Title: Classical and Quantum Chaos 2021/2022 - Lecture 12

Speakers: Meenu Kumari

Collection: Classical and Quantum Chaos 2021/2022

Date: March 25, 2022 - 11:30 AM

URL: https://pirsa.org/22030056

1. Why RMT (Wigner, Dyson)

2. Bohias – Giannoni – Schmidt. (BGS) conjecture.

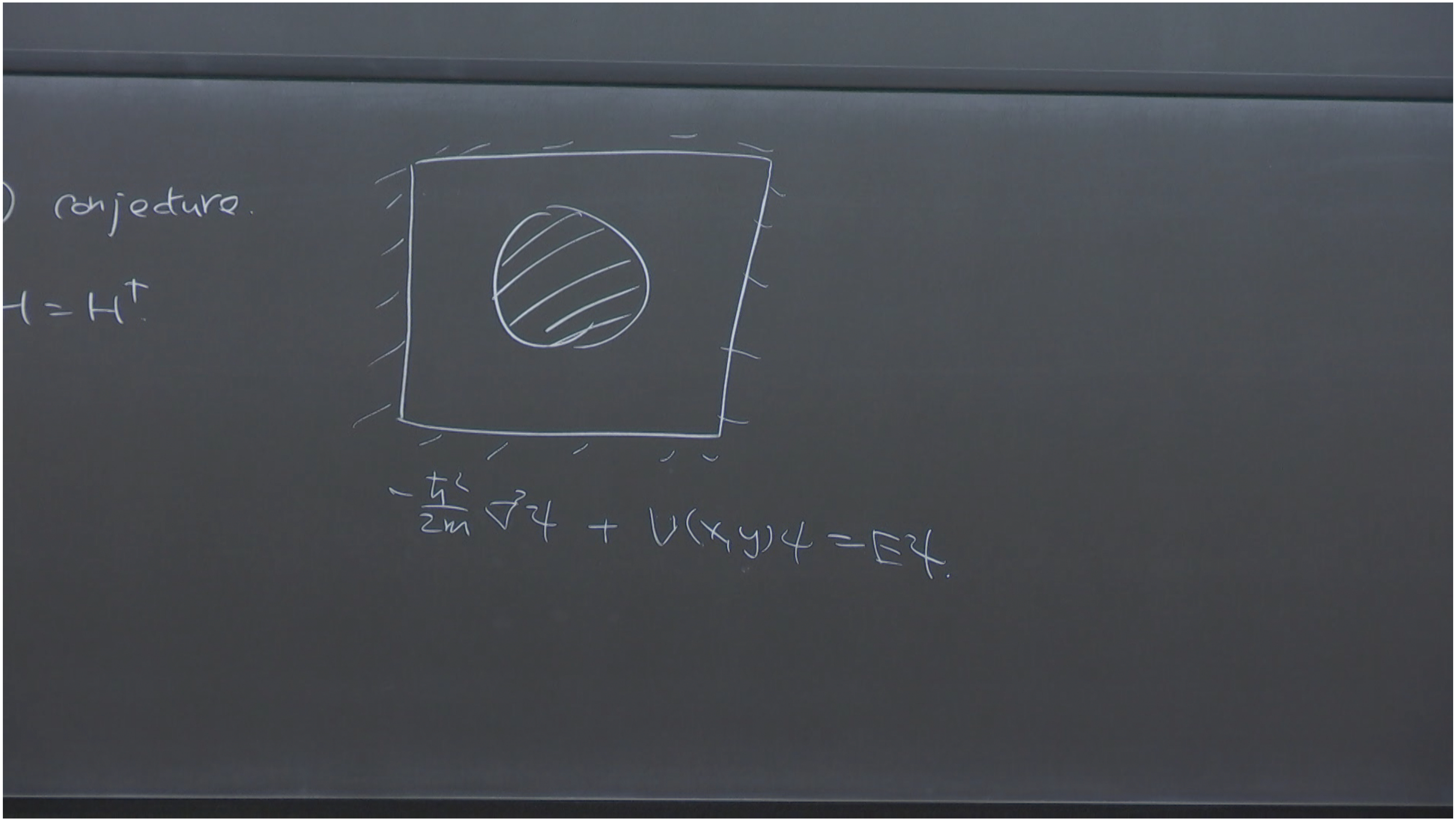
1. Why RMT? (Wigner, Dyson)

2. Bohias – Giannoni – Schmidt. (BGS) conjecture.

3. What is RMT

$$M = \begin{pmatrix} m_{11} & \cdots & m_{1N} \\ \vdots & & \vdots \\ m_{N1} & \cdots & m_{NN} \end{pmatrix}$$

- Hamiltonian $H = H^{\dagger}$


conjecture.
$H = H^\dagger$.
$-\frac{\hbar^2}{2m}\nabla^2\psi + V(x,y)\psi = E\psi$.

(1). H real. $H^T = H$ (real-sym.)

$N \times N$. $\exists$

$$H = \begin{bmatrix} H_{11} & \cdots & H_{1N} \\ & \ddots & \\ & & H_{NN} \end{bmatrix}$$

$$P(H) = \prod_{k \le j} f_{kj}(H_{kj})$$

$1 + 2 \ldots N$

$= \frac{1}{2} N(N+1)$ d.o.f.

(2). H general. $H = H^{\dagger}$

$$P(H) = \prod_{k<j} f^{(real)}_{k<j}\left(H^{(real)}_{kj}\right) f^{(imag)}_{kj}\left(H^{(im)}_{kj}\right) \prod_{j} f^{(real)}_{jj}\left(H^{(real)}_{jj}\right)$$

$$2 \times \frac{1}{2} N(N-1) + N = N^2$$

(I). H real. $H^T = H$ (real-sym.)

$N \times N$ $\Rightarrow$

$$H = \begin{bmatrix} H_{11} & \cdots & H_{1N} \\ & \ddots & \\ & & H_{NN} \end{bmatrix}$$

$$P(H) = \prod_{k \le j} f_{kj}(H_{kj})$$

$1 + 2 \ldots N$

$= \frac{1}{2} N(N+1)$ d.o.f.

