Pset 2

Noah Toyonaga

October 21, 2022

1 Problem 1

See attached *Jupyter* notebook printout.

2 Problem 2

2.1

$$\begin{aligned}\mathbb{E}\Psi &= \mathbb{E}\frac{1}{m}\sum_{i=1}^{m}\sigma_i \cdot a_i a_i^T \\ &= \frac{1}{m}\sum_{i=1}^{m}\mathbb{E}\sigma_i \cdot \mathbb{E}a_i a_i^T \\ &= \frac{1}{m}\sum_{i=1}^{m}\mathbb{E}\sigma_i \cdot I \\ &= \mathbb{E}\sigma_i I\end{aligned} \tag{1}$$

We consider the spectral norm of the variation of Ψ about its mean $\mathbb{E}\Psi$:

$$\mathbb{P}\left(\left|v^\top\left(\Psi - \mathbb{E}\Psi\right)v\right| \leq t\right) = 1 - \mathbb{P}\left(\left|v^\top\left(\Psi - \mathbb{E}\Psi\right)v\right| > t\right) \tag{2}$$

We expand the second term on the LHS as

$$\begin{aligned}\mathbb{P}\left(\left|v^\top\left(\Psi - \mathbb{E}\Psi\right)v\right| > t\right) &= \mathbb{P}\left(\left|v^\top\Psi v - v^\top\mathbb{E}\Psi v\right| > t\right) \\ &= \underbrace{\mathbb{P}\left(\left|v^\top\left(\frac{1}{m}\sum_{i=1}^{m}\sigma_i \cdot a_i a_i^T\right)v - v^\top\mathbb{E}\Psi v\right|\right)}_{\star}\end{aligned} \tag{3}$$

The first term can be written by using rotational invariance:

$$v^\top\left(\frac{1}{m}\sum_{i=1}^{m}\sigma_i \cdot a_i a_i^T\right)v = \frac{\sum_i \sigma_i ||v||^2}{m} \tag{4}$$

Meanwhile the second term simplifies since $v^T I v = ||v||^2$.
From construction $||v|| = 1$ so we have:

$$\star = \mathbb{P}\left(\left|\frac{1}{m}\sum_i \sigma_i - \mathbb{E}\sigma\right|\right) \tag{5}$$

The argument of the absolute value is a sum of independent sub-gaussian random variables with zero mean, so we can apply Bernstein's Inequality. To do so we first show that there is a k that satisfies $\mathbb{E}e^{X_i^2/k^2} \leq 2$.

$$e^{\frac{X_i}{k}} \leq 2 \Rightarrow k \geq \frac{X_i}{\log 2} = \frac{\sigma_i - \mathbb{E}\sigma_i}{\log 2} \tag{6}$$

The term on the right is bounded by B by construction of σ so we have for k:

$$k \geq \frac{B}{\log 2} \tag{7}$$

Finally, we write down the Bernstein Inequality:

$$\mathbb{P}\left(\left|v^\top (\Psi - \mathbb{E}\Psi) v\right| \leq t\right) = 1 - 2\exp\left(-c \cdot \min\left\{\frac{t}{k}, \frac{t^2}{k^2}\right\} m\right) \tag{8}$$

2.2

Notation: let $|A|_v \equiv \max_v \left|v^T A v\right|$.
Let $v \in \mathbb{S}^{n-1}$ and $v_0 \in V_\epsilon$ (i.e. v_0 is in the epsilon net).
Let v be the vector that gives the spectral norm of A. Then from the definition of the ϵ-net vector v_0 and spectral norm we have:

$$|A(v - v_0)| \leq \epsilon\, |A| \tag{9}$$

We now derive the desired result using triangle inequality:

$$\begin{aligned} ||A||_v &\leq ||A||_{v_0} + ||A||_{v-v_0} \\ &\leq ||A||_{v_0} + |vA\,(v - v_0)\,| + |(v - v_0)^T Av| \\ &= ||A||_{v_0} + \epsilon ||A||_v + \epsilon ||A||_v \end{aligned} \tag{10}$$

Rearanging, we have:

$$(1 - 2\epsilon)\, ||A||_v \leq ||A||_{v_0} \Rightarrow ||A||_v \leq \frac{1}{(1 - 2\epsilon)}\, ||A||_{v_0} \tag{11}$$

2.3

From **Lemma 2.** we have that $|V| \leq (2/(1/4)+1)^n = 9^n$.

