Error Propagation
GEO 428 Exercise 7 Lab Instructions

REPORT
This report (Exercise 7) is due Tuesday October 30, at the beginning of lab. Address your report to Farmer Billy. Include a diagram of the Analysis. Explain in a short paragraph why error propagation seems warranted. Then, describe the results of your work. What are the implications for the analysis if the error is closer to 15 meters than to 7? Where, if anywhere, should Farmer Billy put his array? A few maps might illustrate this section nicely. Include no more than 2 pages of written comments with your maps and diagrams.

INSTRUCTIONS
Preliminaries – downloading, projecting, and setting up GRASS
Create a new directory called lab10 under your labs directory. Download the zipped DEM from the lab page. Put it in your new directory. Unzip it – this is a geoTiff. Use gdalinfo to display its information, and gdalwarp (or the QGIS tools) to project it to UTM zone 16. Download grass_errsim.sh and grass_errprop.sh from the same place, and put them in your directory. Start GRASS, set the new location with the projected DEM, import the DEM, and describe and display it with decent colors. If the border is not set to null, use r.null to fix this. Write down its general appearance and elevation characteristics for your report.

Part 1. Generating Error Realizations
The USGS defines their DEM level 1 quality standard as 7 m RMSE desired, with 15m maximum. In addition, they note that error tends to be centered on zero, with many small errors and few large errors. This description suggests a way to characterize error: it is normally distributed (bell-shaped), with mean = 0 and stddev < 15. These are sufficient to plug into a random number generator and churn out error realizations. GRASS has this capability.

1. Find the GRASS 6.4 raster command listing page online, and look up the r.surf.gauss command. Note that it creates a raster the size of your current region and fills it with values from a normal distribution with a user-defined mean and standard deviation. Run this (Raster → Generate surfaces → Gaussian deviates) with the default parameter values. Describe with r.univar (Raster → Reports → Univariate Statistics).

2. You don't want a grid with a standard deviation of about 1. For starters, you want one with mean 0, but a standard deviation of 10. How can you get this with mapcalc? By multiplying the raster by 10 divided by its current standard deviation: r.mapcalc 'error = random * (10/0.98)'. Obviously, don't use 0.98; use whatever the standard deviation of your random map is!

3. Describe error. Do the statistics look correct? Now display it. You may want to run r.colors and set those to 'Differences', which is blues to reds. How does it look?

One important problem with this error map is that actual DEM error is likely to be spatially autocorrelated. It's one thing to generate a bunch of random numbers. It's quite another to do so while ensuring that nearby values influence their neighbors. In this lab you will combine some operations together to do a rough job of this by smoothing an initially random grid.

4. How might you use r.neighbors to do this smoothing, maybe with a 9x9 window? Write down your command, and execute it. Describe the output with r.univar and display it. Does it look reasonable? ______ What is the mean and standard deviation? ______________
5. The standard deviation dropped in the smoothed map. This makes sense, because the new grid is less variable (it’s been smoothed, after all!). However, it is probably a lot lower than you wanted it to be. Use that \texttt{r.mapcalc} command to rescale the smoothed map so that it has a standard deviation of 10. You’ve made a spatially autocorrelated error realization!

6. The final step to create an actual DEM realization is to add the error to the DEM. If your projected DEM is called \texttt{sparks} and your error realization is \texttt{err_real}, you can do this with: \texttt{r.mapcalc ‘dem_realize = sparks + err_real}

Set the colors of \texttt{dem_realize} to elevation or srtm and display it. Use \texttt{r.univar} to view the data. How does it compare to the original DEM?

7. Creating a bunch of error realizations manually would be tedious. One of the shell scripts, \texttt{grass_errsim.sh}, automates this process and lets you create many simulations quickly. Open this script in a text editor. Read the comments at the top, and the shell and GRASS commands start. Much of it should look similar to what you’ve done already. One possible difference is that it uses the \texttt{-c} (circle) parameter of \texttt{r.neighbors}. The second is that the commands occur inside a while loop - they are executed multiple times to create many different maps. Review the syntax for running this command (in the comments section).

8. Execute this script from the GRASS prompt in the terminal window to create 10 realizations called \texttt{loerr1}, \texttt{loerr2}, …, \texttt{loerr10} with radius 9 and a standard deviation of 4. Describe and display a couple of them. How do they look visually? Are the numbers appropriate?

9. Execute it again, but this time use a different model of error – call the output \texttt{hierr}, also with 10 realizations, and use a smaller radius and a target output standard deviation of 14. Write down your radius setting: ____ Describe and display a few. Do they look different from the \texttt{loerr} ones?

\textbf{Part 2. Propagating Error to the Results of GIS Operations}

Farmer Billy of Sparksville, Indiana is exploring new career options. He's thinking about constructing a state of the art solar array on his farm to generate electrical power. But he needs to know where to build it. Having read up on the specs, he has arrived at the following criteria:

1. Aspect must be within 45 degrees of due south
2. Slope must be above 5 degrees but less than 15 degrees

Billy has hired you to run a suitability analysis. Because you are concerned about error in the DEM, you tell him you will run 'high error' and 'low error' scenarios, and get back to him on where he should plan to put the arrays.

1. Develop a flow diagram and run the analysis on the DEM to determine areas in the region that fit these criteria. Include the diagram in your report. Describe and display the suitability map.

2. Now open \texttt{grass_errprop.sh} in a text editor. Read the comments carefully. Note that you will enter the operation you defined in the previous step into this script! Note especially that you must remove any maps you made inside the shell, so that they can be created the next time through the loop. Enter the appropriate commands and save the file. Run it, using \texttt{loerr} as your prefix and \texttt{suitlow} as the output raster map. It may take a while to run. Look carefully for errors!

3. When it's done, describe and display \texttt{suitlow}. Do the values appear ok (i.e. values range from 0-10)? You can color this map in several ways to try to understand it better.

4. Run the program again, but this time with \texttt{hierr} as your prefix and \texttt{suithi} as the output map.
Describe and display suithi. Do the values appear ok? How does it compare with suitlow?

5. One approach to compare the high and low error cases to the original result is to reclass suithi and suitlow so that cell values 5 and larger get classed as 1, and lower values get classed as 0. What does such a map show? Reclass using `r.mapcalc` and compare all three maps.

6. For your report to Farmer Billy, indicate your initial results (including the proportion of the total area that is suitable for solar panels). Then indicate in a paragraph or two how accounting for error in the DEM might affect these results. Maps or changed area proportions would be a good way to do this.