Proyecto

Novoa Gastaldi Alejandro Silvestre

(silvestre.novoa@ciencias.unam.mx)

------ Sistema con deformación en un eje del plano y Dependiente de espín.

```
Clear["Global`*"]
borra

s = -1;  (*Valor Cuantizado de Espín. +1 dara una solución y -1 dara la segunda*)
Rs = +1;
(*Parametro de las dos soluciones de RSOC. +1 dara una solución y -1 dara la segunda*)


Zigzag = 10.;
Armchair = 10.;

NZigzag = 1. + 2 * Zigzag;
NArmchair = 2 * Armchair;

NEdos = 1 * NZigzag * NArmchair;

ε = 0; (*Energía de Sitio*)
εz = 0; (*Valor de Splitting Zeeman de la Energía de
 Sitio. Causado por un potencial externo. Dependiente del Espín.*)
εAB = 0.0; (*Valor de Splitting de la Energía de Sitio. Causado
 por un potencial externo. Dependiente de la Subred*)
t = 1; (*Energía de Transición (Tuneleo)*)



(*******Intrinseco*******)
λI = 1 * 0.01;
tλI = (ⅈ / (3 * Sqrt[3])) * λI * s;
              raíz cuadrada
```

```
                 raíz cuadrada

(*******Valley-Zeeman*******)
λVZ = 1 * 0.01;
tλVZ = (i / (3 * Sqrt[3])) * λVZ * s;
                  raíz cuadrada

(*******RSOC*******)
λR = 1 * 0.01;
tλR = (2 * i / 3) * λR * s;

a = 0.000000000142;  (*Distancia entre átomos*)
b = 0.000000000246; (*Distancia entre hexagonos de Grafeno por el lado zig zag*)

epsi = 1 * 0.2; (*Constante de deformación*)
theta = 0 + 2 * π / 4;  (*Angulo respecto a la deformacion*)
young = 0.165; (*Modulo de Young del Grafeno*)
def =
  epsi * {{Cos[theta - π / 2] ^2 - young * Sin[theta - π / 2] ^2, (1 + young) * Cos[theta - π / 2] *
          coseno                         seno                                   coseno
      Sin[theta - π / 2]}, {(1 + young) * Cos[theta - π / 2] * Sin[theta - π / 2],
      seno                                coseno               seno
     Sin[theta - π / 2] ^2 - young * Cos[theta - π / 2] ^2}};
     seno                            coseno
(*Tensor sociado a la deformación lineal*)

(*Vectores posición originales a primeros vecinos*)
dist10 = {0, b / Sqrt[3]};
                 raíz cuadrada
dist20 = (b / 2) * {1, -1 / Sqrt[3]};
                            raíz cuadrada
dist30 = (b / 2) * {-1, -1 / Sqrt[3]};
                             raíz cuadrada
(*Los vectores fueron asignados considerando el zigzag en la dirección x*)

(*Vectores posición Deformados*)
dist1 = ({{1, 0}, {0, 1}} + def).dist10;
dist2 = ({{1, 0}, {0, 1}} + def).dist20;
dist3 = ({{1, 0}, {0, 1}} + def).dist30;
```

```
(*Campo Magnetico en dirección z*) (*Modifica el tranporte o tuneleo entre
 dos sitios atomicos. Y modifica por efecto Zeeman la energía de sitio*)
(*Se define la fase magnetica*)
B = 0; (*Magnitud de campo magnetico en z. [Tesla]=[kg/(Couloumb*s)]*)
Qe = 1.6 * 10^(-19); (*Carga electrica del electrón. [Couloumb]*)
hbarra = 6.58 * 10^(-16); (*Cte reducida de Planck. [eV*s]*)
MasaQe = 9.1 * 10^(-31); (*Masa del electrón. [kg]*)
ϕB[x1_, y1_, x2_, y2_] := B * hbarra * (y1 - y2) * (x1 - x2) / (2 * Qe);
(*Aij.(ri-rj), donde A=(By,0,0) y Aij=(B(yi-yj)/2,0,0)*)
εz = B * hbarra * Qe / (2 * MasaQe) (*Valor de Splitting Zeeman de la Energía
 de Sitio. Causado por un potencial externo. Dependiente del Espín.*)



(*Para la fase magnetica: Se definiran las posiciones de los átomos de la red,
centrando una esquina los ejes*)
PosAux1 = Table[If[OddQ[i], {a / 2, (Mod[i, NZigzag, 1] - 1) * a * Sqrt[3] / 2},
          tabla si ¿impar?              operación módulo              raíz cuadrada
    {0, (Mod[i, NZigzag, 1] - 1) * a * Sqrt[3] / 2}], {i, 1, NZigzag}];
        operación módulo             raíz cuadrada
 (*Pocisiones de primera columna*)
PosAux2 = Table[{0, 0}, {i, 1, 2 * NZigzag}]; (*Matriz auxiliar para la segunda columna*)
          tabla
Pos = Table[{0, 0}, {i, 1, NEdos}]; (*Matriz auxiliar para toda la nanocinta*)
      tabla

Do[PosAux2[[i]] = PosAux1[[Mod[i, NZigzag, 1]]], {i, 1, 2 * NZigzag}];
repite                     operación módulo
Do[PosAux2[[i, 1]] = PosAux2[[i, 1]] + a * Mod[i - 1, 2, 1], {i, NZigzag + 1, 2 * NZigzag}];
repite                                      operación módulo
(*Se asignan las posiciones de la segunda columna*)

Do[Pos[[i]] = PosAux2[[Mod[i, 2 * NZigzag, 1]]], {i, 1, NEdos}];
repite                 operación módulo
Do[Pos[[i, 1]] = Pos[[i, 1]] + 3 * a * IntegerPart[(i - 1) / (2 * NZigzag)],
repite                                 parte entera
  {i, 2 * NZigzag + 1, NEdos}];
(*Se asignan las posiciones de todos los atomos en la red*)

desplX[x_, y_] := epsi *
   (x * (Cos[theta]^2 - young * Sin[theta]^2) + y * ((1 + young) * Cos[theta] * Sin[theta]));
         coseno              seno                                coseno        seno
desplY[x_, y_] := epsi * (y * (Sin[theta]^2 - young * Cos[theta]^2) +
                               seno                  coseno
     x * ((1 + young) * Cos[theta] * Sin[theta]));
                        coseno       seno
```

```
                                    coseno           seno
PosDef = Table[{Pos[[i, 1]] + desplX[Pos[[i, 1]], Pos[[i, 2]]],
         tabla
    Pos[[i, 2]] + desplY[Pos[[i, 1]], Pos[[i, 2]]]}, {i, 1, NEdos}];
(*Se definen las nuevas posiciones debido a la deformación*)


(*Se dan los parametros de traslape de la
 primer diagonal que consideran de la deformación*)
tDef21 = Table[
         tabla
   {t * Exp[-3.37 * (Norm[PosDef[[i]] - PosDef[[i + 1]]] / a - 1)] * Exp[ⅈ * ϕB[PosDef[[i, 1]],
      exponencial   norma                                            exponencial
       PosDef[[i, 2]], PosDef[[i + 1, 1]], PosDef[[i + 1, 2]]]]}, {i, 1, NEdos - 1}];
(*Se paso la asignación de ceros a elementos en bordes distintos despues
 de Rashba para no dividir entre cero*)

(*Se dan los parametros de traslape entre columnas que consideran de la deformación*)
tDefm1 = Table[{t * Exp[-3.37 * (Norm[PosDef[[i]] - PosDef[[i + NZigzag]]] / a - 1)] *
         tabla     exponencial     norma
     Exp[ⅈ * ϕB[PosDef[[i, 1]], PosDef[[i, 2]], PosDef[[i + NZigzag, 1]],
     exponencial
       PosDef[[i + NZigzag, 2]]]]}, {i, 1, NEdos - NZigzag}];
(*Se paso la asignación de ceros a elementos de columnas que no interactuan
 despues de Rashba para no dividir entre cero*)



(*Analisis de distancias deformadas por tensor y desplazamiento de posiciones*)
Norm[dist2] (*Tensor*)
norma
Norm[dist3]
norma
Norm[dist1]
norma
Norm[PosDef[[1]] - PosDef[[2]]]      (*Posiciones*)
norma
Norm[PosDef[[2]] - PosDef[[3]]]
norma
Norm[PosDef[[1]] - PosDef[[NZigzag + 1]]]
norma

Norm[PosDef[[1]] - PosDef[[2]]] / Norm[dist2]
norma                             norma
Norm[PosDef[[2]] - PosDef[[3]]] / Norm[dist3]
norma                             norma
Norm[PosDef[[1]] - PosDef[[NZigzag + 1]]] / Norm[dist1]
norma                                       norma
```

norma norma

0.

