Modelo Bose - Hubbard en la Aproximación de Campo Medio

Novoa Gastaldi Alejandro Silvestre

(silvestre.novoa@ciencias.unam.mx)

03/2019

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

Hamiltoniano de Bose-Hubbard con μ/U =0.5 para una base de 4 elementos.

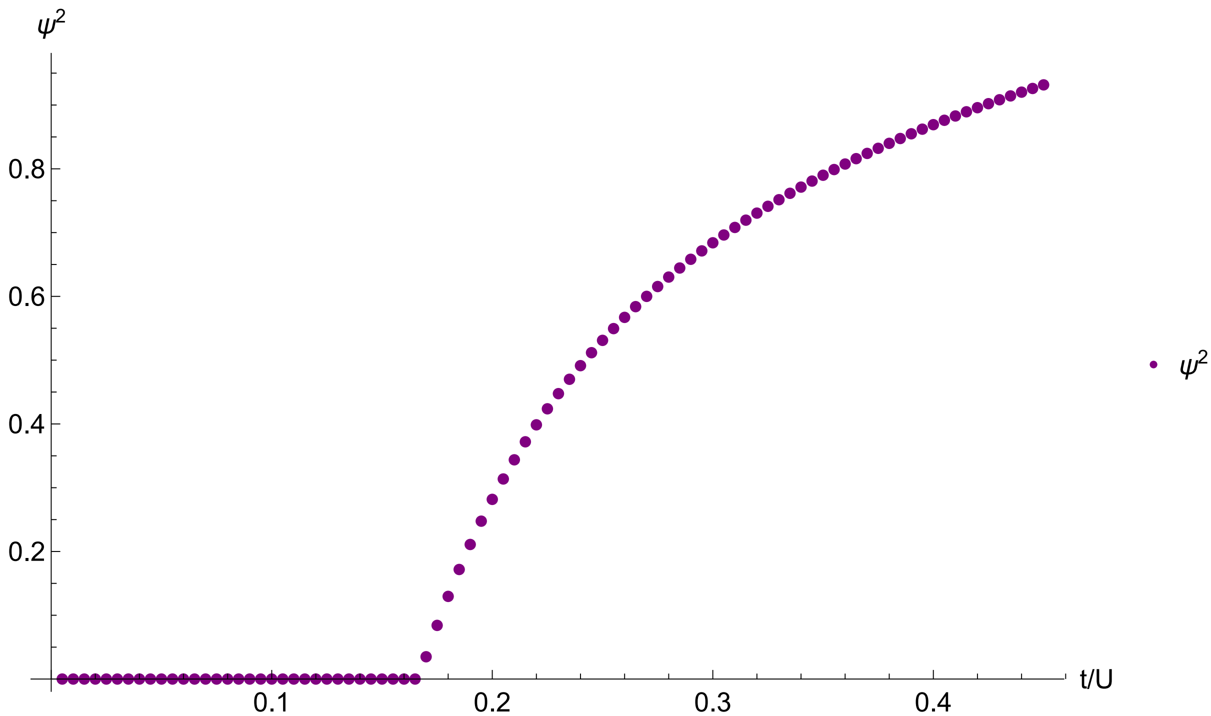
```
Clear["Global`*"]
```
borra

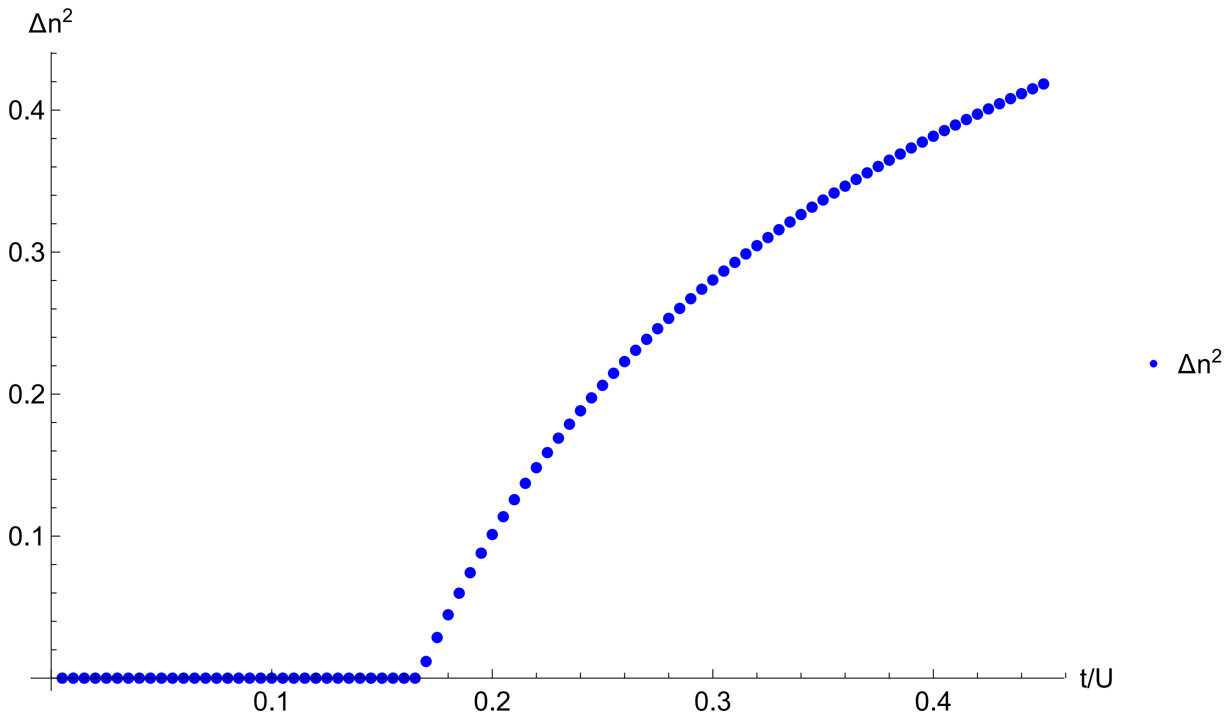
```
nDimensiones = 4.;

(*Parametros Iniciales*)
U = 1.;
Z = 1.;

ψ00 = 0.1;
t0 = 0.005;
μ0 = 0.5;

(*Error de iteración*)
ErrorMax = 0.00001;
NPuntos = 90.;
```

```
(*Parametros Variables*)
             variables
ψ = Table[ψ00, {i, 1, NPuntos}];
    tabla
t = Table[t0 * i, {i, 1, NPuntos}];
    tabla
μ = Table[μ0 * i, {i, 1, NPuntos}];
    tabla
ψ0 = Table[ψ00, {i, 1, NPuntos}];
     tabla
Dn2 = Table[0, {i, 1, NPuntos}];
      tabla



(*Funciones para la construcción del Hamiltoniano*)
FDiagonal[i_, ψ_, t_, U_, μ_, Z_] := Module[{i0 = i, ψ0 = ψ, t0 = t, U0 = U, μ0 = μ, Z0 = Z},
                                     módulo
   U0 * (i0 - 1.) * i0 / 2. - μ0 * i0 + 1. * Z0 * t0 * ψ0^2];
FNoDiagonal[i_, ψ_, t_, Z_] := Module[{i0 = i, ψ0 = ψ, t0 = t, Z0 = Z},
                               módulo
   1. * Z0 * t0 * ψ0 * Sqrt[i0] ];
                       raíz cuadrada



(*Lo siguiente se realiza en un do que varia el parametro t*)
(*Se contruye el Hamiltoniano para el primer caso con ψinicial. Se
  calcula el estado base. Se calcula respecto al edoBase ψ0=<b>,
con b el operador de aniquilación. Se puede estimar un error
 como   Abs[ ψ0-ψ ] tal que si es mayor al ErrorMax,
        valor absoluto
se define ψ=ψ0 y se itera lo anterior en un While  *)
                                            mientras
(*Una vez que termine cada iteración del while, con el ultimo estado base calculado,
se estima el parametro de fluctuación de particulas <n^2>-<n>^2 *)
Do[  {
repite
  H = Table[Table[0, {i, 1, nDimensiones}], {j, 1, nDimensiones}];
      tabla tabla
  Do[H[[i, i + 1]] = FNoDiagonal[ i, ψ[[ln]], t[[ln]], Z ], {i, 1, nDimensiones - 1, 1}];
  repite
  H = H + ConjugateTranspose[H];
          transpuesto conjugado
  Do[H[[i, i]] = -FDiagonal[ i - 1, ψ[[ln]], t[[ln]], U, μ[[1]], Z ],
  repite
   {i, 1, nDimensiones, 1}];

  EdoBase =
   Flatten[ Eigensystem[H, 1, Method → {"Arnoldi", "Criteria" → "RealPart"}][[2]] ];
   aplana    autovalores y autovec⋯  método
```

```
 aplana      autovalores y autovec··· método

  ψ0[[ln]] =
