Semivariogram Models Via Function Handles

A function handle is a Matlab data type (an object) that provides a means for calling a function indirectly; type \texttt{doc function-handle} for more information on this Matlab data type. In what follows, we will be using function handles to construct an exponential semivariogram model \( \gamma(h; b_1, b_2) = b_1\left[1 - \exp\left(-\frac{3h}{b_2}\right)\right] \), as well as a nugget model \( \gamma(h; b_1) = 0 \) if \( h = 0 \), \( b_1 \) otherwise, where \( b_1 \) and \( b_2 \) are, respectively, the sill and range parameters.

1. Create a function handle for the exponential semivariogram model as:
\[
\text{funGamExp} = @(b,x) b(1)*(1-exp(-3*x/b(2)))
\]
where \( x \) stands for an input distance vector and \( b = [b(1) \ b(2)] \) for a parameter vector; type \texttt{whos funGamExp} to see some properties of this data type. You can now construct a distance vector, say:
\[
x = (0:0.1:50)
\]
and compute the associated semivariogram values for an exponential model with sill 1 and range 20, i.e., \( b = [1 \ 20] \), as:
\[
yMod = \text{funGamExp}([1 \ 20],x); \quad \text{plot}(x,yMod,'.');
\]
Repeat this computation and display for different entries in vector \( b \).

2. You can add a nugget effect contribution to the above model, by creating a new function handle as:
\[
\text{funGamNugExp} = @(b,x) b(1)*(x\sim=0) + b(2)*(1-exp(-3*x/b(3)))
\]
where now \( b(1) \) is the nugget sill, \( b(2) \) is the exponential sill, and \( b(3) \) is the range; note that the total sill for this model is: \( b(1)+b(2) \). For the same distance vector \( x \), compute and display the corresponding semivariogram values for \( b = [1 \ 9 \ 25] \) as:
\[
yMod = \text{funGamNugExp}(b,x); \quad \text{plot}(x,yMod,'.');
\]
Repeat this computation and display for different entries in \( b \).

3. Last, you can create a nested model with a nugget contribution and 2 exponential structures as:
\[
\text{funGamNugExpExp} = @(b,x) b(1)*(x\sim=0) + b(2)*(1-exp(-3*x/b(3))) + b(4)*(1-exp(-3*x/b(5)))
\]
where \( b(4) \) and \( b(5) \) are, respectively, the sill and range of a 2-nd exponential model; not that the total sill for this model is: \( b(1)+b(2)+b(4) \). For the same distance vector \( x \), compute and display the corresponding semivariogram values for \( b = [1 \ 4 \ 10 \ 5 \ 40] \) as:
\[
yMod = \text{funGamNugExpExp}(b,x); \quad \text{plot}(x,yMod,'.');
\]
Repeat this computation and display for different entries in \( b \).

4. Describe in writing what are the inputs to the semivariogram model of Requisite #3, and what its parameters. Once you decide on a vector \( b \), how is the statement \( yMod = \text{funGamNugExpExp}(b,x); \) called? If you had a set of observations in a vector \( y \), what is \( y - yMod \) called? How would you go about estimating the entries of vector \( b \) from the observations in \( y \)? Provide examples of different scenarios (not related to the semivariogram) where such a procedure would be useful.

5. Create a function handle that corresponds to a semivariogram model with a nugget contribution and a spherical structure. Last, create a function handle that corresponds to a covariogram model with a nugget contribution and an exponential structure. For the same distance vector \( x \) used above, compute the corresponding model values for different entries in vector \( b \).
Model Fitting via Nonlinear Least Squares

Download archive LabVariogramFit.zip from the class Web site, unzip it, and add it to MATLAB’s path. Load the binary MATLAB data file BayAreaData.mat into your workspace as load BayAreaData.mat. This file contains several arrays pertaining to precipitation in the Bay Area. Array bayprecip contains the time-average (from Nov 01 1981 to Jan 31 1982) of daily precipitation in the Bay Area recorded at 77 rain gauges. More precisely, bayprecip consists of 4 columns with: #1 rain gauge number, #2 and #3 rain gauge longitude/latitude coordinates (in degrees), and #4 precipitation (in mm/day). Array baycoast contains 2 columns with longitude/latitude coordinates of points comprising a digitized coastline for the study region. Display the sample precipitation data as:

\[
\text{scattermap(bayprecip, [2 3], 20, 4, 0, [1 14], [0 1])}; \text{hold on}; \text{plot(baycoast(:,1), baycoast(:,2), 'k-', 'LineWidth', 0.1)}
\]

1. Define the following distance class midpoints: \( \text{lagMid} = (0:0.05:1.5)' \), and construct array \( \text{LagMidTol} = [\text{lagMid} 0.05*\text{ones(size(lagMid))}] \) with midpoints and tolerances for these classes. Now compute the omni-directional sample semivariogram as: \( S = \text{scatter2struct(bayprecip, [2 3], [], LagMidTol, [4 4 1])} \); and display it as:

\[
\text{plotsamplestruct(S, 1, \{[1 1]\}, \{'{+'},1\});}
\]

In what follows, you will be fitting 3 different semivariogram models to the above sample semivariogram using function \text{nlinfit} from the Statistics Toolbox and the 3 function handles you created in the previous page. To do so, consider the \( nLag = 30 \) lag distances \( \text{lag} = S.\text{struct\{1\}}(2:end, 1) \) and the corresponding semivariance values \( \text{gam} = S.\text{struct\{1\}}(2:end, 2) \); the 0-lag distance and 0-semivariance are omitted, because they add false confidence to the results.

2. Use function \text{nlinfit} and function handle \text{funGamExp} to estimate the parameter vector \( bHat \) of an exponential semivariogram model that best fits the data in \( \text{gam} \), as: \( bHat = \text{nlinfit(lag, gam, funGamExp, [5 1])} \), where the last vector [5 1] contains an initial guess for the parameter vector \( b \). Compute the fitted model semivariogram values at the \( nLag = 30 \) distances in vector \( \text{lag} \), as: \( \text{gamHat} = \text{funGamExp(bHat, lag)} \); and overlay the results on the sample semivariogram plot as: \( \text{hold on}; \text{plot(lag, gamHat, 'ro')} \); Compute the semivariogram residuals, the sum of squared errors (SSE), the mean squared error (MSE), and the Akaike information criterion (AIC), as: \( rHat = \text{gam} - \text{gamHat}; \text{MSE} = \text{sum(rHat.^2)}/\text{nLag} \); \( \text{SE} = \text{SSE}/(\text{nLag} - \text{numel(bHat)) AIC} = \text{nLag}*(\log(2*pi*\text{SSE/nLag})+1) + 2*(\text{numel(bHat)+1}) \), and comment on the model fit. Note: Verify that your computed MSE value is the same as the one obtained from function \text{nlinfit} as: \( [b, r, Covb, J, mse2] = \text{nlinfit(lag, gam, funGamExp, [5 1])}; \) Note: See Requisite #2 in the next page for a means of generating simulated realizations from a process with this particular variogram model; such a simulation exercise is useful for appreciating what is implied in terms of attribute smoothness by a variogram model.

3. Repeat Requisite #2, but now considering the nugget + exponential semivariogram model you coded in function handle \text{funGamNugExp}; comment on the model fit.

4. Repeat Requisite #2, but now considering the nugget + exponential + exponential semivariogram model you coded in function handle \text{funGamNugExpExp}.

5. How would you handle a situation where all 3 semivariogram models above yielded approximately the same goodness-of-fit statistics for this particular sample semivariogram?
**Model Fitting via Nonlinear Weighted Least Squares**

In what follows, we will introduce a weighting scheme in computing the overall misfit between observations, here in $\text{gam}$, and fitted model values, here in $\text{gamHat}$. More precisely, we will consider a weights vector with $nLag = 30$ entries defined as: $\text{npairs} = S.\text{struct} \{ \text{2:end}, 3 \}$; $\text{w} = \text{npairs} ./ (\text{lag} \wedge 2)$; this weighting scheme gives more weight to small lag distances associated with many data pairs (other possible weighting schemes exist, too). Note that the actual magnitude of the weights does not matter, just their relative magnitude. Display again the sample semivariogram of precipitation, as: $\text{plotsamplestruct}(S,1,\{1 1\},\{'+'\},1)$;

1. One can account for the weights vector $\text{w}$ in the non-linear least squares fitting procedure, by constructing weighted observations, as $\text{gamW} = \sqrt{\text{w}} \times \text{gam}$, and weighted model predictions; for the exponential semivariogram model, this can be done as: $\text{funGamExpW} = @(b,x) \sqrt{\text{w}} \times \text{funGamExp}(b,x)$; The corresponding parameter vector can be obtained as: $\text{bHatW} = \text{nlinfit}(\text{lag}, \text{gamW}, \text{funGamExpW}, [5 1])$; Compute the fitted model semivariogram values at the $nLag = 30$ distances in vector $\text{lag}$, as: $\text{gamHatW} = \text{funGamExp}(\text{bHatW}, \text{lag})$; – **note** the use of $\text{funGamExp}$ not $\text{funGamExpW}$ – and overlay the results on the sample semivariogram plot as: $\text{hold on}; \text{plot}(\text{lag}, \text{gamHatW}, 'ro')$; Compute the semivariogram residuals, the weighted sum of squared errors (WSSE), and the weighted mean squared error (WMSE) as: $\text{rHatW} = \sqrt{\text{w}} \times (\text{gam} - \text{gamHatW})$; $\text{WSSE} = \text{sum}(\text{rHatW} \times 2)$; $\text{WMSE} = \text{WSSE} / (\text{sum(}\text{w}) - \text{numel(}\text{bHatW}))$, and comment on the model fit.

2. You will now generate simulated attribute realizations from a Gaussian spatial process with a spatially constant mean (no first-order effects) and the above variogram model as its spatial smoothness descriptor (2nd-order stationarity). Realizations will be generated at the grid nodes of a regular raster, whose specification is stored in array $\text{GridSpecs}$, using custom function $\text{mafftsim}$. First, create an array $\text{vModelSpecs}$ with the variogram model specification: $\text{vModelSpecs} = [2 \text{ bHatW}(1) 0 \text{ bHatW}(2) \text{ bHatW}(2) 0]$, where 2 implies an exponential model, 0 $\text{bHatW}(2)$ $\text{bHatW}(2)$ implies an isotropic model with ranges $\text{bHatW}(2)$ along both $x$ and $y$-directions, and the last 0 is a free (arbitrary) parameter. Next, create a vector $\text{simPars}$ with simulation parameters: $\text{simPars} = [10 \ 1234 \ 0]$, corresponding to $\text{nSim} = 10$ realizations with a random number seed of 1234 and random sampling (0). Last, specify the target mean of the simulations as that of the sample precipitation data: $\text{targMean} = \text{mean(bayprecip(:,4))}$. You can now generate the simulated realizations as: $\text{Sim} = \text{mafftsim}(\text{GridSpecs}, [], \text{vModelSpecs}, \text{targMean}, \text{simPars})$; Note that the output from function $\text{mafftsim}$ contains $\text{nx*ny}$ rows and $\text{nSim}$ columns; each column is a simulated attribute image, in this case stored as a long 1D vector not as a 2D matrix. Display a couple of attribute realizations and comment on their spatial patterns. The 3-rd realization, for example, can be displayed as: $\text{rastermap}(\text{GridSpecs}, \text{Sim}, 3, 0, [0 15], [0 1])$, where [0 15] are the limits of the colorscale.

3. Repeat Requisites #1 and #2, but now considering the nugget + exponential semivariogram model you coded in function handle $\text{funGamNugExp}$; comment on the model fit and on the spatial attribute patterns implied by this model.

4. Repeat Requisites #1 and #2, but now considering the nugget + exponential + exponential semivariogram model you coded in function handle $\text{funGamNugExpExp}$; comment on the model fit and on the spatial attribute patterns implied by this model.

5. Comment on the use of weights for assigning more importance to observations or residuals in model fitting and parameter estimation.
Parameter Uncertainty and Confidence Intervals

In this section, you will be first assessing the uncertainty in model estimated parameters and then propagating that uncertainty to model predictions. Here, the term model pertains to a semivariogram model, and the term prediction pertains to semivariogram model values for any arbitrary distance, particularly distances not in the original lag vector \( \text{lag} \). In what follows, consider only the case of an exponential semivariogram model, i.e., the function handle \( \text{funGamExp} \) you created in Requisite \#1 of page 1, as well as the case of non-weighted (ordinary) least squares. To begin, re-display the sample semivariogram on a new figure as: \( \text{clf; plotsamplestruct}(S,1,\{[1 1]\},\{'+\}',1); \)

1. Parameter uncertainty is linked to the covariance matrix of the estimated parameters, provided by \( \text{nlinfit} \) as:
   \[
   [\hat{b},r,J,CovB,mse] = \text{nlinfit}(\text{lag},\text{gam},\text{funGamExp},[5 1]);
   \]
The uncertainty in the individual entries of \( \hat{b} \) is quantified by the standard error vector \( \text{se} = \sqrt{\text{diag}(\text{CovB})} \); The sampling distribution of, say, \( \hat{b}(1) \), has mean \( \hat{b}(1) \) and standard deviation \( \text{se}(1) \). The sampling distribution of the \( t \)-statistic defined as:
   \[
   (b(1)-\hat{b}(1))/\text{se}(1)
   \]
   is a \( t \) PDF with \( \nu = n\text{Lag} - \text{numel}(\hat{b}) \) degrees of freedom. This implies that, for a significance level \( \alpha = 0.05 \), the 2-sided 95\% confidence intervals for the true parameters can be computed as:
   \[
   \delta = \text{se} \cdot tinv(1-\alpha/2,\nu)
   \]
   \[
   \hat{b}\text{CI} = [(\hat{b}(:) - \delta) (\hat{b}(:) + \delta)], \text{where} \delta \text{is the width of the intervals from either side of} \hat{b}.
   \]
The above steps are automated in function \( \text{nlparci} \) as:
   \[
   \text{nlparci}(\hat{b},r,\text{'covar'},\text{CovB},\text{'alpha'},0.05)\]

2. The variance (uncertainty) in the parameter estimates can be transformed into variance (uncertainty) regarding model predictions, which in turn can be used to derive confidence intervals for such predictions. For the distances for which sample semivariance values have been already computed, i.e., for the entries of vector \( \text{lag} \), this can be done as:
   \[
   [\text{gamHat},\delta] = \text{nlpredci}(\text{funGamExp},\text{lag},\hat{b},r,\text{'covar'},\text{CovB},\text{'alpha'},0.05);
   \]
   hold on; \( \text{plot}(\text{lag},\text{gamHat},\text{'r'},\text{lag},\text{gamHat-}\delta,\text{'r+'},\text{lag},\text{gamHat+}\delta,\text{'r+'}); \)
   For values \( \text{lagF} = (0:0.01:1.5)' \) discretizing the distance axis, this can be done as:
   \[
   [\text{gamHatF},\deltaF] = \text{nlpredci}(\text{funGamExp},\text{lagF},\hat{b},r,\text{'covar'},\text{CovB},\text{'alpha'},0.05);
   \]
   \( \text{plot}(\text{lagF},\text{gamHatF},\text{'k'},\text{lagF},\text{gamHatF-}\deltaF,\text{'k-'},\text{lagF},\text{gamHatF+}\deltaF,\text{'k-'}); \)
   Note that all the prediction confidence intervals computed above rely on the \( t \)-PDF.

3. For more information on the capabilities of the Statistics Toolbox for nonlinear least squares fitting, type \( \text{demo('toolbox','Stat')} \) and scroll down to the last 2 demos on the Regression section. Also, check out the interactive nonlinear fitting tool, by typing \( \text{doc nlintool} \).

4. Describe in writing a Monte Carlo simulation procedure for propagating uncertainty in estimated model parameters into model predictions.