Note on Teleseismic Body-Wave Inversion Program

written by M. Kikuchi and H. Kanamori (2003/08/29)

This note gives a brief explanation of the inversion program and relevant data files. The program package contains three different sets of inversion programs as follows:

Set 1: Inversion allowing mechanism changes.
Set 2: Fine tuning of the source time function and slip distribution.
Set 3: Inversion with a fixed fault mechanism.

The above three sets can be used in the following way:

(1) Use Set 1 to determine a gross fault mechanism for a single subevent with a long source duration. If the waveform match is not satisfactory and some different mechanisms are inferred, try more iterations to determine additional subevents.
(2) Use Set 2 to tune the source time functions for subevents derived in (1).
(3) If the mechanism change is not likely to occur (as in the case of subduction zone earthquakes) and the source process still seems complex, try Set 3 to determine a sub-event distribution over a fault plane.

References

Kikuchi,M., & Kanamori,H., Bull. Seism. Soc. Am., 72, 491-506, 1982.
Kikuchi,M., & Kanamori,H., Bull. Seism. Soc. Am., 81, 2335-2350, 1991.
Kikuchi,M., Kanamori,H. & Satake,K., J. Geophys. Res., 98, 15797-15808, 1993.

1. Data Preparation

This program requires an input file (fort.1) prepared in a special format. Although the original programs were written for many different kinds of data, this note is written with the assumption that the broad-band data are most commonly used. Either velocity or displacement data are used in this program.

First, prepare the input data to be placed in fort.1 using the following script, mk_conv.farm.pl (written by Y. Yamanaka, July, 2003) and program, conv.sac.farm.f. Then, edit the appropriate parameters (see below). The 1st line of each PERL program should be modified according to your own environment.

Procedure:

0. Seismic Analysis Code (SAC) must be installed in your computer.
1. Retrieve the waveform data [seed format] with the IRIS-DMC Wilber II tool on Web site of IRIS DMC (http://www.iris.edu/cgi-bin/wilberII_page1.pl). For teleseismic body wave inversions, it is usually enough to choose the stations of the Networks with Code IU and II, but the data from other networks can be added to improve the solution. "BH" or "LH" channels are mainly used.
2. Change waveform data from seed to sac binary by using "rdseed". "rdseed" is a seed reader program written by IRIS and available from IRIS DMC Web site. Select the "R" and "d" options. Sac binary data files (for example, 2003.158.00.38.16.5856.IU.TATO.00.BHZ.D.SAC) and response files (for example, RESP.IU.TATO.00.BHZ) are made in the directory. See IRIS Electronic Newsletter, Volume 1-Number 1: "Feature Article – Location Identifiers" about specification of data file names.
   [see http://www.iris.edu/news/newsletter/vol1no1/page1.htm]

   Example:

```
% rdseed
<< IRIS SEED Reader, Release 4.19.1 >>
Input  File (/dev/nrst0) or 'Quit' to Exit: 030607.seed
Output File (stdout)    :
Volume #  [(1)-N]       :
Options [acCsSpRtde]    : Rd
Summary file (None)     :
Station List (ALL)      :
Channel List (ALL)      :
Network List (ALL)      :
Loc Ids (ALL ["--" for spaces]) :
```

   In order to acquire response information, you may download the seed format data. If you have IRIS RESPONSE Information, you may retrieve Sac binary data directly from IRIS DMC.
3. Make "hypo" file to give the information of hypocenter. A File name must be "hypo". The format of this file is as follows.

```
----------hypo----------
 yymmddhhmm Event-location
 lat lon dep hh mm sec id delmin delmax
--------------------------------------------------
```

   yymmddhhmm: date and time of event
   lat lon dep: Latitude, Longitude, depth for event
   id : velocity or displacement data (0 for displacement, 1 for velocity, 2 for acceleration, 3 for original)
   delmin and delmax: Minimum and maximum distances in degree.

```
      Example:
      ----------------------------------------------------
       0211032212 CENTRAL ALASKA
        63.70 212.30    10 22 12 41 0 30 100
      ----------------------------------------------------
      If you have qed-mail from USGS, you can make this file easely as:
            % cat qed-mail | qed.pl
   4. Run mk_conv.farm.pl script to make response file and i_conv.farm (input
      file for conv.sac.farm.f) as:
            % mk_conv.farm.pl
      In the script, the following parameters should be set by users.
         high_pas_co: Cut-off frequency of the high-pass filter. e.g. 0.002 Hz
         low_pas_co:  Cut-off frequency of the low-pass filter. e.g. 2 Hz
         du:          Duration
         pre_ev:      Duration of the data before the P or S arrival time
   5. Run conv.sac.farm program to make fort.22 as:
            % conv.sac.farm < i_conv.farm
   6. Run rotSH program to make fort.1 as:
            % rotSH
```

