```
In[ ]:= SetDirectory@NotebookDirectory[];
        设置目录          当前笔记本的目录
        Import["Qubits_package.m"];
        导入
        Import["ExactKrylov_package.m"];
        导入
```

Parameters

```
In[ ]:= Nq = 10;
        (*HamType="Heisenberg";*)
        HamType = "FermiHubbard";
        GraphType = ToString[2];
                    转换为字符串

In[ ]:= Ntrials = 20;
        logηList = Table[-0.1 * j, {j, -50, 150}];
                   表格
        PR = {{1.*^-3, 1.*^12}, {1.*^-14, 1.*^1}};

In[ ]:= seed = RandomInteger[{1, 1000000}];
              伪随机整数
        (*seed=123; *)
        SeedRandom[seed];
        随机种子
```

Run

```
In[ ]:= graphs = {};
        plotsRTE = {};
        plotsF = {};
        plots = {};
        γ2ϵK = {};

In[ ]:= trial = 0;
        While[trial < Ntrials, (
        While循环

          Label[begin];
          标签


          If[StringMatchQ[HamType, "Heisenberg"], (
          ··· 字符串匹配判定
            EL = funGraph[Nq, GraphType];
            Model = funHeisenberg[Nq, EL]
           )];
          If[StringMatchQ[HamType, "FermiHubbard"], (
          ··· 字符串匹配判定
            EL = funGraph[Nq / 2, GraphType];
            u = 1.;
            Model = funFermiHubbard[Nq, EL, u]
           )];
```

```
Ham = funHamiltonianQubit[Model];

{EE, ES} = funSpectrum[Ham];
HamNorm = Max[Abs[EE]];
          (…  绝对值
EE = EE / HamNorm;