   Sum[Sqrt[i] * EdoBase[[i]] * Conjugate[EdoBase[[i + 1]]], {i, 1, nDimensiones - 1, 1}];
   s···  raíz cuadrada                     conjugado

  While[     Abs[ ψ0[[ln]] - ψ[[ln]] ] > ErrorMax,
  mientras   valor absoluto
   ψ[[ln]] = ψ0[[ln]];

   H = Table[Table[0, {i, 1, nDimensiones}], {j, 1, nDimensiones}];
       tabla  tabla
   Do[H[[i, i + 1]] = FNoDiagonal[ i, ψ[[ln]], t[[ln]], Z ], {i, 1, nDimensiones - 1, 1}];
   repite
   H = H + ConjugateTranspose[H];
           transpuesto conjugado
   Do[H[[i, i]] = -FDiagonal[ i - 1, ψ[[ln]], t[[ln]], U, μ[[1]], Z ],
   repite
    {i, 1, nDimensiones, 1}];

   EdoBase =
    Flatten[ Eigensystem[H, 1, Method → {"Arnoldi", "Criteria" → "RealPart"}][[2]] ];
    aplana    autovalores y autovec··· método

   ψ0[[ln]] = Sum[Sqrt[i] * EdoBase[[i]] * Conjugate[EdoBase[[i + 1]]],
              s···  raíz cuadrada                 conjugado
     {i, 1, nDimensiones - 1, 1}]     ],
  Dn2[[ln]] = Sum[i^2 * EdoBase[[i + 1]] * Conjugate[EdoBase[[i + 1]]],
              suma                         conjugado
     {i, 1, nDimensiones - 1, 1}] - Sum[i * EdoBase[[i + 1]] * Conjugate[EdoBase[[i + 1]]],
                                    suma                        conjugado
      {i, 1, nDimensiones - 1, 1}]^2      } ,
            {ln, 1, NPuntos, 1}]


(*Se contruyen los conjuntos de puntos (t,ψ^2) y (t,Dn2) que se graficaran *)
ψ2 = Table[{t[[i]], (ψ0[[i]])^2}, {i, 1, NPuntos}];
     tabla
DnCuadrado = Table[{t[[i]], Dn2[[i]]}, {i, 1, NPuntos}];
             tabla


ListPlot[ψ2, ImageSize → Large, AxesStyle → {Black, Black},
representación··· tamaño de i···  grande  estilo de ejes  negro  negro
 PlotStyle → {Thick, Purple}, PlotLegends → {"ψ²"} , ImageSize → Large,
 estilo de repre··· grueso  púrpura  leyendas de representación  tamaño de i···  grande
 LabelStyle → {14, Black}, AxesLabel → {"t/U" , "ψ²"}]
 estilo de etiqueta    negro      etiqueta de ejes
ListPlot[DnCuadrado, ImageSize → Large, AxesStyle → {Black, Black},
                     tamaño de i···  grande  estilo de ejes  negro  negro
```

```
                    tamaño de i···   grande  estilo de ejes     negro    negro
PlotStyle → {Thick, Blue}, PlotLegends → {"Δn²"} , ImageSize → Large,
                 grueso   azul     leyendas de representación    tamaño de i···  grande
LabelStyle → {14, Black}, AxesLabel → {"t/U" , "Δn²"}]
estilo de etiqueta        negro      etiqueta de ejes
```





