the easy juxtaposing of graphs, but the scales and sizes of the plots are often local to the individual window. Therefore it can become extremely time consuming (especially in spreadsheets) to resize windows and plot scales so that they are well prepared for comparison. *Data Desk* 5.0 offers an easy to use direct manipulation solution for this problem. An alternative is offered by the carefully designed Trellis displays that are implemented in *Splus* 3.3.

Raw and aggregated data. Another application of generic command names is the analysis of raw and aggregated data. A command *histogram* can be applied to various role selections: a variable with raw data will be assigned an x-role to; if frequency data are provided by three columns (interval start, interval end, frequency) we can assign the roles ia, ie, x to this triple. In another case, we would assign x-role to a column of values and z-role to a second column with frequencies. The tool box for histograms (for instance, modifying number of classes, adding numerical summaries) is always the same and the user will not be concerned with how differently the operations are performed internally.

5.5 Working with Graphs: Formatting, Interacting, and Enhancing

A better conceptual structure for graph modification options. The manipulation of graphs is particularly complex, and I will briefly describe our solution for the graph manipulation tool box (Figure 7). We assume that it is useful for the novice to distinguish three general types of manipulation: *formatting* or changing the view of data, *interacting* with the graph, and *enhancing* a graph by further information.

Generally, we limited the options to those that we considered most relevant for data *analysis*. Polishing a display for publication should be done with other programs.

