Appendix G
Supplementary EIT Code `EITFUN.F`

The supplementary Fortran 77 code `EITFUN` by J. R. Torczynski computes fundamental voltage solutions from axisymmetric experimental datasets. These may be compared with computational fundamental voltages to evaluate the sensitivity of different electrode geometries. Both input and output are in "least significant bit" (LSB) units. For data taken with the original Data Translation® card, the data can be converted to units of volts by the equality 10^{-4} V = 1 LSB, determined during validation experiments.

The `EITFUN` algorithm begins by calculating weighted voltage "measurements" $\hat{V}_m^{(ij)}$ over all combinations of injection electrode i, withdrawal electrode j, and measurement electrode m. The weighted "measurements" are computed directly from the actual measured voltages $\tilde{V}_m^{(ij)}$ by the formula

$$\hat{V}_m^{(ij)} = \left(\frac{\sigma_L R_{col}}{I}\right)\left[\tilde{V}_m^{(ij)} - \frac{1}{N-2}\sum_{m=1}^{N} w_m^{(ij)}\tilde{V}_m^{(ij)}\right]. \tag{G.1}$$

The weights are equal to unity if all three electrode indices are different, and zero whenever at least two of the indices are equal.

$$w_m^{(ij)} = \begin{cases} 1 \text{ if } i \neq j, i \neq m, j \neq m \\ 0 \text{ otherwise} \end{cases}. \tag{G.2}$$

Next, weight functions F_n are computed as a summation of the weighted measurements over all possible combinations of i, j, and m:

$$F_n = \sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\sum_{m=1}^{N} w_m^{(ij)}\hat{V}_m^{(ij)}\left(\delta_{pn} - \delta_{qn}\right),\ n = 1, \ldots, (N/2)-1. \tag{G.3}$$

Here, δ_{pn} is the Kronecker delta function, and the indices p and q are defined as

$$p = \left|\frac{N}{2} - |i-m|\right|, \tag{G.4}$$

$$q = \left|\frac{N}{2} - |j-m|\right|. \tag{G.5}$$

Finally, the fundamental voltages are computed. At the non-current-bearing electrodes, the fundamental voltages V_n are determined by the formula

$$V_n = \frac{1}{2N(N-1)}\left[F_n + 2\left(\sum_{s=1}^{\frac{N}{2}-1} F_s\right)\right],\ n = 1,\ldots,(N/2)-1. \qquad \text{(G.6)}$$

By definition, the fundamental voltage V_0 at the reference electrode is zero, while the fundamental voltage at the injection electrode is computed using the fundamental voltages at the other electrodes:

$$V_{\frac{N}{2}} = \frac{\sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\frac{1}{2}\left(\hat{V}_i^{(ij)} - \hat{V}_j^{(ij)}\right) + \sum_{i=1}^{N}\sum_{j=i+1}^{N} V_{\left|\frac{N}{2}-|i-j|\right|}}{N(N-1)/2}. \qquad \text{(G.7)}$$

The input file `eitfun_inp.dat` is identical to the first five lines of `eitaxi_inp.dat` in Appendix F. The data file `eitfun_exp.dat` is produced by the data acquisition codes in Appendices B and C and is identical in format to `femeit_exp.dat` in Appendix E. The output file `eitfun_sol.dat` lists the computed fundamental voltages in the order V_0, V_1, ..., $V_{\frac{N}{2}}$ (injection electrode last).

```
c
c234567890123456789012345678901234567890123456789012345678901234567890l2
c
      program eitfun
c
c     Revision 19990419
c
c *** Computes fundamental voltage solution from experimental data.
c
      implicit double precision (a-h,o-z)
c
      parameter (nfun=8)
      dimension f(0:nfun)
      dimension v(0:nfun)
c
      parameter (nelc=2*nfun)
      dimension wt(nelc,nelc,nelc)
      dimension ve(nelc,nelc,nelc)
      dimension vo(nelc,nelc)
c
 1001 format (1x,d18.12)
 1002 format (1x,i4)
 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 eitfun_inp.dat'
      open (unit=23, status='old', file='eitfun_inp.dat')
      read (23,*) convrt
      read (23,*) hoverr
      read (23,*) radius
      read (23,*) curr12
      read (23,*) sigma0
      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
c
c *** Read in experimental voltages and normalize.
c
      write (6,2000) 'Reading experimental voltages from eitfun_exp.dat'
      open (unit=24, status='old', file='eitfun_exp.dat')
      do 0050 ip1 = 1, nelc-1, 1
      do 0050 ip2 = ip1+1, nelc, 1
         do 0040 ip = 1, nelc, 1
            read (24,*) m, n, k, vm, vr, vq
            ve(m,n,k) = vm * vltcon
 0040       continue
 0050    continue
      close (unit=24)
c
c *** Find mean voltages for each (m,n) projection.
c
      do 0150 m = 1, nelc-1, 1
      do 0140 n = m+1, nelc, 1
         vo(m,n) = 0.
         wo = 0.
         do 0130 k = 1, nelc, 1
            wo = wo + wt(m,n,k)
            vo(m,n) = vo(m,n) + wt(m,n,k) * ve(m,n,k)
 0130       continue
         vo(m,n) = vo(m,n) / wo
 0140    continue
 0150    continue
c
c *** Subtract mean voltages from experimental voltages.
c
      do 0250 m = 1, nelc-1, 1
      do 0240 n = m+1, nelc, 1
      do 0230 k = 1, nelc, 1
         ve(m,n,k) = ve(m,n,k) - vo(m,n)
 0230    continue
 0240    continue
 0250    continue
```

```
c
c *** Find the RHS vector and its sum.
c
      do 0350 m = 1, nelc-1, 1
      do 0340 n = m+1, nelc, 1
      do 0330 k = 1, nelc, 1
         i = abs(nfun-abs(m-k))
         j = abs(nfun-abs(n-k))
         f(i) = f(i) + ve(m,n,k) * wt(m,n,k)
         f(j) = f(j) - ve(m,n,k) * wt(m,n,k)
 0330    continue
 0340    continue
 0350    continue
c
      fs = 0.
      do 0380 i = 1, nfun-1, 1
         fs = fs + f(i)
 0380    continue
c
c *** Find the solution vector.
c
      znelc = dfloat(nelc)
      fac = 0.5 / ( znelc * ( znelc - 1. ) )
      v(0) = 0.
      do 0400 i = 1, nfun-1, 1
         v(i) = fac * ( f(i) + 2. * fs )
 0400    continue
c
      vnfun = 0.
      do 0450 m = 1, nelc-1, 1
      do 0440 n = m+1, nelc, 1
         ij = abs(nfun-abs(m-n))
         vnfun = vnfun + 0.5 * ( ve(m,n,m) - ve(m,n,n) ) + v(ij)
 0440    continue
 0450    continue
      v(nfun) = vnfun / dfloat(nelc*(nelc-1)/2)
c
c *** Write out fundamental solution.
c
      write (6,2000) 'Writing fundamental solution to eitfun_sol.dat'
      open (unit=27, status='unknown', file='eitfun_sol.dat')
      do 0700 ifun = 0, nfun, 1
         write (27,1001) v(ifun)
 0700    continue
      close (unit=27)
      do 0750 ifun = 0, nfun, 1
         write  (6,2001) '   v(i)    = ', v(ifun)
 0750    continue
c
c *** Write out input parameters.
c
      write (6,2000) 'Writing output parameters to eitfun_out.dat'
      open (unit=26, status='unknown', file='eitfun_out.dat')
      write (26,1001) convrt
      write (26,1001) hoverr
      write (26,1001) radius
      write (26,1001) curr12
      write (26,1001) sigma0
      close (unit=26)
c
c *** Completed, stop.
c
      stop 'eitfun'
```

```
      end
c
c23456789012345678901234567890123456789012345678901234567890123456789012
c
```