Exercise 1

TMA4300 Computer Intensive Statistical Models

Mads Adrian Simonsen, William Scott Grundeland Olsen

03 februar, 2021

Problem A: Stochastic simulation by the probability integral transform and bivariate techniques

1.

2.

(a)

(b)

3.

(a)

(b)

(c)

4.

5

Problem B: The gamma distribution

1.

(a)

(b)

2.

(a)

(b)

3.

(a)

(b)

4.

5.

(a)

(b)

Problem C: Monte Carlo integration and variance reduction

1.

2.

3.

(a)

(b)

Problem D: Rejection sampling and importance sampling

Subproblem 1.

We consider a vector of multinomially distributed counts and the observed data is $\mathbf{y} = \begin{bmatrix} 125 & 18 & 20 & 34 \end{bmatrix}^\top$. The multinomial mass function is given as

$$f(\mathbf{y} \mid \theta) \propto (2+\theta)^{y_1}(1-\theta)^{y_2+y_3}\theta^{y_3},$$

and assuming a prior that is Uniform$(0,1)$ the posterior will be

$$f(\theta \mid \mathbf{y}) \propto f^*(\theta) := (2+\theta)^{y_1}(1-\theta)^{y_2+y_3}\theta^{y_3},$$

for $\theta \in (0,1)$. We wish to sample from this using a Uniform$(0,1)$ proposal density, that is, $g(\theta \mid \mathbf{y}) = 1$, for $\theta \in (0,1)$. To do a rejection sampling (not weighted rejection sampling), we need to know the normalizing constant of $f(\theta \mid \mathbf{y})$. That is, the constant k such that $f(\theta \mid \mathbf{y}) = kf^*(\theta \mid \mathbf{y})$. This can be found as

$$\frac{1}{K} = \int_{\mathbb{R}} f^*(\theta \mid \mathbf{y})\,\mathrm{d}\theta = \int_0^1 f^*(\theta \mid \mathbf{y})\,\mathrm{d}\theta \approx 2.3577 \cdot 10^{28},$$

and we find it using the `integrate()`-function in `R` below. To use the rejection sampling we also need that

$$\frac{f(\theta \mid \mathbf{y})}{g(\theta \mid \mathbf{y})} = f(\theta \mid \mathbf{y}) \leq k,$$

and a value for k is found in the code block below. We then simulate $\Theta \sim$ Uniform$(0,1)$ and $U \sim$ Uniform$(0,1)$ and calculate $\alpha = f(\theta \mid \mathbf{y})/k$. Then, if $U \leq \alpha$, Θ is returned, and if not, the procedure is run again. We then sample from the posterior distribution in the code block below.

```
y <- c(125, 18, 20, 34)   # Observed data

# Define the un-normalized posterior distribution f*(theta | y)
posterior_star <- function(theta, y) {
  return((2 + theta)^(y[1]) * (1 - theta)^(y[2] + y[3]) * theta^(y[4]))
}

# Find the normalizing constant 1 / K
norm_const <- integrate(function(theta)(posterior_star(theta, y)),
                        lower = 0,
                        upper = 1)$value

# Defining the normalized posterior distribution f(theta | y)
posterior <- function(theta, y) {
  return(posterior_star(theta, y) / norm_const)
}

# Finding the maximum
posterior_star_max <- optimize(function(theta)(posterior_star(theta, y)),
                               interval = c(0, 1),
                               maximum = TRUE)$objective

# k such that f(theta | y) <= k
k <- posterior_star_max / norm_const

# Rejection sampling algorithm
rejection_sampling <- function(M, y) {
  n <- 10 * M   # Wish M samples from f, need to generate more from Unif(0, 1)
  Theta <- runif(n)
  U <- runif(n)
  count <- 0      # Count how many times the algorithm runs
  accept <- c()   # List of the accepted samples
```

```
  while(length(accept) < M) {
    test_u <- U[count]
    alpha <- posterior(Theta[count], y) / k
    if(test_u <= alpha) {
      accept <- rbind(accept, Theta[count])
      count <- count + 1
    } else {
      count <- count + 1
    }
  }
  return(list("accept" = accept, "co" = count))
}
```

Subproblem 2.

Drawing $\Theta_1, \ldots, \Theta_M \sim f(\theta \mid \mathbf{y})$, the Monte Carlo estimate of $\mu = \mathrm{E}(\theta \mid \mathbf{y})$ is

$$\hat{\mu} = \frac{1}{M} \sum_{i=1}^{M} \Theta_i.$$

We do this for $M = 10000$ in the code block below. Figure 1 shows the result of this. We see the estimation of the posterior mean $\mathrm{E}(\theta \mid \mathbf{y})$ using Monte Carlo integration and numerical integration together with the theoretical posterior density distribution and a generated histogram of the samples. In the figure the posterior density is plotted using a normalizing constant we find by numerical integration in `R` below, giving the normalizing constant `norm_const`.

```
M <- 10000    # Number of samples from f(theta | y)

Theta_samp <- rejection_sampling(M, y)   # M samples from f(theta | y)
mu_est <- mean(Theta_samp$accept)    # = 1/M * sum(Theta_samp)

mu_num <- integrate(function(theta)(theta * posterior(theta, y)),
                    lower = 0,
                    upper = 1)$value    # Value of mu by numerical integration

# Plot
ggplot() +
  geom_histogram(
    data = as.data.frame(Theta_samp$accept),
    mapping = aes(x = Theta_samp$accept, y = ..density..),
    binwidth = 0.01,
    boundary = 0
  ) +
  stat_function(
    fun = posterior,
    args = list(y = y),
    aes(col = "Posterior density")
  ) +
  geom_vline(
    aes(xintercept = c(mu_est, mu_num),
        col = c("Estimated posterior mean", "Numerical posterior mean"),
        linetype = c("dashed", "dotted"))
  ) +
  guides(linetype = FALSE) +    # Remove linetype from label
```

```
  ggtitle("Estimation of the posterior mean") +
  xlab("theta") +
  ylab("Density") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5)) +
  theme(legend.title = element_blank())
```