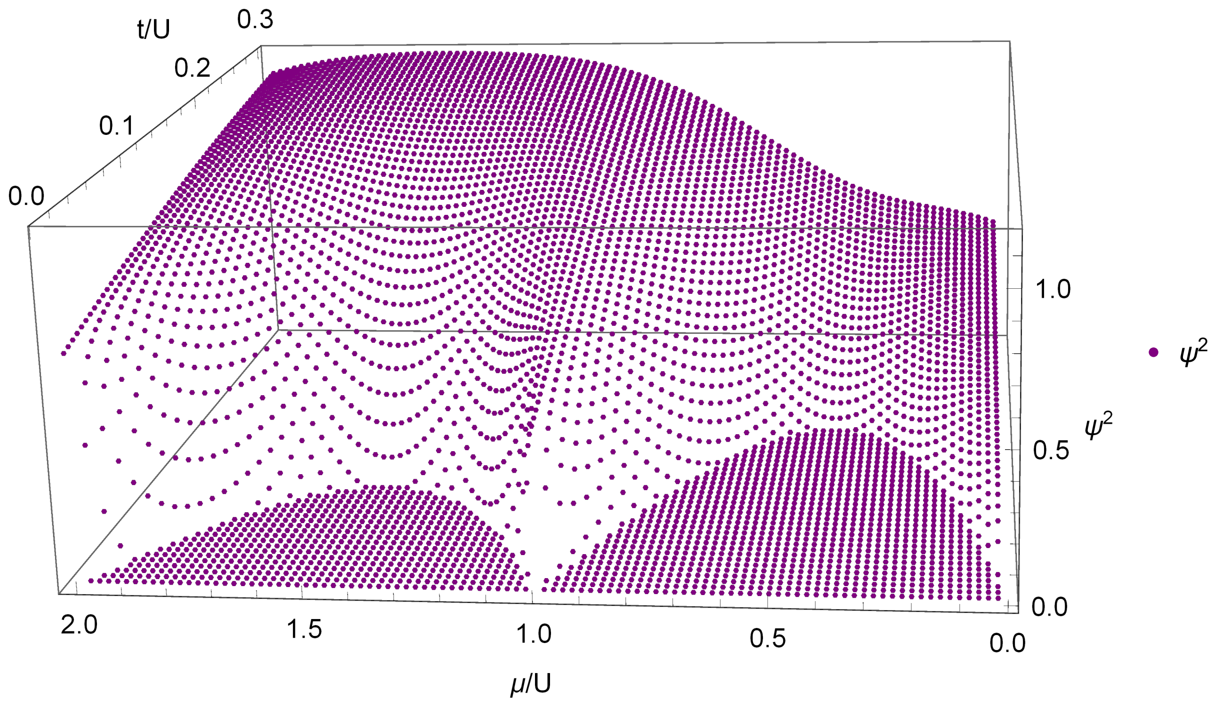
Hamiltoniano de Bose-Hubbard en función de t y μ para una base de 4 elementos.

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

nDimensiones = 4.;

(*Parametros Iniciales*)
U = 1.;
Z = 1.;
```

```
ψ00 = 0.1;
t0 = 0.005;
μ0 = 0.02;

(*Error de iteración*)
ErrorMax = 0.00001;
tNPuntos = 60.;
μNPuntos = 100.;

(*Parametros Variables*)
                 variables
ψ = Table[Table[ψ00, {i, 1, μNPuntos}], {j, 1, tNPuntos}];
    tabla  tabla
t = Table[t0 * i, {i, 1, tNPuntos}];
    tabla
μ = Table[μ0 * i, {i, 1, μNPuntos}];
    tabla
ψ0 = Table[Table[ψ00, {i, 1, μNPuntos}], {j, 1, tNPuntos}];
     tabla  tabla
Dn2 = Table[Table[0, {i, 1, μNPuntos}], {j, 1, tNPuntos}];
      tabla  tabla



(*Funciones para la construcción del Hamiltoniano*)
FDiagonal[i_, ψ_, t_, U_, μ_, Z_] := Module[{i0 = i, ψ0 = ψ, t0 = t, U0 = U, μ0 = μ, Z0 = Z},
                                     módulo
   U0 * (i0 - 1.) * i0 / 2. - μ0 * i0 + 1. * Z0 * t0 * ψ0^2];
FNoDiagonal[i_, ψ_, t_, Z_] := Module[{i0 = i, ψ0 = ψ, t0 = t, Z0 = Z},
                               módulo
   1. * Z0 * t0 * ψ0 * Sqrt[i0] ];
                       raíz cuadrada