1.62793×10^{-10}

1.62793×10^{-10}

1.37341×10^{-10}

1.6276×10^{-10}

1.6276×10^{-10}

1.37314×10^{-10}

0.999802

0.999802

0.999802

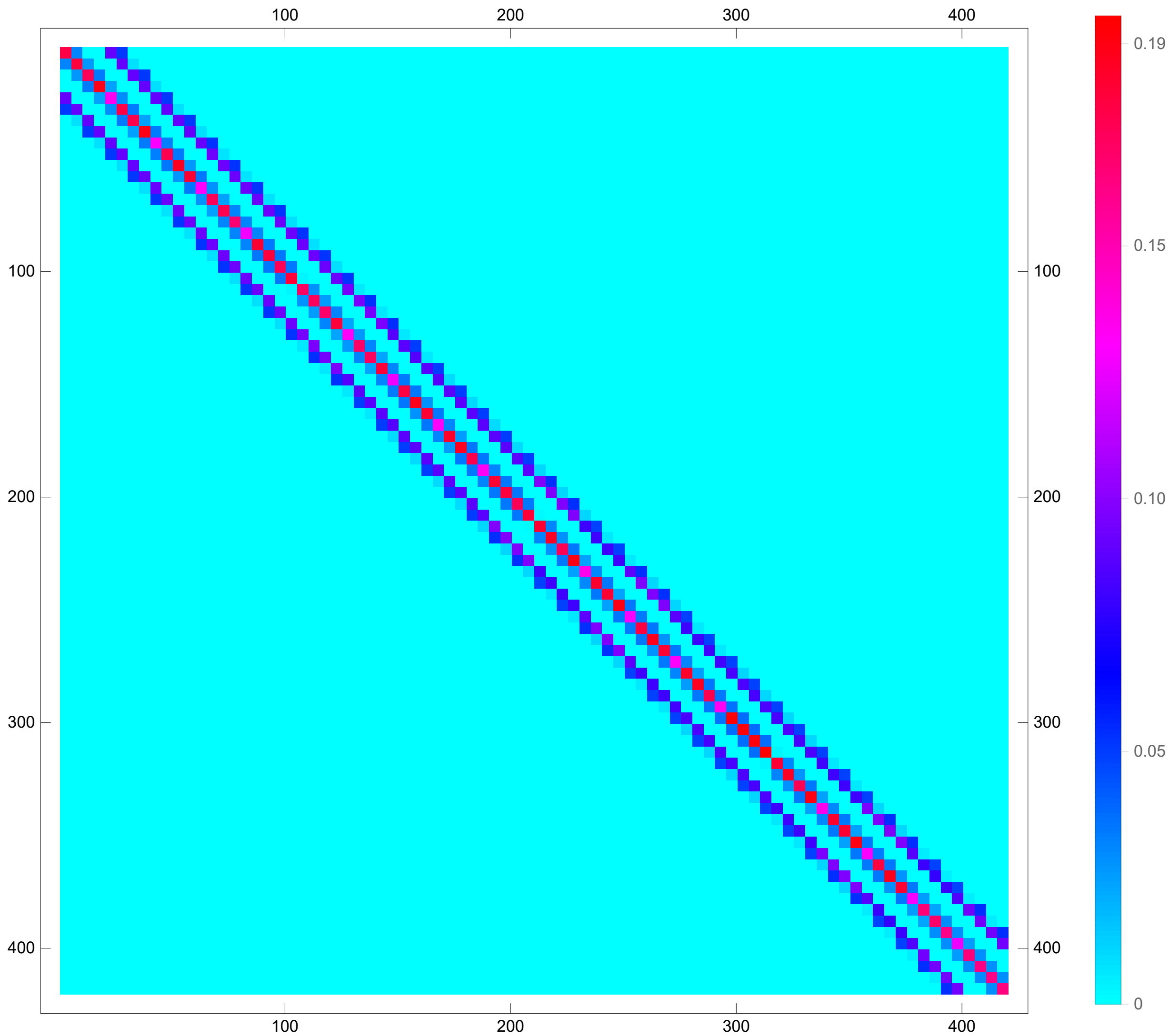
```
t1 = t * Exp[-3.37 * (Norm[dist1] / a - 1)]
         exponencial   norma
(*Energías de Transición asociadas a la deformación por el tensor*)
t2 = t * Exp[-3.37 * (Norm[dist2] / a - 1)]
         exponencial   norma
t3 = t * Exp[-3.37 * (Norm[dist3] / a - 1)]
         exponencial   norma
tDefm1[[1]]
(*Energías de Transición asociadas a la deformación por desplazamiento*)
tDef21[[1]]
tDef21[[2]]

tDefm1[[1]] / t1
tDef21[[1]] / t2
tDef21[[2]] / t3
```

```
1.11691

0.610512

0.610512

{1.11763 + 0. i}

{0.61098 + 0. i}

{0.61098 + 0. i}

{1.00065 + 0. i}

{1.00077 + 0. i}

{1.00077 + 0. i}
```

```
H1 = SparseArray[{Band[{1, 1}] → ε + s * εz}, {NEdos, NEdos}];
       array disperso   banda
(*Diagonal: Energías de Sitio*)
   diagonal
Do[H1[[i + 1, i]] = tDef21[[i, 1]] + Rs * tλR, {i, 1, NEdos - 1}];
repite
(*Primera Diagonal Superior*)
           diagonal
Do[H1[[i + NZigzag, i]] = tDefm1[[i, 1]] + Rs * tλR, {i, 1, NEdos - NZigzag}];
repite
(*Diagonal Superior de interacción entre columnas*)
   diagonal

(*Hamiltoniano de interacciones del grafeno*)
HEdoBase = H1 + ConjugateTranspose[H1];
                transpuesto conjugado


H2 = SparseArray[{Band[{1, 1}] → 0.}, {NEdos, NEdos}];
       array disperso   banda
Do[H2[[i + 1, i]] = tλI, {i, 1, NEdos - 1}]; (*Primera Diagonal Superior*)
repite                                                diagonal
Do[H2[[i + NZigzag, i]] = tλI, {i, 1, NEdos - NZigzag}];
repite
(*Diagonal Superior de interacción entre columnas*)
   diagonal
HEdoBase = HEdoBase + H2 - ConjugateTranspose[H2];
                           transpuesto conjugado

HEdoBase // MatrixForm
               forma de matriz
MatrixPlot[ Re[HEdoBase] , PlotTheme → "Detailed", ImageSize → Large, ColorFunction → Hue]
representació··· parte real  tema de representación  tamaño de i···  grande  función de color  tonalic
```

(⋯ 1 ⋯)

salida grande	Mostrar menos	Mostrar más	Mostrar salida completa	Establecer límite de tamaño



