

Title: Computational Physics - Lecture 22

Date: Nov 23, 2018 01:00 PM

URL: <http://pirsa.org/18110048>

Abstract:

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag3-complete.ipynb

jupyter 2018-comp-phys-ExactDiag3-complete Last Checkpoint: Last Wednesday at 1:35 PM (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

```
end
```

The image displays two side-by-side plots showing the scaling dimension of various operators as a function of their conformal spin. The x-axis for both plots is 'conformal spin' ranging from -3 to 3, and the y-axis is 'scaling dimension' ranging from 0.0 to 2.5. The left plot is titled 'Even Parity, p=0' and shows four data points at (conformal spin, scaling dimension) = (-2, 2.0), (-1, 2.0), (1, 2.0), and (2, 2.0). Two additional points are labeled: ϵ 'energy density' at (0, 1.0) and I 'identity' at (0, 0.0). The right plot is titled 'Odd Parity, p=1' and shows four data points at (conformal spin, scaling dimension) = (-2, 2.15), (-1, 1.15), (1, 1.15), and (2, 2.15). One point is labeled: σ 'spin' at (0, 0.15).

Parity	Operator	Conformal Spin	Scaling Dimension
Even Parity, p=0	-	-2	2.0
	-	-1	2.0
	ϵ 'energy density'	0	1.0
	I 'identity'	0	0.0
Odd Parity, p=1	-	-2	2.15
	-	-1	1.15
	σ 'spin'	0	0.15
	-	1	1.15
-	2	2.15	

12:59 PM 23/11/2018

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag3-complete.ipynb

jupyter 2018-comp-phys-ExactDiag3-complete Last Checkpoint: Last Wednesday at 1:35 PM (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

```
end
```

The image displays two plots side-by-side, illustrating the scaling dimension of various operators as a function of their conformal spin. The x-axis for both plots is 'conformal spin' ranging from -3 to 3, and the y-axis is 'scaling dimension' ranging from 0.0 to 2.5. The left plot is titled 'Even Parity, p=0' and shows four data points at conformal spin values of -2, -1, 1, and 2, all with a scaling dimension of 2.0. Additionally, there are two points at conformal spin 0: one at scaling dimension 1.0 labeled with the Greek letter ϵ and 'energy density', and another at scaling dimension 0.0 labeled with the Greek letter I and 'identity'. The right plot is titled 'Odd Parity, p=1' and shows four data points at conformal spin values of -2, -1, 1, and 2, with scaling dimensions of approximately 2.15, 1.15, 1.15, and 2.15 respectively. There is also a point at conformal spin 0 with a scaling dimension of approximately 0.15 labeled with the Greek letter σ and 'spin'.

Parity	Operator	Conformal Spin	Scaling Dimension
Even Parity, p=0	Energy Density (ϵ)	0	1.0
	Identity (I)	0	0.0
	Operator 1	-2	2.0
	Operator 2	-1	2.0
	Operator 3	1	2.0
	Operator 4	2	2.0
Odd Parity, p=1	Spin (σ)	0	~0.15
	Operator 1	-2	~2.15
	Operator 2	-1	~1.15
	Operator 3	1	~1.15
	Operator 4	2	~2.15



localhost:8888/notebooks/Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag3-complete.ipynb

jupyter 2018-comp-phys-ExactDiag3-complete Last Checkpoint: Last Wednesday at 1:35 PM (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

Exercise (2/2): obtain & plot numerical estimates for 12 scaling dimensions and conformal spins ($N = 10$ spins)

hint: In any CFT, the lowest energy state with momentum $k2 \times 2\pi/N$ (or spin $s = 2$) corresponds to a CFT operator called stress tensor, which has $(\Delta_T, s_T) = (2, 2)$.

Find the stress tensor energy eigenstate $|T\rangle$ and use its energy to estimate the scaling dimensions Δ_i of the rest of energy eigenstates according to $\Delta_i = 2(e_i - e_1)/(e_T - e_1)$.

Solution

```
In [11]: # function buildT takes as input N (number of spins) and outputs T (translatio operator)
function buildT(N::Int64)::Array{Float64,2}
```

1:05 PM 23/11/2018

localhost:8888/notebooks/Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag3-complete.ipynb

jupyter 2018-comp-phys-ExactDiag3-complete Last Checkpoint: Last Wednesday at 1:35 PM (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

Exercise (2/2). Obtain & plot numerical estimates for 12 scaling dimensions and conformal spins ($N = 10$ spins)

hint: In any CFT, the lowest energy state with momentum $k_2 \times 2\pi/N$ (or spin $s = 2$) corresponds to a CFT operator called stress tensor, which has $(\Delta_T, s_T) = (2, 2)$.

Find the stress tensor energy eigenstate $|T\rangle$ and use its energy to estimate the scaling dimensions Δ_i of the rest of energy eigenstates according to $\Delta_i = 2(e_i - e_1)/(e_T - e_1)$.

Solution

```
In [11]: # function buildT takes as input N (number of spins) and outputs T (translatio operator)
function buildT(N::Int64)::Array{Float64,2}
    SWAP = [1 0 0 0; 0 0 1 0; 0 1 0 0; 0 0 0 1]
    E = diagm(0=>ones(2))
    T = copy(SWAP)
    for n=3:N
        SWAPn = kron(diagm(0=>ones(2^(n-2))), SWAP)
        T = SWAPn*kron(T,E)
    end
end
```

1:06 PM 23/11/2018

Google Documents/2018juliacourse/Guilfr/2018-comp-phys-ExactDiag3-complete.ipynb

jupyter 2018-comp-phys-ExactDiag3-complete Last Checkpoint: Last Wednesday at 1:35 PM (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

Run Code

Exercise (2/2): obtain & plot numerical estimates for 12 scaling dimensions and conformal spins ($N = 10$ spins)

hint: In any CFT, the lowest energy state with momentum $k2 \times 2\pi/N$ (or spin $s = 2$) corresponds to a CFT operator called stress tensor, which has $(\Delta_T, s_T) = (2, 2)$.

Find the stress tensor energy eigenstate $|T\rangle$ and use its energy to estimate the scaling dimensions Δ_i of the rest of energy eigenstates according to $\Delta_i = 2(e_i - e_1)/(e_T - e_1)$.

Solution

```
In [11]: # function buildT takes as input N (number of spins) and outputs T (translatio operator)
```

1:11 PM 23/11/2018

localhost:8888/notebooks/Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag3-complete.ipynb

jupyter 2018-comp-phys-ExactDiag3-complete Last Checkpoint: Last Wednesday at 1:35 PM (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

Run Code

Exercise (2/2). Obtain & plot numerical estimates for 12 scaling dimensions and conformal spins ($N = 10$ spins)

hint: In any CFT, the lowest energy state with momentum $k2 \times 2\pi/N$ (or spin $s = 2$) corresponds to a CFT operator called stress tensor, which has $(\Delta_T, s_T) = (2, 2)$.

