```
program ConicalRWE3D
  !! !!!!!!!!!!!!!!!!!!!
  !!
  !!            Written by Jeff Shragge
  !!                           Feb 18/2008
  !!                           contact: jshragge@gmail.com
  !!
  !! !!!!!!!!!!!!!!!!!!!!!!!!!

  !use sep                     !! Basic SEPlib I/O
  use util_                   !! handles axis I/O and aux file I/O
  use ConicalRWE3D_wemig !! Migration driver module
  use ConicalC2R3D            !! Ray-to-Cartesian interpolation
  use sep_timers_mod
  implicit none
  type(myaxis) :: aE1,aE2,aE3,aW,aS1,aS2,aX1,aX2,aX3,aR1,aR2,aT

  integer :: iX1,iX2,iX3,ii,iw,iis,iE1,iE2,iE3,iS1,iS2,ishot,stat,method,ntilt
  integer :: nsx,nsy,nsz,norm,itest,nbytes,i1,i2,i3,ncx,ncy,ncz,nxint,nyint,overlap
  integer :: elint1,elint2,iT,npw,pwmin,pwmax,n1beg,n1end,n2beg,n2end
  real    :: r0,eps,rextra,minang,maxang,rscale,atloc,otilt,dtilt
  real    :: ddx1,ddx2,ddx3,dd1,dd2,dd3,rro,rrd,amax,b,c,tilt_angle
  character (len=128) :: name
  integer, allocatable :: iix1(:),iix2(:),icx(:),icy(:)
  real,    allocatable :: ox1(:),ox2(:),vox1(:),vox2(:)
  real,    allocatable,dimension(:,:)    :: Image2D,fields,gnorm
  real,    allocatable,dimension(:,:,:)  :: RVel,CVel,CImg,RImg
  complex, allocatable,dimension(:,:,:)  :: rwf,swf
  logical :: verbose,adj,newimage

  !! Timers
  logical          :: logic !! For timers
  integer          :: t1,t2,t3,t4,t5,t6,t7,t8,t9,t10


  logic=init_sep_timers()

  !! !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
  logic=setup_next_timer("Headers",t1)
  logic=setup_next_timer("Geometry",t2)
  logic=setup_next_timer("Sail_lines",t4)
  logic=setup_next_timer("Sail_line_Setup",t5)
  logic=setup_next_timer("Plane_waves",t6)
  logic=setup_next_timer("Propagation",t7)
  logic=setup_next_timer("Interpolation",t8)
  logic=setup_next_timer("Write_out",t9)
  logic=setup_next_timer("TOTAL",t10)

  call start_timer_num(t10)
  call start_timer_num(t1)
  call sep_init()
  call getch_add_string("head=/dev/null")
  call getch_add_string("noheader=y")
  call set_no_putch()
  call auxinout("image")
  call sep_begin_prog()
  call from_param("newimage",newimage,.true.)
  call from_param("ntilt",ntilt,1)
  call from_param("otilt",otilt,0.)
  call from_param("dtilt",dtilt,15.)
  call from_param("n1beg",n1beg,0)
  call from_param("n1end",n1end,0)
  call from_param("n2beg",n2beg,0)
  call from_param("n2end",n2end,0)
  call from_param("overlap",overlap,5)

  !! . . Define Tilting parameters
  aT%n = ntilt
!  aT%o = otilt
!  aT%d = dtilt

  allocate( iix1(2),iix2(2),ox1(2),ox2(2),vox1(2),vox2(2),icx(2),icy(2) )

  !!    Read in input dimensions from the "rwf" tag
  !!    Determines the radial steps grid parameters
  call RWF_Info(aE1,aE2,aW,aS1,aS2,aR1,aR2)

  !! . . Padding the axes instead of reading in zero values
  aE1%n = aE1%n + n1beg+n1end
  aE2%n = aE2%n + n2beg+n2end
  aR1%n = aR1%n + n1beg+n1end
  aR2%n = aR2%n + n2beg+n2end
  aE1%o = aE1%o - n1beg*aE1%d
  aE2%o = aE2%o - n2beg*aE2%d
  aR1%o = aR1%o - n1beg*aR1%d
  aR2%o = aR2%o - n2beg*aR2%d

  !! . . Tilt number can't be greater than # of shots
  aT%n = min(aT%n,aS1%n)
  aT%o = aS1%o
  aT%d = aS1%d

  !! . . Read in input dimensions of the "vel" tag
```

```
  !! . . Determines the global image size
  call VEL_Info(aX1,aX2,aX3)

  !! . . See if Image file exists. If not, then write one out..
