Linking ArcView 3.0™ and XGobi: Insight Behind the Front End

Jürgen Symanzik[1], James J. Majure[2], Dianne Cook[1], Inna Megretskaia[1]

[1] Department of Statistics, Iowa State University, Ames, IA 50011, USA

[2] Miner and Miner, Greeley, CO 80632, USA

symanzik@iastate.edu

Abstract

This paper presents aspects of the implementation of a bidirectional link between the Geographic Information System (GIS) ArcView 3.0™ and the interactive dynamic statistical graphics program XGobi. We describe the main functionality of the link, explain the underlying Remote Procedure Call (RPC) mechanism and internal data structures, and deal with topics such as security, concurrency, and linked brushing. We think that these topics are of particular interest to software authors, intending to link similar software packages, and software users, learning about strengths (and weaknesses) of the implementation of our link.

Keywords: Dynamic statistical graphics; Geographic information system; Implementation issues; Spatial cumulative distribution function; Spatial statistics; Spatially lagged scatterplot; Variogram–cloud plot.

1 Introduction

Geographic Information Systems (GISs) are used extensively for examining spatial data but they have few facilities for exploratory spatial data analysis. It is for this reason that in 1994 we started work on a bidirectional link between the GIS ArcView (first versions 2.0 and 2.1, now version 3.0) and an interactive dynamic statistical graphics program in the X Window System™ environment, XGobi (Swayne, Cook & Buja, 1991; Buja, Cook & Swayne, 1996). The first features of the link were the handling of multivariate attribute data and spatial cumulative distribution functions (SCDFs). Later, features such as the variogram–cloud plot (Haslett, Bradley, Craig, Unwin & Wills, 1991; Haslett & Bradley, 1991; Bradley & Haslett, 1992), spatially lagged scatterplot (Cressie, 1984; Rossi, Mulla, Journel & Franz, 1992), and multivariate variogram–cloud plot (Majure & Cressie, 1997) have been added.

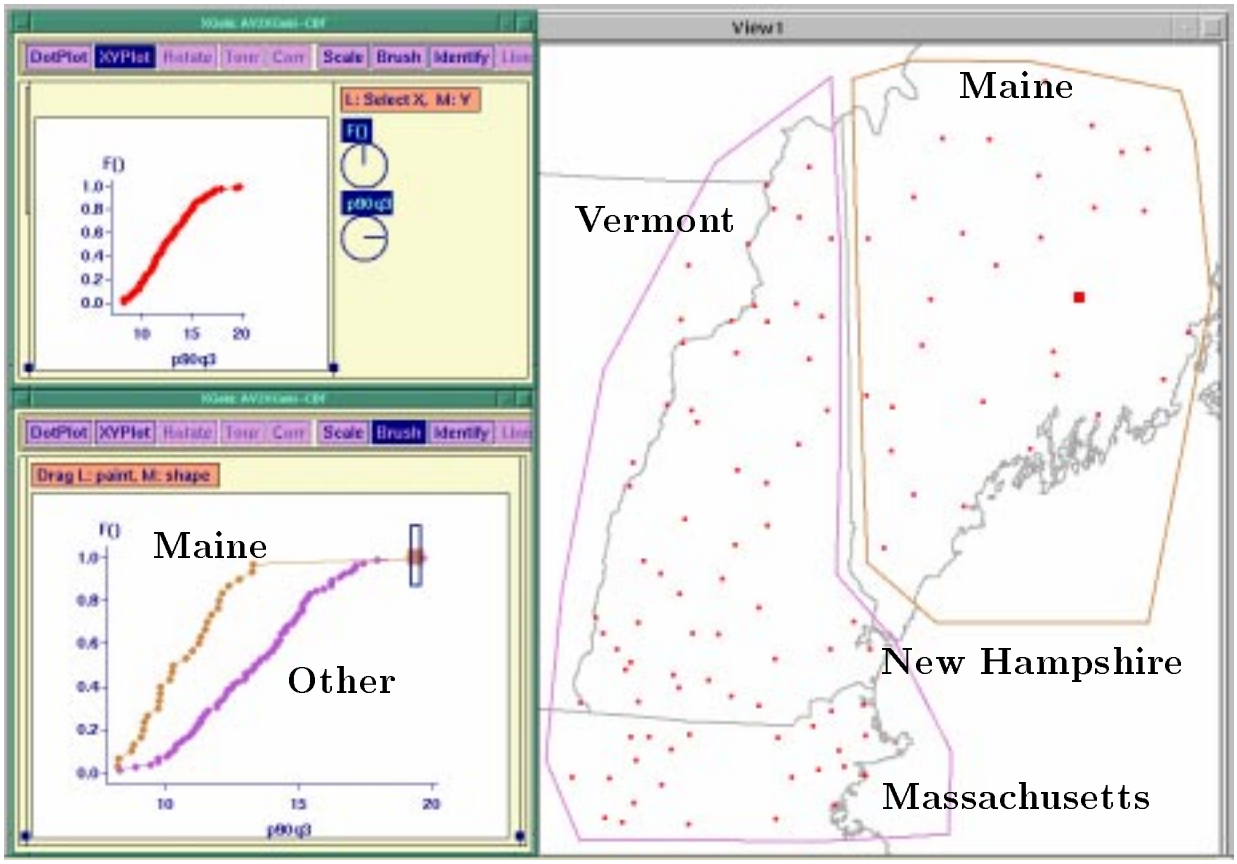

Figure 1: Example for the SCDF View. While the joint SCDF over all observations (top left XGobi) does not reveal any unusual value, the two SCDFs for Maine (upper/brown points) and other states (lower/purple points) reveal one unnaturally large observation in Maine (bottom left XGobi). This location has been brushed with a big filled box in the XGobi view and right ArcView map view.

The usefulness of the link, which allows us to display spatial locations and concomitant geographic variables within the GIS while visualizing and exploring the corresponding data space within XGobi simultaneously, has been highlighted for several different applications such as satellite imagery, forest health monitoring, and precipitation data (Cook, Majure, Symanzik & Cressie, 1996; Majure, Cook, Cressie, Kaiser, Lahiri & Symanzik, 1996a; Majure, Cressie, Cook & Symanzik, 1996c; Symanzik, Majure & Cook, 1996a; Symanzik, Majure & Cook, 1997b). Cook, Symanzik, Majure & Cressie (1997) presents examples (measures of livability in the United States, forest health data, and precipitation measurements) for all five features of the link. There also exist videos that demonstrate the use of the link (Majure, Cook, Cressie, Kaiser, Lahiri & Symanzik, 1995; Symanzik, Majure & Cook, 1995; Majure, Cook, Symanzik & Megretskaia, 1996b).

Before we deal with the technical aspects of the links, we first want to give an example of its use. Figures 1 and 2 show a way to detect a potential spatial

