```
08 11 19 00 00  0.0000000  0  8G02G10G15G21G24G26G29G30
 21934190.757   115264981.43148  21934186.922    89816879.51849
 20561260.643   108050208.01949  20561258.638    84194961.38949
 22757767.515   119592891.94708  22757766.155    93189272.98709
 24557499.591   129050539.18246  24557497.425   100558849.43347
 21044013.491   110587067.23149  21044012.450    86171747.42549
 21821870.867   114674718.36748  21821869.645    89356933.82549
 20785512.770   109228610.93209  20785510.730    85113209.31209
 23916989.875   125684638.28146  23916988.698    97936084.67147
```

```
08 11 19 00 15  0.0000000  0  8G02G10G15G21G24G26G29G30
 22335216.535   117372384.27748  22335212.906    91459011.13049
 20886657.714   109760181.32449  20886655.687    85527408.15849
 22219344.149   116763460.49808  22219342.583    90984521.81109
 24077305.150   126527103.36247  24077302.615    98592536.64347
 20835258.466   109490052.39449  20835257.340    85316930.80149
 21307910.572   111973841.84649  21307909.189    87252355.48449
 20657170.464   108554167.52909  20657168.430    84587668.95509
 24415802.816   128305915.29146  24415801.696    99978637.64047
```

```
08 11 19 00 30  0.0000000  0  8G02G10G15G18G21G24G26G29
 22774574.198   119681221.91248  22774570.568    93258104.78148
 21266143.624   111754393.49049  21266141.590    87081339.77749
 21736218.416   114224620.75908  21736216.765    89006205.91209
 24568164.828   129106591.00444  24568162.694   100602532.54846
 23596518.486   124000555.08146  23596515.666    96623798.33547
 20665463.070   108597772.34749  20665461.953    84621647.76349
 20881003.629   109730432.56349  20881002.150    85504244.84149
 20623505.296   108377255.06809  20623503.239    84449814.97009
```

```
08 11 19 00 45  0.0000000  0  8G02G10G15G18G21G24G26G29
 23249956.870   122179371.77447  23249953.425    95204714.32848
 21688954.356   113976278.91748  21688952.246    88812679.10649
 21322016.745   112047977.84809  21322015.000    87310120.89109
 24079816.165   126540304.08646  24079813.705    98602829.10947
 23118685.672   121489528.44547  23118682.830    94667154.62548
 20538361.959   107929851.71749  20538360.834    84101190.20949
 20556554.913   108025444.99349  20556553.349    84175683.58149
 20687865.323   108715468.60109  20687863.343    84713357.77309
```

```
*  2008 11 19  0  0  0.00000000
P  2  18274.000655 -13736.677924  13062.263585    172.307750
P  3 -10569.649470  22897.979174   7441.230563    313.016068
P  4  26159.603162  -5626.025282    506.957940   -236.812350
P  5 -13657.857318 -21249.955230  -8703.602236 999999.999999
P  6 -14129.809748  19611.350062  11180.683893    140.796202
P  7  18075.525247   8604.023975  17487.183303     23.276951
P  8  25884.964553    746.968309   6924.972037   -178.506704
P  9  -1266.889298 -15295.971766 -22312.030504     25.556504
P 10   7205.065479 -14220.527889  21048.277812     -7.403156
P 11   4577.748766  16241.770430 -20671.559152     15.301842
P 12  -8904.412033 -22170.197123 -11636.852216   -339.736792
P 13   7729.541969  15704.570428  19862.590557    277.784728
P 14 -16228.291640   6047.446669 -19985.274000   -192.841916
P 15   6063.146051 -25691.861710  -2754.834142   -213.120675
P 16 -12232.997390   8932.432641  21881.569180     94.454063
P 17  16415.590229  -3501.283664 -20498.659628     40.700887
P 18 -19881.163902 -16669.287086  -6316.584927   -111.581352
P 19  -6888.503980  25122.003115  -4487.259522     35.471255
P 20  11725.092308  22138.384831  -9005.210141     98.710755
P 21 -22274.179688  -6217.452696  13807.100853     41.986683
P 22 -20092.983202  -5823.931715 -16236.332084    209.153899
P 23   2362.783259  22469.243377  13693.628847    380.388399
P 24  -8125.886028 -13986.680673  21170.196958    142.140574
P 25  11310.609528  11822.145168  21245.414087     34.725033