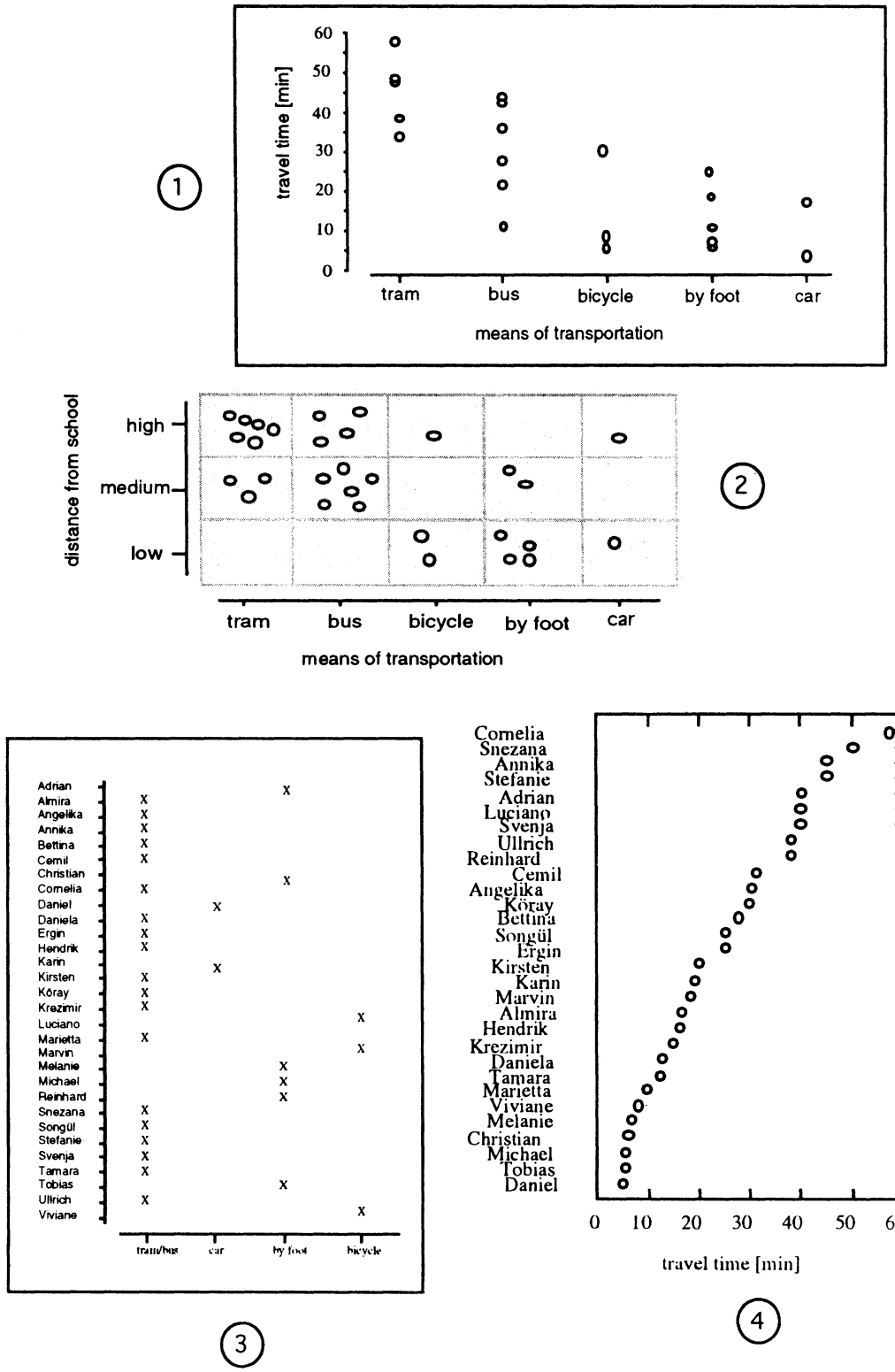

Figure 6. *Scatterplot as a generic method for various variable types*

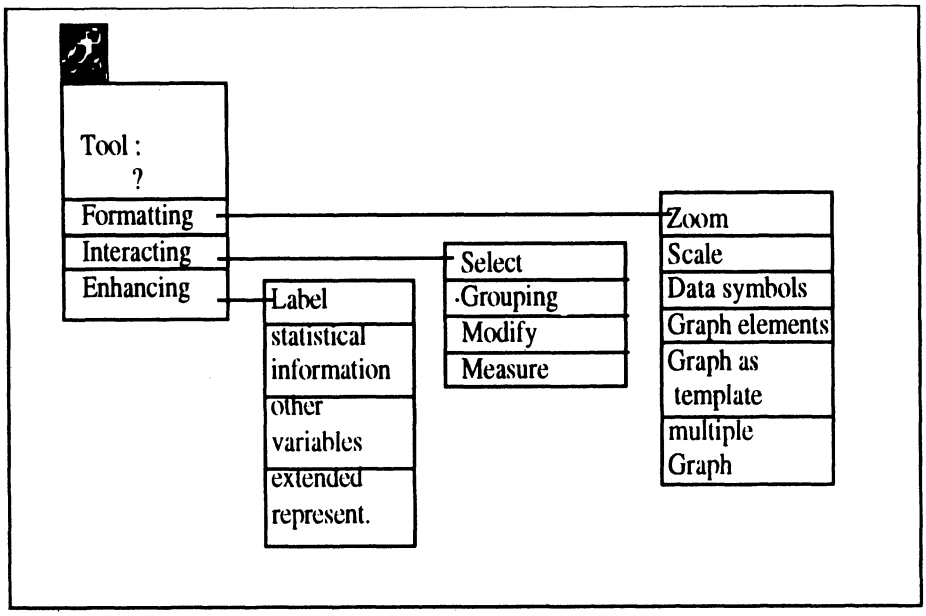

Figure 7. *Graph manipulation tool box of the MEDASS software*

The layer concept for multiple graphs. A complication arises in multiple graphs. We conceptualise a multiple graph as consisting of several *layers*, each of them corresponding to the respective subset of variables. The layer structure can be displayed and the active layer can be changed in order to make clear the layer the user wants to interact with.

Formatting includes scale changes and zooming. Formatting a scatterplot, however, would also include joining points by lines (line chart) or joining points with the x- or y-axis by lines or bars (horizontal or vertical bar charts). From this perspective, the dot chart (Cleveland, 1985) is just an adequately formatted scatterplot for a name variable (y-role) and a numerical variable (x-role). This is another aspect of our scatterplot generalisation.

Interacting summarises tools that realise the idea of using graphs as an interface to data. We add a *grouping tool* for performing a graphical classification to the standard *select tool*. A *measure tool* provides numerical values corresponding to features of a graph (raw data co-ordinates, displayed summary values). These features are more and more available in new professional programs for Windows or MacOS.

The *modify tool* has pedagogical uses for exploring methods. It allows a pointwise graphical input of new data and the modification (moving) of existing data. This feature would support the construction of many microworlds with easy graphical manipulation of parameters. Usual statistics tool do not support such an operation, it is however included, for instance, in *MS Excel 5.0*. Graphical data input to support the construction of adequate artificial data sets for illustrating a certain pedagogical issue is included in the pedagogical software tool *StatLab* (1987).