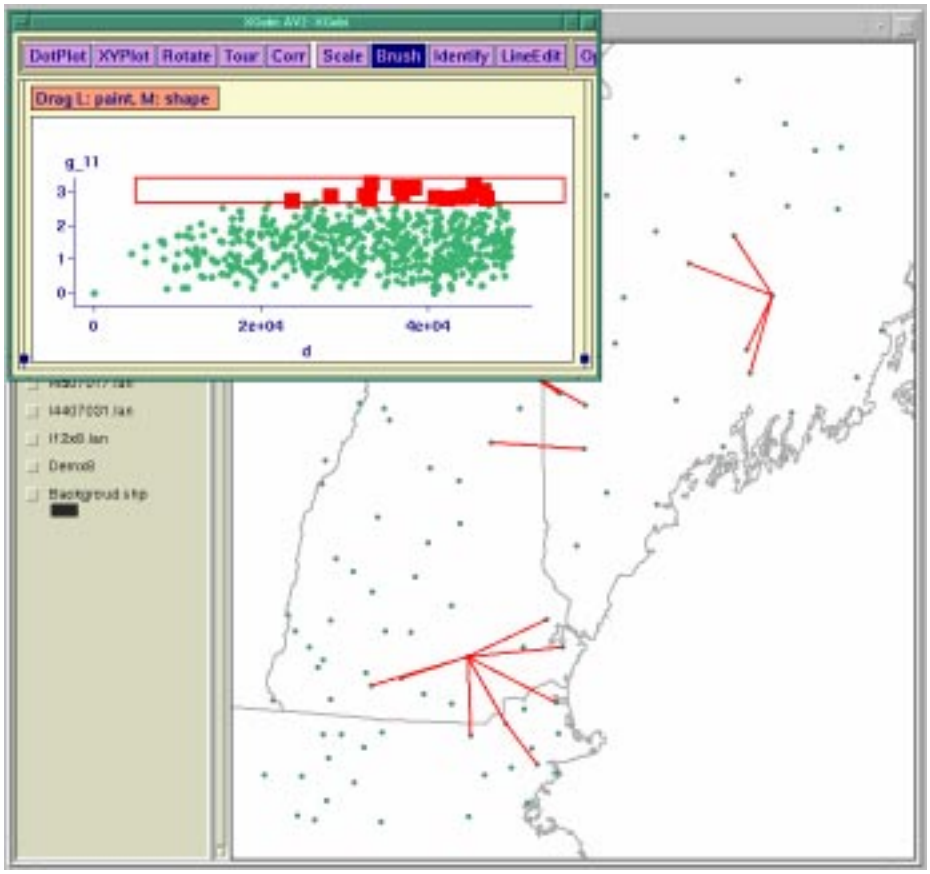

Figure 2: Example for the Variogram–Cloud Plot. In the upper left XGobi, we have brushed (using a red filled square) the highest values in the variogram cloud plot ($g_{11}(\boldsymbol{s_i}, \boldsymbol{s_j})$) versus $d(\boldsymbol{s_i}, \boldsymbol{s_j})$). In the lower right ArcView map view, each pair of locations, related to a point that has been brushed in XGobi, has been connected by a line of the same (red) color

outlier. The ArcView map in each figure shows precipitation recording sites in the northeastern United States, i. e., in Maine, New Hampshire, Vermont, and Massachusetts. The measurements are precipitation totals for the third quarter of 1990. Figure 1 shows the predicted SCDF (in this case we used the usual empirical CDF as predictor) for the entire region in the top left (XGobi) plot, and the predicted SCDF for Maine plotted with the predicted SCDF for the other states in the bottom left (XGobi) plot. A feature of the Maine measurements is immediately visible: one sampling site has a massively large precipitation value in comparison to all other sampling sites in Maine. It is marked in the right (ArcView) map view by a large solid square. This "outlier" was masked in the SCDF for the entire region. This measurement is an outlier in a spatial sense because it is very different from its neighboring sampling sites.

A more common approach to detecting spatial outliers is to use a variogram–cloud type plot due to Cressie (1984) ($g_{11}(\boldsymbol{s_i}, \boldsymbol{s_j}) =\mid Z_1(\boldsymbol{s_i}) - Z_1(\boldsymbol{s_j}) \mid^{1/2}$ versus the Euclidean distance between the locations, $d(\boldsymbol{s_i}, \boldsymbol{s_j}) =\mid\mid \boldsymbol{s_i} - \boldsymbol{s_j} \mid\mid$, where $Z_1(\boldsymbol{s_i})$

represents the amount of precipitation measured at location $\boldsymbol{s_i}$ during the third quarter of 1990). This approach is demonstrated in Figure 2. Each point in the variogram–cloud corresponds to two sampling sites in the map. High values of g_{11} (the precipitation difference between the sampling sites is large) are brushed in the upper left (XGobi) variogram–cloud plot and the corresponding pairs of sampling sites are identified by connecting lines in the lower right (ArcView) map view. Locations that have several lines emanating from them are sites that are most different from their neighbors, that is, spatial outliers. This display shows two such sites: the one in Maine that was also identified using the SCDF, and another one in New Hampshire.

A first technical description of the implementation of the link via Remote Procedure Calls (RPCs) has been given in Symanzik et al. (1996a) and an update appears in Symanzik et al. (1997b). The data structures for linking of the initial features were straightforward. However, the additional spatial plots, i. e., variogram–cloud plots, spatially lagged scatterplots, and multivariate variogram–cloud plots, require efficient handling and storage of large data sets. For example, up to n^2 points may appear in either of these plots, even though there are only n observations in the underlying data set. A cutoff distance d, selected by the user, may considerably reduce the number of points that appear in either of these plots, but efficient linking is paramount. Implementation issues of variogram–cloud plots and spatially lagged scatterplots have been discussed first in Symanzik, Megretskaia, Majure & Cook (1996b).

This current paper not only summarizes and updates information given in Symanzik et al. (1996a, 1996b, 1997b) but also adds many new aspects and details previously unpublished. It is intended to highlight particularly important implementation issues of the link, give ideas how to extend this link, and focus on ideas that are of general interest when linking software systems.

The idea of linking statistical plots with geography as an important factor for analyzing spatially referenced data has been discussed in several places. Also, many software solutions have been developed where either geography and statistical analysis are combined in one software package or packages for geography and statistical analysis have been linked.

In McDonald & Willis (1987), a grand tour (Asimov, 1985; Buja & Asimov, 1986) is linked to an image to assess the clustering of landscape types in the band space of a LandSat image taken over Manaus, Brazil. In Carr, Littlefield, Nicholson & Littlefield (1987) and Monmonier (1989), a scatterplot matrix is linked to a map view. It appears that in this context Monmonier is the first who uses the term "*geographic brushing*". Monmonier (1988) describes a conceptual framework for geographical representations in statistical graphics.

In REGARD (Unwin, Wills & Haslett, 1990), map views are linked with histograms and scatterplots and, moreover, diagnostic plots for assessing spatial dependence are also available. Another explorotary system that links histograms and scatterplots with latitude and longitude (and depth) coordinates is discussed in MacDougall (1992). In Carr, Olsen & White (1992), (bivariate) ray–glyph

maps have been linked with scatterplots. DiBiase, Reeves, MacEachren, von Wyss, Krygier, Sloan & Detweiler (1994) provide an overview on existing multivariate (statistical) displays for geographic data. Klein & Moreira (1994) report on an interface between the image program MTID and XGobi, used for the exploratory analysis of agricultural images. Some of the most recent developments are the cartographic data visualizer, cdv (Dykes, 1996), where a variety of plots are linked with geography, the Space–Time–Attribute Creature/Movie, STAC/M (Openshaw & Perrée, 1996), that searches for patterns in GIS databases under the control of a Genetic Algorithm, and an exploratory spatial analysis system in XLisp–Stat (Brunsdon & Charlton, 1996).

It should be noted that in most geographically influenced publications authors distinguish between "*geographic (or cartographic) space*" and "*attribute (or data) space*" but rarely use the statistical expression "*variable*" to relate to the latter one.

In contrast to the previously mentioned software solutions where smaller packages have been developed or have been linked, there exist several approaches where GISs and (graphical) statistical packages have been linked, utilizing all the strengths of the GIS and being able to overlay additional map features on the scatterplot view in the GIS. Scott (1994) links STATA with ArcView and Anselin & Bao (1996) link the spatial data analysis software SpaceStat with ArcView. In Haining, Ma & Wise (1996), the designing of a software system for interactive exploration of spatial data by linking to ARC/INFO^{TM} is discussed. And, finally, Mathsoft (1996) provides the S+Gislink, a bi–directional link between ARC/INFO and S $-$ PLUS$^{®}$.

Our link between ArcView and XGobi obviously falls into the latter category of software solutions. However, it can not be considered as *yet another* link where GIS and statistical package have been combined, but instead, it provides some plots that have been developed *especially* for exploring spatial dependence.

We will discuss the main functionality of the link in Section 2, XGobi's command structure in Section 3, and main features in ArcView in Section 4. In Section 5, we will consider theoretical aspects of variogram estimation related to variogram–cloud plots and we will discuss XGobi's data structures and the idea of linked brushing for the variogram–cloud link. Security and concurrency issues will be discussed in Section 6. We conclude this paper in Section 7 with an overview on the future directions of the work.

Our link has been developed for DEC^{TM} alphastations. It also has been tested on other platforms such as Sun^{TM}/Sparc^{TM} workstations, SGI^{TM} workstations, and workstations from Data General Corporation. A current version of the soft-

^{TM}*ARC/INFO* is a trademark of Environmental Systems Research Institute, Inc.
S–PLUS is a registered trademark of StatSci, a division of MathSoft, Inc.
^{TM}*DEC* is a trademark of Digital Equipment Corporation.
^{TM}*Sun* is a trademark of Sun Microsystems, Inc.
^{TM}*Sparc* is a trademark of Sun Microsystems, Inc.
^{TM}*SGI* is a trademark of Silicon Graphics, Inc.

ware can be downloaded from the WWW at the URL

`http://www.gis.iastate.edu/XGobi-AV2/XGobi-AV2.html`.

A description of the software from the user's point of view is given in Symanzik, Majure & Cook (1997c).