- Gaussian Orthogonal Ensemble (GOE) (case (I))

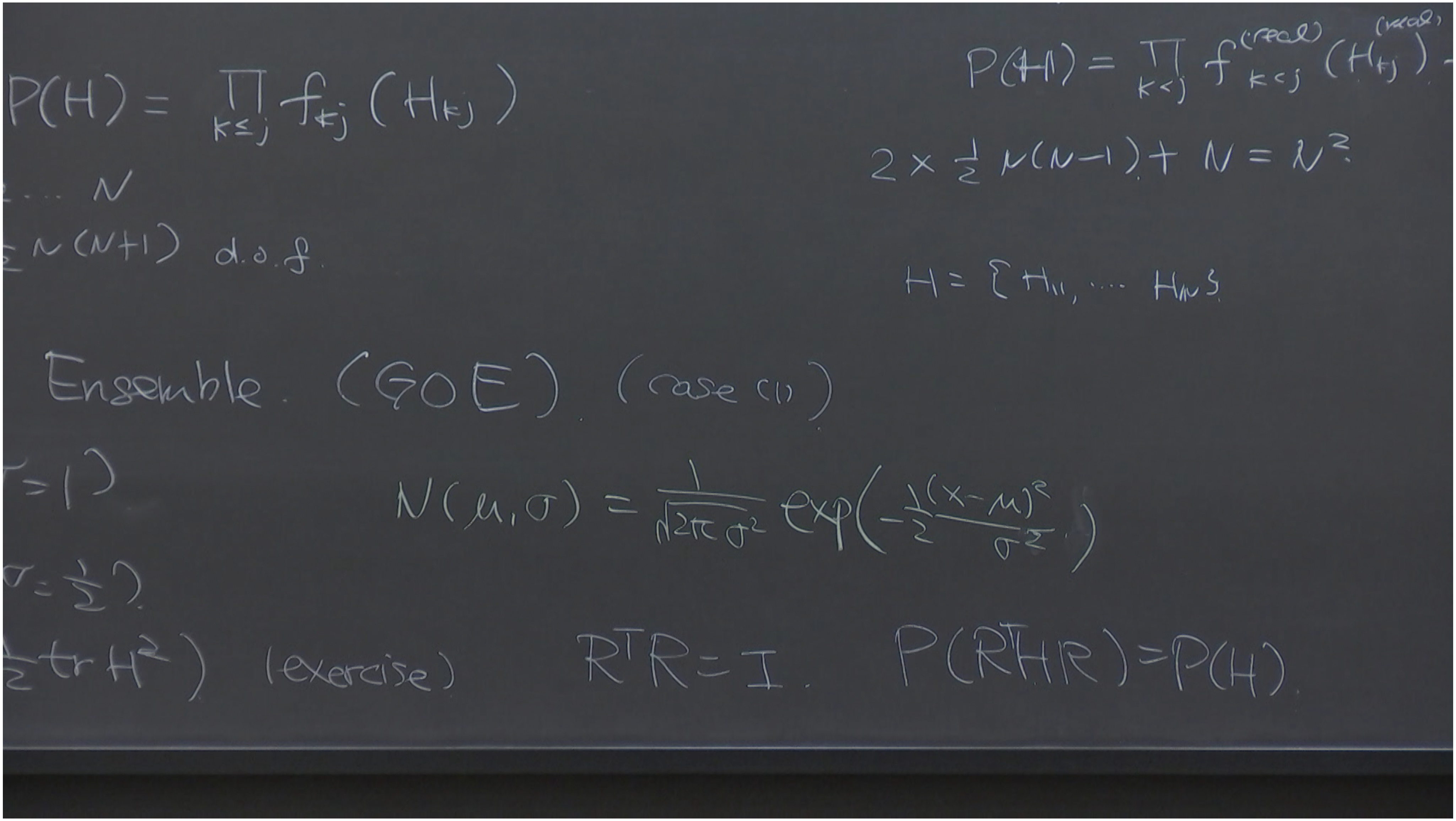
$f_{jj}(x) \sim N(\mu = 0, \sigma = 1)$

$f_{k<j}(x) \sim N(\mu = 0, \sigma = \frac{1}{2})$

$N(\mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e$

$\Rightarrow P(H) \propto \exp(-\frac{1}{2} \mathrm{tr} H^2)$ (exercise)

$R^T R =$



P(H) = ∏_{k≤j} f_{kj}(H_{kj})
... N
N(N+1) d.o.f.
P(H) = ∏_{k<j} f^{(real)}_{k<j}(H_{kj}^{(real)})
2 × ½ N(N−1) + N = N²
H = {H_11, ... H_NN}
Ensemble (GOE) (case (1))
σ = 1)
N(μ, σ) = 1/√(2πσ²) exp(−½ (x−μ)²/σ²)
σ = ½)
tr H²) (exercise)
RᵀR = I
P(RᵀHR) = P(H)

- Gaussian Unitary Ensemble (GUE)

$$f_{jj} \sim N(0,1)$$

$$f^{(\text{real})}_{k<j},\ f^{(\text{imag})}_{k<j} \sim N\left(0, \tfrac{1}{2}\right)$$

$$\Rightarrow P(H) \propto \exp\left(-\mathrm{tr}\, H^2\right)$$

SE)

$$P(H) \propto \exp\left(-a\,\mathrm{tr}H^2 - b\,\mathrm{tr}H - c\right)$$

- Level-spacing stat. (Poor man's derivation)

$$H_{2\times 2} = \begin{bmatrix} a & b/\sqrt{2} \\ b/\sqrt{2} & c \end{bmatrix} \qquad a, b, c \sim N(0,1)$$

