A Graphical User Interface for Spatial Data Analysis in Integrated Circuit Manufacturing

Lorraine Denby and David A. James
AT&T Bell Laboratories
Murray Hill, NJ 07974
ld@research.att.com
dj@research.att.com

Abstract: We describe a graphical user interface (GUI) to an environment for spatial data analysis. The purpose of developing this GUI was to help us transfer new statistical and computing techniques from a research organization to an integrated circuit (IC) manufacturing area. This paper begins by briefly describing a strategy for exploiting the spatial nature of IC data and how we incorporate these with other data sources (e.g., in-line measurements, design of experiments); we then illustrate how the GUI simplifies the implementation of this strategy. We conclude by identifying advantages and disadvantages of developing such an interface.

1 Background

The manufacturing of integrated circuits (IC's) is complex and costly, and it can be briefly described as a process carried out in a clean-room environment that starts with a flat wafer of highly pure silicon upon which multiple layers of electronics are etched. To build the various layers, an electrical insulating film of polysilicon is deposited across the wafer, a second film of photo-sensitive material is added, then a lithography process analogous to photography is used to implant minute structures onto the photo-resist; the photo-resist hardens with light and the unexposed areas are dissolved and etched to leave topological structures with electrical properties specific to that one layer. This process is repeated numerous times to build layer upon layer of structures that will make up the finished chip. Each wafer can hold from as few as 40 chips to as many as 800 or more, ranging in price from more than a thousand dollars to less than one dollar.

Wafers are processed in lots and can take from a few days up to a few weeks to finish processing. Once finished, wafers are typically sent to other facilities for packaging. Typical processing times are between 3 and 8 weeks. This relatively long processing time and high-cost of production create a need for quick problem identification and resolution. A problem that is not detected on time can easily lead to scrap worth hundreds of thousands of dollars or more. On the other hand, even modest yield improvements can translate into significantly improved revenues.

2 Data Sources

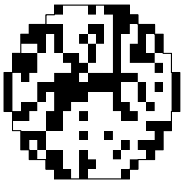
During manufacturing vast amounts of data are collected for process monitoring and improvement; in addition, engineers are continually developing new processes for new technologies and products. These data include "probe" or final test measurements, in-line information, parametric (electrical) testing, particle scans, data from designed experiments, zone-monitors, etc.

Probe data are collected on each chip of each wafer. They describe whether the chips are working or not; data of non-working chips include their failure type or "bin", and data for functional chips include performance characteristics.

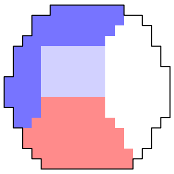
```
            D 1 1 1 C 1 D C D
          1 1 1 1 1 1 1 1 C D D
      B D P B 1 1 1 D J 1 C E 1 D D
    B I D 1 F 1 1 D 2 D 2 C 2 2 D 2 D
    2 2 D 2 I 2 1 2 D 2 2 C 2 D D 2 D
  F F D 2 2 2 D D 2 D 2 2 C 1 D D E 2 2
  D P C 2 2 2 1 F D 2 2 2 C 2 2 2 2 D 2
B 2 D 2 2 2 D D J 2 D D H C E E 2 2 1 2 D
F 2 2 2 D D D F D F G D D D 2 2 H 2 D 2 E
H 2 2 1 D 2 D D D F 2 2 2 D 2 2 F 2 2 2 D
D 2 E D 2 2 D D 2 2 2 2 2 H S 2 2 F 1 2 A
  D S 1 2 D F 2 2 D 2 2 2 2 D D 2 2 2 1
  2 2 2 1 D D 2 2 2 2 2 2 2 2 2 2 S H D
    G D D 2 2 2 2 D 2 2 H 2 1 2 2 2 D
    D 2 D 2 2 1 2 2 2 2 2 2 D 2 H K B
      D 2 I D 2 2 F 2 2 2 D 2 1 2 C
          I P 2 2 D S 2 2 2 2 B
            D B 2 S 2 2 1 2 C
```

Figure 1: Wafer Probe Map. Digits denote functional chips of varying performance and letters denote non-functional defect bins.

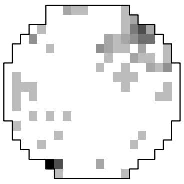
Probe data are always available, since they are imperative for packaging the final chips. From "binned" wafers we compute "binary" wafers using 0 to denote working chips and 1 to denote non-working chips. The following picture depicts the binary wafer corresponding to Figure 1; black squares denote defective chips and white squares functional chips.



