Assignment 4

anonymous

1 General information

2 Bioassay model

2.1 (a)

```
mu_a <- 0  # Mean of the first variable
mu_b <- 10  # Mean of the second variable
sigma_a <- 2  # Standard deviation of the first variable
sigma_b <- 10  # Standard deviation of the second variable
rho <- 0.6  # Correlation coefficient

cov_matrix <- matrix(c(sigma_a^2, rho * sigma_a * sigma_b, rho * sigma_a * sigma_b, sigma_b^2), nrow

mean_vector <- c(mu_a, mu_b)

print("Mean Vector:")
```

```
[1] "Mean Vector:"
```

```
print(mean_vector)
```

```
[1]  0 10
```

```
print("Covariance Matrix:")
```

```
[1] "Covariance Matrix:"
```

```
print(cov_matrix)
```

```
     [,1] [,2]
[1,]    4   12
[2,]   12  100
```

2.2 (b)

```r
alpha = bioassay_posterior[,1]
beta  = bioassay_posterior[,2]

sample_mean = c(mean(bioassay_posterior[,1]), mean(bioassay_posterior[,2]))

print(paste("Sample mean of alpha=", sample_mean[1]," and beta=",sample_mean[2]))
```

```
[1] "Sample mean of alpha= 0.985226289184767  and beta= 10.5964812910431"
```

```r
qa <- quantile(alpha, c(0.05,0.95))
qb <- quantile(beta, c(0.05,0.95))
mcseqa_lower <- mcse_quantile(alpha, 0.05)
mcseqa_upper <- mcse_quantile(alpha, 0.95)
mcseqb_lower <- mcse_quantile(beta, 0.05)
mcseqb_upper <- mcse_quantile(beta, 0.95)
print("")
```

```
[1] ""
```

```r
print(paste("Quantiles of alpha (0.05 and 0.95):", qa[1], qa[2]))
```

```
[1] "Quantiles of alpha (0.05 and 0.95): -0.467591355167553 2.61020281115318"
```

```r
print(paste("Quantiles of beta (0.05 and 0.95):", qb[1], qb[2]))
```

```
[1] "Quantiles of beta (0.05 and 0.95): 3.99140320865043 19.3403654436304"
```

```r
print("")
```

```
[1] ""
```

```r
print(paste("MCSE for quantile of alpha (0.05):", mcseqa_lower))
```

```
[1] "MCSE for quantile of alpha (0.05): 0.0260041159750117"
```

```r
print(paste("MCSE for quantile of alpha (0.95):", mcseqa_upper))
```

```
[1] "MCSE for quantile of alpha (0.95): 0.0420634167918579"
```

```r
print(paste("MCSE for quantile of beta (0.05):", mcseqb_lower))
```

```
[1] "MCSE for quantile of beta (0.05): 0.0704312509185216"
```

```
print(paste("MCSE for quantile of beta (0.95):", mcseqb_upper))
```

```
[1] "MCSE for quantile of beta (0.95): 0.241212887707373"
```

The mean of is 1.0, the mean of is 10.6 The quantiles for are [-0.5, 2.6], the quantiles for are [4, 20]

MCSE for quantile of alpha (0.05) = 0.03 MCSE for quantile of alpha (0.95) = 0.04

MCSE for quantile of beta (0.05) = 0.07 MCSE for quantile of beta (0.95) = 0.24

Likley variation to alphas 5% quantiles is +- 0.03 and 95% quantile is +- 0.04. From this we can interpret that it's very unlikely there would be variation in the reported estimate for the 95% as this would not change. Therefore we use report accurate alpha MCSE.

Likley variation to betas 5% quantiles is +- 0.07 (comparable quantile variation is 0.7) and 95% quantile is +- 0.024 (comparable quantile variation is 2.5) Therefore there is more variation in the higher quantile with it having a good chance to be wrong as through Montecarlo simulations showed that beta can be between 17 and 22.

The rule to choose how many digits to represent is based on how accurate our results are, the more accurate the more digits. The less accurate the less digits.

Loading the library and the data.

```
# Useful functions: quantile()
# and mcse_quantile() (from aaltobda)
data("bioassay_posterior")
# The 4000 draws are now stored in the variable `bioassay_posterior`.
# The below displays the first rows of the data:
head(bioassay_posterior)
```

```
        alpha      beta
1 -0.02050577 10.032841
2  1.21738518  4.504546
3  3.04829407 16.239424
4  1.32272770  4.924268
5  1.36274817 12.880561
6  1.08593225  5.943731
```

3 Importance sampling

3.1 (c)

```
# Useful functions: bioassaylp (from aaltobda)
alpha_test = c(1.896, -3.6,  0.374, 0.964, -3.123, -1.581)
beta_test = c(24.76, 20.04, 6.15, 18.65, 8.16, 17.4)
```

```
log_importance_weights <- function(alpha, beta) {
    return(bioassaylp(alpha, beta, bioassay$x, bioassay$y, bioassay$n))

}
log_importance_weights(alpha_test,beta_test)  #T c(-8.95, -23.47, -6.02, -8.13, -16.61, -14.57)
```

```
[1]  -8.954344 -23.468325  -6.015096  -8.130099 -16.613100 -14.573095
```

In importance sampling, we often use log importance ratios instead of ratios for numerical stability and precision. It is better since the troubles of values close to zero that arises when using floating point numbers is skipped.

