HW7 GPS pseudorange Adjustment
(26 Nov 2011)

(4) Single epoch 13:00:00

$n = 8$
$n_0 = 4$
$r = 4$

Indirect Observations

parameters: X_R, Y_R, Z_R, dt_R

F: $PR_{CORR} - \left[(X_R - X_s)^2 + (Y_R - Y_s)^2 + (Z_R - Z_s)^2\right]^{1/2} - c \cdot dt_R = 0$

$\sigma_0 = 25\,m$ (.025 km!)

Results:
no satellites rejected due to 10° elevation angle mask

receiver coordinates:

$\hat{X}_R = 260.633235$ (km)
$\hat{Y}_R = -4853.044990$
$\hat{Z}_R = 4116.971011$

residuals: .0159 (km)

$-.0078$
$.0323$
$.0197$
$.0016$
$-.0318$
$-.0243$
$-.0056$

global test: $\chi^{2*} = 5.43$, $CV_1 = 0.48$, $CV_2 = 11.14$
(2-sided) $\Rightarrow$ Pass (accept H_0)

---

(5) Three epochs 13:00:00, 13:30:00, 14:00:00

$n = 24$ (1 satellite rejected due to elev. angle < 10° sat 28 in epoch 3)

$n_0 = 6$
$r = 18$

Indirect Observations

parameters: $X_R, Y_R, Z_R, dt_{R_1}, dt_{R_2}, dt_{R_3}$

results:

$\hat{X}_R = 260.639064$ (km)
$\hat{Y}_R = -4853.022217$
$\hat{Z}_R = 4116.937166$

$RMS_v = .0257$ km

residuals:

.0207	.0151	.0175
.0028	.0076	.0231
.0331	.0355	−.0572
.0112	.0107	.0457
.0006	.0039	.0161
−.0281	−.0347	−.0305
−.0357	−.0357	−.0231
−.0046	−.0024	.0083

global test : $\chi^{2*} = 25.49$, $CV_1 = 8.23$, $CV_2 = 31.52$
(2-sided) $\Rightarrow$ pass

---

⑥ epoch 13:00:00, receiver constrained to be 100 m from
(260.679818, −4852.970302, 4116.969922) ←- - → (X_c, Y_c, Z_c)

$n = 8$ $u = 4$: X_R, Y_R, Z_R, dt_R
$n_o = 3$ $s = 1$: one constraint equation
$r = 5$ $c = 8$: indirect observations

$c + s \stackrel{?}{=} r + u$ $\quad F_c : 0.100 - \left[(X_R - X_c)^2 + (Y_R - Y_c)^2 + (Z_R - Z_c)^2\right]^{1/2} = 0$

$8 + 1 = 5 + 4$
$9 = 9$ ✓

results :

$\hat{X}_R = 260.629524$

$\hat{Y}_R = -4853.056171$

$\hat{Z}_R = 4116.979770$

residuals : .0170
−.0123
.0296
.0225
.0037
−.0324
−.0226
−.0055

global test : $\chi^{2*} = 5.50$, $CV_1 = 0.83$, $CV_2 = 12.83$
(2-sided) $\Rightarrow$ pass

$F_c = -1.3 \times 10^{-6}$

---

in listings : ④ gps_nav51a *
⑤ gps_nav53a *
⑥ gps_nav51b *

Check for elevation angle:

$$\begin{bmatrix} e \\ n \\ u \end{bmatrix}_s = M_x(90^\circ - \phi) \cdot M_z(\lambda + 90^\circ) \left[\begin{pmatrix} X_s \\ Y_s \\ Z_s \end{pmatrix} - \begin{pmatrix} X_R \\ Y_R \\ Z_R \end{pmatrix} \right]$$

ϕ, λ & $X_R Y_R Z_R$ all refer to receiver location

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{bmatrix} (N+h)\cos\phi\cos\lambda \\ (N+h)\cos\phi\sin\lambda \\ ((1-e^2)N+h)\sin\phi \end{bmatrix}$$

$\phi, \lambda, h \rightarrow XYZ$ closed form

$$N = a/(1 - e^2\sin^2\phi)^{1/2}$$

$a = 6378137.0$ m

$\frac{1}{f} = 298.257223563$

$e^2 = 2f - f^2$

$b^2 = a^2(1-e^2)$

$e^2 = \frac{a^2 - b^2}{a^2}$

$f = \frac{a-b}{a}$

$XYZ \rightarrow \phi\lambda h$ iterative solution

approximation for ϕ

loop until no change:

- compute N
- $h = \frac{(X^2+Y^2)^{1/2}}{\cos\phi} - N$
- $\phi = \tan^{-1}\left[\frac{Z}{(X^2+Y^2)^{1/2}}\left(1 - e^2\left(\frac{N}{N+h}\right)\right)^{-1}\right]$

