REFERENCES

[1] Kittel, C., **Introduction to Solid State Physics,** 6th Ed., Singapore: John Wiley & Sons Inc., 1991.

[2] Ashcroft, N., **Physics World 8,** 43 (1995).

[3] Mott, N.F., and E.A. Davis, **Electronic Processes in Non-Crystalline Materials,** 2nd Ed., Oxford University Press, 1979.

[4] Ross, M., **Phys. Rev. B 54**, 9589 (1996).

[5] Weir, S.T., A.C., Mitchell and W.J., Nellis, **Phys. Rev. Lett. 76**, 1860 (1996).

[6] Nellis, W.J. et al., **Cond-mat/9708144** (1997).

[7] Hensel, F. and Edwards, P.P., **Physics World,** April, 43. (1996).

[8] Ziman, J.M., **Phil. Mag. 6,** 1013 (1961).

[9] Xu, H. and J.P.,Hansen, **Phys. Rev. E 57,** 211 (1998).

[10] Abramowitz, M. and I. R. Stegun, **Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables,** 2nd Ed., Dover Publications., New York, 1972.

[11] Madelung, O., **Introduction to Solid State Theory,** 2nd Ed., Springer-Verlag, 1981.

[12] Ziman, J.M., **Principles of the Theory of Solids,** 2nd Ed., Cambridge University Press, 1972.

[13] Reif, F., **Fundamentals of Statistical and Thermal Physics,** International Ed., Singapore: McGraw-Hill, 1985.

[14] Raimes, S., **The Wave mechanics of Electrons in Metal,** Amsterdam: North-Holland, 1961.

[15] Goodstien, D.L, **States of Matter,** Prentice-Hall, 1975.

[16] Fournet,G., **Handb. Phys.10**, 49 (1957).

[17] Merzbacher, E., **Quantum Mechanics**, 3rd Ed., Wiley & Sons, 1998.

[18] Sakurai, J.J., **Quantum Mechanics,** Revised Ed., Addison-Wesley, 1982.

[19] Faber, T.E., **IAEA-SMR-46/125.,** ---,645.

[20] Kwon, I., L. Collins, J. Kress, And N. Troullier, **Phys. Rev. E 54,** 2844(1996).

[21] Pines, D. and P. Nozieres, **The Theory of Quantum Liquids: Normal Fermi liquids,** Addison-Wesley, Vol. 1, 1989.

[22] Chihara, J., **Phys. Rev. A 33,** 2575 (1986).

[23] Hildebrand, F. B., **Introduction to Numerical Analysis,** New Delhi, Tata McGraw-Hill, 1956.

[24] Ralston, A., **A first Course in Numerical Analysis,** Tokyo: McGraw-Hill, 1965.

[25] Hoffman, J. D., **Numerical Methods for Engineers and Scientists,** McGraw-Hill, 1993.

[26] Abell, L. M. and J. P. Braselton, **Mathematica by Example,** 2nd Ed., Academic Press, 1997.

APPENDIX A

GAUSS-LEGENDRE INTEGRATION TECHNIQUE

A.1 GAUSS-LEGENDRE QUADRATURE [23]

As one discussed in Chapter IV, the d.c. electrical conductivity of liquid metallic hydrogen was calculated. All involved integration in a source code has used the Gauss-Legendre integration technique. Now one would like to explain briefly to this technique.

In the case of an interval of integration $[a,b]=[-1,1]$, the sequence of polynomials required must be orthogonal over $[-1,1]$. Such a sequence of orthogonal polynomials is the Legendre polynomials. The integration formula based on a polynomials is called the Gauss-Legendre quadrature formula,

$$\int_{-1}^{1} f(x)\,dx = \sum_{i=1}^{n} \alpha_{i,n} f(x_i) + E_n[f]. \qquad \text{(A.1.1)}$$

where

$\alpha_{i,n}$ is weighting factor,

$f(x_i)$ the known ith point of function $f(x)$ and

$E_n[f]$ the fractional remainder or error terms.

The values of nodes x_i and coefficients $\alpha_{i,n}$ of Gauss-Legendre quadrature formula, for example, for some n are given in the source code.

