Appendix C

BoltzTraP for Computing Transport Properties

As explained in the last section of chapter 4, the program BoltzTraP was modified in order to include an energy and temperature dependent relaxation time $\tau(\varepsilon, T)$. The original version already allowed us to use such a τ but several mistakes were present giving totally incoherent results and modifications were necessary to include a user friendly equation for τ

$$\tau(\varepsilon, T) = \tau_0(\varepsilon_0, T_0) \left(\frac{T}{T_0}\right)^s \left(\frac{\varepsilon}{\varepsilon_0}\right)^{r-1/2} \tag{C.1}$$

where the five parameters τ_0, ε_0, T_0, s and r are input parameters.

In order to use this τ the original subroutine `fermiint_fix_ef_T` present in the file `FERMIINTEGRALS` has to be replaced by the following modified subroutine.

```
1
2 !-----------------------------------------------------------------
3 !  calculate transport properties for one fixed value of Fermi
     energy and T
4 !-----------------------------------------------------------------
5 subroutine fermiint_fix_ef_T(volume, efermi, deltaef, ebmin, temp
     , &
6     unit_tr, unit_cond, unit_hall, npoints, dos1, dos_sigxy,
         dos_sigxyz, spinorbit, icut1, prtef, &
7     splobj, dopage)
8
9  USE constants
10 USE input
11 USE bandstructure
12 use fermimod
13 implicit none
14
15 !arguments
16 real(8), intent(in) :: volume
```

```
17  real(8), intent(in) :: efermi, deltaef, ebmin, temp, dopage
18  real(8), intent(in) :: spinorbit
19  integer, intent(in) :: unit_tr, unit_cond, unit_hall, icut1,
       npoints, prtef
20
21  real(8), intent(in) :: dos1(0:npoints)
22  real(8), intent(in) :: dos_sigxy(3,3,0:npoints)
23  real(8), intent(in) :: dos_sigxyz(3,3,3,0:npoints)
24
25  type(fermispl_type), intent(in) :: splobj
26
27 ! local vars
28  INTEGER                 :: ialp,ibet
29  INTEGER                 :: i,j,k
30
31  REAL(8)                 :: ene
32  REAL(8)                 :: lifetime, lifetime_a,lifetime_o,
       lifetime_po
33  REAL(8)                 :: condtr,seebecktr,thermaltr,halltr,
       kappatr
34  REAL(8)                 :: specheat,magsus
35  REAL(8)                 :: sumelec,sumdos
36  REAL(8)                 :: factor1,factor2,factor3
37  REAL(8)                 :: taureffact_a,taureffact_o,
       taureffact_po
38  REAL(8)         :: delta,fpol
39
40  REAL(8)                 :: cond(3,3),nu(3,3),kappa(3,3),sigxyz
       (3,3,3)
41  REAL(8)                 :: minv_sigxy(3,3),seebeck(3,3),thermal
       (3,3),hall(3,3,3)
42
43 ! functions
44  REAL(8)                 :: fermi,dfermide,dfermidt
45
46 !source
47  specheat=ZERO
48  sumelec=ZERO
49  sumdos=ZERO
50  cond(1:3,1:3)=ZERO
51  nu(1:3,1:3)=ZERO
52  kappa(1:3,1:3)=ZERO
53  sigxyz(1:3,1:3,1:3)=ZERO
54  hall=ZERO
55  seebeck=ZERO
56  thermal=ZERO
57  taureffact_a = 1.d0
58  taureffact_o = 1.d0
59  taureffact_po = 1.d0
60
61
```

```
62 ! Acoustic phonons
63  if (taureftemp_a > 1.d-12) then
64   taureffact_a = exp(tempexp_a*log(temp/taureftemp_a)) * &
65       &  ((0.02459 + 8.659341d-5 * taureftemp_a) / (0.02459 +
            8.659341d-5 * temp))
66  end if
67
68 ! Optical phonons
69  if (taureftemp_o > 1.d-12) then
70   taureffact_o = exp(tempexp_o*log(temp/taureftemp_o)) * &
71       &  ((0.02459 + 8.659341d-5 * taureftemp_o) / (0.02459 +
            8.659341d-5 * temp))
72  end if
73
74
75 ! Polar optical phonons
76  if (taureftemp_po > 1.d-12) then
77   taureffact_po = exp(tempexp_po*log(temp/taureftemp_po)) * &
78     &  exp(1/THREE*log((0.02459 + 8.659341d-5 * taureftemp_po) /
           &
79       & (0.02459 + 8.659341d-5 * temp)))
80  end if
81
82
83
84 ! this sum does an integral over the energy, multiplied by the FD
     distribution and/or its derivatives wrt epsilon or T
