Assignment 2

① $L(\theta|y) = \frac{10\,!}{y!\,(10-y)!}\,\theta^{y}(1-\theta)^{10-y}$

$$p(\theta) = \begin{cases} 1 & \theta \in [0,1] \\ 0 & \theta < 0 \text{ or } \theta > 1 \end{cases}$$

$$P(\theta|y) = \frac{L(\theta|y)\,P(\theta)}{\int L(\theta|y)\,p(\theta)\,d\theta}$$

Given : $y = 7$; Marginal likelihood

$$\int L(\theta|y)\,P(\theta)\,d\theta = \frac{1}{11}$$

1.1 (a) $\theta = 0.75$

$$L(\theta|y) = \frac{10!}{7!\,3!}(0.75)^7(0.25)^3$$

$$p(\theta) = 1$$

$$P(\theta|y) = 11 \times \frac{10!}{7!\,3!}(0.75)^7(0.25)^3$$

$$P(\theta|y) = 2.753$$

Made with Goodnotes

(b) $\theta = 0.25$

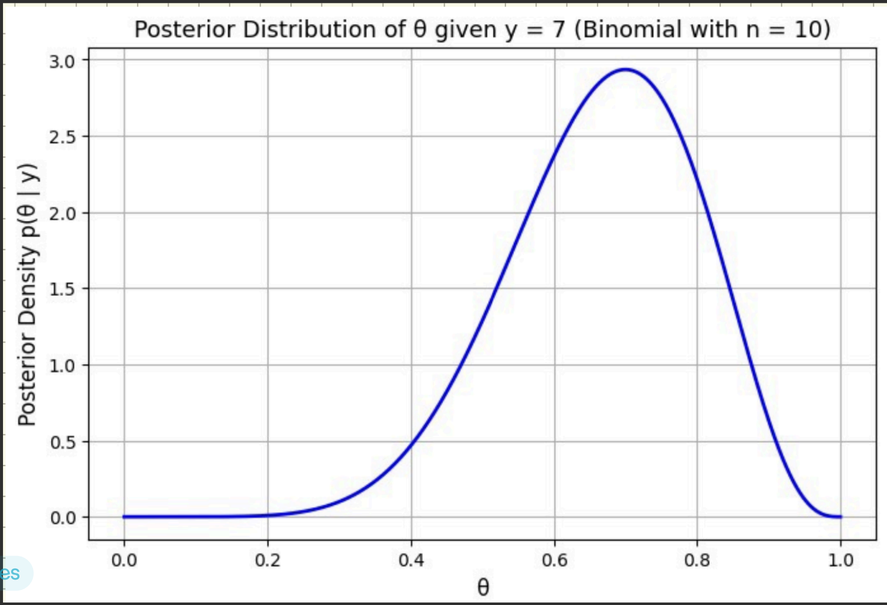
$$P(\theta|y) = 11 \times \frac{10!}{7!\,3!} (0.25)^7 (0.75)^3$$