To evaluate the integral $\int_a^b f(x)\,dx$ using Gauss-Legendre quadrature formula, one must first transform the interval of integration $[a,b]$ to $[-1,1]$ by changing the variable x as

$$y = \tau x + \varepsilon \qquad \text{(A.1.2)}$$

where τ and ε are dummy variables.

The requirements of limit of integration are

$$y = 1 \text{ when } x = b$$

and

$$y = -1 \text{ when } x = a .$$

Thus one obtains

$$y = \frac{(2x - a - b)}{(b - a)},$$

$$x = \frac{[(b - a)y + (a + b)]}{2}.$$

And then integration $\int_a^b f(x)dx$ will become

$$\int_a^b f(x)\,dx = \left[\frac{(b-a)}{2}\right] \sum_{i=1}^{n} \alpha_{i,n}\, g(y_i) + E_n[g] \qquad \text{(A.1.3)}$$

where

$$g(y) = f\left(\frac{[(a-b)y + (a+b)]}{2}\right). \qquad \text{(A.1.4)}$$

y_i and $\alpha_{i,n}$ are nodes and coefficients of Gauss-Legendre quadrature formula of order n. The associated error term is

$$E_n[g] = (b-a)^{2n+1}(n!)^4 \frac{f^{(2n)}(\eta)}{(2n+1)(2n!)^3(2n!)^3} \quad , a < \eta < b . \qquad \text{(A.1.5)}$$

A.2 COMPOSITE GAUSS-LEGENDRE QUADRATURE FORMULA [24]

Breaking up the interval $[a,b]$ into a number s of subintervals can reduce the magnitude of error term. To each subinterval apply an n-points quadrature formula in every subinterval and then sum these results. Thus an extra factor, $1/s^{2n}$, will be introduced into the error term,

$$E_n[g]_s = (b-a)^{2n+1}(n!)^4 \frac{f^{(2n)}(\eta)}{(2n+1)(2n!)^3 s^{2n}} \quad , a < \eta < b . \qquad \text{(A.2.1)}$$

This formula can be rewritten in the form

$$\int_a^b f(x)\,dx = \sum_{i=1}^{n} \alpha_{i,n} \sum_{j=1}^{s} f\left(a + (b-a)\left(\frac{x_i + 2j - 1}{2}\right)\right) + E_n[f]_s . \qquad \text{(A.2.2)}$$

A quadrature formula of this type is called the composite quadrature formula.