Find the stress tensor energy eigenstate $|T\rangle$ and use its energy to estimate the scaling dimensions Δ_i of the rest of energy eigenstates according to $\Delta_i = 2(e_i - e_1)/(e_T - e_1)$.

Solution

```
In [11]: # function buildT takes as input N (number of spins) and outputs T (translatio operator)
function buildT(N::Int64)::Array{Float64,2}
    SWAP = [1 0 0 0; 0 0 1 0; 0 1 0 0; 0 0 0 1]
    E = diagm(0=>ones(2))
    T = copy(SWAP)
    for n=3:N
        SWAPn = kron(diagm(0=>ones(2^(n-2))), SWAP)
        T = SWAPn*kron(T,E)
    end
end
```

1:11 PM 23/11/2018

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag3-complete.ipynb

jupyter 2018-comp-phys-ExactDiag3-complete Last Checkpoint: Last Wednesday at 1:35 PM (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

```
0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0
0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0
```

In [12]: *# function buildP takes as input N (number of spins) and outputs P (parity operator)*

```
function buildP(N::Int64)::Array{Float64,2}
    Z = [1 0; 0 -1]
    P = copy(Z)
    for i=2:N
        P = kron(P,Z)
    end
    return P
end

buildP(3) # test for N=3
```

Out[12]: 8x8 Array{Float64,2}:

```
1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 -1.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 -1.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 -1.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0
```

1:12 PM 23/11/2018

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag3-complete.ipynb

jupyter 2018-comp-phys-ExactDiag3-complete Last Checkpoint: Last Wednesday at 1:35 PM (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

```

1.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0 -1.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0 -1.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  1.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0 -1.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  1.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  1.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0 -1.0

```

In [13]:

```

N=10
Nstates = 12
H = buildH(N,pi/4)
T = buildT(N)
P = buildP(N)
HTP = H + 0.0001*T + im*0.01*P - N*2*I # notice that we need to the Low energy spectrum have the Larges
D,U = eigs(HTP, nev=Nstates); # which is, eigs(HTP, nev=Nstates, which=:LM) by default

```

In [14]:

```

e = real(diag(U'*H*U))
k = angle.(diag(U'*T*U))
p = real(diag(U'*P*U))
p = round.((1.-p)/2)
perm = sortperm(e) # re-order eigenvalues according to E
e = e[perm]
k = k[perm]

```

1:12 PM 23/11/2018

$$|\Psi_\alpha\rangle \quad e_\alpha = A + B \left[\frac{2\pi}{N} \left(\Delta_\alpha - \frac{c}{12} \right) \right] + O\left(\frac{1}{N^{1+\delta}}\right)$$

eigen state of H, T

↑ energy lattice

$$K_\alpha = \frac{2\pi}{N} S_\alpha$$

• identity state (= ground state)

$$|D\rangle \quad (\Delta_D, S_D) = (0, 0)$$

$$e_\alpha^{\text{CFT}} = \frac{2\pi}{N} \left(\Delta_\alpha - \frac{c}{12} \right)$$

• stress tensor state

$$|T\rangle \quad (\Delta_T, S_T) = (2, 2)$$

$$\frac{e_\alpha - e_D}{e_T - e_D} = \frac{\Delta_\alpha - \Delta_D}{\Delta_T - \Delta_D} = \frac{\Delta_\alpha}{2} \Rightarrow \Delta_\alpha = 2 \frac{e_\alpha - e_D}{e_T - e_D}$$

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag3-complete.ipynb

jupyter 2018-comp-phys-ExactDiag3-co... Last Checkpoint: Last Wednesday at 1:35 PM (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

```
1.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0 -1.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0 -1.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  1.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0 -1.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  1.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  1.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0 -1.0
```

In [16]: `M = rand`

Out[16]: `UniformScaling{Bool}`
`true*I`

In [13]: `N=10`
`Nstates = 12`
`H = buildH(N,pi/4)`
`T = buildT(N)`
`P = buildP(N)`
`HTP = H + 0.0001*T + im*0.01*P - N*2*I # notice that we need to the low energy spectrum have the larges`
`D,U = eigs(HTP, nev=Nstates); # which is, eigs(HTP, nev=Nstates, which=:LM) by default`

In [14]: `e = real(diag(U'*H*U))`
`k = angle(diag(U'*T*U))`

1:14 PM 23/11/2018

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag3-complete.ipynb

jupyter 2018-comp-phys-ExactDiag3-co... Last Checkpoint: Last Wednesday at 1:35 PM (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

```

1.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0 -1.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0 -1.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  1.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0 -1.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  1.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  1.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0 -1.0

```

In [17]: `M = rand(2,2)`

Out[17]: 2x2 Array{Float64,2}:
0.653606 0.820513
0.814026 0.258416

In [13]: `N=10`
`Nstates = 12`
`H = buildH(N,pi/4)`
`T = buildT(N)`
`P = buildP(N)`
`HTP = H + 0.0001*T + im*0.01*P - N*2*I # notice that we need to the low energy spectrum have the largest`
`D,U = eigs(HTP, nev=Nstates); # which is, eigs(HTP, nev=Nstates, which=:LM) by default`

In [14]: `e = real(diag(U'*H*U))`

1:14 PM 23/11/2018

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag3-complete.ipynb

jupyter 2018-comp-phys-ExactDiag3-co... Last Checkpoint: Last Wednesday at 1:35 PM (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

```
1.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0 -1.0  0.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0 -1.0  0.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  1.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0 -1.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  1.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  1.0  0.0
0.0  0.0  0.0  0.0  0.0  0.0  0.0 -1.0
```

In [17]: `M = rand(2,2)`

Out[17]: 2x2 Array{Float64,2}:
0.653606 0.820513
0.814026 0.258416

In []: `I`

In [13]: `N=10
Nstates = 12
H = buildH(N,pi/4)
T = buildT(N)
P = buildP(N)
HTP = H + 0.0001*T + im*0.01*P - N*2*I # notice that we need to the Low energy spectrum have the Largest
D,U = eigs(HTP, nev=Nstates); # which is, eigs(HTP, nev=Nstates, which=:LM) by default`

1:14 PM 23/11/2018

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag3-complete.ipynb

jupyter 2018-comp-phys-ExactDiag3-complete Last Checkpoint: Last Wednesday at 1:35 PM (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