Script list:

mk_conv.farm.pl

```
--------------------------------------------------------------------------------
#!/usr/local/bin/perl
# Program to make i_conv.farm & response.data
#      written by Y. YAMANAKA (ERI, Univ. of Tokyo, JAPAN)
#
# Preparation:
#  0. SAC already needs to be installed in your computer.
#  1. Retrieve waveform data [seed format] from IRIS DMC HP (Wilber II)
#  2. Change waveform data from seed to sac binary by using
#      "rdseed" [from IRIS DMC]. Option: Rd
#     Need Response information! If you have IRIS RESPONSE
#      Information, you can retrieve sac binary data from Wilber.
#  3. Make "hypo" file.
#     ----------------------hypo---------------------------
#      yymmddhhmm Event-location
#      lat lon dep hh mm sec id delmin delmax
#     ----------------------------------------------------
#     Example:
#      0306070032 NEW BRITAIN REGION, PAPUA NEW
#       -5.00 152.20    33  0 32 46 2 30 100
#     ----------------------------------------------------
#     If you have qed mail, you can make this file as:
#        % cat qed-mail | qed.pl
#
# +++ set parameters +++
#   high_pass_co: Cut-off frequency of the hygh-pass filter. e.g. 0.002 Hz.
#   low_pass_co:  Cut-off frequency of the low-pass filter. e.g. 1 Hz.
#   du:           Duration
#   pre_ev:       Duration of the data before the P time
#   sample:       Sampling interval (sec)
#

$high_pass_co = 0.002;
$low_pass_co = 1;
$du = 120;
```

```
$pre_ev = 20;
$sample = 1.0;

# file check
if(! -e "hypo" ) {
     print "You need \"hypo\" file in this directory.\n";
     print "Try again after making \"hypo\" file!\n";
     exit(0);
}
if(! -e "sacmacro" ) {
     print "You need \"sacmacro\" file in this directory.\n";
     print "Try again after making \"sacmacro\" file!\n";
     exit(0);
}

@list = `ls *SAC`;

system ("/bin/cp hypo i_conv.farm");
open IN, ">>i_conv.farm";
open RES, ">response";
foreach $ev (@list) {
     chop($ev);
     @info=split('\.',$ev);
     $year=$info[0];
     $day=$info[1];
     if ( $ev =~ /D.SAC/ ) {$n=index($ev,".D.SAC");}
     if ( $ev =~ /Q.SAC/ ) {$n=index($ev,".Q.SAC");}
     $m=$n-23;
     $fname=substr($ev,23,$m);
#  change file name
     system ("/bin/mv $ev $fname ");
}

@list2 = `ls *[B,L]H?`;
foreach $ev (@list2) {
     chop($ev);
     $sname = $ev . ".A";
     $nlen = length($sname);
     $n2=$nlen-6;
     $scode = substr($sname,0,$n2);
     $n1=index($ev,".");
     $net = substr($ev,0,$n1);
     #if ($net =~ 'CD') {$net='IU';}
     $rname= "RESP." . $ev;
#  make response file
     open IRIS, "$rname";
     while (<IRIS>) {
        chop;
        if ($_ =~ /Start date:/ ) {
           $firis=0;$fprint=0;
           $Yst=substr($_,25,4);
           $Dst=substr($_,30,3);
           if ($year > $Yst) {
              $firis++;
           }
           if ($year == $Yst && $day >= $Dst) {
              $firis++;
```

```
   }
}
if ($_ =~ /End date:/ ) {
   if ($_ =~ /No Ending Time/) {
      $Yen=`date +%Y`;
      chop($Yen);
      $Den=365;
      if ($year < $Yen) {
         $firis++;
      }
      if ($year == $Yen && $day <= $Den) {
         $firis++;
      }
   } else {
      $Yen=substr($_,25,4);
      $Den=substr($_,30,3);
      if ($year < $Yen ) {
         $firis++;
      }
      if ($year == $Yen && $day <= $Den) {
         $firis++;
      }
   }
}
if ($_ =~ /B053F05/ ) {$UNI=substr($_,51,25);}
if ($_ =~ /B053F07/ ) {$A0=substr($_,51,25);}
if ($_ =~ /B053F09/ ) {$NZ=substr($_,51,5);}
if ($_ =~ /B053F04/ ) {
   $Sn=substr($_,51,5);
}
if ($_ =~ /B053F14/ ) {
   $NP=substr($_,51,5);
   $fprint = 1;
   if($firis == 2 && $Sn == 1) {
      if($UNI =~ /Displacement/) {$id="0";}
      if($UNI =~ /Velocity/) {$id="1";}
      if($UNI =~ /Acceleration/) {$id="2";}
      print RES "$sname\n";
      print RES "  A0 normalization factor:  $A0\n";
      print RES "  Number of zeroes:         $NZ\n";
      print RES "  Number of poles:          $NP\n";
   }
}
if ($_ =~ /B053F10-13/ && $firis == 2 && $fprint == 1 && $Sn == 1) {
   $tmp = substr($_,11,70);
   print RES "$tmp\n";
}
if ($_ =~ /B053F15-18/ && $firis == 2 && $fprint == 1 && $Sn == 1) {
   $tmp = substr($_,11,70);
   print RES "$tmp\n";
}
if ($_ =~ /Sensitivity/ && $_ =~ /B058F04/) {
   $SE=substr($_,51,15);
   if($firis == 2) {
      print RES "  Sensitivity:              $SE\n";
   }
}
```

```
    }
    close(IRIS);
    #  make i_conv.farm data
    if ($sname !~ /RESP/ ) {
       print IN "$sname\n";
       print IN "$scode\n";
       if ( $ev =~ /.LH/ &&  $sample < 1.0 ) {
          print "sampling interval changed!\n";
          $sample = 1.0;
       }
       if ( $ev =~ /BHZ/ || $ev=~ /LHZ/ ) {print IN "$pre_ev $du $sample
$high_pass_co $low_pass_co 1 1 $id \n";}
       if ( $ev =~ /BHE/ || $ev=~ /LHE/ ) {print IN "$pre_ev $du $sample
$high_pass_co $low_pass_co 2 1 $id \n";}
       if ( $ev =~ /BHN/ || $ev=~ /LHN/ ) {print IN "$pre_ev $du $sample
$high_pass_co $low_pass_co 2 1 $id \n";}
    }
}

print IN "dummy\n";
print IN "dummy\n";
print IN "20 120 .5 .004 1 1 0 1\n";
close(IN); close(RES);
system ("/bin/rm RESP.* ");
system ("/usr/local/bin/sac < sacmacro ");
-------------------------------------------------------------------------------
---------------------------------------
```