P 26  10418.971800 -24016.210044   1644.296356     16.859267
P 27  20616.191809   5994.355496  16390.027532      1.052376
P 28  20325.266555   8087.347593 -14895.664857    -23.551766
P 29  -7638.028827 -17475.289015  18605.433726    -29.329753
P 30 -18413.256096 -19205.083287   -232.447114    108.692691
P 31 -25158.730203   7943.766840   3748.166918    -39.304520
P 32   4411.633002  21963.125352 -14006.693459    313.495297
```

```
*  2008 11 19  0 15  0.00000000
P  2  19695.172725 -13806.179345  10702.603528    172.305549
P  3 -10735.156218  21803.264326   9993.492513    313.020802
P  4  26137.387417  -5333.627570  -2286.700109   -236.823275
P  5 -12660.797443 -20720.674936 -11201.513033 999999.999999
P  6 -14057.633565  18196.889664  13468.210106    140.789466
P  7  16075.031504   9221.360922  19044.700565     23.277172
P  8  24986.290106   1049.411889   9594.044757   -178.507860
P  9   1041.510668 -15909.865015 -21867.086394     25.558201
P 10   8909.954733 -12430.351921  21549.663885     -7.403576
P 11   2426.991650  17164.225522 -20251.205114     15.301005
P 12  -7753.261386 -21235.472396 -13950.263774   -339.734595
P 13   6923.325343  17756.351814  18385.631594    277.786098
P 14 -15552.916498   3776.417222 -21055.620025   -192.837580
P 15   6363.523236 -25770.118595     89.914099   -213.126244
P 16 -13890.339435   7099.832674  21579.439317     94.451909
P 17  15779.732644  -1183.383285 -21263.133354     40.701391
P 18 -20072.460882 -17283.540549  -3588.646798   -111.577896
P 19  -7277.546690  25362.379654  -1655.051244     35.471091
P 20  10724.154233  21499.720827 -11486.109856     98.709814
P 21 -20709.005436  -6760.964405  15814.960670     41.985253
P 22 -21104.628310  -7296.495912 -14262.720518    209.153468
P 23   1753.686018  23796.656921  11324.603204    380.389430
P 24  -5754.709243 -14492.174052  21630.371038    142.143826
P 25   9067.753057  12654.486319  21858.794395     34.752254
P 26  10475.755238 -23555.825595   4566.354050     16.863271
P 27  18877.387137   6650.269822  18199.367315      1.055413
P 28  21282.772961   9415.653970 -12703.346023    -23.551763
P 29  -6888.816932 -19352.038080  16961.921913    -29.328017
P 30 -18113.294446 -19301.177015  -3053.875957    108.694917
P 31 -25445.226633   7790.780917    895.595439    -39.306642
P 32   3162.831184  20683.009794 -16060.150868    313.493945
```

```
*  2008 11 19  0 30  0.00000000
P  2  20844.487779 -13859.723248   8153.800517    172.303333
P  3 -10917.551285  20437.279090  12368.148364    313.025560
P  4  25832.110777  -4952.910563  -5041.939739   -236.834230
P  5 -11427.355354 -20077.293548 -13510.546542 999999.999999
P  6 -14007.818285  16542.233617  15526.249166    140.782737
P  7  13958.270093   9990.329018  20274.981928     23.277387
P  8  23823.105068   1474.238612  12102.099054   -178.508994
P  9   3249.731857 -16629.908869 -21066.396142     25.559927
P 10  10728.210313 -10707.750617  21674.558664     -7.403998
P 11    370.402464  18144.536808 -19486.170589     15.300181
P 12  -6371.985701 -20192.821075 -16023.704360   -339.732402
P 13   6287.204681  19670.660646  16589.456523    277.787448
P 14 -14980.264354   1373.714057 -21759.048030   -192.833248
P 15   6590.807678 -25549.032908   2933.114155   -213.131830
P 16 -15602.261515   5397.814543  20908.795892     94.449760
P 17  15248.948700   1226.628118 -21659.460249     40.701888
P 18 -20076.325315 -17667.131277   -800.088412   -111.574461
P 19  -7558.205100  25312.727182   1206.046972     35.470925
P 20   9493.617538  20732.656410 -13770.888265     98.708859
P 21 -18970.472988  -7438.790711  17561.386715     41.983813
P 22 -22019.788783  -8547.201275 -12041.876972    209.153027
P 23   1279.896293  24880.649601   8757.396848    380.390449
P 24  -3385.091235 -15124.857150  21722.711576    142.147159
P 25   6830.876798  13624.333387  22107.505706     34.779462
P 26  10470.608194 -22776.433165   7406.320594     16.867278
P 27  17008.456978   7463.472860  19712.266791      1.058384
P 28  22121.983445  10503.331026 -10291.218227    -23.551753
P 29  -6287.884698 -21070.309539  15029.112989    -29.326284
P 30 -17582.085367 -19205.493426  -5823.275847    108.697097
P 31 -25431.425651   7555.983747  -1972.256902    -39.308763
P 32   1697.427288  19306.428682 -17831.402608    313.492575
```

```
*  2008 11 19  0 45  0.00000000
P  2  21711.094864 -13860.141606   5460.894892    172.301117
P  3 -11148.307804  18822.238608  14522.701249    313.030348
P  4  25261.632777  -4452.682679  -7712.531777   -236.845193
P  5  -9961.114787 -19359.532856 -15592.292721 999999.999999
P  6 -14011.233620  14673.942764  17320.180718    140.776012
P  7  11772.411832  10913.040555  21156.585629     23.277604
P  8  22426.772116   2048.777828  14406.505491   -178.510104
P  9   5318.840052 -17430.922628 -19921.951515     25.561688
P 10  12625.113688  -9088.563819  21421.788362     -7.404420
P 11  -1560.072878  19154.727125 -18388.396012     15.299366
P 12  -4766.940398 -19085.149716 -17821.194087   -339.730220
P 13   5808.483126  21401.224988  14505.496360    277.788793
P 14 -14532.261685  -1114.123003 -22083.516559   -192.828916
P 15   6781.721046 -25024.458973   5725.763070   -213.137416
P 16 -17328.357076   3852.815177  19881.627393     94.447609
P 17  14839.589215   3682.485266 -21681.296104     40.702388
P 18 -19864.800828 -17842.519777   2001.961931   -111.571028
P 19  -7765.728104  24962.111601   4046.122748     35.470761
P 20   8035.145892  19877.412080 -15820.754580     98.707910
P 21 -17097.617978  -8262.579803  19016.877227     41.982368
P 22 -22797.996108  -9575.331865  -9612.669821    209.152589
P 23    913.661521  25690.703817   6036.855728    380.391466
P 24  -1063.331239 -15874.017913  21446.487260    142.150582
P 25   4644.993858  14716.319144  21988.055404     34.806670
P 26  10444.415491 -21681.267300  10112.534820     16.871291
P 27  15053.849109   8438.821967  20905.241482      1.061285
P 28  22802.787376  11355.406808  -7701.991949    -23.551737
P 29  -5819.336676 -22589.469075  12839.804455    -29.324554
P 30 -16804.658836 -18951.972192  -8493.835652    108.699236
P 31 -25129.506004   7203.376923  -4806.306938    -39.310881
P 32     25.007705  17877.962930 -19287.899797    313.491204
```

```
gps_nav21
sats =
     2
    10
    15
    21
    24
    26
    29
    30
condJ =
          10.2102272897629
condN =
           104.24874130862
disp_del =
                         1
        0.0434820683702974
        0.0259463285748255
       0.00946493011375105
        0.0531775776533494
condJ =
          10.2102555893797
condN =
          104.249319200459
disp_del =
                         2
     5.86535769636854e-008
      3.6730067787779e-008
     5.16812097710618e-008
      -1.697011574629e-007
we have converged
receiver location (km)
ans =
  Columns 1 through 2
          262.047482127024          -4855.0870536347
  Column 3
           4114.3724649818
receiver clock bias (usec)
rdt =
        0.0531774079521919
residuals (km)
v =
        0.0247334732460722
        0.0148439086522564
       -0.0295571581314019
       -0.0371829003325869
       -0.0130621066278135
       -0.0216166281090397
       0.00315393683810729
        0.0586874744644062
rms =
        0.0299590224594122
DOPS in ECEF system
PDOP =
           2.3246252241814
HDOP =
          2.06826325469618
VDOP =
          1.06121135603328
convert XYZ to phi,lam,h
phioo =
         0.705599949005046
N =
```

```
          6387.13404321232
new_phi =
          0.70559986554097
phioo =
          0.70559986554097
phi
result =
                         40
                         25
          40.4195537853496
lambda
result =
                        -86
                        -54
         -37.8970762337678
h
h =
         0.160868566304998
DOPS in ENU system
PDOP =
           2.3246252241814
HDOP =
           1.2262390836704
VDOP =
          1.97489750179081
diary off
```

```
% gpsnav21.m 8-dec-08
% derived from gpsnav20.m
% derived from gpsnav16.m
% solve gps pseudorange problem for 1 epoch
% adapted from brian yentes 2004 solution
% now 2005 problem from bvg & jen-yu han

% maybe need an epoch = 1,2,3,4 variable for multiple epoch case
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);
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

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

S=textscan(fido,'%f %f %f %f',numsat);
c1=S{1};
fclose(fido);

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',32);
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
  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);
if(numsat ~= m)
  disp('error in satellite counting');
  pause
  end
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);

% 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= 262004.0/1000;
Yo= -4855113.0/1000;
Zo= 4114363.0/1000;
%Xo=0.0;
%Yo=0.0;
%Zo=0.0;
rdt=0.0;
old_phi=9.99e+09;
threshold=1.0e-06;
converged=0;
for iter=1:10
  B=zeros(nobs,npar);
  f=zeros(nobs,1);
  for i=1:nobs
    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;
    end
  % 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=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)
disp('DOPS in ECEF system');
PDOP=sqrt(Qdd(1,1) + Qdd(2,2) + Qdd(3,3))
HDOP=sqrt(Qdd(1,1) + Qdd(2,2))
VDOP=sqrt(Qdd(3,3))

% 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
% extract the submatrix for XYZ to transform
M=m1(pi/2 - phi)*m3(lam + pi/2);
J=M;
Qdd_enu=J*Qdd(1:3,1:3)*J';
disp('DOPS in ENU system');
PDOP=sqrt(Qdd_enu(1,1) + Qdd_enu(2,2) + Qdd_enu(3,3))
HDOP=sqrt(Qdd_enu(1,1) + Qdd_enu(2,2))
VDOP=sqrt(Qdd_enu(3,3))
```

```
gps_nav22
sats =
     2
    10
    15
    21
    24
    26
    29
    30
sats =
     2
    10
    15
    21
    24
    26
    29
    30
sats =
     2
    10
    15
    18
    21
    24
    26
    29
sats =
     2
    10
    15
    18
    21
    24
    26
    29
disp_del =
                         1
          265.946267146272
         -5803.08192713671
          4919.50528212298
disp_del2 =
  Columns 1 through 2
          4347.24333144022          4336.23533248361
  Columns 3 through 4
          4330.69449247188          4331.49353821775
disp_del =
                         2
         -6.39316783238721
          919.379796100262
         -780.533957105345
disp_del2 =
  Columns 1 through 2
         -4204.80273276252         -4194.46130793341
  Columns 3 through 4
         -4189.25046752986          -4190.3197749319
disp_del =
                         3
          2.50266595958796
          28.5936005855008
         -24.5741428888577
disp_del2 =
```

```
  Columns 1 through 2
        -142.265099785908         -141.656162230957
  Columns 3 through 4
        -141.245641697804         -140.988348683376
disp_del =
                         4
        0.0043828575592057
        0.0281669602061741
       -0.0245809018849129
disp_del2 =
  Columns 1 through 2
        -0.137466885053919        -0.137043992627453
  Columns 3 through 4
        -0.136754363979066         -0.13644685715736
disp_del =
                         5
     1.04511926283523e-008
     4.49878408854212e-008
    -3.87416497929291e-008
disp_del2 =
  Columns 1 through 2
    -1.83994587945948e-007    -1.83945004302546e-007
  Columns 3 through 4
    -1.84418991244707e-007    -1.84785840022872e-007
we have converged
receiver location (km)
ans =
  Columns 1 through 2
         262.060148141483         -4855.08036344576
  Column 3
         4114.37260118815
receiver clock biases (usec)
ans =
  Columns 1 through 2
        0.0380318227420329       -0.0191818573248935
  Columns 3 through 4
        0.0616286958253219        0.0489675605270463
residuals (km)
v =
        0.0124634796680504
       0.00896137791239973
       -0.0311514759531288
       -0.0297886307559708
      -0.00977725418171759
        -0.026162643224864
       0.00739825702337193
        0.0680568895118591
        0.0113698216229434
       0.00488108121415136
       -0.0293329692806418
       -0.0296255197899284
       -0.0118350011548359
       -0.0244050009749784
        0.0143314432331538
         0.064616145130136
        0.0217221143234424
        0.0119083877151681
        -0.016127239506077
       -0.0176551622710123
       -0.0187214746755974
      -0.00211497014924261
       -0.0115517793442173
         0.032540123907536
```

```
        0.0199276467920816
       0.00682156112681649
       -0.0150629828074057
       -0.0148559436472399
       -0.0200746594888341
      -0.00462577084720703
        -0.010700517279153
        0.0385706661509417
rms =
        0.0251358941830662
DOPS in ECEF system
PDOP =
         1.00967842767553
HDOP =
        0.854763157075842
VDOP =
        0.537429504790223
convert XYZ to phi,lam,h
phioo =
        0.705600574294387
N =
         6387.13405644772
new_phi =
        0.705600493153258
phioo =
        0.705600493153258
phi
result =
                        40
                        25
         40.5490081124668
lambda
result =
                       -86
                       -54
        -37.3452359864393
h
h =
        0.156391178938065
DOPS in ENU system
PDOP =
         1.00967842767553
HDOP =
        0.560661679798866
VDOP =
        0.839707692068162
diary off
```

```
% gpsnav22.m 8-dec-08
% this will be the 4-epoch solution
% derived from gpsnav21.m
% derived from gpsnav16.m
% solve gps pseudorange problem for 1 epoch
% adapted from brian yentes 2004 solution
% now 2005 problem from bvg & jen-yu han

% 4 epochs extracted from NGS/CORS RINEX files
obsfile=['epoch1.txt'; 'epoch2.txt'; 'epoch3.txt'; 'epoch4.txt'];
satfile=['epoch1s.txt'; 'epoch2s.txt'; 'epoch3s.txt'; 'epoch4s.txt'];
nobs=0;

for k=1:4
  % 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

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

  S=textscan(fido,'%f %f %f %f',numsat);
  tc1=S{1};
  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',32);
  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
    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
  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

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

npar=3 + 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);
```

```
% 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= 262004.0/1000;
Yo= -4855113.0/1000;
Zo= 4114363.0/1000;
Xo=0.0;
Yo=0.0;
Zo=0.0;
rdt=[0.0; 0.0; 0.0; 0.0];
old_phi=9.99e+09;
threshold=1.0e-06;
converged=0;
for iter=1:10
  B=zeros(nobs,npar);
  f=zeros(nobs,1);
  for i=1:nobs
    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;
    idx=3+epoch(i);
    B(i,idx)=-c;
    F=pr(i) - D - c*rdt(epoch(i));
    f(i)=-F;
    end
  % 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);
  rdt(4)=rdt(4) + del(7);

  disp_del=[iter; del(1); del(2); del(3)]
  disp_del2=[del(4) del(5) del(6) del(7)]
  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 biases (usec)');
  [rdt']
  end
disp('residuals (km)');
v
rms=sqrt(v'*v/nobs)
disp('DOPS in ECEF system');
PDOP=sqrt(Qdd(1,1) + Qdd(2,2) + Qdd(3,3))
HDOP=sqrt(Qdd(1,1) + Qdd(2,2))
VDOP=sqrt(Qdd(3,3))

% 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
% extract the submatrix for XYZ to transform
M=m1(pi/2 - phi)*m3(lam + pi/2);
J=M;
Qdd_enu=J*Qdd(1:3,1:3)*J';
disp('DOPS in ENU system');
PDOP=sqrt(Qdd_enu(1,1) + Qdd_enu(2,2) + Qdd_enu(3,3))
HDOP=sqrt(Qdd_enu(1,1) + Qdd_enu(2,2))
VDOP=sqrt(Qdd_enu(3,3))
```