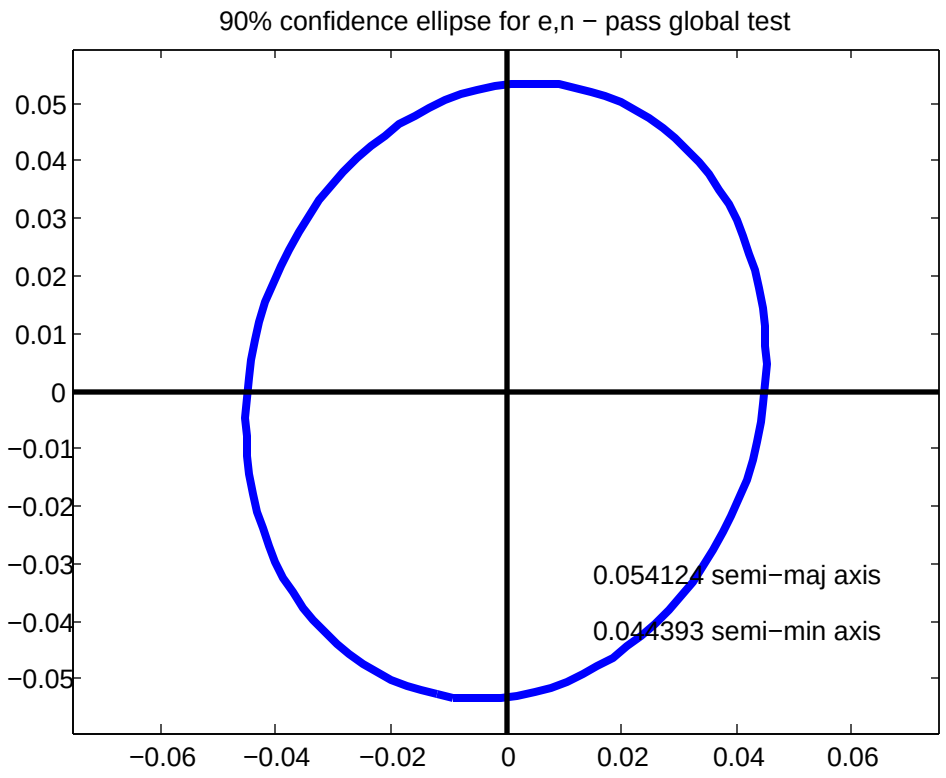
$\lambda = \tan^{-1}\left(\frac{Y}{X}\right)$ use 2 argument arctangent function atan2(Y, X)

```
gps_nav51a
numsat =
     8
sats =
    16
    13
    23
    10
     6
     7
    19
     3
m =
     8
numsat =
     8
after editing out defective sats
numsat =
     8
c1 xs ys zs dt
ans =
  Columns 1 through 3
              21287141.34                9671.17665             -11273.286067
              20287554.48               -5734.807399            -20258.741892
              20605103.28               -1126.552213            -25140.371937
              24293518.04              -22017.451052              -5852.11655
              21364948.06               14140.406396            -14804.666272
               22499418.9              -15186.548693             -8878.439153
              21662186.34                8347.789114            -24844.425342
              20253866.26               11027.953133            -19286.137311
  Columns 4 through 5
             22027.954669                -200.884996
             16005.394146                 248.828766
              7915.839784                 271.527222
             13742.241308                -14.912353
             16924.338317                -58.414824
              19985.16409                 29.497196
              2801.519349                -189.641841
             13886.297212                 775.502939
#   sat. id
ans =
     1    16
east, north, up of satellite for elev. angle check
ans =
          9052.76533048471          9140.70388902187          16884.2719688828
elevd =
           52.694616383131
#   sat. id
ans =
     2    13
east, north, up of satellite for elev. angle check
ans =
         -6812.87323234033         -727.888551905959          19174.7724344859
elevd =
           70.336864806318
#   sat. id
ans =
     3    23
east, north, up of satellite for elev. angle check
ans =
         -2473.01125170978         -10206.5150230189          17822.5684034722
elevd =
```

```
          59.4915299610946
#   sat. id
ans =
     4    10
east, north, up of satellite for elev. angle check
ans =
          -22299.577380379          7451.52569989327          6095.92019233292
elevd =
          14.5351729699529
#   sat. id
ans =
     5     6
east, north, up of satellite for elev. angle check
ans =
          13326.2051335967          2813.76579120993          16438.0353737534
elevd =
          50.3559565824672
#   sat. id
ans =
     6     7
east, north, up of satellite for elev. angle check
ans =
         -15640.7804486962          10003.0547395423          12724.9520507625
elevd =
          34.4264003216796
#   sat. id
ans =
     7    19
east, north, up of satellite for elev. angle check
ans =
          7003.56861856772         -14235.8109717316          14665.6379957201
elevd =
           42.749785202393
#   sat. id
ans =
     8     3
east, north, up of satellite for elev. angle check
ans =
           9977.9238214494          -2293.3252966074          17744.6625657583
elevd =
          60.0164812847079
condN =
          188.244407355102
disp_del =
                         1
        0.0292350003327373
       -0.0269903061604451
        0.0450107887463371
         0.121608840543441
condN =
          188.242825279625
disp_del =
                         2
    -4.77171431272408e-008
    -4.87734835879411e-008
     1.54302779441388e-008
    -2.90479135596788e-008
we have converged
receiver location (km)
ans =
          260.633234952616         -4853.04499035493          4116.97101080418
receiver clock bias (usec)
```

```
rdt =
         0.121608811495527
residuals (km)
v =
         0.015976052560702
       -0.00779922601096748
         0.0323264025303067
         0.0197165570329339
        0.00165072544625723
        -0.0318763011472966
         -0.024377520539306
       -0.00561668987262968
rms =
         0.0205986507011942
r =
     4
test_stat =
          5.43109645708555
cv1 =
          0.48441855708793
cv2 =
          11.1432867818778
pass global test at alpha = 0.05
convert XYZ to phi,lam,h
phioo =
         0.706126464069501
N =
          6387.14518884767
new_phi =
         0.706126341195602
phioo =
         0.706126341195602
phi
result =
                        40
                        27
          29.0129526818379
lambda
result =
                       -86
                       -55
          -33.160920126619
h
h =
         0.236789412511826
J =
          0.99856099354595        0.0536278115209362                          0
        -0.034798617800479         0.647957494057097         0.760881161610946
         0.0408043915247028        -0.759786248708623         0.648891252757792
cov enu
mx =
      0.000443302769107075     5.44218791494313e-005
     5.44218791494312e-005      0.000620746424128162
alen =
         0.0541237883266633
blen =
         0.0443930530407871
theta =
         -1.84590307002344
zfact =
          1.64485362695147
half_width =
```

```
        0.0930159669289425
width =
         0.186031933857885
diary off
```


90% confidence ellipse for e,n − pass global test
0.05
0.04
0.03
0.02
0.01
0
−0.01
−0.02
−0.03
−0.04
−0.05
−0.06
−0.04
−0.02
0
0.02
0.04
0.06
0.054124 semi−maj axis
0.044393 semi−min axis

```
% gps_nav51a.m 15-nov-11
% solve gps pseudorange problem for 1 epoch
% include elevation angle mask

% note you may need to edit the obs file to replace spaces with
% zeros in satellite number string, and you may need to edit the
% data fields to replace spaces with zeros ??? why are
% spaces there ???

fido=fopen('epoch1.txt','rt');

% interpret satellite prn's in the observation file
S=textscan(fido,'%d %d %d %d %d %f %d %s',1);
% need to edit leading zero into single digit sat number fields
str=char(S{8});
ck1=double(str(1));
ck2=double(str(2));
start_char=0;
proceed=0;
if((ck1 >= 48) & (ck1 <= 57))
  start_char=2;
  end
if((ck2 >= 48) & (ck2 <= 57))
  start_char=3;
  end
switch start_char
  case 0
    disp('cannot interpret satellite string');
    proceed=0;
  case 2
    proceed=1;
    numsat=str2num(str(1));
  case 3
    proceed=1;
    numsat=str2num(str(1:2));
    if(numsat > 15)
      disp('too many satellites');
      proceed=0;
      end
  end
sats=zeros(numsat,1);
if(proceed == 1)
  run_char=start_char;
  for i=1:numsat
    sats(i)=str2num(str(run_char+1:run_char+2));
    run_char=run_char+3;
    end
  end

numsat
sats

%disp('pause, press a key to continue');
%pause

% textscan returns cell arrays which are columns from the file
% for 2011 data we want the 3rd column which is C1
% 2011 file has 5 observations (was 6 before eliminating one)
S=textscan(fido,'%f %f %f %f %f',numsat);
c1=S{3};
fclose(fido);
%disp('have read the satellite file');
```

```
%pause

fids=fopen('epoch1s.txt','rt');
% interpret first line of satellite file
S=textscan(fids,'%s %d %d %d %d %d %f',1);
% interpret the satellite data
S=textscan(fids,'%s %f %f %f %f %f',31);
fclose(fids);
tsat=S{2};
XX=S{3};
YY=S{4};
ZZ=S{5};
tdt=S{6};
Xs=zeros(numsat,1);
Ys=zeros(numsat,1);
Zs=zeros(numsat,1);
dt=zeros(numsat,1);
for i=1:numsat
  % last year this number was 31, this year it is 30
  % seems satellite #25 missing
  % now (2011) it is 32, why not just use variable "numsat" ??
  % now (fall, 2011) satellite #24 is missing, so we are back to 31
  for j=1:31
    if(tsat(j) == sats(i))
      Xs(i)=XX(j);
      Ys(i)=YY(j);
      Zs(i)=ZZ(j);
      dt(i)=tdt(j);
      end
    end
  end
[m,n]=size(Xs);
m
numsat
%disp('# matches, # from observations');
%pause

if(numsat ~= m)
  disp('error in satellite counting');
  pause
  end

% edit out any sats with invalid dt and also corresponding obs
%[c1 Xs Ys Zs dt]
keep_going=1;
while (keep_going==1)
  change=0;
  for i=1:numsat
    if(dt(i) > 999999.0)
      c1(i)=[];
      Xs(i)=[];
      Ys(i)=[];
      Zs(i)=[];
      dt(i)=[];
      change=1;
      break
      end
    end
  if(change == 1)
    numsat=numsat-1;
  else
    keep_going=0;
```

```
    end
  end

disp('after editing out defective sats');
numsat
disp('c1 xs ys zs dt');
[c1 Xs Ys Zs dt]
%pause

nobs=numsat;

%disp('ok check variables');
%pause

npar=4;
n=nobs;
n0=npar;
r=n-n0;
% Xs,Ys,Zs (km), c1 (m), DT(us)
% we solve in km & us

rawpr=c1/1000; % convert to km
DT=dt;
W=eye(nobs);
sigma0=0.025;

% units
% c1 (unrefined pseudorange) is in meters in file
% xs,xs,zs in km
% DT 1e-06 sec, i.e. us or microseconds
c=0.299792458; % km/us (km / u-second)
pr=rawpr + c*DT;
% initial approximations to receiver coords
% found in the observation file header
% convert from given m to km
Xo= 260604.0/1000;
Yo= -4853018.0/1000;
Zo= 4116926.0/1000;
%Xo=0.0;
%Yo=0.0;
%Zo=0.0;
rdt=0.0;

% compute matrices for the XYZ -> enu conversion
% XYZ -> phi,lambda,h
% note geo2xyz.m and xyz2geo.m require METERS not KM !!
[P,L,h]=xyz2geo(Xo*1000,Yo*1000,Zo*1000);
ax=pi/2-P;
Mx=[1 0 0;0 cos(ax) sin(ax);0 -sin(ax) cos(ax)];
az=L + pi/2;
Mz=[cos(az) sin(az) 0;-sin(az) cos(az) 0;0 0 1];
M=Mx*Mz;

old_phi=9.99e+09;
threshold=1.0e-06;
converged=0;
elev_thresh=10.0;
for iter=1:10
  n_actual_obs=0;
  B=zeros(nobs,npar);
  f=zeros(nobs,1);
  for i=1:nobs
```

```
    enu=M*[Xs(i)-Xo; Ys(i)-Yo; Zs(i)-Zo];
    elev=atan(enu(3)/(sqrt(enu(1)^2 + enu(2)^2)));
    elevd=57.29577951*elev;
    if(iter == 1)
      disp('#   sat. id');
      [i sats(i)]
      disp('east, north, up of satellite for elev. angle check');
      enu'
      elevd
      end
    if(elevd > elev_thresh)
      % only use this observations if above elevation angle mask
      % otherwise ignore it
      % increment the actual observation count
      n_actual_obs=n_actual_obs + 1;
      D=sqrt((Xs(i)-Xo)^2 + (Ys(i)-Yo)^2 + (Zs(i)-Zo)^2);
      B(i,1)=(Xs(i)-Xo)/D;
      B(i,2)=(Ys(i)-Yo)/D;
      B(i,3)=(Zs(i)-Zo)/D;
      B(i,4)=-c;
      F=pr(i) - D - c*rdt;
      f(i)=-F;
    else
      disp('reject low satellite number');
      i
      end
    end
  %B
  %f
  %pause
  % W
  %condJ=cond(B)
  N=B'*W*B;
  condN=cond(N)
  t=B'*W*f;

  %keyboard

  Ni=inv(N);
  del=Ni*t;
  Qdd=Ni;
  Xo=Xo + del(1);
  Yo=Yo + del(2);
  Zo=Zo + del(3);
  rdt=rdt + del(4);
  disp_del=[iter; del(1); del(2); del(3); del(4)]
  v=f-B*del;
  phi=v'*W*v;
  if(abs(phi-old_phi)/phi < threshold)
    disp('we have converged');
    converged=1;
    break;
    end

  old_phi=phi;
  end

if(converged == 0)
  disp('we did not converge');
else
  % we converged
  % show results
```

```
  disp('receiver location (km)');
  [Xo Yo Zo]
  disp('receiver clock bias (usec)');
  [rdt]
  end
disp('residuals (km)');
v
rms=sqrt(v'*v/nobs)

% make global test and post adjustment statistics
% careful - you must modify redundancy if you reject satellites
% low on the horizon

r
test_stat=v'*W*v/(sigma0^2)
alpha = 0.05;
cv1=icdf('chi2',alpha/2,r)
cv2=icdf('chi2',1-alpha/2,r)
if((test_stat > cv1) && (test_stat < cv2))
  Sdd=sigma0^2 * Qdd;
  pass=1;
  disp('pass global test at alpha = 0.05');
else
  sigma0_hat_sqr=v'*W*v/r;
  Sdd=sigma0_hat_sqr * Qdd;
  pass=0;
  disp('fail global test at alpha = 0.05');
  end

% ok transform cofactor matrix to ENU from XYZ
% first get lat, lon, h from XYZ by iteration
% remember Xo,Yo,Zo are km

disp('convert XYZ to phi,lam,h');
lam=atan2(Yo,Xo);
a=6378137.0/1000;
f=1/298.257223563;
esqr=2*f-f^2;
e=sqrt(esqr);
phioo=atan(Zo/((1-e^2)*sqrt(Xo^2+Yo^2)))
keep_going=1;
while(keep_going == 1)
  N=a/sqrt(1-esqr*(sin(phioo))^2)
  new_phi=atan((Zo/sqrt(Xo^2 + Yo^2))*(1 + esqr*N*sin(phioo)/Zo))
  if(abs(new_phi - phioo) < 1.0e-06);
    keep_going=0;
    end
  phioo=new_phi
  end
N=a/sqrt(1-esqr*(sin(phioo))^2);
phi=phioo;
h=sqrt(Xo^2 + Yo^2)/cos(phi) - N;
disp('phi');
result=raddms(phi)
disp('lambda');
result=raddms(lam)
disp('h');
h

% rotate XYZ into enu
% bvg uses "enu" for local cartesian and ENH for map projections
% extract the submatrix for XYZ to transform
```

```
M=m1(pi/2 - phi)*m3(lam + pi/2);
J=M;
J
Sdd_enu=J*Sdd(1:3,1:3)*J';
disp('cov enu');
mx=Sdd_enu(1:2,1:2)
sigz_sqr=Sdd_enu(3,3);
[V,D]=eig(mx);
if(D(1,1) < D(2,2))
  tmp=D(1,1);
  D(1,1)=D(2,2);
  D(2,2)=tmp;
  tmp=V(:,1);
  V(:,1)=V(:,2);
  V(:,2)=tmp;
  end

% 90%
P=0.90;
alpha=1-P;
if(pass == 1)
  alen=sqrt(D(1,1)*icdf('chi2',P,2))
  blen=sqrt(D(2,2)*icdf('chi2',P,2))
  theta=atan2(V(2,1),V(1,1))
  plot_ell(0,0,alen,blen,theta);
  title('90% confidence ellipse for e,n - pass global test');
  zfact=icdf('norm',1-alpha/2,0,1)
  half_width=zfact*sqrt(sigz_sqr)
  width=2*half_width
else
  alen=sqrt(D(1,1)*2*icdf('f',P,2,r))
  blen=sqrt(D(2,2)*2*icdf('f',P,2,r))
  theta=atan2(V(2,1),V(1,1))
  plot_ell(0,0,alen,blen,theta);
  title('90% confidence ellipse for e,n - fail global test');
  zfact=icdf('t',1-alpha/2,r)
  half_width=zfact*sqrt(sigz_sqr)
  width=2*half_width
  end
axis equal
```

```
gps_nav53a
numsat =
     8
sats =
    16
    13
    23
    10
     6
     7
    19
     3
numsat =
     8
sats =
    16
    13
    23
    10
     6
     7
    19
     3
numsat =
     9
sats =
    16
    13
    11
    23
    28
     6
     7
    19
     3
nobs =
    25
found no observations to throw out
after editing out defective sats
nobs =
    25
display c1  xs  ys  zs  dt
ans =
  Columns 1 through 3
              21287141.34               9671.17665           -11273.286067
              20287554.48             -5734.807399           -20258.741892
              20605103.28             -1126.552213           -25140.371937
              24293518.04            -22017.451052             -5852.11655
              21364948.06             14140.406396           -14804.666272
               22499418.9            -15186.548693            -8878.439153
              21662186.34              8347.789114           -24844.425342
              20253866.26             11027.953133           -19286.137311
              22105196.56             13581.808579            -8214.458315
              20476109.34             -4857.487454           -23346.182827
              21408621.52              -450.357415           -26256.949516
              24797214.06            -23866.148357            -7811.771947
              21822514.44             14311.668857           -10596.944933
               21648563.4            -10869.754054           -10898.301998
              20909935.44              8669.138898           -23462.439265
              20365537.74             11663.989807           -15502.864843
               23021211.5              17521.44413            -5769.712404
              21076358.28             -4401.494092           -25383.444655
```

```
              24209534.96              10059.980919             -23028.026064
              22497845.42                118.087873             -26137.930493
              24951548.94             -23182.08437              -13171.392156
              22605053.64              14916.897496                -5946.74466
              20920105.78              -6602.066393              -13489.883887
               20542044.8               9046.355443              -20922.243207
               20897918.8              12741.361601               -11095.36145
  Columns 4 through 5
             22027.954669               -200.884996
             16005.394146                248.828766
              7915.839784                271.527222
             13742.241308                -14.912353
             16924.338317                -58.414824
              19985.16409                 29.497196
              2801.519349               -189.641841
             13886.297212                775.502939
             21357.257239               -200.888751
             11474.898571                  248.8252
               2331.17586                271.521772
              8997.979116                -14.913646
             19770.511792                     -58.4
             21684.257488                 29.500961
              8340.227951               -189.649224
             17563.988766                775.511615
             19230.899452               -200.892787
              6149.175722                248.822033
             -8412.018346               -190.787419
             -3416.760203                271.516024
              1904.008804                 71.968727
             21269.077697                 -58.38542
             21902.536054                  29.50463
             13294.801402               -189.656602
             19983.866332                775.521111
press key to continue
#  epoch  sat. id
ans =
     1     1    16
enu sat coords, elev ang
ans =
          9052.76533048471          9140.70388902187          16884.2719688828
elevd =
           52.694616383131
#  epoch  sat. id
ans =
     2     1    13
enu sat coords, elev ang
ans =
         -6812.87323234033         -727.888551905959          19174.7724344859
elevd =
           70.336864806318
#  epoch  sat. id
ans =
     3     1    23
enu sat coords, elev ang
ans =
         -2473.01125170978         -10206.5150230189          17822.5684034722
elevd =
           59.4915299610946
#  epoch  sat. id
ans =
     4     1    10
enu sat coords, elev ang
```

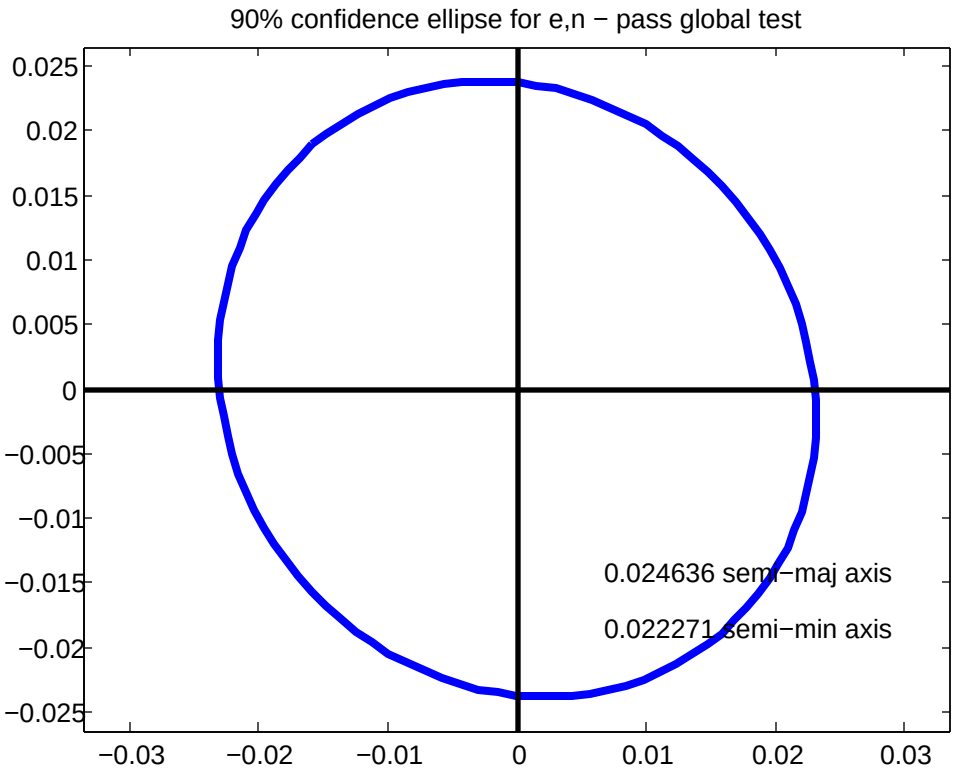
```
ans =
          -22299.577380379           7451.52569989327           6095.92019233292
elevd =
          14.5351729699529
#  epoch  sat. id
ans =
     5     1     6
enu sat coords, elev ang
ans =
          13326.2051335967           2813.76579120993           16438.0353737534
elevd =
          50.3559565824672
#  epoch  sat. id
ans =
     6     1     7
enu sat coords, elev ang
ans =
         -15640.7804486962           10003.0547395423           12724.9520507625
elevd =
          34.4264003216796
#  epoch  sat. id
ans =
     7     1    19
enu sat coords, elev ang
ans =
          7003.56861856772          -14235.8109717316           14665.6379957201
elevd =
           42.749785202393
#  epoch  sat. id
ans =
     8     1     3
enu sat coords, elev ang
ans =
           9977.9238214494           -2293.3252966074           17744.6625657583
elevd =
          60.0164812847079
#  epoch  sat. id
ans =
     9     2    16
enu sat coords, elev ang
ans =
          13121.7918327613           10476.2972687038           14284.5569862263
elevd =
          40.3888833650843
#  epoch  sat. id
ans =
    10     2    13
enu sat coords, elev ang
ans =
         -6102.37058774596          -6206.11604978534           18616.5776283423
elevd =
          64.9427153072618
#  epoch  sat. id
ans =
    11     2    23
enu sat coords, elev ang
ans =
         -1857.66254204288          -15202.8105716079           15074.6905321948
elevd =
          44.5452806306833
#  epoch  sat. id
ans =
```

```
    12     2    10
enu sat coords, elev ang
ans =
         -24250.6958228289          2636.25333472451          4430.91460083473
elevd =
          10.2951165990729
#  epoch  sat. id
ans =
    13     2     6
enu sat coords, elev ang
ans =
          13722.8480991247          7699.82859420493          15094.8980799137
elevd =
          43.8097827842802
#  epoch  sat. id
ans =
    14     2     7
enu sat coords, elev ang
ans =
         -11438.5056927837          9836.88167294048          15538.2692441828
elevd =
          45.8450785992012
#  epoch  sat. id
ans =
    15     2    19
enu sat coords, elev ang
ans =
          7398.56108708732         -9137.21843862679          17222.7416912608
elevd =
          55.6807977273912
#  epoch  sat. id
ans =
    16     2     3
enu sat coords, elev ang
ans =
           10815.912484779          2934.22943639479          17282.5425621228
elevd =
          57.0386237136445
#  epoch  sat. id
ans =
    17     3    16
enu sat coords, elev ang
ans =
          17186.8518649953          10305.3966936034          11208.0348897423
elevd =
          29.2179314449479
#  epoch  sat. id
ans =
    18     3    13
enu sat coords, elev ang
ans =
         -5756.27554302217         -11594.2886384276          16727.2645411605
elevd =
          52.2650975877687
#  epoch  sat. id
ans =
    19     3    11
enu sat coords, elev ang
ans =
          8810.69625603638         -21650.6274970878          6079.07172341916
elevd =
           14.577985478495
```

```
#  epoch  sat. id
ans =
    20     3    23
enu sat coords, elev ang
ans =
         -1283.65302521722         -19518.9759524361          11277.6793342188
elevd =
          29.9649488368088
#  epoch  sat. id
ans =
    21     3    28
enu sat coords, elev ang
ans =
         -23855.0101476307          -6258.0253769785          3927.79815749533
elevd =
          9.04914746594903
reject satellite too low, #
i =
    21
#  epoch  sat. id
ans =
    22     3     6
enu sat coords, elev ang
ans =
          14576.5595498052          11832.1270431225          12558.8282180696
elevd =
          33.7800012499309
#  epoch  sat. id
ans =
    23     3     7
enu sat coords, elev ang
ans =
         -7315.92404657428          8175.24210964246           17823.083180511
elevd =
          58.3862082929748
#  epoch  sat. id
ans =
    24     3    19
enu sat coords, elev ang
ans =
          7911.44560501917         -3734.55926023216          18523.0915248167
elevd =
          64.7182843111287
#  epoch  sat. id
ans =
    25     3     3
enu sat coords, elev ang
ans =
          12128.0738989628          7593.85306249333          15547.9637479794
elevd =
          47.3755708130742
disp_del =
                         1
         0.0350637592224287
        -0.0042168205891990
         0.0111656415605083
disp_del2 =
         0.0277018313953836     -0.000245000957323688       0.00626409559606281
reject satellite too low, #
i =
    21
disp_del =
```

```
                         2
    -9.43920348229025e-009
    -1.81159965780628e-008
     3.05933441911158e-008
disp_del2 =
     9.08167720975492e-009     1.00483116226525e-008    -1.12043524979485e-008
reject satellite too low, #
i =
    21
disp_del =
                         3
      4.6118886124417e-013
    -2.56353202256107e-012
     1.96981333253682e-012
disp_del2 =
     7.78562323968246e-012     8.34202517301392e-012     2.44768091340789e-012
we have converged
receiver location (km)
ans =
          260.639063749784         -4853.02221683871          4116.93716567216
receiver clock biases (usec)
ans =
        0.0277018404848465      -0.00024499090067004       0.00626408439415799
residuals (km) & elevation angle (deg)
ans =
         0.020685495191302          52.6946564362566
       0.00275504279197603          70.3367606233464
        0.0331140763891361          59.4914831722901
        0.0112591013853823          14.5351271626833
      0.000630164188899284          50.3560085234864
       -0.0280950963439292          34.4263367252924
       -0.0357002160295635          42.7497823726633
       -0.0046485675732031          60.0165443449799
        0.0151696439170647           40.388910405275
       0.00765944618913502          64.9426314973891
        0.0355388545393485          44.5452417314939
        0.0106801747517191          10.2950755062754
       0.00391418475905088           43.809819487134
       -0.0347167610050918           45.845012266326
       -0.0357868924408598          55.6808209884346
      -0.00245865071036665          57.0386868679011
        0.0175531045087651          29.2179447590705
        0.0231834534770693          52.2650360338658
       -0.0572258834366306          14.5779635202827
        0.0457940440673544          29.9649140815365
        0.0161309556231242          33.7800183340489
       -0.0305736770405597          58.3861450547179
       -0.0231737870003826          64.7183447756052
       0.00831178980126003          47.3756126360454
rms =
        0.0257684933192967
elevation threshold and rejections
ans =
    10     1
new number of observations
nobs =
    24
r =
    18
test_stat =
          25.4981855211509
cv1 =
```

```
          8.23074619475666
cv2 =
          31.5263784403866
pass global test
convert XYZ to phi,lam,h
phioo =
         0.706124683635765
N =
          6387.14515115495
new_phi =
         0.706124581013248
phioo =
         0.706124581013248
phi
result =
                        40
                        27
            28.649889009809
lambda
result =
                       -86
                       -55
          -32.862061950857
h
h =
         0.197762023461109
J =
           0.99856091584328        0.0536292583412191                         0
        -0.0347994848043434         0.647956106273267           0.7608823037767
         0.0408055536365026        -0.759787330108207         0.648889913467193
alen =
         0.0246356432552529
blen =
         0.0222712398560352
theta =
           2.21036849452655
zfact =
          1.64485362695147
half_width =
         0.0428091983363348
width =
         0.0856183966726697
diary off
```


90% confidence ellipse for e,n − pass global test
0.025
0.02
0.015
0.01
0.005
0
−0.005
−0.01
−0.015
−0.02
−0.025
−0.03
−0.02
−0.01
0
0.01
0.02
0.03
0.024636 semi−maj axis
0.022271 semi−min axis

```
% gpsnav53a.m 26-nov-2011
% this will be the 3-epoch solution

% 3 epochs extracted from NGS/CORS RINEX files
obsfile=['epoch1.txt'; 'epoch2.txt'; 'epoch3.txt'];
satfile=['epoch1s.txt'; 'epoch2s.txt'; 'epoch3s.txt'];
nobs=0;

for k=1:3
  % maybe need an epoch = 1,2,3,4 variable for multiple epoch case
  fido=fopen(obsfile(k,:),'rt');

  % interpret satellite prn's in the observation file
  S=textscan(fido,'%d %d %d %d %d %f %d %s',1);
  str=char(S{8});
  ck1=double(str(1));
  ck2=double(str(2));
  start_char=0;
  proceed=0;
  if((ck1 >= 48) & (ck1 <= 57))
    start_char=2;
    end
  if((ck2 >= 48) & (ck2 <= 57))
    start_char=3;
    end
  switch start_char
    case 0
      disp('cannot interpret satellite string');
      proceed=0;
    case 2
      proceed=1;
      numsat=str2num(str(1));
    case 3
      proceed=1;
      numsat=str2num(str(1:2));
      if(numsat > 15)
        disp('too many satellites');
        proceed=0;
        end
    end
  sats=zeros(numsat,1);
  if(proceed == 1)
    run_char=start_char;
    for i=1:numsat
      sats(i)=str2num(str(run_char+1:run_char+2));
      run_char=run_char+3;
      end
    end

  numsat
  sats

  % multi-epoch satellite ID
  if(k==1)
    totsats=sats;
  else
    totsats=[totsats;sats];
    end

  %disp('pause, press a key to continue');
  %pause
```

```
S=textscan(fido,'%f %f %f %f %f',numsat);
tc1=S{3};
fclose(fido);

fids=fopen(satfile(k,:),'rt');
% interpret first line of satellite file
S=textscan(fids,'%s %d %d %d %d %d %f',1);
% interpret the satellite data
S=textscan(fids,'%s %f %f %f %f %f',31);
fclose(fids);
tsat=S{2};
XX=S{3};
YY=S{4};
ZZ=S{5};
TT=S{6};
tXs=zeros(numsat,1);
tYs=zeros(numsat,1);
tZs=zeros(numsat,1);
tdt=zeros(numsat,1);
for i=1:numsat
  % last year this number was 31, this year it is 30
  % seems satellite #25 missing
  % 2011 seems to be 32 this year
  % fall 2011 sat #24 missing back to 31
  for j=1:31
    if(tsat(j) == sats(i))
      tXs(i)=XX(j);
      tYs(i)=YY(j);
      tZs(i)=ZZ(j);
      tdt(i)=TT(j);
      end
    end
  end
[m,n]=size(tXs);
if(numsat ~= m)
  disp('error in satellite counting');
  pause
  end
nobs=nobs+numsat;
tepch=ones(numsat,1);
tepch=tepch*k;
% transfer data into big arrays
% k is epoch counter 1,2,3, .....
% variable "epoch" will look like: [1;1;1;1;1;2;2;2;2;2;2;3;3;3;3;3]
% i.e. an epoch attribute for each observation
if(k == 1)
  epoch=tepch;
  c1=tc1;
  Xs=tXs;
  Ys=tYs;
  Zs=tZs;
  dt=tdt;
else
  epoch=[epoch; tepch];
  c1=[c1; tc1];
  Xs=[Xs; tXs];
  Ys=[Ys; tYs];
  Zs=[Zs; tZs];
  dt=[dt; tdt];
  end
end
```

```
nobs

% edit out any sats with invalid dt and also corresponding obs
%[c1 Xs Ys Zs dt]
keep_going=1;
while (keep_going==1)
  change=0;
  for i=1:nobs
    if(dt(i) > 999999.0)
      c1(i)=[];
      Xs(i)=[];
      Ys(i)=[];
      Zs(i)=[];
      dt(i)=[];
      epoch(i)=[];
      totsats(i)=[];
      change=1;
      break
      end
    end
  if(change == 1)
    nobs=nobs-1
    disp('threw one out');
  else
    keep_going=0;
    disp('found no observations to throw out');
    end
  end

disp('after editing out defective sats');
nobs
disp('display c1  xs  ys  zs  dt');
[c1 Xs Ys Zs dt]
disp('press key to continue');
pause

%disp('ok check variables');
%pause

npar=3 + 3;
%n=nobs;
%n0=npar;
%r=n-n0;
% Xs,Ys,Zs (km), c1 (m), DT(us)
% we solve in km & us
rawpr=c1/1000; % convert to km
DT=dt;
%
sigma0=0.025;

% units
% c1 (unrefined pseudorange) is in meters in file
% xs,xs,zs in km
% DT 1e-06 sec, i.e. us or microseconds
c=0.299792458; % km/us (km / u-second)
pr=rawpr + c*DT;
% initial approximations to receiver coords
% found in the observation file header
% convert from given m to km
% these are INWL coords
Xo= 260604.0/1000;
Yo= -4853018.0/1000;
```

```
Zo= 4116926.0/1000;
%Xo=0.0;
%Yo=0.0;
%Zo=0.0;
% fixed station coords for comparison (km)
Xfx=260604.165/1000;
Yfx=-4853018.928/1000;
Zfx=4116926.538/1000;

% compute matrices for the XYZ -> enu conversion
% XYZ -> phi,lambda,h
% note geo2xyz.m and xyz2geo.m require METERS not KM !!
[P,L,h]=xyz2geo(Xo*1000,Yo*1000,Zo*1000);
ax=pi/2-P;
Mx=[1 0 0;0 cos(ax) sin(ax);0 -sin(ax) cos(ax)];
az=L + pi/2;
Mz=[cos(az) sin(az) 0;-sin(az) cos(az) 0;0 0 1];
M=Mx*Mz;

% test M see if correct
% note geo2xyz.m and xyz2geo.m require METERS not KM !!

rdt=[0.0; 0.0; 0.0];
old_phi=9.99e+09;
threshold=1.0e-09;
converged=0;
elev_thresh=10.0; % degrees
for iter=1:10
  ii=0; % counter for obs passing elev threshold
  B=zeros(nobs,npar);
  f=zeros(nobs,1);
  elev_disp=zeros(nobs,1);
  for i=1:nobs
    enu=M*[Xs(i)-Xo; Ys(i)-Yo; Zs(i)-Zo];
    elev=atan(enu(3)/(sqrt(enu(1)^2 + enu(2)^2)));
    elevd=57.29577951*elev;
    if(iter == 1)
      disp('#  epoch  sat. id');
      [i epoch(i) totsats(i)]
      disp('enu sat coords, elev ang');
      enu'
      elevd
      end
    if(elevd > elev_thresh)
    % only use this observations if above elevation angle mask
    % otherwise ignore it
    % ii counts actually used observations
      ii=ii+1;
      elev_disp(ii)=elevd;
      D=sqrt((Xs(i)-Xo)^2 + (Ys(i)-Yo)^2 + (Zs(i)-Zo)^2);
      B(ii,1)=(Xs(i)-Xo)/D;
      B(ii,2)=(Ys(i)-Yo)/D;
      B(ii,3)=(Zs(i)-Zo)/D;
      idx=3+epoch(i);
      B(ii,idx)=-c;
      F=pr(i) - D - c*rdt(epoch(i));
      f(ii)=-F;
    else
      disp('reject satellite too low, #');
      i
      end
    end
```

```
  Btemp=B(1:ii,:);
  ftemp=f(1:ii);
  etemp=elev_disp(1:ii);
  B=Btemp;
  f=ftemp;
  elev_disp=etemp;
  rej=nobs-ii;
  nobs_new=ii;
  W=eye(nobs_new);
  % B
  % f
  % W
  %condJ=cond(B)
  N=B'*W*B;
  %condN=cond(N)
  t=B'*W*f;
  Ni=inv(N);
  del=Ni*t;
  Qdd=Ni;
  Xo=Xo + del(1);
  Yo=Yo + del(2);
  Zo=Zo + del(3);
  rdt(1)=rdt(1) + del(4);
  rdt(2)=rdt(2) + del(5);
  rdt(3)=rdt(3) + del(6);
  disp_del=[iter; del(1); del(2); del(3)]
  disp_del2=[del(4) del(5) del(6)]
  v=f-B*del;
  phi=v'*W*v;
  if(abs(phi-old_phi)/phi < threshold)
    disp('we have converged');
    converged=1;
    break;
    end

  old_phi=phi;
  end

disp('enu displacements from known fixed station coords');
denu=M*[Xo-Xfx;Yo-Yfx;Zo-Zfx]

nobs=nobs_new;
n=nobs;
n0=npar;
r=n-n0;

if(converged == 0)
  disp('we did not converge');
else
  % we converged
  % show results
  disp('receiver location (km)');
  [Xo Yo Zo]
  disp('receiver clock biases (usec)');
  [rdt']
  end
disp('residuals (km) & elevation angle (deg)');
[v elev_disp]
rms=sqrt(v'*v/nobs)
disp('elevation threshold and rejections');
[elev_thresh rej]
disp('new number of observations');
```

```
nobs

% make global test and post adjustment statistics

r
alpha=0.05;
test_stat=v'*W*v/(sigma0^2)
cv1=icdf('chi2',alpha/2,r)
cv2=icdf('chi2',1-alpha/2,r)
if((test_stat > cv1) && (test_stat < cv2))
  Sdd=sigma0^2 * Qdd;
  pass=1;
  disp('pass global test');
else
  sigma0_hat_sqr=v'*W*v/r;
  Sdd=sigma0_hat_sqr * Qdd;
  pass=0;
  disp('fail global test');
  end

% ok transform cofactor matrix to ENU from XYZ
% first get lat, lon, h from XYZ by iteration
% remember Xo,Yo,Zo are km

disp('convert XYZ to phi,lam,h');
lam=atan2(Yo,Xo);
a=6378137.0/1000;
f=1/298.257223563;
esqr=2*f-f^2;
e=sqrt(esqr);
phioo=atan(Zo/((1-e^2)*sqrt(Xo^2+Yo^2)))
keep_going=1;
while(keep_going == 1)
  N=a/sqrt(1-esqr*(sin(phioo))^2)
  new_phi=atan((Zo/sqrt(Xo^2 + Yo^2))*(1 + esqr*N*sin(phioo)/Zo))
  if(abs(new_phi - phioo) < 1.0e-06);
    keep_going=0;
    end
  phioo=new_phi
  %pause
  end
N=a/sqrt(1-esqr*(sin(phioo))^2);
phi=phioo;
h=sqrt(Xo^2 + Yo^2)/cos(phi) - N;
disp('phi');
result=raddms(phi)
disp('lambda');
result=raddms(lam)
disp('h');
h

% rotate XYZ into ENU
% extract the submatrix for XYZ to transform
M=m1(pi/2 - phi)*m3(lam + pi/2);
J=M;
J
Sdd_enu=J*Sdd(1:3,1:3)*J';
mx=Sdd_enu(1:2,1:2);
sigz_sqr=Sdd_enu(3,3);
[V,D]=eig(mx);
if(D(1,1) < D(2,2))
  tmp=D(1,1);
```

```
  D(1,1)=D(2,2);
  D(2,2)=tmp;
  tmp=V(:,1);
  V(:,1)=V(:,2);
  V(:,2)=tmp;
  end

% 90%
P=0.90;
alpha=1-P;
if(pass == 1)
  alen=sqrt(D(1,1)*icdf('chi2',P,2))
  blen=sqrt(D(2,2)*icdf('chi2',P,2))
  theta=atan2(V(2,1),V(1,1))
  plot_ell(0,0,alen,blen,theta);
  title('90% confidence ellipse for e,n - pass global test');
  zfact=icdf('norm',1-alpha/2,0,1)
  half_width=zfact*sqrt(sigz_sqr)
  width=2*half_width
else
  alen=sqrt(D(1,1)*2*icdf('f',P,2,r))
  blen=sqrt(D(2,2)*2*icdf('f',P,2,r))
  theta=atan2(V(2,1),V(1,1))
  plot_ell(0,0,alen,blen,theta);
  title('90% confidence ellipse for e,n - fail global test');
  zfact=icdf('t',1-alpha/2,r)
  half_width=zfact*sqrt(sigz_sqr)
  width=2*half_width
  end
axis equal
```

```
gps_nav51b
numsat =
     8
sats =
    16
    13
    23
    10
     6
     7
    19
     3
m =
     8
numsat =
     8
after editing out defective sats
numsat =
     8
ans =
  Columns 1 through 3
              21287141.34                9671.17665             -11273.286067
              20287554.48               -5734.807399            -20258.741892
              20605103.28               -1126.552213            -25140.371937
              24293518.04              -22017.451052              -5852.11655
              21364948.06               14140.406396            -14804.666272
               22499418.9              -15186.548693             -8878.439153
              21662186.34                8347.789114            -24844.425342
              20253866.26               11027.953133            -19286.137311
  Columns 4 through 5
             22027.954669                -200.884996
             16005.394146                 248.828766
              7915.839784                 271.527222
             13742.241308                -14.912353
             16924.338317                -58.414824
              19985.16409                 29.497196
              2801.519349                -189.641841
             13886.297212                 775.502939
c1 xs ys zs dt
i =
     1
ans =
          9052.76533048471           9140.70388902187           16884.2719688828
elevd =
           52.694616383131
i =
     2
ans =
         -6812.87323234033          -727.888551905959           19174.7724344859
elevd =
           70.336864806318
i =
     3
ans =
         -2473.01125170978          -10206.5150230189           17822.5684034722
elevd =
           59.4915299610946
i =
     4
ans =
          -22299.577380379           7451.52569989327           6095.92019233292
elevd =
```

```
          14.5351729699529
i =
     5
ans =
          13326.2051335967           2813.76579120993          16438.0353737534
elevd =
          50.3559565824672
i =
     6
ans =
         -15640.7804486962           10003.0547395423          12724.9520507625
elevd =
          34.4264003216796
i =
     7
ans =
          7003.56861856772          -14235.8109717316          14665.6379957201
elevd =
           42.749785202393
i =
     8
ans =
           9977.9238214494           -2293.3252966074          17744.6625657583
elevd =
          60.0164812847079
g =
     -0.000237229328675009
disp_del =
                         1
         0.010695376270385
        -0.0588965894481257
         0.0449588094882004
          0.179254604835048
g =
         0.0249177233026189
disp_del =
                         2
         0.0146137952969054
         0.0203558628875574
        0.00853413767083341
        -0.0235287330630057
g =
       0.000398519601232453
disp_del =
                         3
         0.0002132990297104
       0.000367967769449774
       0.000260766036245701
      -0.000260407228760294
g =
     4.50388145625813e-007
disp_del =
                         4
     2.24555731739161e-006
     1.07865011263511e-006
     1.63274677696885e-005
     2.34504948483273e-005
we have converged
receiver location (km)
ans =
          260.629524716154          -4853.05617168014          4116.97977004066
receiver clock bias (usec)
```

```
rdt =
          0.15548891503813
residuals (km)
v =
         0.0170051076080605
        -0.0123083961524231
         0.0296604915340265
         0.0225394704410416
        0.00375251685268497
        -0.0324405149146357
        -0.0226445037116168
       -0.00556417165713793
rms =
         0.0207323561481844
constraint misclosure (meters)
constr_misc_m =
    -1.36324615485073e-006
r =
     5
test_stat =
          5.50183157062602
alpha =
                      0.05
cv1 =
         0.831211613486658
cv2 =
           12.83250199403
pass global test at alpha = 0.05
diary off
```

```
% gps_nav51b.m 15-nov-11
% solve gps pseudorange problem for 1 epoch
% include elevation angle mask
% ok for this one add distance constraint for the receiver position

% note you may need to edit the obs file to replace spaces with
% zeros in satellite number string, and you may need to edit the
% data fields to replace spaces with zeros ??? why are
% spaces there ???

fido=fopen('epoch1.txt','rt');

% interpret satellite prn's in the observation file
S=textscan(fido,'%d %d %d %d %d %f %d %s',1);
% need to edit leading zero into single digit sat number fields
str=char(S{8});
ck1=double(str(1));
ck2=double(str(2));
start_char=0;
proceed=0;
if((ck1 >= 48) & (ck1 <= 57))
  start_char=2;
  end
if((ck2 >= 48) & (ck2 <= 57))
  start_char=3;
  end
switch start_char
  case 0
    disp('cannot interpret satellite string');
    proceed=0;
  case 2
    proceed=1;
    numsat=str2num(str(1));
  case 3
    proceed=1;
    numsat=str2num(str(1:2));
    if(numsat > 15)
      disp('too many satellites');
      proceed=0;
      end
  end
sats=zeros(numsat,1);
if(proceed == 1)
  run_char=start_char;
  for i=1:numsat
    sats(i)=str2num(str(run_char+1:run_char+2));
    run_char=run_char+3;
    end
  end

numsat
sats

%disp('pause, press a key to continue');
%pause

% textscan returns cell arrays which are columns from the file
% for 2011 data we want the 3rd column which is C1
% 2011 file has 5 observations (was 6 before eliminating one)
S=textscan(fido,'%f %f %f %f %f',numsat);
c1=S{3};
fclose(fido);
```

```
%disp('have read the satellite file');
%pause

fids=fopen('epoch1s.txt','rt');
% interpret first line of satellite file
S=textscan(fids,'%s %d %d %d %d %d %f',1);
% interpret the satellite data
S=textscan(fids,'%s %f %f %f %f %f',31);
fclose(fids);
tsat=S{2};
XX=S{3};
YY=S{4};
ZZ=S{5};
tdt=S{6};
Xs=zeros(numsat,1);
Ys=zeros(numsat,1);
Zs=zeros(numsat,1);
dt=zeros(numsat,1);
for i=1:numsat
  % last year this number was 31, this year it is 30
  % seems satellite #25 missing
  % now (2011) it is 32, why not just use variable "numsat" ??
  % now (fall, 2011) satellite #24 is missing, so we are back to 31
  for j=1:31
    if(tsat(j) == sats(i))
      Xs(i)=XX(j);
      Ys(i)=YY(j);
      Zs(i)=ZZ(j);
      dt(i)=tdt(j);
      end
    end
  end
[m,n]=size(Xs);
m
numsat
%disp('# matches, # from observations');
%pause

if(numsat ~= m)
  disp('error in satellite counting');
  pause
  end

% edit out any sats with invalid dt and also corresponding obs
%[c1 Xs Ys Zs dt]
keep_going=1;
while (keep_going==1)
  change=0;
  for i=1:numsat
    if(dt(i) > 999999.0)
      c1(i)=[];
      Xs(i)=[];
      Ys(i)=[];
      Zs(i)=[];
      dt(i)=[];
      change=1;
      break
      end
    end
  if(change == 1)
    numsat=numsat-1;
  else
```

```
    keep_going=0;
    end
  end

disp('after editing out defective sats');
numsat
[c1 Xs Ys Zs dt]
disp('c1 xs ys zs dt');
%pause

nobs=numsat;

%disp('ok check variables');
%pause

npar=4;
n=nobs;
n0=npar-1; % we have a constraint !
r=n-n0;
% Xs,Ys,Zs (km), c1 (m), DT(us)
% we solve in km & us

rawpr=c1/1000; % convert to km
DT=dt;
W=eye(nobs);
sigma0=0.025;

% units
% c1 (unrefined pseudorange) is in meters in file
% xs,xs,zs in km
% DT 1e-06 sec, i.e. us or microseconds
c=0.299792458; % km/us (km / u-second)
pr=rawpr + c*DT;
% initial approximations to receiver coords
% found in the observation file header
% convert from given m to km
Xo= 260604.0/1000;
Yo= -4853018.0/1000;
Zo= 4116926.0/1000;
%Xo=0.0;
%Yo=0.0;
%Zo=0.0;
rdt=0.0;

% compute matrices for the XYZ -> enu conversion
% XYZ -> phi,lambda,h
% note geo2xyz.m and xyz2geo.m require METERS not KM !!
[P,L,h]=xyz2geo(Xo*1000,Yo*1000,Zo*1000);
ax=pi/2-P;
Mx=[1 0 0;0 cos(ax) sin(ax);0 -sin(ax) cos(ax)];
az=L + pi/2;
Mz=[cos(az) sin(az) 0;-sin(az) cos(az) 0;0 0 1];
M=Mx*Mz;

old_phi=9.99e+09;
threshold=1.0e-06;
converged=0;
elev_thresh=10.0;
for iter=1:10
  n_actual_obs=0;
  B=zeros(nobs,npar);
  f=zeros(nobs,1);
```

```
for i=1:nobs
  enu=M*[Xs(i)-Xo; Ys(i)-Yo; Zs(i)-Zo];
  elev=atan(enu(3)/(sqrt(enu(1)^2 + enu(2)^2)));
  elevd=57.29577951*elev;
  if(iter == 1)
    i
    enu'
    elevd
    end
  if(elevd > elev_thresh)
    % only use this observations if above elevation angle mask
    % otherwise ignore it
    % increment the actual observation count
    n_actual_obs=n_actual_obs + 1;
    D=sqrt((Xs(i)-Xo)^2 + (Ys(i)-Yo)^2 + (Zs(i)-Zo)^2);
    B(i,1)=(Xs(i)-Xo)/D;
    B(i,2)=(Ys(i)-Yo)/D;
    B(i,3)=(Zs(i)-Zo)/D;
    B(i,4)=-c;
    F=pr(i) - D - c*rdt;
    f(i)=-F;
  else
    disp('reject low satellite number');
    i
    end
  end
%B
%f
%pause
% W
%condJ=cond(B)

% add constraint equation
% Fc = 100 - [(X-Xf)^2 + (Y-Yf)^2 + (Z-Zf)^2]^(1/2) = 0;
% Xf,Yf,Zf : 260679.818,-4852970.302, 4116969.922 (m)
Xf=260.679818; % (km)
Yf=-4852.970302;
Zf=4116.969922;
D=sqrt((Xo-Xf)^2 + (Yo-Yf)^2 + (Zo-Zf)^2);
% note units here are km !!!
Fc=0.100 - D;
dFcdX=-(Xo-Xf)/D;
dFcdY=-(Yo-Yf)/D;
dFcdZ=-(Zo-Zf)/D;
C=[dFcdX dFcdY dFcdZ 0];
g= -Fc

N=B'*W*B;
%condN=cond(N)
t=B'*W*f;

%keyboard

Nc=[-N C';C 0];
z=[-t; g];
delc=inv(Nc)*z;
Xo=Xo + delc(1);
Yo=Yo + delc(2);
Zo=Zo + delc(3);
rdt=rdt + delc(4);
disp_del=[iter; delc(1); delc(2); delc(3); delc(4)]
del=delc(1:4);
```

```
  v=f-B*del;
  phi=v'*W*v;
  if(abs(phi-old_phi)/phi < threshold)
    disp('we have converged');
    converged=1;
    break;
    end

  old_phi=phi;
  end

if(converged == 0)
  disp('we did not converge');
else
  % we converged
  % show results
  disp('receiver location (km)');
  [Xo Yo Zo]
  disp('receiver clock bias (usec)');
  [rdt]
  end
disp('residuals (km)');
v
rms=sqrt(v'*v/nobs)
disp('constraint misclosure (meters)');
constr_misc=0.100 - sqrt((Xo-Xf)^2 + (Yo-Yf)^2 + (Zo-Zf)^2);
constr_misc_m=constr_misc*1000

% make global test and post adjustment statistics
% careful - you must modify redundancy if you reject satellites
% low on the horizon

r
test_stat=v'*W*v/(sigma0^2)
 alpha = 0.05
cv1=icdf('chi2',alpha/2,r)
cv2=icdf('chi2',1-alpha/2,r)
if((test_stat > cv1) && (test_stat < cv2))
  %Sdd=sigma0^2 * Qdd;
  pass=1;
  disp('pass global test at alpha = 0.05');
else
  sigma0_hat_sqr=v'*W*v/r;
  %Sdd=sigma0_hat_sqr * Qdd;
  pass=0;
  disp('fail global test at alpha = 0.05');
  end
```