Appendix F
EIT Reconstruction Code `EITAXI.F`

This Fortran 77 program by J. R. Torczynski determines the radial conductivity distribution in a three-dimensional cylindrical domain from voltage measurements at electrodes on the domain boundary. `EITAXI` uses library files that map the electrode voltages as a function of the conductivity distribution within the domain. The libraries are generated by the commercial code FIDAP (Fluid Dynamics International, 1996), which solves the finite-element representation of the voltage equations (Eq. 3.1). Using cubic-spline interpolation and a Newton-Raphson algorithm, `EITAXI` determines the best-fit parameters for a user-selected conductivity distribution that most closely reproduce the measured voltages. This latest version of `EITAXI` includes the option to convert voltage data to quartic radial conductivity profiles, but permits quadratic and constant conductivity profiles as simpler cases.

`EITAXI` reads from and writes to the following files:

`eitaxi_inp.dat`	general input parameters (input)
`eitaxi_exp.dat`	file of experimental voltages (input)
`eitaxi_coe.dat`	file of coefficients fit to fundamental voltage solutions (input)
`eitaxi_log.dat`	convergence and conductivity parameters after each iteration (diagnostic output)
`eitaxi_out.dat`	general output parameters (output)
`eitaxi_sol.dat`	converged fundamental voltage solution (output)

The file `eitaxi_exp.dat` is identical in format to `femeit_exp.dat` in Appendix E. Examples of pertinent file formats unique to `EITAXI` follow.

`eitaxi_inp.dat:`

`1.0E-04`	Conversion from least significant bits (EIT output units) to volts; nominal value is 0.0001 V/LSB
`1.0`	Ratio of electrode height to domain radius (used only in two-dimensional reconstructions, where input data is based on linear current density instead of total current; set to 1.0 for correct normalization in three-dimensional reconstructions)
`0.09525`	Domain radius, R_{col} (m)
`125.0E-06`	Injection current (A)
`0.0600`	Baseline liquid conductivity, σ_L (S/m)
`20`	Maximum number of iterations
`0.5`	}
`0.`	} Iterative damping coefficients in Newton-Raphson algorithm (dimensionless)

`1.0`	}
`0.0000001`	Convergence criterion for conductivity parameters C_1 and C_2
`0.0000001`	Convergence criterion for conductivity parameter C_0
`1.0`	}C_0
`1.0`	}C_1 Initial values of conductivity parameters
`0.0`	}C_2
`2`	Number of fit parameters: 0 = constant profile, 1 = quadratic, 2 = quartic

`eitaxi_out.dat` echoes the contents of `eitaxi_inp.dat` but replaces the initial values of the conductivity parameters with the final, converged values.