The problem asks for the probability that the operator norm given a maximal $v \in V_\epsilon$ is greater than $t/2$. This is bounded from above by the probability that the quadratic form given *any* v is grater than $t/2$. I.e.

$$\mathbb{P}\left(\max_{v\in V_\epsilon}\left|v^T Av\right| \geq \frac{t}{2}\right) \leq \sum_{v\in V\epsilon} \mathbb{P}\left(\left|v^T Av\right| \geq \frac{t}{2}\right) = 2{*}9^n{*}\exp\left(-c\cdot\min\left\{\frac{t}{k},\frac{t^2}{k^2}\right\}\cdot m\right) \tag{12}$$

3 Problem 3

Notation: In this problem we write $\Lambda \equiv \Lambda(\theta)$. We also suppress the dependence of F and F_n on θ for concision.

3.1

We first write down the expansion of F_n about $\lambda = \Lambda$:

$$\begin{aligned} F_n(\lambda) &= F_n(\lambda)\Big|_\Lambda + \frac{\partial}{\partial\lambda}F_n(\lambda)\Big|_\Lambda(\lambda-\Lambda) \\ &= F_n(\Lambda) + \partial_\lambda F_n(\Lambda)(\lambda-\Lambda) \end{aligned} \tag{13}$$

We know that in the limit $n\to\infty$, $F_n(\lambda_1) \to F(\Lambda)$, so we have:

$$\begin{aligned} F(\Lambda) &\approx F_n(\lambda_1) \\ &\approx F_n(\Lambda) + \partial_\lambda F_n(\Lambda)(\lambda_1-\Lambda) \end{aligned} \tag{14}$$

Rearranging we find:

$$\sqrt{n}(\lambda-\Lambda) = \sqrt{n}\left(\frac{F(\Lambda)-F_n(\Lambda)}{\partial_\lambda F_n(\Lambda)}\right) \tag{15}$$

Multiplying by $\sqrt{n}$ we have the desired expression for $\sqrt{n}(\lambda-\Lambda)$.

3.2

We plug in the given expressions for F_n and F:

$$Y \equiv \sqrt{n}\left(F(\lambda)-F_n(\lambda)\right) = \sqrt{n}\left(-\int\frac{\mu_{SC}}{\lambda-x} + W_{11} + \underbrace{g^\top\left(\lambda I_{n-1} - W_{\setminus 1}\right)^{-1}g}_{\star}\right) \tag{16}$$

We consider the quantity $\star$:

$$\begin{aligned}
g^\top \left(\lambda I_{n-1} - W_{\backslash 1}\right)^{-1} g = g^\top \left(\lambda I_{n-1} - U^\top \Gamma_{\backslash 1} U\right)^{-1} g \quad & \Gamma \text{ is diagonal of eigenvalues of } W_{\backslash 1} \\
= \frac{1}{n} \sum_{i=1}^{n-1} \frac{z_i^2}{\lambda - \gamma_i} \quad & \text{by rotational invariance of } Ug \\
\approx \frac{1}{n} \sum_{i=1}^{n} \frac{z_i^2}{\lambda - \gamma_i} \quad & \lim n \to \infty \\
\approx \frac{\mathrm{Tr}\left(\left(\lambda I_n - W\right)^{-1}\right)}{n} &
\end{aligned} \tag{17}$$

Plugging this into 16 we see that (using the provided fact):

$$Y = \sqrt{n}\left(\mathcal{O}\left(\frac{1}{n}\right) + W_{11}\right) \tag{18}$$

We see that as $n \to \infty$the first term goes to 0 and the second term has mean 0 by construction.

