Outline

• Announcements
  – HW II due Friday
  – HW III option
• Interpolation
• Colormaps

HW III

• On HW III, you will have the option to create your own programming assignment
  – Write a function using Matlab graphics which does something useful for your work, and present an example
    • Ex: Visualize a particular data type
    • Ex: Produce a specific type of plot

HW III Requirements

• Your function
  – Must be a function
  – Must be general (not restricted to a single data set)
  – Must do something interesting
• You must clear your idea with me by next Wednesday
• You will have to give a (very brief) description of your function/example in the last lecture
Syllabus

7. Colormaps & Interpolation
8. Lighting
9. Transparency & Movies
10. Volumetric Visualization
11. GUI’s
12. GUI’s (HW III projects)

Interpolation

- If we want to plot with surfaces (or patches), we need some kind of mesh
- But, we are rarely able to sample on a grid
  - observations are often made at irregular intervals of time and space due to sampling constraints or equipment error (missing data)
- It is possible to calculate what the observations should’ve been at locations where we didn’t sample
  - This is known as interpolation

- It is possible to calculate what the observations should’ve been at locations where we didn’t sample
  - This implies that we know something about the system we’re observing
  - But, if we know so darn much, why bother observing?
  - The bottom line is that we are creating data and we have no way of knowing whether or not we’ve done this correctly
  - All interpolations should be treated with suspicion
Formal Statement of Problem

- **Inputs:**
  - \(X_{\text{obs}}\) = locations where we observed data (time, space, etc., can also have \(Y_{\text{obs}}, Z_{\text{obs}}\))
  - \(V_{\text{obs}}\) = observed values: \(V_{\text{obs}} = f(X_{\text{obs}})\)
    - Remember, we don't know the exact form of \(f\), but we may know something about its structure
  - \(X\) = locations where we would like to know the values

- **Then,**
  - \(V = \text{INTERPMETHOD}(X_{\text{obs}}, V_{\text{obs}}, X)\)
  - Ideally, we have enough observations and know enough about \(f\) so that \(\text{INTERPMETHOD} \approx f\)

Linear Interpolation

- Linear interpolation is the simplest form of interpolation (other than picking a constant)
  - If we have two observations, we can fit a line between them and use the equation of the line to determine \(v\)
  - Linear interpolation is used implicitly when plotting with lines or using interpolated shading

Linear Interpolation in Matlab

- Matlab's interpolation routines use linear interpolation by default
  - \(V = \text{interp1}(X_{\text{obs}}, V_{\text{obs}}, X)\)
  - \(V = \text{interp2}(X_{\text{obs}}, Y_{\text{obs}}, V_{\text{obs}}, X, Y)\)
    - \(X_{\text{obs}}\) and \(Y_{\text{obs}}\) must define a grid (i.e. same form as inputs for \text{pcolor} or \text{surface})
    - \text{interp3}, \text{interpN} work for higher-dimensional data
  - \(V = \text{griddata}(X_{\text{obs}}, Y_{\text{obs}}, V_{\text{obs}}, X, Y)\)
    - Observations need not be gridded
    - Uses Delaunay triangulation
Higher-order Interpolation

- Matlab can also interpolate using cubic functions or splines
  - `v=interp1(xobs, vobs, x, 'spline');`
  - the results are smoother, but potentially very wrong

Objective Analysis and Kriging

- Matlab’s default interpolation schemes are simple, but stupid
- Kriging (a.k.a objective analysis) is a statistical interpolation technique
  - requires you to know (or guess) the structure of your data's spatial variance

Kriging

- In kriging, `Error=f(distance)`
  - Assumes your knowledge about `v` declines as you move away from your observations
  - Can often determine error function from your observations
- `v(j)=w1*vobs(1)+w2*vobs(2)+...+wn*vobs(n)`
  - The `w`'s are weighted means of the observations, the weights are determined by the distance from `v(j)` according to the error function
  - In addition to `v`, we can also get an estimate of the interpolation error
**Kriging in Matlab**

- Kriging is computationally simple, but there are some statistical considerations
  - Isaaks & Srivastava “Applied Geostatistics”
- Matlab does not have a built-in kriging function (that I know of)
  - [http://globec.whoi.edu/software/kriging/easy_krig/easy_krig.html](http://globec.whoi.edu/software/kriging/easy_krig/easy_krig.html)
  - other software exists

**Colormaps**

- Matlab colormaps are m-by-3 matrices, where each row is an RGB vector
- When a color property (face or edge) is set to flat or interp, Matlab will determine the color using Cdata, Clim, and the colormap

**Colormaps**

- Built in colormaps (help graph3d)
  - map=copper(N);--gets copper colormap with N rows
  - map=colormap--gets current colormap (default is jet)
  - colormap(map);--sets colormap to map
  - map could be a built-in colormap (copper)
- Colormap is a property of the figure, not the axes
  - This means that we can have only one colormap per figure
Creating New Colormaps

- Matlab colormaps are usually adequate, but will need to create your own if:
  - You need more than one map/figure
  - You don’t like Matlab’s

Creating New Colormaps

- Simplest approach is modify Matlab’s
  - `map=colormap(gray); map=flipud(map);`
  - `map` will go from black to white rather than white to black
  - `brighten` lets you “brighten” or “darken” current colormap

- Create your own with `interp1`
  - `v=[1 3 4]; col=[0.5 0.5 0.5; .75 0 0; 1 1 0];`
  - `map=interp1(v,col,linspace(1,4,64),'cubic');`

Multiple Colormaps

- Working with multiple colormaps gets very complicated
  - requires lots of handle graphics work
- Tips & Things to remember
  - Single `Clim`-space, so pick something simple `[0 1],[-2 1]`
  - Transform actual `clims` to this space