`eitaxi_coe.dat:`

This library file contains the fitting coefficients, one per line, for all reciprocal nontrivial voltages of the fundamental voltage solution. The outermost loop is the electrode number, the middle loop is the power of C_2, and the innermost loop is the power of C_1.

```
c
c23456789012345678901234567890123456789012345678901234567890123456789012
c
      program eitaxi
c
c     Revision 19990419
c
c *** Finds best fit axisymmetric conductivity profile.
c     Uses 2 shape parameters.
c     Uses analytical representation of fundamental solution.
c
c     Currently eitaxi_coe.dat has fit coefficients for a quartic:
c     (1/c0)(1+c1*(2r^2-1)+c2*(1-6r^2+6r^4))
c     Electrodes are 3" x 0.25", 16 at 22.5 degrees, 7.5" ID.
c     Must recompute eitaxi_coe.dat for different electrode geometry.
c
      implicit double precision (a-h,o-z)
c
      parameter (nfun=8)
      parameter (ncoe=24)
      dimension vcoe0(0:ncoe,0:ncoe,0:nfun)
      dimension vcoe1(0:ncoe,0:ncoe,0:nfun)
      dimension vcoe2(0:ncoe,0:ncoe,0:nfun)
      dimension vfun0(0:nfun)
      dimension vfun1(0:nfun)
      dimension vfun2(0:nfun)
c
      parameter (nelc=2*nfun)
      dimension wt(nelc,nelc,nelc)
      dimension vk0(nelc)
      dimension vkm0(nelc,nelc)
      dimension vkmn0(nelc,nelc,nelc)
      dimension vk1(nelc)
      dimension vkm1(nelc,nelc)
      dimension vkmn1(nelc,nelc,nelc)
      dimension vk2(nelc)
      dimension vkm2(nelc,nelc)
      dimension vkmn2(nelc,nelc,nelc)
      dimension vnrm(nelc,nelc,nelc)
c
 1001 format (1x,d18.12)
 1002 format (1x,i4)
 1003 format (i4,3(1x,d11.5,1x,d8.2),2(1x,d8.2))
 1004 format (3(1x,d11.5))
 2000 format (1x,a)
 2001 format (1x,a12,d18.12)
 2002 format (1x,a12,i4)
c
c *** Initialize the weights.
c
      do 0020 i1 = 1, nelc, 1
      do 0020 i2 = 1, nelc, 1
      do 0020 i3 = 1, nelc, 1
         wt(i1,i2,i3) = 1.
         if ((i1.eq.i2).or.(i1.eq.i3).or.(i2.eq.i3)) wt(i1,i2,i3)=0.
 0020    continue
c
c *** Read in input parameters.
c
      write (6,2000) 'Reading input parameters from eitaxi_inp.dat'
      open (unit=23, status='old', file='eitaxi_inp.dat')
      read (23,*) convrt
      read (23,*) hoverr
      read (23,*) radius
      read (23,*) curr12
      read (23,*) sigma0
      read (23,*) niter
      read (23,*) damp0
      read (23,*) damp1
      read (23,*) damp2
      read (23,*) tolc
      read (23,*) tolr
      read (23,*) c0val
      read (23,*) c1val
      read (23,*) c2val
      read (23,*) nshape
      close (unit=23)
c
      vltref = curr12 / ( convrt * hoverr * sigma0 * radius )
      vltcon = 1. / vltref
      write (6,2001) '   convrt = ', convrt
      write (6,2001) '   hoverr = ', hoverr
      write (6,2001) '   radius = ', radius
      write (6,2001) '   curr12 = ', curr12
      write (6,2001) '   sigma0 = ', sigma0
      write (6,2002) '   niter  = ', niter
      write (6,2001) '   damp0  = ', damp0
      write (6,2001) '   damp1  = ', damp1
      write (6,2001) '   damp2  = ', damp2
      write (6,2001) '   tolc   = ', tolc
      write (6,2001) '   tolr   = ', tolr
      write (6,2001) '   c0val  = ', c0val
      write (6,2001) '   c1val  = ', c1val