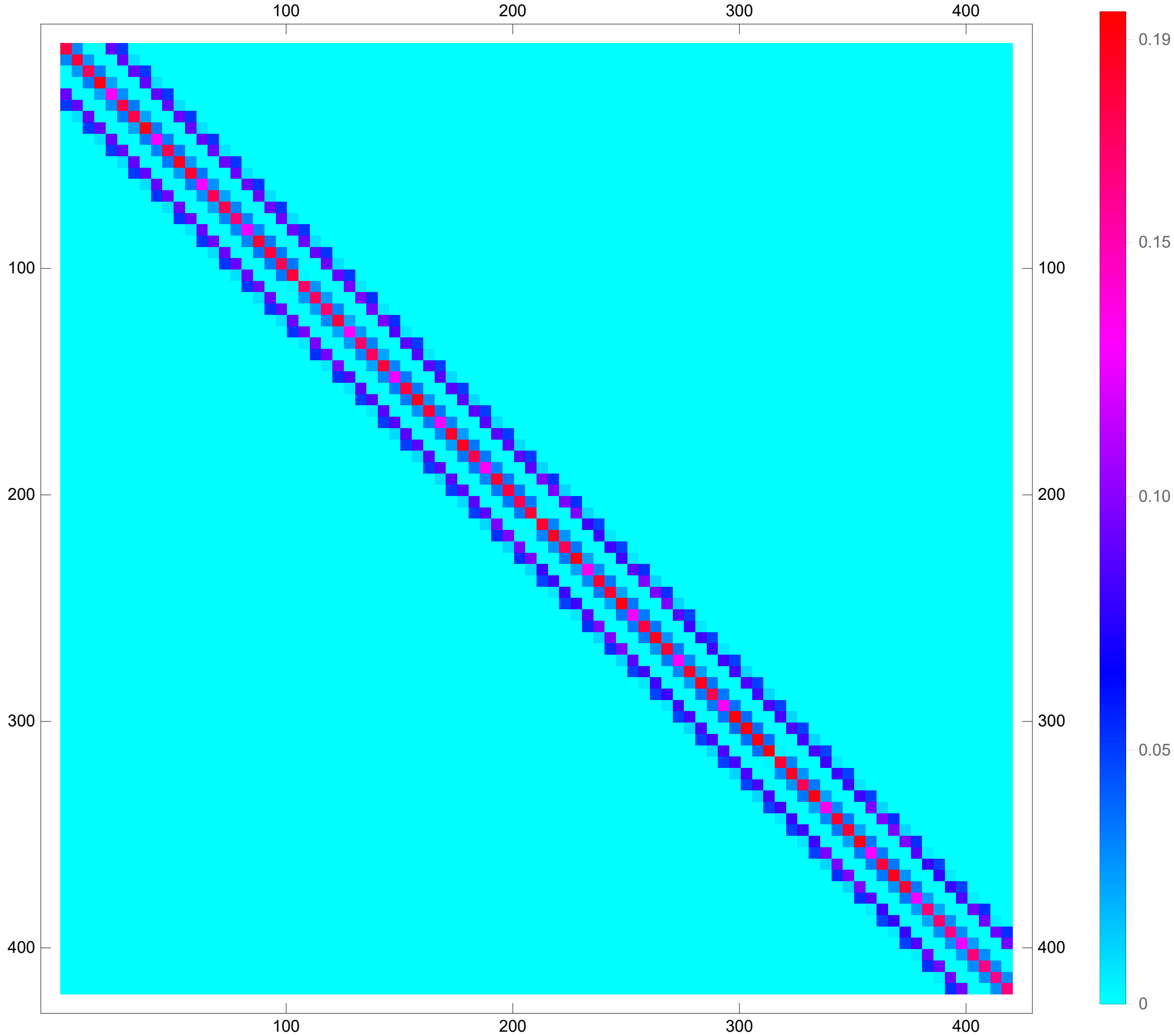

```
(********************* ----- HAMILTONIANO
  en Espacio Real ----- *********************)
```
real

```
(*Para este problema simplificado queda identico*)

HReal = HEdoBase;
HEdoBase // MatrixForm;
```
forma de matriz

```
MatrixPlot[HEdoBase, PlotTheme → "Detailed", ImageSize → Large, ColorFunction → Hue]
```
representación de matriz | tema de representación | tamaño de i… | grande | función de color | tonalidad

```
(* Para usar Green se requiere el
```
verde

```
  verde
 Hamiltoniano discretiado (Matriz) en el espacio Real *)
                                                 real


(* Con Green obtendremos observables en función de la energía *)
     verde
Energia = En * IdentityMatrix[ Round[NEdos]  ];
               matriz identidad  entero más próximo


(* Se ingresa las posiciones atomicas que actuaran de Fuente y Drenante *)
PosicionFuente = Table[ i  , {i, 1, NZigzag, 1}];   (*Table[   i   ,{i,1,NZigzag,1}]*)
                 tabla                                tabla
PosicionDrentante = Table[  NEdos - NZigzag + i  , {i, 1, NZigzag, 1}];
                    tabla

PosicionFuente =  Table  [  Round[{PosicionFuente[[i]] , PosicionFuente[[i]] }]  ,
                  tabla     entero más próximo
   {i, 1, Length[PosicionFuente], 1}];
          longitud
(* Genera las posiciones correspondientes a la matriz *)
PosicionDrentante =  Table  [  Round[{PosicionDrentante[[i]] , PosicionDrentante[[i]] }]  ,
                     tabla     entero más próximo
   {i, 1, Length[PosicionDrentante], 1}];
          longitud

(* Matrices de Autoenergías *)
   matrices
SigmaFuente = -i * tF *
   SparseArray[{    PosicionFuente → Table[1. , {i, 1, Length[PosicionFuente], 1} ]    },
   array disperso                    tabla              longitud
    {NEdos, NEdos}];
SigmaDrenante =  -i * tD * SparseArray[{    PosicionDrentante →
                           array disperso
      Table[1. , {i, 1, Length[PosicionDrentante], 1} ]    }, {NEdos, NEdos}];
      tabla              longitud


tF = 1.;
tD = 1.;
t1 = 1.;
```




```
(********************* ----- I Operador ----- *********************)
                                   número i

(*
(*  Solución Numerica   *)
{eval,evec}=Eigensystem[HEdoBase,  1,Method→{"Arnoldi","Criteria"→"RealPart"} ]
           autovalores y autovectores      método

   Print[     Style["Energias Numericas ",18,Bold,Purple]       ]
   escribe    estilo                         negr… púrpura
```

```
escribe     estilo                          negr··· púrpura
Style[   MatrixForm[  eval  ]   ,{Medium,Bold,Purple}]
estilo   forma de matriz            tamañ···  negr··· púrpura


 (*  Solución Exacta:   Usa el mismo hamiltoniano de interacción   *)

 t1*HEdoBase   // MatrixForm;
                  forma de matriz

Print[    Style["Energias Exactas ",18,Bold,Purple]     ]
escribe   estilo                        negr··· púrpura
 Quiet[        Solve[
 silencioso    resuelve
   Det[           t1*HEdoBase-En*IdentityMatrix[   Round[NEdos]    ]          ]==0    ,
   determinante                   matriz identidad   entero más próximo
   En]            ]

 (*Eigenfunction*)



 eval[[1]];
EdoBase=evec[[1]];


(*    Ii  = ⅈ*(e/h)*2*π*( H Gn - Gn H )         Notacion:  bn+1† bn {n;...} =
  √1 √0+1 {0,...,0,1-1,0+1,...,0;...} = {n+1;...} *)
(HEdoBase- SigmaFuente  -  SigmaDrenante) *GnMatrizDensidad // MatrixForm;
                                                             forma de matriz

GnMatrizDensidad=Table[    Table[
                 tabla     tabla
   Conjugate[  EdoBase[[ j ]]  ]*EdoBase[[ i ]]   ,{i,1,NEdos,1}]   ,{j,1,NEdos,1}];
   conjugado
GnMatrizDensidad // MatrixForm;
                    forma de matriz

IOperador=ⅈ*(   ( HEdoBase    +  SigmaDrenante) *GnMatrizDensidad    +
     GnMatrizDensidad    *  ( HEdoBase   +  SigmaDrenante)    ) ;

IVal=Im[IOperador];
     parte imaginaria
IVal// MatrixForm
       forma de matriz
```

```
 ILocalY=Table[          Table[     -IVal[[il,  (jl-1)*NZigzag+Mod[il-1,NZigzag,1]  ]]+
        tabla            tabla                                   operación módulo
    IVal[[il, (jl-1)*NZigzag+Mod[il+1,NZigzag,1]   ]]      ,
                               operación módulo
   {il, 1+(jl-1)*NZigzag,jl*NZigzag,1}]     , {jl,1,NArmchair,1}];
ILocalY=Flatten[ ILocalY];
        aplana

ILocalX=
 Join[     Table[     IVal[[il,il+NZigzag ]]     ,{il,1,NZigzag*(NArmchair-1),1}  ] ,
 junta     tabla
  Table[ 0., {jl,1,NZigzag,1}]       ];
  tabla



ILocal=              Table[
                     tabla
  {       {  1.+IntegerPart[   (i-1)/(NZigzag ) ],  Mod[ i,NZigzag,1.]      }   ,
             parte entera                             operación módulo
   {ILocalX[[i]],  ILocalY[[i]] }    } ,   {i,1, NEdos,1.}];


ListVectorPlot[      ILocal     ,ImageSize→Large, VectorColorFunction→"Rainbow",
representación vectorial de lista tamaño de ··· grande función de color de vector
  VectorPoints→10, VectorScale→0.08, PlotLegends→Automatic]
  número de puntos de ··· escala de vector leyendas de re·· automático
 ListStreamPlot[      ILocal     ,ImageSize→Large, VectorColorFunction→"Rainbow",
 representación de flujo de lista  tamaño de ··· grande función de color de vector
  VectorPoints→10, VectorScale→0.08, PlotLegends→Automatic]
  número de puntos de ··· escala de vector leyendas de re·· automático


 Clear[En]
 borra
```

(* $J_i = 2*J*(b_{i+1}^{\dagger}\ b_i - b_{i-1}^{\dagger}\ b_i)$ Notacion: $b_{n+1}^{\dagger}\ b_n$ {n;...} =
$\sqrt{1}\sqrt{0+1}$ {0,...,0,1-1,0+1,...,0;...} = {n+1;...} *)

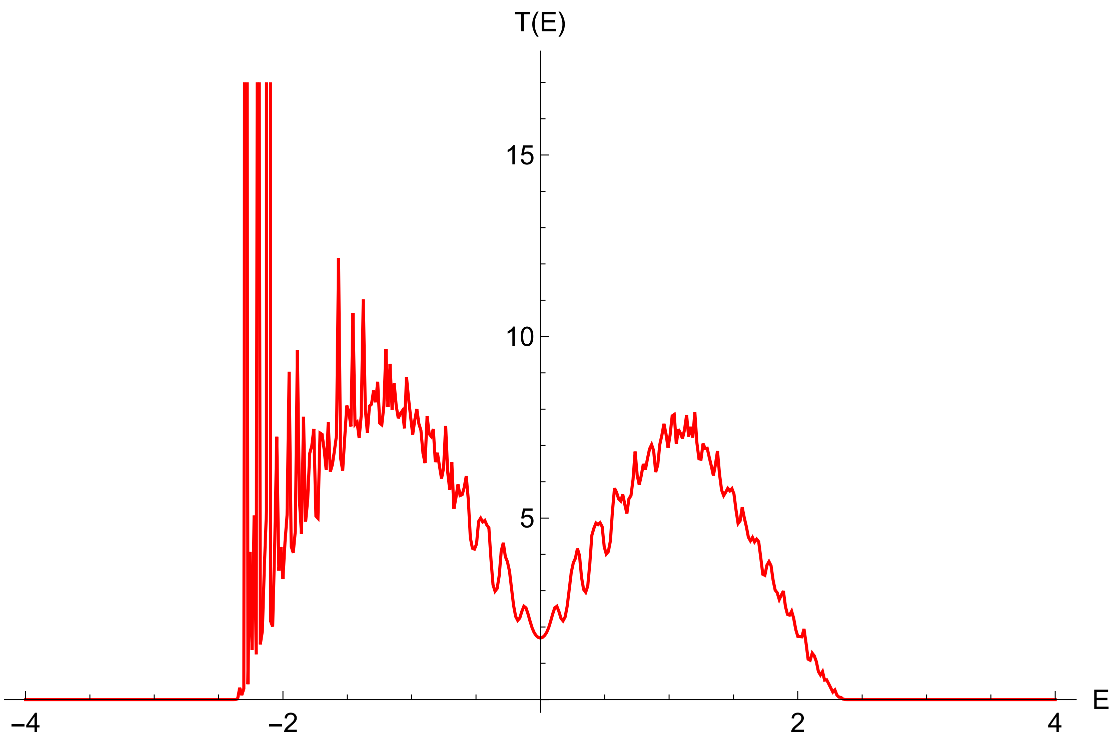
*)

```
(* Con Green obtendremos observables en función de la energía *)
        verde
EnMin = -4.;
EnMax = 4.;
NEnPuntos = 501.;

DEn  = Abs[EnMax - EnMin] / (NEnPuntos - 1.);
       valor absoluto
EnergiasValores = Table [ (i)   , {i, EnMin, EnMax, DEn}  ];
                  tabla

Energia =  Table [ (i) * IdentityMatrix[ Round[NEdos]  ]   , {i, EnMin, EnMax, DEn}  ];
           tabla        matriz identidad   entero más próximo

(*Broadening Matrix*)
GammaFuente  = ⅈ * ( SigmaFuente - ConjugateTranspose[ SigmaFuente ]);
                                   transpuesto conjugado
GammaDrenante = ⅈ * ( SigmaDrenante - ConjugateTranspose[ SigmaDrenante ]);
                                      transpuesto conjugado

(*Funcion de Green del Nanosistema*)
              verde
FGreen = Table[    Inverse[
         tabla     matriz inversa
    Energia [[i]]  -  HEdoBase  - SigmaFuente  -  SigmaDrenante  ]    , {i, 1, NEnPuntos, 1}];

(*Funcion de Transmision*)
Print[   Style["Función de Transmisión", 18, Bold, Purple]    ]
escribe  estilo                                negrita púrpura
```