Eg = EE[[1]];
If[StringMatchQ[HamType, "Heisenberg"], (
(… 字符串匹配判定
  htot = 3. * Length[EL] / HamNorm
              长度
 )];
If[StringMatchQ[HamType, "FermiHubbard"], (
(… 字符串匹配判定
  htot = (2. * Length[EL] + u / 4. * Nq / 2.) / HamNorm
              长度
 )];

If[StringMatchQ[HamType, "Heisenberg"], (
(… 字符串匹配判定
  ψ = funPairwiseSinglet[Nq]
 )];
If[StringMatchQ[HamType, "FermiHubbard"], (
(… 字符串匹配判定
  ψ = funHartreeFock[Nq, EL]
 )];
ψ = Conjugate[ES].ψ;
    共轭
ψ = Flatten[ψ];
    压平
Proψ = Abs[ψ]^2;
       绝对值
pg = Proψ[[1]];
ER = Total[Proψ * EE];
     总计
∈R = ER - Eg;

If[pg < 1.*^-3, (
如果
  Print[pg];
  打印
  Goto[begin];
  转到
 )];

d = RandomChoice[Table[i, {i, 2, 30}]];
    随机选择          表格
Ide = IdentityMatrix[d];
      单位矩阵

E0 = Eg + 1.;
{Hmat, Smat} = funMatPower[EE, Proψ, d, E0];
EB = Hmat[[d, d]] / Smat[[d, d]];
```

```
ϵB = EB - Eg;
logη = -15;
η = 10.^logη;
{EK, cn} = funDiagonalisation[Hmat + 2. * η * Ide, Smat + 2. * η * Ide];
ϵK = EK - Eg;
If[ϵK > 1.*^-2 || ϵK < 1.*^-9, (
如果
  Print[{d, ϵK}];
  打印
  Goto[begin];
  转到
 )];
AppendTo[graphs, Graph[EL]];
附加              图
{γList, ϵList} = funGammaEpsilon[Eg, pg, Hmat, Smat, Ide, 1., 1., logηList];
γListP = γList;
ϵListP = ϵList;
ϵMin = Min[ϵList];
       最小值
If[ϵMin < 2. * ϵK, (
如果
  γ = funInterpolation[ϵList, γList, 2. * ϵK];
 ), (
  γ = 0.;
 )];
γP = γ;

E0 = 0;
{Hmat, Smat} = funMatChebyshev[EE, Proψ, d, htot, E0];
{γList, ϵList} = funGammaEpsilon[Eg, pg, Hmat, Smat, Ide, 1., 1., logηList];
γListCP = γList;
ϵListCP = ϵList;
ϵMin = Min[ϵList];
       最小值
If[ϵMin < 2. * ϵK, (
如果
  γ = funInterpolation[ϵList, γList, 2. * ϵK];
 ), (
  γ = 0.;
 )];
γCP = γ;

E0 = Eg;
τMIN = 0;
τMAX = 64;
Do[(
Do循环
  τ = (τMIN + τMAX) / 2.;
  {Hmat, Smat} = funMatGaussianPower[EE, Proψ, 1, htot, τ, E0];
  EK = Hmat[[1, 1]] / Smat[[1, 1]];
  err = EK - Eg;
```

```
  If[err > ϵB, τMIN = τ];
  如果
  If[err < ϵB, τMAX = τ];
  如果
 ), {j, 1, 30}];
τ = (τMIN + τMAX) / 2.;
δ = RandomReal[{-0.1, 0.1}];
    伪随机实数
E0 = Eg + δ;
{Hmat, Smat} = funMatGaussianPower[EE, Proψ, d, htot, τ, E0];
{γList, ϵList} = funGammaEpsilon[Eg, pg, Hmat, Smat, Ide, htot, 1., logηList];
γListGP = γList;
ϵListGP = ϵList;
ϵMin = Min[ϵList];
       最小值
If[ϵMin < 2. * ϵK, (
如果
  γ = funInterpolation[ϵList, γList, 2. * ϵK];
 ), (
  γ = 0.;
 )];
γGP = γ;

E0 = Eg - 1.;
{Hmat, Smat} = funMatInversePower[EE, Proψ, d, E0];
{γList, ϵList} = funGammaEpsilon[Eg, pg, Hmat, Smat, Ide, 1., 1., logηList];
γListIP = γList;
ϵListIP = ϵList;
ϵMin = Min[ϵList];
       最小值
If[ϵMin < 2. * ϵK, (
如果
  γ = funInterpolation[ϵList, γList, 2. * ϵK];
 ), (
  γ = 0.;
 )];
γIP = γ;

E0 = Eg;
τMIN = 0;
τMAX = 64;
Do[(
Do循环
  τ = (τMIN + τMAX) / 2.;
  {Hmat, Smat} = funMatITE[EE, Proψ, d, τ, E0];
  EK = Hmat[[d, d]] / Smat[[d, d]];
  err = EK - Eg;
  If[err > ϵB, τMIN = τ];
  如果
  If[err < ϵB, τMAX = τ];
  如果
 ), {j, 1, 30}];
```

```
τ = (τMIN + τMAX) / 2.;
E0 = Eg;
{Hmat, Smat} = funMatITE[EE, Proψ, d, τ, E0];
{γList, ϵList} = funGammaEpsilon[Eg, pg, Hmat, Smat, Ide, 1., 1., logηList];
γListITE = γList;
ϵListITE = ϵList;
ϵMin = Min[ϵList];
       最小值
If[ϵMin < 2. * ϵK, (
如果
  γ = funInterpolation[ϵList, γList, 2. * ϵK];
 ), (
  γ = 0.;
 )];
γITE = γ;

ΔtList = Table[2. * PI / 100 j, {j, 1, 100}];
         表格

ϵKList = ΔtList;
Do[(
Do循环
  Δt = ΔtList[[j]];

  E0 = Eg;
  {Hmat, Smat} = funMatRTE[EE, Proψ, d, Δt, E0];
  logη = -15;
  η = 10.^logη;
  {EK, cn} = funDiagonalisation[Hmat + 2. * η * Ide, Smat + 2. * η * Ide];
  err = EK - Eg;

  ϵKList[[j]] = err
 ), {j, 1, Length[ΔtList]}];
           长度
Δt = ΔtList[[Position[ϵKList, Min[ϵKList]][[1, 1]]]];
             位置              最小值
AppendTo[plotsRTE, ListLogPlot[Transpose[{ΔtList, ϵKList}], PlotRange → Full]];
附加                 点集的对数图   转置                    绘制范围      全范围
E0 = Eg;
{Hmat, Smat} = funMatRTE[EE, Proψ, d, Δt, E0];
{γList, ϵList} = funGammaEpsilon[Eg, pg, Hmat, Smat, Ide, 1., 1., logηList];
γListRTE = γList;
ϵListRTE = ϵList;
ϵMin = Min[ϵList];
       最小值
If[ϵMin < 2. * ϵK, (
如果
  γ = funInterpolation[ϵList, γList, 2. * ϵK];
 ), (
  γ = 0.;
 )];
```

```
γRTE = γ;

ΔE = 0.;
E0 = Eg;
τMIN = 0;
τMAX = 64;
Do[(
Do循环
  T = (τMIN + τMAX) / 2.;
  {Hmat, Smat} = funMatFilter[EE, Proψ, d, T, E0, ΔE];
  EK = Hmat[[1, 1]] / Smat[[1, 1]];
  err = EK - Eg;
  If[err > ϵB, τMIN = T];
  如果
  If[err < ϵB, τMAX = T];
  如果
 ), {j, 1, 30}];
T = (τMIN + τMAX) / 2.;
ΔEList = Table[2. / (d * 100) * j, {j, 1, 100}];
          表格

ϵKList = ΔEList;
Do[(
Do循环
  ΔE = ΔEList[[j]];

  E0 = Eg;
  {Hmat, Smat} = funMatFilter[EE, Proψ, d, T, E0, ΔE];
  logη = -15;
  η = 10.^logη;
  {EK, cn} = funDiagonalisation[Hmat + 2. * η * Ide, Smat + 2. * η * Ide];
  err = EK - Eg;

  ϵKList[[j]] = err;
 ), {j, 1, Length[ΔEList]}];
             长度

ΔE = ΔEList[[Position[ϵKList, Min[ϵKList]][[1, 1]]]];
              位置          最小值

AppendTo[plotsF, ListLogPlot[Transpose[{ΔEList, ϵKList}], PlotRange → Full]];
附加              点集的对数图  转置                        绘制范围      全范围

E0 = Eg;
{Hmat, Smat} = funMatFilter[EE, Proψ, d, T, E0, ΔE];
{γList, ϵList} = funGammaEpsilon[Eg, pg, Hmat, Smat, Ide, 1., 1., logηList];
γListF = γList;
ϵListF = ϵList;
ϵMin = Min[ϵList];
       最小值
If[ϵMin < 2. * ϵK, (
如果
  γ = funInterpolation[ϵList, γList, 2. * ϵK];
 ), (
```

```
    γ = 0.;
   )];
  γF = γ;

  AppendTo[plots, ListLogLogPlot[
  附加              点集的双对数图
    {Transpose[{γListP, 0. * ϵListP + ϵK}], Transpose[{γListP, 0. * ϵListP + 2. * ϵK}],
     转置                                    转置
     Transpose[{γListP, ϵListP}], Transpose[{γListCP, ϵListCP}],
     转置                          转置
     Transpose[{γListGP, ϵListGP}], Transpose[{γListIP, ϵListIP}],
     转置                           转置
     Transpose[{γListITE, ϵListITE}], Transpose[{γListRTE, ϵListRTE}],
     转置                             转置
     Transpose[{γListF, ϵListF}]}, PlotRange → PR, Joined → True,
     转置                          绘制范围        连接点    真
    PlotStyle → {Cyan, Brown, Red, Yellow, Blue, Green, Orange, Purple, Magenta}]];
    绘图样式      蓝绿色 棕色  红色  黄色    蓝色   绿色   橙色    紫色     品红色

  AppendTo[γ2ϵK, {γP, γCP, γGP, γIP, γITE, γRTE, γF}];
  附加
  trial = trial + 1;
  Print[{"trial", trial, "d", d, ϵK, ToString[Now]}];
  打印                              转换为…  此刻
 )]
γ2ϵK = Transpose[γ2ϵK];
       转置
```

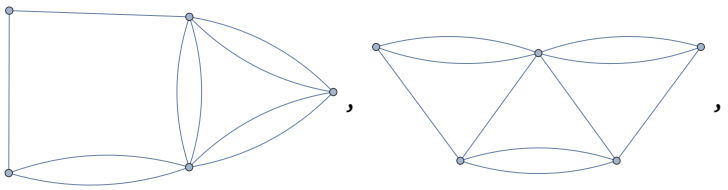
```
0.000884787

{trial, 1, d, 26, 1.33344×10^-6,
 DateObject[{2022, 12, 10, 12, 51, 44.9405384}, Instant, Gregorian, 8.]}

{trial, 2, d, 8, 0.0000190524,
 DateObject[{2022, 12, 10, 12, 51, 49.4416469}, Instant, Gregorian, 8.]}

{trial, 3, d, 5, 0.00416199,
 DateObject[{2022, 12, 10, 12, 51, 53.1100486}, Instant, Gregorian, 8.]}

{trial, 4, d, 22, 3.29341×10^-8,
 DateObject[{2022, 12, 10, 12, 52, 6.8247180}, Instant, Gregorian, 8.]}

{5, 0.0131779}

{trial, 5, d, 20, 1.60191×10^-6,
 DateObject[{2022, 12, 10, 12, 52, 21.1978405}, Instant, Gregorian, 8.]}

{trial, 6, d, 11, 0.000082384,
 DateObject[{2022, 12, 10, 12, 52, 26.6665544}, Instant, Gregorian, 8.]}

{trial, 7, d, 14, 4.94516×10^-8,
 DateObject[{2022, 12, 10, 12, 52, 33.7712455}, Instant, Gregorian, 8.]}

{2, 0.0145355}

{trial, 8, d, 17, 9.69769×10^-8,
 DateObject[{2022, 12, 10, 12, 52, 45.7116394}, Instant, Gregorian, 8.]}
```

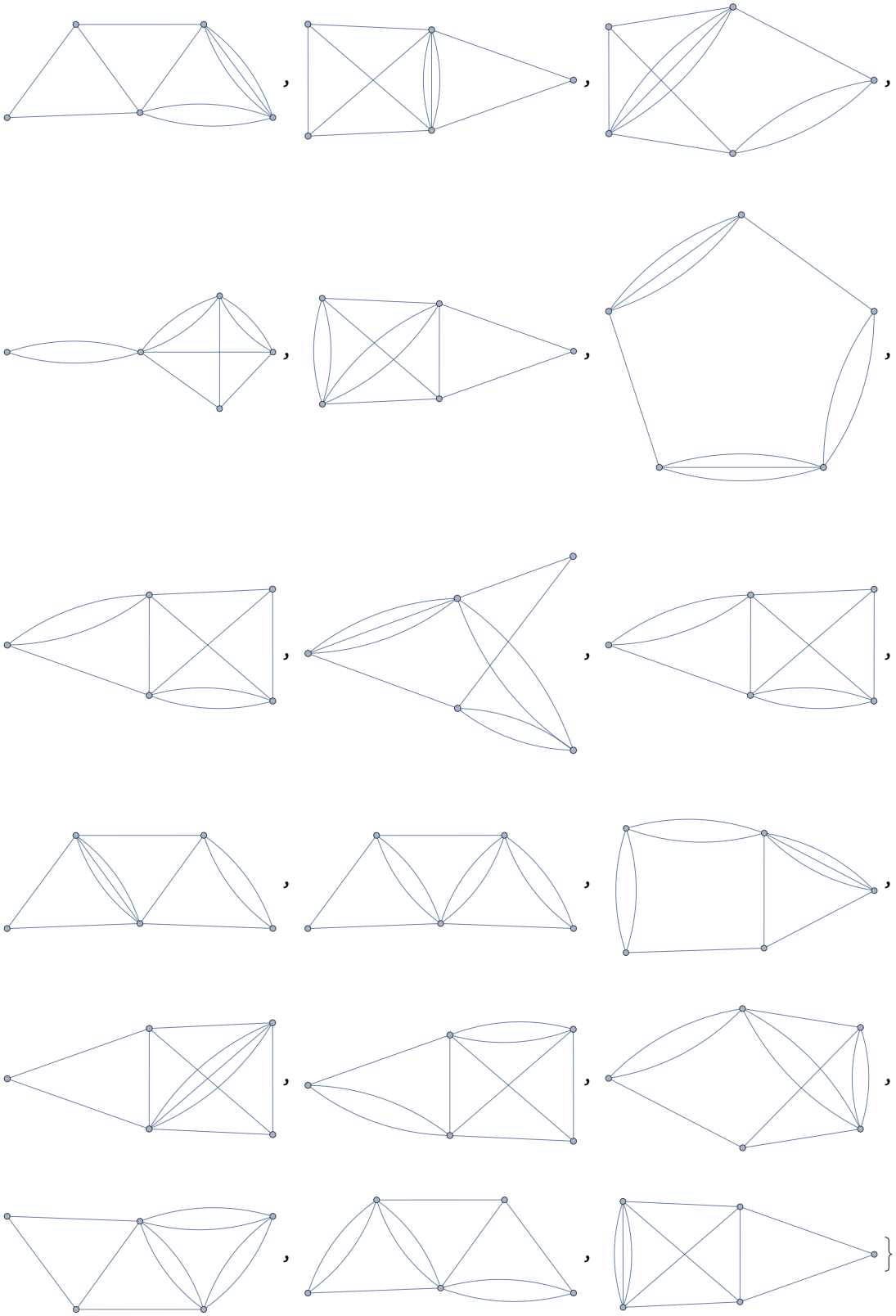
```
{trial, 9, d, 3, 0.00244131,
 DateObject[{2022, 12, 10, 12, 52, 49.2490670}, Instant, Gregorian, 8.]}

{trial, 10, d, 13, 0.000588465,
 DateObject[{2022, 12, 10, 12, 52, 55.7405939}, Instant, Gregorian, 8.]}

{trial, 11, d, 24, 0.00011042,
 DateObject[{2022, 12, 10, 12, 53, 11.7725325}, Instant, Gregorian, 8.]}

{trial, 12, d, 20, 4.3414×10^-8,
 DateObject[{2022, 12, 10, 12, 53, 23.4078201}, Instant, Gregorian, 8.]}

{trial, 13, d, 28, 7.50221×10^-6,
 DateObject[{2022, 12, 10, 12, 53, 47.0080908}, Instant, Gregorian, 8.]}

{2, 0.0241691}

{trial, 14, d, 2, 0.00327647,
 DateObject[{2022, 12, 10, 12, 53, 53.1813887}, Instant, Gregorian, 8.]}

{trial, 15, d, 9, 0.00396963,
 DateObject[{2022, 12, 10, 12, 53, 58.0045832}, Instant, Gregorian, 8.]}

0.000441786

{5, 0.0441659}

{trial, 16, d, 28, 4.34021×10^-6,
 DateObject[{2022, 12, 10, 12, 54, 27.2765105}, Instant, Gregorian, 8.]}

0.00052126

{trial, 17, d, 3, 0.00567203,
 DateObject[{2022, 12, 10, 12, 54, 33.5770070}, Instant, Gregorian, 8.]}

{trial, 18, d, 26, 7.83147×10^-8,
 DateObject[{2022, 12, 10, 12, 54, 54.2416715}, Instant, Gregorian, 8.]}

{trial, 19, d, 20, 1.21933×10^-9,
 DateObject[{2022, 12, 10, 12, 55, 5.9382695}, Instant, Gregorian, 8.]}

{24, 1.22125×10^-14}

{trial, 20, d, 8, 0.0002122,
 DateObject[{2022, 12, 10, 12, 55, 13.0602143}, Instant, Gregorian, 8.]}
```

Plots

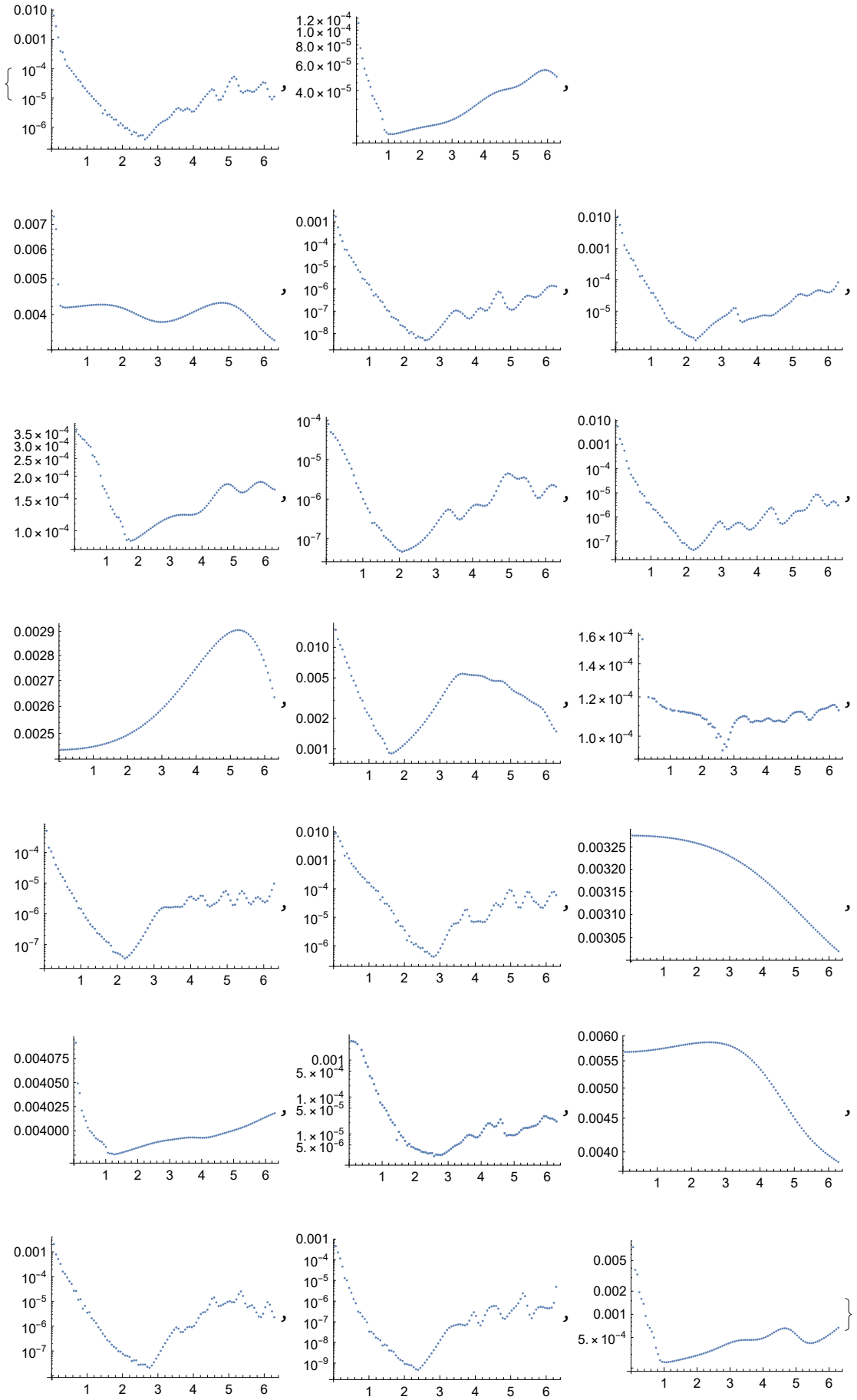
In[ ]:=
```
graphs
plotsRTE
plotsF
plots
```

Out[ ]=

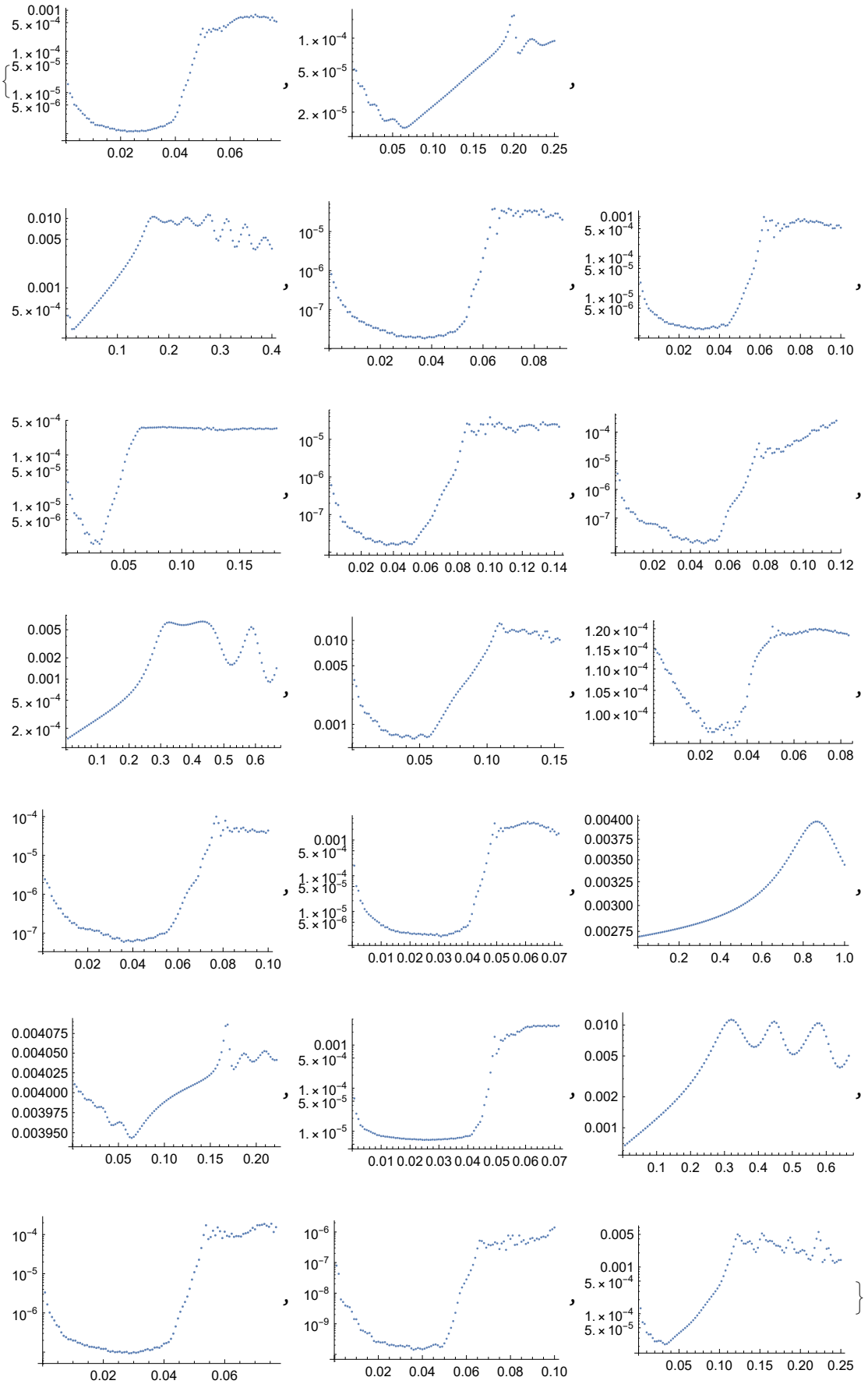




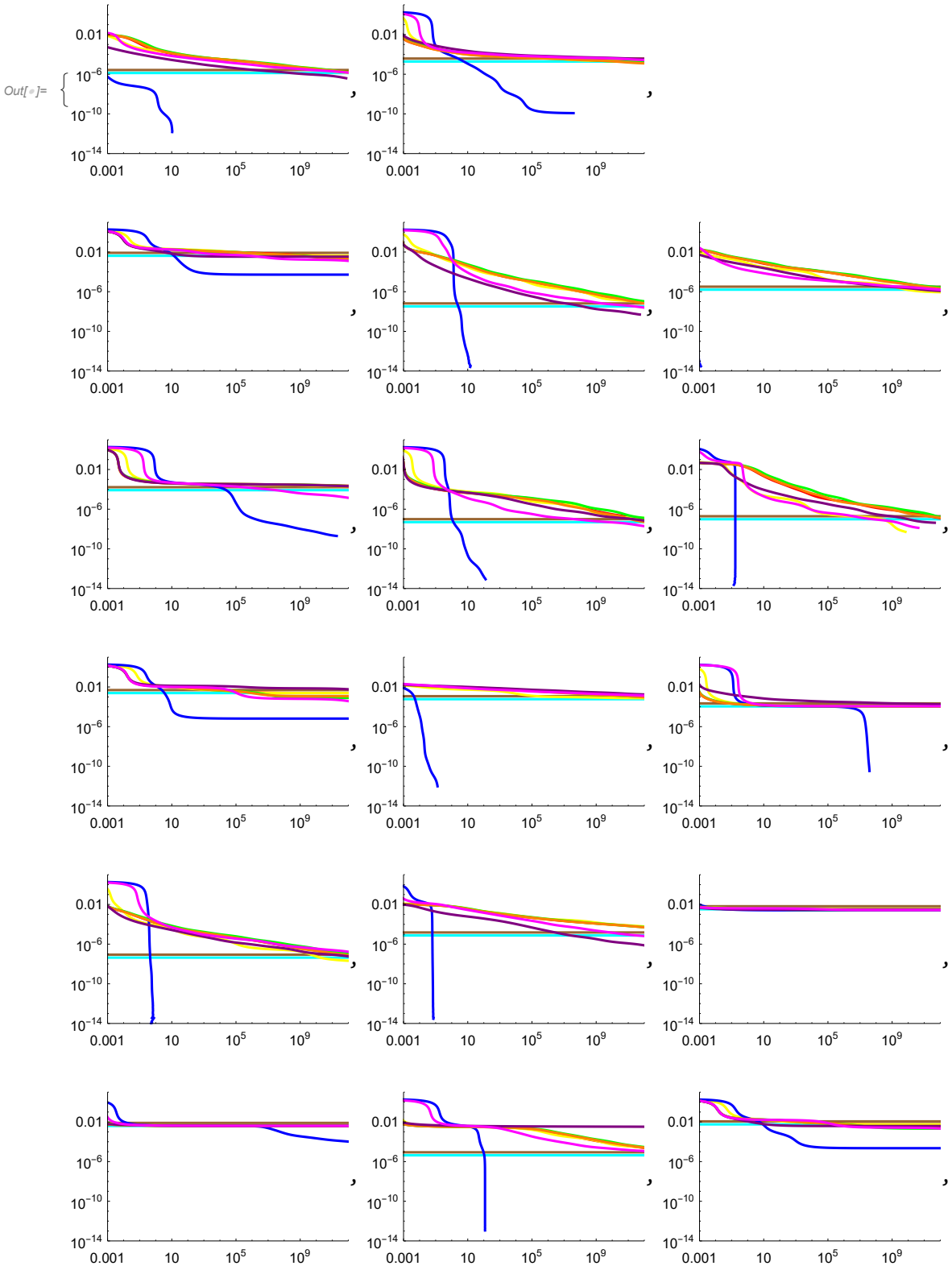
Out[ ]=

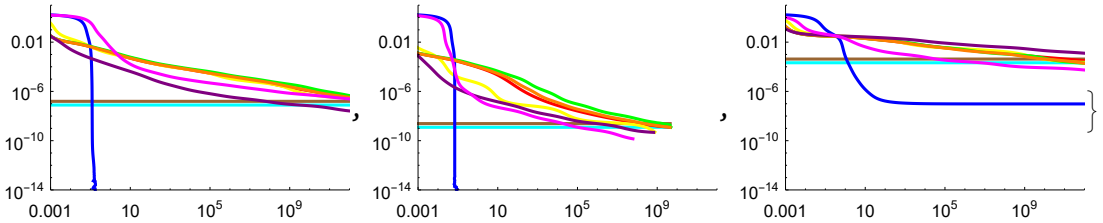


Out[•]=



Out[ ]=





Data

```
In[ ]:= If[! StringMatchQ[GraphType, "Chain"] && ! StringMatchQ[GraphType, "Ladder"],
          GraphType = "Random"];

In[ ]:= path =
          ToString[HamType] <> "-" <> ToString[GraphType] <> "-Nq=" <> ToString[Nq] <> "-5.dat";
        CreateFile[path];
        file = File[path];

In[ ]:= Data = {};
        AppendTo[Data, "γ2∈K:"];
        Data = Join[Data, γ2∈K];
        AppendTo[Data, ""];
        AppendTo[Data, "seed:"];
        AppendTo[Data, seed];
        AppendTo[Data, ""];
        Export[file, Data];
```