Enhancing a graph is essential for constructing visualisations and also for constructing derived graphs from basic forms. It becomes more common that existing programs offer the enhancement of graphs by lines, points and other elements. This is completely insufficient if these geometrical enhancements have no "statistical meaning", i.e. they cannot be used in further calculations and analyses.

We were inspired by the options that *Splus* provides for this feature. We think, however, that direct interaction or choice from menus is preferable to defining *derived graphs* by adding commands to a

script. For instance, a control chart can be constructed as a derived chart from a scatterplot by adding 3 horizontal lines for the mean and the mean±2*standard deviation. The concept of derived graph extends the potential repertoire of graphs while keeping the elementary types limited and surveyable for the novice. Enhancing includes displaying label information, of adding statistical summaries of various kinds (regression line, eye-fitted line, points or vertical lines for medians, means, measures of spread). A useful feature is that derived graphs can be saved as templates (or added as a command to the graph menu).

We will explain the enhancing of scatterplots in more detail. A tool box (see Figure 8) offers three levels of interaction

(1) options with assumed frequent use are directly accessible in the toolbox: eye-fitted lines by parameters m and b, or by graphical input. Moreover, stepwise linear functions can be drawn by graphical input.

(2) The option *add to graph* gives access (new window) to predefined further enhancements of the graph, for instance least squares regression lines and other curves and good scatterplot smoothers.

(3) The option *new* opens a pipe to the formula editor, which can be used to statistically define new graphical elements that can be included in the predefined list referred to in (2).

These features illustrate our concept of configurable and extensible tools. Furthermore, our priority choice for methods is different from standard statistical programs. From a pedagogical standpoint, it is necessary that students can experiment with their own lines before working with least squares lines or experimenting with eye-fitted stepwise linear functions or with average traces for a fixed slicing of the x-variable as primitive smoothers before turning to more sophisticated smoothers or functions. However, residuals should be available for these simple models, not only for regression lines and complicated smoothers. A user can change our priorities; for instance by putting the least squares line on the top level of the tool box.

6 Embedding Microworlds and Defining Random Experiments

We have described several important features of our prototypical software system and will now exemplify how we could embed microworlds and define random experiments and simulations in such a system. The software dilemma is especially grave with regard to elementary tools for simulation and modelling (see Biehler, 1991). Apart from statistical programming environments, statistical systems for data analysis are weak in this respect. *Models* or the respective *probability distributions* are not separate objects of the systems. It is difficult to generate the requested random data, especially with simple multistage experiments, and data analysis tools are not well adapted to thinking in terms of *events* and *random variables* whose frequency distribution has to be recorded during simulation.

We will show how our system or a similar one will support the definition of relevant microworlds and random experiments by means of some examples.

6.1 Experiments for Distribution Summaries and Fitted Lines

We will take two examples from the educational software *Statistics Workshop*. A subprogram called *Stretchy histograms* is designed to foster the understanding of the relation between summary values and shape of distributions. It shows a histogram of a distribution with markers at the distribution's mean, median, and quartiles. The user is able to manipulate the shape of the histogram by directly

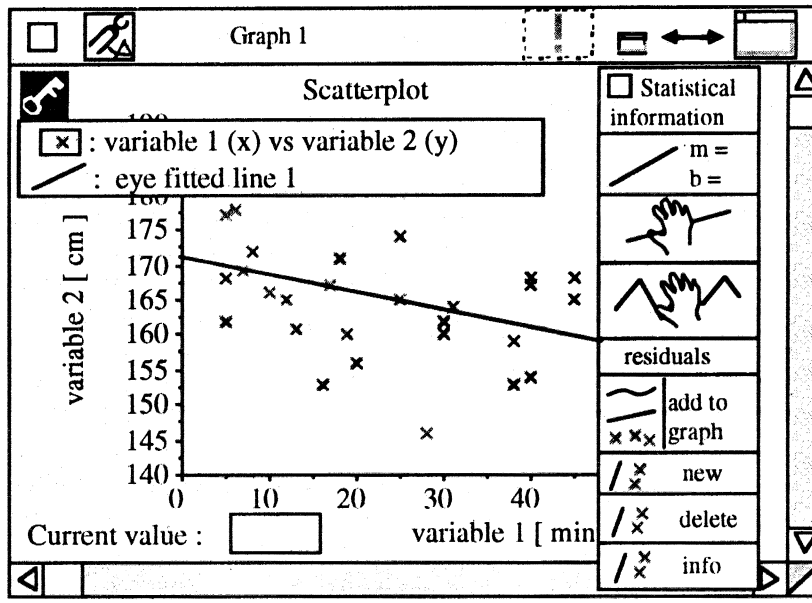

Figure 8. *Screen display from MEDASS : Enhancing scatterplots*

changing the height of the bars in the graph. The summaries are functionally dependent on the distribution and are automatically updated.

