Program For Radial Trace Studies

Jon F. Claerbout

Three programs included in this package are a radial trace program, an inverse radial trace program, and a program to select optimal parameters for the transformations. The radial trace program is everywhere stretching, in order to enable an exact inverse. Constant velocity moveout correction is done during the transformation. (It may be avoided by specifying an infinite moveout velocity.) The inverse program is exact in the interior region (to the precision of the arithmetic). Data may be recorded either off-end or split-spread. The program that determines the range of radial traces attempts to keep as much of the data as possible in the interior of the region. All interpolation is "nearest neighbor". This is crude by ordinary standards, but it seems to be a price we pay for exact invertibility.

Ordinary moveout correction does nothing to the zero-offset trace, and it stretches everything else. Radial trace moveout correction is somewhat the opposite. All traces are compressed except for the zero offset. To make the transformation invertible, it is necessary to impose an overall stretch to counteract the intrinsic compression. I designed the overall stretch so that the net overall stretch is zero for the largest radial angle. Thus all of the remaining radial traces will be stretched. We must also ensure that the transformation stretches space. My first attempt to write this program assumed that the interval between radial traces should be chosen so that the radial trace separation matched the field trace separation at the latest time. Later this criterion was altered slightly to account for moveout correction. Now the radial traces match the field traces at the greatest depth, i.e. along the hyperbola that cuts the widest offset trace at the latest time. What this means is that the radial traces chosen will uniformly sample the tangent of the reflection angle.

In the program the radial parameter is not the obvious $r = x/t$. Instead it is the *tangent* of the radial angle, called g where $g = x/\sqrt{t^2 - x^2}$. (Note that constant x/t implies constant g.) In the program, the letter z does not denote depth. Instead it denotes moveout corrected travel time, which is like depth, but it has physical dimensions of time. So

the radial trace program maps field data from (t,x)-space to the space of (z,g). The method is as follows:

Radial Trace Transformation $(z,g) \leftarrow (t,x)$

```
loop over (z,g)-space
    output(z,g) = 0
loop over (z,g)-space
    compute x(z,g) and t(z,g)
    if (t,x) is on the data
        output(z,g) = input(t,x)
```

Inverse Radial Trace Transformation $(t,x) \leftarrow (z,g)$

```
loop over (t,x)-space
    output(t,x) = 0
    count(t,x) = 0
loop over (z,g)-space
    compute x(z,g) and t(z,g)
    if (t,x) is on the data
        output(t,x) = output(t,x) + input(z,g)
        count(t,x) = count(t,x) + 1
loop over (t,x)-space
    output(t,x) = output(t,x) / count(t,x)
```

The following is a listing of the program as it was last used. The bulk of the code is in a language called *Ratfor* [1] which stands for *Rat*ional *for*tran. It is really a preprocesser to Fortran which I find pleasing to use. It should be readily comprehensible to Fortran programmers.