      write (6,2001) '   c2val  = ', c2val
      write (6,2002) '   nshape = ', nshape
```

```
c
c *** Read in experimental voltages and normalize.
c
      write (6,2000) 'Reading experimental data from eitaxi_exp.dat'
      open (unit=24, status='old', file='eitaxi_exp.dat')
      svcex2 = 0.
      wnexp  = 0.
      do 0050 ip1 = 1, nelc-1, 1
      do 0050 ip2 = ip1+1, nelc, 1
         wtotal = 0.
         svcexa = 0.
         svcexb = 0.
         do 0040 ip = 1, nelc, 1
            read (24,*) ipa, ipb, ipc, vmagn, vcarr, vquad
            vcex = vmagn * vltcon
            wtex = wt(ip,ip1,ip2)
            wtotal = wtotal + wtex
            svcexa = svcexa + wtex * vcex
            svcexb = svcexb + wtex * vcex * vcex
            vnrm(ip,ip1,ip2) = vcex
 0040       continue
         svcexa = svcexa / wtotal
         svcexb = svcexb / wtotal
         svcex2 = svcex2 + svcexb - svcexa ** 2
         wnexp  = wnexp + 1.
 0050    continue
      svcex1 = sqrt(svcex2 / wnexp)
      close (unit=24)
c
c *** Initialize the fundamental solution and derivative parameters.
c     The coefficients vcoe0(*,*,0) must all be zero.
c
      write (6,2000) 'Reading coefficients from eitaxi_coe.dat'
      open (unit=20, status='old', file='eitaxi_coe.dat')
c
      do 0110 ifun = 0, nfun, 1
      do 0105 ico2 = 0, ncoe, 1
      do 0100 ico1 = 0, ncoe, 1
         read (20,*) vcoe0(ico1,ico2,ifun)
 0100    continue
 0105    continue
 0110    continue
c
      close (unit=20)
c
      do 0115 ico2 = 0, ncoe, 1
      do 0115 ico1 = 0, ncoe, 1
         if (vcoe0(ico1,ico2,0).ne.0.) go to 0998
 0115    continue
c
      do 0130 ifun = 0, nfun, 1
      do 0120 ico2 = 0, ncoe, 1
      do 0120 ico1 = 0, ncoe-1, 1
         vcoe1(ico1,ico2,ifun) = dfloat(ico1+1)*vcoe0(ico1+1,ico2,ifun)
 0120    continue
         vcoe1(ncoe,ico2,ifun) = 0.
 0130    continue
c
      do 0150 ifun = 0, nfun, 1
      do 0140 ico1 = 0, ncoe, 1
      do 0140 ico2 = 0, ncoe-1, 1
         vcoe2(ico1,ico2,ifun) = dfloat(ico2+1)*vcoe0(ico1,ico2+1,ifun)
 0140    continue
```

```
         vcoe2(icol,ncoe,ifun) = 0.
 0150    continue
c
c *** Begin the iterative least-squares fit.
c
      write (6,2000) 'Writing iterations to eitaxi_log.dat'
      open (unit=25, status='unknown', file='eitaxi_log.dat')
      write (6,2000) ' '
      write (6,2000) ' it   c0val       c0rel'//
     1               '    c1val       c1inc'//
     2               '    c2val       c2inc'//
     3               '    corr     rmsnrm'
      write  (6,1003) 0, c0val, 0., c1val, 0., c2val, 0., 0., 0.
      write (25,1003) 0, c0val, 0., c1val, 0., c2val, 0., 0., 0.
c
      do 0500 it = 1, niter, 1
c
c *** Find the fundamental solution and derivative.
c
      do 0200 ifun = 0, nfun, 1
         vfun0(ifun) = 0.
         vfun1(ifun) = 0.
         vfun2(ifun) = 0.
 0200    continue
c
      c2arg = 1.
      do 0230 ico2 = 0, ncoe, 1
      c12arg = c2arg
      do 0220 icol = 0, ncoe, 1
      do 0210 ifun = 1, nfun, 1
         vfun0(ifun) = vfun0(ifun) + vcoe0(icol,ico2,ifun) * c12arg
         vfun1(ifun) = vfun1(ifun) + vcoe1(icol,ico2,ifun) * c12arg
         vfun2(ifun) = vfun2(ifun) + vcoe2(icol,ico2,ifun) * c12arg