(GOE)

$$\lambda_{\pm} = \frac{1}{2}(a+c) \pm \frac{1}{2}\sqrt{(a-c)^2 + 2b^2}$$

$$\omega = \lambda_+ - \lambda_- = \sqrt{(a-c)^2 + 2b^2}$$

$$P(\omega) = \frac{1}{(2\pi)^{3/2}} \int da\, db\, dc\, \delta\left(\omega - \sqrt{(a-c)^2 + 2b^2}\right) \exp\left(-\frac{1}{2}(a^2 + b^2 + c^2)\right)$$

- Level-spacing stat. (Poor man's derivation)

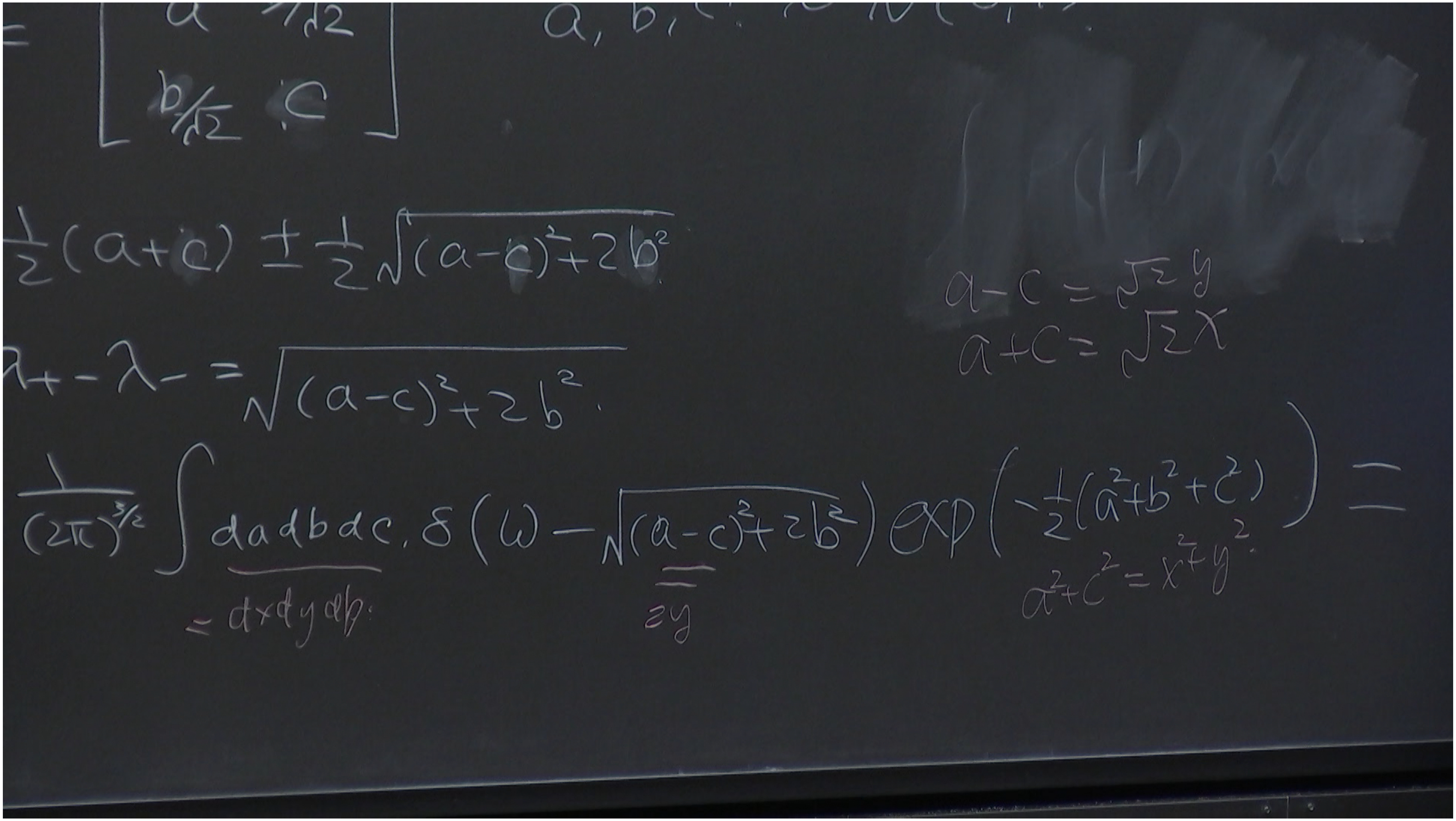
$$H_{2\times 2} = \begin{bmatrix} a & b/\sqrt{2} \\ b/\sqrt{2} & c \end{bmatrix} \qquad a, b, c \sim N(0,1)$$

(GOE)

$$\lambda_\pm = \frac{1}{2}(a+c) \pm \frac{1}{2}\sqrt{(a-c)^2+2b^2}$$

$$\omega = \lambda_+ - \lambda_- = \sqrt{(a-c)^2+2b^2}$$

$$P(\omega) = \frac{1}{(2\pi)^{3/2}} \int da\, db\, dc.\ \delta\left(\omega - \sqrt{(a-c)^2+2b^2}\right) \exp\left(-\frac{1}{2}(a^2+b^2+c^2)\right)$$


a, b, c
1/2 (a+c) ± 1/2 √((a−c)² + 2b²)
λ+ − λ− = √((a−c)² + 2b²)
a−c = √2 y
a+c = √2 x
1/(2π)^{3/2} ∫ da db dc . δ(ω − √((a−c)² + 2b²)) exp(−1/2 (a² + b² + c²)) =
= dx dy db
2y
a² + c² = x² + y²

$$\overline{\ldots - c)^2 + 2b^2}$$

$$\overline{\ldots} \; 2b^2.$$

$$a - c = \sqrt{2}\, y$$

$$a + c = \sqrt{2}\, x$$

$$\omega - \sqrt{(a-c)^2 + 2b^2}\,\Big) \exp\!\left(-\tfrac{1}{2}(a^2 + b^2 + c^2)\right) = \tfrac{1}{2}\omega \exp\!\left(-\frac{\omega^2}{4}\right).$$