$$= \underline{\underline{0.033}}$$

(c) $\theta = 1 \Rightarrow P(\theta|y) = 0$

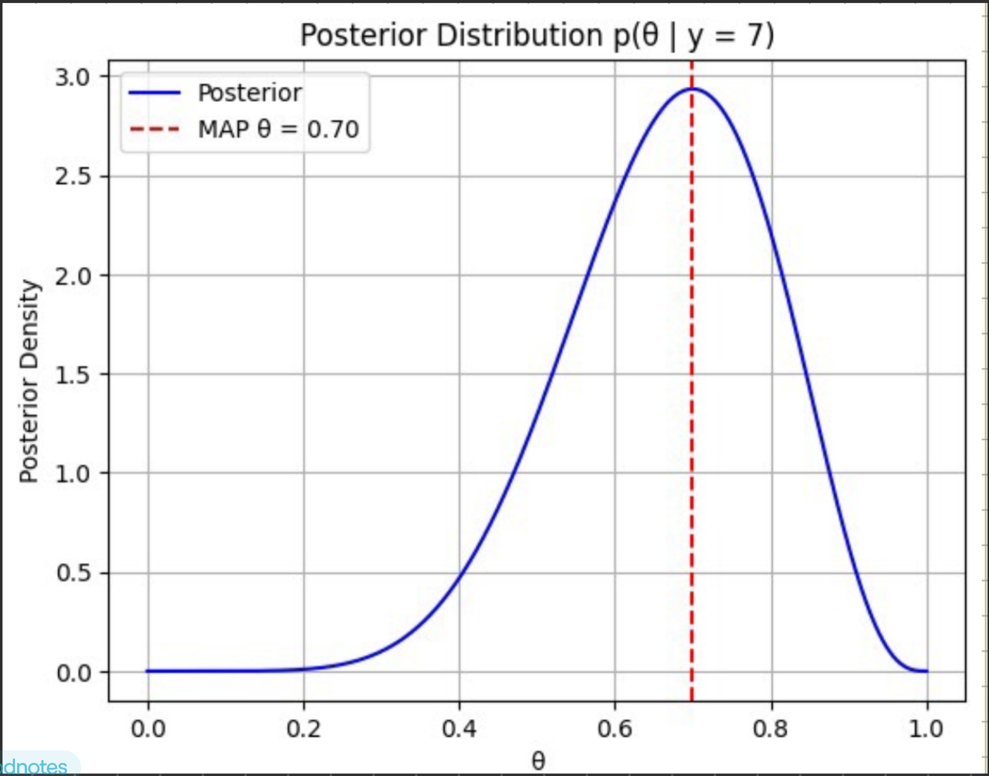
Q2)

```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.special import comb

# Given data
n = 10        # number of trials
y = 7         # observed number of successes

# Create a dense grid of theta values between 0 and 1
theta = np.linspace(0, 1, 1000)

# Likelihood function: Binomial PMF
likelihood = comb(n, y) * (theta ** y) * ((1 - theta) ** (n - y))

# Prior: Uniform between 0 and 1
prior = np.ones_like(theta)  # since p(theta) = 1 when 0 ≤ θ ≤ 1

# Posterior (unnormalized): L(θ|y) * p(θ)
unnormalized_posterior = likelihood * prior

# Normalize the posterior (area under curve = 1)
posterior = unnormalized_posterior / np.trapz(unnormalized_posterior, theta)

# Plotting the posterior
plt.figure(figsize=(8, 5))
plt.plot(theta, posterior, color='blue', lw=2)
plt.title("Posterior Distribution of θ given y = 7 (Binomial with n = 10)", fontsize=13)
plt.xlabel("θ", fontsize=12)
plt.ylabel("Posterior Density p(θ | y)", fontsize=12)
plt.grid(True)
plt.show()
```



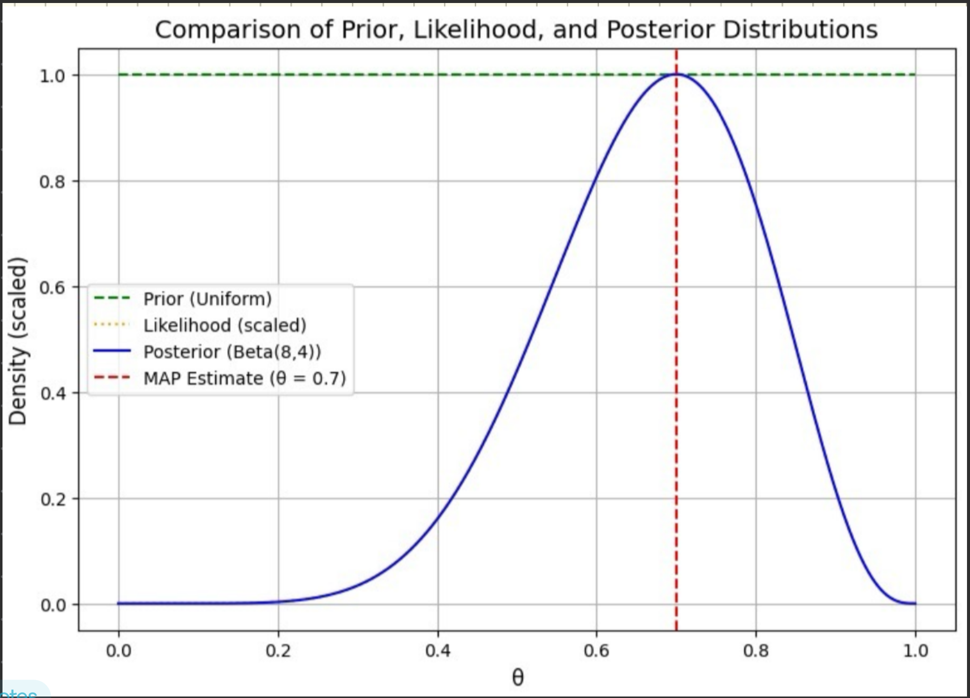
1.3)

```python
import numpy as np
import matplotlib.pyplot as plt
n = 10
y = 7
# Create a dense grid of theta values
theta = np.linspace(0, 1, 1000)

# Compute unnormalized posterior: L(θ|y) ∝ θ^y * (1 - θ)^(n - y)
posterior_unnorm = (theta ** y) * ((1 - theta) ** (n - y))

# Normalize posterior
posterior = posterior_unnorm / np.trapz(posterior_unnorm, theta)

# Find theta with max posterior density
theta_map = theta[np.argmax(posterior)]
max_density = np.max(posterior)

# Plot
plt.plot(theta, posterior, label="Posterior", color='blue')
plt.axvline(theta_map, color='red', linestyle='--', label=f'MAP θ = {theta_map:.2f}')
plt.title("Posterior Distribution p(θ | y = 7)")
plt.xlabel("θ")
plt.ylabel("Posterior Density")
plt.legend()
plt.grid(True)
plt.show()
```