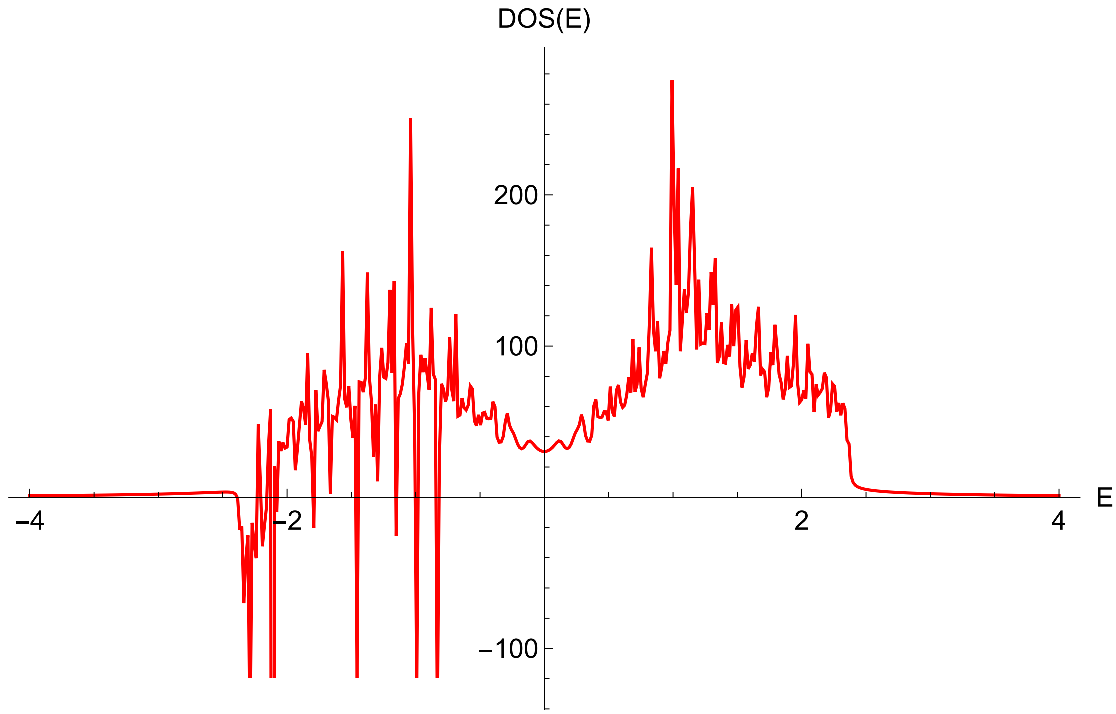
```
escribe    estilo                                     negrita púrpura
FTranmision =
  Table[    {EnMin + DEn * (i - 1) ,    Tr[  GammaDrenante. FGreen[[i]] .GammaFuente.
  tabla                                 traza
      ConjugateTranspose[ FGreen[[i]] ]  ] }  , {i, 1, NEnPuntos, 1}];
      transpuesto conjugado

ListPlot[  FTranmision  , PlotStyle → {{PointSize[0.01], Red}},
representación de lista   estilo de represe··· tamaño de punto   rojo
 LabelStyle → {14, Black}, AxesLabel → {"E" , "T(E)"}, ImageSize → Large,  Joined → True]
 estilo de etiqueta      negro   etiqueta de ejes  númer··· núme··· tamaño de i···  grande  unido  verdade




(*Función Espectral*)
A =  Table[
     tabla
    i * ( FGreen[[i]] - ConjugateTranspose[ FGreen[[i]] ] )   , {i, 1, NEnPuntos, 1}];
                        transpuesto conjugado



(*DOS: Densidad de Estados*)
Print[  Style["DOS", 18, Bold, Purple]   ]
escribe  estilo                negrita púrpura
DOS =  Table[
       tabla
    { EnMin + DEn * (i - 1) ,     Re[ Tr[ A[[i]] ] / (2 * π)  ] }  , {i, 1, NEnPuntos, 1}];
                                  par·· traza

ListPlot[  DOS  , PlotStyle → {{PointSize[0.01], Red}}, LabelStyle → {14, Black},
representación de lista  estilo de represe··· tamaño de punto  rojo  estilo de etiqueta  negro
 AxesLabel → {"E" , "DOS(E)"}, ImageSize → Large,  Joined → True]
 etiqueta de ejes  número e  núme··· tamaño de i···  grande  unido  verdadero


(*LDOS: Densidad Local de Estados*)
Print[  Style["LDOS", 18, Bold, Purple]   ]
escribe  estilo                 negrita púrpura
LDOS =  Table[     Re[ Diagonal[ A[[i]] ] / (2 * π) ]     , {i, 1, NEnPuntos, 1}];
        tabla      pa··· diagonal
(*Los elementos de la Diagonal de A *)
                      diagonal

LDOSSitios = Table[       Table[   { EnMin + DEn * (i - 1) ,    LDOS[[i, j]] }   ,
             tabla        tabla
    {i, 1, NEnPuntos, 1}]        ,   {j, 1, NEdos, 1}] ;
```

```
ListPlot[  LDOSSitios  , PlotStyle → {{PointSize[0.012], Blue},
|representación de lista   |estilo de represe··· |tamaño de punto   |azul
   {PointSize[0.015], Red}, {PointSize[0.01], Green}, {PointSize[0.006], Orange},
   |tamaño de punto   |rojo   |tamaño de punto   |verde   |tamaño de punto   |naranja
   {PointSize[0.003], Yellow}, {PointSize[0.008], Pink}}, LabelStyle → {14, Black},
   |tamaño de punto   |amarillo   |tamaño de punto   |rosa   |estilo de etiqueta   |negro
 AxesLabel → {"E" , "LDOS(E)"}, ImageSize → Large,  Joined → True]
            |número e   |núme···|tamaño de i···  |grande   |unido   |verdadero
```

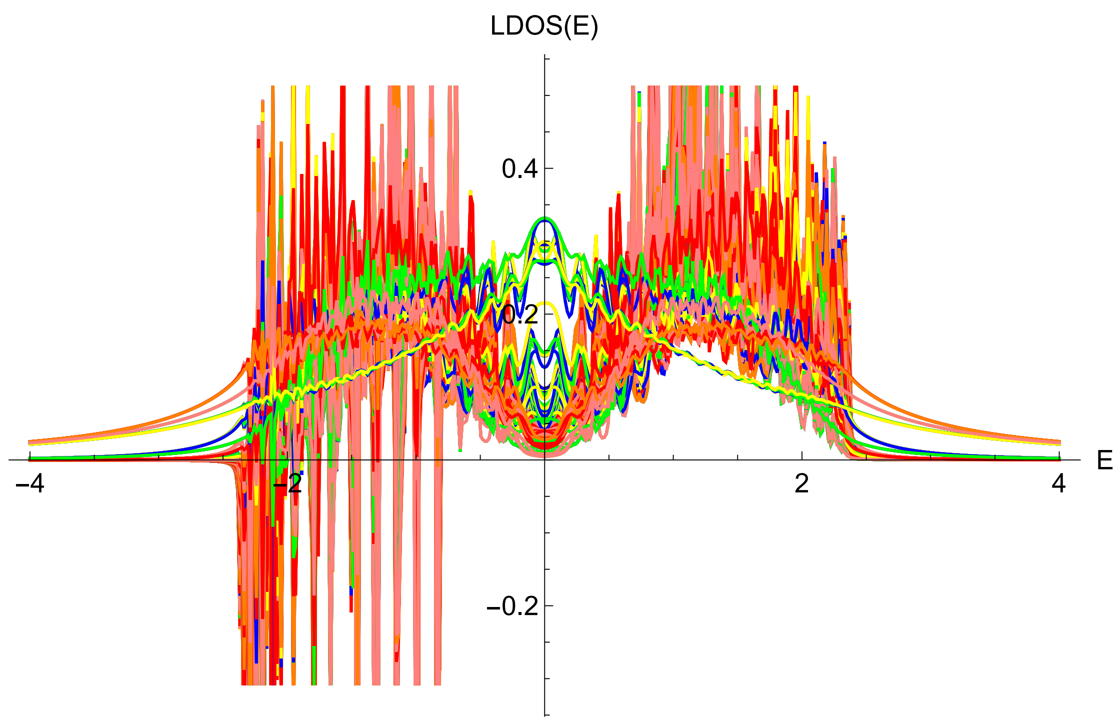
Función de Transmisión



DOS



LDOS

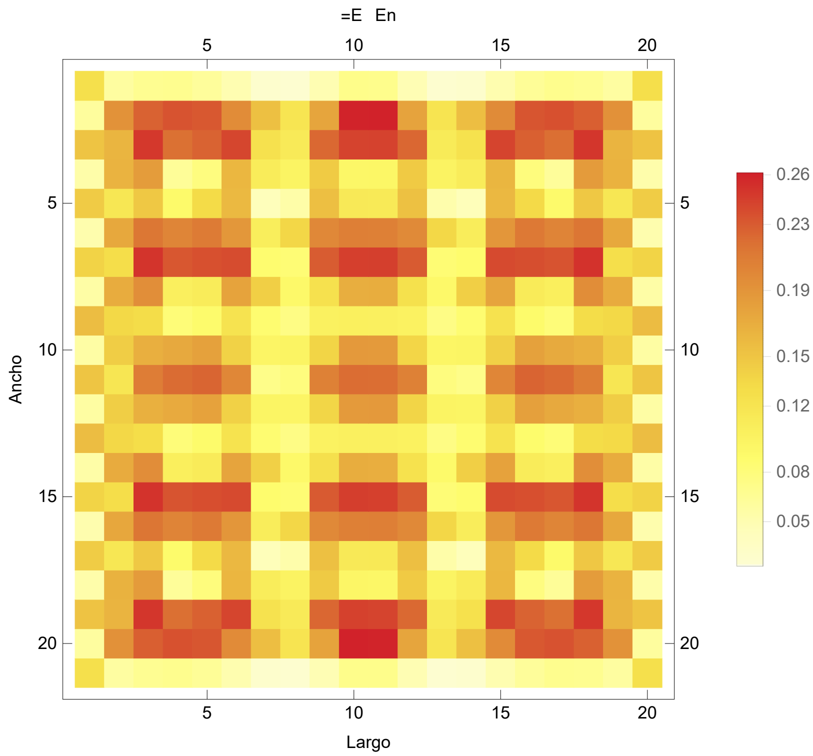


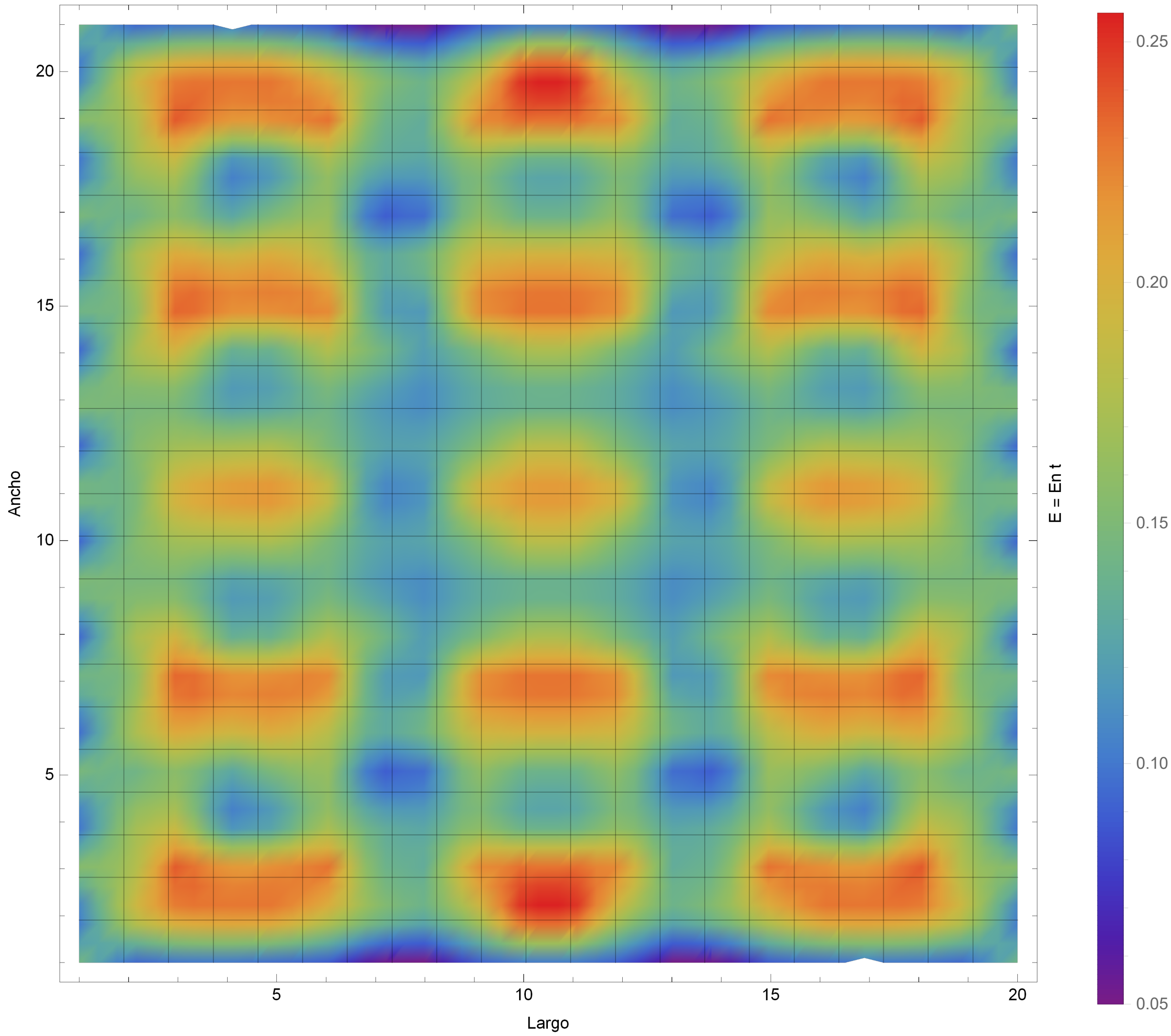
```
EnergiasValores
```

```
{-4., -3.984, -3.968, -3.952, -3.936, -3.92, -3.904, -3.888, -3.872, -3.856, -3.84,
 -3.824, -3.808, -3.792, -3.776, -3.76, -3.744, -3.728, -3.712, -3.696, -3.68, -3.664,
 -3.648, -3.632, -3.616, -3.6, -3.584, -3.568, -3.552, -3.536, -3.52, -3.504, -3.488,
 -3.472, -3.456, -3.44, -3.424, -3.408, -3.392, -3.376, -3.36, -3.344, -3.328, -3.312,
 -3.296, -3.28, -3.264, -3.248, -3.232, -3.216, -3.2, -3.184, -3.168, -3.152, -3.136,
 -3.12, -3.104, -3.088, -3.072, -3.056, -3.04, -3.024, -3.008, -2.992, -2.976,
 -2.96, -2.944, -2.928, -2.912, -2.896, -2.88, -2.864, -2.848, -2.832, -2.816, -2.8,
 -2.784, -2.768, -2.752, -2.736, -2.72, -2.704, -2.688, -2.672, -2.656, -2.64,
 -2.624, -2.608, -2.592, -2.576, -2.56, -2.544, -2.528, -2.512, -2.496, -2.48,
 -2.464, -2.448, -2.432, -2.416, -2.4, -2.384, -2.368, -2.352, -2.336, -2.32, -2.304,
 -2.288, -2.272, -2.256, -2.24, -2.224, -2.208, -2.192, -2.176, -2.16, -2.144,
 -2.128, -2.112, -2.096, -2.08, -2.064, -2.048, -2.032, -2.016, -2., -1.984, -1.968,
 -1.952, -1.936, -1.92, -1.904, -1.888, -1.872, -1.856, -1.84, -1.824, -1.808,
 -1.792, -1.776, -1.76, -1.744, -1.728, -1.712, -1.696, -1.68, -1.664, -1.648,
 -1.632, -1.616, -1.6, -1.584, -1.568, -1.552, -1.536, -1.52, -1.504, -1.488, -1.472,
 -1.456, -1.44, -1.424, -1.408, -1.392, -1.376, -1.36, -1.344, -1.328, -1.312,
 -1.296, -1.28, -1.264, -1.248, -1.232, -1.216, -1.2, -1.184, -1.168, -1.152, -1.136,
 -1.12, -1.104, -1.088, -1.072, -1.056, -1.04, -1.024, -1.008, -0.992, -0.976,
 -0.96, -0.944, -0.928, -0.912, -0.896, -0.88, -0.864, -0.848, -0.832, -0.816, -0.8,
 -0.784, -0.768, -0.752, -0.736, -0.72, -0.704, -0.688, -0.672, -0.656, -0.64,
 -0.624, -0.608, -0.592, -0.576, -0.56, -0.544, -0.528, -0.512, -0.496, -0.48, -0.464,
 -0.448, -0.432, -0.416, -0.4, -0.384, -0.368, -0.352, -0.336, -0.32, -0.304, -0.288,
 -0.272, -0.256, -0.24, -0.224, -0.208, -0.192, -0.176, -0.16, -0.144, -0.128, -0.112,
 -0.096, -0.08, -0.064, -0.048, -0.032, -0.016, 0., 0.016, 0.032, 0.048, 0.064, 0.08,
 0.096, 0.112, 0.128, 0.144, 0.16, 0.176, 0.192, 0.208, 0.224, 0.24, 0.256, 0.272, 0.288,
 0.304, 0.32, 0.336, 0.352, 0.368, 0.384, 0.4, 0.416, 0.432, 0.448, 0.464, 0.48, 0.496,
 0.512, 0.528, 0.544, 0.56, 0.576, 0.592, 0.608, 0.624, 0.64, 0.656, 0.672, 0.688, 0.704,
 0.72, 0.736, 0.752, 0.768, 0.784, 0.8, 0.816, 0.832, 0.848, 0.864, 0.88, 0.896, 0.912,
 0.928, 0.944, 0.96, 0.976, 0.992, 1.008, 1.024, 1.04, 1.056, 1.072, 1.088, 1.104, 1.12,
 1.136, 1.152, 1.168, 1.184, 1.2, 1.216, 1.232, 1.248, 1.264, 1.28, 1.296, 1.312, 1.328,
 1.344, 1.36, 1.376, 1.392, 1.408, 1.424, 1.44, 1.456, 1.472, 1.488, 1.504, 1.52, 1.536,
 1.552, 1.568, 1.584, 1.6, 1.616, 1.632, 1.648, 1.664, 1.68, 1.696, 1.712, 1.728, 1.744,
 1.76, 1.776, 1.792, 1.808, 1.824, 1.84, 1.856, 1.872, 1.888, 1.904, 1.92, 1.936, 1.952,
 1.968, 1.984, 2., 2.016, 2.032, 2.048, 2.064, 2.08, 2.096, 2.112, 2.128, 2.144, 2.16,
 2.176, 2.192, 2.208, 2.224, 2.24, 2.256, 2.272, 2.288, 2.304, 2.32, 2.336, 2.352, 2.368,
 2.384, 2.4, 2.416, 2.432, 2.448, 2.464, 2.48, 2.496, 2.512, 2.528, 2.544, 2.56, 2.576,
 2.592, 2.608, 2.624, 2.64, 2.656, 2.672, 2.688, 2.704, 2.72, 2.736, 2.752, 2.768, 2.784,
 2.8, 2.816, 2.832, 2.848, 2.864, 2.88, 2.896, 2.912, 2.928, 2.944, 2.96, 2.976, 2.992,
 3.008, 3.024, 3.04, 3.056, 3.072, 3.088, 3.104, 3.12, 3.136, 3.152, 3.168, 3.184, 3.2,
 3.216, 3.232, 3.248, 3.264, 3.28, 3.296, 3.312, 3.328, 3.344, 3.36, 3.376, 3.392,
 3.408, 3.424, 3.44, 3.456, 3.472, 3.488, 3.504, 3.52, 3.536, 3.552, 3.568, 3.584, 3.6,
 3.616, 3.632, 3.648, 3.664, 3.68, 3.696, 3.712, 3.728, 3.744, 3.76, 3.776, 3.792,
 3.808, 3.824, 3.84, 3.856, 3.872, 3.888, 3.904, 3.92, 3.936, 3.952, 3.968, 3.984, 4.}
```

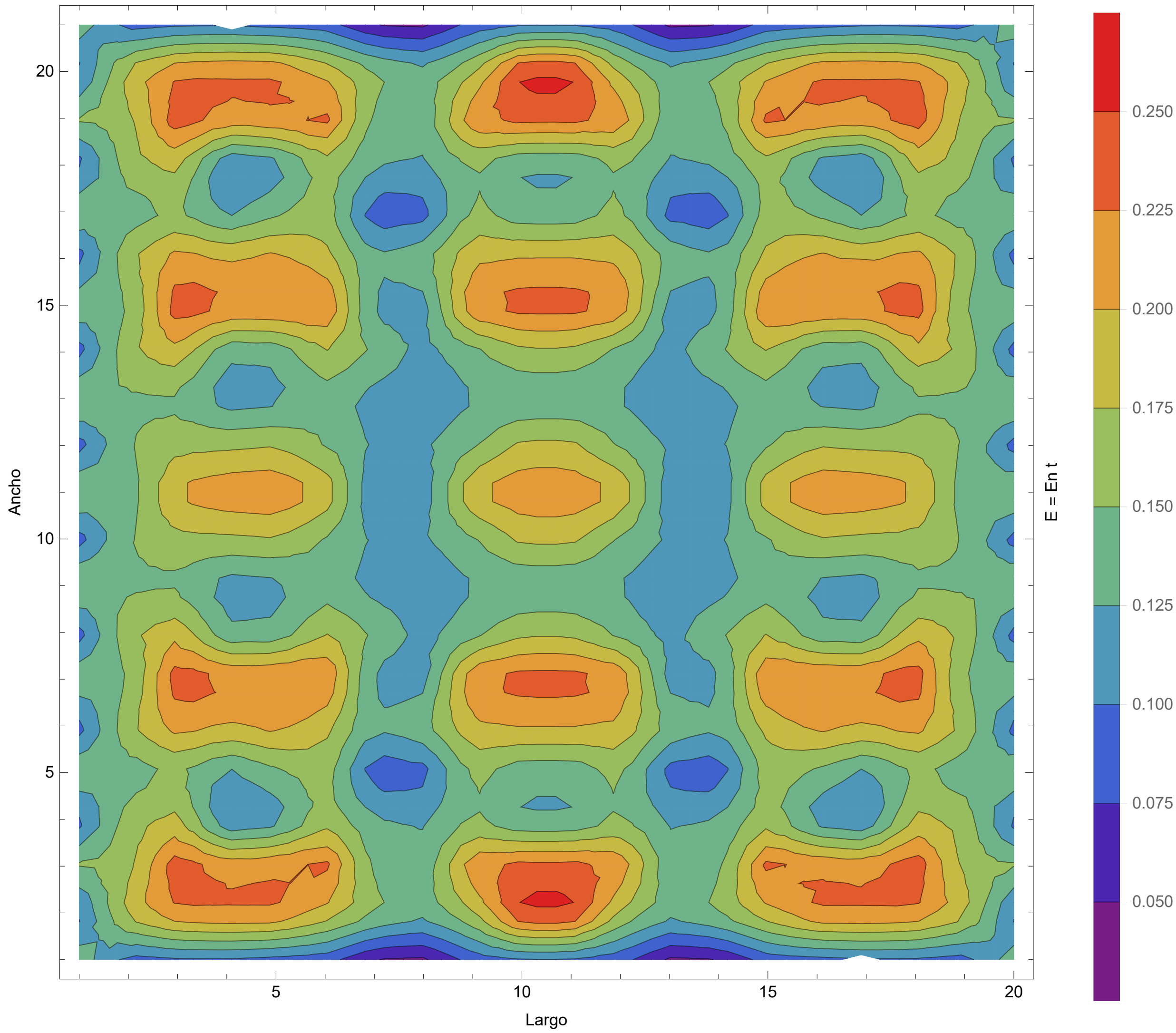
```
(*LDOS en Espacio REAL*)
Print[  Style["LDOS", 18, Bold, Purple]  ]
escribe  estilo                 negrita púrpura
EnVal = 2;
NEn = 1. + (EnVal - EnMin) / DEn;


LDOSRealMatrix =   Table[
                   tabla
   SparseArray[ {{i_, i_} → 0.}, {NZigzag, NArmchair} ]     , {j, 1, NEnPuntos, 1}];
   array disperso
Do[    Do[   LDOSRealMatrix [[j,    Mod[ i, NZigzag, 1.] ,
repite repite                      operación módulo
      1. + IntegerPart[ (i - 1) / NZigzag ]  ]] = Re[ LDOS [[j, i]] ]   ,
           parte entera                           parte real
   {i, 1, NEdos, 1.}]      , {j, 1, NEnPuntos, 1}];

MatrixPlot[  LDOSRealMatrix [[NEn]]  ,
representación de matriz
 PlotTheme → "Detailed", ColorFunction -> "TemperatureMap",
 tema de representación    función de color
 FrameLabel → {{"Largo", HoldForm["=E  "] En}, {"Ancho", None}}]
                         forma sin eva·· número e              ninguno


LDOSRealDensity =
  Table[      Table[ { 1. + IntegerPart[ (i - 1) / NZigzag ], Mod[ i, NZigzag, 1.],
  tabla       tabla       parte entera                          operación módulo
      Re[ LDOS [[j, i]] ]} ,  {i, 1, NEdos, 1.}]      , {j, 1, NEnPuntos, 1}];
      parte real

ListDensityPlot[  LDOSRealDensity [[NEn]]   ,
representación de densidad de lista
 FrameLabel → {{"Ancho", StringTemplate["E = `1` t"][En]}, {"Largo", None}},
 etiqueta de marco       plantilla de cadena ··· número e                ninguno
 ImageSize → Large, ColorFunction → "Rainbow", InterpolationOrder → 1,
             grande  función de color               orden de interpolación
 MaxPlotPoints → 50, Mesh → {20, 21}, PlotLegends → Automatic]
 máximo número de punto·· malla          leyendas de rep··· automático
ListContourPlot[  LDOSRealDensity [[NEn]]   ,
representación de contornos de lista
 FrameLabel → {{"Ancho", StringTemplate["E = `1` t"][En]}, {"Largo", None}},
                         plantilla de cadena ··· número e                ninguno
 ImageSize → Large, ColorFunction → "Rainbow", InterpolationOrder → 1,
             grande  función de color               orden de interpolación
 MaxPlotPoints → 50, Mesh → {6, 7}, PlotLegends → Automatic]
                     malla          leyendas de rep··· automático
```

LDOS



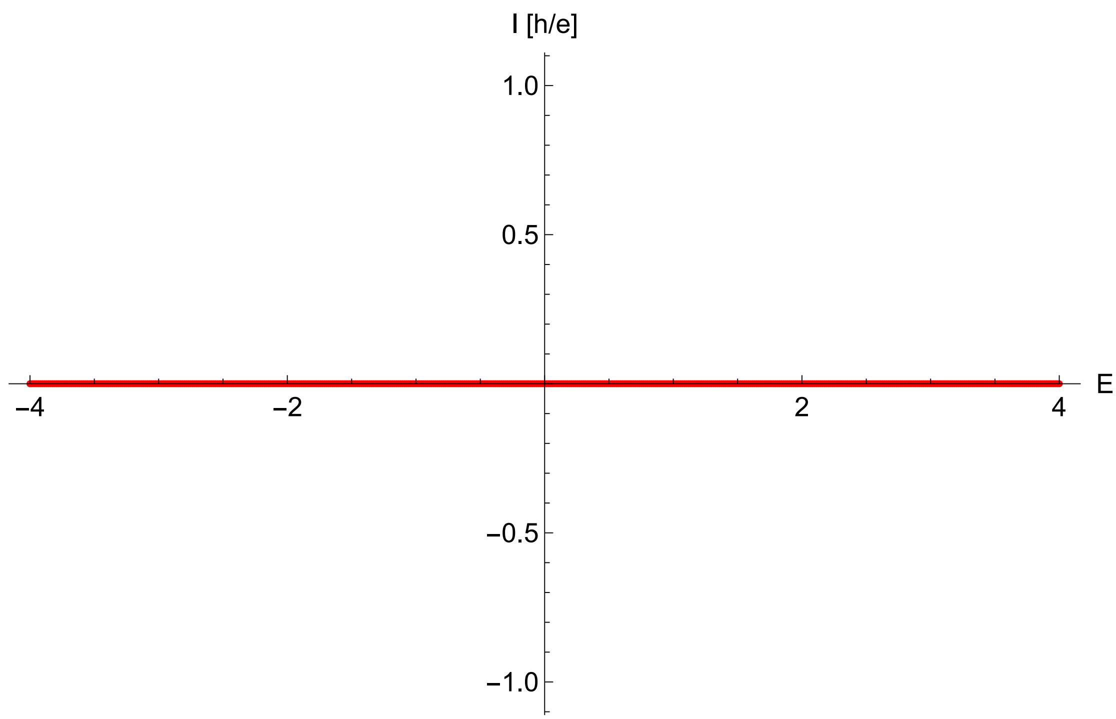


20
15
10
5
Ancho
5
10
15
20
Largo
E = En t
0.25
0.20
0.15
0.10
0.05


20
15
10
5
5
10
15
20
Largo
Ancho
E = En t
0.250
0.225
0.200
0.175
0.150
0.125
0.100
0.075
0.050

```
(********** -------- Corriente -------- **********)
Print[  Style["Corriente Total", 18, Bold, Purple]  ]
escribe  estilo            total          negrita púrpura
fFermi2 = 1;     (*Suponiendo Funciones de Fermi iguales a 1*)
SigmaDispersiva = GammaFuente;    (*Energías de Dispersión, In-Scattering*)
                                                              entrada
FGreenIn =
  Table[      FGreen[[i]] . SigmaDispersiva. ConjugateTranspose[ FGreen[[i]] ]     ,
  tabla                                      transpuesto conjugado
   {i, 1, NEnPuntos, 1}];    (*Funcion de Green de Dispersión*)
                                       verde
IVal =  Table[      Im[ 2 * FGreenIn[[i]] * fFermi2 ]     , {i, 1, NEnPuntos, 1}];
        tabla       parte imaginaria
(*Matrices con los valores de las corrientes*)
  matrices
ITotal =  Table[
          tabla
   { EnMin  + DEn * (i - 1)  ,     Tr[ IVal[[i]] ]   }     , {i, 1, NEnPuntos, 1}];
                                   traza
(*Corriente Total en Función de la Energía: Unidades (h/e) *)
             total


ListPlot[   ITotal   , PlotStyle → {{PointSize[0.0065], Red}},
representación de lista  estilo de represe··· tamaño de punto  rojo
 LabelStyle → {14, Black}, AxesLabel → {"E" , "I  [h/e]"}, ImageSize → Large]
                   negro     etiqueta de ejes  nú···  número i   tamaño de i···  grande
```

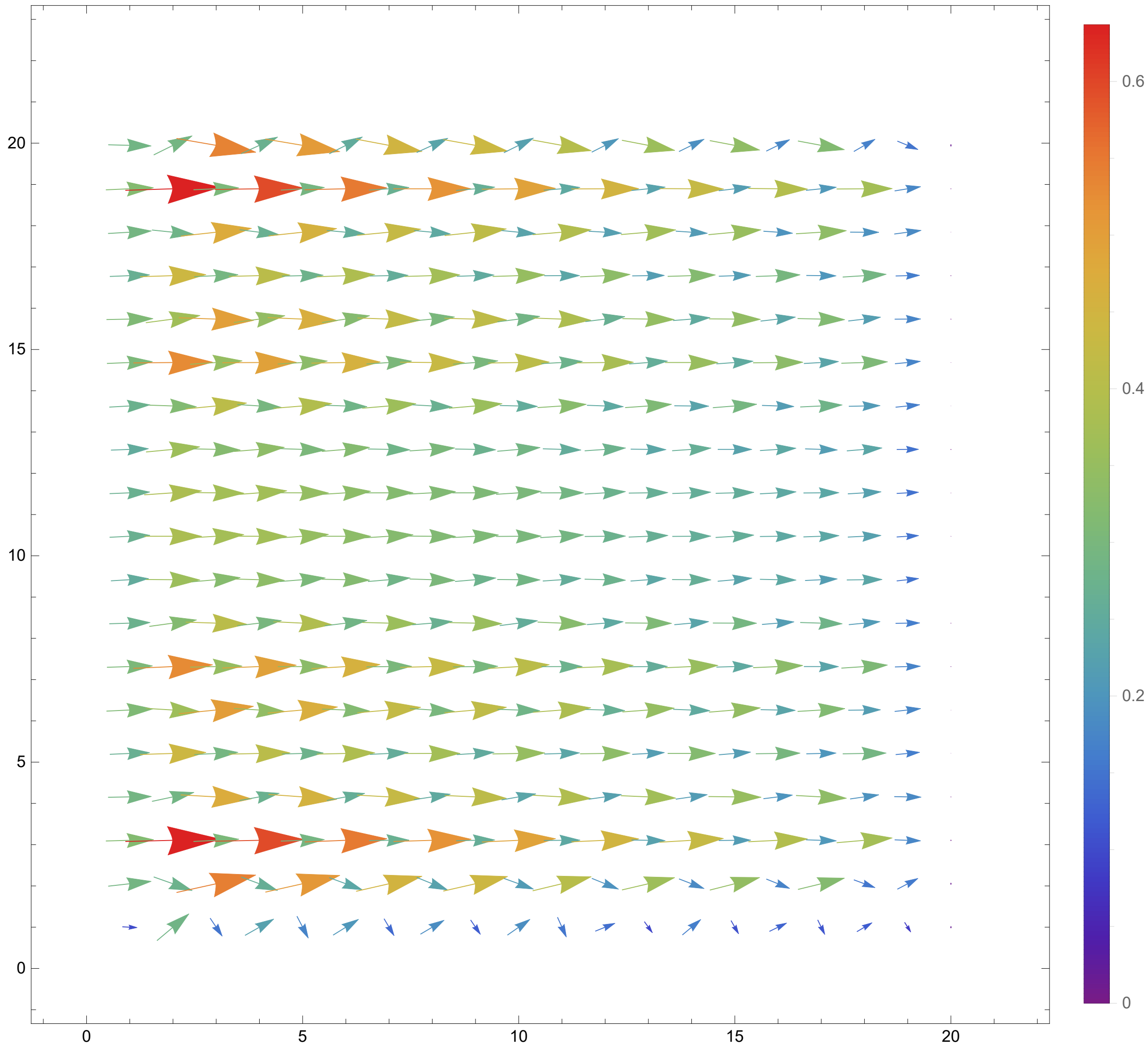
Corriente Total

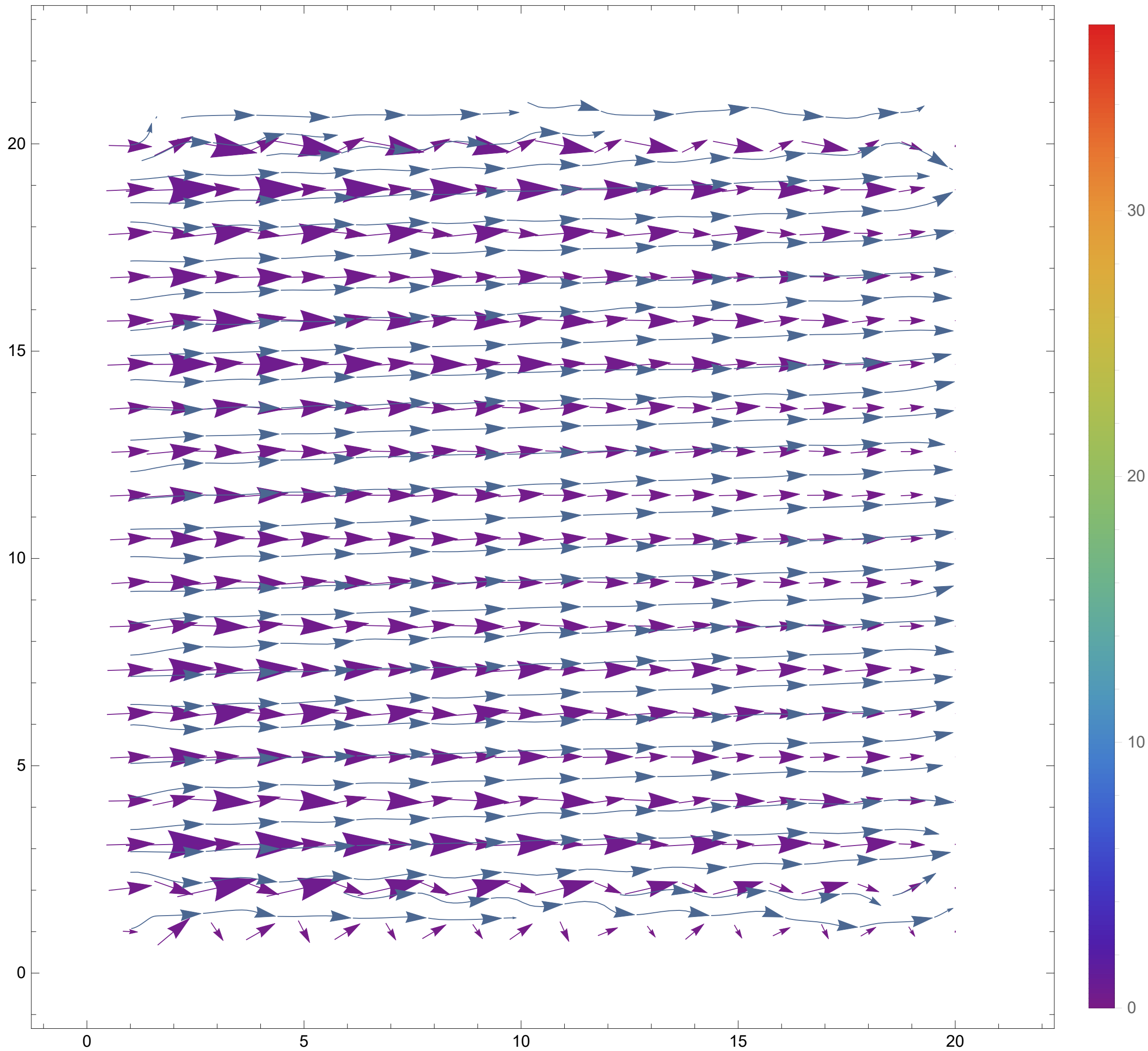

I [h/e]
1.0
0.5
−0.5
−1.0
−4
−2
2
4
E

EnergiasValores

{-4., -3.984, -3.968, -3.952, -3.936, -3.92, -3.904, -3.888, -3.872, -3.856, -3.84, -3.824, -3.808, -3.792, -3.776, -3.76, -3.744, -3.728, -3.712, -3.696, -3.68, -3.664, -3.648, -3.632, -3.616, -3.6, -3.584, -3.568, -3.552, -3.536, -3.52, -3.504, -3.488, -3.472, -3.456, -3.44, -3.424, -3.408, -3.392, -3.376, -3.36, -3.344, -3.328, -3.312, -3.296, -3.28, -3.264, -3.248, -3.232, -3.216, -3.2, -3.184, -3.168, -3.152, -3.136, -3.12, -3.104, -3.088, -3.072, -3.056, -3.04, -3.024, -3.008, -2.992, -2.976, -2.96, -2.944, -2.928, -2.912, -2.896, -2.88, -2.864, -2.848, -2.832, -2.816, -2.8, -2.784, -2.768, -2.752, -2.736, -2.72, -2.704, -2.688, -2.672, -2.656, -2.64, -2.624, -2.608, -2.592, -2.576, -2.56, -2.544, -2.528, -2.512, -2.496, -2.48, -2.464, -2.448, -2.432, -2.416, -2.4, -2.384, -2.368, -2.352, -2.336, -2.32, -2.304, -2.288, -2.272, -2.256, -2.24, -2.224, -2.208, -2.192, -2.176, -2.16, -2.144, -2.128, -2.112, -2.096, -2.08, -2.064, -2.048, -2.032, -2.016, -2., -1.984, -1.968, -1.952, -1.936, -1.92, -1.904, -1.888, -1.872, -1.856, -1.84, -1.824, -1.808, -1.792, -1.776, -1.76, -1.744, -1.728, -1.712, -1.696, -1.68, -1.664, -1.648, -1.632, -1.616, -1.6, -1.584, -1.568, -1.552, -1.536, -1.52, -1.504, -1.488, -1.472, -1.456, -1.44, -1.424, -1.408, -1.392, -1.376, -1.36, -1.344, -1.328, -1.312, -1.296, -1.28, -1.264, -1.248, -1.232, -1.216, -1.2, -1.184, -1.168, -1.152, -1.136, -1.12, -1.104, -1.088, -1.072, -1.056, -1.04, -1.024, -1.008, -0.992, -0.976, -0.96, -0.944, -0.928, -0.912, -0.896, -0.88, -0.864, -0.848, -0.832, -0.816, -0.8, -0.784, -0.768, -0.752, -0.736, -0.72, -0.704, -0.688, -0.672, -0.656, -0.64, -0.624, -0.608, -0.592, -0.576, -0.56, -0.544, -0.528, -0.512, -0.496, -0.48, -0.464, -0.448, -0.432, -0.416, -0.4, -0.384, -0.368, -0.352, -0.336, -0.32, -0.304, -0.288, -0.272, -0.256, -0.24, -0.224, -0.208, -0.192, -0.176, -0.16, -0.144, -0.128, -0.112, -0.096, -0.08, -0.064, -0.048, -0.032, -0.016, 0., 0.016, 0.032, 0.048, 0.064, 0.08, 0.096, 0.112, 0.128, 0.144, 0.16, 0.176, 0.192, 0.208, 0.224, 0.24, 0.256, 0.272, 0.288, 0.304, 0.32, 0.336, 0.352, 0.368, 0.384, 0.4, 0.416, 0.432, 0.448, 0.464, 0.48, 0.496, 0.512, 0.528, 0.544, 0.56, 0.576, 0.592, 0.608, 0.624, 0.64, 0.656, 0.672, 0.688, 0.704, 0.72, 0.736, 0.752, 0.768, 0.784, 0.8, 0.816, 0.832, 0.848, 0.864, 0.88, 0.896, 0.912, 0.928, 0.944, 0.96, 0.976, 0.992, 1.008, 1.024, 1.04, 1.056, 1.072, 1.088, 1.104, 1.12, 1.136, 1.152, 1.168, 1.184, 1.2, 1.216, 1.232, 1.248, 1.264, 1.28, 1.296, 1.312, 1.328, 1.344, 1.36, 1.376, 1.392, 1.408, 1.424, 1.44, 1.456, 1.472, 1.488, 1.504, 1.52, 1.536, 1.552, 1.568, 1.584, 1.6, 1.616, 1.632, 1.648, 1.664, 1.68, 1.696, 1.712, 1.728, 1.744, 1.76, 1.776, 1.792, 1.808, 1.824, 1.84, 1.856, 1.872, 1.888, 1.904, 1.92, 1.936, 1.952, 1.968, 1.984, 2., 2.016, 2.032, 2.048, 2.064, 2.08, 2.096, 2.112, 2.128, 2.144, 2.16, 2.176, 2.192, 2.208, 2.224, 2.24, 2.256, 2.272, 2.288, 2.304, 2.32, 2.336, 2.352, 2.368, 2.384, 2.4, 2.416, 2.432, 2.448, 2.464, 2.48, 2.496, 2.512, 2.528, 2.544, 2.56, 2.576, 2.592, 2.608, 2.624, 2.64, 2.656, 2.672, 2.688, 2.704, 2.72, 2.736, 2.752, 2.768, 2.784, 2.8, 2.816, 2.832, 2.848, 2.864, 2.88, 2.896, 2.912, 2.928, 2.944, 2.96, 2.976, 2.992, 3.008, 3.024, 3.04, 3.056, 3.072, 3.088, 3.104, 3.12, 3.136, 3.152, 3.168, 3.184, 3.2, 3.216, 3.232, 3.248, 3.264, 3.28, 3.296, 3.312, 3.328, 3.344, 3.36, 3.376, 3.392, 3.408, 3.424, 3.44, 3.456, 3.472, 3.488, 3.504, 3.52, 3.536, 3.552, 3.568, 3.584, 3.6, 3.616, 3.632, 3.648, 3.664, 3.68, 3.696, 3.712, 3.728, 3.744, 3.76, 3.776, 3.792, 3.808, 3.824, 3.84, 3.856, 3.872, 3.888, 3.904, 3.92, 3.936, 3.952, 3.968, 3.984, 4.}

```
(*Corriente LOCAL*)
EnVal = 2;
NEn = 1. + (EnVal - EnMin) / DEn;

IVal[[ NEn ]] // MatrixForm;
                 forma de matriz

ILocalY = Table[    Table[   - IVal[[NEn, il,  (jl - 1) * NZigzag + Mod[il - 1, NZigzag, 1] ]] +
           tabla       tabla                                    operación módulo
      IVal[[NEn, il,  (jl - 1) * NZigzag + Mod[il + 1, NZigzag, 1]  ]]   ,
                                            operación módulo
     {il, 1 + (jl - 1) * NZigzag, jl * NZigzag, 1}]   , {jl, 1, NArmchair, 1}];
ILocalY = Flatten[ ILocalY];
           aplana

ILocalX =  Join[   Table[  IVal[[NEn, il, il + NZigzag ]]   ,
            junta   tabla
     {il, 1, NZigzag * (NArmchair - 1), 1} ] , Table[ 0., {jl, 1, NZigzag, 1}]   ];
                                                tabla


ILocal =
  Table[ {    { 1. + IntegerPart[  (i - 1) / (NZigzag ) ], Mod[ i, NZigzag, 1.]   } ,
  tabla              parte entera                          operación módulo
     {ILocalX[[i]],  ILocalY[[i]] }  }, {i, 1, NEdos, 1.}];


ListVectorPlot[   ILocal   , ImageSize → Large, VectorColorFunction → "Rainbow",
representación vectorial de lista  tamaño de i···  grande  función de color de vector
 VectorPoints → 20, VectorScale → 0.08, PlotLegends → Automatic]
                    escala de vector     leyendas de rep··· automático
ListStreamPlot[   ILocal   , ImageSize → Large, VectorColorFunction → "Rainbow",
                             tamaño de i···  grande  función de color de vector
 VectorPoints → 20, VectorScale → 0.08, PlotLegends → Automatic]
                    escala de vector     leyendas de rep··· automático
```


0
5
10
15
20
0.6
0.4
0.2


0
5
10
15
20
0
5
10
15
20
0
10
20
30