
Exercise 9

Level Spacing Distribution for NNN model

■ Hamiltonian in the site-basis

■ Parameters of the Hamiltonian

*) L = number of sites.

In Figs. 1 and 3 [panels (b), (c), (d) and (e)] we consider L=15 and L/3 spins up; dimension=3003.

Figure 3 (a) uses L=14 and L/2 spins up; dimension=3432.

Figure 2 has L=18 and L/3 spins up; dimension=18564. The data was obtained using Fortran 77. *Mathematica* can deal with L=15, but has memory problems with larger systems. The fact that NPC becomes a smooth function of energy in the chaotic regime is more evident for L=18 than L=15, which explains why we resorted to Fortran. However, the *Mathematica* codes to compute NPC when L=15, which are provided below, can still give an idea of the general trend.

*) upspins = number of spins pointing up in the z direction.

*) downspins = number of spins pointing down in the z direction.

*) dim = dimension of the Sz-subspace considered.

*) ϵ = value of the defect (all other Zeeman splittings are assumed = 0).

*) defectsite = site where the defect is placed.

*) Jxy = strength of the flip-flop term between nearest-neighbors (NNs).

*) Jz = strength of the Ising interaction between NNs.

*) alpha different from zero indicates the presence of next-nearest-neighbor (NNN) couplings.

*) JJxy = strength of the flip-flop term between NNNs.

*) JJz = strength of the Ising interaction between NNNs.

*) onebasisvector is the site-basis vector 111110000000000

*) its permutations generate all basis vectors for that particular subspace

*) HH are the elements of the Hamiltonian. In the site-basis, the on-site defect and the Ising interaction contribute to the diagonal elements. The flip-flop term contributes to the off-diagonal elements.

■ The code

```
(* PARAMETERS OF THE HAMILTONIAN *)
Clear[L, upspins, downspins, dim, Jxy, Jz,  $\epsilon$ , defectsite, alpha, JJxy, JJz];
L = 15;
upspins = L / 3;
downspins = L - upspins;
dim = L! / (upspins! downspins!);
(* Impurity *)
 $\epsilon$  = 0.0;
defectsite = Floor[L / 2];
(* Parameters for nearest-neighbor couplings *)
Jxy = 1.0;
Jz = 0.5;
(* Parameters for next-nearest-neighbor couplings *)
(* Set alpha=0 if NNN couplings do not exist *)
alpha = 0.5;
JJxy = 1.0;
JJz = 0.5;

(* BASIS *)
Clear[onebasisvector, basis];
onebasisvector = Flatten[Table[1, {k, 1, upspins}], Table[0, {k, 1, downspins}]];
basis = Permutations[onebasisvector];
```

```

(* ELEMENTS OF THE HAMILTONIAN *)
(* Impurity and NN couplings *)
Clear[HH];
(* Initialization *)
Do[Do[HH[i, j] = 0., {i, 1, dim}], {j, 1, dim}];

(* Diagonal elements *)
Do[
  (* Impurity *)
  If[basis[[i, defectsite]] == 1, HH[i, i] = HH[i, i] +  $\epsilon$  / 2., HH[i, i] = HH[i, i] -  $\epsilon$  / 2.];
  (* Ising interaction *)
  Do[If[basis[[i, j]] == basis[[i, j+1]],
    HH[i, i] = HH[i, i] + Jz / 4., HH[i, i] = HH[i, i] - Jz / 4.];
    , {j, 1, L-1}];
    , {i, 1, dim}];

(* Off-diagonal elements *)
Clear[howmany, site];
Do[
  Do[
    (* Initialization *)
    howmany = 0;
    Do[site[kk] = 0, {kk, 1, L}];
    (* Sites where states i and j differ *)
    Do[If[basis[[i, k]] != basis[[j, k]], {howmany = howmany + 1, site[howmany] = k}];
      , {k, 1, L}];
    (* If only two neighbor sites differ, there is a coupling matrix element *)
    If[howmany == 2,
      If[site[2] - site[1] == 1, {HH[i, j] = Jxy / 2., HH[j, i] = Jxy / 2.}];
      , {j, i+1, dim}];
      , {i, 1, dim-1}];

(* ELEMENTS OF THE HAMILTONIAN WITH NNN COUPLINGS *)
If[alpha > 0,

(* Diagonal elements *)
Do[
  (* Ising interaction *)
  Do[If[basis[[i, j]] == basis[[i, j+2]],
    HH[i, i] = HH[i, i] + alpha JJz / 4., HH[i, i] = HH[i, i] - alpha JJz / 4.];
    , {j, 1, L-2}];
    , {i, 1, dim}];

(* Off-diagonal elements *)
Clear[howmany, site];
Do[
  Do[
    (* Initialization *)
    howmany = 0;
    Do[site[kk] = 0, {kk, 1, L}];
    (* Sites where states i and j differ *)
    Do[If[basis[[i, k]] != basis[[j, k]], {howmany = howmany + 1, site[howmany] = k}];
      , {k, 1, L}];
    (* If only two next-neighbor sites differ, there is a coupling matrix element *)
    If[howmany == 2,
      If[site[2] - site[1] == 2, {HH[i, j] = alpha JJxy / 2., HH[j, i] = alpha JJxy / 2.}];
      , {j, i+1, dim}];
      , {i, 1, dim-1}];

];

(* TOTAL HAMILTONIAN AND DIAGONALIZATION *)
Clear[Hamiltonian, Energy, Vector];