A.3 THE SOURCE CODE FOR CALCULATION

As discussed in section 4.4, the source code using the C language for electrical conductivity is given. In this program a radial distribution function of liquid metallic hydrogen is provided in the tw0_5(x) function, 198 known points. The function to calculate the structure factor is ak (k,C).

```
/** Electrical Conductivity of Liquid Metallic Hydrogen **/

/*rs=0.5 at Temperature 3000 K*/

#include<stdio.h>
#include<math.h>

#define zp 1.0                          /* Proton number or Atomicnumber*/
#define e_m 9.10956e-28                 /* Electron's mass*/
#define e_cq 4.8032068e-10              /* Electron's charge */
#define PI 3.141592653589793238462
#define p_m 1.673e-24                   /* Proton's mass */
#define ry 2.180e-18                    /* Rydberg's constant */
#define hp 6.626075e-27                 /* Planck's constant */
#define hbar 1.054e-27                  /* hbar=h/2PI */
#define e_0 1 /*SI 8.854e-12*/          /* Permittivity of free space */
#define mu_0 1 /*SI 1.257e-6*/          /* Permeability of free space */
#define c 2.998e10                      /* Light's velocity */
#define kb 1.380e-23                    /* Boltzmann's constant */
#define av 6.02e23                      /* Avogrado's number */
#define aB 0.529177e-10                 /* Bohr's radius */

double rs=0.5;                          /* r_s= r/a_B */

void main(void)
{
   int i,j,N;
```

```
double au,k,l,R,avn,r,rs,kf,n;
double sum1,st,tr,lam,lama,lamb;
double a,tw0_5(),ak(),ua,ub,meanf,Ef;
double g,b,h,s,s1,sum,e,a1,a2,d,vf,cond;
double z[48],w[48];
double abcissa(),weighting();
FILE *fp1,*fp2;

fp1 = fopen("1rs0_5.dat","w");
if(fp1== NULL) {
        printf("cannot open file 1rs0_5.dat\n");
        exit(0);
}
fp2 = fopen("2rs0_5.dat","w");
if(fp2 == NULL) {
        printf("cannot open file 2rs0_5.dat\n");
        exit(0);
}
rs = 0.5;
 n = pow(rs,3)*4*PI/3;
 kf = pow(3*PI*PI*n,0.33);
l = 0.02*kf/(2*kf);

 for(i=0;i<=1999;i++){
          tr=ak(l,n);
          fprintf(fp1,"{%.2f,%lf},\n",l,tr);
          l+=0.01;
 }
 fprintf(fp1,"\nelectron concentration n=%le mol. per c.c.\n",n);
 fprintf(fp1,"\nProton concentration n=%le mol. per c.c.\n",n);
 fclose(fp1);
 fprintf(fp2,"\n\tFermi wave vector kf=%le (1/cm)\n",kf);
```

```
fprintf(fp2,"\n\tProton concentration n=%le mol. per c.c.\n",n);


a1 = 0.0;
a2 = 2*kf;
d = a2-a1;
N = 100;
e = d/(2*N);
sum = 0.0;
lam = hbar*hbar*kf*kf/(12*PI*n*e_m*pow(e_cq,2));
s1 = 0.0;
        for(j=0;j<=47;j++){
                z[j] = abcissa(j);
                w[j] = weighting(j);
                for(i=0;i<=N-1;i++){
                        h = 2.0*i+1.0;
                        a = a1+e*(h-z[j]);
                        ua = 4*PI*e_cq*lam/(1+lam*pow(a,2));
                        b = a1+e*(h+z[j]);
                        ub = 4*PI*e_cq*lam/(1+lam*pow(b,2));
                        s = ua*ua*ak(a,n)*pow(a,3)+ub*ub*ak(b,n)*pow(b,3);
                        s1 = s1+s*0.01;
                }
      sum = sum+s1*w[j];
      }
au = sum*e/(4*pow(kf,4));
fprintf(fp2,"\n\tElectron concentration n=%le mol.per c.c.\n",n);
Ef = pow(hbar*kf,2)/(2*e_m);
fprintf(fp2,"\n\tEf=%le Rydberg or= %lf eV\n",Ef/rs,Ef);
meanf = pow(2*Ef,1.5)/(3*PI*au);
fprintf(fp2,"\n\tmean free path=%le cm\n",meanf);
vf = hbar*kf/e_m;
fprintf(fp2,"\n\tFermi velocity vf=%le cm/s\n",vf);
```

```
    cond = n*pow(e_cq,2)*meanf/(e_m*vf);
    fprintf(fp2,"\n\tElectrical conductivity of metallicH=%le 1/(Ohm cm)\n",cond);
    fprintf(fp2,"\n\tResistivity=%le  Ohm cm\n",1/cond);
