Paper 213-27

Applied Geostatistics with JMP®

Mauromoustakos A. and K. Thompson, U of Arkansas, Fayetteville, AR

ABSTRACT

Presentation of basic visual geostatistical techniques using JMP V4 software will include empirical variogram calculations, along with interactive graphing and variogram modeling. The JSL script presented here uses the powerful new scripting language JSL to code the routines and enhances existing JMP contour platform with graphing scripts suited to provide better visualization interactivity and understanding of spatially correlated observations.

INTRODUCTION

A very simple language called JMP Scripting Language, or JSL scripts JMP. You may not need to ever learn JSL, because almost every feature in the product is accessible through a direct user interface, as well as through the scripting language. Even if you use JSL, you can usually get JMP to write the scripts for you rather than typing them in yourself. JSL will be most useful to the power user that wants to extend JMP past its normal operating realm, or for the production user who wants to automate a regularly scheduled analysis. We will put the typical applied geostatistical researcher in the above-defined category of a power user. JSL is used in many places in JMP internally:

- Column Formulas are implemented internally in JSL.
- Platforms are launched using JSL internally.
- Platforms are interactively modified using JSL internally.
- Some graphics are performed through JSL.

OBJECTIVES

This paper will show examples of geostatistical exploration of data done with JSL scripts with a goal to perform analysis similar to that provided by PROC VARIOGRAM found in SAS/STAT version 8 but in a more visual and interactive implementation. This is an attempt to build on the strengths of JMP as a discovery tool and expose to casual JMP users to the strengths of JSL (a hybrid between SAS Macro Language and IML that look more than C and Java code than SAS® code). We hope that the program discussed here demonstrate the flexibility and power of JSL language as toolkit to perform needed calculations that can aid researchers with exploring and modeling geostatistical relations.

RUNNING A JSL SCRIPT IN JMP

To type and run a script you need to:

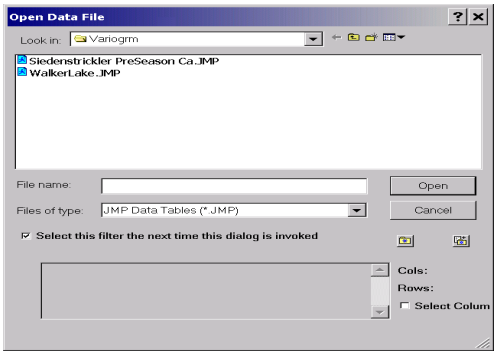
- Start JMP
- In the JMP Starter window, click Open Script
- In the Open JSL Script Dialog window select the Variogram.JSL, a script program (provided by the authors)
- Click CNTL+R to execute the script.

We will not discuss (review) here the "theory/practice of applied Geostatistics" that a reader can pick up in any of the references provided below or in the more extensive reference sections of those books/papers found therein. Instead we will assume that the readers will first familiarize themselves with those practices available for geostatistical data (at least the ones available in SAS) before they will have an appreciation/understanding and need to try and play around with the tools that are discussed here. We will make publicly available upon request the progrmas used here for exploring and modeling spatially correlated data. The screen captures provided here will help as a tutorial that attempts to showcase the functionality of the JMP/JSL as a geostatistical exploration tool.

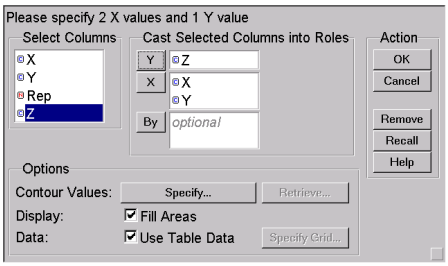
VARIOGRAM SCRIPT

This script is used to perform the calculations needed for statistics required to graph and model semi-variogram(s) for a spatially distributed random variable. This is something that is routinely computed and plotted to investigate and model spatially correlated data. Similar calculations are performed when using PROC VARIOGRAM available in recent releases of SAS/STAT.

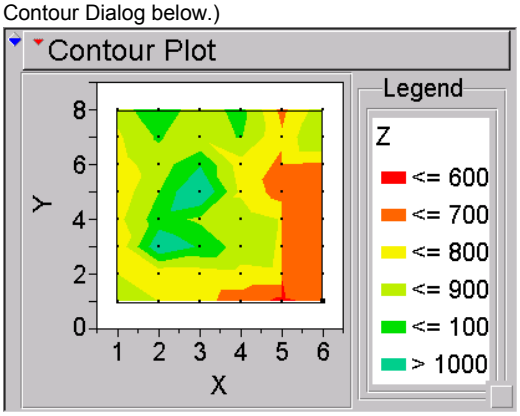
When first running variogram.jsl it opens a dialog box asking you for the current dataset that contains data that has to include XY coordinates for the observations since the data considered here are always referenced by their field locations.



The data in used here shown above by the name (Siedenstricker Preseason Ca) contain pre season Calcium values on 8 by 6 one meter grid from a farm in South East Arkansas. We will display the data here using JMP's Contour graph platform to illustrate the only already available tool to display geo-referenced data in the current production version. To show a contour map of data you need to supply the required inputs shown below and click OK.

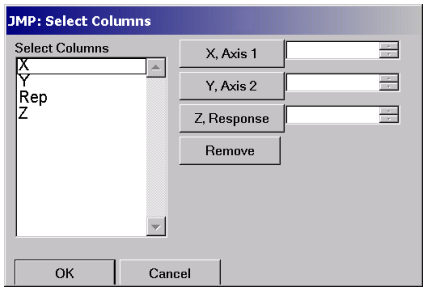


The resulted contour plot displays the Ca values annotated by their location. (Selecting the option Show data Points in the Contour Dialog below.)

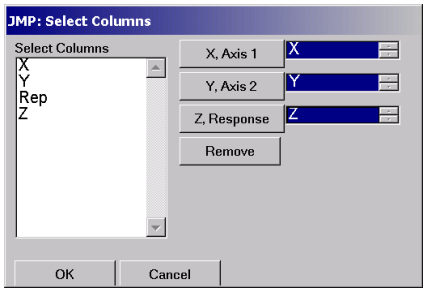


These graphs are the furthest that a user currently can get with the current version of JMP in order to better understand and visualize geostatistical data. What we are shown next is a list of screen captures with calculations and displays that a knowledgeable applied geo-statistician will need to better understand and model spatial correlation with JMP.

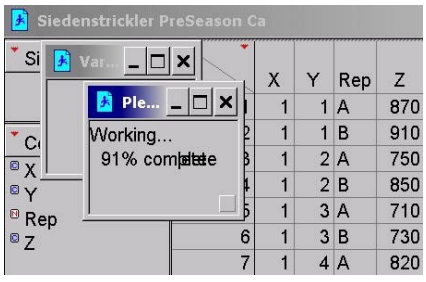
The user provided necessary input after selecting the data table includes the identification of the response Z along with the columns names that provides the required XY coordinates by specifying X and Y-axes. Note that the columns names for this example are chosen so that the user does not wonder what is the response variable (Z) and which columns contain the easting/northing coordinates that conveniently labeled here as "X" and "Y". The first dialog window after executing the script appears below.



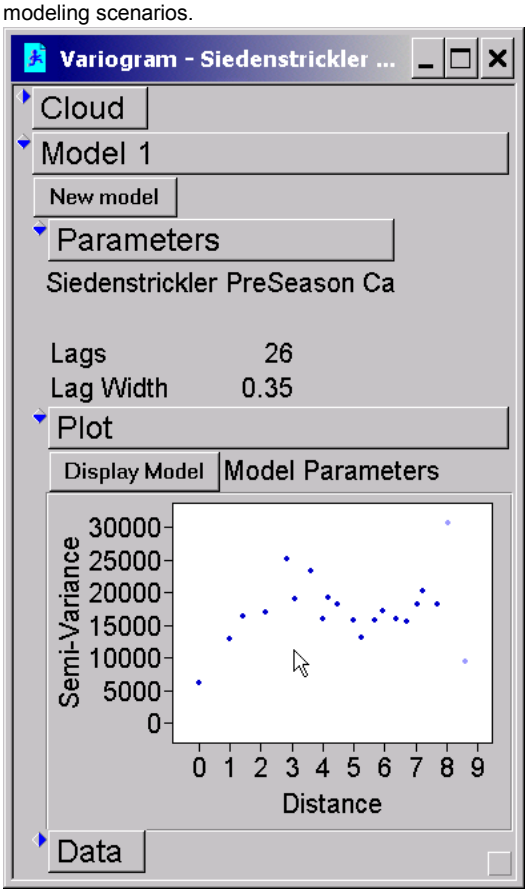
Then the user is expected to fill the required input fields similar to the way shown in the capture below.



After clicking the OK button the user then sees a dialog window with an advancing counter as a percent of calculations complete advances as shown below. We want to warn the user that these calculations can take some time depending on the size of the input dataset.

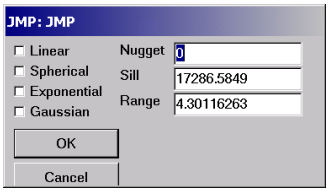


When the calculations finish the user sees the following dialog output with the isotropic semi-variogram in the bottom of that output. Note here that the furthest two semi-variances associated with the two largest distances have different color and when clicked by the user they display the number of pairs that the specific semi-variance is based on. Since most literature seems to agree that semi-variances based on a smaller than "an adequate number of pairs" (typically suggested as 30 pairs) is troublesome we warn the user visually that he might want to exclude those two observations semi-variances in further modeling scenarios.

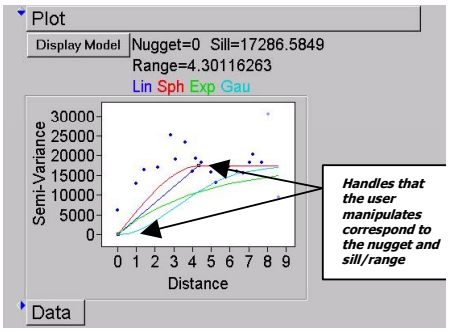


VARIOGRAM MODELING (ISOTROPIC CASE)

Important statistics (independent of direction and based only on different lag distances) are usually investigated in the so-called isotropic case scenario. The JSL code provided below calculates the statistics for "all possible distance and angle classes". Code not shown in the paper collapses the variogram cloud into angle and distance lag classes and will be discussed later when considering anisotropy. By clicking on Display model above you get a dialog box with the four most common theoretical geostatistical models shown below with script chosen default values defining the handles that the user can manipulate.

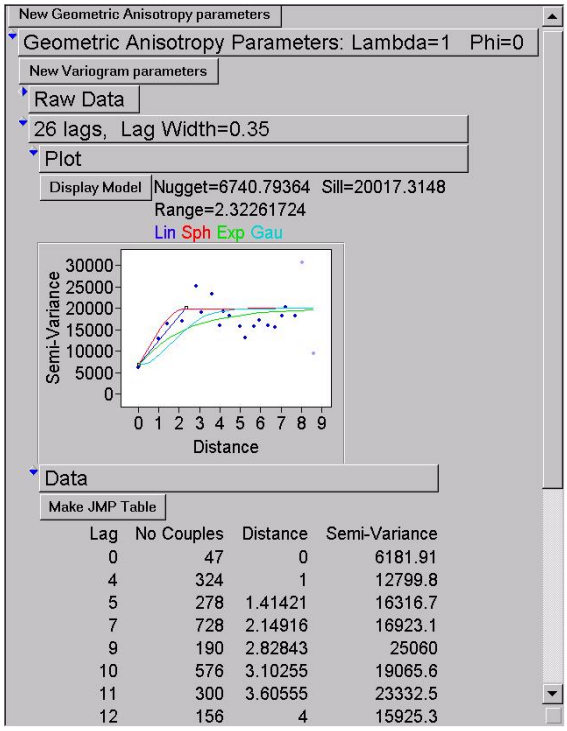


If you check all four models above and click OK you get

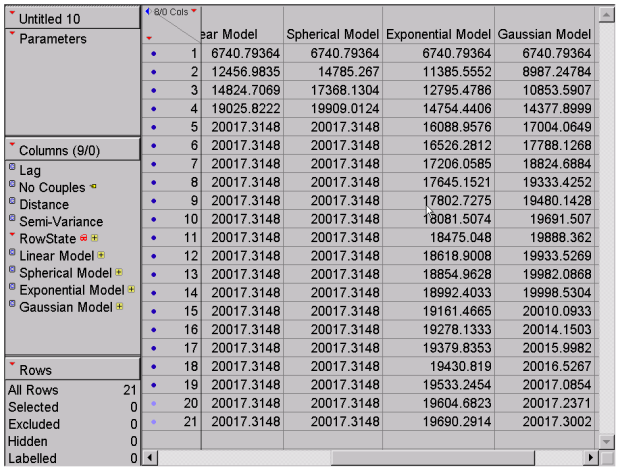


The two handles provided courtesy of the graphing ability of JSL (shown above in the annotated capture below) allow the user to visual fit the various models by altering the key nonlinear parameters. Better inputs are usually needed to fine-tune the parameters for JMP nonlinear platform. This assures that the nonlinear fitting of the various nonlinear models will converge even faster. Although many practitioners might be happy with the visual fitting/adjustment of the parameters most would chose to further improve their subjective choices by utilizing JMP nonlinear fit platform. The advantages to proceeding with the nonlinear fits is that the user can further rely on provided statistics such as the RMSE that can be used as indicators of which of the four theoretical semi-variogram fits best the empirical semi-variogram that leads to the best choice of parameters.

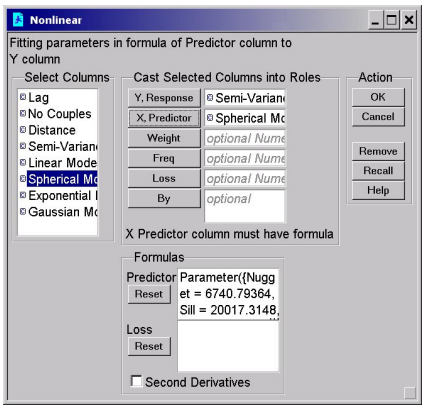
Note that the screen below shows the results of our fine-tuning after we drag the handles to what we thought are more appropriate choices for *nugget* and *sill/range*



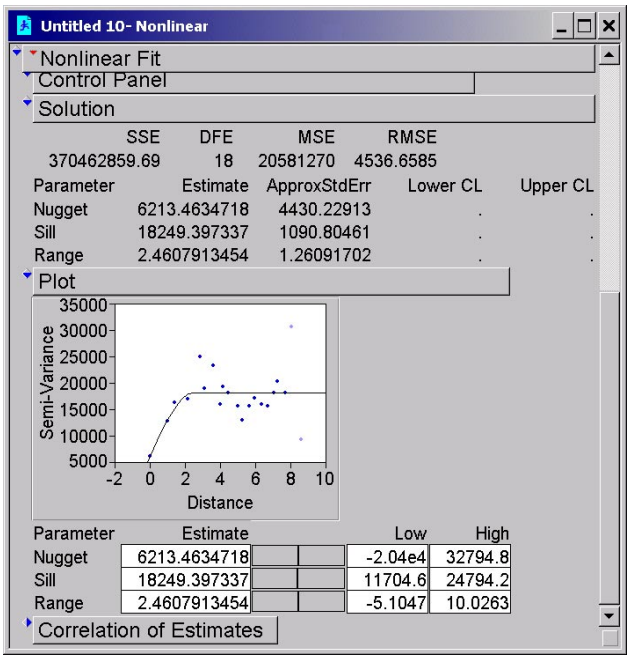
Clicking on the Make JMP Table c\button we get the following JMP table that you will note has already build the formulas for the four selected nonlinear models with the appropriate choices for initial values to the important nonlinear parameters.



One can then utilize JMP's nonlinear platform to fit and compare any or all four nonlinear models (after he decides if he/she wants to use all the data). Below we show the steps of fitting the spherical model.



The output after clicking OK above and again GO on the subsequent dialog is shown below



VARIOCLOUD SCRIPT (PART OF THE VARIOGRAM JSL CODE APPLICABLE FOR ALL SITUATIONS)

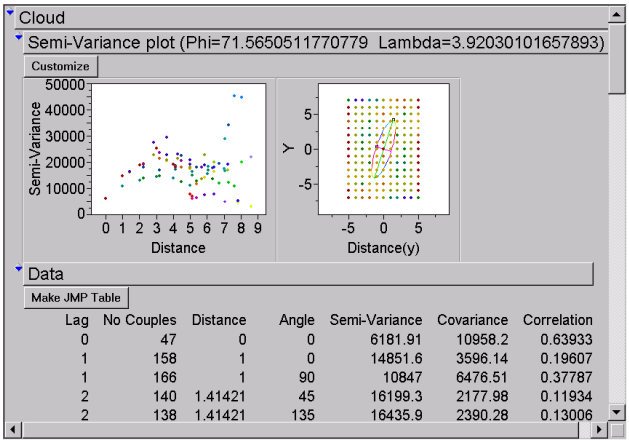
```
// Calculate Variogram Cloud
// Called by VariogramCloud
::CalculateVariogramCloud=Expr(
// Please Wait
::dbPleaseWait=NewWindow( "Please Wait" ,
       VListBox(
       TextBox("Working..."),
       ::dbPleaseWaitCounter=TextBox("
Initializing")
       )
       );
::dbPleaseWait<<SizeWindow(150,100):
::dbPleaseWait<<Reshow;
Wait(1);

// Initialize results matrices
::fuzz=1e-5;
```

```
::matCloudDist      = j( 1, 1, 0);
::matCloudAngle     = j( 1, 1, 0);
::matCloudHx        = j( 1, 1, 0);
::matCloudHy        = j( 1, 1, 0);
::matCloudN         = j( 1, 1, 0);
::matCloudZMeanI    = j( 1, 1, 0);
::matCloudZMeanJ    = j( 1, 1, 0);
::matCloudZVarI     = j( 1, 1, 0);
::matCloudZVarJ     = j( 1, 1, 0);
::matCloudZCovIJ    = j( 1, 1, 0);
::matCloudGaussian  = j( 1, 1, 0);
::matCloudRobust    = j( 1, 1, 0);

// ToDo: allow for Robust Estimation
::k=0;
::kmax=NRow(::matDataX)*(NRow(::matDataX)-1);
for( ::i=1, ::i<=NRow(::matDataX), ::i++,
::dbPleaseWaitCounter<<SetText( "  " ||
Char(100*::k/::kmax,3,0) || "% complete" );
::dbPleaseWaitCounter<<Reshow ;
for( ::j=1, ::j<=NRow(::matDataX), j++,
if(  ::i !=::j,
    ::k++;
    ::dx= ::matDataX[::i] - ::matDataX[j];
    ::dy= ::matDataY[::i] - ::matDataY[j];
    ::dz= ::matDataZ[::i] - ::matDataZ[::j];
    ::dist=Sqrt( ::dx*::dx +  ::dy*::dy) ;
    if(::dist==0, ::angle=0,
                  ::ACos=ArCos(::dx/::dist) *
180 / pi() ;
                  ::angle = ::ACos*(::dy>=0)
+ Mod(360-::ACos,360)*(::dy<0)
      );
    ::zi  = ::matDataZ[::i];
    ::zj  = ::matDataZ[::j];
    ::zii = ::matDataZ[::i] :*
::matDataZ[::i];
    ::zjj = ::matDataZ[::j] :*
::matDataZ[::j];
    ::zij = ::matDataZ[::i] :*
::matDataZ[::j];
    ::dvar = 0.5*::dz*dz ;
// Gaussian

    // Classify by distance and angle
    // Bug in Ver 4.0.4  And() returns
Unknown if arguments are single element
matrices
    // Bug in Ver 4.0.4  EMult() returns a
scalar instead of a matrix if arguments are
single element matrices
   ::indx=Loc( Matrix((::matCloudHx ==::dx)
:* (::matCloudHy ==::dy) ));

    if( NRow(::indx)==0,
      ::matCloudDist      = ::matCloudDist
|/ ::dist   ;
      ::matCloudAngle     = ::matCloudAngle
|/ ::angle  ;
      ::matCloudHx        = ::matCloudHx
|/ ::dx;    ;
      ::matCloudHy        = ::matCloudHy
|/ ::dy     ;
      ::matCloudN         = ::matCloudN
|/   1      ;
      ::matCloudZMeanI    = ::matCloudZMeanI
|/ ::zi     ;
      ::matCloudZMeanJ    = ::matCloudZMeanJ
|/ ::zj     ;
      ::matCloudZVarI     = ::matCloudZVarI
|/ ::zii    ;
      ::matCloudZVarJ     = ::matCloudZVarJ
|/ ::zjj    ;
      ::matCloudZCovIJ    = ::matCloudZCovIJ
|/ ::zij    ;
      ::matCloudGaussian  =
::matCloudGaussian  |/ ::dvar   ;
     0,
      ::matCloudN       [::indx[1]] +=   1
;
      ::matCloudZMeanI  [::indx[1]] += ::zi
;
      ::matCloudZMeanJ  [::indx[1]] += ::zj
;
      ::matCloudZVarI   [::indx[1]] += ::zii
;
      ::matCloudZVarJ   [::indx[1]] += ::zjj
;
      ::matCloudZCovIJ  [::indx[1]] += ::zij
;
      ::matCloudGaussian[::indx[1]] += ::dvar
;
     0);
0)));

// Remove empty rows
::indx = Loc(::matCloudN) ;
::matCloudDist     = ::matCloudDist     [
::indx ];
::matCloudAngle    = ::matCloudAngle    [
::indx ];
::matCloudHx       = ::matCloudHx       [
::indx ];
::matCloudHy       = ::matCloudHy       [
::indx ];
::matCloudN        = ::matCloudN        [
::indx ];
::matCloudZMeanI   = ::matCloudZMeanI   [
::indx ];
::matCloudZMeanJ   = ::matCloudZMeanJ   [
::indx ];
::matCloudZVarI    = ::matCloudZVarI    [
::indx ];
::matCloudZVarJ    = ::matCloudZVarJ    [
::indx ];
::matCloudZCovIJ   = ::matCloudZCovIJ   [
::indx ];
::matCloudGaussian = ::matCloudGaussian [
::indx ];

// Adjust for dist=0;
::matCloudN        = ::matCloudN        :/
(1+(::matCloudDist==0)) ;
::matCloudZMeanI   = ::matCloudZMeanI   :/
(1+(::matCloudDist==0)) ;
::matCloudZMeanJ   = ::matCloudZMeanJ   :/
(1+(::matCloudDist==0)) ;
::matCloudZVarI    = ::matCloudZVarI    :/
(1+(::matCloudDist==0)) ;
::matCloudZVarJ    = ::matCloudZVarJ    :/
(1+(::matCloudDist==0)) ;
::matCloudZCovIJ   = ::matCloudZCovIJ   :/
(1+(::matCloudDist==0)) ;
::matCloudGaussian = ::matCloudGaussian :/
(1+(::matCloudDist==0)) ;

// Calculate
::matCloudZMeanI   = ::matCloudZMeanI   :/
::matCloudN ;
::matCloudZMeanJ   = ::matCloudZMeanJ   :/
::matCloudN ;
::matCloudZVarI    =(::matCloudZVarI    :/
::matCloudN)-(::matCloudZMeanI :*
::matCloudZMeanI);
::matCloudZVarJ    =(::matCloudZVarJ    :/
::matCloudN)-(::matCloudZMeanJ :*
::matCloudZMeanJ);
::matCloudZCovIJ   =(::matCloudZCovIJ   :/
::matCloudN)-(::matCloudZMeanI :*
```

```
::matCloudZMeanJ);
::matCloudZCorIJ   =(::matCloudZCovIJ)  :/
Sqrt( ::matCloudZVarI :* ::matCloudZVarJ ) ;
::matCloudGaussian = ::matCloudGaussian :/
::matCloudN ;

// Sort by Distance
::matRank=Rank(::matCloudDist);
::matCloudDist     = ::matCloudDist
[::matRank];
::matCloudAngle    = ::matCloudAngle
[::matRank];
::matCloudHx       = ::matCloudHx
[::matRank];
::matCloudHy       = ::matCloudHy
[::matRank];
::matCloudN        = ::matCloudN
[::matRank];
::matCloudZMeanI   = ::matCloudZMeanI
[::matRank];
::matCloudZMeanJ   = ::matCloudZMeanJ
[::matRank];
::matCloudZVarI    = ::matCloudZVarI
[::matRank];
::matCloudZVarJ    = ::matCloudZVarJ
[::matRank];
::matCloudZCovIJ   = ::matCloudZCovIJ
[::matRank];
::matCloudZCorIJ   = ::matCloudZCorIJ
[::matRank];
::matCloudGaussian = ::matCloudGaussian
[::matRank];

// Sort by Angle within Distance
::matCloudLag = ::matCloudDist>0 ;
for(::i=2,::i<=NRow(::matCloudDist), ::i++,
       ::matCloudLag[::i] =
::matCloudLag[::i-1] + (
::matCloudDist[::i]!=::matCloudDist[::i-1])
   );
::matRank=Rank(360*::matCloudLag +
::matCloudAngle);
::matCloudDist     = ::matCloudDist
[::matRank];
::matCloudAngle    = ::matCloudAngle
[::matRank];
::matCloudHx       = ::matCloudHx
[::matRank];
::matCloudHy       = ::matCloudHy
[::matRank];
::matCloudN        = ::matCloudN
[::matRank];
::matCloudZMeanI   = ::matCloudZMeanI
[::matRank];
::matCloudZMeanJ   = ::matCloudZMeanJ
[::matRank];
::matCloudZVarI    = ::matCloudZVarI
[::matRank];
::matCloudZVarJ    = ::matCloudZVarJ
[::matRank];
::matCloudZCovIJ   = ::matCloudZCovIJ
[::matRank];
::matCloudZCorIJ   = ::matCloudZCorIJ
[::matRank];
::matCloudGaussian = ::matCloudGaussian
[::matRank];

// Color of points
::matCloudHue=Floor(12*mod(::matCloudAngle,18
0) :/ 180);
::matCloudShade=  2 - (::matCloudN>2 ) -
(::matCloudN>5 ) -  (::matCloudN>10) -
(::matCloudN>30) ;

// Subset cloud
::SubsetCloudData( Choice="Semi-Variance",
Angle=90, AngleTol=90);
// Please Wait Cancel
::dbPleaseWait << CloseWindow;
0);//End::CalculateVariogramCloud
```

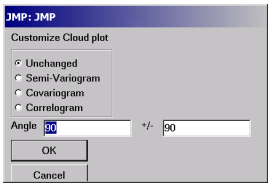
VISUAL EXLORATION OF ANISOTROPIC CASES

Anisotropy scenarios imply the need for the study of the semi-variogram in different directions and angles. This is accomplished by opening the cloud box in the top of the initial output window and experimenting with handles provided to explore the key anisotropy parameters usually refer to as phi and lambda. The colors are selected by in the semi-variance plot below to the left are chosen by the script to accommodate anisotropy angles with red corresponding to the East-West axis and cyan to the North to South axis. Note here that the overall semi-variogram discussed above that appeared below the cloud was using the color blue and was angle independent.

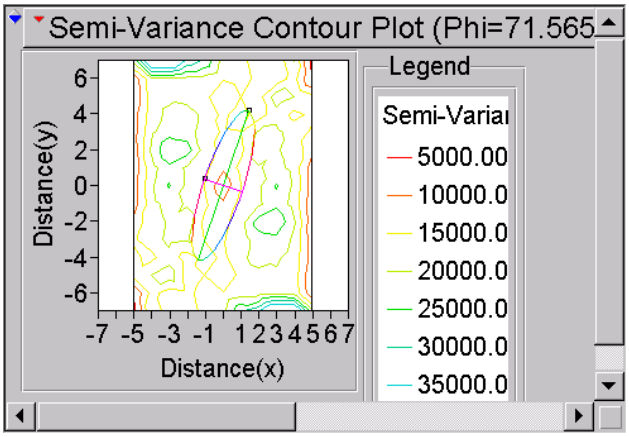


Lag	No Couples	Distance	Angle	Semi-Variance	Covariance	Correlation
0	47	0	0	6181.91	10958.2	0.63933
1	158	1	0	14851.6	3596.14	0.19607
1	166	1	90	10847	6476.51	0.37787
2	140	1.41421	45	16199.3	2177.98	0.11934
2	138	1.41421	135	16435.9	2390.28	0.13006

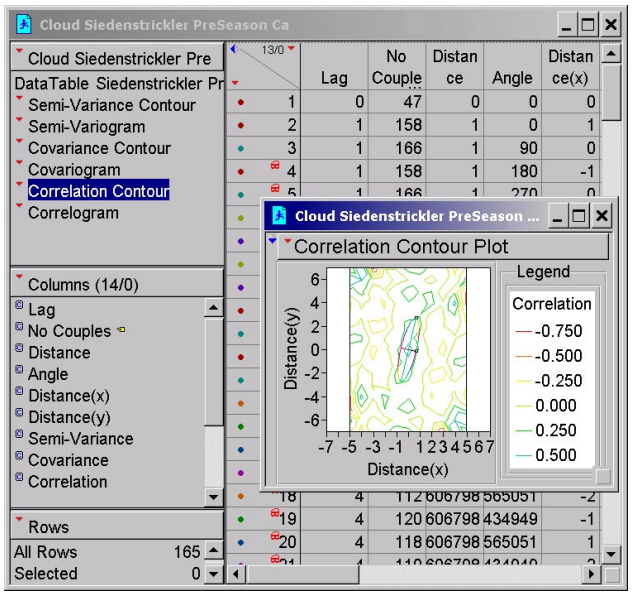
The Customize button hides or displays points from a particular angle based on user input for direction and tolerance and allows displaying other related statistics in addition to semi-variance as shown in the right.



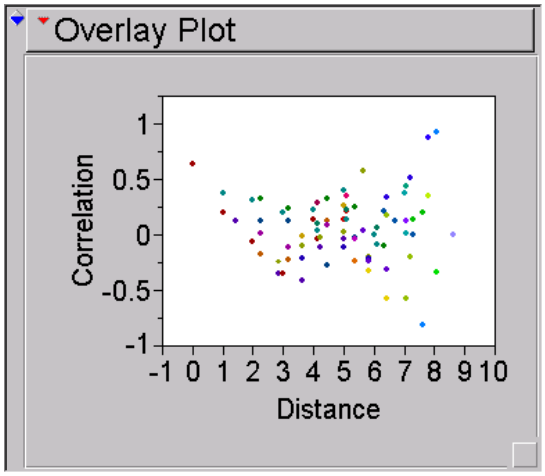
Instead of discussing here the pseudo contour plot provided to the right in the above capture we manipulated (by dragging on the two provided handles) the initially drawn isotropic circle into ellipse that provides a better visual towards understanding the potential anisotropy in these data. Again manipulation the provided handles by utilizing graphing scripts can help the user to choose appropriate phi and lambda parameters values.



The results of clicking on the Make JMP table are shown below and provides in tabular form the statistics commonly calculated in such cases in addition to the techniques used to manipulate and display various contour and graphs that the user can interact with and that are all dynamically linked. The two handles in all contours help the user with the selection of the phi (angle) and lambda (ratio of the two major and minor ellipsoid axis) that are commonly used to control the directionally induced anisotropy. Also in the capture below one can see the output of running the highlighted script ant the resulted output. Since these plots are all dynamically link reflect previous choices of the anisotropy parameters. One needs to remember that the default display without user modification corresponds to a circle with lambda equal to 1 but what is shown here on all these captures are the result of changing the inputs with the help of the two handles that allows us to modify and select appropriate chosen for the phi and lambda parameters by dragging these two handles. Again the use of colors here is similar to the use above discussed above.



In addition to the contour one can chose to display other kinds of related graphs that summarize visually other calculated statistics by the variogram script. For example below we displayed the results from running the Correlogram script attached to the above Table Panel (by option clicking on it and selecting to Run Script). The output shown below shows how the spatial correlations decline for observations further. Again color-coding these spatial correlations indicates the effects of directional relationships.



CONCLUSION

JMP Scripting Language
. Extends JMP for users that need to do more things.
. Empower users with packaging production features.
. Gives new expressive power to concepts.

We hoped that we have demonstrated that our variogram script along with JSL and JMP makes for a "sufficient exploratory environment" where an experienced geostatistical analyst would fill comfortable. We hope that we have shown that JSL provides the necessary tools where a knowledgeable programmer can tailor his needs for exploring and visualizing spatial relations that can be a challenging to understand and model without the proper tools. In the near future we hope to enhance the tools discussed above by addition to providing tools that accomplish the next step of such analyses that usually include numerical interpolation known as Kriging.

REFERENCES

JMP® Scripting Guide, Version 4 Copyright © 2000 by SAS Institute Inc., Cary, NC, USA.

Clark, I., and W. Harper 2000, *Practical Geostatistics 2000*, ECOSSE North America, LLC.

Goovaerts, P. 1997. *Geostatistics for Natural Resources Evaluation*. Oxford Univ. Press, New-York.

Isaaks E. H., and Srivastava R. M, 1989, *An Introduction to Applied Geostatistics,* Oxford University Press.

Marx D and Kevin Thompson, 1987, Practical Aspects of Agricultural Kriging, Arkansas Agricultural Experiment Station, Bulletin 903.

SAS/STAT® User's Guide, Version 8, PROC KRIGE2D, Cary, NC:SAS Institute Inc., 1999, Vol 2, pag. 1707.

SAS/STAT® User's Guide, Version 8, PROC VARIOGRAM, Cary, NC:SAS Institute Inc., 1999, Vol 3, pag. 3641.

ACKNOWLEDGMENTS

We would like to thank Dr. Scott of the Department of Crop, Soil, and Environmental Sciences at the University of Arkansas for providing the data set used here as an example.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the authors at:

Author Name: Andy Mauromoustakos
Company: U of Arkansas
Address: Agri Stat Lab, AGRX 101
City state ZIP: Fayetteville AR, 72701
Work Phone: (501) 575-5678
Fax: (501) 575-8643
Email: andym@uark.edu
Web:http://www.uark.edu/misc/andym/andy1.html

To get a copy of the script please email the author whose is in the process of continually upgrading and enhancing its functionality

Author Name: Kevin Thompson
Company: U of Arkansas
Address: Agri Stat Lab, AGRX 101
City state ZIP: Fayetteville AR, 72701
Work Phone: (501) 575-6816
Fax: (501) 575-8643
Email: kthompsn@uark.edu