1.4)

```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import beta
# Data
n = 10
y = 7
# Theta values
theta = np.linspace(0, 1, 1000)
# Likelihood (unnormalized)
likelihood = (theta ** y) * ((1 - theta) ** (n - y))
likelihood /= np.max(likelihood)  # normalize for plotting (max = 1)

# Prior: Uniform(0,1)
prior = np.ones_like(theta)  # already uniform
prior /= np.max(prior)       # normalize for plotting
# Posterior: Beta(8, 4)
posterior = beta.pdf(theta, a=y + 1, b=n - y + 1)
posterior /= np.max(posterior)  # normalize for plotting (max = 1)

# Plotting
plt.figure(figsize=(9, 6))
plt.plot(theta, prior, label='Prior (Uniform)', linestyle='--', color='green')
plt.plot(theta, likelihood, label='Likelihood (scaled)', linestyle=':', color='orange')
plt.plot(theta, posterior, label='Posterior (Beta(8,4))', color='blue')
plt.axvline(x=0.7, color='red', linestyle='--', label='MAP Estimate (θ = 0.7)')

plt.title("Comparison of Prior, Likelihood, and Posterior Distributions", fontsize=14)
plt.xlabel("θ", fontsize=12)
plt.ylabel("Density (scaled)", fontsize=12)
plt.legend()
plt.grid(True)
plt.show()
```



2.1)

$$L(\mu, \sigma | y) = \frac{1}{(50\sqrt{2\pi})^8} \exp\left(-\frac{1}{2 \cdot 50^2} \sum_{i=1}^{8} (y_i - \mu)^2\right)$$

$$p(\mu) = \frac{1}{25\sqrt{2\pi}} \exp\left(-\frac{1}{2 \cdot 25^2} (\mu - 250)^2\right)$$

$$S(\mu) = \sum_{i=1}^{8} (y_i - \mu)^2 \quad | \quad \text{For } \underline{\mu = 300}:$$

$$S(300) = (300-300)^2 + (270-300)^2 + (390-300)^2 + (450-300)^2 + (500-300)^2 + (290-300)^2 + (680-300)^2 + (450-300)^2$$

$$S(300) = 238{,}600$$

$$L(300) \propto \exp\left(-\frac{238600}{2 \cdot 2500}\right) = \exp(-47.72)$$

$$p(300) \propto \exp\left(-\frac{(300-250)^2}{2 \cdot 625}\right) = \exp(-2)$$

$$p'(300) \propto \exp(-47.72) \cdot \exp(-2)$$

$$p'(300) \propto \exp(-49.72)$$

For $\mu = 900$,

$$S(900) = \sum_{i=1}^{8} (y_i - 900)^2 = 1992500$$

$$L(900) \propto \exp\left(-\frac{1992500}{2 \cdot 2500}\right) = \exp(-398.5)$$

$$p(900) \propto \exp\left(-\frac{(900-250)^2}{2 \cdot 625}\right) = \exp(-338)$$