$$2y$$

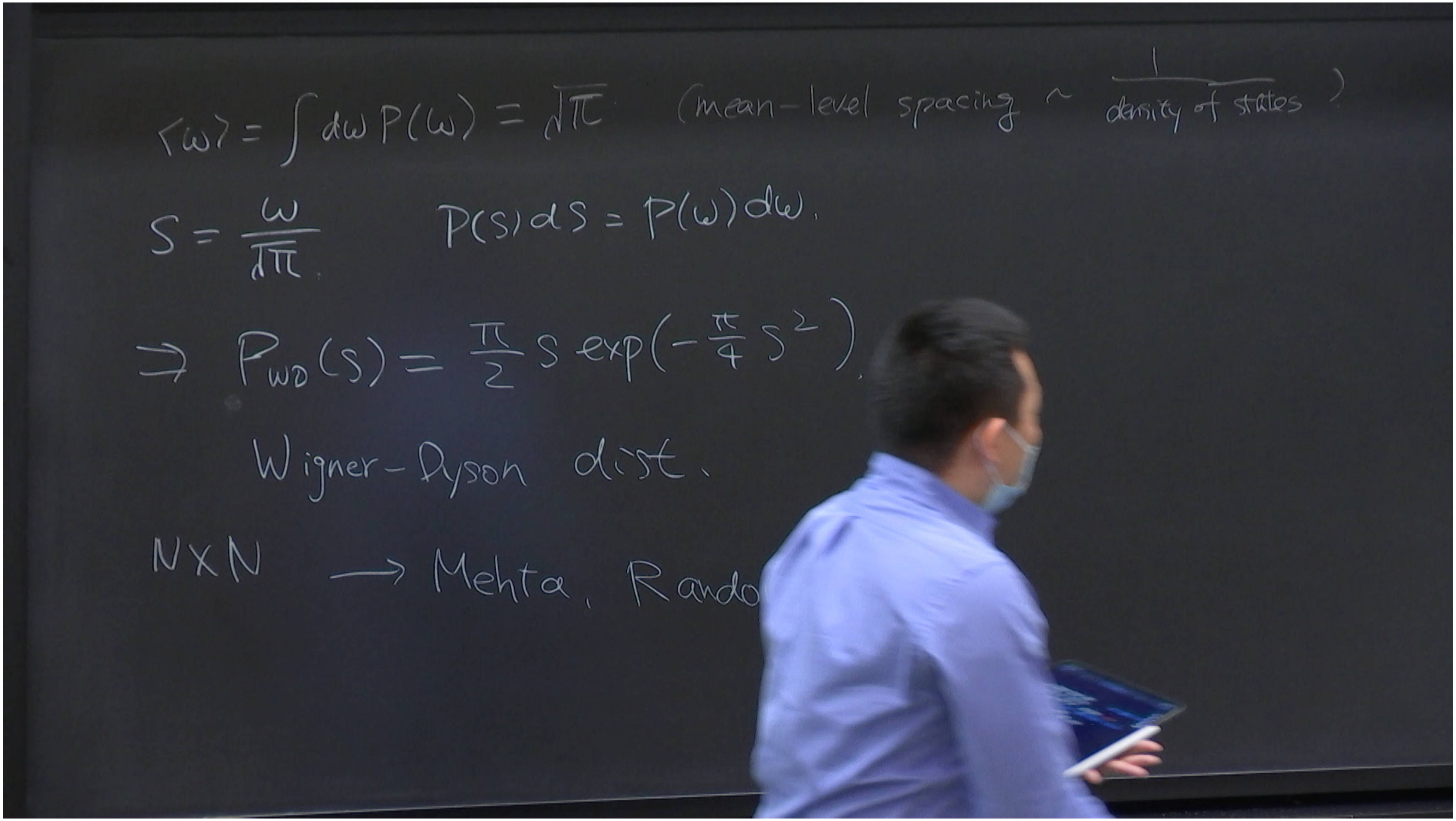
$$a^2 + c^2 = x^2 + y^2.$$

$$\langle \omega \rangle = \int d\omega \, P(\omega) = \sqrt{\pi} \quad \left(\text{mean-level spacing} \sim \frac{1}{\text{density of states}} \right)$$

$$s = \frac{\omega}{\sqrt{\pi}} \qquad P(s)\, ds = P(\omega)\, d\omega .$$

$$\Rightarrow P_{WD}(s) = \frac{\pi}{2} s \exp\left(-\frac{\pi}{4} s^2 \right).$$

Wigner-Dyson dist.


$\langle \omega \rangle = \int d\omega \, P(\omega) = \sqrt{\pi}$ (mean-level spacing ~ $\frac{1}{\text{density of states}}$)
$S = \frac{\omega}{\sqrt{\pi}}$ $P(S)\,dS = P(\omega)\,d\omega$.
$\Rightarrow P_{WD}(S) = \frac{\pi}{2} S \exp\left(-\frac{\pi}{4} S^2\right)$
Wigner-Dyson dist.
N x N → Mehta, Rando

Level-spacing demo

```julia
using DelimitedFiles,Statistics,Plots,CurveFit
```

Load data

The eigenenergies are obtained from the Hamiltonian given in https://arxiv.org/abs/2005.07048.

```julia
filename="EigenvaluesJ30Mu0Pt5.csv"
eigvalsμ05=readdlm(filename)[:,1];
filename="EigenvaluesJ30Mu1Pt4.csv"
eigvalsμ14=readdlm(filename)[:,1];
filename="EigenvaluesJ30Mu1Pt85.csv"
eigvalsμ185=readdlm(filename)[:,1];
filename="EigenvaluesJ30Mu2Pt5.csv"
eigvalsμ25=readdlm(filename)[:,1];
```

Unfold the spectrum

The distributions of the level-spacing statistics describing the energy spacings when the density of states is normalized to one. Given a spectrum (obtained numerically or experimentally), we therefore need to normalize the density of states to have a meaningful level-spacing statistics. This procedure is called "unfolding" the spectrum. The unfolding procedure is not unique, and some of the statistical properties might be more sensitve to different procedures. However, in general, the level-spacing statistics is less sensitive to it. If one have an analytical desciption or some guess of the functional form of the density of states, then one can fit the eigenenergy data with such a functional form. Here, we use the "polynomial fitting" procedure which is a popular procudure when no known functional form of the density of states is given.