In addition to probe measurements, over 60 electrical tests are conducted on random samples of wafers from each lot. Tests are conducted at 5, 10, or more sites on each sampled wafer and provide insights into processing problems as well as the overall electrical properties in specific areas of the wafer. The following figure shows the value at each of 5 (unequal) wafer regions of one such parametric test.



Particles (dust and processing materials) can damage a chip and thus are monitored both in the clean-room facility as well as on wafers. Particles may be detected on wafers by using special equipment that records location of particles on the wafer surface. It is thus possible to relate probabilities of chip malfunction to particle counts and location. The following particle map shows the square root of the number of particles on one wafer.

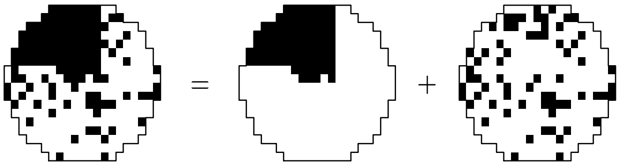


In-line data refers to information taken on the manufacturing environment, not the product itself. During critical steps, information such as cassette where each wafer is processed, wafer position on the cassette where it is processed, processing machine, chamber where the wafer is processed, etc., are recorded. Other sources of data include designed experiments, "zone" monitors (special non-production wafers that exercise only certain aspects or "zones" of the overall processing), etc. For a description of some of these other data and analysis see [Hansen and Nair, 1995].

3 A Strategy for IC Spatial Analysis

A strategy for exploiting the spatial nature of wafer data and for incorporating the above data sources has been developed [Friedman et al., 1995a]. This approach is based on the fact that many failure mechanisms manifest themselves as patterns of defects on the wafer surface, e.g., a radial pattern may be related to uneven coating, a horse-shoe pattern to diffusion problems, or a checker-board pattern to a "stepper" during lithography. Thus identifying and grouping wafers with similar patterns and then relating the patterned information (i.e., its shape, location, and size) to in-process and electrical test data provides an economical way for process improvement and problem diagnosis. Tools that we have developed provide the means to carry out this strategy.

The basic idea is to model observed defect data as the superposition of two classes of failure mechanisms: One that generates defects in patterns, and another that generates defects randomly on the wafers, i.e.,



We have techniques [Friedman et al., 1995b] for separating the patterned defects from the background defects based on a test for complete spatial randomness [Cliff and Ord, 1981]. Once patterned wafers have been identified, we can run a clustering algorithm on a smoothed and suitably transformed version of the binary wafers to group them according to patterns. Figure 2 shows an example consisting of each of six identified groups. (See Section 6 for details on how to identify these groups.)

We then relate the groups to all other available data. For instance Figure 3 shows a plot of wafer yield versus position on each cassette and the group membership encoded as the plotting symbol. The fact that wafers belonging to groups were processed in adjacent positions on the cassette in this (and only this) step provides strong evidence that the problem occurred during this processing step. Moreover, wafers whose patterns belong to groups 3 and 4 were only processed in cassette 2, while wafers with patterns in groups 5 and 6 were exclusively processed in cassette 1. This strong evidence, albeit circumstantial, was used by failure-mode analysis engineers to quickly narrow down their search for the cause of the problem.

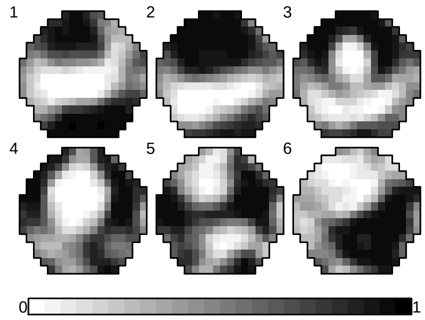


Figure 2: Average wafer maps for each of six groups of similarly patterned wafers.