(* Se extiende el Do para las variaciones de μ *)
                      repite
Do[
repite
 (*Lo siguiente se realiza en un do que varia el parametro t*)
 (*Se contruye el Hamiltoniano para el primer caso con ψinicial. Se
   calcula el estado base. Se calcula respecto al edoBase ψ0=<b>,
 con b el operador de aniquilación. Se puede estimar un error
  como   Abs[ ψ0-ψ ] tal que si es mayor al ErrorMax,
         valor absoluto
 se define ψ=ψ0 y se itera lo anterior en un While  *)
                                             mientras
 (*Una vez que termine cada iteración del while, con el ultimo estado base calculado,
 se estima el parametro de fluctuación de particulas <n^2>-<n>^2 *)
 Do[  {
 repite
```

```
repite
H = Table[Table[0, {i, 1, nDimensiones}], {j, 1, nDimensiones}];
    tabla   tabla
Do[H[[i, i + 1]] = FNoDiagonal[ i, ψ[[ln, kn]], t[[ln]], Z ],
repite
  {i, 1, nDimensiones - 1, 1}];
H = H + ConjugateTranspose[H];
        transpuesto conjugado
Do[H[[i, i]] = -FDiagonal[ i - 1, ψ[[ln, kn]], t[[ln]], U, μ[[kn]], Z ],
repite
  {i, 1, nDimensiones, 1}];

EdoBase =
  Flatten[ Eigensystem[H, 1, Method → {"Arnoldi", "Criteria" → "RealPart"}][[2]] ];
  aplana       autovalores y autovec··· método

ψ0[[ln, kn]] =
  Sum[Sqrt[i] * EdoBase[[i]] * Conjugate[EdoBase[[i + 1]]], {i, 1, nDimensiones - 1, 1}];
  s···  raíz cuadrada                conjugado

While[     Abs[ ψ0[[ln, kn]] - ψ[[ln, kn]] ] > ErrorMax,
mientras   valor absoluto
  ψ[[ln, kn]] = ψ0[[ln, kn]];

  H = Table[Table[0, {i, 1, nDimensiones}], {j, 1, nDimensiones}];
      tabla   tabla
  Do[
  repite
    H[[i, i + 1]] = FNoDiagonal[ i, ψ[[ln, kn]], t[[ln]], Z ], {i, 1, nDimensiones - 1, 1}];
  H = H + ConjugateTranspose[H];
          transpuesto conjugado
  Do[H[[i, i]] = -FDiagonal[ i - 1, ψ[[ln, kn]], t[[ln]], U, μ[[kn]], Z ],
  repite
    {i, 1, nDimensiones, 1}];

  EdoBase =
    Flatten[ Eigensystem[H, 1, Method → {"Arnoldi", "Criteria" → "RealPart"}][[2]] ];
    aplana       autovalores y autovec··· método

  ψ0[[ln, kn]] = Sum[Sqrt[i] * EdoBase[[i]] * Conjugate[EdoBase[[i + 1]]],
                 s···  raíz cuadrada                conjugado
    {i, 1, nDimensiones - 1, 1}]      ],
Dn2[[ln, kn]] = Sum[i^2 * EdoBase[[i + 1]] * Conjugate[EdoBase[[i + 1]]],
                suma                        conjugado
    {i, 1, nDimensiones - 1, 1}] - Sum[i * EdoBase[[i + 1]] * Conjugate[EdoBase[[i + 1]]],
                                   suma                        conjugado
     {i, 1, nDimensiones - 1, 1}]^2      } ,
               {ln, 1, tNPuntos, 1}],
                        {kn, 1, μNPuntos, 1}]
```

```
(*Se contruyen los conjuntos de puntos (t,μ,ψ^2) y (t,μ,Dn2) que se graficaran *)
ψ2 = Flatten[Table[
     aplana    tabla
    Table[{t[[i]], μ[[kn]], (ψ0[[i, kn]])^2}, {i, 1, tNPuntos}], {kn, 1, μNPuntos}], 1];
    tabla
DnCuadrado = Flatten[Table[Table[{t[[i]], μ[[kn]], Dn2[[i, kn]]}, {i, 1, tNPuntos}],
             aplana  tabla tabla
    {kn, 1, μNPuntos}], 1];


ListPointPlot3D[ψ2, PlotStyle → {Thick, Purple}, PlotLegends → {"ψ²"} ,
representación 3D en pu··  estilo de repre···  grueso  púrpura  leyendas de representación
 ImageSize → Large, LabelStyle → {14, Black}, AxesLabel → {"t/U" , "μ/U", "ψ²"}]
 tamaño de i···  grande  estilo de etiqueta  negro  etiqueta de ejes
ListPointPlot3D[DnCuadrado, PlotStyle → {Thick, Blue}, PlotLegends → {"Δn²"} ,
                            estilo de repre···  grueso  azul  leyendas de representación
 ImageSize → Large, LabelStyle → {14, Black}, AxesLabel → {"t/U" , "μ/U", "Δn²"}]
             grande  estilo de etiqueta  negro  etiqueta de ejes


(*Diagrama de Fase*)
Print[ Style["Diagrama de Fase", 18, Bold, Blue]]
       estilo                        negrita azul
ListDensityPlot[ψ2, ColorFunction → "LakeColors",
                    función de color
 FrameLabel → {"t/U" , "μ/U"}, LabelStyle → {12.5, Black}]
                               estilo de etiqueta  negro