    fclose(fp2);
}
/*****************************************************************************/

                                    double ak(k,C)

/*****************************************************************************/

     double k;
     double C;
{
        int i,j,N;
        double h,s,s1,sum,e,a1,a2,d;
        float gx,gy,a,b;
        double tw();
        double abcissa(),weighting();
        double z[48],w[48];
        a1 = 0.0;
        a2 = 7.4;
        d = a2-a1;
        N = 100;
        e = d/(2*N);
        sum = 0.0;
        for(j=0;j<=47;j++){
                s1 = 0.0;
                z[j] = abcissa(j);
                w[j] = weighting(j);
                for(i=0;i<=N-1;i++){
                        h = 2.0*i+1.0;
```

```
                    a = a1+e*(h-z[j]);
                    b = a1+e*(h+z[j]);
                    gx = tw0_5(a); /* gx and gy stand for the radial distribution function*/
                    gy = tw0_5(b);
                    s = (gx-1)*sin(k*a)*a+(gy-1)*sin(k*b)*b;
                    s1 = s1+s;
                }
                sum = sum+s1*w[j];
        }
        sum = 0.1*sum*e;
        sum = 1.0 + C*4*PI*sum/k;
        return(sum);
}

/*****************************************************************************/

                                double abcissa(j)

/*****************************************************************************/

        int j;
{
 double z[48]={1.6276744849602970e-2,
4.8812985136049731e-2, 8.129749546425559e-2, 0.11369585011066592,
0.14597371465489694, 0.17809688236761860, 0.21003131046056720, 0.24174315616384001,
0.27319881259104914, 0.30436494435449635, 0.33520852289262542, 0.36569686147231364,
0.39579764982890860, 0.42547898840730054, 0.45470942216774301, 0.48345797392059636,
0.51169417715466767, 0.53938810832435744, 0.56651041856139717, 0.59303236477757208,
0.61892584012546857, 0.64416340378496711, 0.66871831004391615, 0.69256453664217156,
0.71567681234896763, 0.73803064374440013, 0.75960234117664750, 0.78036904386743322,
0.80030874413914082, 0.81940031073793168, 0.83762351122818712, 0.85495903343460146,
0.87138850590929650, 0.88689451740242042, 0.90146063531585234, 0.91507142312089807,
```

```
0.92771245672230869, 0.93937033975275522, 0.95003271778443764, 0.95968829144874254,
0.96832682846326421, 0.97593917458513647, 0.98251726356301468, 0.98805412632962380,
0.99254390032376262, 0.99598184298720929,0.99836437586318168, 0.99968950388323077};
        return(z[j]);
}
/*****************************************************************************/

                                double weighting(j)

/*****************************************************************************/

int j;
{
double w[48]={0.32550614492363166e-1,
0.32516118713868836e-1,0.32447163714064269e-1, 0.32343822568575928e-1,
0.32206204794030251e-1,0.32034456231992663e-1,0.31828758894411006e-
1,0.31589330770727168e-1,0.31316425596861355e-1,0.31010332586313837e-
1,0.30671376123669149e-1,0.30299915420827594e-1,0.29896344136328386e-
1,0.29461089958167906e-1,0.28994614150555236e-1,0.28497411065085386e-
1,0.27970007616848334e-1,0.27412962726029243e-1,0.26826866725591762e-
1,0.26212340735672414e-1,0.25570036005349361e-1,0.24900633222483610e-
1,0.24204841792364691e-1,0.23483399085926220e-1,0.22737069658329374e-
1,0.21966644438744349e-1,0.21172939892191300e-1,0.20356797154333324e-
1,0.19519081140145022e-1,0.18660679627411467e-1,0.17782502316045261e-
1,0.16885479864245172e-1,0.15970562902562291e-1,0.15038721026994938e-
1,0.14090941772314861e-1,0.13128229566961573e-1,0.12151604671088320e-
1,0.11162102099838498e-1,0.10160770535008416e-1,0.91486712307833866e-
2,0.81268769256987592e-2,0.70964707911538653e-2,0.60585455042359617e-
2,0.50142027429275177e-2,0.39645543384446867e-2,0.29107318179349464e-
2,0.18539607889469217e-2,0.79679206555201243e-3};
                return(w[j]);
}
```

```
/*****************************************************************************/

                        double tw0_5(x)

/*****************************************************************************/
        double x;
{
double X[198]={0.00, 1.00, 1.15, 1.19, 1.22, 1.23, 1.24, 1.25, 1.26, 1.27, 1.28, 1.30, 1.31, 1.31,
1.32, 1.33, 1.33,1.33, 1.34, 1.346, 1.35, 1.354, 1.357, 1.36, 1.371, 1.376, 1.385, 1.3855, 1.39,
1.396, 1.3965,1.401, 1.402, 1.407, 1.4075, 1.4125, 1.4180, 1.4185, 1.423, 1.427, 1.428, 1.431,
1.432, 1.435,1.436, 1.438, 1.50, 1.532, 1.538, 1.56, 1.57, 1.59, 1.60, 1.61, 1.62, 1.626, 1.629,
1.63, 1.64, 1.65,1.67, 1.69,1.70, 1.71, 1.72, 1.73, 1.74, 1.75, 1.76, 1.77, 1.78,1.783, 1.788, 1.80,
1.84, 1.88, 1.92, 1.96, 2.00, 2.03,2.07, 2.079, 2.11, 2.15, 2.19, 2.23, 2.26, 2.30, 2.34, 2.38, 2.42,
2.46, 2.50, 2.53, 2.57, 2.61, 2.65, 2.69, 2.692,  2.73, 2.76, 2.80, 2.84, 2.88, 2.92, 2.96, 2.962, 3.00,
3.03, 3.07, 3.11, 3.15, 3.19, 3.23, 3.26, 3.30,3.34, 3.38, 3.42, 3.46, 3.50, 3.53, 3.57, 3.61, 3.62,
3.65, 3.73, 3.76, 3.80, 3.84, 3.88, 3.92, 3.96, 4.00,4.03, 4.07, 4.11, 4.15, 4.19, 4.23, 4.26, 4.30,
4.34, 4.38, 4.42, 4.46, 4.50, 4.53, 4.57, 4.61, 4.65, 4.69, 4.73, 4.76, 4.80, 4.84, 4.88, 4.92, 4.96,
5.00, 5.03, 5.07, 5.11, 5.15, 5.19, 5.23, 5.26, 5.30, 5.34, 5.38, 5.42, 5.46, 5.50, 5.53, 5.57, 5.61,
5.65, 5.69, 5.73, 5.76, 5.80, 5.84, 5.88, 5.92, 5.96, 6.00, 6.03, 6.17, 6.35, 6.53, 6.82, 6.89, 7.00,
7.14, 7.28, 7.39, 7.40, 7.50};
double F[198]={0.00, 0.00, 0.00, 0.00, 0.0012, 0.0035, 0.0059, 0.0083, 0.01, 0.014, 0.02, 0.04,
0.05, 0.07, 0.08, 0.10, 0.11, 0.11, 0.12, 0.13, 0.15, 0.16, 0.17, 0.20, 0.22, 0.23, 0.24, 0.25, 0.27,
0.28, 0.30, 0.31, 0.32, 0.34, 0.35, 0.37, 0.38, 0.40, 0.41, 0.42, 0.44, 0.45, 0.47, 0.48, 0.50, 0.51,
1.24, 1.50, 1.51, 1.70, 1.80, 1.90, 1.91,1.93, 1.95, 1.96, 1.98, 2.00, 2.04, 2.04, 2.04, 2.04, 2.046,
2.042, 2.02, 2.01, 2.00, 1.98, 1.96, 1.95, 1.93, 1.91, 1.90, 1.89, 1.68, 1.57, 1.42, 1.30, 1.15, 1.08,
1.025, 1.028, 0.95, 0.90, 0.85, 0.80, 0.77, 0.75, 0.73, 0.72, 0.71, 0.714, 0.7142, 0.72, 0.74, 0.75,
0.77, 0.80, 0.81, 0.82, 0.85, 0.88, 0.92, 0.95, 1.00, 1.02, 1.04, 1.07, 1.09, 1.12, 1.15, 1.16, 1.176,
1.178, 1.171, 1.16, 1.15, 1.13, 1.11, 1.09, 1.06, 1.05, 1.04, 1.01, 1.00, 0.95, 0.93, 0.928, 0.921,
0.914, 0.91, 0.90, 0.903, 0.914, 0.914, 0.928, 0.929, 0.942, 0.9423, 0.956, 0.95, 0.97, 0.985, 0.99,
1.00, 1.01, 1.02, 1.029, 1.03, 1.035, 1.038, 1.039, 1.04, 1.039, 1.038, 1.03, 1.03, 1.028, 1.02, 1.01,
1.007, 1.00, 0.995, 0.992, 0.985, 0.98, 0.978, 0.976, 0.971, 0.9714, 0.964, 0.965, 0.971, 0.972,
```

```
0.9720, 0.972, 0.978, 0.9850, 0.9853, 0.9857, 0.9860, 0.99, 1.00, 1.007, 1.014, 1.017, 1.029,
1.023, 1.00, 0.988, 0.981, 0.99, 1.001, 1.004,1.00,1.00,1.00};
        int i,j,k;
        double gx,p;

        for(i=0;i<=197;i++){
            if((X[i]<=x) && (x<=X[i+1])) {
            gx = 0.0;
            for(j=i;j<=i+1;j++){
                    p = 1.0;
                    for(k=i;k<=i+1;k++){
                            if(j!=k)
                                    p*=(x-X[k])/(X[j]-X[k]);
                    }
            gx = gx+p*F[j] ;
            }
          }
        }
        return(gx);
}
```

APPENDIX B

ANALYTICAL FORM OF RADIAL DISTRIBUTION FUNCTION

B.1 LEAST SQUARES APPROXIMATION

An approximate fitting yields a polynomial that passes through the set of points in the best possible manner without being required to pass exactly through any of the points. Consider the set of discrete points $[x_i, Y(x_i)] = (x_i, Y_i)$ and the approximate polynomial $y(x)$ chooses to represent the set of discrete points. These discrete points do not fall on the approximating polynomial. The deviations of the points from the approximating function,

$$e_i = Y_i - y_i, \tag{B.1.1}$$

must be minimized in some manner.

The least square procedure is defined as follows [25]. Given N data points $[x_i, Y(x_i)] = (x_i, Y_i)$, choose the functional form of the approximating function to be fit, $y = y(x)$, and minimized the sum of the squares of deviations e_i.

For the higher-degree polynomial approximation,

$$y = \sum_{i=0}^{n} a_i x^i, \tag{B.1.2}$$

where

n is an integer,

a_i a coefficients,

the sum of the squares of the deviations is given by

$$S(a_0, a_1, ..., a_n) = \sum_{i=1}^{N} (e_i)^2 \qquad \text{(B.1.3a)}$$

$$= \sum_{i=1}^{N} \left(Y_i - a_0 - a_1 x_i - , ..., - a_n x_i^n\right)^2 \qquad \text{(B.1.3b)}$$

The function $S(a_0, a_1, ..., a_n)$ is minimum when all partial derivative S with respect to a_i equal to zero.

$$\frac{\partial}{\partial a_i} S(a_0, a_1, ..., a_i, ..., a_n) = 0 \qquad \text{(B.1.4)}$$

As a consequence, all coefficients a_i can be determined by Gaussian elimination.

B.2 ANALYTIC FORM OF RADIAL DISTRIBUTION FUNCTION

In order to get analytical form of proton-proton radial distribution function $g(r)$, for $r_S = 0.5$ at 3000 K from Xu and Hansen's paper [9], we use the least-square method to apply to 198 given points. It is easily to attain this objective by using a Mathematica program [26]. The following real coefficients are provided in pair in form of (power of x, its coefficient), where x stands for $\frac{r}{a_1}$, r is in the Bohr unit, and a_1 as previously defined.

(-32, 3478.75), (-31, 1653.04), (-30, 137.461), (-29, -1031.25), (-28, -1827.35), (-27, -2240.11), (-26, -2277.98), (-25, -1972.17), (-24, -1379.20), (-23, -581.523), (-22, 314.616), (-21, 1185.07), (-20, 1897.16), (-19, 2329.97), (-18, 2383.89), (-17, 2012.82), (-16, 1240.34), (-15, 175.346), (-14, -987.237), (-13, -1987.65), (-12, -2550.27), (-11, -2459.88), (-10, -1653.3), (-9, -254.413), (-8, 1199.83),

(-7, 2261.57), (-6, 2380.58), (-5, 1410.22), (-4, -209.745), (-3, -1543), (-2, 1710.87), (0, 1479.25), (1, 405.124), (2, -425.162), (3, -453.514), (4, 124.037), (5, 396.496), (6, 470.295), (7, -15.1896), (8, 0.837143), (9, 0.0607708), (10, -0.0111314), (11, 0.00045847).

And the added Associated Legendre P (6,3) part is

$$33.1178\, i\sqrt{1-x^2}\,\left(x^2-1\right)\left(-3x+11x^3\right).$$

In addition, the last expression that is added to this polynomial is $-1479.25\,exp\left(-x^3\right)$. This analytical $g(r)$ result is shown in Fig. 3.1.

CURRICULUM VITAE

Mr.Chatchawal Sripakdee was born on January 4, 1971 in Roi-Ed. He received his B.Ed. degree in Physics from Nakhonrajsima Rajpat Institute in 1993. He is a teacher at Saithongwitaya secondary school, amphor Phonsai, changwat Roi-Ed.