Lab 3.8 excerses

Exercise 3-1

We have 50 integer observations and model

$$Y_i \mid \lambda \sim \text{Poisson}(\lambda), \quad i = 1, \ldots, 50.$$

We will:

- plot the data,
- compute a likelihood on a grid,
- define an Exponential prior,
- compute and normalise the grid posterior,
- compare with the conjugate analytic posterior (Gamma).

1) View the data

```
# View Data ------------------------------------------------------------------

y <- c(2, 6, 2, 3, 4, 3, 4, 3, 1, 2, 3, 2, 6, 6, 2, 3, 5, 1, 2, 2, 4, 2, 5, 3,
       6, 4, 1, 2, 7, 8, 4, 3, 7, 3, 3, 5, 2, 6, 1, 3, 7, 4, 2, 6, 8, 8, 4, 5,
       7, 4)

hist(y, main = "", xlab = "Reaction time (ms)")
```

2) Likelihood and log-likelihood

Poisson pmf:

$$p(y_i \mid \lambda) = \frac{\lambda^{y_i} e^{-\lambda}}{y_i!}, \qquad \lambda \geq 0.$$

Likelihood:

$$L(\lambda; y) = \prod_{i=1}^{n} p(y_i \mid \lambda) \;\propto\; \lambda^{\sum_i y_i} e^{-n\lambda}.$$

Log-likelihood:

$$\log L(\lambda; y) = \sum_{i=1}^{n} \log p(y_i \mid \lambda).$$

3) Likelihood on a grid (R)

```
# Set Up Likelihood Function ------------------------------------------------
lambda <- seq(0, 10, 0.01) #grid of lambda values
likelihood.function <- function(lambda, y) prod(dpois(y, lambda))
#compute likelihood

log.likelihood.function  <- function(lambda, y) sum(dpois(y, lambda, log = TRUE))
#compute loglikelihood

likelihood <- sapply(lambda,  likelihood.function, y)
#evaluate at grid of points

log.likelihood <- sapply(lambda,  log.likelihood.function, y)
#evaluate at grid of points
```

4) Prior on lambda (R)

Prior choice:

$$\lambda \sim \text{Exponential}(0.1), \qquad \pi(\lambda) = 0.1e^{-0.1\lambda}.$$

```
# Set Up Prior Computationally  ------------------------------------------
lambda   <- seq(0, 10, 0.01) #grid of lambda values
prior    <- dexp(lambda, 0.1)
log.prior <- dexp(lambda, 0.1, log = TRUE)
plot(lambda, prior, type = 'l', xlab = expression(lambda), ylab = "density")
```

5) Grid posterior + trapezium normalisation (R)

Bayes rule (up to proportionality):

$$\pi(\lambda \mid y) \propto \pi(\lambda)\, L(\lambda; y).$$

```
# Construct Posterior Distribution Computationally ------------------------
posterior <- prior*likelihood
integrating.factor <- 0.5*0.01*(posterior[1] + posterior[1001]
+ 2*sum(posterior[-c(1, 1001)])) #Using trapezium rule

posterior <- posterior/integrating.factor #normalise

plot(lambda, posterior, type = 'l', xlab = expression(lambda),
     ylab = "posterior density")
```

6) Analytic posterior (conjugacy)

Exponential(0.1) is Gamma(shape=1, rate=0.1). Conjugacy gives:

$$\lambda \mid y \sim \mathrm{Gamma}\Big(1 + \sum_{i=1}^{n} y_i,\ 0.1 + n\Big).$$

With $n = 50$:

$$\lambda \mid y \sim \mathrm{Gamma}\big(\sum y + 1,\ 50.1\big).$$

7) Overlay grid vs analytic posterior (R)

```
# Construct Posterior Distribution Analytically ---------------------------
# The analytical distribution is Gamma(sum(y)+1, 50.1) (shape-rate)

posteiror.analytical <- dgamma(lambda, sum(y) + 1, 50.01)
plot(lambda, posterior, type = 'l', xlab = expression(lambda),
     ylab = "posterior density")
lines(lambda, posteiror.analytical, col = 2)
max(abs(posterior - posteiror.analytical)) #maximum absolute error
```

Note: the rate should be 50.1 (not 50.01) if prior rate is 0.1 and $n = 50$.