ListContourPlot[ψ2, FrameLabel → {"t/U" , "μ/U"}, LabelStyle → {12.5, Black}]
                    etiqueta de marco             estilo de etiqueta  negro
```



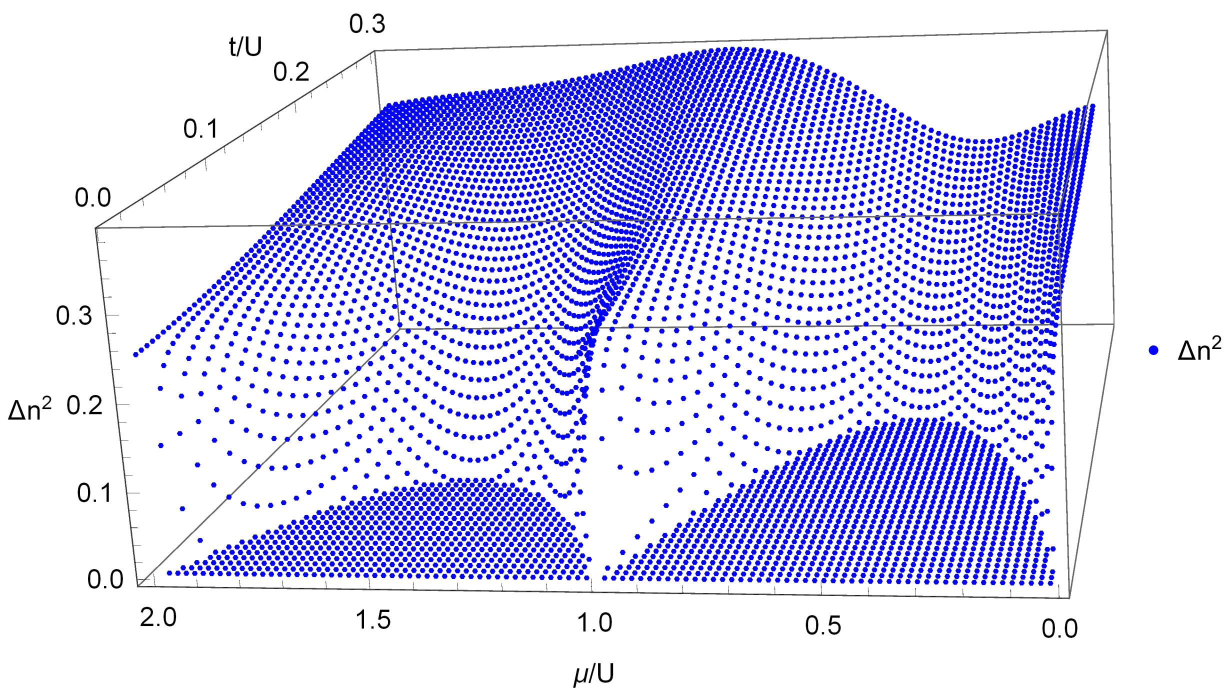
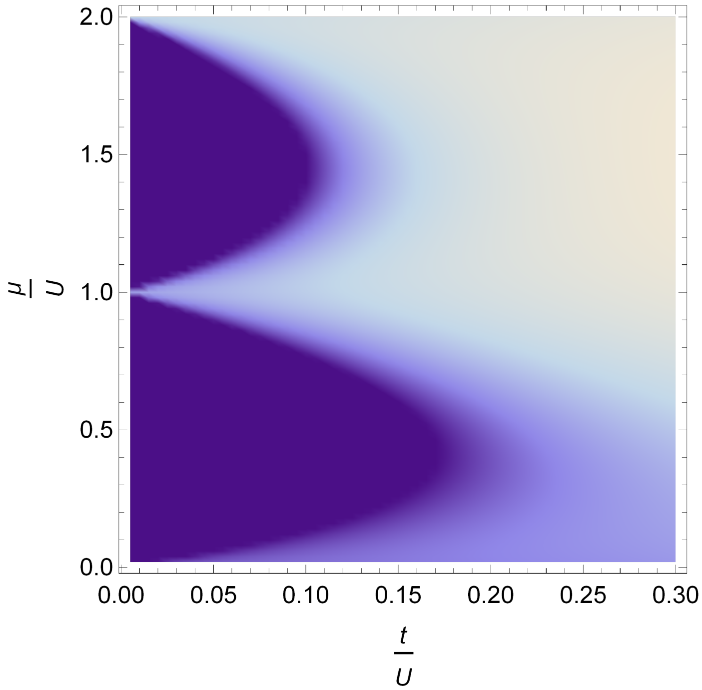
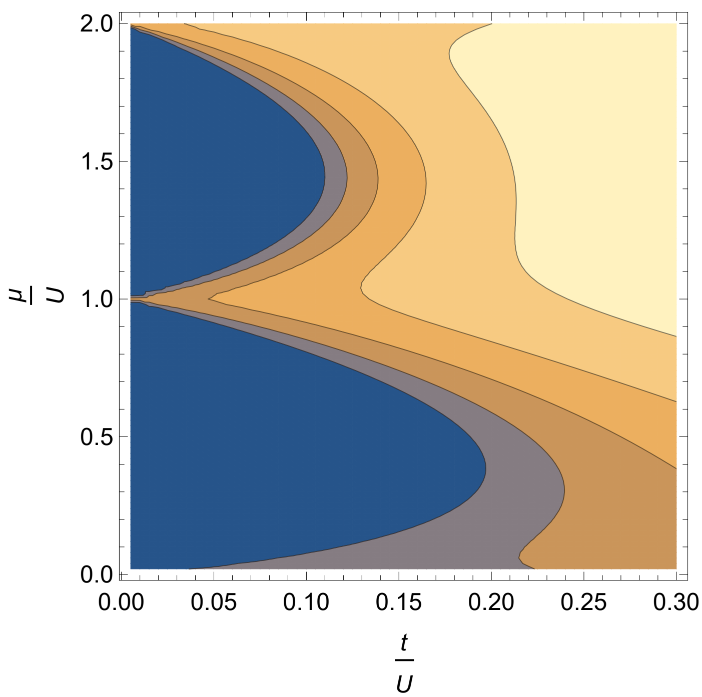


Diagrama de Fase



2.0
1.5
1.0
0.5
0.0
μ/U
0.00
0.05
0.10
0.15
0.20
0.25
0.30
t/U



2.0
1.5
1.0
0.5
0.0
μ/U
0.00
0.05
0.10
0.15
0.20
0.25
0.30
t/U

Hamiltoniano de Bose-Hubbard en función de t y μ para una base de 5 elementos.

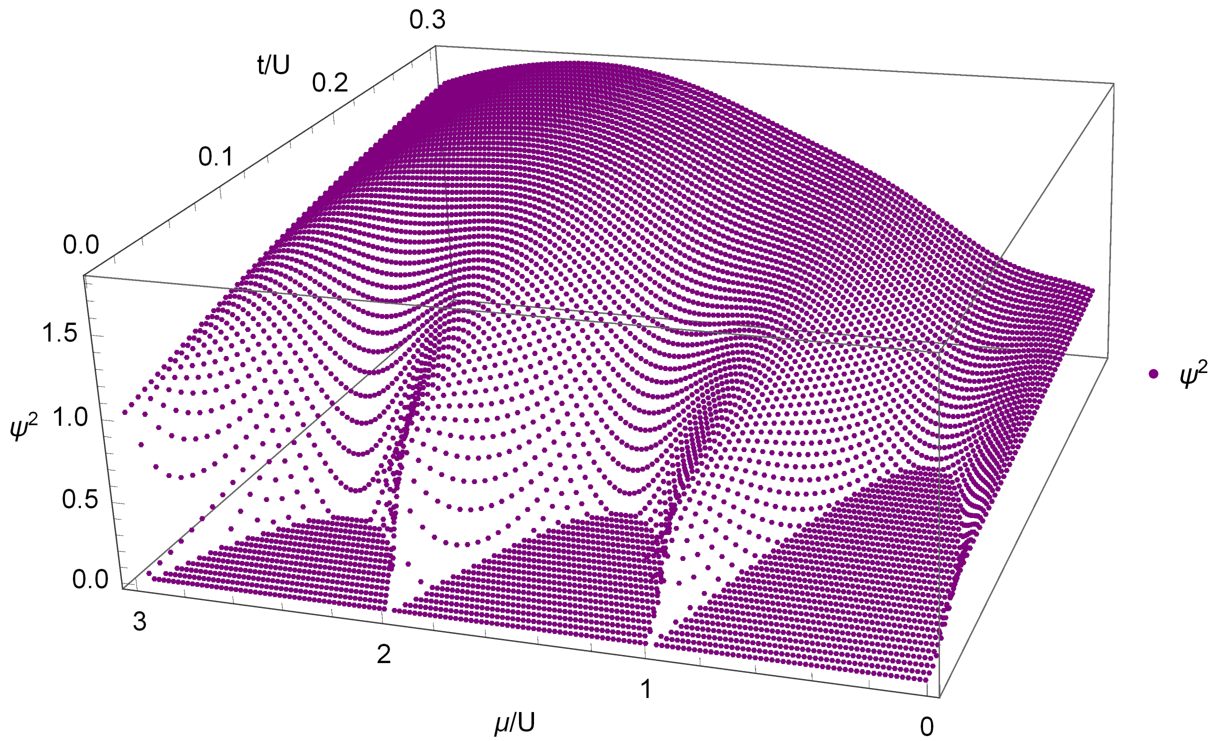
```
Clear["Global`*"]
borra
```