2 The Main Functionality of the Link

The main goal of our research during the last few years was to establish a bidirectional link between ArcView (first versions 2.0 and 2.1, now version 3.0) and XGobi. This link utilizes Remote Procedure Calls (RPCs), an Interprocess Communication (IPC) feature available in ArcView. The use of RPCs is a programming technique where a process on the local system (i. e., the client) invokes a procedure on a remote system (i. e., the server). In this context, the term request is used to refer to the client's desire to execute a particular remote procedure, and the term response is used for the result produced by the remote procedure (Stevens, 1990).

RPCs are not the only existing facility for IPC. Stevens (1990) mentions pipes, FIFOs (named pipes), message queues, semaphores, shared memory, Berkeley sockets, and the System V Transport Layer Interfce (TLI) as additional facilities for IPC on UNIX® systems. Using the term "*close coupling*", introduced in Goodchild, Haining & Wise (1992) to describe systems where one package is calling other packages or user–written routines, one has to assume that such systems make use of these previously listed IPC facilities.

There exists even a simpler solution for IPC, i. e., multiple processes that are simultaneously accessing (reading and writing) the same file(s) or multiple processes that are simply passing data files (ASCII or binary) among each other. This approach is called "*loose coupling*" in Goodchild et al. (1992).

Communication between ARC/INFO (or ArcView) and a statistical package can be either based on loose or close coupling. The link between ArcView and SpaceStat (Anselin & Bao, 1996) and the software system designed by Haining et al. (1996) are examples for loose coupling.

There exists a third approach where no IPC facility is needed, namely, purpose written software that allows to conduct spatial data analysis directly in the geographic context. REGARD (Unwin et al., 1990) and cdv (Dykes, 1990) fall into this category.

ArcView (2.0, 2.1, and 3.0) does not allow any choice among IPC facilities for close coupling since, on UNIX systems, it uniquely supports RPCs. From within ArcView, external remote procedures can be called and external programs can invoke internal ArcView functions via RPCs. In the beginning, i. e., in the link with ArcView 2.0 and 2.1, XGobi has been adapted to the subroutine template code provided with the XGobi source code. This approach previously has been used for the implementation of the XGvis software system (Littman, Swayne,

UNIX is a registered trademark of UNIX Systems Laboratories.

Dean & Buja, 1992), for the interface between the image program MTID and XGobi (Klein & Moreira, 1994), and for our previous link between ARC/INFO and XGobi (Symanzik, Majure, Cook & Cressie, 1994; Cook, Cressie, Majure & Symanzik, 1994). Finally, in the link to ArcView 3.0, the RPC mechanism became a permanent part of the XGobi source code and can be (de–)activated on platforms that (do not) support this type of RPC mechanism when compiling the software.

ArcView has been customized for this application using its built–in Avenue programming language. Virtually, all of the default ArcView functionality is available, plus several operations that deal only with the link. Specifically, ArcView has been modified to do the following: initiate an RPC server and client, initiate and pass data to the XGobi process, brush points and request XGobi to brush the corresponding points, and process requests from XGobi to brush points. This functionality can be described in the following pseudo code:

```
repeat
  wait for action
  case action {
    when USER action:
      process USER action
    when START_LINK request:
      init ArcView as RPC server
      initiate the XGobi process
      init ArcView as RPC client (i.e., connect to
        XGobi RPC server)
      send RPC request to XGobi containing the data to
        be input into XGobi
      activate appropriate GUI control buttons
    when STOP_LINK request:
      send RPC request to XGobi to shut down
      shut down RPC server
      shut down RPC client
      deactivate appropriate GUI control buttons
    when BRUSH_POINTS request:
      change the color/glyph/size of specified points
      send RPC request to XGobi to brush corresponding points
    when RPC request from XGobi:
      change the color/glyph/size of specified points or draw
        connecting lines between specified points
  }
until (ArcView QUIT action)
```

Both, ArcView and XGobi have been set up as a server for RPCs as well as a client for RPCs. This allows ArcView (as a client) to access the functionality added to XGobi (the server). In Section 3, we will describe which XGobi remote procedure can be called from within ArcView. It is not possible to give XGobi the full functionality of an RPC server that automatically processes all requests from its clients since such an automation would never terminate and thus, no user action within XGobi could be processed any more. Instead, a working procedure, that is a routine that runs once whenever the X Window System event loop finds no event, has been added to XGobi to check for incoming RPC requests from clients (namely ArcView) and to process these requests.

On the other hand, XGobi (as a client) sends requests to ArcView (the server), asking for the update of the ArcView view in accordance with the brushing and subsetting information of points within XGobi where this information has been altered. The modified XGobi can be described via the following pseudo code:

```
init XGobi as RPC server
init XGobi as RPC client
init XGobi defaults
repeat
  wait for action
  case action {
    when RPC request from ArcView:
      execute appropriate function (i.e., update XGobi
        structures, data sets, and views)
      return response (success, fail, or result) to client
    when USER action:
      process USER action
      if (USER action = Move_Brush_Symbol)
        make RPC call (i.e., submit brushing information)
        wait for response (success or fail) from server
        if (fail)
          print warning
  }
until (RPC request = Quit_XGobi)
shutdown RPC server
exit
```

3 XGobi's Command Structure

It was necessary to add a large number of remote procedures to XGobi, executable through the XGobi RPC server, to obtain the functionality for the five features of the link. These remote procedures, callable from within ArcView, can be grouped into six classes: general use (function numbers 01—03), multivariate attribute data (11–16), SCDF plots (21–29), variogram–cloud plots (31–35), spatially lagged scatterplots (41–46), and multivariate variogram–cloud plots (51–55). Remote procedures for general use and multivariate attribute data are listed below. A similar command structure exists for all other classes, i. e., features of the link.

01: RPC_Send_Colnames
In: colp[ncols] : ncols × Str[COLLABLEN]
Pre: is_init (11) or is_cdf_init (21)
Res: NULL
Set: NULL

02: RPC_Send_Rownames
In: rowp[nrows] : nrows × Str[ROWLABLEN]
Pre: is_init (11) or is_cdf_init (21)
Res: NULL
Set: NULL

03: RPC_Clone_XGobi
In: NULL
Pre: xgobi_is_up (14 or 24 or 34 or 44 or 54)
Res: NULL
Set: NULL

11: RPC_Init_Data
In: (name, ncols, nrows) : (String, int, int)
Pre: NULL
Res: is_init, has_data, has_symbols, is_cdf_init, has_cdf_data, has_cdf_symbols, has_cdf_bitmap, has_cdf_weights, is_vario_init, has_vario_data, has_vario_symbols, is_lag_init, has_lag_data, has_lag_symbols, is_vario2_init, has_vario2_data, has_vario2_symbols
Set: is_init

12: RPC_Send_Init_Data
In: datap[nrows, ncols] : (nrows · ncols) × float
Pre: is_init (11)
Res: NULL
Set: has_data

13: RPC_Send_Init_Symbols
In: size_color_glyph[nrows] : nrows × int
Pre: has_data (12)
Res: NULL
Set: has_symbols

14: RPC_Make_XGobi
In: NULL
Pre: has_symbols (13)
Res: NULL
Set: is_running, xgobi_is_up

15: RPC_Update_All_Symbols
In: size_color_glyph[nrows] : nrows × int
Pre: has_symbols (13) and xgobi_is_up (14)
Res: NULL
Set: NULL

16: RPC_Update_Some_Symbols
In: list of index:size_color_glyph : list of (int:int)
Pre: has_symbols (13) and xgobi_is_up (14)
Res: NULL
Set: NULL

In indicates the input to a remote procedure. Everything (e. g., numbers and names) that is passed from ArcView into XGobi (and vice versa) has to be coded as a single ASCII string to fulfill ArcView's internal specifications. Unfortunately, this encoding is not very efficient for larger data sets, but the current specification in ArcView (given by ESRI) does not allow a more efficient data format.

Pre shows the prerequirements for a remote procedure. E. g., procedures 11, 12, 13, and 14 have to be called in this order. Optionally, remote procedures 01 and 02 can be called when 11 (or 21) has been called.

Res indicates which flags are set to *false* when a remote procedure is called.

Set indicates which flags are set to *true* when a remote procedure is called.

The result of the XGobi remote procedures that is returned to ArcView usually consists of 5 character symbols: two digits representing the function number, a dot as a separator, and another two digits describing the result (error). ArcView uses a particular file (*AV2XGobi.errors*) that contains all possible cases of "*function number.error digits*" that may possibly occur. This file consists of three columns, separated by a blanc. The first column contains the "*function number.error digits*" described above.

The second column contains the severity of the error. A "0" means success. It is used only in combination with the error digits "00". An "R" indicates that incorrect numerical values have been received and ArcView should repeat (this is what the "R" stands for) its last request using correct data. A "W" indicates a warning. However, it is possible to continue work. An "F" indicates a fatal error. It is impossible to recover from this type of error and the link is terminated.