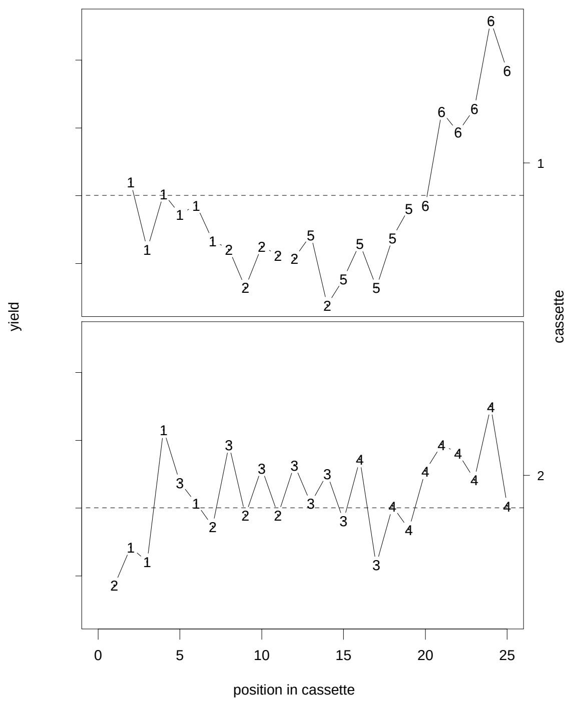


Figure 3: This plot shows wafer yield versus position on processing cassettes during one major manufacturing process. The plotting symbols encode the wafer's group membership, as shown in the previous figure.

3.1 Tools for Spatial Analysis

An interactive environment for spatial IC data analysis was written in the **S** language[Becker et al., 1988, Chambers and Hastie, 1992]. The environment exploits the object-oriented paradigm in **S** to seamlessly integrate IC spatial analysis into the existing general purpose data analysis environment. The environment provides

- visualization of spatial wafer data,
- assessment of complete spatial randomness,
- estimation of area defects and background defect intensity,
- classification of wafers according to patterns,
- correlation of groups of patterned wafers to other data,
- spatial analysis of data from designed experiments, e.g., analysis of variance and regression.

3.2 A Graphical Interface

Unfortunately many factory engineers are not familiar with spatial data analysis nor with the **S** language. For these reasons, a graphical user interface (GUI) was designed to allow engineers to use the underlying **S** tools. The GUI front-end allows users to carry out the strategy we just described by point-and-clicking icons on their computer screens.

The GUI was designed to exploit engineers' familiarity with the existing Microsoft Windows environment. For users familiar with the **S** language, a commands window is always available.

The underlying tools were ported from the UNIX environment to the MS-Windows system to run in S-PLUS [S-PLUS, 1994], the commercial implementation of the **S** language; the graphical interface was written in the Visual Basic language [Visual Basic, 1994]. This setup allows the GUI program to invoke S-PLUS and to send strings through the VB SendKeys function and to receive strings from S-PLUS through the Windows clipboard as a means of interprocess communication. Thus, when the user clicks on an icon, the Visual Basic GUI constructs the appropriate **S** command and sends it to the S-PLUS program. The construction of the **S** expressions is based on the current state of the analysis as defined by the available data, any previously chosen groups, and the particular icon the user has clicked on. The GUI reads strings that are placed on the clipboard by the S-PLUS program and uses these to update the state of the analysis and provide feedback to the user.

4 The Desktop Presentation of the GUI and S-Plus

Figure 4 shows how the GUI appears on a computer screen. It consists of a top tool bar which depicts the array of available analysis tools, the left hand `Data` window for feedback and the S-PLUS window. The S-PLUS window contains four child windows: a commands window