To compute the variance we need to consider the deviation of z_i^2 about its mean. We have:

$$\begin{aligned}
\mathrm{Var}\left(Y\right) &= \mathrm{Var}\sqrt{n}\left(W_{11} + \underbrace{\frac{1}{n}\sum_{i=1}^{n} \frac{z_i^2 - 1}{\lambda - \gamma_i}}_{\text{error}}\right) \\
&= n\left(\mathrm{Var} W_{11} + \frac{1}{n}\mathrm{Var} sum_{i=1}^{n} \frac{z_i^2 - 1}{\lambda - \gamma_i}\right)
\end{aligned} \tag{19}$$

We evaluate the variance in the second term, noting that by constuction of W, z_i and γ_i are independent distributions, so we can take the expectation value of the former conditioned on the latter.

$$\begin{aligned}
\mathrm{Var} \sum_{i=1}^{n} \frac{z_i^2 - 1}{\lambda - \gamma_i} &= \mathbb{E} \left(\sum_{i=1}^{n} \frac{z_i^2 - 1}{\lambda - \gamma_i} \right)^2 \\
&= \left(\sum_{i=1}^{n} \frac{\mathbb{E} \left(z_i^2 - 1 \right)^2}{\left(\lambda - \gamma_i \right)^2} \right) \qquad \text{Only diagonal terms have nonzero expectation} \\
&= \left(\sum_{i=1}^{n} \frac{\mathbb{E} \left(z_i^{-} 2 z_i^2 + 1 \right)}{\left(\lambda - \gamma_i \right)^2} \right) \\
&= \left(\sum_{i=1}^{n} \frac{3 - 2 + 1}{\lambda - \gamma_i} \right) \\
&= 2 \left(\int \frac{\mu_{SC}}{\left(\lambda - x \right)^2} \right)
\end{aligned} \tag{20}$$

Where in the last line we have taken the limit $n \to \infty$ and used the identity between the discrete distribution μ^n and continuous distribution μ_{SC} which we showed for arbitrary test functions in class.

Putting the preceding results together we see that:

$$Y \sim \mathcal{N} \left(0, 2 + 2 \int \frac{\mu_{SC}}{\left(\lambda - x \right)^2} \right) \tag{21}$$

3.3

We note that we can rewrite the desired integral in terms of a derivative of the Stieltjes transform we saw in class so:

$$\begin{aligned}
\int \frac{\mu_{SC} \left(dx \right)}{\left(\lambda - x \right)^2} &= - \frac{\partial}{\partial \lambda} \int \frac{\mu_{SC} \left(dx \right)}{\left(\lambda - x \right)} \\
&= - \frac{\partial}{\partial \lambda} \left(\frac{\lambda - \sqrt{\lambda^2 - 4}}{2} \right) \\
&= - \frac{1}{2} + \frac{\lambda}{\sqrt{\lambda^2 - 4}}
\end{aligned} \tag{22}$$

Finally, we can write down an expression for s^2:

$$s^2 \left(\theta \right) = 2 \left(\frac{1}{2} + \frac{\Lambda}{\sqrt{\Lambda^2 - 4}} \right) \tag{23}$$

3.4

Since the z_i are normally distributed, they will average to 1 and we can simplify the expression :

$$\begin{aligned} g^\top \left(\lambda I_{n-1} - W_{\setminus 1}\right)^{-1} g &\stackrel{d}{=} \frac{1}{n}\sum_{i=1}^{n-1}\frac{z_i^2}{\lambda-\gamma_i} \\ &\approx \frac{1}{n}\sum_{i=1}^{n-1}\frac{1}{\lambda-\gamma_i} \\ &\approx -m^{(n)}\left(\lambda\right) \\ &\approx -m(\lambda) \end{aligned} \tag{24}$$

Where the last line follows when $n \to \infty$. We can use this to evaluate $\partial_\lambda F_n$:

$$\begin{aligned} \partial_\lambda F_n &= \partial_\lambda \left(\lambda - \theta - W_{\setminus 1} + m(\lambda)\right) \\ &= 1 + m'(\lambda) \\ &= 1 + \left(-\frac{1}{2} + \frac{\lambda}{\sqrt{\lambda^2-4}}\right) \\ &= \frac{1}{2} + \frac{\lambda}{\sqrt{\lambda^2-4}} \end{aligned} \tag{25}$$

Evaluated as $n \to \infty$ we have:

$$\partial_\lambda F_n = \frac{1}{2} + \frac{\Lambda(\theta)}{\sqrt{\Lambda(\theta)^2-4}} \equiv \kappa(\theta) \tag{26}$$

3.5

We can plug the preceding results into 15 to find:

$$\sqrt{n}\left(\lambda - \Lambda\right) = \frac{X}{\kappa\left(\theta\right)} \tag{27}$$

Where $X \sim \mathcal{N}\left(0, s^2\right)$. We can rescale X by κ so we have:

$$\sqrt{n}\left(\lambda - \Lambda\right) = \mathcal{N}\left(0, s^2/\kappa^2\right) = \mathcal{N}\left(0, 2\left(\frac{1}{2} + \frac{\lambda}{\sqrt{\lambda^2-4}}\right)^{-1}\right) \tag{28}$$

3.6

We validate 28 using a numerical experiment. See attached *Jupyter* notebook.

