SDS_459_HW1

2025-02-17

Question 3 Part C

```r
# Set seed for reproducibility
set.seed(42)
# Number of samples
n <- 40
lambda <- 6
pois <- rpois(n, lambda)
head(pois)
```

```
## [1]  9 10  5  8  7  6
```

```r
poisson_nll <- function(lambda, pois){
  return(-sum(dpois(pois, lambda, log=TRUE)))
}

mle_result <- optim(par = 5, fn = poisson_nll, pois = pois, method = 'L-BFGS-B', lower = 0.01 )
lambda_mle <- mle_result$par
lambda_mle
```

```
## [1] 6.65
```

Given the observed data, the most likely Poisson rate parameter is 6.65. While we know the true rate parameter is 6, the MLE estimate differs slightly due to having only 40 observations. This variability is expected in finite samples. However, as we increase the number of observations, the MLE estimate will converge to the true value, demonstrating the consistency property of MLE.

Question 3 Part D

```r
posterior_nll <- function(lambda, pois, alpha, beta){
  S <- sum(pois)
  log_prior <- (alpha - 1) * log(lambda) - beta*lambda
  log_likelihood <- sum(dpois(pois, lambda, log=TRUE))
  negative_log_posterior <- -(log_prior+log_likelihood)
  return(negative_log_posterior)
}

alpha <- 2
beta <- 3

posterior_result <- optim(par = 5, fn = posterior_nll, pois = pois, alpha = alpha, beta = beta, method =
lambda_post_mode <- posterior_result$par
lambda_post_mode
```

```
## [1] 6.209302
```

This code defines a function to compute the negative log-posterior of a Poisson likelihood with a Gamma(2,3) prior and uses optim() to find the posterior mode (MAP estimate) of lambda. By minimizing the negative log-posterior, the optimization function effectively maximizes the posterior, giving the most likely value of lambda given both the data and prior information to be approximately 6.209.

Question 4 Part C

```
library(VGAM)
```

```
## Loading required package: stats4
## Loading required package: splines
```

```
set.seed(42)
n <- 100
data <- rpareto(n, scale = 1, shape = 1)

U <- sum(log(data))

head(data)
```

```
## [1] 1.093128 1.067150 3.494798 1.204170 1.558250 1.926426
```

```
eta_mle <- n / U
eta_mle
```

```
## [1] 0.9363
```

```
alpha <- 2
beta <- 0.5

alpha_post <- alpha + n
beta_post <- beta + U

alpha_post
```

```
## [1] 102
```

```
beta_post
```

```
## [1] 107.3034
```

```
# (c) Compute 95% credible interval using qgamma()

ci_lower <- qgamma(0.025, alpha_post, beta_post)  # Lower bound
ci_upper <- qgamma(0.975, alpha_post, beta_post)  # Upper bound
credible_interval <- c(ci_lower, ci_upper)

# Print credible interval
credible_interval
```

```
## [1] 0.775081 1.143712
```

The credible interval for part c is [0.775081, 1.14712].

Question 4 Part D

```r
# (d) Adjust hyperparameters to get a 95% credible interval of length ~1

# Modify prior parameters to produce a narrower credible interval
alpha_tuned <- 4.5 # Adjusted shape parameter (increase to reduce variance)
beta_tuned <- 8 # Adjusted rate parameter (increase to shift mass right)

# Compute updated posterior parameters
alpha_post_tuned <- alpha_tuned
beta_post_tuned <- beta_tuned

# Compute new 95% credible interval
ci_lower_tuned <- qgamma(0.025, alpha_post_tuned, beta_post_tuned)  # Lower bound
ci_upper_tuned <- qgamma(0.975, alpha_post_tuned, beta_post_tuned)  # Upper bound
adjusted_credible_interval <- c(ci_lower_tuned, ci_upper_tuned)

# Print adjusted credible interval
adjusted_credible_interval
```

```
## [1] 0.1687743 1.1889230
```

```r
adder <- (ci_upper_tuned - ci_lower_tuned)

adder
```

```
## [1] 1.020149
```

```r
eta_mle
```

```
## [1] 0.9363
```

The MLE is 0.9363, which is in the credible interval created assuming the hyper parameters alpha = 4.5 and beta = 8, and the length is approximately 1.