รายการอ้างอิง

[1] สมบูรณ์ รัศมี. การเปลี่ยนแปลงอุณหภูมิเทียบกับเวลาในโลหะทรงกลมตันซึ่งตกอย่างอิสระภายใต้การเดือดเป็นชั้นฟิล์ม. วิทยานิพนธ์ปริญญามหาบัณฑิต ภาควิชานิวเคลียร์เทคโนโลยี วิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย,2548.

[2] เต็มศิริ ป้อมประภา. การจำลองการเดือดเป็นชั้นฟิล์มที่เกิดกับเวลาบนพื้นผิววัตถุทรงกลม. วิทยานิพนธ์ปริญญามหาบัณฑิต ภาควิชานิวเคลียร์เทคโนโลยี วิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย,2545.

[3] สัญชัย นิลสุวรรณโฆษิต. A Numerical model for Time Dependent Film Boiling on a sphere. The Fourth Annual National Symposium on Computational Science and Engineering March 2000.

[4] K.H. BANG. Numerical prediction of forced convection film boiling heat transfer from sphere. Int.J. Heat Mass Transfer 37, 16 (1994) :2415-2424

[5] มนตรี พิรุณเกษตร. การถ่ายเทความร้อน. พิมพ์ครั้งที่ 1. กรุงเทพมหานคร: สำนักพิมพ์วิทยพัฒน์ , 2545.

[6] Nukiyama, S. The Maximum and Minimum Values of the Heat Q Transmitted from Metal to Boiling under Atmospheric Pressure. Journal Japan Soc. Mech. Engrs 37 (1934): 367-374

[7] Farber, E.A. and Scorah, R.L. Heat Transfer to Water Boiling under Pressure. Trans. ASME 79 (1948): 369-384

[8] สุนันท์ ศรัณยนิตย์. การถ่ายเทความร้อน.พิมพ์ครั้งที่ 1. กรุงเทพมหานคร: สำนักพิมพ์สมาคมส่งเสริมเทคโนโลยี (ไทย-ญี่ปุ่น), 2545.

[9] Ozisik,M.N. Heat Transfer. McGrall-Hill ,1985

[10] R. Byron Brid,Warren E. Stewart and Edwin N.Lightfoot. Transport Phenomena. Second Edition. John Wiley & Sons, 2002

[11] ปราโมทย์ เดชะอำไพ. ระเบียบเชิงตัวเลขในงานวิศวกรรม. พิมพ์ครั้งที่ 2. กรุงเทพมหานคร: สำนักพิมพ์แห่งจุฬาลงกรณ์มหาวิทยา, 2541.