"sacmacros" used above

```
-------------------------------------------------------------------------------
---------------------------------------
r *.*.*.??E
w alpha append .A
r *.*.*.??N
w alpha append .A
r *.*.*.??Z
w alpha append .A
-------------------------------------------------------------------------------
```

qed.pl

```
-------------------------------------------------------------------------------
#!/usr/local/bin/perl
#    written by K. TAKANO (ERI, Univ. of Tokyo, JAPAN)
#    revised by Y. YAMANAKA (ERI, Univ. of Tokyo, JAPAN)

require 'timelocal.pl';
$bb = "";
$ifrg=0;

while(<>){
    if( /^the following /i .. /^stations used:/i ) {
        $ifrg=1;
        tr/\n/ / ; s/,/ ,/g ; s/^ *// ; s/  */ /g ; s/normal/33 km/;s/-/ /;
s/shallow/10 km/;
        $bb = $bb . $_ ;
    } elsif ( $bb ne "" ) {
        $_ = $bb; $bb = "";
```

```
        ($YY,$M,$DD,$loc)=/Preliminary hypocenter for earthquake of (\d+) (\w+)
(\d+) ?,? ([ '\w\.]+,? [ '\w\.]+),?.*:/i ;
        $MM=1 if $M=~/jan/i ; $MM=2 if $M=~/feb/i ;
        $MM=3 if $M=~/mar/i ; $MM=4 if $M=~/apr/i ;
        $MM=5 if $M=~/may/i ; $MM=6 if $M=~/jun/i ;
        $MM=7 if $M=~/jul/i ; $MM=8 if $M=~/aug/i ;
        $MM=9 if $M=~/sep/i ; $MM=10 if $M=~/oct/i ;
        $MM=11 if $M=~/nov/i ; $MM=12 if $M=~/dec/i ;
        ($lat,$NS,$lon,$EW)=/latitude (\d+.\d+) degrees (\w+) ?,? ?longitude
(\d+.\d+) degrees (\w+)/i ;
        if($NS =~ /south/i) { $NS = "S";}
        if($NS =~ /north/i) { $NS = "N";}
        if($EW =~ /east/i) { $EW = "E";}
        if($EW =~ /west/i) { $EW = "W";}
        ($hh,$mm,$sec,$dep)=/origin time (\d\d) (\d\d) (\d+.\d) utc ?,? ?depth
(\d+) ?k?m? ?,? /i;
        ($mag)=/magnitude (\d.\d)/i;
        $_ = $loc ; s/ ,/,/g ; $loc = $_;
        printf
"%04d%02d%02d %02d:%02d:%s %6.1f%s %5.1f%s %3d %3.1f%s %s\n",$YY,$MM,$DD,$hh,$mm
,$sec,$lon,$EW,$lat,$NS,$dep,$mag,$mty,$loc ;
        }
}

if($ifrg==1) {
$Y=$YY-2000;
$LAT=$lat; $LON=$lon;
if($NS =~ /S/i) { $LAT = -$lat;}
if($EW =~ /W/i) { $LON = 360-$lon;}
$dname=sprintf("%02d%02d%02d%02d%02d",$Y,$MM,$DD,$hh,$mm);

# Make “hypo” file
$kik = "hypo";
open(KIK, "> $kik");
printf KIK "%s %s\n",$dname,$loc;
printf KIK "%6.2f %6.2f %5.0f %2d %2d %.0f 0 30
100\n",$LAT,$LON,$dep,$hh,$mm,$sec;
close(KIK);
}
```

If one wishes to understand the parameters in fort.1, the following is the structure.

fort.1

Example:

```
Denali, Alaska 02/11/03  disp.  63.74-147.69  10.00 22 12 41.00
DUG    19   2.71  -20.00   0.00   1.00
  122.33  -29.30   31.21  0.081 12.840   40
  1  1  1
  0  0  0
  1.0  1.0
  0.1548e+03  480  0.500
 -0.075 -0.074 -0.071 -0.070 -0.068 -0.066 -0.064 -0.062 -0.061 -0.058
 -0.057 -0.055 -0.052 -0.050 -0.048 -0.046 -0.045 -0.043 -0.042 -0.040
```

Explanation:

Line 1. ID
Line 2. Station Name, the rest of the line is not used in the current program
Line 3. Azimuth, Back Azimuth, Distance, Slowness, G factor, ix0 (# of points before P or S arrival)
Line 4. im (0 for WWSSN, 1 for GDSN, IRIS etc), ib (1 for P, 2 for SV, 3 for SH, 4 for PP), ic (1 for Z, 2 for N, 3 for E, if ib=3, then ic is irrelevant, ib=2, then 1 for vertical, 2 for horizontal)
Line 5. This line is for im=1. izp (# of poles), izz(# of zeros), ip(power of z), for the present application using only displacement or velocity from broad-band data, these 3 constants should be set 0, 0, 0, for displacement, and 0, 0, 1 for velocity.
Line 6. Magnification, Normalization factor
Line 7. Maximum amp., # of points, dt (unit=[cm]/[micro-meter] for im=0/1)

2. General Inversion (use green.f, inversion.f and graphics.f)

The first inversion is to determine the mechanism. This step requires some experience and judgment, depending on the event size, mechanism, etc. It is unrealistic to try to get a very detailed rupture pattern at this stage. It is advisable to use a relatively small number of subevents and a broad source time function.