[1] Kernighan and Plauger, *Software Tools*, Addison-Wesley Publishing Company

```
# Moveout corrected radial traces.
subroutine rad(nt,nx,xx,ng,gg)
real xx(nt,nx), gg(nt,ng),
     t0, x0,  dt, dx,  ix0, it0,  vrmo, vhalf,
     z0, g0,     dz, dg,
     t, x, z, g,
     cosi
integer nt, nx, ng, pad,
     it, ix, ig, iz, nz

x0=0.;  call fetch("x0","f",x0);    call putch("x0","f",x0)
t0=0.;  call fetch("t0","f",t0);    call putch("t0","f",t0)

ix0=0;  call fetch("ix0","f",ix0);    call putch("ix0","f",ix0)
it0=0;  call fetch("it0","f",it0);    call putch("it0","f",it0)

dx=1.;  call fetch("dx","f",dx);
dt=1.;  call fetch("dt","f",dt);
vrmo = 4*dx/dt
call fetch("vrmo","f",vrmo);    call putch("vrmo","f",vrmo)
vhalf = vrmo / 2.
pad = 1;  call fetch("pad","i",pad);    call putch("pad","i",pad)
call grdset(pad,nt,nx,t0,x0,dt,dx,ix0,it0,vhalf,nz,ng,z0,g0,dz,dg)
call putch("tg0","f",g0)
call putch("dtg","f",dg)

do iz=1,nz
     do ig=1,ng
          gg(iz,ig) = 0.
do ig=1,ng {
     g = g0 + ig*dg
     cosi =  sqrt( 1. + g*g )
     do iz=1,nz {
          z = z0 + iz*dz
          x = g * z * vhalf
          ix = (x - x0) / dx + .5
          t = z * cosi
          it = (t - t0) / dt + .5
          if( 0 < ix  &  ix <= nx )
               if( 0 < it  &  it <= nt )
                    gg(iz,ig) = xx(it,ix)
          }
     }
return
end
```

```
# Inverse Moveout corrected radial traces.
subroutine radi(nt,ng,gg,nx,xx,count)
real xx(nt,nx), gg(nt,ng), count(nt,nx),
     t0, x0,  dt, dx,  ix0, it0,  vrmo, vhalf,
     z0, g0,     dz, dg,
     t, x, z, g,
     cosi
integer nt, nx, ng, pad,
     it, ix, ig, iz, nz

x0=0.;   call fetch("x0","f",x0);
t0=0.;   call fetch("t0","f",t0);

ix0=0;   call fetch("ix0","f",ix0);
it0=0;   call fetch("it0","f",it0);

dx=1.;   call fetch("dx","f",dx);
dt=1.;   call fetch("dt","f",dt);
vrmo = 4*dx/dt
call fetch("vrmo","f",vrmo);
vhalf = vrmo / 2.
pad = 1;   call fetch("pad","i",pad);
call grdset(pad,nt,nx,t0,x0,dt,dx,ix0,it0,vhalf,nz,ng,z0,g0,dz,dg)

do it=1,nt
     do ix=1,nx {
          xx   (it,ix) = 0.
          count(it,ix) = 0.
          }
do ig=1,ng {
     g = g0 + ig*dg
     cosi =  sqrt( 1. + g*g )
     do iz=1,nz {
          z = z0 + iz*dz
          x = g * z * vhalf
          ix = (x - x0) / dx + .5
          t = z * cosi
          it = (t - t0) / dt + .5
          if( 0 < ix  &  ix <= nx )
               if( 0 < it  &  it <= nt ) {
                    xx(it,ix) = xx(it,ix) + gg(iz,ig)
                    count(it,ix) = count(it,ix) + 1.
                    }
          }
     }
do ix=1,nx
     do it=1,nt
          if ( count(it,ix) > 0.0 )
               xx(it,ix) = xx(it,ix) / count (it,ix)
return
end
```

```
# Determine the grid for radial traces
subroutine grdset(pad,nt,nx,t0,x0,dt,dx,ix0,it0,vhalf,nz,ng,z0,g0,dz,dg)
real    t0, x0,  dt, dx,  ix0, it0,  vhalf,
      z0, g0,     dz, dg,
      tmax, gritee, glefte, gmax, zmax,
      xl, xr, cl, cr
integer  nt, nx, nz, ng, pad, iter

# The real world shifts with    x = x0 + ( ix0 + (ix-1)/pad ) ) * dx
x0 = x0 + (ix0-1./pad)*dx
dx = dx/pad
# Now we can shift with        x = x0 + ix*dx
t0 = t0 + (it0-.99)*dt
# Likewise                    t = t0 + 1*dt   > 0.

tmax = t0 + nt*dt
glefte = (x0+ 1*dx) / sqrt( (vhalf*tmax)**2 + (x0+ 1*dx)**2 )
gritee = (x0+nx*dx) / sqrt( (vhalf*tmax)**2 + (x0+nx*dx)**2 )
gmax = amax1( abs(glefte), abs(gritee) )
zmax = tmax / sqrt( (1. + gmax*gmax ) )
# Notice that z has physical dimensions of time

nz = nt
z0 = (nz*(t0+dt) - zmax) / (nz-1)
dz = ( - (t0+dt) + zmax) / (nz-1)

dg = dx / (zmax*vhalf)

# match time cutoffs on both sides = xleft/sinleft = xrite/sinrite
cr = 1.          # cosine on the right side
cl = 1.          # cosine on the left side
do iter=1,5 {
      xl = x0 + 1*dx
      xr = x0 + nx*dx
      g0 = (xl*cr*ng*dg - xr*cl*dg)/(xr*cl-xl*cr)
      cr = 1. / sqrt( 1. + (g0+ng*dg)**2 )
      cl = 1. / sqrt( 1. + (g0+ 1*dg)**2 )
      }
return
end
```

What is the only game in which the defensive team always has possession of the ball, the offensive player can score without ever having touched the ball, and games itself is timeless?

##########

How many F's are there in the following passage?

"Finished files are the result of years of scientific study combined with the experience of many years."

##########

What do the following mean in plain English:

a) Conducting to the watering place a quadruped of equine race is simple, but he may not care to practice imbibition there.

b) When, nimbus-free, Sol marches across the circumambient sky, to graminiferous meads repair - your instant task awaits you there!

c) That unit of the avian tribe whose movements one can circumscribe "In manu," as a pair will rate subarboreally situate.

d) Of little value his compunctions, who arrogates clavigerous functions, when once from circumambient pen, is snatched its equine denizen.

e) Faced with material esculent as source of liquid nourishment, avoid excess -

##########

What set of well-known three digit numbers does this rhyming riddle describe?

The first is never zero or one,
The second always is,
And the third - not zero -
but maybe one,
Can you pick up on this?

##########