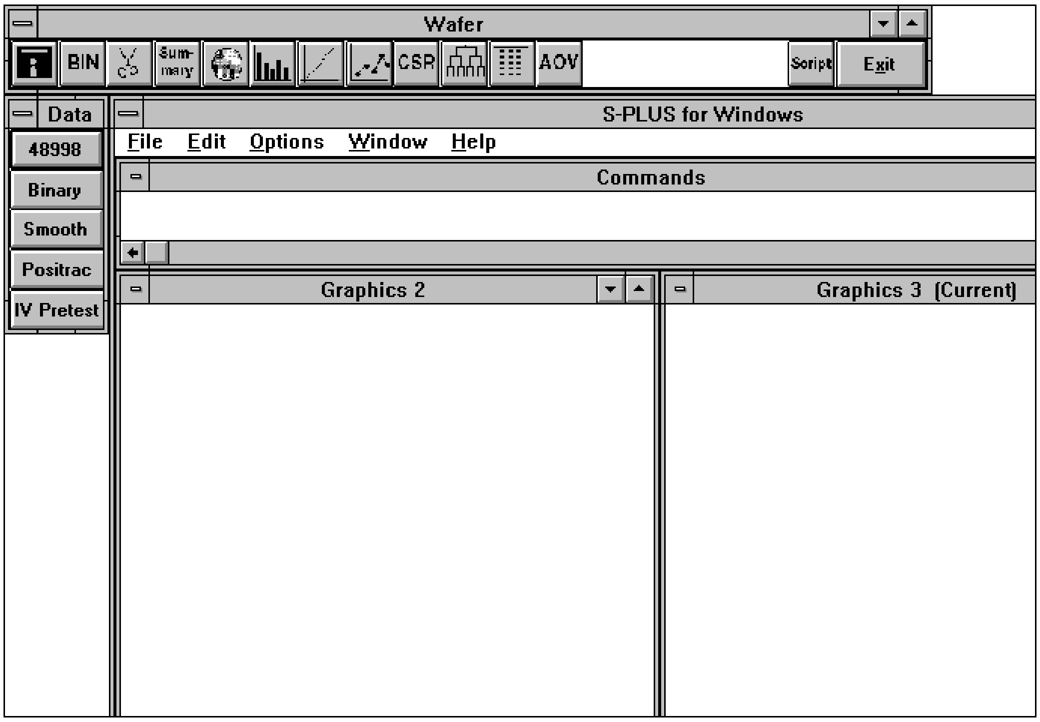


Figure 4: Desktop appearance of the GUI

for keystroke invoking of functions and three graphics windows. The GUI toggles between graphics windows 2 and 3 and uses graphics window 4 for larger plots.

The buttons in the top tool bar are used to read in data, define failure bins, subset and group wafers, perform various analyses and to extend the GUI by interfacing with S script files. The buttons are defined as follows:

 – reads in wafer map data, positrac, parametric data and split-lot tables;

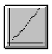
– defines binary (0/1) maps from probe data;

 – selects subset of wafers from a lot;

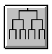
– prints an average map of the lot;

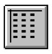
 – generates most of the plots of the raw, binary, smoothed, thresholded, mean wafers and of probe versus in-line and electrical test data;

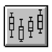
 –displays various barplots;

 – generates quantile-quantile plots of number of defectives by bin;

 – produces a chart of yield versus wafer label;

 –clusters wafers according to similar defect patterns;

 – a creates a table of counts of defective chips by wafer and failure mode;

 – displays boxplots of the square root of the counts of a particular bin for each of the groups previously formed;

– computes analysis of variance of data collected through split-lot or other designed experiments;

– runs **S** scripts.

5 An Example Analysis: The Power of Spatial Analysis

We next demonstrate the power of the graphical analysis of wafer failure data. In Figure 5 we show the binary wafers of a particular lot. From this plot we easily identify "scratches" and "blobs" for the various failure

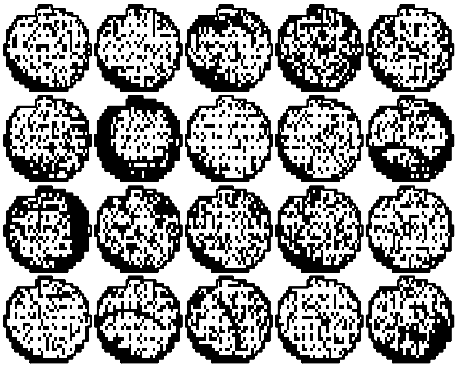


Figure 5: Binary wafers of example lot

modes. In Figure 6 we show the mean of the zero/one binary data across all wafers in the lot at each chip site, i.e., we show the proportion of defective chips at each site: values in the interval [0,1] have been mapped into shades of grey. We immediately see the high concentration of black chips in the lower left portion of the wafer, indicating sites that yielded almost no good chips; on a

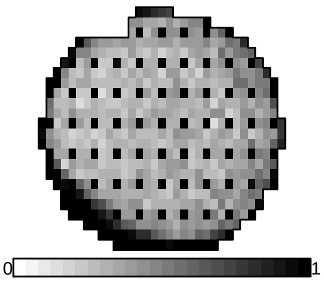
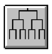


Figure 6: Mean at each chip site calculated across the binary wafers shown in Figure 5.

closer look, we see a systematic checker board pattern of zero-yield sites every third column and every third row that is not easily distinguishable in Figure 5. This pattern suggests a "masking" problem during lithography, and it would have been undetected using the old approach that discarded spatial information and monitored only wafer yield.