2.1 green.f

This program computes elementary Green's functions appropriate for the source-station geometries.

Input files:
fort.1: station parameters and data (not used in green.f)
fort 2: structure
stdin: i_green

Output file
fort.3: Green's functions (binary)

fort.2

Example:

```
J-B model
1.0 4.0 3 5.57 3.36 2.65 15.0
          6.50 3.74 2.87 18.0
          8.10 4.68 3.30 0.0
        3 5.57 3.36 2.65 15.0
          6.50 3.74 2.87 18.0
          8.10 4.68 3.30 0.0
        1 6.50 3.74 2.87 0.0
```

Explanation:

```
ID
TQP, TQS,   NL ,(VP (L),VS (L),DEN (L),DEP (L),L=1,NL )
            NL1,(VP1(L),VS1(L),DEN1(L),DEP1(L),L=1,NL1)
            NL2,(VP2(L),VS2(L),DEN2(L),DEP2(L),L=1,NL2)
```

Line 1: Title
Line 2: t* for P, t* for S, and “source” structure
    Vp,Vs= velocity [km/s]; den = density [e3 kg/m**3];
    Dep = thickness [km]
Line 3: “Receiver” structure
Line 4: “PP bounce point” structure

i_green

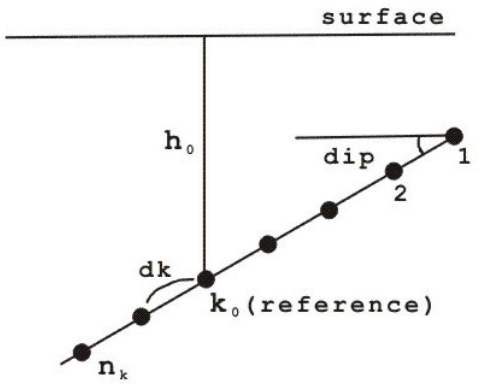
Example:

```
256  0.5  15.0  5.0  3   2   90.
```

Explanation:

```
NT   DT    H0  Dk   Nk  K0  DIP
```

NT (# of points, should be a power of 2), DT (dt, This can be larger than dt of the data. However, it is better to use the same dt as the data to avoid confusion later), H0 (depth of the reference point), Dk, Nk, K0, and DIP (determine the down-dip location of the point sources to be placed on a plane dipping “DIP”. See the figure below)



Note for a deep-focus earthquake

Since the source response function is computed using a frequency domain method, NT*DT should be sufficiently large to include pP and sP even if only the P part of the Green's functions is used for inversion. If NT*DT is too small. the P part of the Green's function may contain spurious signals due to wrap-around of the Fourier transform. To check whether the Green's functions are correctly computed or not use gregra.f to plot the Green's functions.

2-2 inversion.f

This program inverts the data in fort.1 using the Green's functions in fort.3.

Input files:
fort.1:
fort.3:
stdin: i_inversion

Output files:
fort.16: Source parameters determined in the inversion
fort.4: to be used by graphics.f (binary)

i_inversion

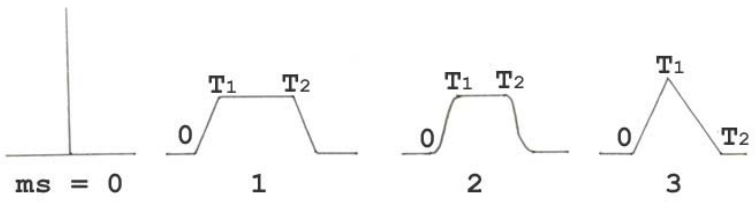
Example:

```
Denali, Alaska 02/11/03
80 4.
1 6. 10.
3
100 100 100
24 .5 1 .0 1 1 1 1 1 1 0 1 .1 1
   .1 .1 .1 .1 .1 .1 .1 .1 .1 .1 .1
-30 1 1
120. 30. 8 1
```

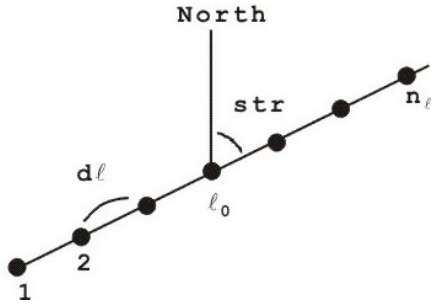
Explanation:

```
ID
Tw, Vr
Ms, T1, T2
  Ne
 (Te(i),i=1,Ne)
 Ns,(Fc(j),j=1,Ns)
 Idly, Isr, Igrid
 (for IFP=0) Nlen, (R(l),l=1,Nlen), (Fi(l),l=1,Nlen)
 (for IFP=1) Str, Dl, Nl, L0
```