```

```

Hamiltonian = Table[Table[HH[i, j], {j, 1, dim}], {i, dim}];
Energy = Eigenvalues[Hamiltonian];
Vector = Eigenvectors[Hamiltonian];

```

■ **Level spacing distribution for NN+NNN**
(parity **is** taken into account)

```

(* PARITY *)
(* First we find all pairs of equivalent basis vectors *)
(* Example: state 1100 pairs with state 0011 *)
Clear[i, j, k, pair];
Do[
  Do[Clear[mirror];
    mirror = 0;
    Do[mirror = mirror + Mod[basis[[i, k]] + basis[[j, L + 1 - k]], 2];
      , {k, 1, L}];
    If[mirror == 0, pair[i] = j];
      , {j, 1, dim}];
    , {i, 1, dim}];

(* Now we separate the eigenstates according to their parity, even or odd *)
Clear[Neven, Nodd];
(* Neven is the number of states with even parity *)
(* Nodd is the number of states with odd parity *)
Neven = 0;
Nodd = 0;
Do[
  Clear[project];
  project = 0;
  Do[project = project + (1 / 4.) (Vector[[i]][[j]] + Vector[[i]][[pair[j]]])^2;
    , {j, 1, dim}];
  (* project will be very close to 1 if the parity is even *)
  (* project will be very close to 0 if the parity is odd *)
  If[1. - project < 0.001,
    {Neven = Neven + 1, even[Neven] = i}, {Nodd = Nodd + 1, odd[Nodd] = i}];
    , {i, 1, dim}];

(* FOR THE HISTOGRAM *)
Clear[spcmin, spcmax, bin, Nofbins];
spcmin = 0.;
spcmax = 8.;
bin = 0.1;
Nofbins = IntegerPart[(spcmax - spcmin) / bin];
(* Theoretical curves *)
Clear[WignerDyson, Poisson];
WignerDyson =
  Plot[Pi s / 2. Exp[-Pi s^2 / 4.], {s, 0, 8}, PlotRange -> {0, 1}, PlotStyle -> {Red, Thick},
    LabelStyle -> Directive[Black, Bold, Medium], AxesLabel -> {"s", "P"}];
Poisson = Plot[Exp[-s], {s, 0, 8}, PlotRange -> {0, 1}, PlotStyle -> {Blue, Thick},
  LabelStyle -> Directive[Black, Bold, Medium], AxesLabel -> {"s", "P"}];

(* EVEN EIGENSTATES *)
(* LEVEL SPACINGS OF THE UNFOLDED SPECTRUM *)
(* Order the eigenvalues *)
Clear[Ener];
Ener = Sort[Table[Energy[[even[k]]], {k, 1, Neven}]];
(* Discard ~10% of the eigenvalues located at the borders of the spectrum *)
Clear[percentage, half, spacing];
percentage = 0.1 Neven;
half = Floor[percentage / 2.];
(* Compute the level spacings for the remaining eigenvalues after unfolding them *)
(* Unfolding here means that the average of each group of 10 level spacings = 1 *)
Do[
  Clear[average];

```

```

average = (Ener[[half + 10 j]] - Ener[[half + 10 (j - 1)]]) / 10.;
Do[spacing[i] = (Ener[[half + i]] - Ener[[half - 1 + i]]) / average;
, {i, 1 + 10 (j - 1), 10 j}];
, {j, 1, Floor[(Neven - percentage) / 10]}];
(* ... *)
(* HISTOGRAM -- EVEN *)
Clear[SPChist, Nhist];
SPChist[1] = spcmin;
Do[SPChist[i + 1] = SPChist[i] + bin, {i, 1, Nofbins}];
Do[Nhist[j] = 0., {j, 1, Nofbins}];
(* ... *)
Do[
  Do[
    If[SPChist[j] ≤ spacing[k] < SPChist[j + 1], Nhist[j] = Nhist[j] + 1];
    , {j, 1, Nofbins}];
    , {k, 1, 10 Floor[(Neven - percentage) / 10]}];
(* Normalization *)
Clear[Norma];
Norma = Sum[bin Nhist[j], {j, 1, Nofbins}];
Do[Nhist[j] = Nhist[j] / Norma, {j, 1, Nofbins}];
(* ... *)
(* ListPlot with the obtained data *)
Clear[jj, histEven, PlotEven];
jj = 0;
histEven = {};
Do[jj += 1;
  histEven = Append[histEven, {SPChist[jj], Nhist[jj]}];
  histEven = Append[histEven, {SPChist[jj + 1], Nhist[jj]}];
  , {j, 1, Nofbins - 1}];
PlotEven =
  ListPlot[histEven, Joined → True, PlotRange → {{0, 8}, {0, 1}}, PlotStyle → {Black, Thick},
  LabelStyle → Directive[Black, Bold, Medium], AxesLabel → {"s", "P"}];
(* The three curves together *)
Print["Histogram for the even eigenstates"];
Show[{PlotEven, WignerDyson, Poisson}, PlotRange → {{0, 4}, {0, 1.2}}]

```

```

(* ODD EIGENSTATES *)
(* LEVEL SPACINGS OF THE UNFOLDED SPECTRUM *)
(* Order the eigenvalues *)
Clear[Ener];
Ener = Sort[Table[Energy[[odd[k]]], {k, 1, Nodd}]];
(* Discard ~10% of the eigenvalues located at the borders of the spectrum *)
Clear[percentage, half, spacing];
percentage = 0.1 Nodd;
half = Floor[percentage / 2.];
(* Compute the level spacings for the remaining eigenvalues after unfolding them *)
(* Unfolding here means that the average of each group of 10 level spacings = 1 *)
Do[
  Clear[average];
  average = (Ener[[half + 10 j]] - Ener[[half + 10 (j - 1)]]) / 10.;
  Do[spacing[i] = (Ener[[half + i]] - Ener[[half - 1 + i]]) / average;
  , {i, 1 + 10 (j - 1), 10 j}];
  , {j, 1, Floor[(Nodd - percentage) / 10]}];
(* ... *)
(* HISTOGRAM -- ODD *)
Clear[SPChist, Nhist];
SPChist[1] = spcmin;
Do[SPChist[i + 1] = SPChist[i] + bin, {i, 1, Nofbins}];
Do[Nhist[j] = 0., {j, 1, Nofbins}];
(* ... *)
Do[
  Do[

```

```

    If[SPChist[j] ≤ spacing[k] < SPChist[j + 1], Nhist[j] = Nhist[j] + 1];
    , {j, 1, Nofbins}];
    , {k, 1, 10 Floor[(Nodd - percentage) / 10]}];
(* Normalization *)
Clear[Norma];
Norma = Sum[bin Nhist[j], {j, 1, Nofbins}];
Do[Nhist[j] = Nhist[j] / Norma, {j, 1, Nofbins}];
(* ... *)
(* ListPlot with the obtained data *)
Clear[jj, histOdd, PlotOdd];
jj = 0;
histOdd = {};
Do[jj += 1;
    histOdd = Append[histOdd, {SPChist[jj], Nhist[jj]}];
    histOdd = Append[histOdd, {SPChist[jj + 1], Nhist[jj]}];
    , {j, 1, Nofbins - 1}];
PlotOdd =
ListPlot[histOdd, Joined → True, PlotRange → {{0, 8}, {0, 1}}, PlotStyle → {Black, Thick},
    LabelStyle → Directive[Black, Bold, Medium], AxesLabel → {"s", "P"}];
(* The three curves together *)
Print["Histogram for the odd eigenstates"];
Show[{PlotOdd, WignerDyson, Poisson}, PlotRange → {{0, 4}, {0, 1.2}}]

(* AVERAGE of EVEN and ODD HISTOGRAMS *)
Clear[AveHist];
AveHist = (histEven + histOdd) / 2.;
PlotAve =
ListPlot[AveHist, Joined → True, PlotRange → {{0, 8}, {0, 1}}, PlotStyle → {Black, Thick},
    LabelStyle → Directive[Black, Bold, Medium], AxesLabel → {"s", "P"}];
(* The three curves together *)
Print["Histogram for the averaged P(s) over even and odd parity"];
Show[{PlotAve, WignerDyson, Poisson}, PlotRange → {{0, 4}, {0, 1.2}}]

```