Out[19]: 2x2 Array{Float64,2}:
1.65361 0.820513
0.814026 1.25842

```
In [13]: N=10
Nstates = 12
H = buildH(N,pi/4)
T = buildT(N)
P = buildP(N)
HTP = H + 0.0001*T + im*0.01*P - N*2*I # notice that we need to the Low energy spectrum have the Largest
D,U = eigs(HTP, nev=Nstates); # which is, eigs(HTP, nev=Nstates, which=:LM) by default
```

```
In [14]: e = real(diag(U'*H*U))
k = angle.(diag(U'*T*U))
p = real(diag(U'*P*U))
p = round.((1 .- p)/2)
perm = sortperm(e) # re-order eigenvalues according to E
e = e[perm]
k = k[perm]
p = p[perm]
display([e k p])

s = k/(2pi)*N
```

1:15 PM 23/11/2018

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag3-complete.ipynb

2018-comp-phys-ExactDiag3-complete (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

In [19]: `M+I`

Out[19]: `2x2 Array{Float64,2}:
1.65361 0.820513
0.814026 1.25842`

In [20]: `N=10
Nstates = 12
H = buildH(N,pi/4)
T = buildT(N)
P = buildP(N)
HTP = H + 0.0001*T + im*0.01*P - N*2*I # notice that we need to the Low energy sp
D,U = eigs(HTP, nev=Nstates); # which is, eigs(HTP, nev=Nstates, which=:LM) by de`

In [14]: `e = real(diag(U'*H*U))
k = angle.(diag(U'*T*U))
p = real(diag(U'*P*U))
p = round.((1 .- p)/2)
perm = sortperm(e) # re-order eigenvalues according to E`

1:19 PM 23/11/2018

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag3-complete.ipynb

2018-comp-phys-ExactDiag3-complete (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

```
P = buildP(N)
HTP = H + 0.0001*T + im*0.01*P - N*2*I # notice that we need to the Low energy sp
D,U = eigs(HTP, nev=Nstates); # which is, eigs(HTP, nev=Nstates, which=:LM) by de
```

In [14]:

```
e = real(diag(U'*H*U))
k = angle.(diag(U'*T*U))
p = real(diag(U'*P*U))
p = round.((1 .- p)/2)
perm = sortperm(e) # re-order eigenvalues according to E
e = e[perm]
k = k[perm]
p = p[perm]
display([e k p])

s = k/(2pi)*N

i_stress = 0
for i in 1:size(e,1)
    if abs(s[i] - 2.0) < 1e-5 && i_stress == 0
        i_stress = i
    end
end
```

1:20 PM 23/11/2018

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag3-complete.ipynb

jupyter 2018-comp-phys-ExactDiag3-complete (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

```
end
d = (2.0-0)*(e.-e[1])/(e[i_stress]-e[1])
display([d s p])
```

12x3 Array{Float64,2}:
-9.04029 1.19262e-17 -0.0
-8.92899 7.63278e-17 1.0
-8.15537 -1.53523e-16 -0.0
-8.05496 0.628319 1.0
-8.05496 -0.628319 1.0
-7.31375 0.628319 -0.0
-7.31375 1.25664 -0.0
-7.31375 -1.25664 -0.0
-7.31375 -0.628319 -0.0
-7.26649 -1.25664 1.0
-7.26649 1.25664 1.0
-7.18093 -2.94903e-17 1.0

12x3 Array{Float64,2}:
0.0 1.89812e-17 -0.0
0.128929 1.2148e-16 1.0

1:20 PM 23/11/2018

eigen state
of
 \mathcal{H}, T

energy
lattice

$$K_\alpha = \frac{2\pi}{N} S_\alpha$$

• identity
state
(= ground
state)

$|\mathbb{1}\rangle$

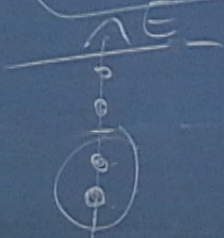
$$(\Delta_{\mathbb{1}}, S_{\mathbb{1}}) = (0, 0)$$

$$e_\alpha^{CFT} = \frac{2\pi}{N} \left(\Delta_\alpha - \frac{c}{12} \right)$$

• stress tensor
state

$|T\rangle$

$$(\Delta_T, S_T) = (2, 2)$$



$$\frac{e_\alpha - e_1}{e_T - e_1} = \frac{\Delta_\alpha - \Delta_{\mathbb{1}}}{\Delta_T - \Delta_{\mathbb{1}}} + \dots = \frac{\Delta_\alpha}{2} \Rightarrow \Delta_\alpha = 2 \frac{e_\alpha - e_1}{e_T - e_1}$$

eigs (M, $n_{ev} = 2$, which =: SR)

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag3-complete.ipynb

jupyter 2018-comp-phys-ExactDiag3-complete (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

```
perm = sortperm(e) # re-order eigenvalues according to E
e = e[perm]
k = k[perm]
p = p[perm]
display([e k p])

s = k/(2pi)*N

i_stress = 0
for i in 1:size(e,1)
    if abs(s[i] - 2.0) < 1e-5 && i_stress == 0
        i_stress = i
    end
end
d = (2.0-0)*(e.-e[1])/(e[i_stress]-e[1])

display([d s p])
```

12x3 Array{Float64,2}:
-9.04029 1.19262e-17 -0.0
-8.92899 7.63278e-17 1.0
8.15527 1.52522e-16 0.0

1:22 PM 23/11/2018

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag3-complete.ipynb

jupyter 2018-comp-phys-ExactDiag3-complete (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

Run Code

```
-7.31375 -1.25664 -0.0
-7.31375 -0.628319 -0.0
-7.26649 -1.25664 1.0
-7.26649 1.25664 1.0
-7.18093 -2.94903e-17 1.0

12x3 Array{Float64,2}:
 0.0      1.89812e-17 -0.0
 0.128929 1.2148e-16  1.0
 1.02509  -2.44339e-16 -0.0
 1.14139  1.0          1.0
 1.14139  -1.0         1.0
 2.0      1.0         -0.0
 2.0      2.0         -0.0
 2.0      -2.0        -0.0
 2.0      -1.0        -0.0
 2.05475  -2.0         1.0
 2.05475  2.0          1.0
 2.15386  -4.69353e-17 1.0
```

In [15]: *# Exact and numerical values together in a plot*

1:23 PM 23/11/2018

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag3-complete.ipynb

jupyter 2018-comp-phys-ExactDiag3-complete (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