Line 1: ID (Event, Run ID etc)
Line 2: Tw (duration of the records to be inverted), Vr(Max. rupture speed)
Line 3: MS (source time function, 0 for impulse, 1 for trapezoid (T2>=T1), for ramp (T2<T1), 2 for cosine-tapered trapezoid (T2>=T1), for cosine-tapered ramp (T2<T1), 3 for triangle



Line 4: Ne(# of iterations, i.e. # of subevents)
Line 5: (Te(i), i=1,Ne) (time intervals for grid search at each iteration
Line 6: Ns,(FC(j), j=1, Ns)(# of phases, weights)
Line 7: Idly (shifts of data in sampling points (if minus, then the records are moved forward), this number refers to the number in the original data, not the number of data points in the Green's functions.), Isr (a parameter that controls the type of source,0 for deviatric moment tensor, 1 for Double Couple, 2 for pure strike-slip, 3 for general moment tensor with isotropic component, 4 for double-couple with a vertical nodal plane), IFP (a parameter that controls the geometry of grid scheme; if IFP=0, then point source locations are given in polar coordinates, if IFP=1, then point source locations are given by azimuth and distance.)
Line 8: if IFP=0
Nlen (# of grid points), R (distance), Fi (azimuth)
Line 8: If IFP=1
Str (strike), Dl (grid spacing), Nl (# of grid points), L0 (index of the reference point (epicenter))



2-3 graphics.f

This program plots the data in fort.4. It requires i_graphics which contains the parameters to control the plot options.

Input files:
fort.4:
stdin: i_graphics

Output file:
plot8: Postscript file

i_graphics

Example:

20.0 3 1.5 0 1.0
1 1 1

Explanation:

Spcm, Norm, Height, Line, Hor
Npl, Nps, Fill

Line 1: Spcm (sec/cm), Norm (a parameter that controls the amplitude normalization, if 0 scale is common for obs. & syn., if 1 each waveform is normalized, if 2 scale is common for every body wave type), Height (Max. Amp. in cm), Line (a parameter that controls base-line, if 1, it draws the base-line), Hor (grid spacing)
Line 2: Npl (a parameter that controls nodal plane plot, if =1, then draw nodal plane, if >1, then draw with T axes), Nps (controls stations plot, if=0, then station not plotted, if >0, stations plotted), fill(fill quadrants, if=0 not filled, >1 fill quadrants)

3. Fixed mechanism inversion (green3.f, inversion3.f, and graphics3.f, mom3.f, gr3.f)

These programs invert the data with a fixed mechanism. The main purpose is to determine the rupture pattern. If, from the result of **2**., no significant change in the mechanism is recognized, this inversion can be used to refine the rupture pattern.

3-1 green3.f

This program computes Green's functions for a fixed mechanism.

Input files:
fort.1: station parameters and data (not used in green3.f)
fort 2: structure
stdin: i_green3

Output file
fort.33: Green's functions (binary)

i_green3

Example:

256 0.5 309. 21. 59. 30. 30. 6 2

Explanation:

NT DT Str Dip Rak H0 Dk Nk K0

NT(# of points), DT(dt), Str(strike), Dip(dip), Rak(rake), H0(depth of the reference point), Dk(grid spacing along dip-direction), Nk(# of grid point), K0(index of the reference point)

Note for a deep-focus earthquake

Since the source response function is computed using a frequency domain method, NT*DT should be sufficiently large to include pP and sP even if only the P part of the Green's functions is used for inversion. If NT*DT is too small. the P part of the Green's function may contain spurious signals due to wrap-around of the Fourier transform. To check whether the Green's functions are correctly computed or not use green.f to compute the Green's functions and plot them with gregra.f.

3-2 inversion3.f

This program inverts the data using the Green's functions computed by green3.f.

Input files:
- fort.1:
- fort.33:
- stdin: i_inversion3

Output file
- fort.36: source parameters etc.
- fort.34: to be used by graphics3.f

i_inversion3 (similar to i_inversion, fewer parameters)

Example:

```
Peru 01/06/23
150 3.5 120
1 5. 8.
15
30. 10 9
-10
22 1 1 1 1 1 1 1 1 1 1 1 1
.2 .2 .2 .2 .2 .2 .2 .2 .2
```

Explanation:

```
ID
Tw, Vr, Te
Ms, T1, T2
Ne
Dl, Nl, L0
Idly
Ns, (Fc(j),j=1,Ns)
```

Line 1: ID(Event, Run ID)
Line 2: Tw(Duration of data), Vr(Max rupture speed), Te(time span for grid search)
Line 3: Ms, T1, T2 (see i_inversion)
Line 4: Ne(# of iteration)
Line 5: Dl, Nl, L0 (see i_inversion)
Line 6: Idly (see i_inversion)
Line 7: Ns,(FC(j),j=1, Ns)(see i_inversion)

3-3 graphics3.f

Input files:
- fort.34
- stdin: i_graphics3

Output file:

plot38: Postscript file

i_graphics3 (similar to i_graphics)

Example

```
20. 3 1.0 1
1 1 1
```

Explanation:

```
Spcm,Norm,Height,Line
Npl,Nps,Fill
```

3.4 mom3.f

This program inverts the data in fort.1 to determine slip distribution on a given fault plane. The programs described above (e.g., green, inversion, green3, inversion3 etc) have all the flexibilities to explore the details and trade-offs of this type of inversion. After having obtained a reasonably good picture of the source process, it is useful to try this program.