 0210    continue
         c12arg = c12arg * c1val
 0220    continue
         c2arg = c2arg * c2val
 0230    continue
c
      do 0240 ifun = 1, nfun, 1
         vfun0(ifun) = 1. / max(vfun0(ifun),1.D-03)
         vfun1(ifun) = - vfun1(ifun) * ( vfun0(ifun) ** 2 )
         vfun2(ifun) = - vfun2(ifun) * ( vfun0(ifun) ** 2 )
 0240    continue
c
c *** Find voltages and derivatives for all pairwise combinations.
c
      do 0310 ip = 1, 1+nfun, 1
         ifun = 1 + nfun - ip
         vk0(ip) = vfun0(ifun)
         vk1(ip) = vfun1(ifun)
         vk2(ip) = vfun2(ifun)
 0310    continue
c
      do 0320 ip = 2+nfun, nelc, 1
         ifun = ip - ( 1 + nfun )
         vk0(ip) = vfun0(ifun)
         vk1(ip) = vfun1(ifun)
         vk2(ip) = vfun2(ifun)
 0320    continue
c

      do 0330 ipm = 1, nelc, 1
      do 0330 ipk = 1, nelc, 1
         ip = ipk + 1 - ipm
         if (ip.le.0) ip = ip + nelc
         vkm0(ipk,ipm) = vk0(ip)
         vkm1(ipk,ipm) = vk1(ip)
         vkm2(ipk,ipm) = vk2(ip)
 0330    continue
c
      do 0340 ipk = 1, nelc, 1
      do 0340 ipm = 1, nelc, 1
      do 0340 ipn = 1, nelc, 1
         vkmn0(ipk,ipm,ipn) = vkm0(ipk,ipm) - vkm0(ipk,ipn)
         vkmn1(ipk,ipm,ipn) = vkm1(ipk,ipm) - vkm1(ipk,ipn)
         vkmn2(ipk,ipm,ipn) = vkm2(ipk,ipm) - vkm2(ipk,ipn)
 0340    continue
c
c *** Find the matrix and vector for least-squares fit.
c
      e00 = 0.
      e10 = 0.
      e20 = 0.
      c00 = 0.
      c01 = 0.
      c02 = 0.
      c10 = 0.
      c11 = 0.
      c12 = 0.
      c20 = 0.
      c21 = 0.
      c22 = 0.
c
      svrms2 = 0.
      do 0400 ipm = 1, nelc-1, 1
      do 0400 ipn = ipm+1, nelc, 1
         wtotal = 0.
         svrmsa = 0.
         svrmsb = 0.
      do 0390 ipk = 1, nelc, 1
         verr = vnrm(ipk,ipm,ipn) - vkmn0(ipk,ipm,ipn) * c0val
         wtipk = wt(ipk,ipm,ipn)
         wtotal = wtotal + wtipk
         svrmsa = svrmsa + wtipk * verr
         svrmsb = svrmsb + wtipk * verr * verr
      do 0380 ipl = 1, nelc, 1
         weight = wtipk * wt(ipl,ipm,ipn)
         vdif0 = vkmn0(ipk,ipm,ipn) - vkmn0(ipl,ipm,ipn)
         vdif1 = vkmn1(ipk,ipm,ipn) - vkmn1(ipl,ipm,ipn)
         vdif2 = vkmn2(ipk,ipm,ipn) - vkmn2(ipl,ipm,ipn)
         edif0 = vnrm(ipk,ipm,ipn) - vnrm(ipl,ipm,ipn)
         c00 = c00 + weight * vdif0 * vdif0
         c01 = c01 + weight * vdif0 * vdif1
         c02 = c02 + weight * vdif0 * vdif2
c         c10 = c10 + weight * vdif1 * vdif0
         c11 = c11 + weight * vdif1 * vdif1
         c12 = c12 + weight * vdif1 * vdif2
c         c20 = c20 + weight * vdif2 * vdif0
c         c21 = c21 + weight * vdif2 * vdif1
         c22 = c22 + weight * vdif2 * vdif2
         e00 = e00 + weight * edif0 * vdif0
         e10 = e10 + weight * edif0 * vdif1
         e20 = e20 + weight * edif0 * vdif2
 0380    continue
 0390    continue
```

```
         svrmsa = svrmsa / wtotal
         svrmsb = svrmsb / wtotal
         svrms2 = svrms2 + svrmsb - svrmsa ** 2
 0400    continue
      svrms1 = sqrt(svrms2 / wnexp)
      rmsnrm = svrms1 / svcex1
c
      c10 = c01
      c20 = c02
      c21 = c12
      e01 = e00
      e02 = e00
      e11 = e10