$$p'(900) \propto \exp(-736.5)$$

practically zero

For $\mu = 50$,

$$S(50) = \sum_{i=1}^{8} (y_i - 50)^2 = 1203000$$

$$L(50) \propto \exp\left(-\frac{1203000}{2 \cdot 2500}\right) = \exp(-240.6)$$

$$p(50) \propto \exp\left(-\frac{(50-250)^2}{2 \cdot 625}\right) = \exp(-64)$$

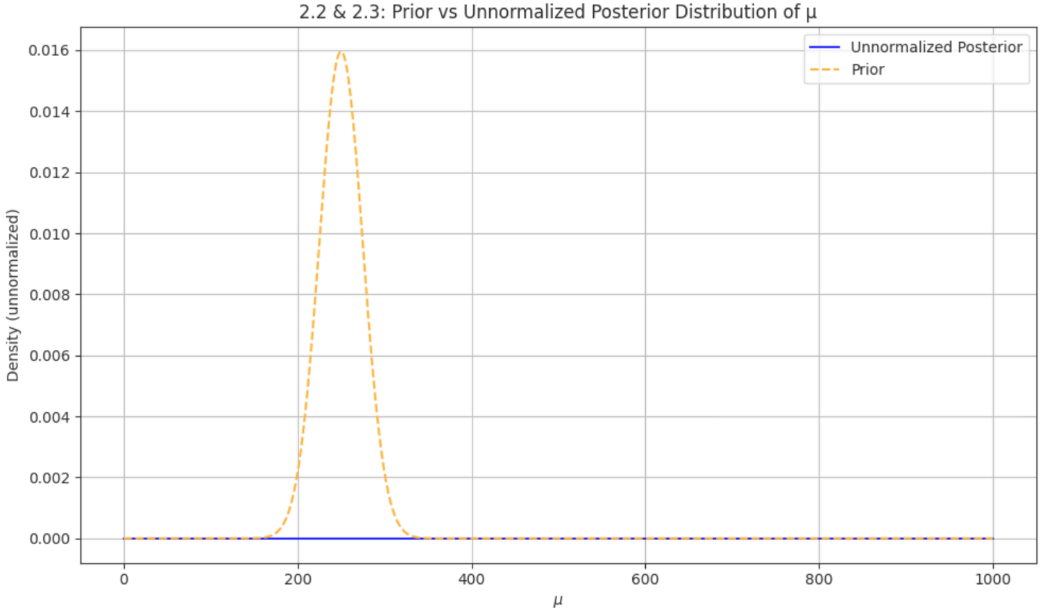
$$p'(50) \propto \exp(-304.6)$$

very close to 0.

2.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import norm

# ------------- Given Data and Parameters -------------
y = np.array([300, 270, 390, 450, 500, 290, 680, 450])
n = len(y)

# Prior for μ ~ Normal(250, 25)
mu_prior_mean = 250
mu_prior_sd = 25

# Fixed σ = 50
sigma_fixed = 50

# Values of μ to evaluate (2.1)
mu_values_to_evaluate = [300, 900, 50]

# ------------- Likelihood & Prior Functions -------------

def exponent(y, mu, sig):
    squared_diffs = np.sum((y - mu) ** 2)
    denom = 2 * (sig ** 2)
    return np.exp(-squared_diffs / denom)

def likelihood(y, mu, sig):
    ex = exponent(y, mu, sig)
    coeff = 1 / ((sig * np.sqrt(2 * np.pi)) ** len(y))
    return coeff * ex

def prior(mu, mean, sd):
    return norm.pdf(mu, loc=mean, scale=sd)

# ------------- 2.1 Unnormalized Poster for Given μ -------------
print("2.1 Unnormalized Posterior Densities for specific μ values:")
for mu_val in mu_values_to_evaluate:
    lkl = likelihood(y, mu_val, sigma_fixed)
    pr = prior(mu_val, mu_prior_mean, mu_prior_sd)
    posterior = lkl * pr
    print(f"μ = {mu_val}: {posterior:.3e}")

# ------------- 2.2 & 2.3: Plot Prior vs Unnormalized Posterior -------------

# Grid of μ values
mu_grid = np.linspace(0, 1000, 1000)

# Prior and Likelihood over grid
prior_values = norm.pdf(mu_grid, loc=mu_prior_mean, scale=mu_prior_sd)
likelihood_values = np.array([likelihood(y, mu, sigma_fixed) for mu in mu_grid])

# Unnormalized Posterior
unnormalized_posterior = likelihood_values * prior_values

# Plotting
plt.figure(figsize=(10, 6))
plt.plot(mu_grid, unnormalized_posterior, label='Unnormalized Posterior', color='blue')
plt.plot(mu_grid, prior_values, label='Prior', linestyle='--', color='orange')
plt.title("2.2 & 2.3: Prior vs Unnormalized Posterior Distribution of μ")
plt.xlabel(r"$\mu$")
plt.ylabel("Density (unnormalized)")
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

```
2.1 Unnormalized Posterior Densities for specific μ values:
μ = 300: 6.824e-41
μ = 900: 0.000e+00
μ = 50: 9.691e-138
```



Part 3) 3.1)

After seeing count k, posterior becomes:

