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

Model

In[ ]:=
```
Ham = HeisenbergHam;
```

Spectrum

In[ ]:=
```
{Λ, U} = funSpectrum[Ham];
HamNorm = Max[Abs[Λ]];
```
…　绝对值
```
Λ = Λ / HamNorm;
Eg = Λ[[1]]
```
Out[ ]=
```
-1.
```
In[ ]:=
```
htot = 27. / HamNorm
```
Out[ ]=
```
1.58524
```

Reference state

In[ ]:=
```
φ = φHeisenberg;
φ = Flatten[Conjugate[U].φ];
```
压平　共轭
```
probφ = Abs[φ]^2;
```
绝对值

In[ ]:=
```
pg = probφ[[1]] (*pg>10^-3*)
ER = Total[probφ * Λ];
```
总计
```
ϵR = ER - Eg
```
Out[ ]=
```
0.682614
```
Out[ ]=
```
0.119312
```

parameters

In[ ]:=
```
E1P = {}; (*single basis energy, d=1, P*)
E1GP = {};
E2P = {};
E2GP = {};
E3P = {};
E3GP = {};
E4P = {};
E4GP = {};
E5P = {};
E5GP = {};
```

d = 1

In[ ]:=
```
d = 1;
```

Power

In[ ]:=
```
E0 = Eg + 1.;
{Hmat, Smat} = funMatP[Λ, E0, d, probφ];
```

In[ ]:=
```
Do[
Do循环
 AppendTo[E1P, Hmat⟦i, i⟧ / Smat⟦i, i⟧];
 附加
 , {i, 1, d}]
```

In[ ]:=
```
EB = Hmat⟦d, d⟧ / Smat⟦d, d⟧;
ϵB = EB - Eg(*used to identify τ for GP*)
```

Out[ ]=
```
0.119312
```

Gaussian-Power

In[ ]:=
```
τMIN = 0;
τMAX = 64;
While[True,
 τ = (τMIN + τMAX) / 2.;
 {Hmat, Smat} = funMatGP[Λ, Eg, τ, 1, probφ, htot];
 EK = Hmat[[1, 1]] / Smat[[1, 1]];
 err = EK - Eg;
 If[err > ϵB, τMIN = τ];
 If[err < ϵB, τMAX = τ];
 If[Abs[err - ϵB] < 10^-10, Break[]];
];
τ
```

Out[ ]= 0.0000610352

In[ ]:=
```
τ > Sqrt[(d - 1) / E]
```

Out[ ]= True

In[ ]:=
```
E0 = Eg;
{Hmat, Smat} = funMatGP[Λ, E0, τ, d, probφ, htot];
```

In[ ]:=
```
Do[
 AppendTo[E1GP, Hmat[[i, i]] / Smat[[i, i]]];
 , {i, 1, d}]
```

In[ ]:=
```
Print[E1P]
Print[E1GP]
```

{-0.880688}

{-0.880688}

d = 2

In[ ]:=
```
d = 2;
```

Power

In[ ]:=
```
E0 = Eg + 1.;
{Hmat, Smat} = funMatP[Λ, E0, d, probφ];
```

In[∘]:=
```
Do[
  AppendTo[E2P, Hmat⟦i, i⟧ / Smat⟦i, i⟧];
  , {i, 1, d}]
```

In[∘]:=
```
EB = Hmat⟦d, d⟧ / Smat⟦d, d⟧;
ϵB = EB - Eg(*used to identify τ for GP*)
```

Out[∘]= 0.0479653

Gaussian-Power

In[∘]:=
```
τMIN = 0;
τMAX = 64;
While[True,
  τ = (τMIN + τMAX) / 2.;
  {Hmat, Smat} = funMatGP[Λ, Eg, τ, 1, probφ, htot];
  EK = Hmat⟦1, 1⟧ / Smat⟦1, 1⟧;
  err = EK - Eg;
  If[err > ϵB, τMIN = τ];
  If[err < ϵB, τMAX = τ];
  If[Abs[err - ϵB] < 10^-10, Break[]];
 ];
τ
```

Out[∘]= 2.56034

In[∘]:=
```
τ > Sqrt[(d - 1) / E]
```

Out[∘]= True

In[∘]:=
```
E0 = Eg;
{Hmat, Smat} = funMatGP[Λ, E0, τ, d, probφ, htot];
```

In[∘]:=
```
Do[
  AppendTo[E2GP, Hmat⟦i, i⟧ / Smat⟦i, i⟧];
  , {i, 1, d}]
```

In[∘]:=
```
Print[E2P]
Print[E2GP]
```

```
{-0.880688, -0.952035}
{-0.952035, -0.631498}
```

d = 3

In[ ]:=
```
d = 3;
```

Power

In[ ]:=
```
E0 = Eg + 1.;
{Hmat, Smat} = funMatP[Λ, E0, d, probφ];
```

In[ ]:=
```
Do[
 AppendTo[E3P, Hmat[[i, i]] / Smat[[i, i]]];
 , {i, 1, d}]
```

In[ ]:=
```
EB = Hmat[[d, d]] / Smat[[d, d]];
ϵB = EB - Eg (*used to identify τ for GP*)
```

Out[ ]= 0.0234086

Gaussian-Power

In[ ]:=
```
τMIN = 0;
τMAX = 64;
While[True,
  τ = (τMIN + τMAX) / 2.;
  {Hmat, Smat} = funMatGP[Λ, Eg, τ, 1, probφ, htot];
  EK = Hmat[[1, 1]] / Smat[[1, 1]];
  err = EK - Eg;
  If[err > ϵB, τMIN = τ];
  If[err < ϵB, τMAX = τ];
  If[Abs[err - ϵB] < 10^-10, Break[]];
 ];
τ
```

Out[ ]= 3.9666

In[ ]:=
```
τ > Sqrt[(d - 1) / E]
```

Out[ ]= True

In[ ]:=
```
E0 = Eg;
{Hmat, Smat} = funMatGP[Λ, E0, τ, d, probφ, htot];
```

In[ ]:=
```
Do[
  AppendTo[E3GP, Hmat[[i, i]] / Smat[[i, i]]];
  , {i, 1, d}]
```
Do循环
附加

In[ ]:=
```
Print[E3P]
Print[E3GP]
```
打印
打印

```
{-0.880688, -0.952035, -0.976591}
{-0.976591, -0.724535, -0.647446}
```

d = 4

In[ ]:=
```
d = 4;
```

Power

In[ ]:=
```
E0 = Eg + 1.;
{Hmat, Smat} = funMatP[Λ, E0, d, probφ];
```

In[ ]:=
```
Do[
  AppendTo[E4P, Hmat[[i, i]] / Smat[[i, i]]];
  , {i, 1, d}]
```
Do循环
附加

In[ ]:=
```
EB = Hmat[[d, d]] / Smat[[d, d]];
ϵB = EB - Eg (*used to identify τ for GP*)
```

Out[ ]=
```
0.0126574
```

Gaussian-Power

In[∘]:=
```
τMIN = 0;
τMAX = 64;
While[True,
  τ = (τMIN + τMAX) / 2.;
  {Hmat, Smat} = funMatGP[Λ, Eg, τ, 1, probφ, htot];
  EK = Hmat[[1, 1]] / Smat[[1, 1]];
  err = EK - Eg;
  If[err > ϵB, τMIN = τ];
  If[err < ϵB, τMAX = τ];
  If[Abs[err - ϵB] < 10^-10, Break[]];
 ];
τ
```
Out[∘]= 5.24256

In[∘]:=
```
τ > Sqrt[(d - 1) / E]
```
Out[∘]= True

In[∘]:=
```
E0 = Eg;
{Hmat, Smat} = funMatGP[Λ, E0, τ, d, probφ, htot];
```

In[∘]:=
```
Do[
 AppendTo[E4GP, Hmat[[i, i]] / Smat[[i, i]]];
 , {i, 1, d}]
```

In[∘]:=
```
Print[E4P]
Print[E4GP]
```
{-0.880688, -0.952035, -0.976591, -0.987343}

{-0.987343, -0.774182, -0.730553, -0.664494}

d = 5

In[∘]:=
```
d = 5;
```

Power

In[∘]:=
```
E0 = Eg + 1.;
{Hmat, Smat} = funMatP[Λ, E0, d, probφ];
```

```
In[•]:= Do[
        Do循环
         AppendTo[E5P, Hmat⟦i, i⟧ / Smat⟦i, i⟧];
         附加
         , {i, 1, d}]

In[•]:= EB = Hmat⟦d, d⟧ / Smat⟦d, d⟧;
        ϵB = EB - Eg(*used to identify τ for GP*)

Out[•]= 0.0072981
```

Gaussian-Power

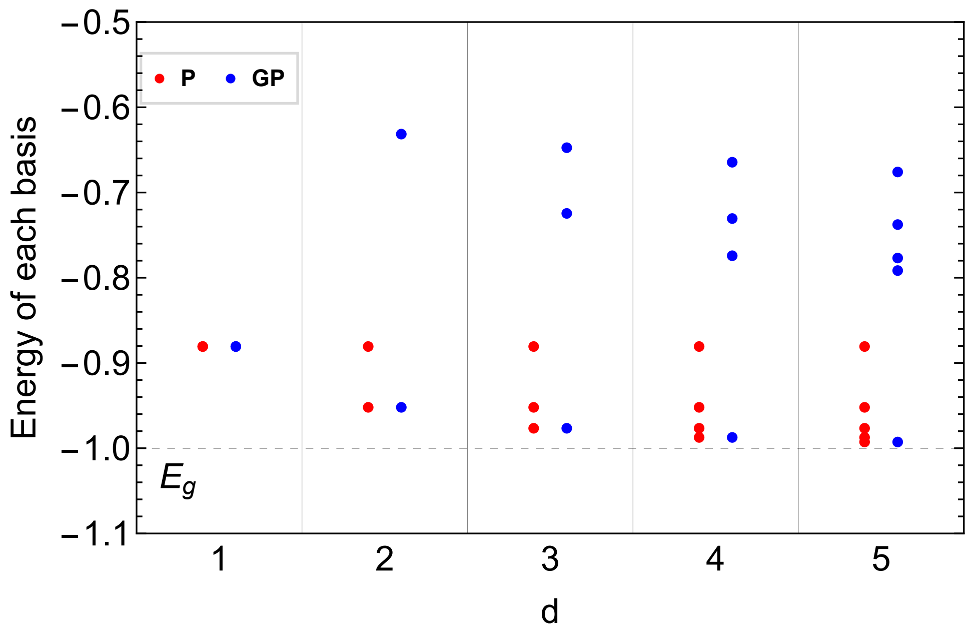
```
In[•]:= τMIN = 0;
        τMAX = 64;
        While[True,
        Whil⋯  真
          τ = (τMIN + τMAX) / 2.;
          {Hmat, Smat} = funMatGP[Λ, Eg, τ, 1, probφ, htot];
          EK = Hmat⟦1, 1⟧ / Smat⟦1, 1⟧;
          err = EK - Eg;
          If[err > ϵB, τMIN = τ];
          如果
          If[err < ϵB, τMAX = τ];
          如果
          If[Abs[err - ϵB] < 10^-10, Break[]];
          ⋯  绝对值                  跳出循环
         ];
        τ

Out[•]= 6.34638

In[•]:= τ > Sqrt[(d - 1) / E]
           平方根          自

Out[•]= True

In[•]:= E0 = Eg;
        {Hmat, Smat} = funMatGP[Λ, E0, τ, d, probφ, htot];

In[•]:= Do[
        Do循环
         AppendTo[E5GP, Hmat⟦i, i⟧ / Smat⟦i, i⟧];
         附加
         , {i, 1, d}]

In[•]:= Print[E5P]
        打印
        Print[E5GP]
        打印

        {-0.880688, -0.952035, -0.976591, -0.987343, -0.992702}

        {-0.992702, -0.79158, -0.776876, -0.737622, -0.67593}
```