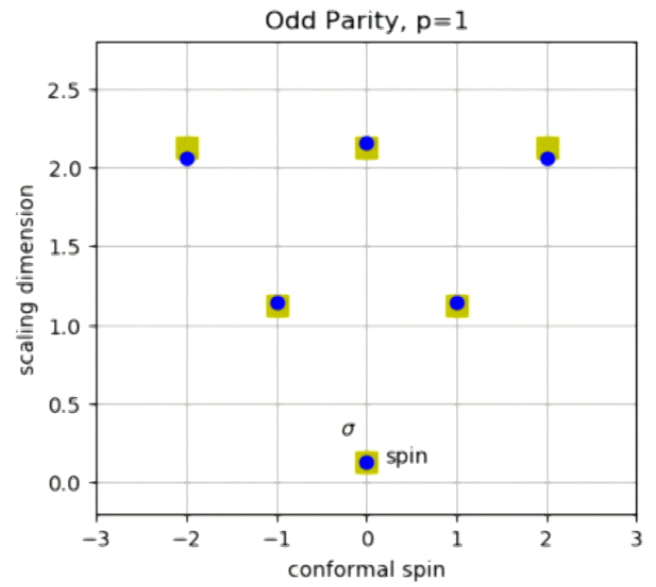
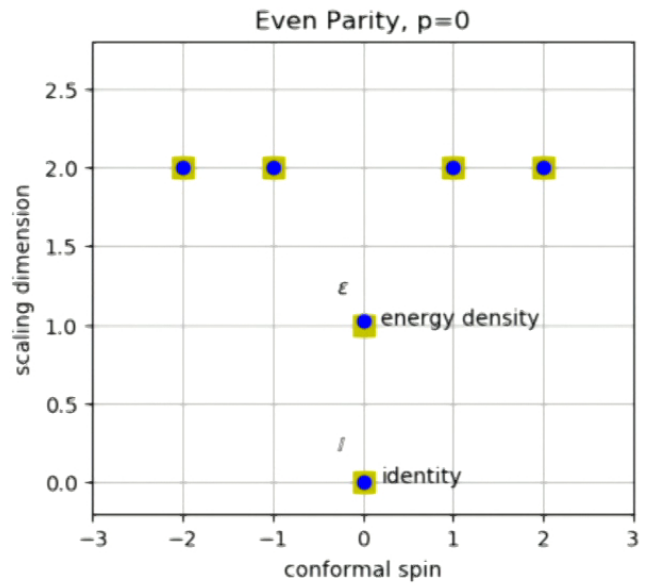
```
plot(s[p[i]],d[p[i]], marker = "o", color = "b")
subplot(121)
plot(s[i],d[i], marker = "o", color = "b")
elseif p[i] == 1
subplot(122)
plot(s[i],d[i], marker = "o", color = "b")
end
end
```

The figure consists of two side-by-side plots. The left plot is titled "Even Parity, p=0" and the right plot is titled "Odd Parity, p=1". Both plots have "scaling dimension" on the y-axis (ranging from 0.5 to 2.5) and "energy density" on the x-axis. In the "Even Parity, p=0" plot, there are four blue circles at a scaling dimension of approximately 2.0, and one blue circle at a scaling dimension of approximately 1.0 labeled with the Greek letter epsilon. In the "Odd Parity, p=1" plot, there are four blue circles at a scaling dimension of approximately 2.1, and two blue circles at a scaling dimension of approximately 1.1.

Parity	Energy Density	Scaling Dimension
Even Parity, p=0	Low	~2.0
	Medium-Low	~2.0
	Medium	~2.0
	Medium-High	~2.0
	High	~1.0 (epsilon)
Odd Parity, p=1	Low	~2.1
	Low	~1.1
	Medium	~2.1
	Medium	~1.1
	High	~2.1
	High	~1.1

1:24 PM 23/11/2018

```
end
```



Lecture 4 Quantum entanglement

Q.M. bipartite system AB

$$V_{AB} \cong V_A \otimes V_B$$

$$\{|ij\rangle = |i\rangle_A \otimes |j\rangle_B\} \quad \left\{ |i\rangle_A \right\} \quad \left\{ |j\rangle_B \right\}$$

A	B
\equiv	$\otimes \equiv$
$ 1\rangle$	$ 1\rangle$
$ 2\rangle$	$ 2\rangle$
$ 3\rangle$	

example

dimension $d_A = 3$
 $d_B = 2$

$$|\Psi_{AB}\rangle = \psi_{11}|11\rangle + \psi_{12}|12\rangle + \psi_{21}|21\rangle + \psi_{22}|22\rangle + \psi_{31}|31\rangle + \psi_{32}|32\rangle$$

$$d_{AB} = d_A \cdot d_B = 6$$

Q.M. bipartite system AB

$$V_{AB} \cong V_A \otimes V_B$$

$$\{|ij\rangle = |i\rangle_A \otimes |j\rangle_B\} \quad \left\{ \begin{array}{l} |i\rangle_A \\ |j\rangle_B \end{array} \right\}$$

A	B
—	⊗ —
11)	11)
12)	12)
13)	

example

dimension $d_A = 3$

$d_B = 2$

$$|\Psi_{AB}\rangle = \psi_{11}|11\rangle + \psi_{12}|12\rangle + \psi_{21}|21\rangle + \psi_{22}|22\rangle + \psi_{31}|31\rangle + \psi_{32}|32\rangle$$

$$d_{AB} = d_A \cdot d_B = 6$$

eigs (M, $n_{ev} = 2$, which = SR)

Google Documents/2018juliacourse/Guifre 2018-comp-phys-ExactDiag4-co 2018-comp-phys-ExactDiag3-co localhost:8888/notebooks/Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag4-complete.ipynb

jupyter 2018-comp-phys-ExactDiag4-complete (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

Run Markdown

Perimeter Institute Nov 23rd

2018 computational physics course

Exact Diagonalization

Guifre Vidal

IJulia nb4: quantum entanglement

```
In [3]: using PyPlot
using LinearAlgebra
using Arpack
```

1:34 PM 23/11/2018

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag4-complete.ipynb

jupyter 2018-comp-phys-ExactDiag4-complete (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

Run Code

Guifre Vidal

IJulia nb4: quantum entanglement

```
In [*]: using PyPlot
        using LinearAlgebra
        using Arpack
```

```
In [4]: Psi = randn(6) + im*randn(6) # complex vector
```

```
Out[4]: 6-element Array{Complex{Float64},1}:
 -1.0628211930185867 + 3.7582299766274283im
 -0.6671223192696618 - 1.505781790901663im
 -1.424755712503474 - 0.6693549772722557im
 -0.5052810121957295 - 1.3203229246796055im
 0.31735715100567324 - 0.04520142414086508im
 -0.9875028869967035 + 0.4364784125635265im
```

1:34 PM 23/11/2018

example

dimension $d_A = 3$

$d_B = 2$

$$|\Psi_{AB}\rangle = \psi_{11}|11\rangle + \psi_{12}|12\rangle \\ + \psi_{21}|21\rangle + \psi_{22}|22\rangle \\ + \psi_{31}|31\rangle + \psi_{32}|32\rangle$$