In [142…

```python
import numpy as np
from scipy import stats

# for pi
from mpmath import mp
import mpmath
```

In [8]:

```python
import matplotlib.pyplot as plt
```

In [22]:

```python
def plot_complex_eigenvalues(w, ax):
    """Plots `w` on the complex plane."""
    w_r, w_c = w.real, w.imag
    ax.scatter(w_r, w_c)
```

Choose n

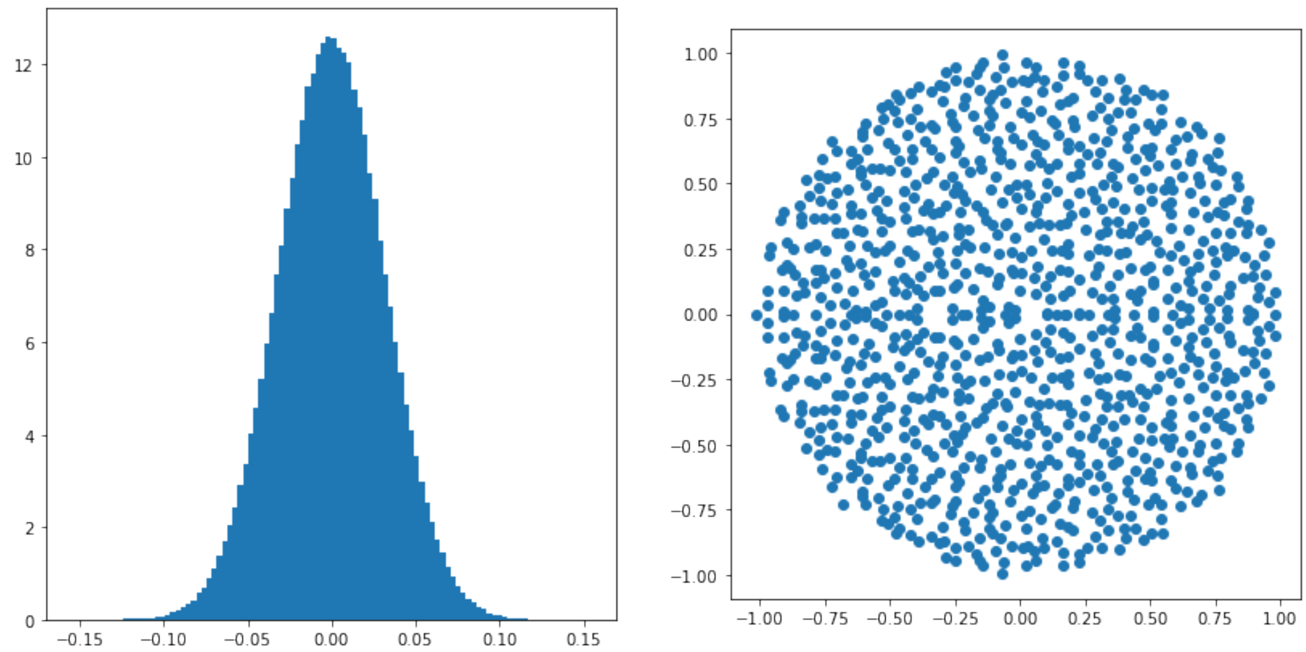
In [114…

```python
n = 1000
```

1.1 i.i.d Gaussian entries.