```
nDimensiones = 5.;

(*Parametros Iniciales*)
U = 1.;
Z = 1.;

ψ00 = 0.1;
t0 = 0.005;
μ0 = 0.02;

(*Error de iteración*)
ErrorMax = 0.00001;
tNPuntos = 60.;
μNPuntos = 150.;

(*Parametros Variables*)
                    variables
ψ = Table[Table[ψ00, {i, 1, μNPuntos}], {j, 1, tNPuntos}];
    tabla  tabla
t = Table[t0 * i, {i, 1, tNPuntos}];
    tabla
μ = Table[μ0 * i, {i, 1, μNPuntos}];
    tabla
ψ0 = Table[Table[ψ00, {i, 1, μNPuntos}], {j, 1, tNPuntos}];
     tabla  tabla
Dn2 = Table[Table[0, {i, 1, μNPuntos}], {j, 1, tNPuntos}];
      tabla  tabla
```

```
(*Funciones para la construcción del Hamiltoniano*)
FDiagonal[i_, ψ_, t_, U_, μ_, Z_] := Module[{i0 = i, ψ0 = ψ, t0 = t, U0 = U, μ0 = μ, Z0 = Z},
                                     módulo
   U0 * (i0 - 1.) * i0 / 2. - μ0 * i0 + 1. * Z0 * t0 * ψ0^2];
FNoDiagonal[i_, ψ_, t_, Z_] := Module[{i0 = i, ψ0 = ψ, t0 = t, Z0 = Z},
                               módulo
   1. * Z0 * t0 * ψ0 * Sqrt[i0] ];
                       raíz cuadrada
```