Exercise 3.2

We observe positive data $y_1, \ldots, y_N$ and assume an Exponential model:

$$Y_i \mid \lambda \sim \text{Exponential}(\lambda), \qquad f(y_i \mid \lambda) = \lambda e^{-\lambda y_i}, \quad y_i \geq 0, \quad \lambda > 0.$$

We place a Beta prior on λ and compute the posterior $\pi(\lambda \mid y)$ numerically on a grid.

Important constraint: A Beta prior is supported on $[0, 1]$, so we are implicitly restricting $\lambda \in [0, 1]$.

1) View data

```
# View Data ------------------------------------------------------------------

y <- c(1.95, 1.46, 4.81, 1.52, 4.24, 3.00, 0.46, 2.27, 1.76, 0.41)
hist(y, main = "", xlab = "y")
N <- length(y)
```

Here $N =$ the number of observations (sample size).

2) Likelihood for the Exponential rate λ

Because

$$f(y_i \mid \lambda) = \lambda e^{-\lambda y_i},$$

the likelihood is

$$L(\lambda; y) = \prod_{i=1}^{N} \lambda e^{-\lambda y_i} = \lambda^N \exp\Big(- \lambda \sum_{i=1}^{N} y_i \Big).$$

So, up to proportionality,

$$L(\lambda; y) \;\propto\; \lambda^N \exp\Big(- \lambda \sum_{i=1}^{N} y_i \Big).$$

Log-likelihood:

$$\log L(\lambda; y) = N \log \lambda - \lambda \sum_{i=1}^{N} y_i.$$

3) Compute likelihood on a grid (R)

```
# Set Up Likelihood Function ----------------------------------------------
lambda <- seq(0, 1, 0.01) #grid of lambda values

likelihood.function <- function(lambda, y) prod(dexp(y, lambda))
#compute likelihood

log.likelihood.function  <- function(lambda, y) sum(dexp(y, lambda, log = TRUE))
#compute loglikelihood

likelihood <- sapply(lambda,  likelihood.function, y)
#evaluate at grid of points

log.likelihood <- sapply(lambda,  log.likelihood.function, y)
#evaluate at grid of points
```

Note: `dexp(y, rate=lambda)` expects $\lambda \geq 0$. Including $\lambda = 0$ can produce `-Inf` in logs.

4) Prior: Beta(α, β) on $\lambda \in [0, 1]$

We use a Beta prior:

$$\lambda \sim \mathrm{Beta}(\alpha, \beta), \qquad \pi(\lambda) \propto \lambda^{\alpha-1}(1-\lambda)^{\beta-1}, \quad 0 < \lambda < 1.$$

In your code: $\alpha = 1, \beta = 1$, so $\pi(\lambda)$ is Uniform$(0, 1)$.

5) Prior computation (R)

```
# Set Up Prior Computationally  ----------------------------------------
prior    <- dbeta(lambda, 1, 1)
log.prior <- dbeta(lambda, 1, 1, log = TRUE)

plot(lambda, prior, type = 'l',
     xlab = expression(lambda), ylab = "density")
```

6) Posterior up to proportionality

Posterior is proportional to prior $\times$ likelihood:

$$\pi(\lambda \mid y) \;\propto\; \pi(\lambda)\, L(\lambda; y).$$

Combining the expressions,

$$\pi(\lambda \mid y) \;\propto\; \lambda^{\alpha-1}(1-\lambda)^{\beta-1} \cdot \lambda^{N} \exp\Big(-\lambda \sum_{i=1}^{N} y_i \Big).$$

So

$$\pi(\lambda \mid y) \;\propto\; \lambda^{N+\alpha-1}(1-\lambda)^{\beta-1} \exp\Big(-\lambda \sum_{i=1}^{N} y_i \Big),$$

which matches your comment.

Key point: This is *not* a standard named posterior family (because of the $\exp(-\lambda \sum y_i)$ term with Beta support).