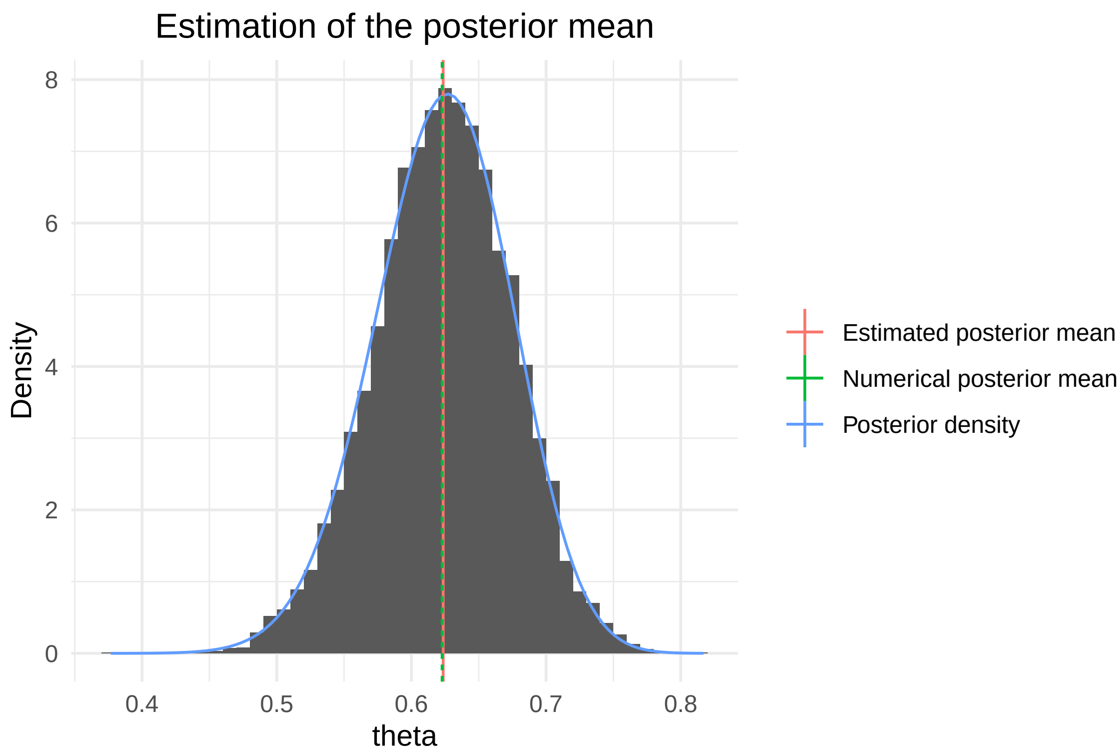

Figure 1: Estimation of the posterior mean $\mathrm{E}(\theta \mid \mathbf{y})$ using Monte Carlo integration and numerical integration. A histogram of the samples is also shown together with the theoretical posterior density distribution.

In the following code block we find the values of `mu_est` and `mu_num`.

```
mu_est
```

```
## [1] 0.6225143
```

```
mu_num
```

```
## [1] 0.6228061
```

From this it is clear that the estimated posterior mean is $\hat{\mu} \approx 0.623$ using Monte Carlo integration, and $\mu \approx 0.623$ using numerical integration with `integrate()`. Figure 1 also shows that these means corresponds well to the real posterior mean.

Subproblem 3.

We are now interested in the number of random numbers the sampling algorithm needs to obtain one sample from $f(\theta \mid \mathbf{y})$. The expected number of trials up to the first sample from $f(\theta \mid \mathbf{y})$ is c given by the condition

$$\frac{f(\theta \mid \mathbf{y})}{g(\theta \mid \mathbf{y})} = f(\theta \mid \mathbf{y}) \leq c.$$

We may then choose

$$c \geq \max_{\theta \in [0,1]} f(\theta \mid \mathbf{y}),$$

and we choose the equality. Thus we may find c in `R` using the `optimize()`-function, and call this `const_num` to symbolize that this is the numerically calculated value, as in the following code block.

```
const_num <- optimize(function(theta)(posterior_f(theta, y)),
                      interval = c(0, 1),
                      maximum = TRUE)$objective
const_num
```

```
## [1] 7.799308
```

Using the sampler, the expected number of random numbers that has to be generated in order to obtain one sample of $f(\theta \mid \mathbf{y})$ is given in the following code block.

```
Theta_samp$co / M
```

```
## [1] 7.8237
```

Subsection 4.

...