In [135…

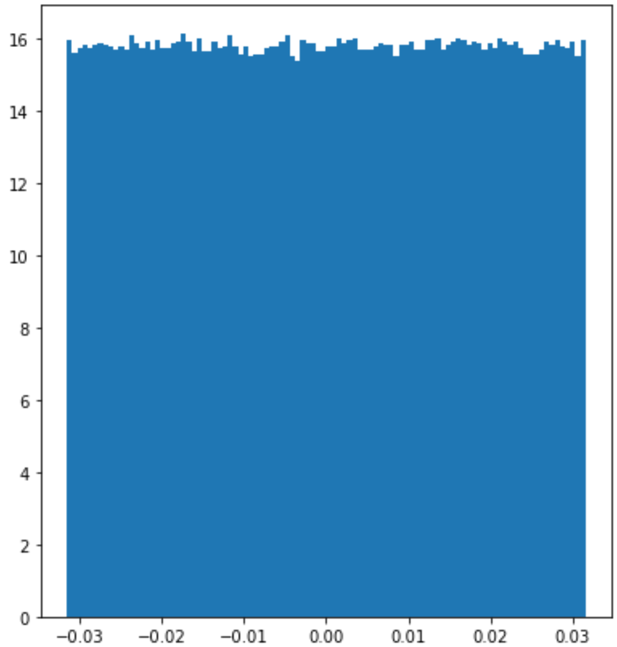
```python
H = np.random.normal(scale=1, size=(n,n)) / np.sqrt(n)

# Extract eigenvalues.
w, v = np.linalg.eig(H)

# Plot.
fig, ax = plt.subplots(1,2,figsize=(14,7))

# Plot distribution.
ax[0].hist(H.flatten(), bins=100, density=True, stacked=True)

# Plot circular law.
plot_complex_eigenvalues(w[1:], ax[1])
ax[1].set_aspect('equal')
```

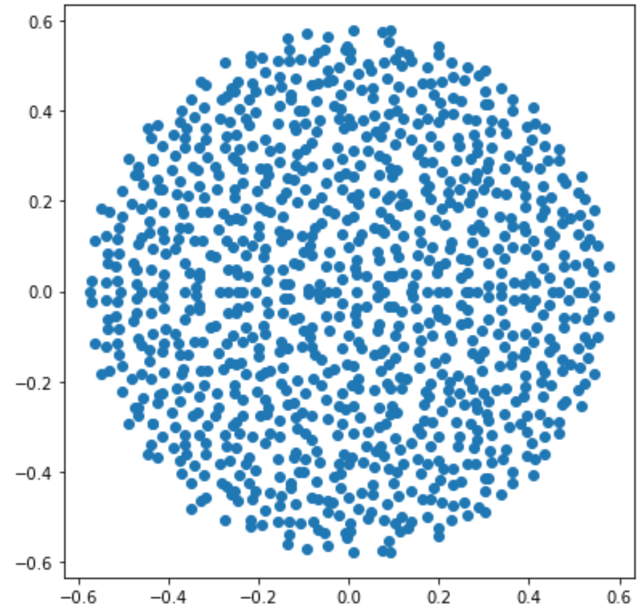


Discrete random w/ equal probability.

In [176…

```
# Random values [0, 1]
H = np.random.rand(n,n)

# Rescale H to appropriate bounds (+-1/sqrt(n))
H = (H - .5) * 2 / np.sqrt(n)

w, v = np.linalg.eig(H)
fig, ax = plt.subplots(1,2,figsize=(14,7))

# Plot distribution.
ax[0].hist(H.flatten(), bins=100, density=True, stacked=True)

# Plot circular law.
plot_complex_eigenvalues(w[1:], ax[1])
ax[1].set_aspect('equal')
```





Create matrix from digits of number (c)