The method is similar to the one used by Hartzell and Heaton [1983]. The fault plane is gridded, and the amount of slip and the time history of slip are determined at each grid point. Since the fault plane and the rupture front speed are fixed, the inversion is very stable. The rake angle is constrained within +-45° of the prescribed rake angle. A smoothness constraint can also be imposed.

Input files:

- fort.1: Observed records
- fort.2: structures

Output files:

- fort.24: Output for graphics
- fort.37: Results (Time & Space)
- fort.46: Results (Spatial distribution)
- stdin: i_mom3

i_mom3

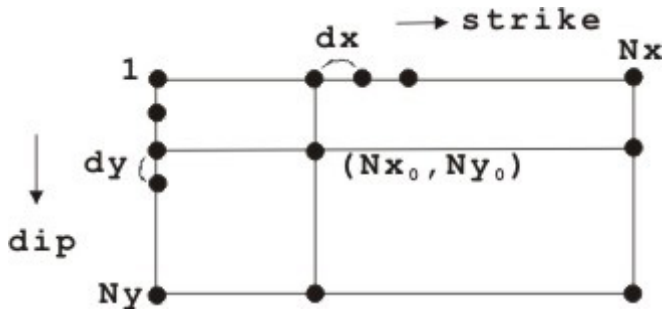
Example:

```
Peru 01/06/23
fort.1
-5. 150. 0.5 30 309. 21. 59.
8 8 30. 5 3 30. 2.5 5. 4.0 4.0 4 0.5 1
22 1 1 1 1 1 1 1 1 1 1 1 1
  .2 .2 .2 .2 .2 .2 .2 .2 .2
256
```

Explanation:

```
ID
Inf
T0, TL, DT, H0, str, dip, slip0
```

```
Nx, Nx0, dx, Ny, Ny0, dy, Vr, Tst, t1, dlt, nw, beta0
Njs,(Fc(j),j=1,Njs)
Nt

Line 1: ID (event, Run ID etc)
Line 2: Inf (file name of observed records, usually fort.1)
Line 3: T0 (Start-time), Tl (Time-window), dt (Sampling-time), h0 (depth of the
     hypocenter), str, dip, slip0 (Strike, Dip, Slip0; Slip is varied between
     Slip0 +- 45)
Line 4: Nx,Nx0,dx,Ny,Ny0,dy (grid scheme = Nx x Ny, reference grid (=
     hypocenter) = (Nx0,Ny0), (dx,dy) = grid spacings for strike and dip
     directions. The grid scheme is shown below), Vr (Rupture front velocity),
     Tst (Rupture start time), t1,dlt,nw ( parameters that specify a series of
     triangular time-functions, t1: half of triangle base, dlt: time-shift, nw:
     number of triangles(*)), beta0 (controls the smoothness)
Line 5: Njs (# of wave-records),Fc(weight factors)
Line 6: Nt(# of data points for each Green's function (power of 2))

(*)The source time function at each grid point is given by a staggered
(shift=dlt) superposition of nw triangles with the base=2*t1.  The amplitude of
each triangle is determined by inversion.
```

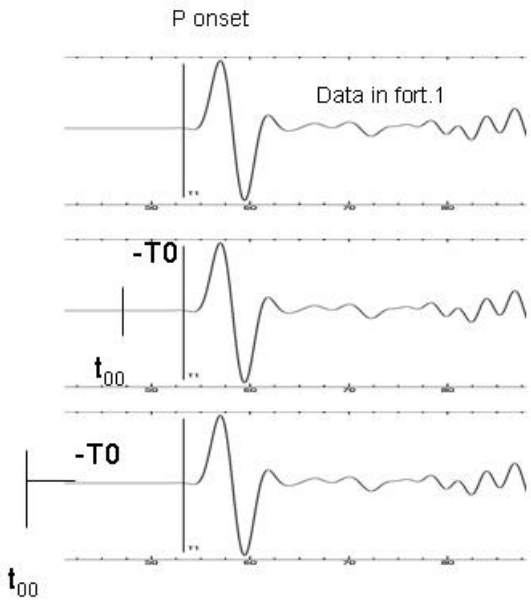


Explanation of T0 and Tst.

T0

The seismograms stored in fort.1 have a short lead before the onset of P wave. T0 determines the beginning of the data to be used for inversion. The starting time of the data to be used is chosen at -T0 sec before the onset of P, as shown in the following figure. For example, if T0=-10 sec, 10 sec before the P time is set to be the beginning of the records to be used for inversion. Let the beginning of the record be t_{00}. Then Tst is defined as follows.

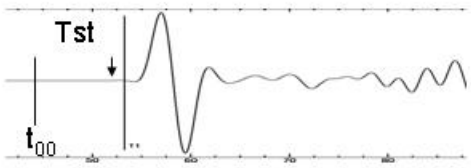
```
Note for a deep-focus earthquake

      Since the source response function is computed using a frequency domain
method, NT*DT should be sufficiently large to include pP and sP even if only the
P part of the Green's functions is used for inversion.  If NT*DT is too small.
the P part of the Green's function may contain spurious signals due to wrap-
around of the Fourier transform.   To check whether the Green's functions are
correctly computed or not use green.f to compute the Green's functions and plot
them with gregra.f.
```



Tst

The time when rupture begins at the chosen rupture starting point (x_0, y_0), measured from the beginning of the record set above, i.e., t_{00}. In other words, the waveform starting at t_{00}+Tst is used to determine the slip at (x_0, y_0),

So, if Tst is too large (like, Tst=30 sec), the initial part of the waveform is ignored. In general, as Tst increases, the effective rupture length is shortened. In contrast, if Tst is too small, no slip can be placed at (x_0, y_0), and the overall rupture length is increased.