The unfolding procedure is summarized in the following.

1. Take the middle $1/2$ or some other fraction of the spctrum (why?) and form the cummulated density of states $G(E)=\sum_n \Theta(E-E_n)$, where $\Theta(x)$ is the Heaviside step function. (This function counts how many eigenstates have the energies below E.)
2. Use a polynomial of degree N: $f_N(x)=\sum_{m=0}^N f_m x^m$ to fit $G(E)$ via least square fitting.

(Dated: September 2, 2020)

We consider a coupled top model describing two interacting large spins, which is studied semiclassically as well as quantum mechanically. This model exhibits variety of interesting phenomena such as quantum phase transition (QPT), dynamical transition and excited state quantum phase transitions above a critical coupling strength. Both classical dynamics and entanglement entropy reveals ergodic behavior at the center of energy density band for an intermediate range of coupling strength above QPT, where the level spacing distribution changes from Poissonian to Wigner-Dyson statistics. Interestingly, in this model we identify quantum scars as reminiscence of unstable collective dynamics even in presence of interaction. Statistical properties of such scarred states deviate from ergodic limit corresponding to random matrix theory and violate Berry's conjecture. In contrast to ergodic evolution, oscillatory behavior in dynamics of unequal time commutator and survival probability is observed as dynamical signature of quantum scar, which can be relevant for its detection.

PACS numbers:

Introduction: After the recent experiment on a chain of Rydberg atoms [1], quantum scar (QS) in many body systems has drawn significant interest due to its connection with ergodicity and non equilibrium dynamics. Observed athermal behavior and periodic revival phenomena in dynamics of a specific initial state has been attributed to the many body quantum scarring phenomena [2–8]. Many body quantum scar (MBQS) gives rise to the deviation from ergodicity [2–22] leading to the violation of eigenstate thermalization hypothesis (ETH) [23] in quantum systems [5, 8–12]. Recent theoretical studies on interacting spin systems reveal that emergent symmetries and symmetry protected many body states are the key ingre-

field Ising model [40, 41], which is governed by the Hamiltonian,

$$\hat{\mathcal{H}} = -\hat{S}_{1x} - \hat{S}_{2x} - \frac{\mu}{S}\hat{S}_{1z}\hat{S}_{2z}, \qquad (1)$$

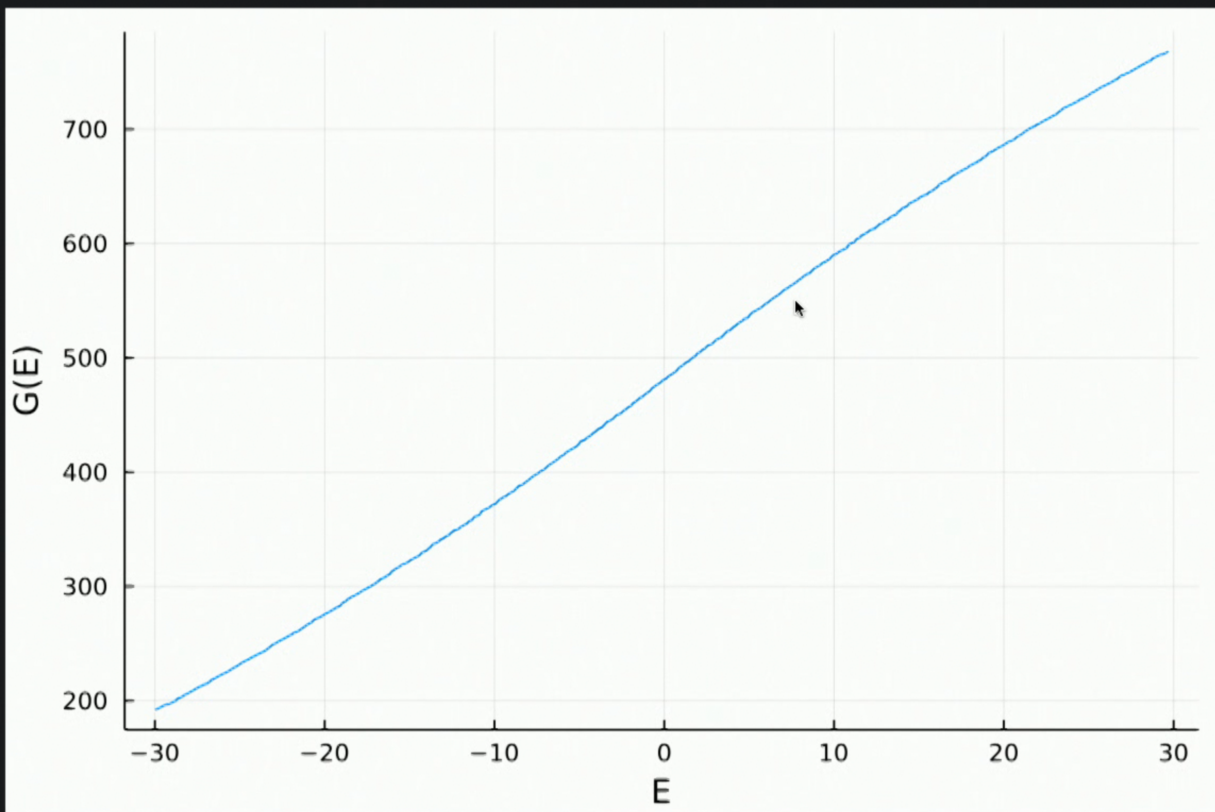
where $\hat{S}_{ia}$ represents components ($a = x, y, z$) of two large spins of equal magnitude S denoted by index $i = 1, 2$ and μ is the ferromagnetic coupling strength.