In the MEDASS prototype, the experiment *Stretchy histogram* would be authorable as follows: We can select *any* one-dimensional display of a quantitative variable (histogram, dot plot, stem-and-leaf), add the summary values to the display by the enhancing tool, and then change the *data* values (not the shape) with the modifying tool. The dynamic linking feature will show changes in the summary values or even in a linked box plot.

Another subprogram is called *Shifty Lines*: the user can change a line in a scatterplot by pushing it with a mouse and observe the numerical effect on the sum of squares of the residuals. This experiment is designed to support learning the method of least squares. In MEDASS, a similar experiments would be possible. The users, however, can define their own graphical representation or numerical summaries they want to observe or optimise. For instance, we could be interested in the structure of residuals as was the case in the example in part 4 of this paper.

6.2 *Visualising Random Confidence Intervals*

How can we construct the standard visualisation of confidence intervals as random intervals where the confidence level can be interpreted as the frequency of correctly captured true probabilities?

We can define an experiment in the following steps

(1) Choose a probability *distribution* with expected value μ. (model editor)
(2) Draw m samples of size n. (simulation control)
(3) Calculate the m means. (data analysis tool, results are data)
(4) Choose the level $(1 - \alpha)$.
(5) Calculate the confidence interval bounds according to a method K. (results are data)
(6) Plot the m means vs. $1, \ldots m$ in a scatter plot.
(7) Add bars to the m points in the scatterplot that correspond to the confidence intervals.
(8) Add a dotted line at $y = \mu$ to the scatterplot (enhancement of graphs).

Figure 9 displays the structure of the experiment. The steps are executed by selecting commands from the main menu or from the tool boxes. They result in new objects (data tables or plots) which are dependent of the previous objects and which can be represented in separate windows. In step (7) and (8), a derived graph is constructed. Now, the user might like to control and change the parameters *distribution*, $\mu, m, n, (1-\alpha), K$. It is possible to use another random seed, that is, repeating the whole random experiment in a simple way, by pressing a "new seed button". The final plot with the series of confidence intervals will depend dynamically on these parameters. The basic relation is *result = f (chance, model, method).* The arguments of the function f can be varied for experimental purposes. In the MEDASS prototype, the model editor and the simulation control tool are integrated in the data analysis tool and the model is attached to a column of the data table. This was a pragmatic decision, it may be conceptually simpler for novices to think in terms of different co-operating tools.

Our particular experiment with confidence intervals can be fairly easily programmed in standard spreadsheets with all the above linkages. However, a macro for calculating confidence limits has to be added, and typically the handling of the $n \times m$ data matrix with arbitrary n and m will cause some difficulties. With standard statistical systems, the random numbers can be generated, but it is often not possible to store the results of the multiple application of the confidence method in a table, because the numerical results of that procedure are not available to the user, but only for the system to produce an output display for each individual interval.

6.3 Studying Statistical Methods Under Certain Probability Model Assumptions

An important pedagogical computer use is observing the hypothetical random variation of a statistical function under repeated sampling. For instance, we intend to determine the t-statistics for several samples from a known probability distribution and analyse how the t-statistics is distributed. Similar to the preceding example, we have to apply the t-test to each of the m rows which represent a sample of a distribution and put the resulting t-value in a new column. This column will represent a sample of the t-statistics' distribution under the chosen probability distribution!

In more general terms, we have to be aware that rows represent (independent) repetitions. Columns represent the various random variables that may be involved in each trial. Note that the confidence intervals and the means have to be taken rowwise for producing the confidence interval table. In most programs, statistical methods can be applied only to columns. For the above approach, it is necessary to apply a statistical method also rowwise with a resulting new column with the results calculated rowwise. All numerical statistical functions have therefore to be available from the formula editor, which is uncommon in most available tools.

6.4 Events in Multistage Experiments and Simulation in Elementary Probability

Columns in a data table can be functions from each other. This is an easy way to construct multistage dependent experiments in a similar manner as in the above examples. Moreover, the formula editor has to be rich enough to define complex events on the basis of elementary ones. Let us discuss the example of tossing 4 coins (sample space {0, 1}). Four columns (length m) of the data table are used to represent the results of the 4 trials that are repeated m times. It is easy to define $X_5 := sum(X_1, X_2, X_3, X_4)$. This new column represents a random variable with binomial distribution.

However, how can we determine the frequency of the complex event 0110? We have to represent this event by an indicator variable (characteristic variable). We define a sixth (derived) column X_6 in the table that is dependent on the columns X_1, X_2, X_3, X_4:

$$X_6 := \left\{ \begin{array}{llll} 1 & \text{if } X_1 = 0 \quad \text{and } X_2 = 1 & \text{and } X_3 = 1 & \text{and } X_4 = 0. \\ 0 & & \text{else} & \end{array} \right\}$$

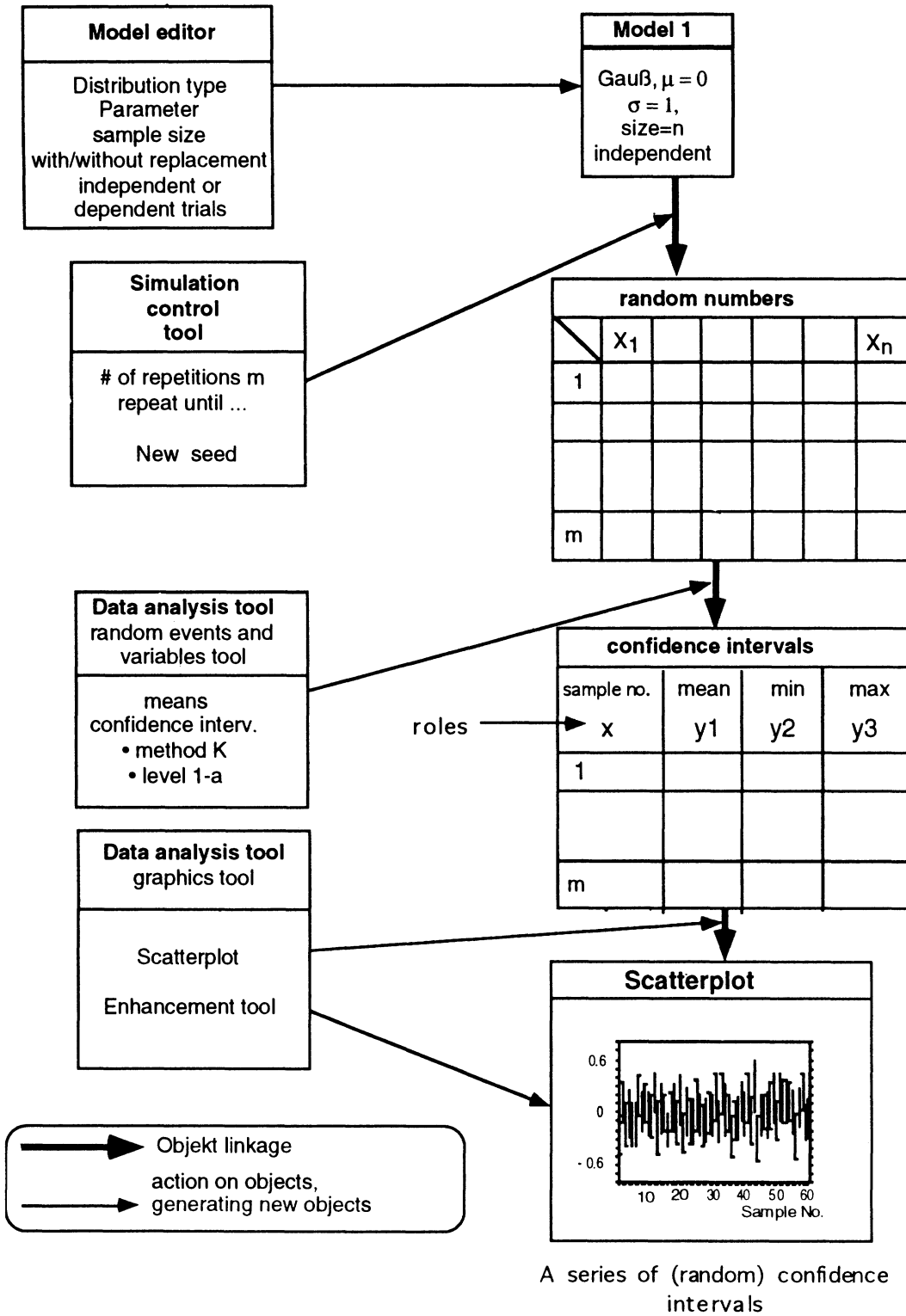

Figure 9. *Visualising the frequent conception of confidence interval*

The frequency distribution of the binary variable X_6 provides the frequency of "0110". An alternative solution would be to concatenate the 4 variables X_1, X_2, X_3, X_4 and "0110" would be one of the values of our new variable. Such transformations are available only in some professional programs but tools for easy working with such events are not provided. Anyway, novices may be entitled to an easier solution, such as the way that is used for some situations by the educational software tool *Probability Simulator*. This tool was designed for many relevant applications in elementary probability education (see Konold, 1991). Disparate development projects and constrained resources for educational software development have not yet led to satisfying overall solutions. There are some new projects for developing educational tools for simulation and data analysis pointing into a promising direction such as the software *DataSpace* (Finzer, 1993; Finzer, Jackiw & Erickson, 1993). Another domain which should be further explored with regard to simulation and modelling are the specific simulation tools either for dynamical systems, which often contain possibilities for stochastic elements (such as *Stella*), or the tools for simulating (discrete) stochastic processes for instance in queueing theory. An analysis of these programs from our point of view was not part of our project, but it would be important to look closer at these developments.

7 Concluding Remarks

Our prototype was presented to stimulate discussions on useful features of statistical software from a pedagogical point of view. The features of a software tool that we have suggested are in some sense informed hypotheses. They are rooted in discussions on design of statistical software, and some evidence concerning software use in introductory statistics education. Various experiences from our teaching experiments and discussion with teachers who have used other statistics software in their classroom have influenced the design. A next step would be to conduct more systematic empirical research studies in laboratory and classroom settings, either with existing systems that have similar features to our prototype, or to integrate such research studies into the overall process of programming and testing parts of our prototype and the *MEDASS light* software. Such research could confirm or put into question the assumptions underlying our design. Some preliminary results of such studies are published elsewhere (Biehler, 1996).

Acknowledgements

The MEDASS project was financially supported by the German Federal and State Ministers of Education as part of a larger project (Modellversuch im Bildungswesen) of the *FWU Institut für Film und Bild in Wissenschaft und Unterricht* in München. I am very indebted to Wolfram Rach, with whom I jointly developed the MEDASS prototype and with whom I had many discussions on the topic discussed in this paper. I am grateful to an anonymous referee and to Vijay Nair as the responsible editor for their careful comments on my paper, which were very helpful for its revision.

Software

BMDP New System 1.0. BMDP statistical software, 1440 Sepulveda Blvd, Suite 316, Los Angeles, CA 90025, (MS-Windows).

Data Desk 5.0. (1996). Velleman, P., Data Description Inc., P.O.Box 4555, Ithaca, NY 14852 (Macintosh, version 6.0 is available for Windows, too).

DataScope. (1994). Version 1.4. Konold, C. & Miller, C.D., Scientific Reasoning Research Institute, Hasbrouck Lab., Univ. of Massachussetts, Amherst, MA 01003, published by Intellimation, P.O. Box 1922 Santa Barbara, CA 93116-1922 (Macintosh).

DataSpace. (1997). Developmental Release 1. Key Curriculum Press.

MEDASS light. (1996). Biehler, R. & Rach, W. (design); Bauer, S. (programming). Software in development. Institut für Didaktik der Mathematik, Bielefeld und RWTH Aachen (Lehrstuhl Angew. Mathem. und Informatik). (MS Windows).

Minitab. MINITAB Inc., 3081 Enterprise Drive, State College, PA, 16801 USA (MS-DOS, Macintosh and others).

ProbSim - A Probability Simulator. (1994). Version 1.4, Konold, C. & Miller, C.D., Scientific Reasoning Research Institute, Hasbrouck Lab., Univ. of Massachussetts, Amherst, MA 01003, published by Intellimation, P.O. Box 1922 Santa Barbara, CA 93116-1922 (Macintosh).

S and Splus 3.3 (1995). Becker, R.A., Chambers, J.M. & Wilks, A.R., see: Becker, Richard A., Chambers, John M., Wilks, Allan R., The New S Language, Wadsworth & Brooks, Pacific Grove. Splus published by StatSci, a division of MathSoft (Unix, Windows).

SPSS. SPSS Inc., 444 North Michigan Avenue, Chicago, IL 60611 (Macintosh, MS-DOS and various other systems).

Stella II. (1994). High Performance Systems Inc., 45 Lyme Road, Hanover NH 03755 (Macintosh, MS Windows).

Statistics Workshop (1991). Rubin, A. & Bruce, B. (BBN Inc., Cambridge, MA, USA). Wings for Learning/Sunburst, 1600 Green Hills Road P.O. Box 660002, Scotts Valley, CA 95067-0002 (Macintosh).

StatLab (1987). W. Douglas Stirling. The New Zealand Statistical Association, P.O. Box 1731, Wellington, New Zealand (Macintosh).

STATlab by slp (1993). slp Statistiques, 51/59 Rue Ledru-Rollin, F-94853 Ivry Cedex, France (MS-Windows Macintosh).

StatView SE+Graphics. (1988). Feldman, D.S., jr & Gagnon, J., Abacus Concepts, Inc., 1984 Bonita Ave., Berkeley, CA 94704, USA (Macintosh).

StatView 4.0 (1992). Feldman, D.S., jr & Gagnon, J., Abacus Concepts, Inc., 1984 Bonita Ave., Berkeley, CA 94704, USA (Macintosh, newer versions also available for Windows).

Systat 5.0 (1989). SYSTAT Inc., 1800 Sherman Avenue, Evanston, IL 60201-3793 (Macintosh, MS-Windows).

TableTop. Hancock, C.& Kaput, J.J., TERC, 2067 Massachusetts Avenue, Cambridge MA 02140, published by Brøderbund Software, (Macintosh, Windows).

References

Bertin, J. (1977). *La Graphique et le Traitement Graphique de l'Information [Engl. Transl. : Graphics and Graphic Information-Processing, Berlin: de Gruyter 1981]*. Paris: Flammarion.

Biehler, R. (1991). Computers in probability education. In R. Kapadia & M. Borovcnik (eds.), *Chance Encounters—Probability in Education*, pp. 169–211. Dordrecht: Kluwer.

Biehler, R. (1992). *Intendierte Anwendungen und didaktische Begründungen zu einem Softwarewerkzeug zur Explorativen Datenanalyse und Stochastischen Simulation für Schule und Ausbildung*. München: Inst. f. Film und Bild in Wissenschaft und Unterricht (FWU).

Biehler, R. (1993). Software tools and mathematics education: the case of statistics. In C. Keitel & K. Ruthven (eds.), *Learning From Computers: Mathematics Education and Technology*, pp. 68–100. Berlin: Springer.

Biehler, R. (1996). Students' difficulties in practising computer supported data analysis—Some hypothetical generalizations from results of two exploratory studies. In C. Batanero (ed.) *Research on the Role of Technology in Teaching and Learning Statistics. 1996 IASE Round Table Conference Papers [edited volume will be edited by Joan Garfield & Gail Burrill and published by ISI]* , pp. 163–182. Granada: University of Granada.

Biehler, R. & Rach, W. (1990a). Softwaretools for statistical data analysis in education and teacher training? An analysis of software conceptions from an educational perspective. In F. Faulbaum, et al. (ed.) *SOFTSTAT '89: Fortschritte der Statistiksoftware 2*, pp. 575–583. Stuttgart: G. Fischer.

Biehler, R. & Rach, W. (1990b). *Softwaretools zur Statistik und Datenanalyse: Beispiele, Anwendungen und Konzepte aus didaktischer Sicht*. Soest: Soester Verlagskontor.

Biehler, R. & Rach, W. (1992). *MEDASS: Explorative Datenanalyse und Stochastische Simulation—Anforderungsbeschreibung zu einem Softwarewerkzeug für Schule und Ausbildung*. Bielefeld: Universität Bielefeld, Institut für Didaktik der Mathematik.

Cleveland, W.S. (1985). *The Elements of Graphing Data*. Monterey: Wadsworth.

Finzer, W.F. (1993). *StochasticsLab—a Computer Learning environment for Data Analysis and Statistics in Secondary Schools. A Proposal*. Berkeley, CA: Key Curriculum Press.

Finzer, W.F., Jackiw, R.N. & Erickson, T.E. (1993). *Probability and statistics for Secondary Schools—A Computer Learning Environment and Curriculum. Final Report*. Berkeley, CA: Key Curriculum Press.

Gordon, F. & Gordon, S. (eds.). (1992). *Statistics for the Twenty-First Century. MAA Notes 26*. Washington: Mathematical Association of America.

Kaput, J.J. (1988). *Looking Back From the Future: A History of Computers in Mathematics Education*, 1978–1998. Unpublished Manuscript, Educational Technology Center, Cambridge (MA).

Kohorst, H. (1992). *Bevölkerungsexplosion—MEDASS Materialien*. München: FWU Institut für Film und Bild in Wissenschaft und Unterricht.

Konold, C. (1991). *ChancePlus: A Computer Based Currriculum for Probability and Statistics*. Second Year Report, Scientific Reasoning Research Institute, University of Massachusetts, Amherst.

Moore, D.S. (1994). The uses of video in teaching statistics. In L. Brunelli & G. Cicchitelli (eds.), *Proceedings of the First Scientific Meeting of the International Association for Statistical Education*, pp. 213–220. Perugia: Università di Perugia.

Moore, D.S. (1997). New pedagogy and new content: The case of statistics. *International Statistical Review*, **65**, 123–165.

Naeve, P., Trenkler, D. & Wolf, H.P. (1991). How to make the teaching of statistics roar—Some thoughts on computer based experiments. *Computational Statistics Quarterly*, **6**, 325–353.

Noll, G. & Schmidt, G. (1994). *Trends und statistische Zusammenhänge. Materialien zur Explorativen Datenanalyse und Statistik in der Schule*. Soest: Landesinstitut für Schule und Weiterbildung.

Portscheller, P. (1992). *Industrialisierung Deutschlands—MEDASS Materialien*. München: FWU Institut für Film und Bild in Wissenschaft und Unterricht.

Puranen, J. (1994). Teaching data analysis with the help of Survo. In L. Brunelli & G. Cicchitelli (eds.), *Proceedings of the First Scientific Meeting of the International Association for Statistical Education*, pp. 277–284. Perugia: Università di Perugia.

Snell, J.L. & Peterson, W.P. (1992). Does the computer help us understand statistics? In F. Gordon & S. Gordon (eds.), *Statistics for the Twenty-First Century. MAA Notes #26*, pp. 167–188. Washington, DC: Mathematical Association of America.

Steinecker, J. (1990). *Individualisierbarkeit von statistischer Software*. Münster: Lit.

Thisted, R.A. (1986). Computing environment for data analysis. *Statistical Science*, **1**, 259–275.

Thisted, R.A. & Vellemann, P.F. (1992). Computers and modern statistics. In D.C. Hoaglin & D.S. Moore (eds.), *Perspectives on Contemporary Statistics. MAA Notes 21*, pp. 41–54. Washington: Mathematical Association of America.

Witten, I.H., MacDonald, B.A., Maulsby, D.L. & Heise, R. (1992). Programming by example: The human face of AI. *Artificial Intelligence and Society*, **6**, 166–185.

Résumé

La communauté des statisticiens et des éducateurs en statistique est appelée à prendre responsabilité pour l'évaluation et l'amélioration de la qualité de la programmerie sous une perspective d'éducation. La contribution developpera une telle perspective, un système idéal de critères facilitant une évaluation criticale de la programmerie existante et une production future de programmerie plus apte à l'apprentissage et à la pratique de la statistique dans des cours d'introduction. Divers types d'outils et de micromondes sont necessaires. La discussion des necessités générales pour ce type de programmerie est suivie d'une présentation détaillée d'un système de programmerie de prototype idéal, démontrant comment un tel système pourra être utilisé pour construire des milieux d'apprentissage et pour soutenir une analyse de données élémentaire avec un style de travail exploratoire.

Mots clefs: Education en statistique; Conception d'une programmerie statistique; Evaluation des logiciels statistiques; Analyse des données exploratoire; Simulation.

[*Received August 1996, accepted February 1997*]