  name = "image"
  if ( newimage ) then
     call IMAGE_Out_Info(aX1,aX2,aX3)

     allocate ( Image2D( aX3%n,aX1%n) )
     call sseek('image', 0 , 0 )
     do iE2 = 1,aX2%n
        call srite("image", Image2D, 4*aX1%n*aX3%n)
     end do
     deallocate( Image2D )
  end if

  !! Velocity Indicies
  vox1(1)=aX1%o
  vox1(2)=aX1%o+(aX1%n-1.)*aX1%d
  vox2(1)=aX2%o
  vox2(2)=aX2%o+(aX2%n-1.)*aX2%d
  call IMAGE_Info(aX1,aX2,aX3)
  call sseek('image', 0 , 0 ) !! Rewind image

  !! . . Read in the parameters
  call Read_In_Parameters(aE3,verbose,norm,nsx,nsy,nsz,eps,&
  minang,maxang,method,newimage,nxint,nyint)

  call RIMAGE_Out_Info(aE1,aE2,aE3)
  call RVEL_Out_Info(aE1,aE2,aE3)
  !! . . Initialize propagation module
  call wemig_axes_init(aE1,aE2,aE3,aW,aX1,aX2,aX3,aS1,aS2,aR1,aR2,aT)

  !! . . Set scaling of the coordinate system.
  if ( method .eq. 1 ) then
     r0=1.
     write(0,*) 'before wemig_init'
     call wemig_param_init(verbose,r0,minang,maxang,eps,method)
     write(0,*) 'gogo'
  else
     if (aR2%n .eq. 1) then
        r0 = dble(aE3%n-1.)*aR1%d/2.
     else
        r0 = abs(aR2%o) !dble(aE3%n-1.)*aR2%d/2.
        !        r0 = dble(aE3%n-1.)*aR2%d/2.
     end if
     call wemig_param_init(verbose,r0,minang,maxang,eps,method)
  end if
  write(0,*) 'to here0'
  call stop_timer_num(t1)
  !! . . END OF HEADERS IN/OUT
  !!
  !! ------------------------------------------------------------

  write(0,*)  'end headers'
  !! ------------------------------------------------------------
  !!
  !! . . GEOMETRY SECTION
  !! . . Generate Coordinate System
  call start_timer_num(t2)
  allocate ( gnorm(aE2%n,aE3%n),fields(aE2%n,aE3%n))

  if (method .eq. 1) then
     call  CART_Setup(fields,gnorm)
     aE1%o=aR1%o; aE1%d=aR1%d
     aE2%o=aR2%o; aE2%d=aR2%d
  else
     !! Set tilt_angle = 0. for initial setup
     call CYNEL_SETUP(fields,gnorm)
     aE1%o=aR1%o;
     aE1%d=aR1%d
     aE2%o=acos(-1.)/180.*minang
     aE2%d=(maxang-minang)/180.*acos(-1.)/(aE2%n-1.)
     !     aE2%d=(maxang-minang)/180.*acos(-1.)/(aE3%n-1.)
  end if

  !! . . Reinitialize the axes because E1 or E3 could have changed
  call wemig_axes_init(aE1,aE2,aE3,aW,aX1,aX2,aX3,aS1,aS2,aR1,aR2,aT)
  call Geometry_QC(aE1,aE2,aE3,aS1,aS2,aW,aR1,aR2,aT,aT%n/2,1)
  call stop_timer_num(t2)

  !! . . END OF GEOMETRY SECTION
  !! ---------------------------------------------------------------

  !!--------------------------------------------------------------
  !! . . START MIGRATION SECTION
  !! . . Allocate space for source/receiver wavefields and local velocities

  allocate ( swf(aE1%n,aE2%n, aW%n),  rwf(aE1%n,aE2%n, aW%n) )
  allocate (RVel(aE1%n,aE2%n,aE3%n), RImg(aE1%n,aE2%n,aE3%n) )

  !! Initialize the axes
  call C2R3D_axes(aE1,aE2,aE3,aX1,aX2,aX3,aS1,aS2,aR1,aR2,aT)
```

```
!! . . Loop over sail lines
SAIL_LINES: do iS2 = 1,aS2%n
   call start_timer_num(t4)
   call start_timer_num(t5)

   !! . . Loop over number of coordinate systems
   TILT_ANGLES: do iT = 1,aT%n

      !! Define coordinate system tilt and update rays
      tilt_angle = (iT-1.)*aT%d+aT%o

      call wemig_axes_init(aE1,aE2,aE3,aW,aX1,aX2,aX3,aS1,aS2,aR1,aR2,aT)
      call Geometry_QC(aE1,aE2,aE3,aS1,aS2,aW,aR1,aR2,aT,iT,iS2)

      RVel=0.;
      RImg=0.
      call Get_Vel_Indicies(iix1,iix2,ox1,ox2,it,is2)
      write(0,*) 'VEL INDICIES',iix1(1),iix1(2),iix2(1),iix2(2)
      ncx = iix1(2)-iix1(1)+1;
      ncy = iix2(2)-iix2(1)+1;
      ncz = aX3%n
      icx=iix1
      icy=iix2
      ncy = max(ncy,1); ncx=max(ncx,1)

      !! . . Allocate necessary arrays
      allocate ( CImg(ncz,ncx,ncy) , CVel(ncz,ncx,ncy) )

      !! Read in minimal velocity subset
      name="vel"
      call sseek("vel",0,0)
      do iE2 = 1,ncy
         nbytes = 4*( ncz*aX1%n*(iix2(1)-1+(iE2-1)) + ncz*(iix1(1)-1) )
         call sseek("vel",nbytes,0)
         call sreed( name, CVel(:,:,iE2) , ncz*ncx*4)
      end do

      !! . . Interpolate Vel from Cart to Riemannian
      call C2R3D_init(0,0,0,norm,aX1%d,aX2%d,aX3%d,&
      ncx,ncy,ncz,aE1%n,aE2%n,aE3%n,&
      aX1%o+(iix1(1)-1.)*aX1%d,&
      aX2%o+(iix2(1)-1.)*aX2%d+(is2-1.)*aS2%d,&
      0.,method,minang,maxang)

      call C2R3D_CART (RVel,CVel,gnorm,.false.,1,aE1%n,tilt_angle,iS2,iT)

      !! . . Check to see if velocities are good
      call Velocity_Check(CVel,RVel)
      call Get_Vel_Indicies(iix1,iix2,ox1,ox2,it,is2)
      call stop_timer_num(t5)

      !! . . Find the start and end of the plane waves to do
      call PLANEWAVES_TODO(iT,pwmin,pwmax)
      write(0,'(a,i4,i4,2f11.2)') '     * USING PWMIN/PWMAX at angle: ',pwmin,pwmax,tilt_angle
      !! . . Image each plane wave for a given tilted coordinate systems
      PLANEWAVES: do iS1 = pwmin,pwmax
         call start_timer_num(t6)
         !! . . Read in wavefields and position wavefield on surface
         name="swf";
         call READ_IN_WAVEFIELDS(name,swf,aE1%n,aE2%n,aW%n,n1beg,n1end,n2beg,n2end)
         name="rwf";
         call READ_IN_WAVEFIELDS(name,rwf,aE1%n,aE2%n,aW%n,n1beg,n1end,n2beg,n2end)

         !! . . Call propagation and imaging module
         call start_timer_num(t7)
         ishot = iS1+aS1%n*(iS2-1)

         !! . . Main propagation routine
         call wemig( rwf,swf,RImg,ishot,iT,is2,RVel,fields)
         call stop_timer_num(t7)