Large magnitude of spin $S \gg 1$ allows us to analyze the model semiclassically, where the spin vectors are represented by $\vec{S}_i = (S\sin\theta_i\cos\phi_i, S\sin\theta_i\sin\phi_i, S\cos\theta_i)$. In terms of these dynamical variables, corresponding classical Hamiltonian can be written as, Dropbox (PI)/Proj... | localhost | https://arxiv.org/p...

iCloud Facebook PI YouTube Amazon Finance Overleaf Journals Entertainment AJO

[5]:

```
gr()
plot(Epts,GEpts,legend=false,xaxis="E",yaxis="G(E)")
```



Polynomial fitting

We use the Julia package CurveFit, and the degree $N = 5$.

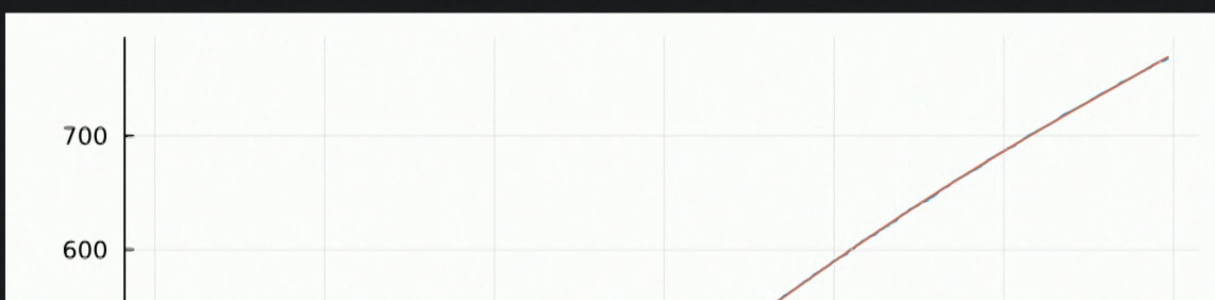
[6]:

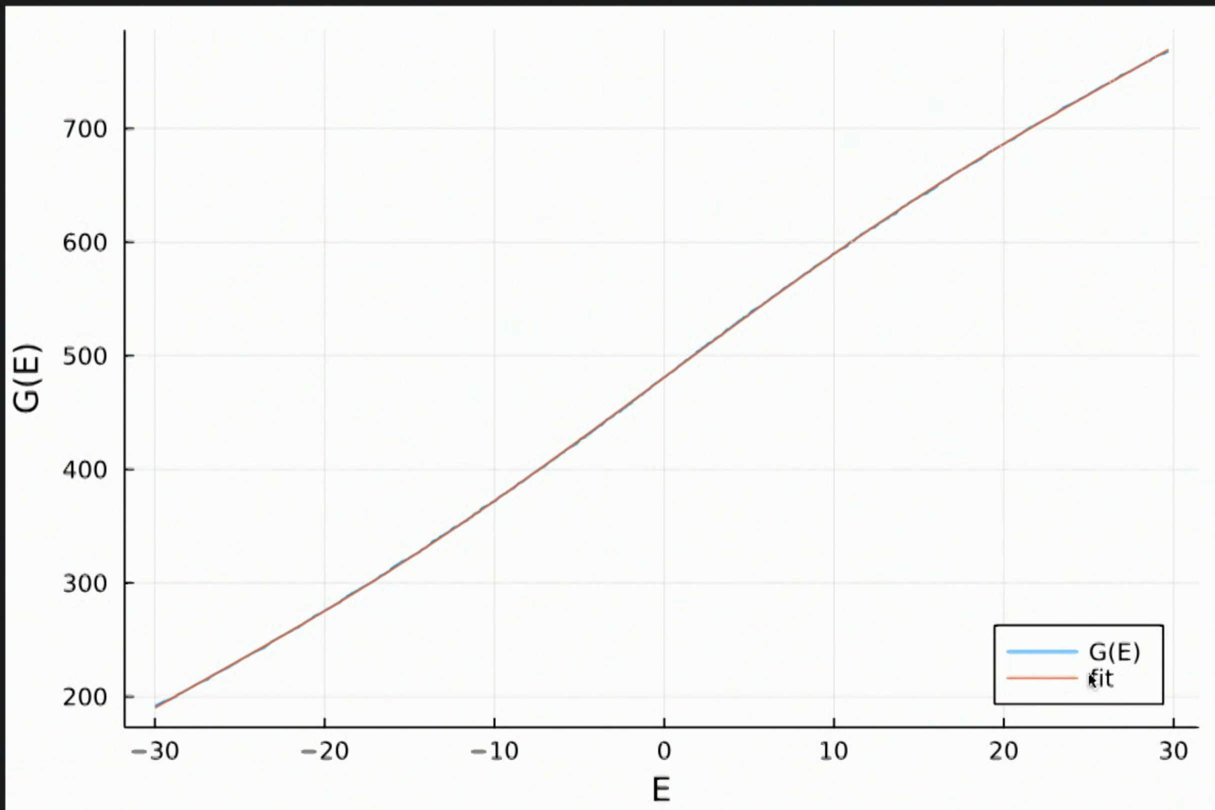
```
fm=poly_fit(Epts, GEpts, 5)
fN(x)=sum(fm .* [x^m for m=0:5])
```

```
fN (generic function with 1 method)
```