ภาคผนวก

ภาคผนวก ก

โปรแกรมคอมพิวเตอร์สำหรับใช้ในการคำนวณ

สำหรับโปรแกรมคอมพิวเตอร์ที่ใช้ในการคำนวณหาการกระจายตัวของอุณหภูมิภายในทรงกลมและค่าฟลักซ์ความร้อนเทียบกับเวลา นั้นถูกเขียนด้วยโปรแกรมคอมพิวเตอร์ภาษาซี ซึ่งประกอบไปด้วยรายละเอียด Source code ดังต่อไปนี้

```
#include <stdio.h>
#include <math.h>
#define PI 4.0*atan(1.0)
#define MXPOI 20
#define max 100
void TEMPSURFACE(int h,double dc,
double T0,double T45,double T90,
double T135,double TS[max]);
void HEAT(int n,int h,double dt,
double k1,double dr,double dc,
double T[max][max],double dT0[max][max]);
void main()
{
   int i,j,k,h,n,s;
   float t[300],T0[300],T45[300],T90[300],
T135[300];
   double R0,dr,dc,dt,k1,tr,TC,ks,TS1[max],
TS2[max],dT1[max][max], dT2[max][max];
 double T[max][max],T1[max][max],
q[max];
   FILE *inpf,*t0,*t1,*t2,*t3,*t4,*t5,*t6,*t7,
*t8,*t9;
   R0=0.0254;
  n=50;
  dr=R0/n;
  s=75;
  h=s-1;
   dc=PI/s;
   dt=0.00001;
        ks=73.0;
        k1=(7735*460)/73.0;
        TC=500.0;
   inpf = fopen("T500N.DAT","r");
   t0=fopen("T0.txt","w");
   t1=fopen("T1.txt","w");
   t2=fopen("T2.txt","w");
   t3=fopen("T3.txt","w");
   t4=fopen("T4.txt","w");
   t5=fopen("T5.txt","w");
   t6=fopen("T6.txt","w");
   t7=fopen("T7.txt","w");
   t8=fopen("T8.txt","w");
   t9=fopen("T9.txt","w");

 for (i=1 ; i<=203; i++){
  fscanf(inpf,"%f %f %f %f%f",
&t[i],&T0[i],&T45[i],&T90[i],&T135[i]);
  }
        for(i = 1;i < n;i++) {
        for(j = 1;j <= h;j++){
        T[i][j] = TC;
                }
```

```
    }
    tr=0.0;
   do{
    tr=tr+dt;
   k=0;
   do{
    k++;
    }while(tr>t[k]);

    TEMPSURFACE(h,dc,T0[k-1],T45[k-1],
T90[k-1],T135[k-1],TS1);
    TEMPSURFACE(h,dc,T0[k],T45[k],T90[k],
T135[k],TS2);
   for(j=1;j<=h;j++){
   T[n][j]=(((tr-t[k-1])/(t[k]-t[k-1]))*(TS2[j]-
TS1[j]))+TS1[j];
   }
    for(i = 0;i <= n;i++){
         for(j = 0;j <= h;j++){
     T1[i][j]=0.0;
     dT1[i][j]=0.0;
     dT2[i][j]=0.0;
     q[j]=0.0;
    }
    }
    HEAT(n,h,dt,k1,dr,dc,T,dT1);
    for(i = 0;i <= n;i++){
         for(j = 1;j <= h;j++){
         T1[i][j]=T[i][j]+ dT1[i][j];
         }
    }
    HEAT(n,h,dt,k1,dr,dc,T1,dT2);
    for(i = 1;i < n;i++){
         for(j = 1;j <= h;j++){
         T[i][j]=T[i][j]+ (dT1[i][j]+dT2[i][j])/2.0;
         }
    }

if (tr <0.00003){
for(j = 1;j <= h;j++){
q[j] = -ks*(T[n][j]-T[n-1][j])/dr;
}
  fprintf(t0,"%.6f \n",tr);
     for(j=h;j>=1;j--){
