Computer exercise 1

Download the matlab files from the webpage www.maths.lth.se/matstat/kurser/fmsn20masm25/fmsn20m.zip, unpack and add the folder (with subfolders) to the matlab path. Data can be found on the webpage www.maths.lth.se/matstat/kurser/fmsn20masm25/matlab.html.

Gaussian fields, covariance estimation, and Kriging

Covariance matrices and simulation of Gaussian fields

1. Use the function matern_covariance (exponential_covariance, gaussian_covariance, spherical_covariance, or cauchy_covariance) to plot some (Matérn) covariance functions and investigate the behaviour of altering the parameters. A suitable range of distances for the plotting are 0 to 80.

2. Create a grid of suitable size (at least 50-by-60 pixels) and simulate samples of Matérn fields using different parameters. See the help of the functions distance_matrix and matern_covariance, for details. Simulation:

```matlab
% First use matern_covariance to create a Sigma-covariance matrix.
% and set mu=(a constant mean).
sz = [50 60];
R = chol(Sigma); % Calculate the Cholesky factorisation
eta = mu+R’*randn(N,1); % Simulate a sample
eta_image = reshape(eta,sz); %reshape the column to an image
imagesc(eta_image)
```

View the results, and note the behaviour for different parameter values.

For large values of $\nu$ the addition of small diagonal elements might be needed to ensure numerical stability:

```matlab
R = chol(Sigma + eye(size(Sigma))*1e-5)
```

Non-parametric estimation of covariances

1. Generate an observed image $y=\eta + \text{randn}(N,1)*\sigma_{\text{epsilon}}$ (use one of your simulations as $\eta$, so you know the true covariance)

2. Plot the residual products $z_i z_j$ for the data (i.e. the covariance cloud):

```matlab
z=y-mu; plot(D,z*z',',k')
```

where $D$ is the distance matrix. Compare with the true covariance function.
3. Use the function `covest_nonparametric` to compute the “binned” residual covariance function. Compare the estimate to the true covariance and investigate what happens when you changing the number of bins $K_{\text{max}}$ and maximum distance $D_{\text{max}}$, also investigate how miss-specifying the mean, $\mu$, effects the estimate.

**Parametric estimation of covariances**

1. Use the function `covest_ls` to compute the Least Squares estimate of the parameters for the covariance function. Compare the estimated parameters and resulting covariance function to the known truth for some different simulated fields.

2. The shape parameter ($\nu$ in matern, $\kappa$ in cauchy) is especially hard to estimate. Investigate how fixing this parameter effects the estimates of the other parameters.

**Kriging from incomplete data (if you run out of time, don't worry; this is part of the project)**

In spatial data settings, a common situation is that not all pixels are observed.

Construct an “indicator image”, $I_{\text{obs}}$ that indicates which pixels are observed. Let each pixel be observed with some probability $p$:

$$I_{\text{obs}} = \text{(rand(sz)} \leq p);$$

Divide observations and covariance matrix into observed, unobserved, and cross-covariance components

%add nugget to the covariance matrix
Sigma_yy = Sigma + sigma_epsilon^2*eye(size(Sigma));
%and divide into observed/unobserved
Sigma_uu = Sigma_yy(~I_obs, ~I_obs);
Sigma_uo = Sigma_yy(~I_obs, I_obs);
Sigma_oo = Sigma_yy(I_obs, I_obs);
y_o = y(I_obs);
y_o = y(~I_obs);

The constant mean can be modeled as a vector of ones multiplied by a known $\beta$ (in the project you will estimate $\beta$), $\mu = 1\beta$, and we construct the

$$X = \text{ones(prod(sz)}, 1);$$
$$X_u = X(\sim I_{\text{obs}});$$
$$X_o = X(I_{\text{obs}});$$

Use these components to estimate the unobserved components

$$y_{\text{rec}} = \text{nan(sz);}$$
$$y_{\text{rec}}(I_{\text{obs}}) = y_o;$$
$$y_{\text{rec}}(~I_{\text{obs}}) = \ldots;$$

and the prediction variances.