The third column contains the name of the remote procedure and a verbal description of the error. Both, name and description, are displayed in an ArcView message window in case of an "R", "W", or "F" condition. The first few lines of *AV2XGobi.errors*, related to the remote procedures for general use and multivariate attribute data, are listed below:

```
01.00 0 RPC_Send_Colnames: OK.
01.01 F RPC_Send_Colnames: Call RPC_Init_Data or RPC_Init_CDF_Data first.
01.02 F RPC_Send_Colnames: Submit number of columns many column names
  (separated by space).

02.00 0 RPC_Send_Rownames: OK.
02.01 F RPC_Send_Rownames: Call RPC_Init_Data or RPC_Init_CDF_Data first.
02.02 F RPC_Send_Rownames: Submit number of rows many row names
  (separated by space).

03.00 0 RPC_Clone_XGobi: OK.
03.01 F RPC_Clone_XGobi: Call RPC_Make_XGobi, RPC_Make_CDF_XGobi,
  RPC_Make_VARIO_XGobi, or RPC_Make_LAG_XGobi first.
03.02 W RPC_Clone_XGobi: Can not clone XGobi - set environment variable
  AV2XGOBI_HOME first.

11.00 0 RPC_Init_Data: OK.
11.01 F RPC_Init_Data: Submit a name, number of columns, and number of rows
  (separated by space).

12.00 0 RPC_Send_Init_Data: OK.
12.01 F RPC_Send_Init_Data: Call RPC_Init_Data first.
12.02 F RPC_Send_Init_Data: Submit (number of rows x number of columns)
  many values (separated by space).

13.00 0 RPC_Send_Init_Symbols: OK.
13.01 F RPC_Send_Init_Symbols: Call RPC_Send_Init_Data first.
13.02 F RPC_Send_Init_Symbols: Submit number of rows many values
  (separated by space).
13.03 R RPC_Send_Init_Symbols: Wrong value for color/glyph/size.

14.00 0 RPC_Make_XGobi: OK.
14.01 F RPC_Make_XGobi: Call RPC_Send_Init_Symbols first.
14.02 F RPC_Make_XGobi: Can not evoke XGobi.

15.00 0 RPC_Update_All_Symbols: OK.
15.01 F RPC_Update_All_Symbols: Call RPC_Send_Init_Symbols and
  RPC_Make_XGobi first.
15.02 F RPC_Update_All_Symbols: Submit number of rows many values
  (separated by space).
15.03 R RPC_Update_All_Symbols: Wrong value for color/glyph/size.

16.00 0 RPC_Update_Some_Symbols: OK.
16.01 F RPC_Update_Some_Symbols: Call RPC_Send_Init_Symbols and
  RPC_Make_XGobi first.
16.02 F RPC_Update_Some_Symbols: Submit a list of number:color/glyph/size
  (separated by space).
```

```
16.03 R RPC_Update_Some_Symbols: Wrong value for number.
16.04 R RPC_Update_Some_Symbols: Wrong value for color/glyph/size.
```

4 Features in ArcView

The implementation of the link in ArcView was accomplished by a combination of Avenue scripts and modifications of the graphical user interface (GUI). In the context of the link, ArcView has to do two things: 1) respond to user actions and eventually request appropriate changes in XGobi, and 2) respond to RPC requests from XGobi and take appropriate action.

User actions include all normal ArcView capabilites and those added specifically to support the link. New functionality added includes:

- the ability to start and stop XGobi (a green and red diamond in the GUI);
- the ability to set the current brushing characteristics: color, glyph, and size of the symbol used for plotting the points ("Color", "Glyph", and "Size" pulldown menus in the GUI); and
- the ability to brush points on the ArcView map (the paint brush symbol in the GUI).

Figure 3 shows the *View* Document GUI that is being used for the multivariate attribute data, the variogram–cloud plot, the spatially lagged scatterplot, and the multivariate variogram–cloud plot features of the link. A slightly different GUI is being used for the SCDF feature of the link.

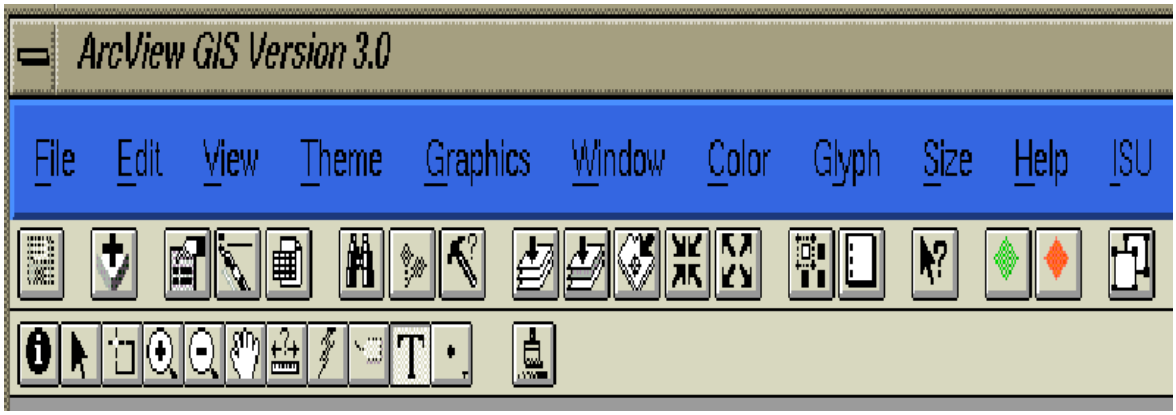

Figure 3: The *View* Document GUI in ArcView 3.0.

The following Avenue scripts, listed in alphabetical order, are required in ArcView to support the multivariate attribute data feature of the link:

aa_global_variable_declaration: Here, all global Avenue variables that are later used in other scripts are defined. This definition has to happen in the first (according to ASCII order) Avenue script.

CreateErrorDictionary: Builds the table of error messages that is internally used in ArcView based on the file *AV2XGobi.errors* (described in Section 3).

RGBColorNameDictionary: Creates a dictionary containing ASCII color names and RGB color values that are related to the colors used in XGobi.

SetCurrentColor: Selects and sets a new color in the "Color" pulldown menu.

SetCurrentGlyph: Selects and sets a new glyph in the "Glyph" pulldown menu.

SetCurrentSize: Selects and sets a new size in the "Size" pulldown menu.

Startup: Reads environment variables and executes the scripts **RGBColorNameDictionary** and **CreateErrorDictionary**.

XG.Clone: Calls remote procedure 03 (described in Section 3) in XGobi.

XG.Finish: Sends a "finish" request to XGobi, terminates the RPC Server within ArcView, and resets the buttons in the ArcView GUI.

XG.ReadXGobiColors: Attempts to read the `$HOME/app-defaults/XGobi` file (if available) and tries to get the current XGobi colors from it.

XG.RPC.Brush.Some: This script is called from within XGobi when some points have been brushed in the XGobi window and an update of the ArcView view is required. The data that is passed is of the type "list of index:size_color_glyph" where "index" is the internal number of the point and "size_color_glyph" is the color/glyph/size information encoded according to internal XGobi specifications and, therefore, first has to be decoded in ArcView.

XG.RPC.XGobi_Is_Up: This script is used to consume some time while ArcView is waiting for XGobi to start up. It does not do anything other than delaying the execution for a while.

XG.Select This script allows the user to select points in the current ArcView view by drawing a boundary based on polygons. The points inside the boundary will be brushed using the current selections in the "Color", "Glyph", and "Size" pulldown menus. Also, remote procedure 16 (described in Section 3) in XGobi is called, with the color/glyph/size information of all selected points, encoded according to internal XGobi specifications.

XG.Start This script initiates the ArcView to XGobi link. It activates the RPC mechanism in ArcView, starts XGobi as a background process, and initializes the new buttons and functionality in the ArcView GUI. Internally, it calls remote procedures 11, 1, 2, 12, 13, and 14 (described in Section 3) in XGobi.

Similar scripts are used to support the four other features of the link.