3.6 gr3.f

```
    This program displays contour map of the slip distribution on (strike,
dip)-plane, source-time function and mechanism diagram.

Input files:
  fort.1:
  fort.24:
  stdin:   i_gr3

Output file:
   plot3: Postscript file
```

i_gr3

Example:

```
30.0 3 1.0 0 5 6.0
1 1 1
1 1.5 0.1
```

Explanation:

```
spcm, norm, height, line, Nc, Amu
npl, nps, fill
narw,amp0,tip
```

Line 1: spcm ([sec/cm]), norm (if= 0 scale is common for obs. & syn., if=1, each waveform is normalized, if=2, scale is common for each body wave), height, line (if=1, plot base-line), Nc(# of contour lines),Amu (rigidity [x10GPa], This is necessary to convert seismic moment to slip)
Line 2: npl (if=1, draw nodal-plane), nps (if=1, plot stations), fill (if=1, fill push-domain of mechanism diagram)
Line 3: narw (if=1, draw slip vector), amp0 (maximum length [cm] of the slip vector), tip (maximum length [cm] of the arrow)

4. Correlation Diagram (plotcm.f)

This program determines the mechanism of point sources placed at x-t grids, and computes correlations between the observed and synthetics. In a way this is a very useful program to look at the overall mechanism. This can be run in the beginning.

Input files:
- fort.1: event-station-parameter-data
- fort.3: Green's function file
- stdin: i_plotcm

Output file:
- plot.cm: Postscript file

i_plotcm

Example:

```
Hector Mine
80 80  3.5  10  1
1 2  4
24 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1
-20  1  1
15  1.0  1  0.45  0.
335 3  12  3
```

Explanation:

ID

```
TL,TE,V1,dur,kk
MS,T1,T2
Ns,(FC(J),J=1,Ns)
IDLY,ISR,IFP
NC,dx,dy,r,thre
Str,Dl,Nl,L0
```

Line 1: ID (Event ID)
Line 2: TL, TE, V1 are the same as elsewhere. Dur (the time interval used in the correlation diagram), kk (index # along-dip position of the source. Refer to h0, dk and k0 of the Green's function. The source depth used is then h=h0+dk*(kk-k0)*sin(dip))
Line 3: MS, T1, T2 (same as in i_inversion)
Line 4: NJS, FC (same as in i_inversion)
Line 5: IDLY, ISR, IFP (same as in i_inversion)
Line 6: NC (Number of contour lines), (dx,dy) (unit cell for the mechanism diagram: beach ball), r (radius in [cm] of the beach ball), thre (a threshold to draw the beach ball)
Line 7: Str, Dl, Nl, L0 (same as in i_inversion)

5. Fine tuning of the source time function

At all stages, if some shifts of the data (in time) are desired (mainly because of the travel time anomalies due to Earth's heterogeneity), it can be accomplished by changing IX0 in fort.1 for each station. This can be done only manually. IX0 is the number of data points. Increasing IX0 by ΔIX0 shifts the record backward by ΔIX0*Δt seconds, where Δt is the sampling interval of the data.

Once an overall rupture pattern has been determined, it can be refined with some judicious judgments using inversion2.f. inversion2.f requires a somewhat complicated input file i_inversion2 that defines the structure of the rupture model desired. This can be written by use of fort.16 used by inversion.f

5-1 inversion2.f

This program makes re-adjustment of time function for each subevent obtained by inversion.f.

Input files:
- fort.1: observed data
- fort.2: same as in green.f
- stdin: i_inversion2

Output files:
- fort.24: to be used by graphics2.f
- fort.26: source parameters etc.

i_inversion2

Example:

```
Denali, Alaska 02/11/03
   240 0.500   15.0   90.0    15    1  -30 256 1.
   1  13.0    0.0  120.0    0.0   75.209    0.000  226.8   39.7  101.0  3 4 8
   1  17.0    0.0  120.0    0.0   75.209    0.000  226.8   39.7  101.0  3 4 8
   1  21.0    0.0  120.0    0.0   75.209    0.000  226.8   39.7  101.0  3 4 8
   2  42.5   60.0  120.0    0.0  366.035    0.000  117.1   89.9  177.6  3 4 8
   2  46.5   60.0  120.0    0.0  366.035    0.000  117.1   89.9  177.6  3 4 8
   2  50.5   60.0  120.0    0.0  366.035    0.000  117.1   89.9  177.6  3 4 8
   3  54.5   90.0  120.0    0.0  366.035    0.000  117.1   89.9  177.6  3 4 8
   3  58.5   90.0  120.0    0.0  366.035    0.000  117.1   89.9  177.6  3 4 8
   3  62.5   90.0  120.0    0.0  366.035    0.000  117.1   89.9  177.6  3 4 8
   4  66.5  120.0  120.0    0.0  366.035    0.000  117.1   89.9  177.6  3 4 8
   4  70.5  120.0  120.0    0.0  366.035    0.000  117.1   89.9  177.6  3 4 8
   4  74.5  120.0  120.0    0.0  366.035    0.000  117.1   89.9  177.6  3 4 8
   5  78.5  150.0  120.0    0.0  366.035    0.000  117.1   89.9  177.6  3 4 8
   5  82.5  150.0  120.0    0.0  366.035    0.000  117.1   89.9  177.6  3 4 8
   5  86.5  150.0  120.0    0.0  366.035    0.000  117.1   89.9  177.6  3 4 8
24 .5 1 .0 1 1 1 1 1 1 0 1 .1 1
.1 .1 .1 .1 .1 .1 .1 .1 .1 .1 .1
Explanation:

ID
Nn, DT, H0, Dip, Ne, Idummy, Ixa, Nt
(gr, Tm, Fr, Phi, Yax, Dummy, Dummy, Str, Dip, Rake, Ms,T1, T2, i=1,Ne)
Ns,(Fc(j),j=1,Ns)

Line 1: ID
Line 2: NN (# of points (time window)), DT (dt), H0 (depth), DIP (dip), NE
     (Total # of events), Idummy (dummy), Ixa =(a shift parameter), Nt (# of
     points for Green function (power of 2))
Line 3: gr (group #), TM (timing),  fr (distance), Phi (azimuth), Yax (distance
     along dip), Dummy (dummy), Str (strike), Dip (dip), Rake (rake), Ms, T1, T2
     (source time function. See i_inversion)
Line 4: NS (# of phases), fc (weight)
```

6. Utility Programs

6-1. plotw.f

This program plots the data in fort.1.

```
Input files:
   fort.1:
   stdin:  i_plotw

Output file:
   plot.w: Postscript file
```

i_plotw

```
-------------------------------------------
Example

Fort.1
20 0.5 0 120. 1 1.0 2.0 10 1
```

```
300.

Explanation

Inf
spcm amp ishift tlen line penw dy ny norm
Camp    (if norm=1)

Line 1: Inf (file name of observed records)
Line 2: spcm (tip-mark [sec/cm]), amp (amplitude in [cm]), ishift (if 0,
     specified time-correction is neglected), tlen (time interval), line (if=1,
     draw base-line), penw (pen-width), dy (spacing in [cm] between traces), ny
     (# of traces per page), norm (if 0, each trace is normalized, if 1, scale
     is common)
Line 3: Camp (common scale in [micro-meter/cm])
```

6-2. gregra.f

```
    This program plots the Green's functions.

Input files:
   fort.3: Green function (binary)
   stdin:  i_gregra

Output file:
   plot.g:  Postscript file
```

i_gregra

```
-------------------------------------
Example

120. 1 2 2 20.0 1.0 1 1

Explanation

Tl, MS, T1, T2, spcm, height, line, Amag

Tl (duration), MS, T1, T2 (specify source time function. See i_inversion), spcm
   (sec/cm), height (cm), line (if 1, draw base-line), Amag (moment scaling
   factor)
```

6.3 lists.f

```
    This program presents a list of stations in fort.1.

Input file:
   fort.1
```

6.4 listg.f

```
    This program presents a list of Green's functions in fort.3.

Input file:
```

fort.3

6.5 rotSH.f

This program rotates displacement-vector to produce SH-component from NS & EW components.

Input-file
fort.22
Output-file
fort.11

7. List of files

Main/

conv.sac.farm.f	green.f	inversion3.f	plotw.f
gr3.f	green3.f	listg.f	rotSH.f
graphics.f	gregra.f	lists.f	
graphics2.f	inversion.f	mom3.f	
graphics3.f	inversion2.f	plotcm.f	

Subroutine/

readme	sub.corr.f	sub.init.f	sub.nnls.f	sub.radpmt.f
sub.axs.f	sub.cubicd.f	sub.instg.f	sub.nplane.f	sub.refl.f
sub.bodyfc.f	sub.dbas.f	sub.instw.f	sub.order.f	sub.reflsh.f
sub.bodyw.f	sub.dcp.f	sub.laspac.f	sub.plotw.f	sub.source.f
sub.cfft.f	sub.dctomt.f	sub.lvns.f	sub.plotw0.f	sub.stnplo.f
sub.clear.f	sub.det0.f	sub.lvns1.f	sub.pole.f	sub.stnplx.f
sub.cline.f	sub.det00.f	sub.mcplo.f	sub.prod.f	sub.zmom.f
sub.cnvr.f	sub.det1.f	sub.mcplot.f	sub.qfm.f	
sub.cnvrsh.f	sub.eig1.f	sub.mcplot0.f	sub.quadrd.f	
sub.contmap.f	sub.equarea.f	sub.mcplot1.f	sub.radp.f	
sub.conv.f	sub.filter.f	sub.mtrx.f	sub.radpfc.ff	

Input/

fort.1 fort.2 stdin-files jb.table jb.ptime jb.stime

Output/

fort.16, plot8, fort.26, plot28, fort.36, fort.37, plot38, plot.cm, plot3

8. Procedure for computation environment

1. Make library files:
   Subroutine/mylib.a
   Recipes/recipes.a
   recipes.a must contain following subroutine objects:
   eigsrt.f jacobi.f lubksb.f ludcmp.f quad3d.f
   These files may be installed from:
   “NUMERICAL RECIPES The art of scientific computing”
   eds. William H. Press, Brian P. Flannery, Saul A. Teukolsky,
   William T. Vetterling, 963pp., Cambridge University Press
   At parameter statement of jacobi.f and ludcmp.f, nmax must be set at a larger value than 1000.
2. Compile and make executable programs
   use Makefile