$$\lambda \sim \text{Gamma}(a+k,\ b+1)$$

Day 0 : $\lambda \sim \text{Gamma}(40, 2)$

Day 1 : $k_1 = 25$

$$\lambda \sim \text{Gamma}(40+25=65,\ 2+1=3)$$

Day 2 : $k_2 = 20$

$$\lambda \sim \text{Gamma}(85, 4)$$

Day 3 : $k_3 = 23$

$$\lambda \sim \text{Gamma}(108, 5)$$

Day 4 : $k_4 = 27$

$$\lambda \sim \text{Gamma}(135, 6)$$

3.2) For $\lambda \sim \text{Gamma}(a, b)$

mean, $E[\lambda] = a/b$

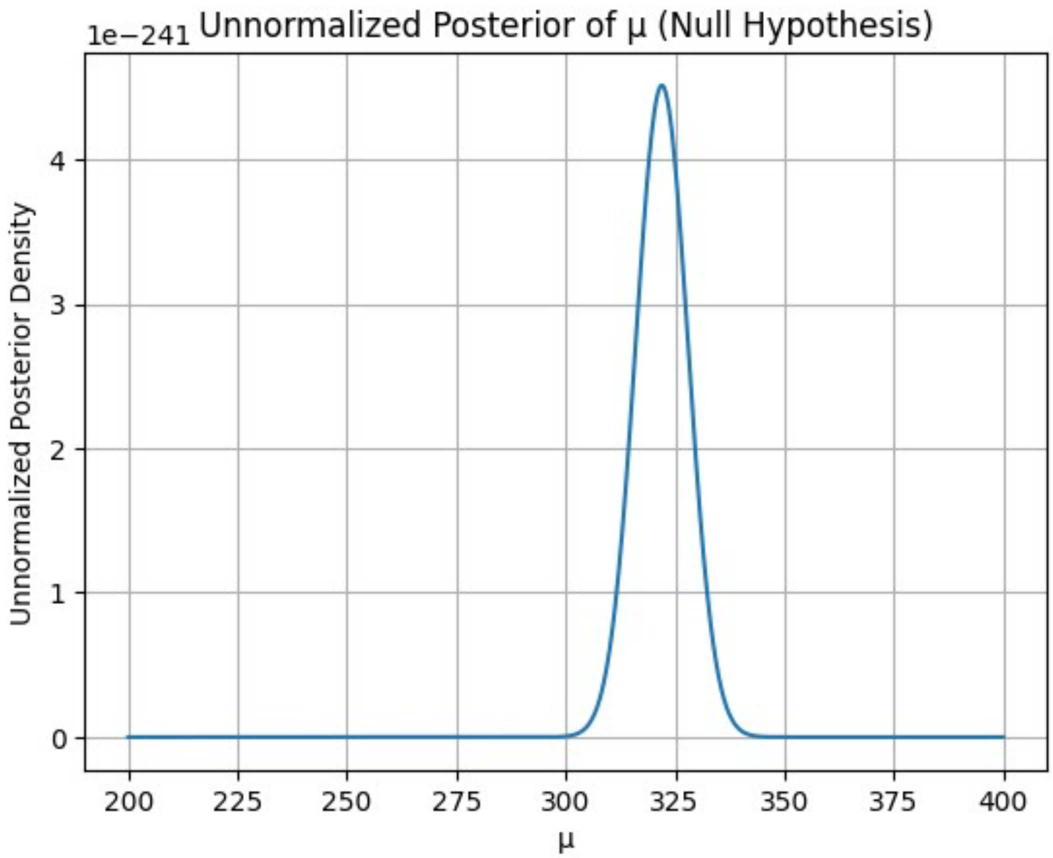
so, $E[\lambda] = \dfrac{135}{6} = 22.5$

$$E[k_5] = E[\lambda] = 22.5$$

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy.stats import norm, truncnorm

url =
"https://raw.githubusercontent.com/yadavhimanshu059/CGS698C/main/notes
/Module-2/recognition.csv"
raw_data = pd.read_csv(url)                     # Read CSV
data = raw_data.drop(columns=raw_data.columns[0])  # Drop first column
(ID)

Tw = data['Tw'].values
Tnw = data['Tnw'].values
σ = 60
```

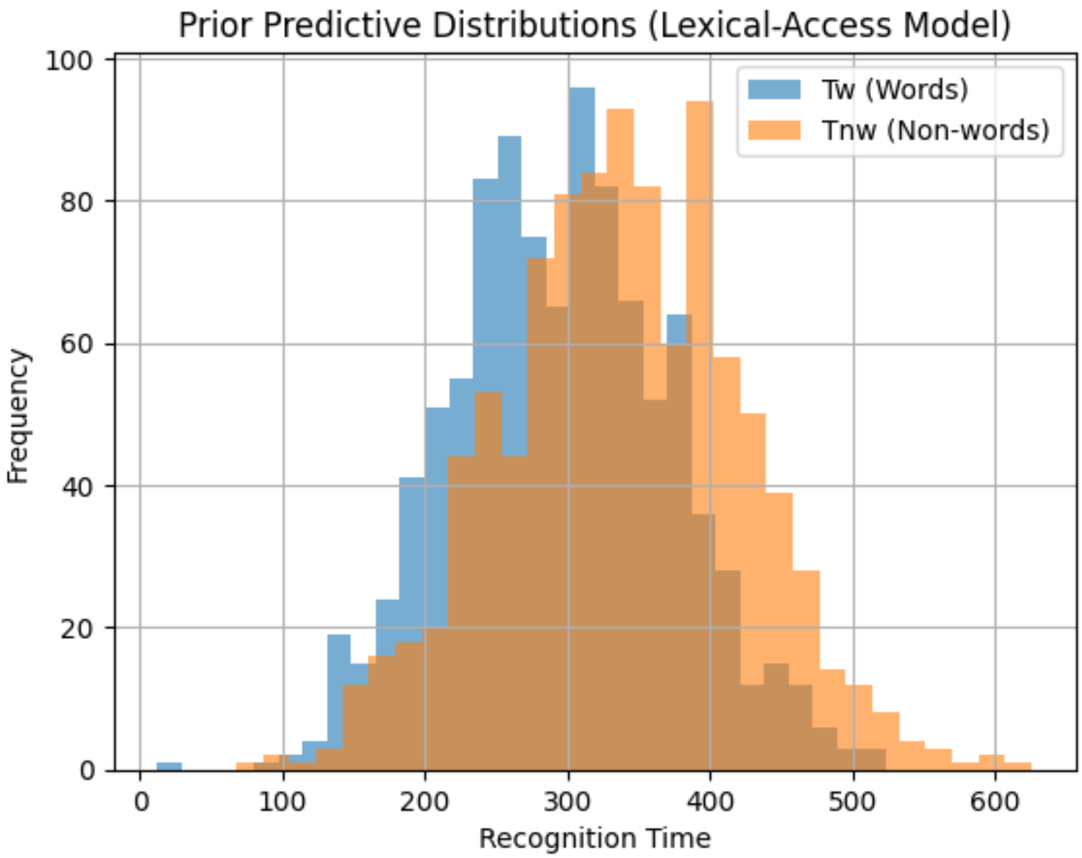
4.5.1: Unnormalized Posterior of μ for Null Hypothesis