         call stop_timer_num(t6)

      end do PLANEWAVES

      !! . . Image Interpolation to Cartesian
      do i1=1,nxint
         e1int1 =   floor( aE1%n *dble(i1-1.)/dble(nxint) ) + 1
         e1int2 = ceiling( aE1%n *dble(i1   )/dble(nxint) )

         call start_timer_num(t8)

         !! Cart Cut indicies
         call Get_Img_Indicies(overlap,iix1,iix2,ox1,ox2,vox1,vox2,iT,is2,e1int1,e1int2)
         !! Don't interpolate if all off grid
         if (iix1(1) .eq. iix1(2) .and. aE1%n .ne. 1 ) cycle
         if (iix2(1) .eq. iix2(2) .and. aE2%n .ne. 1 ) cycle
         write(0,*) 'IMG INDICIES 1',iix1(1),iix1(2),ox1,vox1,icx(1),icx(2)

         iix1(1) = max(icx(1),iix1(1)-nsx-overlap)-icx(1)+1
         iix1(2) = min(icx(2),iix1(2)+nsx+overlap)-icx(1)+1

         iix2(1) = max(icy(1),iix2(1)-nsy-overlap)-icy(1)+1
```

```
         iix2(2) = min(icy(2),iix2(2)+nsy+overlap)-icy(1)+1


         !! Initialize interp routine
         call C2R3D_init(nsx,nsy,nsz,norm,&
         aX1%d,aX2%d,aX3%d,&
         iix1(2)-iix1(1)+1,iix2(2)-iix2(1)+1,ncz,&
         e1int2-e1int1+1,aE2%n,aE3%n,&
         ox1(1)-(overlap+nsx)*aX1%d,&
         ox2(1)+(is2-1.)*aS2%d,0.,&
         method,minang,maxang)
         !! Interp Image: RWE -> Cartesian

         write(0,*) 'IMG INDICIES 2',iix1(1),iix1(2),ox1,vox1,icx(1),icx(2)
         call C2R3D_CART (  &
         RImg  (e1int1:e1int2,:,:),&
         CImg  (:,iix1(1):iix1(2),iix2(1):iix2(2)),&
         gnorm,.true.,e1int1,e1int2,tilt_angle,iS2,iT)
         write(0,*) 'OUT OF C2R3D'
         call stop_timer_num(t8)
      end do

      !! . . Add single image to global image
      !! . . Read in image subset
      if (verbose) write(0,*) '     * Reading in Image volume'
      if (verbose) write(0,*) '     * RIMG/CIMG max: ',maxval(RImg),maxval(CImg)

      call start_timer_num(t9)
      !! Reset indicies
      call Get_Vel_Indicies(iix1,iix2,ox1,ox2,iT,is2)
      call sseek('image', 0 , 0 ) !! Rewind

      !! Do for each y section
      allocate( Image2D(ncz,ncx) )

      !! . . For each y-slice
      Sum_y_interpolate: do iE2 = 1,ncy
         nbytes = 4*( ncz*aX1%n*(iix2(1)-1+(iE2-1)) + ncz*(iix1(1)-1) )
!         nbytes = 4*( ncz*aX1%n*(iix2(1)-1+(iE2-1)) + ncz*(icx(1)-1) )
         call sseek("image",nbytes,0)
         call sreed('image', Image2D, 4*ncz*ncx)
         call sseek('image', 0 , 0 )

         !! Update image volume
         do i1=1,ncx
            do i3=1,ncz
               Image2D(i3,i1) = Image2D(i3,i1) + CImg(i3,i1,iE2)
            end do
         end do

         !! Write out image
!         nbytes = 4*( ncz*aX1%n*(iix2(1)-1+(iE2-1)) + ncz*(icx(1)-1) )
         nbytes = 4*( ncz*aX1%n*(iix2(1)-1+(iE2-1)) + ncz*(iix1(1)-1) )
         call sseek("image",nbytes,0)
         call srite("image", Image2D, 4*ncz*ncx)

      end do Sum_y_interpolate

      !! . . deallocate variables associated with temp Cart image
      deallocate ( CVel , CImg , Image2D  )
      call stop_timer_num(t9)

    end do TILT_ANGLES

    call stop_timer_num(t4)

 end do SAIL_LINES
 call stop_timer_num(t10)
 call print_timers() !! Output benchmarking statistics

 call sep_end_prog()
 call exit()

end program ConicalRWE3D
```