1)	1)
2)	2)
3)	

$$d_{AB} = d_A \cdot d_B = 6$$

def: product state

$$|\Psi_{AB}\rangle = |\varphi_A\rangle \otimes |\tilde{\varphi}_B\rangle$$

$$|23\rangle = |2\rangle \otimes |3\rangle$$

example

$$|11\rangle = |1\rangle \otimes |1\rangle$$

$$\frac{1}{2} (|11\rangle + |22\rangle + |21\rangle + |12\rangle)$$

$$\frac{1}{\sqrt{2}} (|11\rangle + |12\rangle) = |1\rangle \otimes \frac{1}{\sqrt{2}} (|1\rangle + |2\rangle)$$

$$= \frac{1}{\sqrt{2}} (|1\rangle + |2\rangle) \otimes \frac{1}{\sqrt{2}} (|1\rangle + |2\rangle)$$

def: entangled state

$$|\Psi_{AB}\rangle \neq |\varphi_A\rangle \otimes |\tilde{\varphi}_B\rangle$$

example

$$\frac{1}{\sqrt{2}} (|11\rangle + |22\rangle)$$

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag4-complete.ipynb

jupyter 2018-comp-phys-ExactDiag4-complete (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

Run Code

```
In [2]: Psi = randn(6) + im*randn(6) # complex vector
```

```
Out[2]: 6-element Array{Complex{Float64},1}:
 0.02071526164674701 - 0.8793503250752175im
 3.184642188269118 - 0.14011955696303274im
 0.24199811925445225 + 2.2583738210192976im
 -0.227880181159223 - 1.04726447740748im
 -1.5543915798725163 + 0.9745080465187976im
 0.8993329218356584 + 1.033528188643464im
```

```
In [3]: Psi'*Psi # already just a complex number! (not an array)
```

```
Out[3]: 22.485552478941294 + 0.0im
```

```
In [6]: Psi = Psi/sqrt(abs(Psi'*Psi)) # normalized vector
Psi'*Psi
```

```
Out[6]: 1.0000000000000002 + 0.0im
```

Windows taskbar: 1:35 PM 23/11/2018

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag4-complete.ipynb

jupyter 2018-comp-phys-ExactDiag4-complete (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

Run Code

```
-0.227880181159223 - 1.04726447740748im
-1.5543915798725163 + 0.9745080465187976im
0.8993329218356584 + 1.033528188643464im
```

In [3]: `Psi'*Psi` # *already just a complex number! (not an array)*

Out[3]: 22.485552478941294 + 0.0im

In [4]: `Psi = Psi/sqrt(abs(Psi'*Psi))` # *normalized vector*
`Psi'*Psi`

Out[4]: 1.0 + 0.0im

In [7]: `PsiAB = reshape(Psi, (3,2))` # *bipartite vector*

Out[7]: 3x2 Array{Complex{Float64},2}:
-0.218264+0.771802im -0.103766-0.271146im
-0.137002-0.309232im 0.0651735-0.00928271im
-0.292592-0.137461im -0.202797+0.0896366im

1:36 PM 23/11/2018

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag4-complete.ipynb

jupyter 2018-comp-phys-ExactDiag4-complete (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

[-] + [x] [y] [z] [a] [b] [c] [d] [e] [f] [g] [h] [i] [j] [k] [l] [m] [n] [o] [p] [q] [r] [s] [t] [u] [v] [w] [x] [y] [z] [0] [1] [2] [3] [4] [5] [6] [7] [8] [9] [code] [run] [stop] [refresh] [back] [forward] [help]

```
-0.227880181159223 - 1.04726447740748im
-1.5543915798725163 + 0.9745080465187976im
0.8993329218356584 + 1.033528188643464im
```

In [3]: `Psi'*Psi` # *already just a complex number! (not an array)*

Out[3]: 22.485552478941294 + 0.0im

In [4]: `Psi = Psi/sqrt(abs(Psi'*Psi))` # *normalized vector*
`Psi'*Psi`

Out[4]: 1.0 + 0.0im

In [5]: `PsiAB = reshape(Psi, (3,2))` # *bipartite vector*

Out[5]: 3x2 Array{Complex{Float64},2}:
0.00436856-0.185443im -0.0480568-0.220854im
0.671597-0.0295493im -0.3278+0.20551im
0.0510341+0.47626im 0.189657+0.217957im

Windows taskbar: 1:38 PM 23/11/2018

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag4-complete.ipynb

2018-comp-phys-ExactDiag4-complete (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

Run Code

```
-0.227880181159223 - 1.04726447740748im
-1.5543915798725163 + 0.9745080465187976im
0.8993329218356584 + 1.033528188643464im
```

In [3]: `Psi'*Psi # already just a complex number! (not an array)`

Out[3]: `22.485552478941294 + 0.0im`

In [4]: `Psi = Psi/sqrt(abs(Psi'*Psi)) # normalized vector`
`Psi'*Psi`

Out[4]: `1.0 + 0.0im`

In [5]: `PsiAB = reshape(Psi, (3,2)) # bipartite vector`

Out[5]: `3x2 Array{Complex{Float64},2}:`
`0.00436856-0.185443im -0.0480568-0.220854im`
`0.671597-0.0295493im -0.3278+0.20551im`
`0.0510341+0.47626im 0.189657+0.217957im`

1:38 PM 23/11/2018

Q.M. bipartite system AB $V_{AB} \cong V_A \otimes V_B$

$$\{ |c_j\rangle = |c_A\rangle \otimes |c_B\rangle \} \quad \{ |c_A\rangle \} \quad \{ |c_B\rangle \}$$

A B
 = ⊗ =
 11) 11)
 12) 12)
 13) 12)

example
 dimension $d_A = 3$
 $d_B = 2$

$$|\Psi_{AB}\rangle = \psi_{11}|11\rangle + \psi_{12}|12\rangle + \psi_{21}|21\rangle + \psi_{22}|22\rangle + \psi_{31}|31\rangle + \psi_{32}|32\rangle$$

$$d_{AB} = d_A \cdot d_B = 6$$

$(\text{P.S.}, (3,2))$ $|11\rangle = |1\rangle \otimes |1\rangle$
 $\frac{1}{\sqrt{2}}(|11\rangle + |12\rangle) = |1\rangle \otimes \frac{1}{\sqrt{2}}(|1\rangle + |2\rangle)$

example $|\Psi_{AB}\rangle \neq |\varphi_A\rangle \otimes |\tilde{\varphi}_B\rangle$
 $\frac{1}{\sqrt{2}}(|11\rangle + |22\rangle)$

Q.M. bipartite system AB

$$V_{AB} \cong V_A \otimes V_B$$

$$|\psi\rangle_{AB} = \sum_{ij} \psi_{ij} |i_A\rangle |j_B\rangle$$

reduced density matrix

$$\rho_A \equiv \text{tr}_B |\psi\rangle\langle\psi| = \sum_j \langle j_B | (|\psi\rangle\langle\psi|) |j_B\rangle = \sum_{m,n} (\rho_A)_{m,n} |m_A\rangle\langle n_A|$$

$$(\rho_A)_{m,n} = \sum_j \psi_{mj} \psi_{nj}^*$$

$$\psi \psi^\dagger = \rho_A$$

$$\begin{pmatrix} \vdots \\ \vdots \\ \vdots \end{pmatrix} \begin{pmatrix} \vdots & \vdots & \vdots \end{pmatrix} = \begin{pmatrix} \circ & \circ & \circ \\ \circ & \circ & \circ \\ \circ & \circ & \circ \end{pmatrix}$$

entanglement entropy of $|\psi_{AB}\rangle$

$$S[\rho_A] \equiv -\text{tr}(\rho_A \log_2 \rho_A)$$

convention

$$= -\sum_{\alpha} p_{\alpha} \log_2 p_{\alpha}$$

p_{α} eigenvalues of ρ_A

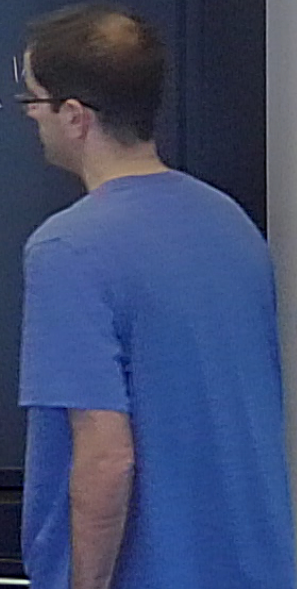
$$|\psi_{AB}\rangle = |1_A\rangle \otimes |\tilde{\varphi}_B\rangle$$

$$|1\ 1\rangle = |1\rangle \otimes |1\rangle$$

$$\frac{1}{\sqrt{2}}(|11\rangle + |12\rangle) = |1\rangle \otimes \frac{1}{\sqrt{2}}(|1\rangle + |2\rangle)$$

$$|\psi_{AB}\rangle \neq |1_A\rangle \otimes |\tilde{\varphi}_B\rangle$$

$$\frac{1}{\sqrt{2}}(|11\rangle + |22\rangle)$$



Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag4-complete.ipynb

jupyter 2018-comp-phys-ExactDiag4-complete (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

Run Code

```
In [3]: Psi'*Psi # already just a complex number! (not an array)
Out[3]: 22.485552478941294 + 0.0im

In [4]: Psi = Psi/sqrt(abs(Psi'*Psi)) # normalized vector
Psi'*Psi
Out[4]: 1.0 + 0.0im

In [5]: PsiAB = reshape(Psi, (3,2)) # bipartite vector
Out[5]: 3x2 Array{Complex{Float64},2}:
 0.00436856-0.185443im -0.0480568-0.220854im
 0.671597-0.0295493im -0.3278+0.20551im
 0.0510341+0.47626im 0.189657+0.217957im

In [8]: rdmA = PsiAB*PsiAB' # reduced density matrix for part A
rdmB = PsiAB'*PsiAB # reduced density matrix for part B
(size(PsiAB), size(rdmA), size(rdmB))
Out[8]: ((3, 2), (3, 3), (2, 2))
```

1:48 PM 23/11/2018

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag4-complete.ipynb

2018-comp-phys-ExactDiag4-complete (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

Run Code

```
In [4]: Psi = Psi/sqrt(abs(Psi'*Psi)) # normalized vector
Psi'*Psi

Out[4]: 1.0 + 0.0im

In [6]: PsiAB = reshape(Psi, (3,2)) # bipartite vector

Out[6]: 3x2 Array{Complex{Float64},2}:
 0.00436856-0.185443im  -0.0480568-0.220854im
 0.671597-0.0295493im  -0.3278+0.20551im
 0.0510341+0.47626im   0.189657+0.217957im

In [7]: rdmA = PsiAB*PsiAB' # reduced density matrix for part A
rdmB = PsiAB'*PsiAB # reduced density matrix for part B
(size(PsiAB), size(rdmA), size(rdmB))

Out[7]: ((3, 2), (3, 3), (2, 2))

In [9]: tr(rdmA) # checking that the trace is 1
```

1:49 PM 23/11/2018

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag4-complete.ipynb

jupyter 2018-comp-phys-ExactDiag4-complete (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

```
nA = 4 # number spins/qubits in A
nB = 5 # number of spins/qubits in B
Psi = randn(2^(nA+nB)) + im*randn(2^(nA+nB))
Psi = Psi/sqrt(Psi'*Psi)
PsiAB = reshape(Psi, (2^(nA),2^(nB)))
rdmA = compute_rdmA(PsiAB)
eigvals(rdmA)
```

Out[10]: 16-element Array{Float64,1}:

```
0.00883168828727382
0.013958842374827975
0.017931727477443433
0.022452254737674638
0.029736609027394414
0.0392657430384324
0.04136933103094236
0.04913131294877505
0.05972979128900736
0.07029279344988208
0.081655252228140082
```

1:50 PM 23/11/2018

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag4-empt... 2018-comp-phys-ExactDiag4-empt... 2018-comp-phys-ExactDiag4-co... 2018-comp-phys-ExactDiag3-co... localhost:8888/notebooks/Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag4-empty.ipynb

jupyter 2018-comp-phys-ExactDiag4-empty (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Julia 1.0.0

Run Markdown

1) Given a bipartite wavefunction Ψ_{AB} , compute the reduced density matrix $\rho_A \equiv \text{Tr}_B(|\Psi_{AB}\rangle\langle\Psi_{AB}|)$

```
In [ ]: function compute_rdmA(PsiAB)
    @assert ndims(PsiAB) == 2 # did we input a matrix?
    @assert size(PsiAB,1) < 2^13 # make sure the resulting matrix is not too large
    rdm = PsiAB*PsiAB'
end

nA = 4 # number spins/qubits in A
nB = 5 # number of spins/qubits in B
Psi = randn(2^(nA+nB)) + im*randn(2^(nA+nB))
Psi = Psi/sqrt(Psi'*Psi)
PsiAB = reshape(Psi, (2^(nA),2^(nB)))
rdmA = compute_rdmA(PsiAB)
eigvals(rdmA)
```

2) Given a ... compute its entropy

1:51 PM 23/11/2018

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag4-complete.ipynb

jupyter 2018-comp-phys-ExactDiag4-complete (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

0.11759673589533899
0.1362892899492188
0.17330311891747938

2) Given ρ_A , compute its entropy

```
In [11]: function compute_entropy(dm)
p = eigvals(dm)
entropy = 0.0
#entropy = - p'*Log2.(p) # problems when p[n] is of order 0 (possibly negativ
for n in 1:size(p,1)
    if abs(p[n]) > 1e-12
        entropy = entropy - p[n]*log2(p[n])
    end
end
entropy
end

compute_entropy(rdmA)
```

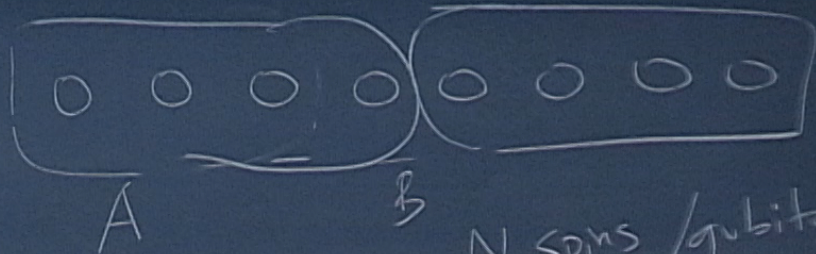
1:56 PM 23/11/2018

entanglement entropy of $|\psi_{AB}\rangle$

$$S[\rho_A] \equiv -\text{tr}(\rho_A \log_2 \rho_A)$$

convention

$$\lim_{\epsilon \rightarrow 0} \epsilon \log \epsilon = 0$$



N spins / qubits
1/2

$$\equiv -\sum_{\alpha} p_{\alpha} \log_2 p_{\alpha} \geq 0$$

p_{α} eigenvalues
of ρ_A

$$|\psi_{AB}\rangle \neq |\psi_A\rangle \otimes |\tilde{\psi}_B\rangle$$

$$\frac{1}{\sqrt{2}}(|11\rangle + |22\rangle)$$

localhost:8888/notebooks/Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag4-empty.ipynb

jupyter 2018-comp-phys-ExactDiag4-empty (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Julia 1.0.0

```
compute_entropy(rdmA)
```

3) write a function `compute_entropies` that takes a state $|\Psi\rangle$ of N spins, and outputs the entropy of the reduced density matrix on $n = 1, 2, \dots, N/2$ spins

```
In [ ]: function compute_entropies(Psi)
    @assert ndims(Psi) == 1
    N = floor{Int64, log2(size(Psi, 1) + 1e-5)}
    # your code comes here
    entropies
end
```

4) for N spins ($N \geq 12$) create ground state of Ising model (for $\theta = \pi/2$ and $\theta = \pi/3$) as well as a random state

1:58 PM 23/11/2018

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag4-complete.ipynb

jupyter 2018-comp-phys-ExactDiag4-complete Last Checkpoint: 6 hours ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

x	Blue Series (y)	Yellow Series (y)
1	0.68	0.27
2	0.85	0.32
3	0.93	0.33
4	0.98	0.34
5	1.00	0.34
6	1.01	0.34

```
In [22]: plot(collect(1:size(entropiesRand,1)),collect(1:size(entropiesRand,1)), color = "r")
plot(collect(1:size(entropiesRand,1)),entropiesRand, marker = "o", color = "k")
plot(collect(1:size(entropiesIsingCrit,1)),entropiesIsingCrit, marker = "o", color = "b");
plot(collect(1:size(entropiesIsing2,1)),entropiesIsing2, marker = "o", color = "y");
```

1:59 PM 23/11/2018

entanglement entropy of $|\Psi_{AB}\rangle$

$$S[\rho_A] \equiv -\text{tr}(\rho_A \log_2 \rho_A)$$

convention

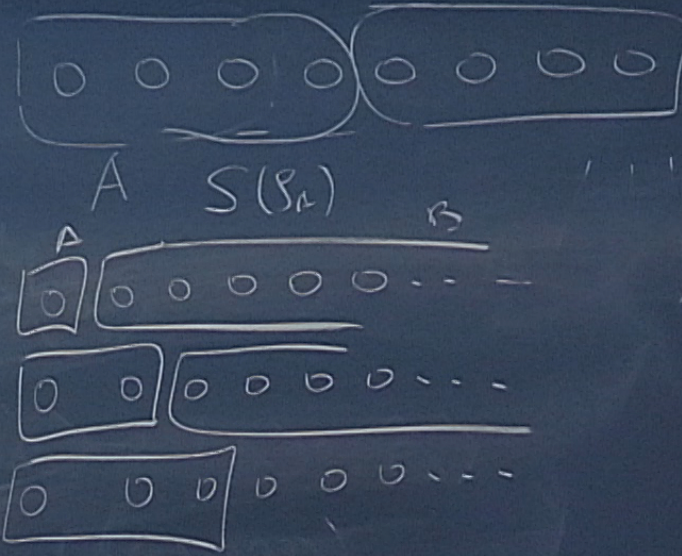
$$\equiv -\sum_{\alpha} p_{\alpha} \log_2 p_{\alpha}$$

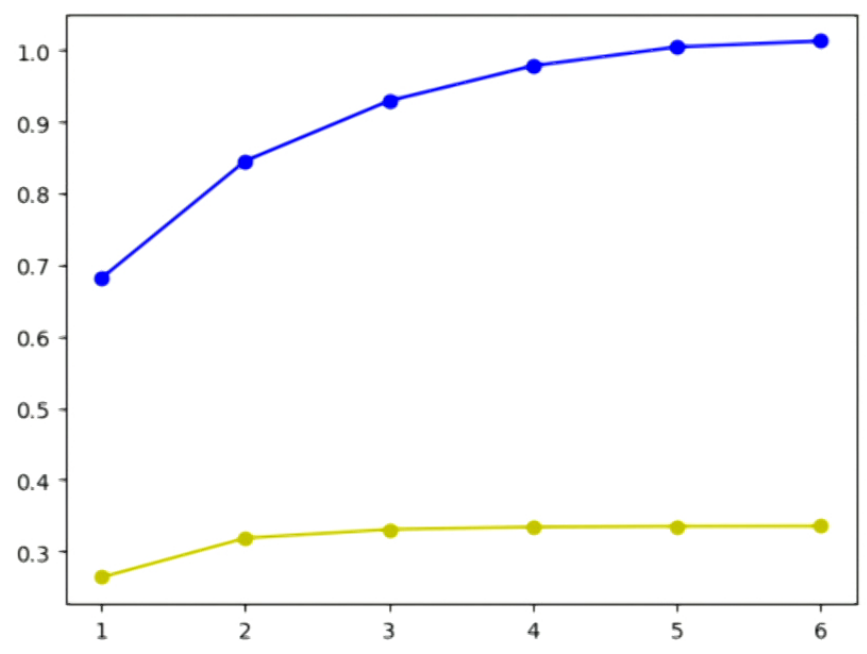
p_{α} eigenvalues
of ρ_A

$n_A=1$

$n_A=2$

$n_A=3$





```
In [22]: plot(collect(1:size(entropiesRand,1)),collect(1:size(entropiesRand,1)), color = "r")
plot(collect(1:size(entropiesRand,1)),entropiesRand, marker = "o", color = "k")
plot(collect(1:size(entropiesIsingCrit,1)),entropiesIsingCrit, marker = "o", color = "b");
plot(collect(1:size(entropiesIsing2,1)),entropiesIsing2, marker = "o", color = "y");
```

localhost:8888/notebooks/Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag4-empty.ipynb

jupyter 2018-comp-phys-ExactDiag4-empty Last Checkpoint: 6 hours ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted | Julia 1.0.0

3) write a function `compute_entropies` that takes a state $|\Psi\rangle$ of N spins, and outputs the entropy of the reduced density matrix on $n = 1, 2, \dots, N/2$ spins

```
In [ ]: function compute_entropies(Psi)
    @assert ndims(Psi) == 1
    N = floor(Int64, log2(size(Psi,1) + 1e-5))
    # your code comes here
    entropies
end
```

4) for N spins ($N \geq 12$) create ground state of Ising model (for $\theta = \pi/2$ and $\theta = \pi/3$), as well as a random state

```
In [ ]: # Here is buildH from previous Lectures
function buildH(N::Int64, theta::Float64)::Array{Float64,2}
    X = [0. 1; 1 0]
    Z = [1. 0; 0 -1]
    E = diagm(0=>ones(2))
    XX = kron(X,X)
    HXX = XX
    HZ = kron(Z,E) + kron(E,Z)
    for n = 3:N
        HXX = kron(HXX,E)+kron(diagm(0=>ones(2^(n-2))), XX)
        HZ = kron(HZ,E) + kron(diagm(0=>ones(2^(n-1))), Z)
    end
    HXX = HXX + kron(X,kron(diagm(0=>ones(2^(N-2))),X))
end
```

2:01 PM 23/11/2018

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag4-empty.ipynb 2018-comp-phys-ExactDiag4-en... 2018-comp-phys-ExactDiag4-co... 2018-comp-phys-ExactDiag3-co... local:localhost:8888/notebooks/Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag4-empty.ipynb Apps Frequent bookmarks Bookmarks Topological Phases fendley www.lancaster.ac.uk Symmetry and Topo David Tong -- Cambi CFT-2014.pdf AdSCFTCourseNote

jupyter 2018-comp-phys-ExactDiag4-empty Last Checkpoint: 6 hours ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Julia 1.0.0

Run Code

3) write a function `compute_entropies` that takes a state $|\Psi\rangle$ of N spins, and outputs the entropy of the reduced density matrix on $n = 1, 2, \dots, N/2$ spins

```
In [ ]: function compute_entropies(Psi)
    @assert ndims(Psi) == 1
    N = floor(Int64, log2(size(Psi,1) + 1e-5))
    # your code comes here
    entropies
end
```

4) for N spins ($N \geq 12$) create ground state of Ising model (for $\theta = \pi/2$ and $\theta = \pi/3$), as well as a random state ¶

```
In [ ]: # Here is buildH from previous lectures
function buildH(N::Int64, theta::Float64)::Array{Float64,2}
    X = [0. 1; 1 0]
    Z = [1. 0; 0 -1]
    E = diagm(0=>ones(2))
    XX = kron(X,X)
    HXX = XX
```

2:17 PM 23/11/2018

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag4-empty.ipynb

jupyter 2018-comp-phys-ExactDiag4-empty Last Checkpoint: 6 hours ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Julia 1.0.0

4) for N spins ($N \geq 12$) create ground state of Ising model (for $\theta = \pi/2$ and $\theta = \pi/3$), as well as a random state

```
In [ ]: # Here is buildH from previous lectures
function buildH(N::Int64,theta::Float64)::Array{Float64,2}
    X = [0. 1; 1 0]
    Z = [1. 0; 0 -1]
    E = diagm(0=>ones(2))
    XX = kron(X,X)
    HXX = XX
    HZ = kron(Z,E) + kron(E,Z)
    for n = 3:N
        HXX = kron(HXX,E)+kron(diagm(0=>ones(2^(n-2))), XX)
        HZ = kron(HZ,E) + kron(diagm(0=>ones(2^(n-1))),Z)
    end
    HXX = HXX + kron(X,kron(diagm(0=>ones(2^(N-2))),X))
    H = -cos(theta)*HXX - sin(theta)*HZ
    return H
end
```

```
In [ ]:
```

2:18 PM 23/11/2018

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag4-complete.ipynb

jupyter 2018-comp-phys-ExactDiag4-complete Last Checkpoint: 6 hours ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

```
In [20]: entropiesRand = compute_entropies(PsiRand)
entropiesIsingCrit = compute_entropies(PsiIsingCrit)
entropiesIsing2 = compute_entropies(PsiIsing2)
# Rand entropies
plot(collect(1:size(entropiesRand,1)),entropiesRand, marker = "o", color = "k");
```

Index	Entropy Value
1	2.0
2	3.0
3	4.0
4	5.2

Windows taskbar: 2:38 PM 23/11/2018

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag4-complete.ipynb

jupyter 2018-comp-phys-ExactDiag4-complete Last Checkpoint: 6 hours ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

Run Markdown

x	y
1	1.0
2	2.0
3	3.0
4	3.9
5	4.8
6	5.3

Windows taskbar: 2:38 PM 23/11/2018

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag4-complete.ipynb

jupyter 2018-comp-phys-ExactDiag4-complete Last Checkpoint: 6 hours ago (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

```
plot(collect(1:size(entropiesIsing1,1)),entropiesIsing1, marker = "o", color = "b"),  
plot(collect(1:size(entropiesIsing2,1)),entropiesIsing2, marker = "o", color = "y");
```

Series	Point 1	Point 2	Point 3	Point 4	Point 5	Point 6
Blue (entropiesIsing1)	0.68	0.85	0.93	0.98	1.00	1.01
Yellow (entropiesIsing2)	0.27	0.32	0.33	0.34	0.34	0.34

Google Documents/2018juliacourse/Guifre/2018-comp-phys-ExactDiag4-complete.ipynb

jupyter 2018-comp-phys-ExactDiag4-complete Last Checkpoint: 6 hours ago (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Julia 1.0.0

x	Black Series	Blue Series	Yellow Series	Red Line
1	1.0	0.7	0.3	1.0
2	2.0	0.85	0.35	2.0
3	3.0	0.95	0.35	3.0
4	4.0	1.0	0.35	4.0
5	4.8	1.0	0.35	5.0
6	5.3	1.0	0.35	6.0