```python
mu_values = np.linspace(200, 400, 1000)
posterior_mu = []

for mu in mu_values:
    Lw = np.prod(norm.pdf(Tw, loc=mu, scale=σ))
    Lnw = np.prod(norm.pdf(Tnw, loc=mu, scale=σ))  # since δ = 0 in
null model
    prior_mu = norm.pdf(mu, loc=300, scale=50)
    posterior_mu.append(Lw * Lnw * prior_mu)

posterior_mu = np.array(posterior_mu)

plt.plot(mu_values, posterior_mu)
plt.title("Unnormalized Posterior of μ (Null Hypothesis)")
plt.xlabel("μ")
plt.ylabel("Unnormalized Posterior Density")
plt.grid()
plt.show()
```



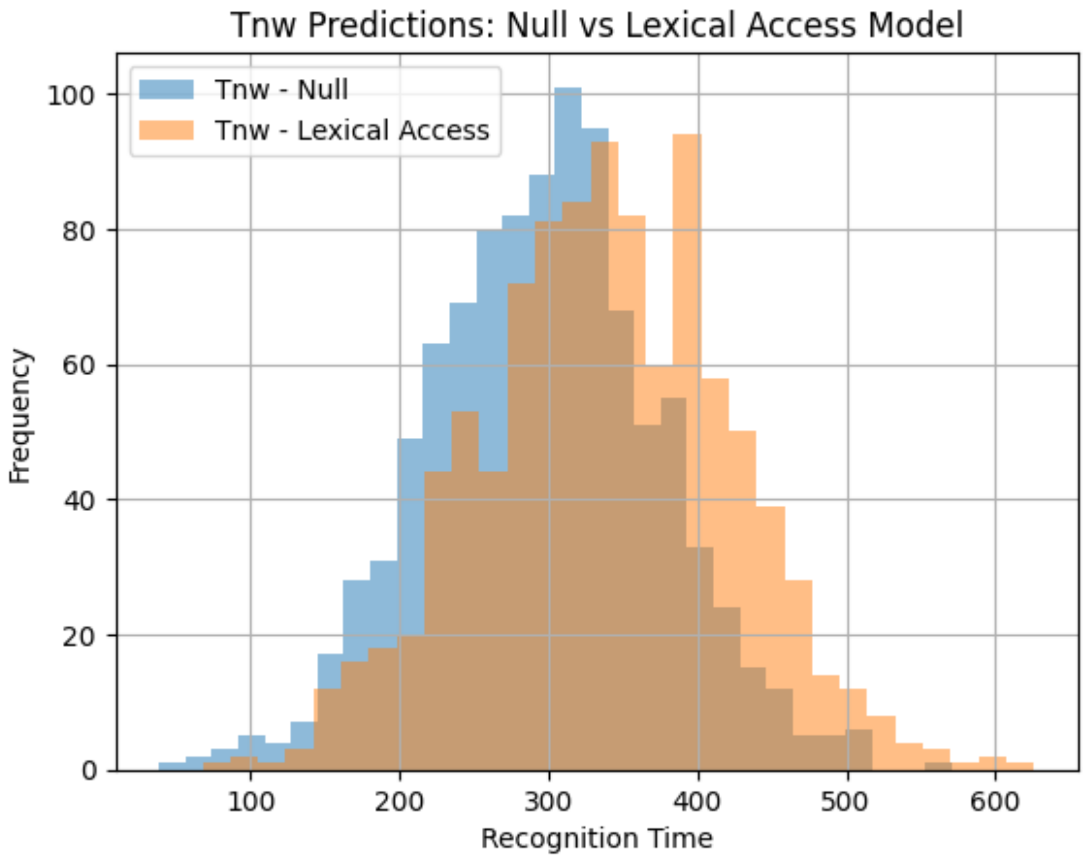
4.5.2: Prior Predictions from Lexical-Access Model

