Stats 253 Summer 2015
Homework 1: Covariance Modeling
Due: Friday, July 10

Policies

- Please submit all homework files through the online homework system on the course website. You can find more instructions there.
- You may work on this homework in pairs. If you choose to do so, one person should submit all of the files (writeup, code, predictions, etc.). However, both partners should submit a statement indicating their contributions.

Description

As most of you know, it rarely rains in California during the summer. Although the sunny weather is nice, it means that California's water supply in summer is almost entirely dependent on snow that fell in the mountains during the winter. For three years now, California has seen record low amounts of snow. As a result, the entire state is currently in a severe drought (see `droughtmonitor.unl.edu`), and the government has instituted mandatory water restrictions.

The snow depth (in inches) is measured daily at a number of locations in California. In this assignment, we will look at data collected during a "wet" year and data collected during a "dry" year. The data are contained in the file `snowpack.csv`. You will notice that this file contains many missing values. Your job is to fill in all of these missing values. This assignment is a **prediction competition**: the two teams that achieve the smallest prediction errors on the "wet" year data and the "dry" year data will earn prizes.

Tasks

1. Make spatial plots to show the snow depth measurements from the "wet" and "dry" years. You should use the same color scale for both plots (i.e., the same color should correspond to the same snow depth in both plots). This can be done manually (just plot points and set the color of each point) or using specialized functions like `spplot()` in the `sp` package. If you would like to learn more about packages for spatial statistics in `R`, please refer to the Bivand et al. book, which is linked on the course website.

2. Model the trend. You should at least include linear terms for latitude and longitude, but you may want to include other predictors. To do well in the prediction competition, you will almost certainly have to include predictors like elevation (more snow falls at higher elevations) and average temperature.

   To obtain these predictors, you can query public databases. For example, a Google search for "query elevation by latitude and longitude" returns several services that return the elevation of a given location. Since you do not want to look up the 260 locations manually, you will have to learn to use the API so that you can write code to do it for you automatically. (Usually, these APIs work in a similar way: the query is passed in the URL, so you write code to fetch the URL and parse the output.) This is strictly optional, but this is a good opportunity to practice your data mining chops!

3. Once you've modeled the trend, run OLS (`lm` in `R`) and extract the residuals. Make spatial plots of the residuals as you did in Question 1. We will be developing a covariance model using these residuals. To do this, you will need to know the distance between each pair of observations. Form an $n \times n$ matrix of these distances. (Note that distance does not have to be Euclidean distance!)

4. Complete the first four tasks marked `TODO` in `covariance.R`. The code implements the "hack" method described in lecture.

   You now have a function `estimate.cov.fun()`, which takes the residuals, distance matrix, and class of covariance function (e.g., exponential, Gaussian, etc.) as inputs and returns the estimated covariance function. Apply this function to snowpack the data, and make a plot of the estimated covariance function.

5. Now that you have the estimated covariance function, you can obtain the covariance matrix and calculate the GLS estimator. Complete the last `TODO` task in `covariance.R`, which is to fill in the `gls()` function. This function outputs the GLS coefficients (or predictions, if a matrix `X0` of new data is specified). Along the way, it prints a table with the estimated coefficients and standard errors.

   It's probably easiest to do this manually. This means you'll have to do a bit of matrix manipulation in `R`. Here are some tips:

   - You'll need the X matrix to do calculations. The `model.matrix()` function extracts the X matrix from a formula like `y ~ X` or an `lm` object.
   - To multiply two matrices `A` and `B`, you write `A %*% B`.
   - To solve a system like $A\mathbf{x} = \mathbf{b}$, you write `solve(A, b)`. To calculate A^{-1}, you write `solve(A)`.

   Use the `gls` function to make predictions for the missing snowpack values. Make spatial plots of your predictions.

6. Estimate the prediction error of your model using cross-validation. That is, leave out some observations at random, fit your model to the remaining data, and calculate the average error of your predictions on the left-out data. Do this several times and report the average prediction error.

Don't forget to save your predictions and upload the completed `.csv` file with your report and code!

If you try several trend models, you may find yourself repeating Steps 2-6 for each model. If you do this, please report the estimated prediction error for all of the models you try, even if your writeup only describes the final model.