    for(i=1;i<=n;i++){
    fprintf(t0," %9.2f ",T[i][j]);
    }
    fprintf(t0," %9.2f \n",q[j]);
    }
}
else if(tr >0.09998 && tr <0.10002 ){
for(j = 1;j <= h;j++){
      q[j] = -ks*(T[n][j]-T[n-1][j])/dr;
}
  fprintf(t1,"%.6f \n",tr);
   for(j=h;j>=1;j--){
   for(i=1;i<=n;i++){
    fprintf(t1," %9.2f ",T[i][j]);
    }
    fprintf(t1," %9.2f \n",q[j]);
    }
}
else if(tr >0.19998 && tr <0.20002 ){
for(j = 1;j <= h;j++){
      q[j] = -ks*(T[n][j]-T[n-1][j])/dr;
}
  fprintf(t2,"%.6f \n",tr);
   for(j=h;j>=1;j--){
   for(i=1;i<=n;i++){
    fprintf(t2," %9.2f ",T[i][j]);
```

```
    }
    fprintf(t2," %9.2f \n",q[j]);
    }
  }
  else if(tr >0.29998 && tr <0.30002 ){
  for(j = 1;j <= h;j++){
      q[j] = -ks*(T[n][j]-T[n-1][j])/dr;
  }
   fprintf(t3,"%.6f \n",tr);
    for(j=h;j>=1;j--){
     for(i=1;i<=n;i++){
      fprintf(t3," %9.2f ",T[i][j]);
     }
     fprintf(t3," %9.2f \n",q[j]);
     }
  }
  else if(tr >0.39998 && tr <0.40002 ){
  for(j = 1;j <= h;j++){
      q[j] = -ks*(T[n][j]-T[n-1][j])/dr;
   }
   fprintf(t4,"%.6f \n",tr);
for(j=h;j>=1;j--){
     for(i=1;i<=n;i++){
      fprintf(t4," %9.2f ",T[i][j]);
     }
     fprintf(t4," %9.2f \n",q[j]);
     }
  }
  else if(tr >0.49998 && tr <0.50002 ){
  for(j = 1;j <= h;j++){
      q[j] = -ks*(T[n][j]-T[n-1][j])/dr;
  }
   fprintf(t5,"%.6f \n",tr);
for(j=h;j>=1;j--){
     for(i=1;i<=n;i++){
      fprintf(t5," %9.2f ",T[i][j]);
     }
     fprintf(t5," %9.2f \n",q[j]);
     }
  }
  else if(tr >0.59998 && tr <0.60002 ){
  for(j = 1;j <= h;j++){
      q[j] = -ks*(T[n][j]-T[n-1][j])/dr;
  }
    fprintf(t6,"%.6f \n",tr);
for(j=h;j>=1;j--){
     for(i=1;i<=n;i++){
      fprintf(t6," %9.2f ",T[i][j]);
     }
     fprintf(t6," %9.2f \n",q[j]);
     }
  }
  else if(tr >0.69998 && tr <0.70002 ){
  for(j = 1;j <= h;j++){
      q[j] = -ks*(T[n][j]-T[n-1][j])/dr;
  }
   fprintf(t7,"%.6f \n",tr);
for(j=h;j>=1;j--){
     for(i=1;i<=n;i++){
      fprintf(t7," %9.2f ",T[i][j]);
     }
     fprintf(t7," %9.2f \n",q[j]);
     }
  }
  else if(tr >0.79998 && tr <0.80002 ){
  for(j = 1;j <= h;j++){
      q[j] = -ks*(T[n][j]-T[n-1][j])/dr;
  }
  fprintf(t8,"%.6f \n",tr);
for(j=h;j>=1;j--){
```

```
        for(i=1;i<=n;i++){
         fprintf(t8," %9.2f ",T[i][j]);
         }
         fprintf(t8," %9.2f \n",q[j]);
         }
     }
     else if(tr >0.89998 ){
     for(j = 1;j <= h;j++){
          q[j] = -ks*(T[n][j]-T[n-1][j])/dr;
}
       fprintf(t9,"%.6f \n",tr);
   for(j=h;j>=1;j--){
        for(i=1;i<=n;i++){
         fprintf(t9," %9.2f ",T[i][j]);
         }
         fprintf(t9," %9.2f \n",q[j]);
         }
     }
     }while(tr <= 0.9);
     fclose(t0);
     fclose(t1);
     fclose(t2);
     fclose(t3);
     fclose(t4);
     fclose(t5);
     fclose(t6);
     fclose(t7);
     fclose(t8);
     fclose(t9);
}
void TEMPSURFACE(int h,double dc,
double T0,double T45,double T90,double
T135,double TS[max])
{
   void GAUSS(int N,
 double A[MXPOI][MXPOI],
double B[MXPOI],double X[MXPOI]);
  int i,j,Eq;
  double X0,X45,X90,X135,X180;
  double A[MXPOI][MXPOI],B[MXPOI],
AS[MXPOI];
X0=0.0;
X45=PI/4.0;
X90=PI/2.0;
X135=3*PI/4.0;
X180=PI;
for(i=1;i<=18;i++){
for(j=1;j<=18;j++){
 A[i][j]=0.0;
 B[i]=0.0;
}
}
A[1][1]=pow(X0,3);
A[1][2]=pow(X0,2);
A[1][3]=X0;
A[1][4]=1.0;
A[2][1]=pow(X45,3);
A[2][2]=pow(X45,2);
A[2][3]=X45;
A[2][4]=1.0;
A[3][5]=pow(X45,3);
A[3][6]=pow(X45,2);
A[3][7]=X45;
A[3][8]=1.0;
A[4][5]=pow(X90,3);
A[4][6]=pow(X90,2);
A[4][7]=X90;
A[4][8]=1.0;
A[5][9]=pow(X90,3);
A[5][10]=pow(X90,2);
```

```
A[5][11]=X90;
A[5][12]=1.0;
A[6][9]=pow(X135,3);
A[6][10]=pow(X135,2);
A[6][11]=X135;
A[6][12]=1.0;
A[7][13]=pow(X135,3);
A[7][14]=pow(X135,2);
A[7][15]=X135;
A[7][16]=1.0;
A[8][1]=3.0*pow(X45,2);
A[8][2]=2.0*X45;
A[8][3]=1.0;
A[8][5]=-3.0*pow(X45,2);
A[8][6]=-2*X45;
A[8][7]=-1.0;
A[9][5]=3.0*pow(X90,2);
A[9][6]=2.0*X90;
A[9][7]=1.0;
A[9][9]=-3.0*pow(X90,2);
A[9][10]=-2.0*X90;
A[9][11]=-1.0;
A[10][9]=3.0*pow(X135,2);
A[10][10]=2.0*X135;
A[10][11]=1.0;
A[10][13]=-3.0*pow(X135,2);
A[10][14]=-2*X135;
A[10][15]=-1;
A[11][1]=6.0*X45;
A[11][2]=2.0;
A[11][5]=-6.0*X45;
A[11][6]=-2.0;
A[12][5]=6.0*X90;
A[12][6]=2.0;
A[12][9]=-6.0*X90;
A[12][10]=-2.0;
A[13][9]=6.0*X135;
A[13][10]=2.0;
A[13][13]=-6.0*X135;
A[13][14]=-2.0;
A[14][1]=3.0*pow(X0,2);
A[14][2]=2.0*X0;
A[14][3]=1.0;
A[15][13]=3.0*pow(X180,2);
A[15][14]=2.0*X180;
A[15][15]=1.0;
A[16][13]=6.0*X180;
A[16][14]=2.0;

B[1]=T0;
B[2]=T45;
B[3]=T45;
B[4]=T90;
B[5]=T90;
B[6]=T135;
B[7]=T135;
 Eq=16;
GAUSS(Eq,A,B,AS);
for(j=1;j<=h;j++){
if(dc*j < PI/4){
TS[j]=(AS[1]*pow(dc*j,3))+(AS[2]*pow(dc*j,2)
)+(AS[3]*dc*j)+AS[4];
}
else if(dc*j < PI/2){
TS[j]=(AS[5]*pow(dc*j,3))+(AS[6]*pow(dc*j,2)
)+(AS[7]*dc*j)+AS[8];
}
else if(dc*j < 3*PI/4){
TS[j]=(AS[9]*pow(dc*j,3))+(AS[10]*
pow(dc*j,2))+(AS[11]*dc*j)+AS[12];
```

```
}
else {
TS[j]=(AS[13]*pow(dc*j,3))+(AS[14]*
pow(dc*j,2))+(AS[15]*dc*j)+AS[16];
}
}
}

void HEAT(int n,int h,double dt,double k1,
double dr,double dc,double T[max][max],
 double dT0[max][max])
    {
     int i,j,s;
     double a,b,c,d,e,f;
    s=h+1;
     for (i=1;i<n;i++){
  for(j=1;j<=h;j++){
  if ( i == 1 && j == 1){
  a=(dt/k1)/((i*dr)*(i*dr));
  b=(i+0.5)*(i+0.5)*(T[i+1][j]-T[i][j]);
   c=(i-0.5)*(i-0.5)*(T[i][j]-T[i][s-j])/2.0;
   d=(dt/k1)/((i*dr)*(i*dr)*sin(dc*j));
   e=sin(dc*(j+0.5))*(T[i][j+1]-T[i][j])/(dc*dc);
   f=0.0;
   }
  else if ( i == 1 && j == h ){
   a=(dt/k1)/((i*dr)*(i*dr));
   b=(i+0.5)*(i+0.5)*(T[i+1][j]-T[i][j]);
   c=(i-0.5)*(i-0.5)*(T[i][j]-T[i][s-j])/2.0;
   d=(dt/k1)/((i*dr)*(i*dr)*sin(dc*j));
   e=0.0;
   f=sin(dc*(j-0.5))*(T[i][j]-T[i][j-1])/(dc*dc);
   }
  else if ( i == 1 ){
   a=(dt/k1)/((i*dr)*(i*dr));
   b=(i+0.5)*(i+0.5)*(T[i+1][j]-T[i][j]);
   c=(i-0.5)*(i-0.5)*(T[i][j]-T[i][sj])/2.0;
    d=(dt/k1)/((i*dr)*(i*dr)*sin(dc*j));
    e=sin(dc*(j+0.5))*(T[i][j+1]-T[i][j])/(dc*dc);
   f=sin(dc*(j-0.5))*(T[i][j]-T[i][j-1])/(dc*dc);
    }
    else if ( j == 1 ){
   a=(dt/k1)/((i*dr)*(i*dr));
   b=(i+0.5)*(i+0.5)*(T[i+1][j]-T[i][j]);
   c=(i-0.5)*(i-0.5)*(T[i][j]-T[i-1][j]);
   d=(dt/k1)/((i*dr)*(i*dr)*sin(dc*j));
   e=sin(dc*(j+0.5))*(T[i][j+1]-T[i][j])/(dc*dc);
   f=0.0;
   }
  else if ( j == h ){
   a=(dt/k1)/((i*dr)*(i*dr));
   b=(i+0.5)*(i+0.5)*(T[i+1][j]-T[i][j]);
   c=(i-0.5)*(i-0.5)*(T[i][j]-T[i-1][j]);
   d=(dt/k1)/((i*dr)*(i*dr)*sin(dc*j));
   e=0.0;
   f=sin(dc*(j-0.5))*(T[i][j]-T[i][j-1])/(dc*dc);
    }
      else{
   a=(dt/k1)/((i*dr)*(i*dr));
   b=(i+0.5)*(i+0.5)*(T[i+1][j]-T[i][j]);
   c=(i-0.5)*(i-0.5)*(T[i][j]-T[i-1][j]);
   d=(dt/k1)/((i*dr)*(i*dr)*sin(dc*j));
   e=sin(dc*(j+0.5))*(T[i][j+1]-T[i][j])/(dc*dc);
   f=sin(dc*(j-0.5))*(T[i][j]-T[i][j-1])/(dc*dc);
  }
      dT0[i][j] =((a*(b-c))+(d*(e-f)));
          }
          }
}
```

```
 void GAUSS(int N,double A[][MXPOI],
double B[],double X[])
{
  int IC,IE,IP;
  void SCALE(int N,
double A[MXPOI][MXPOI],double B[MXPOI]);
  void PIVOT(int N,double A[MXPOI][MXPOI],
double B[MXPOI],int IP);
    double RATIO,SUM;
/*....PERFORM SCALING:                    */
    SCALE(N,A,B);
/*....FORWARD ELIMINATION: PERFORM
ACCORDING TO                            */
/*....THE ORDER OF 'PRIME' FROM 1 TO N-1:
                                        */
    for (IP = 1; IP <= N-1; IP++) {
/*....PERFORM PARTIAL PIVOTING            */
     PIVOT(N,A,B,IP);
/*....LOOP OVER EACH EQUQTION
STARTING FROM THE ONE THAT              */
/*....CORRESPONDS WITH THE ORDER OF
'PRIME' PLUS ONE:                       */
     for (IE = IP+1; IE <= N; IE++) {
      RATIO = A[IE][IP]/A[IP][IP];
/*....COMPUTE NEW COEFF. OF THE EQ.
CONSIDERED:                             */
       for (IC = IP+1; IC <= N; IC++)
        A[IE][IC] = A[IE][IC] - RATIO*A[IP][IC];
       B[IE] = B[IE] - RATIO*B[IP];
    }
/*....SET COEFF. ON LOWER LEFT PORTION
TO ZERO:                                */
     for (IE = IP+1; IE <= N; IE++)
      A[IE][IC] = 0.;
    }
/*....BACK SUBSTITUTION                   */
/*....COMPUTE SOLUTION OF THE LAST
EQUATION:                               */
    X[N] = B[N]/A[N][N];
/*....THEN COMPUTE SOLUTION FROM
EQUATION N-1 TO 1                       */
    for (IE = N-1; IE >= 1; IE--) {
     SUM = 0.;
     for (IC = IE+1; IC <= N; IC++)
      SUM = SUM + A[IE][IC]*X[IC];
     X[IE] = (B[IE] - SUM)/A[IE][IE];
    }
}

void PIVOT(int N,double A[][MXPOI],double
B[], int IP)
{
    int I,J,JP;
    double AMAX,BIG,DUMY;
/*...PERFORM PARTIAL PIVOTING:            */
          JP = IP;
   BIG = fabs(A[IP][IP]);
   for (I = IP+1; I <= N; I++) {
    AMAX = fabs(A[I][IP]);
    if (AMAX > BIG) {
     BIG = AMAX;
     JP = I;
    }
   }
   if (JP != IP) {
    for (J = IP; J <= N; J++) {
     DUMY = A[JP][J];
     A[JP][J] = A[IP][J];
     A[IP][J] = DUMY;
    }
```

```
    DUMY = B[JP];
    B[JP] = B[IP];
    B[IP] = DUMY;
   }
}

void SCALE(int N,double A[][MXPOI],
double B[])
{
     int IC,IE;
    double AMAX,BIG;
/*....PERFORM SCALING                    */
                                          for (IE = 1; IE <= N; IE++) {
                                           BIG = fabs(A[IE][1]);
                                          for (IC = 2; IC <= N; IC++) {
                                            AMAX = fabs(A[IE][IC]);
                                            if (AMAX > BIG) BIG = AMAX;
                                          }
                                          for (IC = 1; IC <= N; IC++)
                                            A[IE][IC] = A[IE][IC]/BIG;
                                          B[IE] = B[IE]/BIG;
                                          }
                                      }
```

ภาคผนวก ข

ตัวอย่าง การคำนวณหา Nu ของการเดือดเป็นชั้นฟิล์มบนผิวทรงกลม

ฟังก์ชันที่ใช้บรรยายสัมประสิทธิ์การพาความร้อนโดยบังคับแบบไม่ขึ้นกับเวลามักจะนิยมเขียนในรูปของตัวเลขไร้มิติ ซึ่งมีรูปทั่วไป คือ

$$Nu = C\,\mathrm{Re}^{n}\,\mathrm{Pr}^{m}$$

โดย C, nและ m เป็นค่าคงที่

ตัวอย่างของสมการในรูปแบบต่างๆ

สมการของ Nu สำหรับการไหลผ่านท่อของ ซีเดอร์-เทต (Sieder- Tate Equation) [5]

ในกรณีการไหลแบบลามินาร์

$$Nu = 1.86(\mathrm{Re}\,\mathrm{Pr})^{1/3}\left(\frac{D}{L}\right)^{1/3}\left(\frac{\mu_m}{\mu_s}\right)^{0.14} \qquad (1)$$

โดยมีเงื่อนไขดังนี้

$$0.5 < \mathrm{Pr} < 17000$$

$$0.0044 < \frac{\mu_m}{\mu_s} < 9.80$$

เมื่อ μ_m = ความหนืดของของไหลที่อุณหภูมิเฉลี่ยของของไหล

μ_s = ความหนืดของของไหลที่อุณหภูมิของผิวท่อ

ในกรณีการไหลแบบเทอร์บิวเลนท์

$$Nu = 0.027\,\mathrm{Re}^{0.8}\,\mathrm{Pr}^{1/3}\left(\frac{\mu_m}{\mu_s}\right)^{0.14} \qquad (2)$$

โดยมีเงื่อนไขดังนี้

$$0.7 \le \mathrm{Pr} \le 16700$$

$$\mathrm{Re} \ge 10^{4}$$

$$L/D > 60 \text{ (ท่อผิวเรียบ)}$$

สมการของ Nu สำหรับการไหลผ่านวัตถุรูปทรงกลมสำหรับก๊าซและของเหลวของวิตเทกเกอร์ (Whitaker) [5]

$$Nu = 2 + \left(0.4\,\mathrm{Re}^{1/2} + 0.06\,\mathrm{Re}^{2/3}\right)\mathrm{Pr}^{0.4}\left(\frac{\mu_\infty}{\mu_s}\right)^{1/4} \quad (3)$$

โดยมีเงื่อนไขดังนี้

$$3.5 < \mathrm{Re} < 80{,}000$$

$$0.7 < \mathrm{Pr} < 380$$

$$1 < \frac{\mu_\infty}{\mu_s} < 3.2$$

เมื่อ μ_∞ = ความหนืดของของไหลที่อุณหภูมิของของไหล

μ_s = ความหนืดของของไหลที่อุณหภูมิของผิวทรงกลม

สำหรับผลดังแสดงในรูปที่ 5.48 จะพิจารณา Nu ว่าเขียนได้ในรูป $Nu = \frac{C\mathrm{Re}^n}{\mathrm{Pr}^m}$ ซึ่งจะทำให้เขียนได้ว่า

$$Nu = \frac{0.0043\mathrm{Re}^n}{\mathrm{Pr}^m} \quad (4)$$

โดยที่ n และ m มีค่าเป็น1.7 และ 5 ตามลำดับ

คุณสมบัติของน้ำที่อุณหภูมิ = 27 °C

$\rho = 997.01$ kg/m^3, $\mu = 8.55 \times 10^{-4}$ Ns/m^2, Pr = 5.83

ค่า Re สำหรับของไหลไหลผ่านทรงกลมหาได้จาก

$$\mathrm{Re} = \frac{\rho v D}{\mu} \quad (5)$$

ซึ่งค่าRe นั้นสามารถเขียนแทนฟังก์ชันเวลาได้เนื่องจาก ความเร็วขึ้นกับเวลาดังสมการ

$$v = u + at \quad (6)$$

กำหนดให้ความเร็วเริ่มต้น(u) มีค่าเท่ากับ 0 m/s และความเร่ง (a) มีค่าเท่ากับ 9.81 m/s^2 จะได้

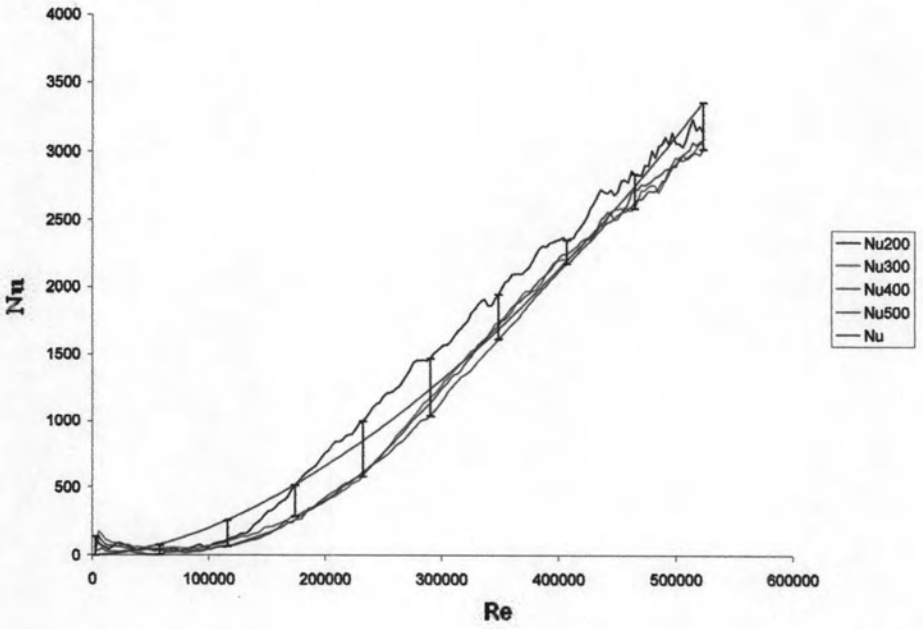
$$v = 9.81t \quad (7)$$

เมื่อแทนค่าv ลงในสมการที่ 7 จะได้

$$\mathrm{Re} = \frac{9.81\rho D t}{\mu} \quad (8)$$

จากการเปรียบเทียบค่านัสเซลท์นัมเบอร์ที่ได้จากการคำนวณตามสมการที่ 4 กับการคำนวณค่านัสเซลท์นัมเบอร์จากผลการทดลองจะพบว่ามีค่าความคลาดเคลื่อนสัมบูรณ์ทางด้านบนสูงสุด=326.28 และความคลาดเคลื่อนสัมบูรณ์ทางด้านล่างสูงสุด=349.05

การหาความคลาดเคลื่อนนั้นเนื่องจากในช่วงที่ค่าเรย์โนลด์นัมเบอร์ตั้งแต่0 ถึง 20000 มีค่านัสเซลท์นัมเบอร์ที่คำนวณได้มีค่าน้อยทำให้ความคลาดเคลื่อนสัมพัทธ์มีค่าสูงดังนั้นในช่วงดังกล่าวจึงหาในรูปของความคลาดเคลื่อนสัมบูรณ์ ซึ่งมีอยู่ระหว่าง 0 ถึง 263.70 ส่วนในช่วงที่ค่าเรย์โนลด์นัมเบอร์มากกว่า 20000 จะหาค่าความคลาดเคลื่อนในรูปของความคลาดเคลื่อนสัมพัทธ์ซึ่งมีอยู่ระหว่าง 0% ถึง 40%



ผลการเปรียบเทียบค่านัสเซลท์นัมเบอร์ที่ได้จากฟังก์ชันกับที่ได้จากผลการทดลอง

ประวัติผู้เขียนวิทยานิพนธ์

นายเอกวิทย์ พิจิตรศิริ เกิดเมื่อวันที่ 2 พฤศจิกายน พ.ศ. 2520 ที่กรุงเทพมหานคร จบการศึกษาระดับวิศวกรรมศาสตรบัณฑิต สาขาวิศวกรรมโยธา คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าพระนครเหนือ เมื่อปี พ.ศ. 2544 เข้าศึกษาต่อระดับวิศวกรรมศาสตรมหาบัณฑิต สาขาวิชานิวเคลียร์เทคโนโลยี จุฬาลงกรณ์มหาวิทยาลัยในปีการศึกษา 2546