5 The Variogram–Cloud Link

5.1 Theoretical Aspects of Variogram and Cross–Variogram Estimation

The function

$$2\gamma(\boldsymbol{h}) = 2\gamma(\boldsymbol{s_1} - \boldsymbol{s_2}) = Var(Z(\boldsymbol{s_1}) - Z(\boldsymbol{s_2})),$$

where $\boldsymbol{s_1}, \boldsymbol{s_2} \in I\!R^d$ (the spatial domain), is called the variogram. One often assumes that $2\gamma(\boldsymbol{h})$ only depends on the distance $\boldsymbol{h} = \| \boldsymbol{s_1} - \boldsymbol{s_2} \|$, but not on particular locations $\boldsymbol{s_1}$ and $\boldsymbol{s_2}$. Several variogram models exist and have been discussed in the literature (e. g., Cressie, 1993, Section 2.3.1). Several estimators for $2\gamma(\boldsymbol{h})$ are known, e. g., the natural estimator

$$2\hat{\gamma}(\boldsymbol{h}) = \frac{1}{\mid N(\boldsymbol{h}) \mid} \sum_{N(\boldsymbol{h})} (Z(\boldsymbol{s_i}) - Z(\boldsymbol{s_j}))^2, \quad \boldsymbol{h} \in I\!R^d,$$

where

$$N(\boldsymbol{h}) = \{(\boldsymbol{s_i}, \boldsymbol{s_j}) \mid \| \boldsymbol{s_i} - \boldsymbol{s_j} \| = \boldsymbol{h}; \quad i, j = 1, \ldots, n\}.$$

However, Cressie & Hawkins (1980) and Hawkins & Cressie (1984) recommend to use the robust and approximately unbiased estimator

$$2\overline{\gamma}(\boldsymbol{h}) = \frac{\overline{Y}^4}{0.457 + \frac{0.494}{|N(\boldsymbol{h})|} + \frac{0.045}{|N(\boldsymbol{h})|^2}},$$

where

$$\overline{Y} = \frac{1}{\mid N(\boldsymbol{h}) \mid} \sum_{N(\boldsymbol{h})} \mid Z(\boldsymbol{s_i}) - Z(\boldsymbol{s_j}) \mid^{1/2},$$

as an estimator of $2\gamma(\boldsymbol{h})$. The expression "approximately unbiased estimator" is used, because $\overline{Y}$ is only approximately normally distributed and, thus, $2\overline{\gamma}(\boldsymbol{h})$ is not exactly unbiased. From the central limit theorem, the approximation improves as $\mid N(\boldsymbol{h}) \mid \to \infty$.

The following generalization of the variogram, the cross variogram

$$2\gamma_{lm}(\boldsymbol{h}) = 2\gamma_{lm}(\boldsymbol{s_1} - \boldsymbol{s_2}) = Var(Z_l(\boldsymbol{s_1}) - Z_m(\boldsymbol{s_2})), \quad \boldsymbol{s_1}, \boldsymbol{s_2} \in I\!R^d,$$

has been introduced by Clark, Basinger & Harper (1989). $Z_l(\boldsymbol{s_1})$ and $Z_m(\boldsymbol{s_2})$ $(l, m \leq k)$ are any two variables of a k–dimensional data set. It has been shown in Ver Hoef & Cressie (1993, 1994) that this generalization of the variogram should be preferred to other possible generalizations that can be found in the literature. Here we assume that the processes $\{Z_l(\cdot)\}$ and $\{Z_m(\cdot)\}$ have zero mean and unit variance. A natural estimator of $2\gamma_{lm}(\boldsymbol{h})$ is

$$2\hat{\gamma}_{lm}(\boldsymbol{h}) = \frac{1}{\mid N(\boldsymbol{h}) \mid} \sum_{N(\boldsymbol{h})} (Z_l(\boldsymbol{s_i}) - Z_m(\boldsymbol{s_j}))^2, \quad \boldsymbol{h} \in I\!R^d,$$

assuming that $\{Z_l(\boldsymbol{s_i})\}$ and $\{Z_m(\boldsymbol{s_j})\}$ have been first standardized such that $\overline{Z}_l = \overline{Z}_m = 0$ and $s_l^2 = s_m^2 = 1$. Similar to the variogram, it can be shown that

$$2\overline{\gamma}_{lm}(\boldsymbol{h}) = \frac{\overline{Y}_{lm}^4}{0.457 + \frac{0.494}{|N(\boldsymbol{h})|} + \frac{0.045}{|N(\boldsymbol{h})|^2}},$$

where

$$\overline{Y}_{lm} = \frac{1}{\mid N(\boldsymbol{h}) \mid} \sum_{N(\boldsymbol{h})} \mid Z_l(\boldsymbol{s_i}) - Z_m(\boldsymbol{s_j}) \mid^{1/2},$$

is a robust and approximately unbiased estimator of $2\gamma_{lm}(\boldsymbol{h})$.

In practice, a (cross) variogram–cloud plot is often examined before fitting a (cross) variogram model. We will discuss issues related to (cross) variogram–cloud plots in the remainder of this section.

5.2 XGobi's Data Structures

XGobi's data structures for the variogram–cloud link, the spatially lagged scatterplot link, and the multivariate variogram–cloud link are almost identical. We will explain them based on the variogram–cloud link. This link allows to display variogram–cloud plots and cross variogram–cloud plots while the multivariate variogram–cloud link allows to display back–to–back cross variogram–cloud plots. A small example in Figure 4 is an illustration of data used for the variogram–cloud link.

Originally, we have an array of *Initial Data* passed from ArcView to XGobi. For illustration purposes, we will take the number of points, n, to be 3. x and y are the coordinates of each point (we will call this point $\boldsymbol{s_i}, i = 1, \ldots, n$) in the 2–dimensional plane. Z_1 and Z_2 (in the general case $Z_1, \ldots, Z_k$) are the measures taken at point $\boldsymbol{s_i} = (x_i, y_i)$. They may be the same variable taken at different times, or they may be different variables. Parameter d, the cutoff distance, is also passed from ArcView. In this example, d is 1.1.

The *Array of Pairs* is the main data structure used to support this part of the link. It consists of records corresponding to each pair $(\boldsymbol{s_i}, \boldsymbol{s_j})$ of points satisfying the cutoff condition $\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \leq d$. Each record consists of the point numbers (i and j), the angle between $\boldsymbol{s_j} - \boldsymbol{s_i}$ and the horizontal axis measured in degrees (A), the set $\gamma_{lm}, l, m = 1, \ldots, k$, where $\gamma_{lm}(\boldsymbol{s_i}, \boldsymbol{s_j}) = \mid (Z_l(\boldsymbol{s_i}) - Z_m(\boldsymbol{s_j})) \mid^{1/2}$, and $D(\boldsymbol{s_i}, \boldsymbol{s_j}) = \sqrt{\sum_{l=1}^{k} (Z_l(\boldsymbol{s_i}) - Z_l(\boldsymbol{s_j}))^2}$. It should be noted that variogram–clouds relate to γ_{ii}, cross variogram–clouds relate to γ_{ij}, $i \neq j$, and multivariate variogram–clouds relate to γ_{ij} back to back with γ_{ji}, $i \neq j$. The latter name intrinsically contains the word "cross" since γ_{ii} back to back with γ_{ii} would be nothing else but a mirror image of the positive x–axis mapped onto the negative x–axis.

It is for the reason mentioned in Section 5.1 that we use $\gamma_{lm}(\boldsymbol{s_i}, \boldsymbol{s_j})$ in the variogram–cloud link since it is the major part of the robust and approximately

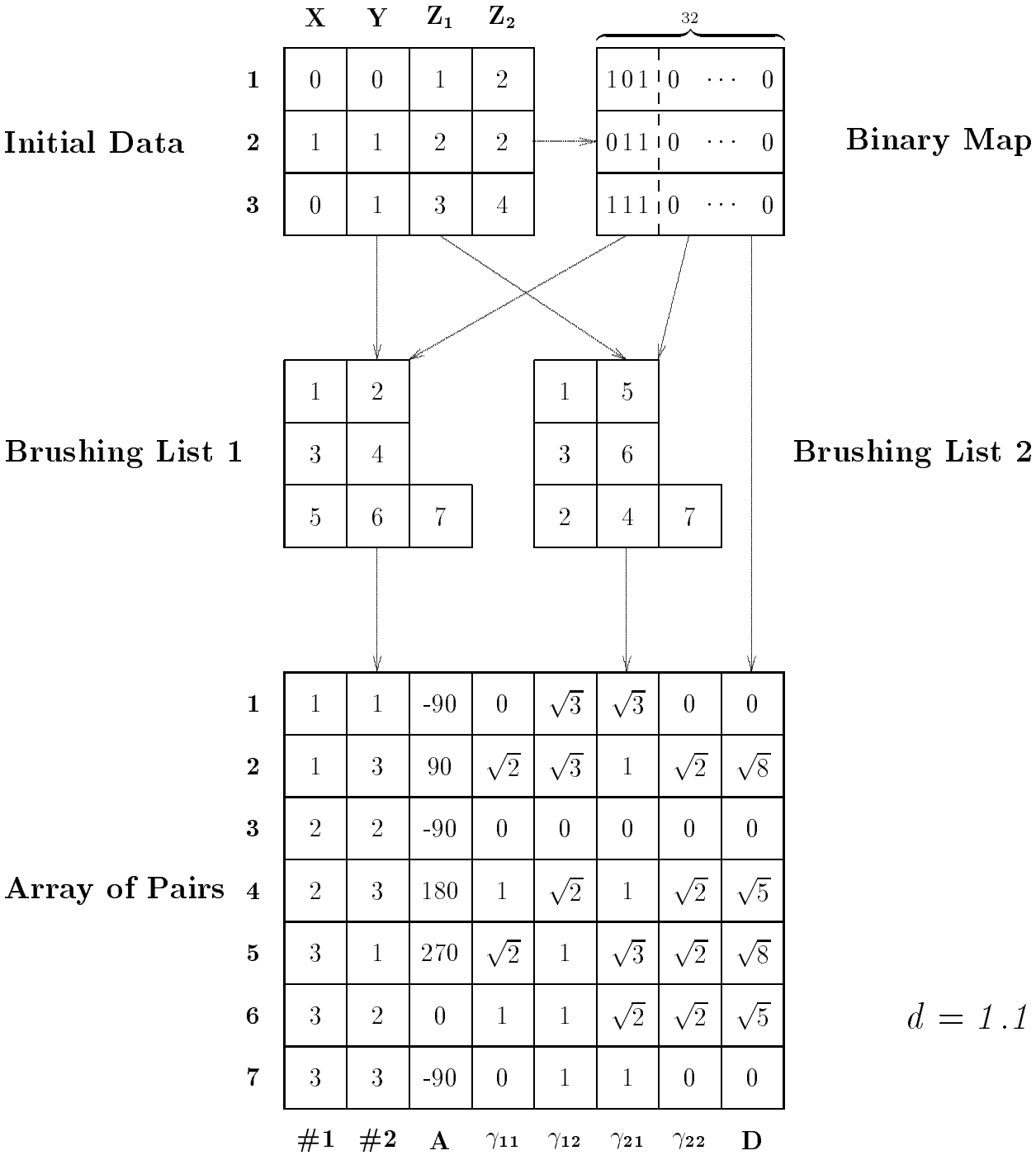

Figure 4: Data Structures for the Variogram–Cloud Link.

unbiased estimator $2\overline{\gamma}_{lm}(\boldsymbol{h})$ for the (cross) variogram. In many situations, we will consider a plot of $\gamma_{lm}(\boldsymbol{s_i}, \boldsymbol{s_j})$ versus $d(\boldsymbol{s_i}, \boldsymbol{s_j}) = || \boldsymbol{s_i} - \boldsymbol{s_j} || = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$ (given in Figure 2) to assess the spatial dependence before fitting a (cross) vari-

ogram model.

The value -90 is assigned to A for identical points ($i = j$), and the value -45 is assigned to A for observations from the same point ($\boldsymbol{s_i} = \boldsymbol{s_j}, i \neq j$). The record also contains $\sin(A)$ and $\cos(A)$ which are useful for brushing in the arbitrary angle class, but they are not displayed in Figure 4.

All other arrays shown in Figure 4 are designed to create and support the link between points in XGobi and ArcView and are required to do brushing in both directions. A *Binary Map* is created upon initialization of the link. It is a 2–dimensional array of size $n \cdot \text{ceil}(n/32)$. A "1" at the jth bit in the ith row indicates that the distance between $\boldsymbol{s_i}$ and $\boldsymbol{s_j}$ is smaller than d (e. g., it is 1 between $\boldsymbol{s_1}$ and $\boldsymbol{s_3}$). A "0" indicates that the distance between these two points is bigger than d (e. g., it is $\sqrt{2}$ between $\boldsymbol{s_1}$ and $\boldsymbol{s_2}$).

Information about 32 pairs of points is packed into one unsigned integer. Simultaneously with the creation of *Binary Map*, we count the number of points that lie within the cutoff distance for every initial point. After we finish looking at *Initial Data*, we know exactly how many pairs with point $\boldsymbol{s_i}$ are going to be in *Array of Pairs* for every $i = 1, \ldots, n$. This allows us to allocate memory for *Array of Pairs* and *Brushing List 1* and *2*. *Brushing List 1* corresponds to all pairs for which $\boldsymbol{s_i}$ is the first point in the pair (e. g., 1 and 2 for $\boldsymbol{s_1}$), and *Brushing List 2* corresponds to all pairs for which $\boldsymbol{s_i}$ is the second point in the pair (e. g., 1 and 5 for $\boldsymbol{s_1}$).

Now we describe the logical part overlaying these data structures. Using *Binary Map* we pick proper pairs of points, add information to *Brushing List 1* and *2*, and fill the record for this pair in the *Array of Pairs*. As for *Brushing List 1* and *2*, we know at each step exactly how many "boxes" are filled (there exists a special array of counters for those). So, we place the number of the record in the first free positions of *Brushing List 1* and *2* and increment the corresponding counters by one.

After we have gone through *Binary Map*, we have *Array of Pairs* and *Brushing List 1* and *2* completely filled. Now we can brush points in two directions. If we take a record from *Array of Pairs* we immediately know which points this entry consists of. If we pick a point in the array of *Initial Data*, say number i, the ith line of *Brushing List 1 (2)* contains all numbers of the pairs that contain this point i as first (second) component.

Finally, we want to comment on the efficiency of our data structures. Assume there are $n = 128$ points in ArcView. This might yield up to $128^2 = 16384$ points in XGobi. However, XGobi is suited best for small to medium size data sets. It might be desirable to have a cutoff distance d selected such that only 1024, i. e., 1/16th of the possible number of points, are displayed in XGobi. The additional memory required for *Binary Map* ($128^2/32 = 512$ unsigned integers) can be neglected in comparison to the gain of memory for not having allocated 16384 records of *Array of Pairs*. Each line of *Brushing List 1* and *2* is reasonably small ($1024/128 = 8$ in average) to allow efficient handling and brushing of points.

5.3 Linked Brushing

Linked brushing between ArcView and versions of XGobi that display variogram–cloud plots, spatially lagged scatterplots, and multivariate variogram–cloud plots works slightly different than the standard linked brushing. There, exactly one point in one application is related to one point in the other application. For each point brushed, the related point is brushed as well. Here, for every point in XGobi, there are two related points in ArcView. Actually, there might be up to n^2 points in XGobi even if there are only n points in ArcView. We assume we brush only one point at a time in either application. Then, we make use of the following strategy for linked brushing:

Brushing in XGobi: Mark the point in XGobi with the selected color, glyph, and size. Determine in XGobi which are the two related points (using *Array of Pairs* described in Section 5.2) and pass this information to ArcView. Connect these points in ArcView with a line of the selected color (provided by XGobi).

Brushing in ArcView: This can be described as a three phase process, where ArcView, XGobi, and again ArcView proceed. First, brush the point in the ArcView view (nothing visible happens) and pass the information, which point has been brushed and which color, glyph, and size have been selected, to XGobi. Then, in XGobi, determine all points that are related to this point (using *Brushing List 1* and *2* described in Section 5.2). There might be up to $2n-1$ such points. For each of these points, assume it actually has been brushed in XGobi. Follow the instructions on brushing in XGobi, using the color, glyph, and size provided by ArcView. Finally, this will result in lines drawn in ArcView between the selected point and all points that are not farther away than the cutoff distance d.

6 Security and Concurrency

In our bidirectional link between ArcView (2.0, 2.1, and 3.0) and XGobi, security and concurrency are important issues that had to be addressed. Unfortunately, an ArcView (2.0, 2.1, and 3.0) RPC server supports only one function: "1 – script execution – executes the given script and returns a string representation of the last object referenced or produced during the execution of the script" (Environmental Systems Research Institute, Inc., 1996). A string containing an Avenue script is required as an argument for the script execution. This string can be anything from a single Avenue statement through the complete text of an Avenue script. Since ArcView 2.0 only supported null authentication, there was no protection against the manipulation or deletion of entire ArcView 2.0 data bases or any of the analyst's files by an external user connected to the running ArcView 2.0 RPC server, hence a total lack of data security.

RPCs and existing security issues have been described in the technical literature, e. g., Corbin (1991). Unix authentication causes the transmission of

additional fields (such as a time stamp, the name of the local host, the client's effective user and group IDs) with every RPC request. Data Encryption Standard (DES) authentication promotes secure exchange of data in a standard fashion since it encrypts/decrypts data through public and private keys associated with the effective user ID of the calling process. For both types of authentication only valid requests are granted.