7) Grid posterior + trapezium normalisation (R)

```
# Construct Posterior Distribution Computationally ------------------------
posterior <- prior*likelihood

integrating.factor <- 0.5*0.01*(posterior[1] + posterior[101] +
                                2*sum(posterior[-c(1, 101)])) # trapezium rule

posterior <- posterior/integrating.factor # normalise

plot(lambda, posterior, type = 'l',
     xlab = expression(lambda), ylab = "posterior density")
```

This numerically approximates $\int_0^1 \pi(\lambda)L(\lambda; y)\, d\lambda$ to normalise the posterior.

8) Recommended stability tweak (avoid $\lambda = 0$)

Because $\log \lambda$ blows up at $\lambda = 0$, it is often safer to start the grid at a small positive value.

```
# Safer grid (avoid lambda = 0 exactly)
lambda <- seq(1e-4, 1, 0.01)

log.likelihood <- sapply(lambda, log.likelihood.function, y)
log.prior <- dbeta(lambda, 1, 1, log = TRUE)

log.post <- log.prior + log.likelihood
log.post <- log.post - max(log.post)    # stabilise
post.unnorm <- exp(log.post)

h <- 0.01
Z <- 0.5*h*(post.unnorm[1] + post.unnorm[length(post.unnorm)] +
           2*sum(post.unnorm[-c(1, length(post.unnorm))]))

posterior <- post.unnorm / Z
```

Take-home

- Exponential rate likelihood gives $L(\lambda; y) \propto \lambda^N e^{-\lambda \sum y}$.
- Beta(α, β) prior restricts $\lambda \in [0, 1]$.
- Posterior is proportional to

$$\lambda^{N+\alpha-1}(1-\lambda)^{\beta-1} \exp\Big(-\lambda \sum y\Big),$$

  which we normalise numerically using the trapezium rule.
- For numerical stability, avoid including $\lambda = 0$ exactly when using log computations.

Excersie 3-3

We assume a Binomial likelihood with a Beta prior:

$$X \mid p \sim \mathrm{Binomial}(n, p), \qquad p \sim \mathrm{Beta}(\alpha, \beta).$$

Conjugacy gives the posterior

$$p \mid x \sim \mathrm{Beta}\big(\alpha + x,\ \beta + (n - x)\big).$$

In your code the "effective number of trials" is $n = 100N$ and the total number of successes is $\sum x$, so

$$p \mid x \sim \mathrm{Beta}\Big(\sum x + \alpha,\ 100N - \sum x + \beta\Big).$$

Posterior-evaluation function (R)

```
# Function for posterior distribution ------------------------------------
#This function computes the posterior distribution
#It has four inputs: sum.x and N describing the data, and alpha and beta
#It outputs a vector evaluating the posterior distribution at [0, 0.01, ..., 1]
evaluate.posterior <- function(sum.x, N, alpha, beta){

  #Create grid
  p <- seq(0, 1, 0.01)

  #Evaluate posterior
  posterior <- dbeta(p, sum.x + alpha, 100*N - sum.x + beta)

  #Return posterior
  return(posterior)
}
```

Note: the grid is $[0, 1]$ (so the comment "...,10" should be "...,1").

What are we varying and why?

Here we explore **prior sensitivity** by fixing $\alpha = 2$ and varying β over a grid:

$$\beta \in [0.01, 10].$$

This changes the prior mean

$$\mathbb{E}[p]_{\text{prior}} = \frac{\alpha}{\alpha + \beta},$$

and therefore changes how strongly the prior pulls the posterior when data are limited.

We will compare:

- a **large data** case: $N = 150$, $\sum x = 2971$,
- a **low data** case: $N = 5$, $\sum x = 101$.

Large data scenario: compute posteriors for many β

```
# Large data scenario ------------------------------------------------------
p <- seq(0, 1, 0.01)          #Create grid of p values
beta <- seq(0.01, 10, 0.01)   #Create grid of beta values

all.posteriors <- sapply(beta, evaluate.posterior,
                         sum.x = 2971, N = 150, alpha = 2)
```

Interpretation: `all.posteriors` is a matrix:

- rows correspond to $p \in \{0, 0.01, \ldots, 1\}$,
- columns correspond to different β values.

Large data: overlay posterior vs prior for one β

```
#Plot Posterior and Prior
plot(p, all.posteriors[, 1000], type = 'l', xlab = "p", ylab = "density")
lines(p, dbeta(p, 2, beta[1000]), col = 2)
```

What you should see:

- The posterior (black) is much tighter than the prior (red).
- With lots of data, the posterior is dominated by the likelihood, so it barely changes as β changes.

Large data: prior mean vs posterior mean

For Beta(α, β):

$$\mathbb{E}[p]_{\text{prior}} = \frac{\alpha}{\alpha + \beta}.$$

For the posterior Beta$(\sum x + \alpha,\ 100N - \sum x + \beta)$:

$$\mathbb{E}[p]_{\text{post}} = \frac{\sum x + \alpha}{(\sum x + \alpha) + (100N - \sum x + \beta)} = \frac{\sum x + \alpha}{100N + \alpha + \beta}.$$

With $\alpha = 2$, $N = 150$, $\sum x = 2971$:

$$\mathbb{E}[p]_{\text{post}} = \frac{2971 + 2}{100 \cdot 150 + 2 + \beta}.$$

Large data: mean curves (R)

```
#The prior mean is alpha/(alpha + beta).
#The posterior mean is (sum(x)+alpha)/(alpha + beta + 100N)


prior.mean <- 2/(2 + beta)
posterior.mean <- (2971 + 2)/(2 + 100*150 + beta)

#Create Plots
plot(beta, posterior.mean, type = 'l', ylim = c(0, 1),
     xlab = expression(beta), ylab = "posterior mean")
plot(prior.mean, posterior.mean, type = 'l', ylim = c(0, 1),
     xlab = "prior mean", ylab = "posterior mean")
```

Idea: with large data, $\mathbb{E}[p]_{\text{post}}$ changes very little as β changes.

Low data scenario: compute posteriors for many β

```
# Low data scenario ----------------------------------------------------------
p <- seq(0, 1, 0.01)          #Create grid of p values
beta <- seq(0.01, 10, 0.01)   #Create grid of beta values

all.posteriors <- sapply(beta, evaluate.posterior,
                         sum.x = 101, N = 5, alpha = 2)
```

Now $100N = 500$ trials in total, so the data are much less informative than the $100N = 15000$ case.

Low data: overlay posterior vs prior for one β

```
#Plot Posterior and Prior
plot(p, all.posteriors[, 500], type = 'l', xlab = "p", ylab = "density")
lines(p, dbeta(p, 2, beta[500]), col = 2)
```

What you should see:

- The posterior is still influenced by the prior (less peaked than in large data).
- Changing β changes the prior mean and can shift/spread the posterior.

Low data: posterior mean

With $\alpha = 2$, $N = 5$, $\sum x = 101$:

$$\mathbb{E}[p]_{\text{post}} = \frac{101 + 2}{100 \cdot 5 + 2 + \beta} = \frac{103}{502 + \beta}.$$

We again compare prior mean vs posterior mean:

$$\mathbb{E}[p]_{\text{prior}} = \frac{2}{2 + \beta}, \qquad \mathbb{E}[p]_{\text{post}} = \frac{103}{502 + \beta}.$$

Low data: mean curves (R)

```
#The prior mean is alpha/(alpha + beta).
#The posterior mean is (sum(x)+alpha)/(alpha + beta + 100N)


prior.mean <- 2/(2 + beta)
posterior.mean <- (101 + 2)/(2 + 100*5 + beta)

#Create Plots
plot(beta, posterior.mean, type = 'l', ylim = c(0, 1),
     xlab = expression(beta), ylab = "posterior mean")
plot(prior.mean, posterior.mean, type = 'l', ylim = c(0, 1),
     xlab = "prior mean", ylab = "posterior mean")
```

Conclusion: prior sensitivity vs data size

- **Large data:** the posterior is dominated by the likelihood. Varying β has a small effect.
- **Low data:** the posterior is more sensitive to the prior. Varying β shifts the posterior and its mean more noticeably.
- Your final comment matches the general message: some posterior summaries can look "invariant" to β when data are informative, but changing α (shape near 0) can have a stronger effect.