```
(* Se extiende el Do para las variaciones de μ *)
                     repite
Do[
repite
 (*Lo siguiente se realiza en un do que varia el parametro t*)
 (*Se contruye el Hamiltoniano para el primer caso con ψinicial. Se
   calcula el estado base. Se calcula respecto al edoBase ψ0=<b>,
 con b el operador de aniquilación. Se puede estimar un error
  como   Abs[ ψ0-ψ ] tal que si es mayor al ErrorMax,
         valor absoluto
 se define ψ=ψ0 y se itera lo anterior en un While  *)
                                              mientras
 (*Una vez que termine cada iteración del while, con el ultimo estado base calculado,
 se estima el parametro de fluctuación de particulas <n^2>-<n>^2 *)
 Do[  {
 repite
   H = Table[Table[0, {i, 1, nDimensiones}], {j, 1, nDimensiones}];
       tabla tabla
   Do[H[[i, i + 1]] = FNoDiagonal[ i, ψ[[ln, kn]], t[[ln]], Z ],
   repite
    {i, 1, nDimensiones - 1, 1}];
   H = H + ConjugateTranspose[H];
           transpuesto conjugado
   Do[H[[i, i]] = - FDiagonal[ i - 1, ψ[[ln, kn]], t[[ln]], U, μ[[kn]], Z ],
   repite
    {i, 1, nDimensiones, 1}];

   EdoBase =
    Flatten[ Eigensystem[H, 1, Method → {"Arnoldi", "Criteria" → "RealPart"}][[2]] ];
    aplana   autovalores y autovec··· método

   ψ0[[ln, kn]] =
    Sum[Sqrt[i] * EdoBase[[i]] * Conjugate[EdoBase[[i + 1]]], {i, 1, nDimensiones - 1, 1}];
    s···  raíz cuadrada              conjugado

   While[    Abs[ ψ0[[ln, kn]] - ψ[[ln, kn]] ] > ErrorMax,
   mientras  valor absoluto
    ψ[[ln, kn]] = ψ0[[ln, kn]];

    H = Table[Table[0, {i, 1, nDimensiones}], {j, 1, nDimensiones}];
        tabla tabla
    Do[
    repite
     H[[i, i + 1]] = FNoDiagonal[ i, ψ[[ln, kn]], t[[ln]], Z ], {i, 1, nDimensiones - 1, 1}];
    H = H + ConjugateTranspose[H];
            transpuesto conjugado
    Do[H[[i, i]] = - FDiagonal[ i - 1, ψ[[ln, kn]], t[[ln]], U, μ[[kn]], Z ],
    repite
     {i, 1, nDimensiones, 1}];

    EdoBase =
```

```
Flatten[ Eigensystem[H, 1, Method → {"Arnoldi", "Criteria" → "RealPart"}][[2]] ];
aplana   autovalores y autovec··· método

ψ0[[ln, kn]] = Sum[Sqrt[i] * EdoBase[[i]] * Conjugate[EdoBase[[i + 1]]],
               s···  raíz cuadrada                 conjugado
  {i, 1, nDimensiones - 1, 1}]    ],
Dn2[[ln, kn]] = Sum[i^2 * EdoBase[[i + 1]] * Conjugate[EdoBase[[i + 1]]],
                suma                          conjugado
  {i, 1, nDimensiones - 1, 1}] - Sum[i * EdoBase[[i + 1]] * Conjugate[EdoBase[[i + 1]]],
                                 suma                        conjugado
   {i, 1, nDimensiones - 1, 1}]^2    },
          {ln, 1, tNPuntos, 1}],
                 {kn, 1, μNPuntos, 1}]


(*Se contruyen los conjuntos de puntos (t,μ,ψ^2) y (t,μ,Dn2) que se graficaran *)
ψ2 = Flatten[Table[
    aplana   tabla
  Table[{t[[i]], μ[[kn]], (ψ0[[i, kn]])^2}, {i, 1, tNPuntos}], {kn, 1, μNPuntos}], 1];
  tabla
DnCuadrado = Flatten[Table[Table[{t[[i]], μ[[kn]], Dn2[[i, kn]]}, {i, 1, tNPuntos}],
             aplana   tabla  tabla
  {kn, 1, μNPuntos}], 1];


ListPointPlot3D[ψ2, PlotStyle → {Thick, Purple}, PlotLegends → {"ψ²"},
representación 3D en pu··  estilo de repre···  grueso  púrpura  leyendas de representación
 ImageSize → Large, LabelStyle → {14, Black}, AxesLabel → {"t/U", "μ/U", "ψ²"}]
 tamaño de i···  grande  estilo de etiqueta  negro  etiqueta de ejes
ListPointPlot3D[DnCuadrado, PlotStyle → {Thick, Blue}, PlotLegends → {"Δn²"},
                            estilo de repre···  grueso  azul  leyendas de representación
 ImageSize → Large, LabelStyle → {14, Black}, AxesLabel → {"t/U", "μ/U", "Δn²"}]
             grande  estilo de etiqueta  negro  etiqueta de ejes


(*Diagrama de Fase*)
Print[ Style["Diagrama de Fase", 18, Bold, Blue]]
       estilo                     negrita azul
ListDensityPlot[ψ2, ColorFunction → "LakeColors",
                    función de color
 FrameLabel → {"t/U", "μ/U"}, LabelStyle → {12.5, Black}]
                              estilo de etiqueta  negro

ListContourPlot[ψ2, FrameLabel → {"t/U", "μ/U"}, LabelStyle → {12.5, Black}]
                    etiqueta de marco            estilo de etiqueta  negro
```



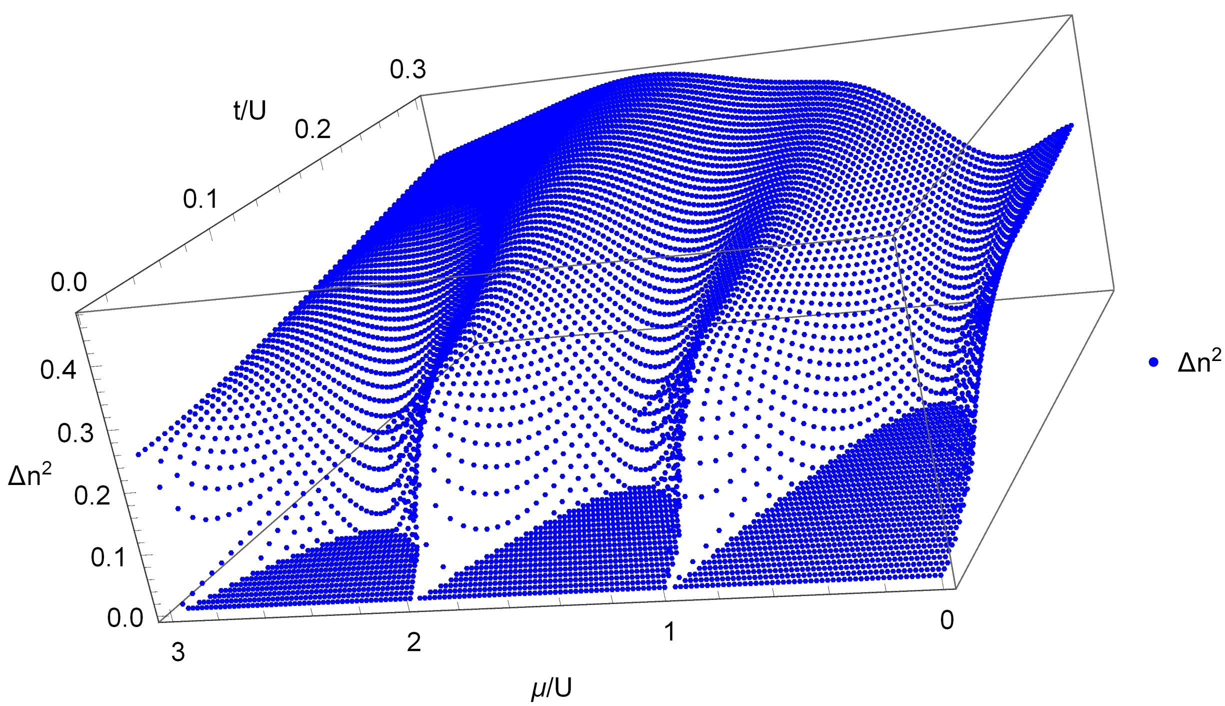
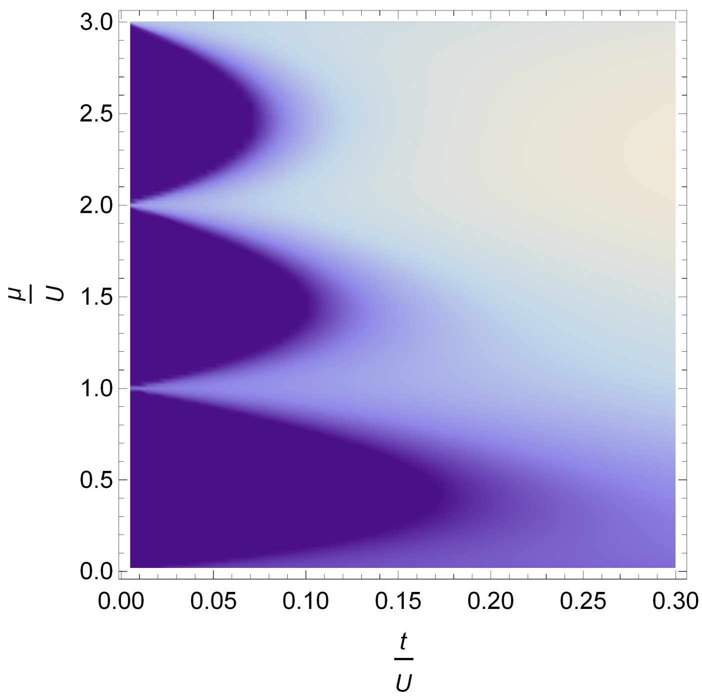
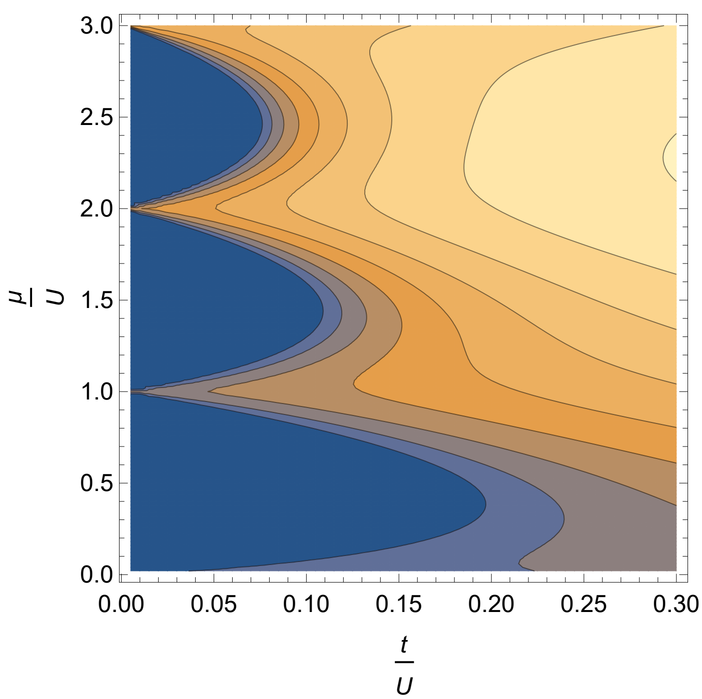


Diagrama de Fase


3.0
2.5
2.0
1.5
1.0
0.5
0.0
μ/U
0.00 0.05 0.10 0.15 0.20 0.25 0.30
t/U


3.0
2.5
2.0
1.5
1.0
0.5
0.0
μ/U
0.00 0.05 0.10 0.15 0.20 0.25 0.30
t/U