3.2 (d)

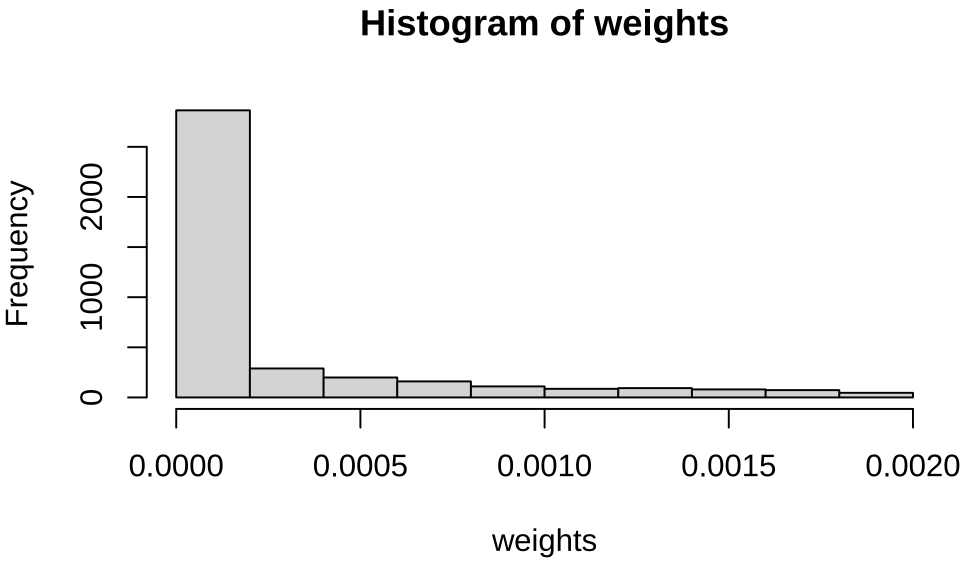
```
normalized_importance_weights <- function(alpha, beta) {

    weights <- log_importance_weights(alpha,beta)
    exponated_weights = exp(weights)
    return( exponated_weights / sum(exponated_weights) )

}
#normalized_importance_weights(alpha_test,beta_test) #Test with right answers = c(0.045, 0.000, 0.85
```

3.3 (e)

```
n <- 4000
alphaSample <- rnorm(n,mu_a,sigma_a)
betaSample <- rnorm(n,mu_b,sigma_b)
weights <- normalized_importance_weights(alphaSample,betaSample)
hist(weights)
```



The exponentiating returns the weights to regular weights from log weights. The summing to one and dividing by that makes them normalised so they can be used for further calculations.

3.4 (f)

```
S_eff <- function(alpha, beta) {
  weights <- normalized_importance_weights(alpha,beta)
  return(1/sum(weights^2))
}
round(S_eff(alpha = alphaSample, beta = betaSample),3)
```

```
[1] 949.201
```

```
#S_eff(alpha_test,beta_test) # right = 1.354
```

The importance sampling effective sample size is around 936.

3.5 (g)

It represents the approximate equivalent sample size if we were sampling the true distribution. In the histogram in assignment e) there are many larger weights thus the sample effective sample size is reduced from the original 4000.

3.6 (h)

Using equation 10.3 from the book, where h( ) is or to calculate E( ) or E( ). Importance sampling takes observations from a known sample and multiplies with weights that relate the known sample distribution with the unknown to get the expectation from the unknown distribution.

```
posterior_mean <- function(alpha, beta) {
  S <- length(alpha)
  weights <- normalized_importance_weights(alpha,beta)
  Ea <- (1/S*sum(alpha*weights))/(1/S*sum(weights))
  Eb <- (1/S*sum(beta*weights))/(1/S*sum(weights))
  return(c(Ea,Eb))
    c(0.503, 8.275)
}
```

```
MCSE <- function(alpha,beta) { S <- length(alpha)
  weights <- normalized_importance_weights(alpha,beta)
  Ea <- (1/S*sum(alpha*weights))/(1/S*sum(weights))
  Eb <- (1/S*sum(beta*weights))/(1/S*sum(weights))
  Ea2 <- (1/S*sum(alpha^2*weights))/(1/S*sum(weights))
  Eb2 <- (1/S*sum(beta^2*weights))/(1/S*sum(weights))
  Seff <- S_eff(alpha,beta)
  return(c(sqrt((Ea2-Ea^2)/Seff), sqrt((Eb2-Eb^2)/Seff)) )
}
```

```
MCSE(alphaSample,betaSample)
```

```
[1] 0.02807771 0.13953475
```

```
posterior_mean(alpha = alphaSample, beta = betaSample)
```

```
[1]  0.9782756 10.0648450
```

The mean is [1.0, 10]