ArcView 2.1 and 3.0 provide at least Unix authentication. In our link between ArcView (2.1 and 3.0) and XGobi we make use of this authentication mechanism and process only those requests where local host and user ID match between ArcView and XGobi (which is started from within ArcView — thus IDs will match for valid requests). Unfortunately, there is no verifier for Unix authentication, thus the previously mentioned credentials are still easy to fake. Therefore, we suggest that ESRI provides an RPC mechanism for DES authentication in addition to the current null and Unix authentication for future releases of ArcView.

The concurrency issue (What happens if at least two XGobi processes provide different update information to ArcView at the same time?) has been solved by a built–in feature of XGobi. Only one XGobi process, the one invoked from within ArcView, provides the features of an RPC server and RPC client. The other XGobi processes that have been cloned are "regular" XGobi processes that do not support RPCs. All XGobi processes (the one invoked from within ArcView and the cloned ones) communicate to each other through the production and consumption of *XEvents*. According to Nye (1992), p. 38, "an event is a packet of information that is generated by the [XWindow] server when certain actions occur and is queued for later use by the [XWindow] client". Usually, events are created upon pressing or releasing a mouse button, window mapping or unmapping, and crossing a window boundary with the mouse. However, they can also be created by any of the (XWindow) client programs. In the context of the X Window System, one uses the term XEvent.

When points are brushed or subsetted, an appropriate XEvent is generated. Even if one XGobi process is delayed for a while, it sequentially processes the XEvents that have been generated by other XGobi processes. Whenever the XGobi invoked from within ArcView processes an XEvent related to brushing or subsetting, it requests ArcView to update its view. Due to this one–to–one link and the serialization of the XEvents, it was not necessary to consider additional steps to solve the concurrency issue — it was already solved internally.

7 Future Directions

Currently, our work has been stopped as far as it involves the incorporation of additional methods of spatial statistics into the link. Nevertheless, since the generic work required to link ArcView and XGobi can be adapted to any graphical method of spatial statistics which can be displayed within XGobi itself, it is easy to add such methods whenever desired.

Recently, we started work on modifications of XGobi, e. g., different types of smoothers that are helpful, for example, for the variogram–cloud plot and for picking up large–scale variation in the data. Missing values, another recent addition to XGobi, are supported by the link as well. Some additional work has to be done to finish these extensions.

One of the current limitations of the link is the number of points that can be handled. No severe problems arise for the multivariate attribute data and SCDF link since in both cases, the link reasaonably supports as many points as can be managed within XGobi, typically a few thousand. However, severe problems arise for the variogram–cloud plot, spatially lagged scatterplot, and multivariate variogram–cloud plot if the selected cutoff distance results in ten thousands or even hundred thousands of points in XGobi. While XGobi does still support ten thousands of points, the RPC mechanism does not. The communication and encoding/decoding of data into strings that have to be passed between ArcView and XGobi requires that much time that no reasonable work is possible on any of the hardware platforms we currently use. It could also happen that linked brushing between ArcView and XGobi works only partially or even fails completely when multiple XGobi processes are involved. There is no immediate solution of this problem due to the restrictions imposed by the ArcView 3.0 RPC mechanism. However, it has been indicated by ESRI that future versions of ArcView will be able to process C code. This would allow us to set up a faster RPC mechanism where compressed data is passed between ArcView and XGobi or even to develop a link where both programs make use of the same shared memory and no data has to be passed at all.

We plan to implement a new type of linked brushing, called "hierarchical" linked brushing. This linked brushing is required for a future environment where ArcView communicates with several XGobis, each of them displaying a different feature of the link, e. g., one XGobi and a cloned child for the attribute data, one XGobi for the spatial cumulative distribution functions, and one XGobi for the variogram–cloud plot. The current linked brushing in XGobi does not reasonably support such an environment. Extensions to our Avenue code within ArcView are required as well to allow the communication with multiple XGobis through RPCs.

A first description of an extension of the link towards another statistical package, XploRe (Härdle, Klinke & Turlach, 1995), is given in Symanzik, Kötter, Schmelzer, Klinke, Cook & Swayne (1997a). In the future, other packages such as S (Becker, Chambers & Wilks, 1988) might be linked as well. It might be advisable to frequently check the WWW page mentioned at the end of Section 1 for new releases of the software and for updates on recently detected or fixed bugs in the code.

Acknowledgements

Research related to this article was supported by an EPA grant under cooperative agreement # CR822919–01–0. The article has not been subjected to the review of the EPA and thus does not necessarily reflect the view of the agency and no official endorsement should be inferred. Symanzik's research was also partially supported by a German "DAAD–Doktorandenstipendium aus Mitteln des zweiten Hochschulsonderprogramms". We want to thank Noel Cressie for the helpful discussion of the theory related to the software. Thanks are also due to Nicholas Lewin for his help in upgrading the software from ArcView 2.1 to ArcView 3.0.

References

Anselin, L. and Bao, S. (1996). Exploratory Spatial Data Analysis Linking SpaceStat and ArcView. Technical Report 9618, West Virginia University, Morgantown, WV.

Asimov, D. (1985). The Grand Tour: A Tool for Viewing Multidimensional Data. *SIAM Journal Scientific and Statistical Computing*, 6(1):128–143.

Becker, R., Chambers, J., and Wilks, A. (1988). *The New S Language — A Programming Environment for Data Analysis and Graphics*. Wadsworth and Brooks/Cole, Pacific Grove, CA.

Bradley, R. and Haslett, J. (1992). Interactive Graphics for the Exploratory Analysis of Spatial Data — The Interactive Variogram Cloud. In Dowd, P. A. and Royer, J. J., editors, *2nd CODATA Conference on Geomathematics and Geostatistics, Sci. de la Terre, Ser. Inf.*, volume 31, pages 373–386, Nancy.

Brunsdon, C. and Charlton, M. (1996). Developing an Exploratory Spatial Analysis System in XLisp–Stat. In Parker, D., editor, *Innovations in GIS 3*, pages 135–145, London, U.K. Taylor & Francis.

Buja, A. and Asimov, D. (1986). Grand Tour Methods: An Outline. *Computing Science and Statistics*, 17:63–67.

Buja, A., Cook, D., and Swayne, D. F. (1996). Interactive High–Dimensional Data Visualization. *Journal of Computational and Graphical Statistics*, 5(1):78–99.

Carr, D. B., Littlefield, R. J., Nicholson, W. L., and Littlefield, J. S. (1987). Scatterplot Matrix Techniques for Large N. *Journal of the American Statistical Association*, 82(398):424–436.

Carr, D. B., Olsen, A. R., and White, D. (1992). Hexagon Mosaic Maps for Displays of Univariate and Bivariate Geographical Data. *Cartography and Geographic Information Systems*, 19(4):228–236, 271.

Clark, I., Basinger, K. L., and Harper, W. V. (1989). MUCK — A Novel Approach to Co–Kriging. In Buxton, B. E., editor, *Proceedings of the Conference on Geostatistical, Sensitivity, and Uncertainty Methods for Ground–Water Flow and Radionuclide Transport Modeling*, pages 473–493, Columbus, OH. Battelle Press.

Cook, D., Cressie, N., Majure, J., and Symanzik, J. (1994). Some Dynamic Graphics for Spatial Data (with Multiple Attributes) in a GIS. In Dutter, R. and Grossmann, W., editors, *COMPSTAT 1994: Proceedings in Computational Statistics*, pages 105–119, Heidelberg. Physica–Verlag.

Cook, D., Majure, J. J., Symanzik, J., and Cressie, N. (1996). Dynamic Graphics in a GIS: Exploring and Analyzing Multivariate Spatial Data Using Linked Software. *Computational Statistics*, 11(4):467–480.

Cook, D., Symanzik, J., Majure, J. J., and Cressie, N. (1997). Dynamic Graphics in a GIS: More Examples Using Linked Software. *Computers and Geosciences*, Forthcoming: Contains live video footage, available at `http://www.public.iastate.edu/~dicook/compgeo.html`.

Corbin, J. R. (1991). *The Art of Distributed Applications: Programming Techniques for Remote Procedure Calls*. Springer, New York, Berlin, Heidelberg.

Cressie, N. (1984). Towards Resistant Geostatistics. In Verly, G., David, M., Journel, A. G., and Marechal, A., editors, *Geostatistics for Natural Resources Characterization, Part 1*, pages 21–44, Dordrecht. Reidel.