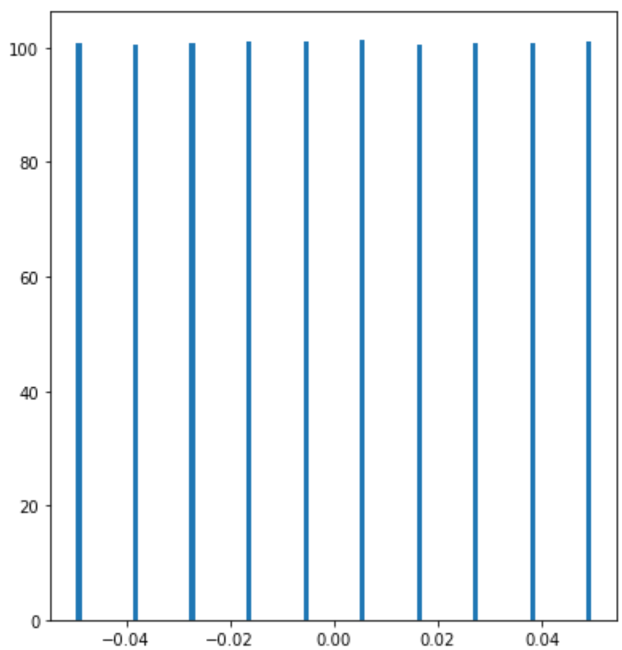
In [173…

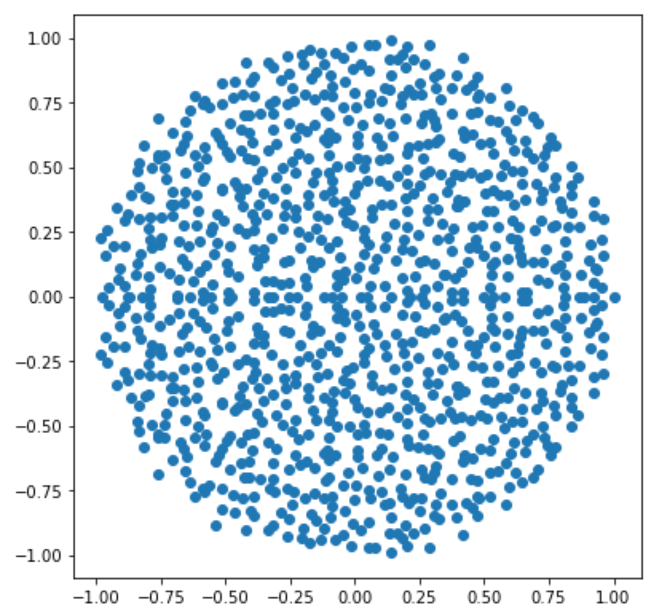
```
def split_str_into_vector(number_as_str):
    return np.array([float(letter) for letter in number_as_str if letter.isdi

def construct_H_from_vector(v):
    b, var = np.mean(v), np.var(v)
    v_scale = 1/ np.sqrt(n * var)
    return (v.reshape(n, n) - b) * v_scale

mp.dps = n ** 2
```

c = π

In [184…

```
## PI
c = mp.pi

# Create string with appropriate number of digits.
c_str = mpmath.nstr(c, n = mp.dps)

# Split string into vector such that each element is a digit from original nu
c_vec = split_str_into_vector(c_str)

# Construct `H` matrix.
H = construct_H_from_vector(c_vec)

# Analyze.
w, v = np.linalg.eig(H)

# Plot.
fig, ax = plt.subplots(1,2,figsize=(14,7))

# Plot distribution.
ax[0].hist(H.flatten(), bins=100, density=True, stacked=True)

# Plot circular law.
plot_complex_eigenvalues(w[1:], ax[1])
ax[1].set_aspect('equal')
```

