CE506 Fall 2006 Homework 5 – 2D Network Adjustment

- Make a general 2D least squares network adjustment program, using common data input format and Matlab graphical user interface (GUI)
- You will read 2 ascii text input files: a point file with contents:

x1	y1	0/1
x2	y2	0/1
…	…	…
xn	yn	0/1

- This list should include all points, designate via 3rd field, 0=unknown, 1=known, fixed control point
- Suggest read via: [x,y,cpflag]=textread('%f %f %f');
- Second file is an observation file with angles and horizontal distances:

D	I	0	j	dij	0	0
A	I	j	k	deg	min	sec
…	…	…	…	…	…	…

•Suggest read via: [obscode,at,from,to,obs1,obs2,obs3]=textread('%s %f %f %f %f %f %f');

•Get number of observations, n, by [n,m]=size(at)

•Get number of points (total), np, by [np,mp]=size(x)

•For each iteration initialize matrices B=zeros(n,2*np), f=zeros(n,1);

•Within each iteration, cycle through all n observations and branch (if obscode(i) == 'D') etc. based on the obsercation type

•Suggest building 2 functions:

•[b,F,computed_obs]=dist2d(x,y,at(i),to(i),obs1(i)); where

•b = [dF/dxi dF/dyi dF/dxj dF/dyj] place into B via index calculation

•F=d – computed_obs, f = -F

•Computed_obs=sqrt[(xj-xi)^2 + (yj-yi)^2]

•[b,F,computed_obs]=angle(x,y,at(i),from(i),to(i),obs1(i),obs2(i),obs3(i)) where

•b = [dF/dxi dF/dyi dF/dxj dF/dyj dF/dxk dF/dyk] place into B via index calc.

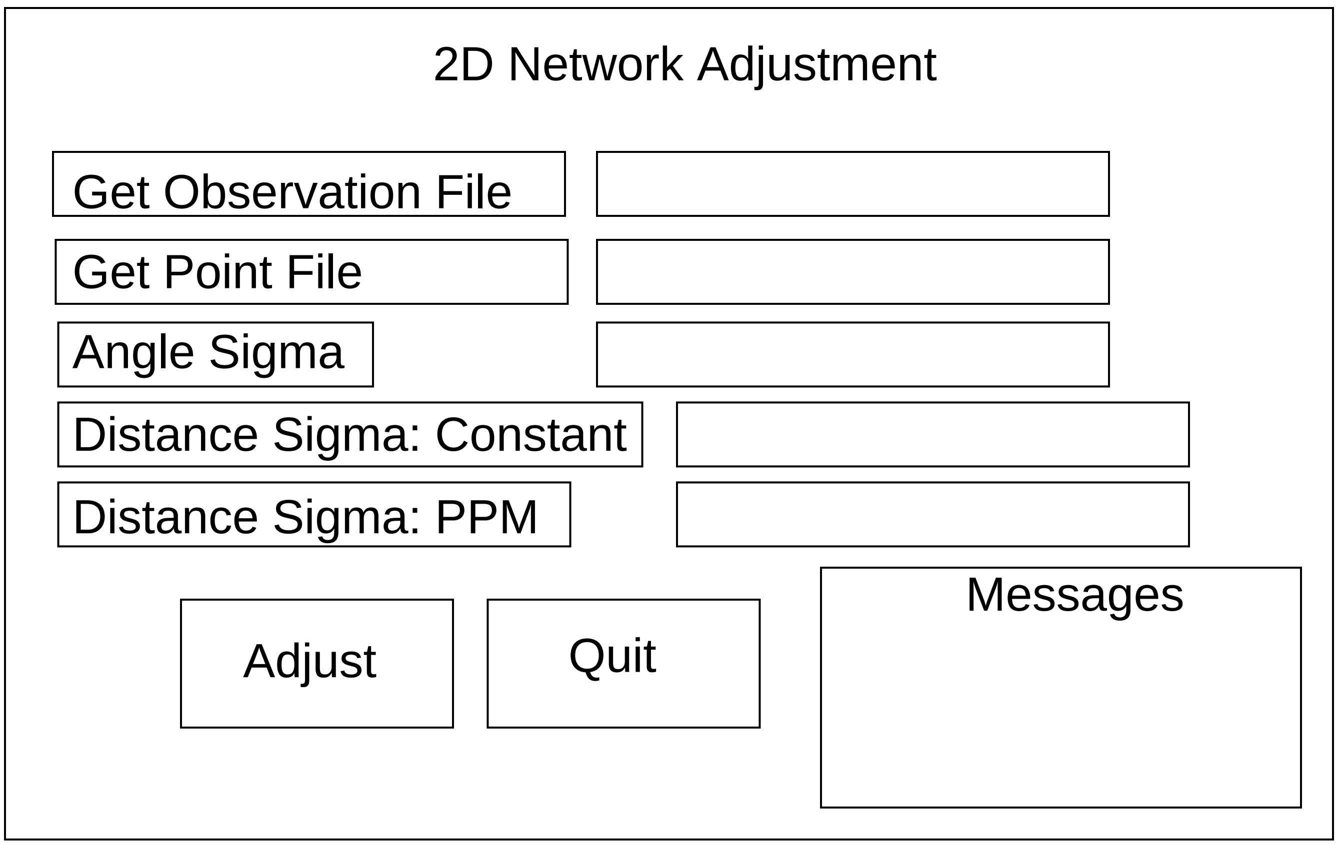
•F=alpha – computed obs, f=-F

•Computed_obs=atan[(xk-xi)/(yk-yi)] – atan[(xj-xi)/yj-yi)] (handle quadrant !!)

•See matlab gui example in “notes” section of course web page, use that as an example for file open dialogue boxes, and adding code to the function buttons.

•When done filling B-matrix, make a an array “purge” with column numbers of all control coordinates, then call B=elim_col(B,purge). After solving for delta, make augmented delta vector with zeros for cp’s by: del=insert_zerv(del,purge). Then you may just add to the x’s and y’s without regard to which are unknowns, etc.

Suggested MATLAB GUI Layout:



•Make a graphic plot of the network before adjustment with at least the points labeled (sequence number), 2 line segments for angles, 1 line segment for distance, maybe heavier weight line if distance.

•Some useful graphics commands:

plot(x,y,'r-','linewidth',2); = plot red line through vertices described by xi,yi, make it line width 2

Plot(x,y,'g*',linewidth',3); = plot green point symbol at locations xi,yi with line width 3

Hold on; = after first plot command, retain all graphic elements, otherwise each new draw command will erase previously drawn elements

Axis equal; = force x and y axes to have same scale, so shape of the network will be correct

You may want to get the adjustment part working independently of the gui, then paste it in after you are sure it works, your choice, you should output a listing file with the important results from the adjustment, formatted in a nice way

```
while (keep_going == 1)
  B=zeros(nobs,npnt*2);
  f=zeros(nobs,1);
  for i=1:nobs
    if(code(i) == 1)
      [b,F,comp_obs]=dist2d(x,y,at(i),to(i),obs1(i));
      W(i,i)=1.0/sig(i)^2;
      at_idx=(at(i)-1)*2 + 1;
      to_idx=(to(i)-1)*2 + 1;
      B(i,at_idx)=b(1);
      B(i,at_idx+1)=b(2);
      B(i,to_idx)=b(3);
      B(i,to_idx+1)=b(4);
      f(i)=-F;
      end
    if(code(i) == 2)
      [b,F,comp_obs]=ang(x,y,at(i),fr(i),to(i),obs1(i),obs2(i),obs3(i));
      sigr=(sig(i)/3600)/degrad;
      W(i,i)=1.0/sigr^2;
      at_idx=(at(i)-1)*2 + 1;
      fr_idx=(fr(i)-1)*2 + 1;
      to_idx=(to(i)-1)*2 + 1;
      B(i,at_idx)=b(1);
      B(i,at_idx+1)=b(2);
      B(i,fr_idx)=b(3);
      B(i,fr_idx+1)=b(4);
      B(i,to_idx)=b(5);
      B(i,to_idx+1)=b(6);
      f(i)=-F;
      end
    end
% now purge columns of B and solve
% set “keep_going” per the convergence status
% output results to file
```

Possibly useful code snippet for getting the partials into the right place – don’t just blindly copy – make sure you know why it works, not all statements here are pertinent to our case, note the code for this case was numeric, ours will be a string, ‘D’ or ‘A’, etc.