```
# Sample μ and δ from priors
N = 1000
mu_prior = np.random.normal(300, 50, N)
delta_prior = truncnorm.rvs(a=0, b=np.inf, loc=0, scale=50, size=N)

# Generate recognition times
Tw_sim = np.random.normal(mu_prior, σ)
Tnw_sim = np.random.normal(mu_prior + delta_prior, σ)

# Plot
plt.hist(Tw_sim, bins=30, alpha=0.6, label="Tw (Words)")
plt.hist(Tnw_sim, bins=30, alpha=0.6, label="Tnw (Non-words)")
plt.title("Prior Predictive Distributions (Lexical-Access Model)")
plt.xlabel("Recognition Time")
plt.ylabel("Frequency")
plt.legend()
plt.grid()
plt.show()
```



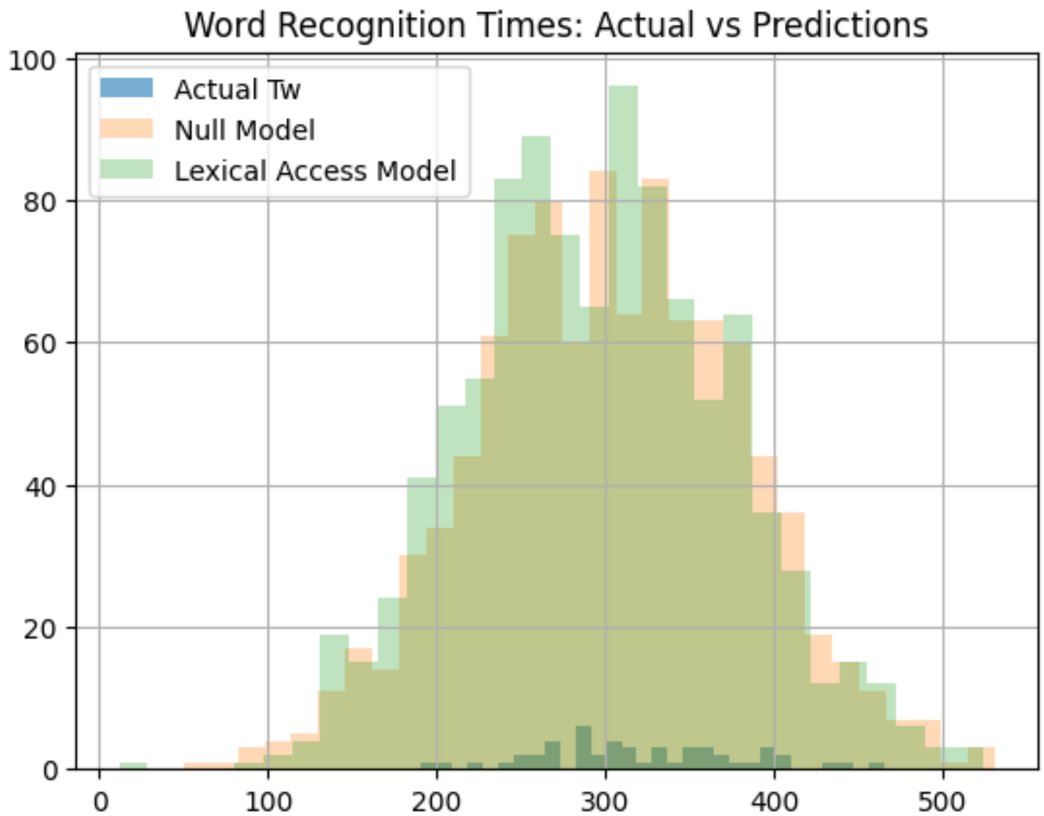
4.5.3: Compare Prior Predictions (Null vs Lexical Access)

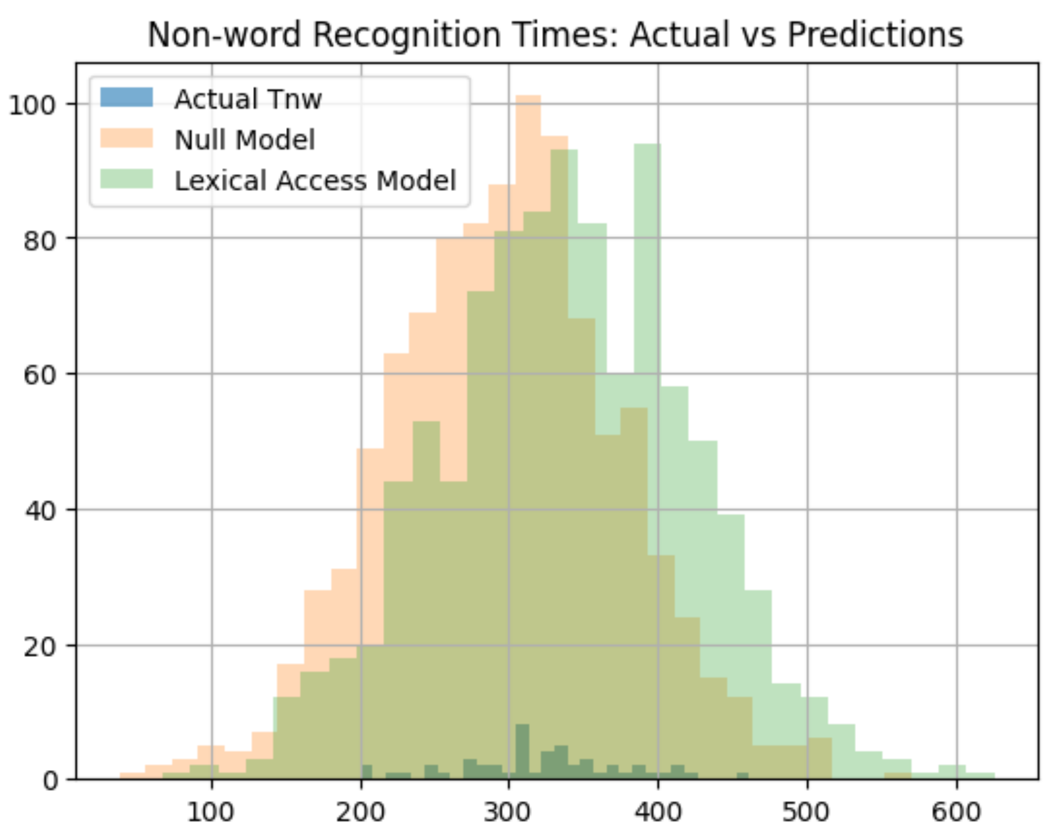
```
# Null model predictions: δ = 0
mu_null = np.random.normal(300, 50, N)
Tw_null = np.random.normal(mu_null, σ)
Tnw_null = np.random.normal(mu_null, σ)