85  DO i=0,npoints
86     ene=(ebmin+deltaef*(REAL(i)+HALF))
87     lifetime_a = 1.d0
88     lifetime_o = 1.d0
89     lifetime_po = 1.d0
90
91     if (abs(tauref_a) > 1.d-10) then
92       lifetime_a = (tauref_a*SECOND*1.e-15) * &
93         &  exp((tauexp_a-0.5d0) * log(abs(ene / (taurefen_a*
              RYDBERG)))) * &
94         &  taureffact_a
95     end if
96
97     if (abs(tauref_o) > 1.d-10) then
98       lifetime_o = (tauref_o*SECOND*1.e-15) * &
99         &  exp((tauexp_o-0.5d0) * log(abs(ene / (taurefen_o*
              RYDBERG)))) * &
100        &  taureffact_o
101    end if
102
103    if (abs(tauref_po) > 1.d-10) then
104      lifetime_po = (tauref_po*SECOND*1.e-15) * &
105        &  exp((tauexp_po-0.5d0) * log(abs(ene / (taurefen_po*
              RYDBERG)))) * &
```

```
106          &  taureffact_po
107      end if
108
109 !     if (abs(tauref_po) > 1.d-10) then
110 !       lifetime_po = (tauref_po) * &
111 !         &  exp((tauexp_po) * log(abs(ene))) * &
112 !         &  exp(tempexp_po * log(temp)) * SECOND
113 !     end if
114
115 ! delta = 2.2262d+5 * 5e19/(abs(dopage)) * abs(ene) * temp*
    BOLTZMANN * &
116 !    &  exp(1/THREE*log(0.24 * (0.02459 + 8.659341d-5 * temp) **
    2))
117 !
118 ! fpol = 1 - 2*log(delta+1)/delta + 1/(delta+1)
119 !
120 !     if (abs(tauref_po) > 1.d-10) then
121 !       lifetime_po = (tauref_po*SECOND*1.e-15) * &
122 !         &  exp((tauexp_po-0.5d0) * log(abs(ene / (taurefen_po*
    RYDBERG)))) * &
123 !         &  taureffact_po/fpol
124 !     end if
125
126      lifetime = ((1/lifetime_a) + (1/lifetime_o) + (1/lifetime_po)
           ) ** (-1)
127
128
129      factor1=fermi(ene,efermi,temp)*spinorbit
130      factor2=dfermide(ene,efermi,temp)*spinorbit
131      factor3=dfermidt(ene,efermi,temp)*spinorbit
132      !factor1=fermi_(ene,efermi,temp,splobj)*spinorbit
133      !factor2=dfermide_(ene,efermi,temp,splobj)*spinorbit
134      !factor3=dfermidt_(ene,efermi,temp,splobj)*spinorbit
135
136      sumelec=sumelec+dos1(i)*factor1
137      sumdos=sumdos+dos1(i)*(-factor2)
138      specheat=specheat+dos1(i)*volume*(ene-efermi)*factor3
139      cond(1:3,1:3)=cond(1:3,1:3) + lifetime*dos_sigxy(1:3,1:3,i)*
           factor2
140      nu(1:3,1:3)=nu(1:3,1:3) + lifetime*dos_sigxy(1:3,1:3,i)*
           factor2*(ene-efermi)
141      kappa(1:3,1:3)=kappa(1:3,1:3) + lifetime*dos_sigxy(1:3,1:3,i)
           *factor2*(ene-efermi)**2
142      sigxyz(1:3,1:3,1:3)=sigxyz(1:3,1:3,1:3) + lifetime**2 *
           dos_sigxyz(1:3,1:3,1:3,i)*factor2
143   ENDDO
144
145
146
147
148 ! Convert to SI units
```

```
149  IF(sumdos>TWO*TEST) THEN
150   if(tauref_a > 1.d-10) then
151       cond(1:3,1:3) = -cond(1:3,1:3)*deltaef &
152            * OHM * METER
153       nu(1:3,1:3)    =    -nu(1:3,1:3)*deltaef/temp &
154            * METER / AMPERE
155       kappa(1:3,1:3)=-kappa(1:3,1:3)*deltaef/temp &
156            * METER / WATT
157       sigxyz(1:3,1:3,1:3)=-sigxyz*deltaef &
158            * OHM**2 * SECOND * AMPERE / METER
159       specheat       = specheat*deltaef &
160            * MOL / JOULE
161     else
162       cond(1:3,1:3) = -cond(1:3,1:3)*deltaef &
163            * OHM * METER * SECOND
164       nu(1:3,1:3)    =    -nu(1:3,1:3)*deltaef/temp &
165            * METER * SECOND / AMPERE
166       kappa(1:3,1:3)=-kappa(1:3,1:3)*deltaef/temp &
167            * METER * SECOND / WATT
168       sigxyz(1:3,1:3,1:3)=-sigxyz*deltaef &
169            * OHM**2 * SECOND**3 * AMPERE / METER
170       specheat       = specheat*deltaef &
171            * MOL / JOULE
172     end if
173
174     CALL invert_3x3(cond,minv_sigxy,i)
175  ! for many energy points if dos is small but not small enough
        this explodes a bit on the diagonal elements