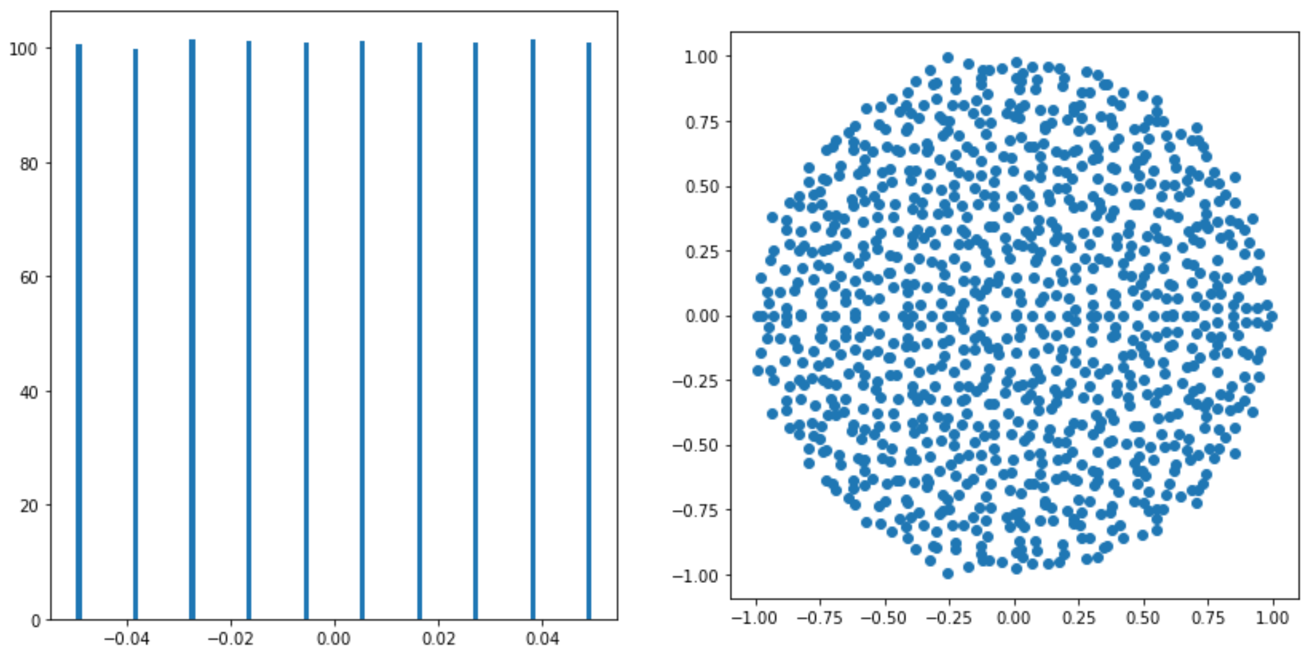




$c = \sqrt{2}$

In [182…

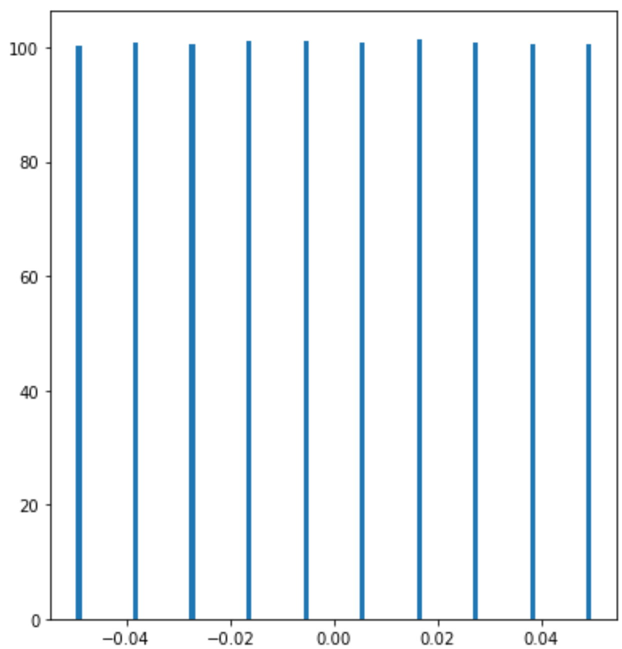
```
c = mp.sqrt(2.)

# Create string with appropriate number of digits.
c_str = mpmath.nstr(c, n = mp.dps)

# Split string into vector such that each element is a digit from original nu
c_vec = split_str_into_vector(c_str)

# Construct `H` matrix.
H = construct_H_from_vector(c_vec)

# Analyze.
w, v = np.linalg.eig(H)

# Plot.
fig, ax = plt.subplots(1,2,figsize=(14,7))

# Plot distribution.
ax[0].hist(H.flatten(), bins=100, density=True, stacked=True)

# Plot circular law.
plot_complex_eigenvalues(w[1:], ax[1])
ax[1].set_aspect('equal')
```