Simple graphics such as these are effective—they help engineers to easily look for and exploit spatial structure of defects. Together with other advanced statistical techniques engineers routinely monitor production for early detection of process problems as a means of continuous process improvements to maximize plant yield.

6 Second Example: How to Identify Wafers with Similar Spatial Patterns

The  button is used to find groups of wafers with similar spatial patterns. To do this, we use *cluster analysis* [Everitt, 1974]; the cluster analysis technique that we use is based on an iterative algorithm that at each step combines the two wafer groups with closest "centers". Conceptually the task is easy: smoothed data from the ith wafer may be thought as a vector $x_i \in \Re^k$, that is, the data denotes the location of a point in k dimensional euclidean space (k is the number of chips per wafer), and the task is to find groups of vectors whose distances are closest. Initially each wafer is defined to form its own (singleton) group; the method then computes all $n \times (n-1)$ distances among the n wafers and combines the two wafers that are closest; the location of this new group is defined to be its centroid (i.e., the mean of its members). The method then computes $n-2$ distances from the newly created group to the remaining groups (wafers), and combines the two closest groups. This process is repeated until all wafers are combined into one big group. The output of the method is the sequence of groups and their distances formed at each step, and it can be displayed concisely in a "tree" or "dendrogram." (See Figure 7.) Nodes in the tree denote

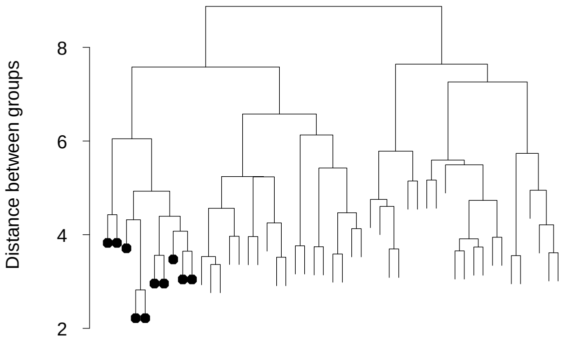


Figure 7: Appearance of the cluster tree after the leftmost node is clicked

groups formed at each iteration, and the groups consist of the wafers that fall under the node. The vertical distance between two nodes indicates the distance between the centroids of those two groups; the lower a node is in the tree, the closer their centroids are and the more similar its members are considered.

Notice that this technique computes a hierarchy of groups from which one decides which set of groups is the most meaningful according to their degree of "closeness." This can be accomplished in a number of ways: by looking at the tree for nodes close to the bottom, by fixing a (small) number of groups to be defined, or by merging all groups whose distances are smaller than a cut-off value.

Clicking the button results in a new window for the user to choose among various options, see Figure 8. We have chosen to cluster the smoothed wafer data (in-

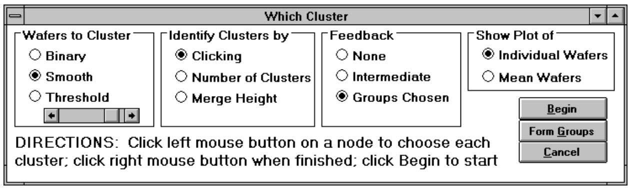


Figure 8: Form displayed by the GUI for the user to choose among the various options available for clustering the wafers and displaying the clusters

stead of binary or thresholded), to click on nodes to define groups, to have the GUI produce feedback as groups are being formed, and to show the individual wafers (instead of mean wafers) that fall under a node or group. When the user clicks on `Begin` a cluster tree appears in either the 2nd or 3rd graphics windows. The software then waits for the user to click on a node to indicate which leaves under a node will form a group. Figure 7 shows the cluster tree after the user has clicked on the left hand branch whose height is about 6. The leaves in that node are marked with a filled circle. The smoothed wafers in that branch or group appear in the other graphics window (Figure 9). This group appears

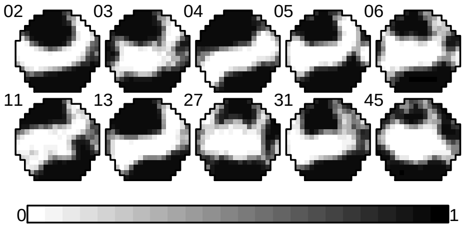


Figure 9: Plot of wafers that are clustered together in the leftmost node of the cluster tree in Figure 7.