176  ! R_Hall is then huge, but never if dos(ef) > 1.e-10
177     seebeck(1:3,1:3)=ZERO
178     DO i=1,3
179        DO j=1,3
180           DO ialp=1,3
181              seebeck(i,j)=seebeck(i,j)-minv_sigxy(ialp,i)*nu(ialp
                    ,j)
182           ENDDO
183        ENDDO
184     ENDDO
185     thermal(1:3,1:3)=kappa(1:3,1:3)
186     hall(1:3,1:3,1:3)=ZERO
187     DO i=1,3
188        DO j=1,3
189           DO k=1,3
190              DO ialp=1,3
191                 DO ibet=1,3
192                    hall(i,j,k)=hall(i,j,k)+minv_sigxy(ialp,j)*
                          sigxyz(ialp,ibet,k)*minv_sigxy(i,ibet)
193                 ENDDO
194              ENDDO
195           ENDDO
196        ENDDO
```

```
197     ENDDO
198  ENDIF
199
200
201 !!Convert to SI units
202 !  IF(sumdos>TWO*TEST) THEN
203 !     cond(1:3,1:3) = -cond(1:3,1:3)*deltaef &
204 !          * OHM * METER * SECOND
205 !     nu(1:3,1:3)    =    -nu(1:3,1:3)*deltaef/temp &
206 !          * METER * SECOND / AMPERE
207 !     kappa(1:3,1:3)=-kappa(1:3,1:3)*deltaef/temp &
208 !          * METER * SECOND / WATT
209 !     sigxyz(1:3,1:3,1:3)=-sigxyz*deltaef &
210 !          * OHM**2 * SECOND**3 * AMPERE / METER
211 !     specheat       = specheat*deltaef &
212 !          * MOL / JOULE
213 !     CALL invert_3x3(cond,minv_sigxy,i)
214 !! for many energy points if dos is small but not small enough
    this explodes a bit on the diagonal elements
215 !! R_Hall is then huge, but never if dos(ef) > 1.e-10
216 !     seebeck(1:3,1:3)=ZERO
217 !     DO i=1,3
218 !        DO j=1,3
219 !           DO ialp=1,3
220 !              seebeck(i,j)=seebeck(i,j)-minv_sigxy(ialp,i)*nu(
    ialp,j)
221 !           ENDDO
222 !        ENDDO
223 !     ENDDO
224 !     thermal(1:3,1:3)=kappa(1:3,1:3)
225 !     hall(1:3,1:3,1:3)=ZERO
226 !     DO i=1,3
227 !        DO j=1,3
228 !           DO k=1,3
229 !              DO ialp=1,3
230 !                 DO ibet=1,3
231 !                    hall(i,j,k)=hall(i,j,k)+minv_sigxy(ialp,j)*
    sigxyz(ialp,ibet,k)*minv_sigxy(i,ibet)
232 !                 ENDDO
233 !              ENDDO
234 !           ENDDO
235 !        ENDDO
236 !     ENDDO
237 ! ENDIF
238
239
240  seebecktr=(seebeck(1,1)+seebeck(2,2)+seebeck(3,3))/THREE
241  condtr=(cond(1,1)+cond(2,2)+cond(3,3))/THREE
242  thermaltr=(thermal(1,1)+thermal(2,2)+thermal(3,3))/THREE
243  kappatr=(kappa(1,1)+kappa(2,2)+kappa(3,3))/THREE
244  halltr=(hall(2,3,1)+hall(3,1,2)+hall(1,2,3))/THREE
```

```
245
246  sumelec=nval-(icut1-1)*spinorbit-(sumelec*deltaef*volume)
247  sumdos=sumdos*deltaef*volume
248  magsus=sumdos*MU0*MUB**2*MOL/METER**3
249
250  if (prtef == 1) then
251 !  Units of traces:    Ry            K     e/uc    e/uc    V/K
          1/(ohm m)*1/s m^3/C  W/mK*1/s   J/(mol K)
252    WRITE(unit_tr,101) efermi/RYDBERG,temp,sumelec,sumdos,
         seebecktr,condtr       ,halltr,thermaltr, specheat,magsus
253    WRITE(unit_cond,101) efermi/RYDBERG,temp,sumelec,cond(1:3,1:3)
         ,seebeck(1:3,1:3),thermal(1:3,1:3)
254    WRITE(unit_hall,101) efermi/RYDBERG,temp,sumelec,hall
         (1:3,1:3,1:3) ! **** c11,15,25 are yzx,zxy,xyz respectively
         ***
255  else
256    WRITE(unit_tr,102)   temp,sumelec,sumdos,seebecktr,condtr
              ,halltr,thermaltr, specheat,magsus,efermi/RYDBERG
257    WRITE(unit_cond,102) temp,sumelec,cond(1:3,1:3),seebeck
         (1:3,1:3),thermal(1:3,1:3),efermi/RYDBERG
258    WRITE(unit_hall,102) temp,sumelec,hall(1:3,1:3,1:3),efermi/
         RYDBERG ! **** c11,15,25 are yzx,zxy,xyz respectively ***
259  end if
260
261 !100  FORMAT(F10.5,F10.4,F16.8,F16.8,9E16.8)
262  101  FORMAT(F10.5,F10.4,F16.8,27E16.8)
263 ! same without the Fermi energy
264  102  FORMAT(F10.4,F16.8,27E16.8,F10.5)
265
266 end subroutine fermiint_fix_ef_T
```