# Histogram comparison
plt.hist(Tnw_null, bins=30, alpha=0.5, label="Tnw - Null")
plt.hist(Tnw_sim, bins=30, alpha=0.5, label="Tnw - Lexical Access")
plt.title("Tnw Predictions: Null vs Lexical Access Model")
plt.xlabel("Recognition Time")
plt.ylabel("Frequency")
plt.legend()
plt.grid()
plt.show()
```



4.5.4: Compare Prior Predictions vs Actual Data

```
# Overlay with real data
plt.hist(Tw, bins=30, alpha=0.6, label="Actual Tw")
plt.hist(Tw_null, bins=30, alpha=0.3, label="Null Model")
plt.hist(Tw_sim, bins=30, alpha=0.3, label="Lexical Access Model")
plt.title("Word Recognition Times: Actual vs Predictions")
plt.legend()
plt.grid()
plt.show()

plt.hist(Tnw, bins=30, alpha=0.6, label="Actual Tnw")
plt.hist(Tnw_null, bins=30, alpha=0.3, label="Null Model")
plt.hist(Tnw_sim, bins=30, alpha=0.3, label="Lexical Access Model")
plt.title("Non-word Recognition Times: Actual vs Predictions")
plt.legend()
plt.grid()
plt.show()
```


Word Recognition Times: Actual vs Predictions
Actual Tw
Null Model
Lexical Access Model
0
20
40
60
80
100
0
100
200
300
400
500


Non-word Recognition Times: Actual vs Predictions
Actual Tnw
Null Model
Lexical Access Model
0
20
40
60
80
100
100
200
300
400
500
600

4.5.5: Posterior Distribution of δ (Lexical-Access Model)

```
mu_fixed = 300  # reasonable fixed value

delta_vals = np.linspace(0, 200, 500)
posterior_delta = []

for delta in delta_vals:
    Lw = np.prod(norm.pdf(Tw, loc=mu_fixed, scale=σ))
    Lnw = np.prod(norm.pdf(Tnw, loc=mu_fixed + delta, scale=σ))
    prior_delta = truncnorm.pdf(delta, a=0, b=np.inf, loc=0, scale=50)
    posterior_delta.append(Lw * Lnw * prior_delta)

posterior_delta = np.array(posterior_delta)
plt.plot(delta_vals, posterior_delta)
plt.title("Unnormalized Posterior of δ (Lexical Access Model)")
plt.xlabel("δ")
plt.ylabel("Unnormalized Posterior Density")
plt.grid()
plt.show()
```