Plot

In[ ]:=
```
E1P = {0.9, E1P[[#]]} & /@ {1};
E1GP = {1.1, E1GP[[#]]} & /@ {1};
E2P = {1.9, E2P[[#]]} & /@ {1, 2};
E2GP = {2.1, E2GP[[#]]} & /@ {1, 2};
E3P = {2.9, E3P[[#]]} & /@ {1, 2, 3};
E3GP = {3.1, E3GP[[#]]} & /@ {1, 2, 3};
E4P = {3.9, E4P[[#]]} & /@ {1, 2, 3, 4};
E4GP = {4.1, E4GP[[#]]} & /@ {1, 2, 3, 4};
E5P = {4.9, E5P[[#]]} & /@ {1, 2, 3, 4, 5};
E5GP = {5.1, E5GP[[#]]} & /@ {1, 2, 3, 4, 5};
```

In[ ]:=
```
PR = {{0.5, 5.5}, {-1.1, -0.5}};
plot = ListPlot[{E1P, E1GP, E2P, E2GP, E3P, E3GP, E4P, E4GP, E5P, E5GP}, PlotRange → PR,
       绘制点集                                                                绘制范围
    Joined → False, PlotStyle → {Red, Blue, Red, Blue, Red, Blue, Red, Blue, Red, Blue},
    连接点    假      绘图样式      红色 蓝色  红色 蓝色  红色 蓝色  红色 蓝色  红色 蓝色
    GridLines → {{1.5, 2.5, 3.5, 4.5}, {-1}},
    网格线
    GridLinesStyle → {Directive[Gray], Directive[Black, Dashed]},
    网格线样式         指令      灰色    指令      黑色   虚线
    Frame → True, FrameStyle → Directive[Black, Thickness[0.002]],
    边框    真     边框样式      指令      黑色   粗细
    FrameTicksStyle → Directive[Black, Thickness[0.002]],
    边框刻度样式        指令      黑色   粗细
    FrameLabel → {"d", "Energy of each basis"}, LabelStyle →
    边框标签                                      标签样式
     {FontSize → 18, FontFamily → "Arial"}, FrameTicks → {{Automatic, Automatic},
      字体大小        字体系列                   边框刻度        自动        自动
      {{{1, 1, 0}, {2, 2, 0}, {3, 3, 0}, {4, 4, 0}, {5, 5, 0}}, None}},
                                                                无
    ImageSize → 500, PlotLegends → Placed[LineLegend[{"P", "GP"}, LegendFunction →
    图像尺寸          绘图的图例      放置    线的图例                    图例函数
        (Framed[#, FrameStyle → LightGray] &), LegendMarkerSize → {10, 5},
         加边框     边框样式      浅灰色          图例标记尺寸
       LabelStyle → {Black, Bold, FontSize → 12, FontFamily → "Arial"},
       标签样式       黑色   粗体   字体大小         字体系列
       LegendMargins → 0, LegendLayout → {"Column", 2}], {0.1, 0.89}]];
       图例边幅            图例布局          列
txt = Text[Style["Eg", 18], {0.75, -1.01}, {0, 1}];
      文本  样式
Fig = Show[plot, Graphics[txt]]
      显示        图形
```

Out[ ]=




```
path = "energy_of_each_basis.pdf";
Export[path, Fig];
导出
```