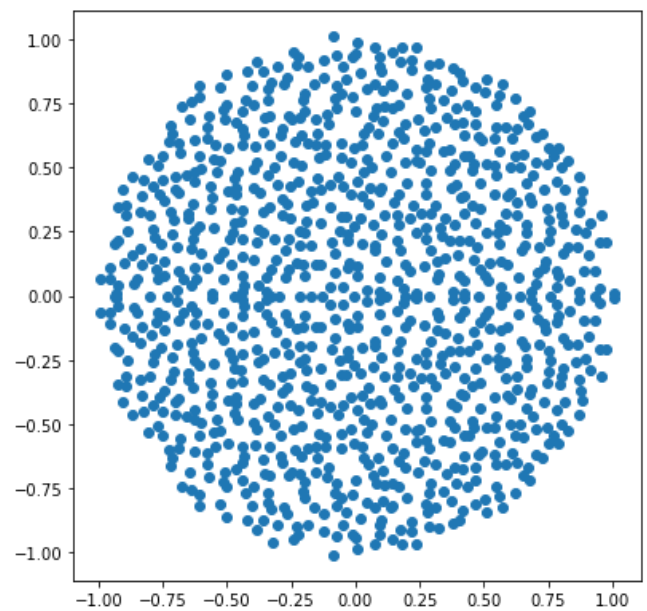


c = e

In [183…

```
c = mp.e

# Create string with appropriate number of digits.
c_str = mpmath.nstr(c, n = mp.dps)

# Split string into vector such that each element is a digit from original nu
c_vec = split_str_into_vector(c_str)

# Construct `H` matrix.
H = construct_H_from_vector(c_vec)

# Analyze.
w, v = np.linalg.eig(H)

# Plot.
fig, ax = plt.subplots(1,2,figsize=(14,7))

# Plot distribution.
ax[0].hist(H.flatten(), bins=100, density=True, stacked=True)

# Plot circular law.
plot_complex_eigenvalues(w[1:], ax[1])
ax[1].set_aspect('equal')
```





Problem 3

In [205…

```
theta = 4
n = 1000

def sigma(theta):
    return None

def construct_W(n):
    W = np.random.normal(scale=1/np.sqrt(n), size=(n,n))
    return 1/np.sqrt(2) * (W + np.transpose(W))
```

In [206…

```
m = 1000
largest_eigenvector=[]

# Construct signal matrix.
signal_matrix = np.zeros((n,n))
signal_matrix[0,0] = theta

for _ in range(m):
    # Construct `W`
    W = construct_W(n)

    # Construct `Y`
    Y = signal_matrix + W

    w = np.linalg.eigvalsh(Y)

    # Get eigenvectors.
#     w, v = np.linalg.eig(H)

    # Sort by size (smallest to largest).
#     w = sorted(w)

    largest_eigenvector.append(w[-1])

largest_eigenvector = np.array(largest_eigenvector)
```

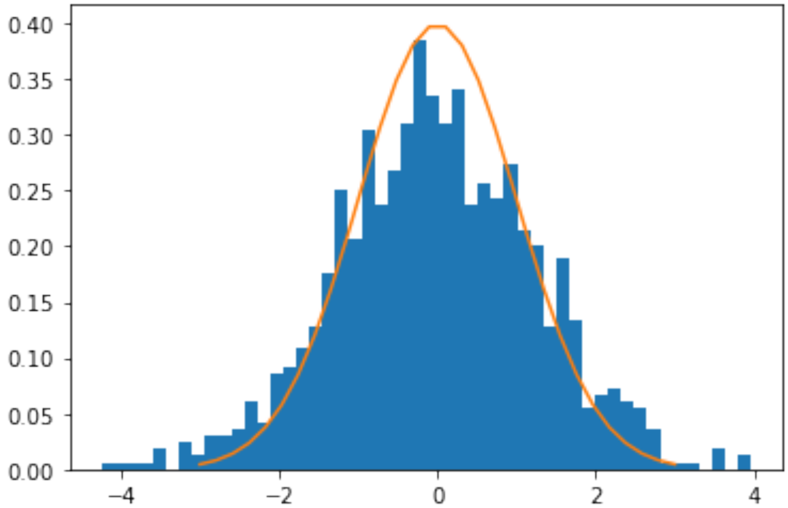
In [207…

```
# Define big Lambda.
Lambda = theta + 1 / theta

# Define variance.
analytic_variance =2 * ((1/2) + Lambda / np.sqrt(Lambda **2 - 4)) ** -1

# Scale eigenvectors
scaled_eigenvectors = np.sqrt(n) * (largest_eigenvector - Lambda) / np.sqrt(a
```

In [208…

```
x_plot = np.linspace(-3,3,30)
plt.hist(scaled_eigenvectors, bins=50, density=True, stacked=True)
plt.plot(x_plot, stats.norm.pdf(x_plot))
```

Out[208… [<matplotlib.lines.Line2D at 0x7f7818909550>]



In [ ]: