Universidad de Valladolid
Facultad de Ciencias

ANEXO

Trabajo Fin de Grado

Grado en Física

Modelado atomístico de las implantaciones de alta energía en silicio

Autor:

Irene Muñoz Velasco

Tutores:

Lourdes Pelaz Montes

Iván Santos Tejido

```c
/*-------------------------------------------------------------------------------*/
/*                                                                               */
/*     Programa que representa las perdidas inelasticas locales L y              */
/*     no locales N en funcion de la energia instantanea del ion                 */
/*                                                                               */
/*     EJECUCION: ./PerdQ_Eactual_LN.x EINICIAL NEFF NIONES                      */
/*                                                                               */
/*     ENTRADAS: Energia_Depositada.dat                                          */
/*                                                                               */
/*     SALIDAS: PerdQ_Eactual_L_EINICAL.txt                                      */
/*              PerdQ_Eactual_L_neg_EINICAL.txt                                  */
/*              PerdQ_Eactual_N_EINICAL.txt                                      */
/*              PerdQ_Eactual_N_neg_EINICAL.txt                                  */
/*                                                                               */
/*-------------------------------------------------------------------------------*/

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

int iaux, nion, NIONES;
double EINICIAL, NEFF, PMAX;
double x, y, z, Eaux, Ener, EnerQ, Eactual;
char tipo, repre[100], repreneg[100];

main(argc, argv)
int argc;
char *argv[];
{
    FILE *entrada, *salidaL, *salidaLneg, *salidaN, *salidaNneg;

    // PARAMETROS DE ENTRADA
    if(argc>4)
    {
      printf("Más de 3 argumentos de entrada\n");
      printf("USO: ./PerdQ_Eactual_LN.x EINICIAL NEFF NIONES\n");
      exit(0);
    }
    else if(argc==4)
    {
      EINICIAL=atof(argv[1]);
      NEFF=atof(argv[2]);
      NIONES=atof(argv[3]);
      printf("La energia inicial es EINICIAL = %lf\n",EINICIAL);
      printf("El valor NEffective es NEFF = %lf\n",NEFF);
      printf("El numero de iones es NIONES = %d\n",NIONES);
    }
    else
    {
      printf("No hay argumentos suficientes\n");
      printf("USO: ./PerdQ_Eactual_LN.x EINICIAL NEFF NIONES\n");
      exit(0);
    }

    // INICIALIZACIONES
    Eactual = EINICIAL;
    nion = 1;

    // LECTURA DE DATOS
    entrada = fopen("Energia_Depositada.dat","r");
    sprintf(repre,"PerdQ_Eactual_L_%lf_mal_2.txt", EINICIAL);
    salidaL=fopen(repre,"w");
    sprintf(repreneg,"PerdQ_Eactual_L_neg_%lf_mal_2.txt", EINICIAL);
    salidaLneg=fopen(repreneg,"w");
    sprintf(repre,"PerdQ_Eactual_N_%lf_mal_2.txt", EINICIAL);
    salidaN=fopen(repre,"w");
    sprintf(repreneg,"PerdQ_Eactual_N_neg_%lf_mal_2.txt", EINICIAL);
    salidaNneg=fopen(repreneg,"w");
    while( (fscanf(entrada, "%d %le %le %le %le %c\n", &iaux, &x, &y, &z, &Eaux, &
    tipo))!=EOF){
        // Tipos de eventos
        // - R -> 0 (y 6): Reminder energy
```

```
73          // - F -> 1 (y 7): Free flight
74          // - E -> 2 (y 6): Nuclear transferred energy to an atom that does not
            become recoil
75          // - Q -> 3 (y 7): Inelastic energy (local L + nonlocal N)
76          // - L -> 4 : Local inelastic energy
77          // - N -> 5 : Nonlocal inelastic energy
78          // - Total Elastico = R + I -> 6
79          // - Total Inelastico = F + Q -> 7
80
81          // Correcion de la energia multiplicando por el numero de iones y dividiendo
            por Neffective
82          Ener=Eaux*NIONES/NEFF;
83
84          if(iaux > NIONES)
85          {
86              printf("El numero de iones en Energia_Depositada.dat es mayor que el
                parametro NIONES.\n");
87              exit(0);
88          }
89
90          if (iaux == nion) // Misma cascada
91          {
92              if (tipo == 'Q')
93              {
94                  Eactual = Eactual - Ener;
95              }
96              else if (tipo == 'L')
97              {
98                  if (Eactual < 0.0)
99                  {
100                     fprintf(salidaLneg,"%d\t%lf\t%lf\t%lf\t%lf\t%lf\n", nion, x, y, z
                        , Eactual, Ener);
101                 }
102                 else
103                 {
104                     fprintf(salidaL,"%lf\t%lf\n", Eactual, Ener);
105                 }
106             }
107             else if (tipo == 'N')
108             {
109                 if (Eactual < 0.0)
110                 {
111                     fprintf(salidaNneg,"%d\t%lf\t%lf\t%lf\t%lf\t%lf\n", nion, x, y, z
                        , Eactual, Ener);
112                 }
113                 else
114                 {
115                     fprintf(salidaN,"%lf\t%lf\n", Eactual, Ener);
116                 }
117             }
118             else
119                 Eactual = Eactual - Ener;
120         }
121         else
122         {
123             Eactual = EINICIAL;
124             nion ++;
125             if (tipo == 'Q')
126             {
127                 Eactual = Eactual - Ener;
128             }
129             else if (tipo == 'L')
130             {
131                 if (Eactual < 0.0)
132                 {
133                     fprintf(salidaLneg,"%d\t%lf\t%lf\t%lf\t%lf\t%lf\n", nion, x, y, z
                        , Eactual, Ener);
134                 }
135                 else
136                 {
137                     fprintf(salidaL,"%lf\t%lf\n", Eactual, Ener);
138                 }
139             }
```

```
140                 else if (tipo == 'N')
141                 {
142                     if (Eactual < 0.0)
143                     {
144                         fprintf(salidaNneg,"%d\t%lf\t%lf\t%lf\t%lf\t%lf\n", nion, x, y, z
                            , Eactual, Ener);
145                     }
146                     else
147                     {
148                         fprintf(salidaN,"%lf\t%lf\n", Eactual, Ener);
149                     }
150                 }
151                 else
152                     Eactual = Eactual - Ener;
153             }
154         }
155         fclose(entrada);
156         fclose(salidaL);
157         fclose(salidaLneg);
158         fclose(salidaN);
159         fclose(salidaNneg);
160     }
```

```c
/*-------------------------------------------------------------------------------*/
/*                                                                               */
/*     Programa que calcula los promedios y las desviaciones estandar de las     */
/*     perdidas debidas a los eventos L en funcion de la energia actual del ion  */
/*                                                                               */
/*     EJECUCION: ./Promedios_Desviacion_PerdQ_Eactual_QLIM_L.x EINICIAL QLIM     */
/*                                                                               */
/*     ENTRADAS: PerdQ_Eactual_L_EINICAL.txt                                     */
/*                                                                               */
/*     SALIDAS: histo_Ener_Eactual_L_EINICIAL_QLIM.txt                           */
/*                                                                               */
/*-------------------------------------------------------------------------------*/

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

// Intervalos en la energia actual del ion calculados en Scilab con logspace(-1,8,181)
double intervalos_Eactual[182]={0.0, 0.1, 0.1122018, 0.1258925, 0.1412538, 0.1584893,
 0.1778279, 0.1995262, 0.2238721, 0.2511886, 0.2818383, 0.3182278, 0.3548134,
0.3981072, 0.4466836, 0.5011872, 0.5623413, 0.6309573, 0.7079458, 0.7943282,
0.8912509, 1., 1.1220185, 1.2589254, 1.4125375, 1.5848932, 1.7782794, 1.9952623,
2.2387211, 2.5118864, 2.8183829, 3.1822777, 3.5481339, 3.9810717, 4.4668359,
5.0118723, 5.6234133, 6.3095734, 7.0794578, 7.9432823, 8.9125094, 10., 11.220185,
12.589254, 14.125375, 15.848932, 17.782794, 19.952623, 22.387211, 25.118864,
28.183829, 31.622777, 35.481339, 39.810717, 44.668359, 50.118723, 56.234133,
63.095734, 70.794578, 79.432823, 89.125094, 100., 112.20185, 125.89254, 141.25375,
158.48932, 177.82794, 199.52623, 223.87211, 251.18864, 281.83829, 316.22777,
354.81339, 398.10717, 446.68359, 501.18723, 562.34133, 630.95734, 707.94578,
794.32823, 891.25094, 1000., 1122.0185, 1258.9254, 1412.5375, 1584.8932, 1778.2794,
1995.2623, 2238.7211, 2511.8864, 2818.3829, 3182.2777, 3548.1339, 3981.0717,
4466.8359, 5011.8723, 5623.4133, 6309.5734, 7079.4578, 7943.2823, 8912.5094, 10000.,
11220.185, 12589.254, 14125.375, 15848.932, 17782.794, 19952.623, 22387.211,
25118.864, 28183.829, 31822.777, 35481.339, 39810.717, 44668.359, 50118.723,
56234.133, 63095.734, 70794.578, 79432.823, 89125.094, 100000., 112201.85, 125892.54,
 141253.75, 158489.32, 177827.94, 199526.23, 223872.11, 251188.64, 281838.29,
318227.77, 354813.39, 398107.17, 446683.59, 501187.23, 562341.33, 630957.34,
707945.78, 794328.23, 891250.94, 1000000., 1122018.5, 1258925.4, 1412537.5, 1584893.2
, 1778279.4, 1995262.3, 2238721.1, 2511886.4, 2818382.9, 3182277.7, 3548133.9,
3981071.7, 4466835.9, 5011872.3, 5623413.3, 6309573.4, 7079457.8, 7943282.3,
8912509.4, 10000000., 11220185., 12589254., 14125375., 15848932., 17782794.,
19952623., 22387211., 25118864., 28183829., 31822777., 35481339., 39810717.,
44668359., 50118723., 56234133., 63095734., 70794578., 79432823., 89125094.,
100000000.};

int dim_Eactual = 182, i , j;
int guardado;

double EINICIAL, QLIM, Eactual, Ener, aux;
double histo_cont[182], histo_Ener[182];
double histo_PromedioEner[182], histo_DesvEner[182];
char shisto[182];

main(argc, argv)
int argc;
char *argv[];
{
    FILE *entrada, *salida;

    // PARAMETROS DE ENTRADA
    if(argc>3)
    {
      printf("Mas de un argumento de entrada\n");
      printf("USO: ./Promedios_Desviacion_PerdQ_Eactual_QLIM_L.x EINICIAL QLIM\n");
      exit(0);
    }
    else if(argc==3)
    {
      EINICIAL=atof(argv[1]);
      printf("La energia inicial es EINICIAL = %lf\n",EINICIAL);
      QLIM=atof(argv[2]);
      printf("El limite inferior para las perdidas L es QLIM = %lf\n",QLIM);
```

```
49      }
50      else
51      {
52        printf("No hay argumentos suficientes\n");
53        printf("USO: ./Promedios_Desviacion_PerdQ_Eactual_QLIM_L.x EINICIAL QLIM\n");
54        exit(0);
55      }
56
57      // INICIALIZACIONES
58      for (i = 0; i< dim_Eactual-1 ; i++)
59      {
60          // Variables del histograma en funcion de Eactual
61          histo_cont[i] = 0.0; // Eventos almacenados en el intervalo i
62          histo_Ener[i] = 0.0; // Perdidas Q almacenadas en el intervalo i
63          histo_PromedioEner[i] = 0.0; // Promedio de las perdidas Q almacenadas en el
            intervalo i
64          histo_DesvEner[i] = 0.0; // Desviacion estandar de las perdidas Q
            almacenadas en el intervalo i
65      }
66
67      // LECTURA DE DATOS
68      printf("Inicio de lectura de datos...\n");
69      sprintf(shisto,"PerdQ_Eactual_L_%lf.txt", EINICIAL);
70      entrada = fopen(shisto, "r");
71
72      //CALCULO DE LOS PROMEDIOS
73      while( (fscanf(entrada, "%le %le \n", &Eactual, &Ener))!=EOF) // Ener son las
        pérdidas L
74      {
75          if (Ener >= QLIM) // Valores de Ener por encima de QLIM eV
76          {
77              guardado = 0;
78
79              // Intervalos equiespaciados de la energia actual del ion en escala
                logaritmica
80              for (i = 0; i < dim_Eactual-1 ; i++ )
81              {
82                  if ((intervalos_Eactual[i] <= Eactual) && (Eactual <
                    intervalos_Eactual[i+1]))
83                  {
84                      // Histograma en funcion de la energia actual del ion
85                      histo_cont[i] ++;
86                      histo_Ener[i] += Ener;
87                      guardado = 1;
88                  }
89              }
90              if (guardado == 0)
91              {
92                  printf("Energia %le fuera del intervalo [%le, %le]\n", Eactual,
                    intervalos_Eactual[0], intervalos_Eactual[dim_Eactual-1]);
93                  exit(0);
94              }
95          }
96      }
97      fclose(entrada);
98
99      printf("... fin\n");
100
101     // PROMEDIO EN CADA INTERVALO
102     printf("Promedios Ener...\n");
103     for (j = 0; j < dim_Eactual-1; j++)
104     {
105         histo_PromedioEner[j] = histo_Ener[j]/histo_cont[j];
106     }
107     printf("... fin\n");
108
109     // DESVIACION ESTANDAR EN CADA INTERVALO
110     printf("Desviacion estandar E_actual...\n");
111     sprintf(shisto,"PerdQ_Eactual_L_%lf.txt", EINICIAL);
112     entrada = fopen(shisto, "r");
113     while( (fscanf(entrada, "%le %le \n", &Eactual, &Ener))!=EOF) // Ener son las
        pérdidas L
114     {
```

```
115             if (Ener >= QLIM) // Valores de Ener por encima de QLIM eV
116             {
117                 guardado = 0;
118 
119                 // Intervalos equiespaciados de la energia actual del ion en escala
                    logaritmica
120                 for (i = 0; i < dim_Eactual-1 ; i++ )
121                 {
122                     if ((intervalos_Eactual[i] <= Eactual) && (Eactual <
                        intervalos_Eactual[i+1]))
123                     {
124                         // Histograma en funcion de la energia actual del
                            ion
125                         histo_DesvEner[i] += (Ener - histo_PromedioEner[i])*(Ener -
                            histo_PromedioEner[i])/histo_cont[i];
126                         guardado = 1;
127                     }
128                 }
129                 if (guardado == 0)
130                 {
131                     printf("Energia %le fuera del intervalo [%le, %le]\n", Eactual,
                        intervalos_Eactual[0], intervalos_Eactual[dim_Eactual-1]);
132                     exit(0);
133                 }
134             }
135         }
136         fclose(entrada);
137         printf("... \n");
138         for (j = 0; j < dim_Eactual-1; j++)
139         {
140             aux = sqrt(histo_DesvEner[j]);
141             histo_DesvEner[j] = aux;
142         }
143         printf("... fin\n");
144 
145         // SALIDA DE DATOS
146         // Estructura del fichero de salida
147         //      1            2               3
            4
148         // bin inicial    bin final    promedio L (eV)   desviacion L (eV)
149 
150         printf("salida de datos ...\n");
151         sprintf(shisto,"histo_Ener_Eactual_L_%lf_%lf.txt", EINICIAL,QLIM);
152         salida=fopen(shisto,"w");
153         for (j = 0; j < dim_Eactual-1; j++)
154         {
155             fprintf(salida, "%le\t%le\t%le\t%le\n", intervalos_Eactual[j],
                intervalos_Eactual[j+1], histo_PromedioEner[j], histo_DesvEner[j]);
156         }
157         fclose(salida);
158 
159         printf("... fin\n");
160     }
161 
162 
163 
164 
165 
166 
167 
168 
169 
170 
```

```c
/*-------------------------------------------------------------------------------*/
/*                                                                               */
/*     Programa que calcula los promedios y las desviaciones estandar de las     */
/*     perdidas debidas a los eventos N en funcion de la energia actual del ion  */
/*                                                                               */
/*     EJECUCION: ./Promedios_Desviacion_PerdQ_Eactual_QLIM_N.x EINICIAL QLIM     */
/*                                                                               */
/*     ENTRADAS: PerdQ_Eactual_N_EINICAL.txt                                     */
/*                                                                               */
/*     SALIDAS: histo_Ener_Eactual_N_EINICIAL_QLIM.txt                           */
/*                                                                               */
/*-------------------------------------------------------------------------------*/

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

// Intervalos en la energia actual del ion calculados en Scilab con logspace(-1,8,181)
double intervalos_Eactual[182]={0.0, 0.1, 0.1122018, 0.1258925, 0.1412538, 0.1584893,
 0.1778279, 0.1995262, 0.2238721, 0.2511886, 0.2818383, 0.3182278, 0.3548134,
0.3981072, 0.4466836, 0.5011872, 0.5623413, 0.6309573, 0.7079458, 0.7943282,
0.8912509, 1., 1.1220185, 1.2589254, 1.4125375, 1.5848932, 1.7782794, 1.9952623,
2.2387211, 2.5118864, 2.8183829, 3.1822777, 3.5481339, 3.9810717, 4.4668359,
5.0118723, 5.6234133, 6.3095734, 7.0794578, 7.9432823, 8.9125094, 10., 11.220185,
12.589254, 14.125375, 15.848932, 17.782794, 19.952623, 22.387211, 25.118864,
28.183829, 31.622777, 35.481339, 39.810717, 44.668359, 50.118723, 56.234133,
63.095734, 70.794578, 79.432823, 89.125094, 100., 112.20185, 125.89254, 141.25375,
158.48932, 177.82794, 199.52623, 223.87211, 251.18864, 281.83829, 316.22777,
354.81339, 398.10717, 446.68359, 501.18723, 562.34133, 630.95734, 707.94578,
794.32823, 891.25094, 1000., 1122.0185, 1258.9254, 1412.5375, 1584.8932, 1778.2794,
1995.2623, 2238.7211, 2511.8864, 2818.3829, 3182.2777, 3548.1339, 3981.0717,
4466.8359, 5011.8723, 5623.4133, 6309.5734, 7079.4578, 7943.2823, 8912.5094, 10000.,
11220.185, 12589.254, 14125.375, 15848.932, 17782.794, 19952.623, 22387.211,
25118.864, 28183.829, 31822.777, 35481.339, 39810.717, 44668.359, 50118.723,
56234.133, 63095.734, 70794.578, 79432.823, 89125.094, 100000., 112201.85, 125892.54,
 141253.75, 158489.32, 177827.94, 199526.23, 223872.11, 251188.64, 281838.29,
318227.77, 354813.39, 398107.17, 446683.59, 501187.23, 562341.33, 630957.34,
707945.78, 794328.23, 891250.94, 1000000., 1122018.5, 1258925.4, 1412537.5, 1584893.2
, 1778279.4, 1995262.3, 2238721.1, 2511886.4, 2818382.9, 3182277.7, 3548133.9,
3981071.7, 4466835.9, 5011872.3, 5623413.3, 6309573.4, 7079457.8, 7943282.3,
8912509.4, 10000000., 11220185., 12589254., 14125375., 15848932., 17782794.,
19952623., 22387211., 25118864., 28183829., 31822777., 35481339., 39810717.,
44668359., 50118723., 56234133., 63095734., 70794578., 79432823., 89125094.,
100000000.};

int dim_Eactual = 182, i , j;
int guardado;

double EINICIAL, QLIM, Eactual, Ener, aux;
double histo_cont[182], histo_Ener[182];
double histo_PromedioEner[182], histo_DesvEner[182];
char shisto[182];

main(argc, argv)
int argc;
char *argv[];
{
    FILE *entrada, *salida;

    // PARAMETROS DE ENTRADA
    if(argc>3)
    {
      printf("Mas de un argumento de entrada\n");
      printf("USO: ./Promedios_Desviacion_PerdQ_Eactual_QLIM_N.x EINICIAL QLIM\n");
      exit(0);
    }
    else if(argc==3)
    {
      EINICIAL=atof(argv[1]);
      printf("La energia inicial es EINICIAL = %lf\n",EINICIAL);
      QLIM=atof(argv[2]);
      printf("El limite inferior para las perdidas N es QLIM = %lf\n",QLIM);
```

```
49      }
50      else
51      {
52        printf("No hay argumentos suficientes\n");
53        printf("USO: ./Promedios_Desviacion_PerdQ_Eactual_QLIM_N.x EINICIAL QLIM\n");
54        exit(0);
55      }
56
57      // INICIALIZACIONES
58      for (i = 0; i< dim_Eactual-1 ; i++)
59      {
60          // Variables del histograma en funcion de Eactual
61          histo_cont[i] = 0.0; // Eventos almacenados en el intervalo i
62          histo_Ener[i] = 0.0; // Perdidas Q almacenadas en el intervalo i
63          histo_PromedioEner[i] = 0.0; // Promedio de las perdidas Q almacenadas en el
            intervalo i
64          histo_DesvEner[i] = 0.0; // Desviacion estandar de las perdidas Q
            almacenadas en el intervalo i
65      }
66
67      // LECTURA DE DATOS
68      printf("Inicio de lectura de datos...\n");
69      sprintf(shisto,"PerdQ_Eactual_N_%lf.txt", EINICIAL);
70      entrada = fopen(shisto, "r");
71
72      // CALCULO DE LOS PROMEDIOS
73      while( (fscanf(entrada, "%le %le \n", &Eactual, &Ener))!=EOF) // Ener son las
        pérdidas N
74      {
75          if (Ener >= QLIM) // Valores de Ener por encima de QLIM eV
76          {
77              guardado = 0;
78
79              // Intervalos equiespaciados de la energia actual del ion en escala
                logaritmica
80              for (i = 0; i < dim_Eactual-1 ; i++ )
81              {
82                  if ((intervalos_Eactual[i] <= Eactual) && (Eactual <
                    intervalos_Eactual[i+1]))
83                  {
84                      // Histograma en funcion de la energia actual del ion
85                      histo_cont[i] ++;
86                      histo_Ener[i] += Ener;
87                      guardado = 1;
88                  }
89              }
90              if (guardado == 0)
91              {
92                  printf("Energia %le fuera del intervalo [%le, %le]\n", Eactual,
                    intervalos_Eactual[0], intervalos_Eactual[dim_Eactual-1]);
93                  exit(0);
94              }
95          }
96      }
97      fclose(entrada);
98
99      printf("... fin\n");
100
101     // PROMEDIO EN CADA INTERVALO
102     printf("Promedios Ener...\n");
103     for (j = 0; j < dim_Eactual-1; j++)
104     {
105         histo_PromedioEner[j] = histo_Ener[j]/histo_cont[j];
106     }
107     printf("... fin\n");
108
109     // DESVIACION ESTANDAR EN CADA INTERVALO
110     printf("Desviacion estandar E_actual...\n");
111     sprintf(shisto,"PerdQ_Eactual_N_%lf_mal_2.txt", EINICIAL);
112     entrada = fopen(shisto, "r");
113     while( (fscanf(entrada, "%le %le \n", &Eactual, &Ener))!=EOF) // Ener son las
        pérdidas N
114     {
```

```
115             if (Ener >= QLIM) // Valores de Ener por encima de QLIM eV
116             {
117                 guardado = 0;
118 
119                 // Intervalos equiespaciados de la energia actual del ion en escala
                    logaritmica
120                 for (i = 0; i < dim_Eactual-1 ; i++ )
121                 {
122                     if ((intervalos_Eactual[i] <= Eactual) && (Eactual <
                        intervalos_Eactual[i+1]))
123                     {
124                         // Histograma en funcion de la energia actual del
                            ion
125                         histo_DesvEner[i] += (Ener - histo_PromedioEner[i])*(Ener -
                            histo_PromedioEner[i])/histo_cont[i];
126                         guardado = 1;
127                     }
128                 }
129                 if (guardado == 0)
130                 {
131                     printf("Energia %le fuera del intervalo [%le, %le]\n", Eactual,
                        intervalos_Eactual[0], intervalos_Eactual[dim_Eactual-1]);
132                     exit(0);
133                 }
134             }
135         }
136         fclose(entrada);
137         printf("... \n");
138         for (j = 0; j < dim_Eactual-1; j++)
139         {
140             aux = sqrt(histo_DesvEner[j]);
141             histo_DesvEner[j] = aux;
142         }
143         printf("... fin\n");
144 
145         // SALIDA DE DATOS
146         // Estructura del fichero de salida
147         //      1            2               3
            4
148         // bin inicial    bin final    promedio N (eV)   desviacion N (eV)
149 
150         printf("salida de datos ...\n");
151         sprintf(shisto,"histo_Ener_Eactual_N_%lf_%lf.txt", EINICIAL,QLIM);
152         salida=fopen(shisto,"w");
153         for (j = 0; j < dim_Eactual-1; j++)
154         {
155             fprintf(salida, "%le\t%le\t%le\t%le\n", intervalos_Eactual[j],
                intervalos_Eactual[j+1], histo_PromedioEner[j], histo_DesvEner[j]);
156         }
157         fclose(salida);
158 
159         printf("... fin\n");
160     }
161 
162 
163 
164 
165 
166 
167 
168 
169 
170 
```