Cressie, N. and Hawkins, D. M. (1980). Robust Estimation of the Variogram: I. *Journal of the International Association for Mathematical Geology*, 12(2):115–125.

Cressie, N. A. C. (1993). *Statistics for Spatial Data (Revised Edition)*. Wiley, New York, NY.

DiBiase, D., Reeves, C., MacEachren, A. M., von Wyss, M., Krygier, J. B., Sloan, J. L., and Detweiler, M. C. (1994). Multivariate Display of Geographic Data: Applications in Earth System Science. In MacEachren, A. M. and Taylor, D. R. F., editors, *Visualization in Modern Cartography*, pages 287–312. Pergamon (Elsevier), Oxford, U.K.

Dykes, J. (1996). Dynamic Maps for Spatial Science: A Unified Approach to Cartographic Visualization. In Parker, D., editor, *Innovations in GIS 3*, pages 177–187, London, U.K. Taylor & Francis.

Environmental Systems Research Institute, Inc. (1996). *ArcView 3.0 On–Line Help: RPCServer Page*. Redlands, CA.

Goodchild, M. F., Haining, R. P., and Wise, S. (1992). Integrating GIS and Spatial Data Analysis: Problems and Possibilities. *International Journal of Geographical Information Systems*, 6(5):407–423.

Haining, R., Ma, J., and Wise, S. (1996). Design of a Software System for Interactive Spatial Statistical Analysis Linked to a GIS. *Computational Statistics*, 11(4):449–466.

Härdle, W., Klinke, S., and Turlach, B. A. (1995). *XploRe: An Interactive Statistical Computing Environment*. Springer, New York, Berlin, Heidelberg.

Haslett, J. and Bradley, R. (1991). Dynamic Graphics: Linked Points, Lines and Regions with Applications to Spatial Data Modelling. *Computing Science and Statistics*, 23:425–429.

Haslett, J., Bradley, R., Craig, P., Unwin, A., and Wills, G. (1991). Dynamic Graphics for Exploring Spatial Data with Application to Locating Global and Local Anomalies. *The American Statistician*, 45(3):234–242.

Hawkins, D. M. and Cressie, N. (1984). Robust Kriging — A Proposal. *Journal of the International Association for Mathematical Geology*, 16(1):3–18.

Klein, R. and Moreira, R. I. (1994). Exploratory Analysis of Agricultural Images via Dynamic Graphics. Technical Report 9/94, Laboratório Nacional de Computação Científica, Rio de Janeiro, Brazil.

Littman, M., Swayne, D. F., Dean, N., and Buja, A. (1992). Visualizing the Embedding of Objects in Euclidean Space. *Computing Science and Statistics*, 24:208–217.

MacDougall, E. B. (1992). Exploratory Analysis, Dynamic Statistical Visualization, and Geographic Information Systems. *Cartography and Geographic Information Systems*, 19(4):237–246.

Majure, J. J., Cook, D., Cressie, N., Kaiser, M., Lahiri, S., and Symanzik, J. (1995). Spatial CDF Estimation and Visualization with Applications to Forest Health Monitoring. ASA Statistical Graphics Video Lending Library (contact: dfs@research.att.com).

Majure, J. J., Cook, D., Cressie, N., Kaiser, M., Lahiri, S., and Symanzik, J. (1996a). Spatial CDF Estimation and Visualization with Applications to Forest Health Monitoring. *Computing Science and Statistics*, 27:93–101.

Majure, J. J., Cook, D., Symanzik, J., and Megretskaia, I. (1996b). An Interactive Environment for the Graphical Analysis of Spatial Data. ASA Statistical Graphics Video Lending Library (contact: dfs@research.att.com).

Majure, J. J. and Cressie, N. (1997). Dynamic Graphics for Exploring Spatial Dependence in Multivariate Spatial Data. *Geographical Systems*, 4:Forthcoming.

Majure, J. J., Cressie, N., Cook, D., and Symanzik, J. (1996c). GIS, Spatial Statistical Graphics, and Forest Health. In *Proceedings of the Third International Conference/Workshop on Integrating GIS and Environmental Modeling, Santa Fe, NM, January 21–26, 1996*, Santa Barbara, CA. National Center for Geographic Information and Analysis. CD and `http://www.ncgia.ucsb.edu/conf/SANTA_FE_CD-ROM/main.html`.

MathSoft (1996). *S+Gislink*. MathSoft, Inc., Seattle.

McDonald, J. A. and Willis, S. (1987). Use of the Grand Tour in Remote Sensing. ASA Statistical Graphics Video Lending Library (contact: dfs@research.att.com).

Monmonier, M. (1988). Geographical Representations in Statistical Graphics: A Conceptual Framework. In *ASA Proceedings of the Section on Statistical Graphics*, pages 1–10, Alexandria, VA. American Statistical Association.

Monmonier, M. (1989). Geographic Brushing: Enhancing Exploratory Analysis of the Scatterplot Matrix. *Geographical Analysis*, 21(1):81–84.

Nye, A. (1992). *Xlib Programming Manual (Third Edition, for Version 11 of the X Window System)*. O'Reilly & Associates, Inc., Sebastopol, CA.

Openshaw, S. and Perrée, T. (1996). User–Centred Intelligent Spatial Analysis of Point Data. In Parker, D., editor, *Innovations in GIS 3*, pages 119–134, London, U.K. Taylor & Francis.

Rossi, R. E., Mulla, D. J., Journel, A. G., and Franz, E. H. (1992). Geostatistical Tools for Modeling and Interpreting Ecological Spatial Dependence. *Ecological Monographs*, 62:277–314.

Scott, L. M. (1994). Identification of a GIS Attribute Error Using Exploratory Data Analysis. *The Professional Geographer*, 46(3):378–386.

Stevens, W. R. (1990). *UNIX Network Programming*. Prentice–Hall, Englewood Cliffs, NJ.

Swayne, D. F., Cook, D., and Buja, A. (1991). XGobi: Interactive Dynamic Graphics in the X Window System with a Link to S. In *ASA Proceedings of the Section on Statistical Graphics*, pages 1–8, Alexandria, VA. American Statistical Association.

Symanzik, J., Kötter, T., Schmelzer, S., Klinke, S., Cook, D., and Swayne, D. (1997a). Spatial Data Analysis in the Dynamically Linked ArcView/XGobi/XploRe Environment. *Computing Science and Statistics*, 29:Forthcoming.

Symanzik, J., Majure, J., Cook, D., and Cressie, N. (1994). Dynamic Graphics in a GIS: A Link between ARC/INFO and XGobi. *Computing Science and Statistics*, 26:431–435.

Symanzik, J., Majure, J. J., and Cook, D. (1995). Dynamic Graphics in a GIS: Analyzing and Exploring Multivariate Spatial Data. ASA Statistical Graphics Video Lending Library (contact: dfs@research.att.com).

Symanzik, J., Majure, J. J., and Cook, D. (1996a). Dynamic Graphics in a GIS: A Bidirectional Link between ArcView 2.0 and XGobi. *Computing Science and Statistics*, 27:299–303.

Symanzik, J., Majure, J. J., and Cook, D. (1997b). Dynamic Graphics in a GIS: A Bidirectional Link between ArcView 2.1 and XGobi — An Update. In *Second World Conference of the International Association for Statistical Computing (IASC), Pasadena, California, USA, February 19–22, 1997*. Forthcoming.

Symanzik, J., Majure, J. J., and Cook, D. (1997c). The Linked ArcView 2.1 and XGobi Environment — GIS, Dynamic Statistical Graphics, and Spatial Data. In Shekhar, S. and Bergougnoux, P., editors, *Proceedings of the Fourth ACM Workshop on Advances in Geographic Information Systems, Rockville, Maryland, November 15–16, 1996*, pages 147–154, New York, NY. ACM.

Symanzik, J., Megretskaia, I., Majure, J. J., and Cook, D. (1996b). Implementation Issues of Variogram Cloud Plots and Spatially Lagged Scatterplots in the Linked ArcView 2.1 and XGobi Environment. *Computing Science and Statistics*, 28:Forthcoming.

Unwin, A., Wills, G., and Haslett, J. (1990). REGARD — Graphical Analysis of Regional Data. In *ASA Proceedings of the Section on Statistical Graphics*, pages 36–41, Alexandria, VA. American Statistical Association.

Ver Hoef, J. M. and Cressie, N. (1993). Multivariable Spatial Prediction. *Mathematical Geology*, 25(2):219–240.

Ver Hoef, J. M. and Cressie, N. (1994). Errata: Multivariable Spatial Prediction. *Mathematical Geology*, 26(2):273–275.