      e12 = e10
      e21 = e20
      e22 = e20
c
c *** Solve for c0inc, c1inc, c2inc.
c
      dt012 = 1.
      s0inc = c0val
      s1inc = 0.
      s2inc = 0.
c
      if (nshape.eq.0) then
         dt012 = c00
         s0inc = e00
         s1inc = 0.
         s2inc = 0.
         end if
c
      if (nshape.eq.1) then
         dt012 = c00*c11 - c01*c10
         s0inc = e00*c11 - c01*e10
         s1inc = c00*e11 - e01*c10
         s2inc = 0.
         end if
c
      if (nshape.eq.2) then
         dt012 = c00*c11*c22 + c02*c10*c21 + c01*c12*c20
     1         - c00*c12*c21 - c02*c11*c20 - c01*c10*c22
         s0inc = e00*c11*c22 + c02*e10*c21 + c01*c12*e20
     1         - e00*c12*c21 - c02*c11*e20 - c01*e10*c22
         s1inc = c00*e11*c22 + c02*c10*e21 + e01*c12*c20
     1         - c00*c12*e21 - c02*e11*c20 - e01*c10*c22
         s2inc = c00*c11*e22 + e02*c10*c21 + c01*e12*c20
     1         - c00*e12*c21 - e02*c11*c20 - c01*c10*e22
         end if
c
      s0inc = s0inc / dt012
      s1inc = s1inc / dt012
      s2inc = s2inc / dt012
      c0tmp = s0inc
      c0inc = c0tmp - c0val
      c1inc = s1inc / c0tmp
      c2inc = s2inc / c0tmp
c
c *** Compute adaptive damping.
c
      damp = damp0
      damp = max(damp,abs(c0inc/c0val))
      corr = max(damp1,min(damp2,damp0/damp))
c
```

```
c *** Update c0val, c1val.
c
      c0val = c0val + c0inc * corr
      c1val = c1val + c1inc * corr
      c2val = c2val + c2inc * corr
c
c *** Write result to monitor and file.
c
      c0rel = c0inc / c0val
c
      write  (6,1003) it,c0val,c0rel,c1val,c1inc,c2val,c2inc,corr,rmsnrm
      write (25,1003) it,c0val,c0rel,c1val,c1inc,c2val,c2inc,corr,rmsnrm
c
c *** Branch out of loop if tolerances satisfied.
c
      if ((abs(c0rel).lt.tolr).and.(abs(c1inc).lt.tolc).and.
     1    (abs(c2inc).lt.tolc)) go to 0600
c
c *** Iteration finished.
c
 0500 continue
c
c *** Close file.
c
 0600 continue
      close (unit=25)
c
c *** Write out fundamental solution.
c
      write (6,2000) ' '
      write (6,2000) 'Writing fundamental solution to eitaxi_sol.dat'
      open (unit=27, status='unknown', file='eitaxi_sol.dat')
      do 0700 ifun = 0, nfun, 1
         write (27,1001) vfun0(ifun) * c0val
 0700    continue
      close (unit=27)
c
c *** Write out input parameters.
c
      write (6,2000) 'Writing output parameters to eitaxi_out.dat'
      open (unit=26, status='unknown', file='eitaxi_out.dat')
      write (26,1001) convrt
      write (26,1001) hoverr
      write (26,1001) radius
      write (26,1001) curr12
      write (26,1001) sigma0
      write (26,1002) niter
      write (26,1001) damp0
      write (26,1001) damp1
      write (26,1001) damp2
      write (26,1001) tolc
      write (26,1001) tolr
      write (26,1001) c0val
      write (26,1001) c1val
      write (26,1001) c2val
      write (26,1002) nshape
      close (unit=26)
c
      write (6,*) ' '
      write (6,1004) c0val, c1val, c2val
      write (6,*) ' '
c
c *** Completed, stop.
```

```
c
      go to 0999
 0998 write (6,2000) '*** ABNORMAL STOP ***'
 0999 continue
c
      stop 'eitaxi'
      end
c
c23456789012345678901234567890123456789012345678901234567890123456789012
c
```