to include those wafers with a large area of failed chips in the northwest quadrant and an elongated region of working chips from the top-right to the bottom-left. If the user clicks on the adjacent node with a merge height of around 5, we see the ten wafers in Figure 10; this node contains wafers with a failure pattern somewhat similar to the previous group, but the elongated strip of good chips has moved towards the bottom of the wafers. It is the location and size of the patterns that is different between these two groups, not so much the shape. Lastly,

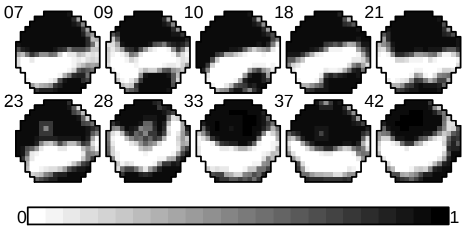


Figure 10: Plot of wafers that are clustered together in the second from the left node of the cluster tree in Figure 7

we show in Figure 11 the wafers in the right hand node of the tree; the failed pattern for these wafers occurs in the southeast area of the wafers. The user can continue clicking on nodes and viewing the resulting groups until satisfied that each group contains similar enough wafers. To retain these groups and to use their color coding in

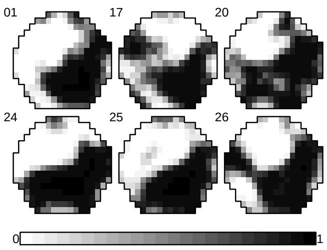


Figure 11: Plot of wafers that are clustered together in the rightmost node of the cluster tree in Figure 7

other analyses the user clicks the `Form Groups` command button and exits the clustering algorithm.

Figure 12 shows the final 6 groups used in producing Figure 3 in Section 3.

The spatial information in these six groups (i.e., defective areas' shape, location, and size) was used in relating wafer yield to in-line process measurements, as shown in Figure 3 and described in Section 3.

7 Advantages of Providing a GUI

The GUI provides easy access to a subset of packaged techniques. We have tried hard to make the GUI as flexible as possible and to provide a broad selection of statistical techniques.

This GUI does not access the full range of statistical techniques available in the S-PLUS language. It is

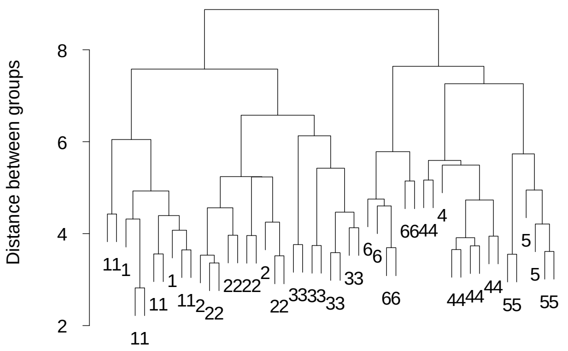


Figure 12: Dendrogram showing the final chosen groups.

instead directed to a particular audience, namely, engineers who analyze wafer yield data for the purpose of enhancing yield. These engineers typically do not know **S**, nor are they familiar with the syntax of the spatial analysis tools built in **S**. Being able to produce these analyses by pointing and clicking icons affords them the opportunity to very quickly conduct the sophisticated spatial analysis needed without being bogged down with a new language: there is very little training involved. Most standard graphs can be generated by a mouse click, instead of typing complicated expressions. Any subsetting or grouping of the data is accomplished by interactive mouse clicks and then used throughout subsequent analyses.

Since there are a large number of processing steps, electrical tests and possible binary definitions, there is potential for a tremendous number of plots that could be viewed in search for "interesting" relationships between the various sets of data. Therefore, the GUI provides some guidance to finding plots that might depict meaningful relationships. When the user requests certain types of plots, the GUI orders those plots according to some reasonable criterion and lists them in a way that choices at the beginning have a better chance of showing a stronger relationship than those at the end.

The S-PLUS commands window remains available at all times. The user can use the GUI to produce standard analyses and then use keystroke commands to perform any analysis that is not available in the GUI. The GUI is easily extended through the Script button. **S** script files can be linked to the GUI and selected by a simple mouse click. Therefore, experienced engineers can facilitate the inclusions of any new techniques that may be found useful and make them easily available to other users.

8 Problems and Issues in Writing a GUI

Providing a GUI along with the underlying S-PLUS functions requires extra programming, much of it is tedious. The testing and debugging of the GUI is not straightforward. Supplying test routines that run in the background is not possible without additional expensive tools. Testing of new releases involves not only the testing of the underlying **S** functions but also of the order of the point-and-clicking function invocation.

Designing a GUI for statistical software raises issues on human-computer interface which one does not face when designing a keystroke analysis package. Currently we have only completed a PC based GUI. Implementation on UNIX might be done using a different interface language.

In the way that we implemented the GUI, namely a Visual Basic program calling S-PLUS and sending keystroke commands to S-PLUS, we are faced with many timing issues and automatic control of window focus. The resolution of the timing and control issues is tedious and time consuming. Some of the problems are machine dependent, e.g., they only occur on slow or fast machines. So, testing involves trying the software on a number of different PC's of different configurations.

Decisions need to be made as to how to divide the labor between the VB subroutines and S functions. At one extreme, the VB program could be programmed so that a click on a tool bar involves invoking an S function that does all the work of figuring out the user's request and performing the tasks. At the other extreme, the clicking of a toolbar could involve the VB program forming a large number of S-PLUS command statements and sending them one at a time to the core S-PLUS functions. We have programmed this implementation somewhere in the middle. When a large number of statement calls are needed, they have been bundled into S functions. When fewer are needed, they are sent individually. In some cases VB puts together very complicated S-PLUS function calls.

There are some problems with error handling. If S-PLUS dies ungracefully, then so does the GUI. If S-PLUS falls sick with an error, then the GUI is not able to detect that error and recover from it. Better error handling should be available in S-PLUS 3.3.

Most **S** functions have a number of default settings for their arguments. Some of these defaults are set from the GUI but many are not at the present time. Decisions were made as to which ones the user could get point-and-click access to and about the balance that should be attained between user control and default choices.

The performance of S-PLUS is slightly degraded by the GUI overhead. This is more serious on systems with limited (8 Meg) RAM.

9 Conclusions

A graphical user interface has helped us transfer an advanced statistical environment to factory engineers. The GUI enables engineers to to use spatial analysis techniques and software with little initial investment of their time and to quickly concentrate on understanding the data without worrying about the syntax of the underlying **S** environment. As they become more familiar with the techniques they can extend them by learning more about spatial analysis, about the **S** language and the underlying wafer analysis environment. For more sophisticated users the GUI remains unobtrusive if they want to perform analyses directly in the underlying environment. Keystroke commands can easily be input into the Commands window.

To a large extent, the GUI can guide engineers through complex analysis. The "best current practice" of data analysis of the wafer data can be conveyed to less experienced engineers through the guidance provided in the GUI. The GUI is extensible and as new analysis techniques are developed they can be easily incorporated.

10 Acknowledgments

We want to thank Mark Hansen for his numerous comments and suggestions regarding the software and this paper.

References

[Becker et al., 1988] Becker, R. A., Chambers, J. W., and Wilks, A. R. (1988). *The New S Language.* Wadsworth, Pacific Grove, California.

[Chambers and Hastie, 1992] Chambers, J. W. and Hastie, T., editors (1992). *Statistical Models in S.* Wadsworth, Pacific Grove, California.

[Cliff and Ord, 1981] Cliff, A. D. and Ord, J. K. (1981). *Spatial Processes: Models and Applications.* Pion Limited, London, UK.

[Everitt, 1974] Everitt, B. (1974). *Cluster Analysis.* John Wiley & Sons, New York.

[Friedman et al., 1995a] Friedman, D. J., Hansen, M. H., Hoyer, J. R., and Nair, V. N. (1995a). Process improvement through analysis of spatially clustered defects on wafer maps. In preparation.

[Friedman et al., 1995b] Friedman, D. J., Hansen, M. H., James, D. A., and Nair, V. N. (1995b). Estimation of d_0 and y_0 using a non-model based approach. In preparation.

[Hansen and Nair, 1995] Hansen, M. H. and Nair, V. N. (1995). Process and productivity improvement in semiconductor manufacturing. In Mayer, M., editor, *Computing Science and Statistics*, volume 27. Interface Foundation of North America. to appear.

[S-PLUS, 1994] S-PLUS (1994). *S-PLUS Software Documentation.* StatSci Division of MathSoft Inc., Seattle, WA. Version 3.2.

[Visual Basic, 1994] Visual Basic (1994). *Visual Basic Language.* Microsoft Corporation, Redmond, WA. Version 3.0.