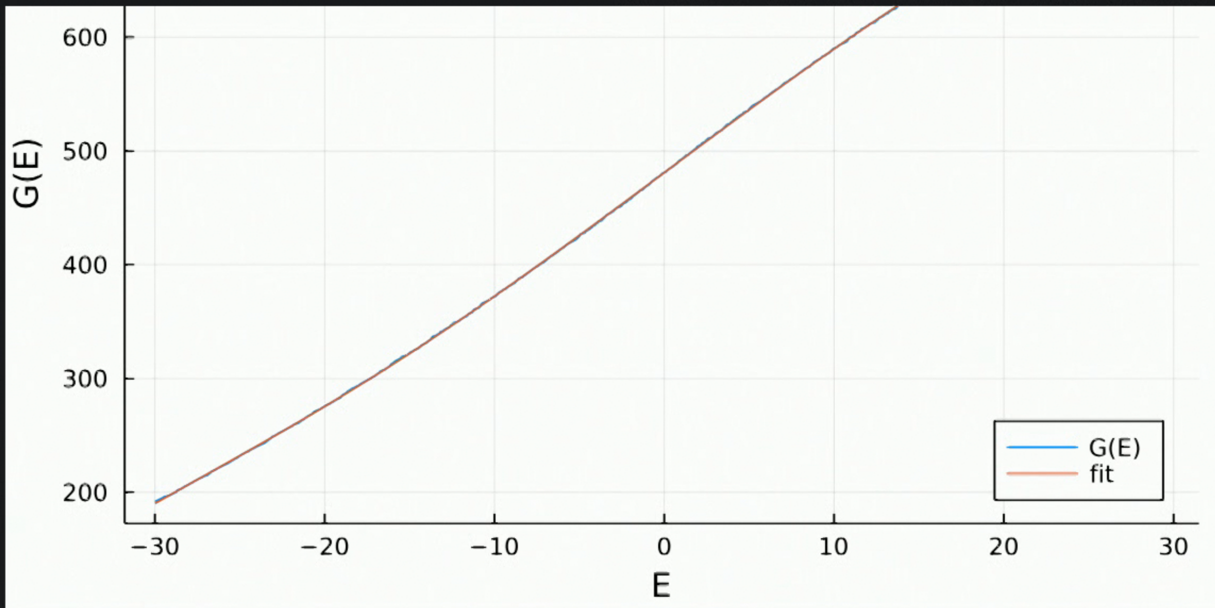
[7]:

```
gr()
plot(Epts,GEpts,xaxis="E",yaxis="G(E)",lab="G(E)")
plot!(Epts,fN.(Epts),lab="fit",legend=:bottomright)
```





Generate $s_n = f_N(E_{n+1}) - f_N(E_n)$

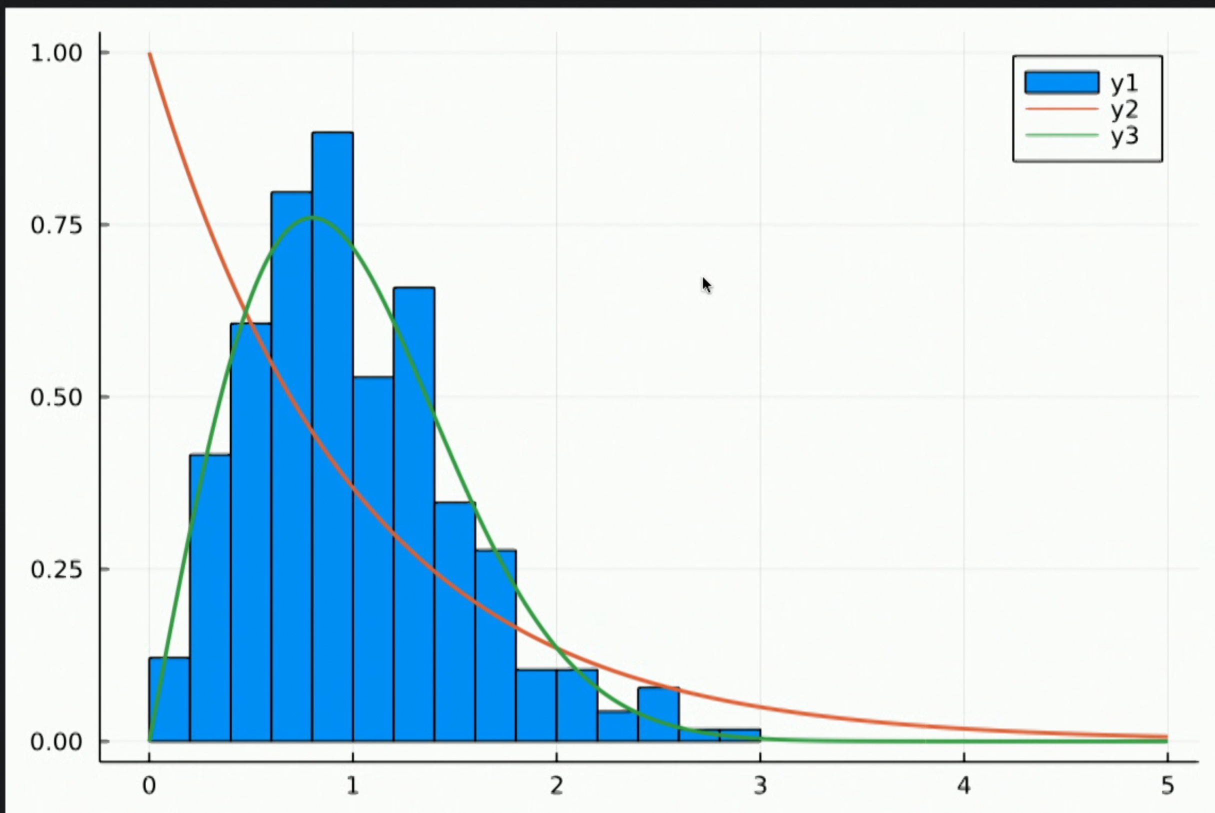


Generate $s_n = f_N(E_{n+1}) - f_N(E_n)$

[8]:

```
function generatesn(spec,fN)
    numberofE = length(spec) #the number of eigenstates at hand
    lowend = div(numberofE,5)
    upend = 4*div(numberofE,5)
```

```
plot!(xpts,Ppoisson.(xpts),line=2)
plot!(xpts,PGOE.(xpts),line=2)
```



Dropbox (PI)/Proj... localhost https://arxiv.org/p...

iCloud Facebook PI YouTube Amazon Finance Overleaf Journals Entertainment AJO

Level-spacing demo

Logout

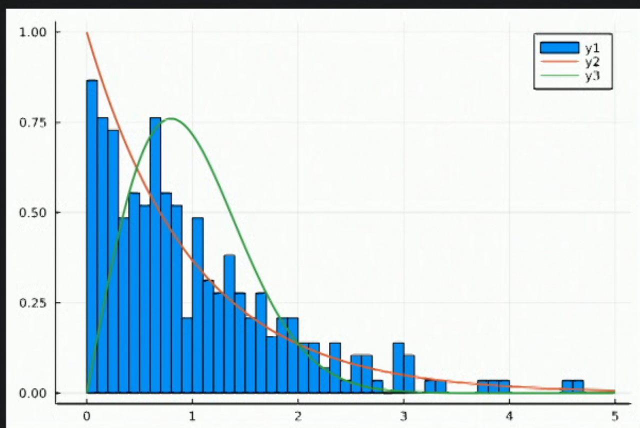
File Edit View Insert Cell Kernel Widgets Help Snippets

Not Trusted | Julia 1.7.2

Run Markdown Notify: Disabled

In [20]:
```
snpts=generatesn(eigvalsμ05,fN);
```

In [23]:
```
xpts=Array(0:0.01:5)
histogram(snpts,bins=50,normalize=true)
plot!(xpts,Ppoisson.(xpts),line=2)
plot!(xpts,PGOE.(xpts),line=2)
```



Gap-ratio statistics

Consider $r_n = \frac{\delta_n}{\delta_{n-1}}$ where $\delta_n = E_{n+1} - E_n$. We then consider the distribution of $\tilde{r}_n = min(r_n, r_n^{-1})$.

$P_{GOE}(\tilde{r}) \approx \frac{27}{4} \frac{\tilde{r}+\tilde{r}^2}{(1+\tilde{r}+\tilde{r}^2)^{\frac{5}{2}}}; \tilde{r}_{avg} \approx 0.5307$

$P_{Poisson}(\tilde{r}) = e^{-r}; \tilde{r}_{avg} \approx 0.39.$

Level-spacing demo

Logout

File Edit View Insert Cell Kernel Widgets Help Snippets

Not Trusted | Julia 1.7.2

Run | Markdown | Notify: Disabled

Gap-ratio statistics

Consider $r_n = \frac{\delta_n}{\delta_{n-1}}$ where $\delta_n = E_{n+1} - E_n$. We then consider the distribution of $\tilde{r}_n = min(r_n, r_n^{-1})$.

$$P_{GOE}(\tilde{r}) \approx \frac{27}{4} \frac{\tilde{r}+\tilde{r}^2}{(1+\tilde{r}+\tilde{r}^2)^{\frac{5}{2}}}; \tilde{r}_{avg} \approx 0.5307$$

$$P_{Poisson}(\tilde{r}) = e^{-r}; \tilde{r}_{avg} \approx 0.39.$$

In [121]:

```
function rstat(E;toptrunc=0.20,bottrunc=0.20)
    botind=floor(Int64,length(E)*bottrunc)
    topind=ceil(Int64,length(E)*(1-toptrunc))
    Etrunc=E[botind:topind]  #truncated spectrum, throwing out top and bot
    δ=Etrunc[2:end]-Etrunc[1:end-1]
    r=δ[2:end]./δ[1:end-1]
    rtilde=min.(r , 1 ./r)
    return sum(rtilde)/length(rtilde),rtilde
end
```

```
rstat (generic function with 1 method)
```

In [123]:

```
rstat(eigvalsµ185)
```

```
(0.5385156166943207, [0.746042957135997, 0.838702257235882, 0.6279924253768115, 0.32810424930202253, 0.5013073042174631, 0.7978633906968184, 0.6317773158722735, 0.1
```

Dropbox (PI)/Proj... localhost https://arxiv.org/p...

iCloud Facebook PI YouTube Amazon Finance Overleaf Journals Entertainment AJO

Level-spacing demo

Logout

File Edit View Insert Cell Kernel Widgets Help Snippets

Not Trusted Julia 1.7.2

Run Code Notify: Disabled

Gap-ratio statistics

Consider $r_n = \frac{\delta_n}{\delta_{n-1}}$ where $\delta_n = E_{n+1} - E_n$. We then consider the distribution of $\tilde{r}_n = min(r_n, r_n^{-1})$.

$P_{GOE}(\tilde{r}) \approx \frac{27}{4} \frac{\tilde{r}+\tilde{r}^2}{(1+\tilde{r}+\tilde{r}^2)^{\frac{5}{2}}}; \tilde{r}_{avg} \approx 0.5307$

$P_{Poisson}(\tilde{r}) = e^{-r}; \tilde{r}_{avg} \approx 0.39.$

In [24]:

```
function rstat(E;toptrunc=0.20,bottrunc=0.20)
    botind=floor(Int64,length(E)*bottrunc)
    topind=ceil(Int64,length(E)*(1-toptrunc))
    Etrunc=E[botind:topind]  #truncated spectrum, throwing out top and bot
    δ=Etrunc[2:end]-Etrunc[1:end-1]
    r=δ[2:end]./δ[1:end-1]
    rtilde=min.(r , 1 ./r)
    return sum(rtilde)/length(rtilde),rtilde
end
```

```
rstat (generic function with 1 method)
```

In [25]:

```
rstat(eigvalsμ185)
```

```
(0.5385156166943207, [0.746042957135997, 0.838702257235882, 0.6279924253768115, 0.32810424930202253, 0.5013073042174631, 0.7978633906968184, 0.6317773158722735, 0.1
3530669281559923, 0.49931346827069484, 0.22611335693820425  …  0.5128923549502897, 0.22611335693852322, 0.49931346827255635, 0.13530669281630692, 0.6317773158729396
, 0.797863390696959, 0.5013073042176446, 0.3281042493019472, 0.6279924253766439, 0.8387022572362655])
```

In [ ]: