Assignment 2 - Stochastic processes and Bayesian inference (MVE550)

Luca Modica
Hugo Manuel Alves Henriques e Silva
Linus Haraldsson

November 30, 2023

Part 1

Question (a)

1. Consider a discrete-time Markov chain with possible states $1, 2, 3$. The transition matrix P is unknown, but we have observed the first 12 steps of a realization of the chain:

$$1, 2, 3, 2, 3, 1, 2, 1, 3, 2, 1, 3.$$

Let P_1, P_2, and P_3, be the rows of P. We assume independent Dirichlet priors for these, with all pseudo-counts equal to 1.

(a) Given the data, describe the posterior distributions for P_1, P_2, and P_3, and thus for P. Compute the expectation of the posterior for P.

a)

Count Matrix

	1	2	3
1	0	2	2
2	2	0	2
3	1	2	0

$P_1 \sim$ Dirichlet $(1,1,1)$

$P_2 \sim$ Dirichlet $(1,1,1)$

$P_3 \sim$ Dirichlet $(1,1,1)$

$P_1 \mid \text{data} \sim$ Dirichlet $(1,3,3)$

$P_2 \mid \text{data} \sim$ Dirichlet $(3,1,3)$

$P_3 \mid \text{data} \sim$ Dirichlet $(2,3,1)$

$E[P \mid \text{data}] =$

	1	2	3
1	1/7	3/7	3/7
2	3/7	1/7	3/7
3	2/6	3/6	1/6

$E[P_1 \mid \text{data}] = \frac{(1,3,3)}{1+3+3}$

$E[P_2 \mid \text{data}] = \frac{(3,1,3)}{3+1+3}$

$E[P_3 \mid \text{data}] = \frac{(2,3,1)}{2+3+1}$

Question (b)

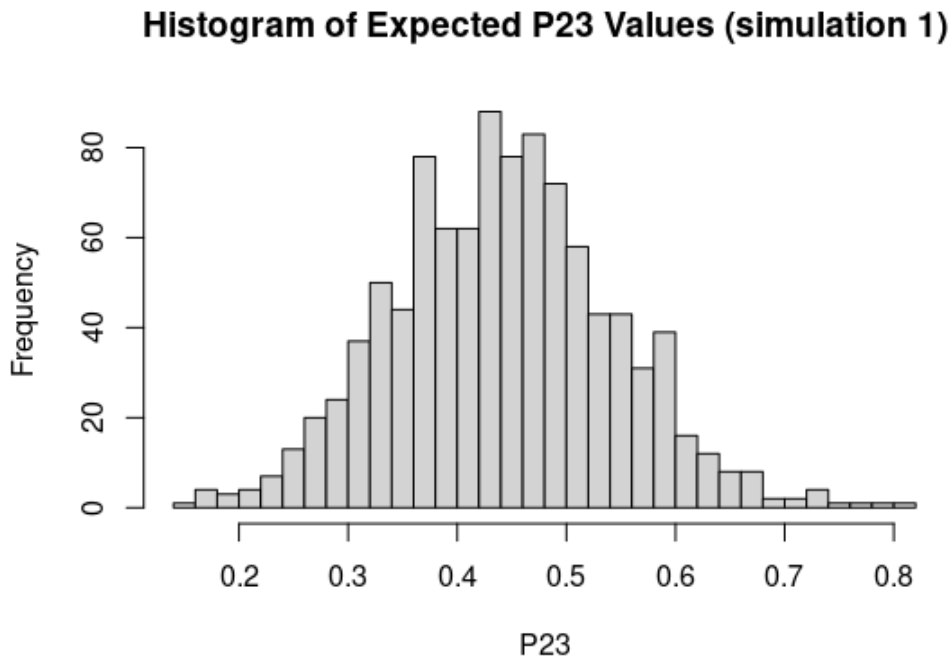
```r
simulate1 <- function() {
  initial_chain <- c(1, 2, 3, 2, 3, 1, 2, 1, 3, 2, 1, 3)
  transition_counts <- matrix(1, nrow = 3, ncol = 3)

  # Update counts with initial chain
  for (i in 1:(length(initial_chain) - 1)) {
    transition_counts[initial_chain[i], initial_chain[i + 1]] <-
        transition_counts[initial_chain[i], initial_chain[i + 1]] + 1
  }

  expected_transition_matrix <- apply(transition_counts, 1, function(row) row / sum(row))
  total_length <- 500
  current_state <- initial_chain[length(initial_chain)]

  # Simulate the rest of the chain
  for (i in (length(initial_chain) + 1):total_length) {
    transition_prob <- expected_transition_matrix[current_state, ]
    next_state <- sample(1:3, size = 1, prob = transition_prob)

    transition_counts[current_state, next_state] <- transition_counts[current_state,
        next_state] + 1
    expected_transition_matrix <- apply(transition_counts, 1, function(row) row / sum(row))
    current_state <- next_state
  }

  return(expected_transition_matrix)
}
```

Expected value of P when the data is all observed and all simulated values:

$$P = \begin{pmatrix} 0.02013423 & 0.7114094 & 0.2684564 \\ 0.5513514 & 0.1297297 & 0.3189189 \\ 0.2471264 & 0.316092 & 0.4367816 \end{pmatrix}.$$

Question (c)

Histogram:



The code used to produce the histogram:

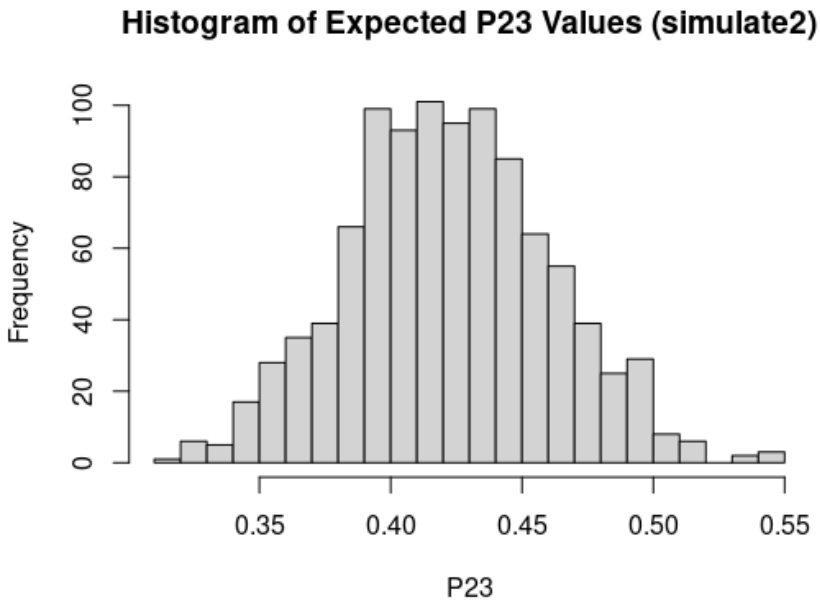
```r
# Function to extract P23 from the transition matrix
extract_P23 <- function(matrix) {
  return(matrix[2, 3])
}
P23_values <- replicate(1000, extract_P23(simulate1()))
hist(P23_values, main = "Histogram of Expected P23 Values (simulation 1)", xlab = "P23",
    breaks = 30)
```

Question (d) & (e)

Expected value of P in simulation 2:

$$P = \begin{pmatrix} 0.1722222 & 0.4333333 & 0.3944444 \\ 0.4267516 & 0.0955414 & 0.477707 \\ 0.4795322 & 0.374269 & 0.1461988 \end{pmatrix}.$$

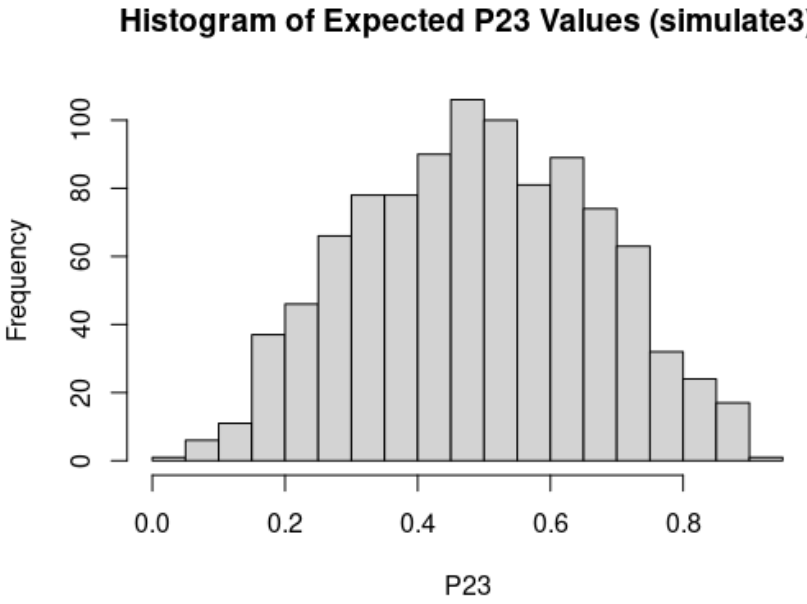
Histogram for simulation 2:



Value of P in simulation 3:

$$P = \begin{pmatrix} 0.1377246 & 0.497006 & 0.3652695 \\ 0.3965517 & 0.137931 & 0.4655172 \\ 0.4431138 & 0.4071856 & 0.1497006 \end{pmatrix}.$$

Histogram for simulation 3:



Comparison of histograms created by simulate1, simulate2 and simulate3.

For ***simulate1*** we utilize dynamic updating. The expected transition matrix is continuously updated at every single step of the simulation. Therefore, there is a continuous learning process where the simulated data adjusts the transition probabilities and how we understand them. The Markov chain essentially adapts as it grows. This approach aligns with Bayesian updating, where prior beliefs are continuously updated with new evidence. The results for P23 on this simulation show greater variability as they may vary based on the outcomes of the first simulations.

For ***simulate2*** we utilize the fixed initial matrix calculated only from the initially observed data, for every single simulated step. In this case there is no adaptation of the transition matrix but the final expected transition matrix is computed using all data. In this scenario, it is important to refer that the transition probabilities remain constant and are not affected by the chain's progression. A static approach considers that the observed data is robust enough to model the future states without having to progressively update the transition probabilities. As for the values of P23, in this case they will be more uniform compared to *simulate1* given that the simulated states do not influence the transition probabilities.

Finally, in ***simulate3*** we resort to sampling the transition matrix P from the posterior distribution that was obtained in question a), before the simulation starts. We then continue using this sampled matrix throughout the whole simulation. Despite the probabilities remaining constant during the execution of each simulation, they are different from simulation to simulation, which is one way to introduce variability in the results. Therefore, the obtained values for P23 can demonstrate a different type of variability which shows the diversity of possible transition matrices generated from the posterior.

Code for simulation 2 and the related histogram:

```r
simulate2 <- function() {
  initial_chain <- c(1, 2, 3, 2, 3, 1, 2, 1, 3, 2, 1, 3)
  transition_counts <- matrix(1, nrow = 3, ncol = 3)

  # Update counts with initial chain
  for (i in 1:(length(initial_chain) - 1)) {
    transition_counts[initial_chain[i], initial_chain[i + 1]] <-
        transition_counts[initial_chain[i], initial_chain[i + 1]] + 1
  }

  initial_expected_matrix <- apply(transition_counts, 1, function(row) row / sum(row))
  total_length <- 500
  current_state <- initial_chain[length(initial_chain)]

  # Simulate the rest of the chain
  for (i in (length(initial_chain) + 1):total_length) {
    transition_prob <- initial_expected_matrix[current_state, ]

    next_state <- sample(1:3, size = 1, prob = transition_prob)
    transition_counts[current_state, next_state] <- transition_counts[current_state,
        next_state] + 1
    current_state <- next_state
  }
  final_expected_matrix <- apply(transition_counts, 1, function(row) row / sum(row))

  return(final_expected_matrix)
}

simulate2()

P23_values_simulate2 <- replicate(1000, extract_P23(simulate2()))
hist(P23_values_simulate2, main = "Histogram of Expected P23 Values (simulate2)", xlab =
    "P23", breaks = 30)
```

Code for simulation 3 and the related histogram:

```r
simulate3 <- function() {
  initial_chain <- c(1, 2, 3, 2, 3, 1, 2, 1, 3, 2, 1, 3)
  transition_counts <- matrix(1, nrow = 3, ncol = 3)

  # Update counts with initial chain
  for (i in 1:(length(initial_chain) - 1)) {
    transition_counts[initial_chain[i], initial_chain[i + 1]] <-
        transition_counts[initial_chain[i], initial_chain[i + 1]] + 1
  }

  # Parameters for the Dirichlet posterior for each row of P
  dirichlet_parameters <- list(c(1, 3, 3), c(3, 1, 3), c(2, 3, 1))

  sampled_transition_matrix <- t(sapply(dirichlet_parameters, rdirichlet, n = 1))
  total_length <- 500

  # Simulate the rest of the chain
```

```
  current_state <- initial_chain[length(initial_chain)]
  for (i in (length(initial_chain) + 1):total_length) {
    # Simulate the next state based on the sampled transition matrix
    transition_prob <- sampled_transition_matrix[current_state, ]

    next_state <- sample(1:3, size = 1, prob = transition_prob)
    transition_counts[current_state, next_state] <- transition_counts[current_state,
        next_state] + 1
    current_state <- next_state
  }

  final_expected_matrix <- apply(transition_counts, 1, function(row) row / sum(row))
  return(final_expected_matrix)
}

P23_values_simulate3 <- replicate(1000, extract_P23(simulate3()))
hist(P23_values_simulate3, main = "Histogram of Expected P23 Values (simulate3)", xlab =
    "P23", breaks = 30)
```

All in all, ***simulate1*** is a continuous Bayesian update, while ***simulate2*** is a static empirical approach, and ***simulate3*** reflects the exploration of different probabilistic scenarios.

Question (f)

i) $P_1 \sim \text{Dirichlet}(0,1,1)$
$P_2 \sim \text{Dirichlet}(1,0,1)$
$P_3 \sim \text{Dirichlet}(1,1,0)$

$P_1 \mid \text{data} \sim \text{Dirichlet}(0,3,3)$
$P_2 \mid \text{data} \sim \text{Dirichlet}(3,0,3)$
$P_3 \mid \text{data} \sim \text{Dirichlet}(2,3,0)$

$E[P \mid \text{data}] =$

	1	2	3
1	0	3/6	3/6
2	3/6	0	3/6
3	2/5	3/5	0

$E[P_1 \mid \text{data}] = \frac{(0,3,3)}{3+3}$

$E[P_2 \mid \text{data}] = \frac{(3,0,3)}{3+3}$

$E[P_3 \mid \text{data}] = \frac{(2,3,0)}{2+3}$

Part 2

In this part, we assume a branching process with the following data.

- It's been observed for the following generations:

$$Z_0 = 1, Z_1 = 1, Z_2 = 2, Z_3 = 5, Z_4 = 7.$$

- The offspring distribution is Poisson, with expectation $\mu = \lambda$.
- It's assumed an improper prior $\pi(\lambda) \propto_\lambda 1/\lambda$ for λ.

Question (a)

Recall the offspring distribution is the following:

$$P(Z_n = k_n) = e^{-\lambda}\frac{\lambda^{k_n}}{k_n!}, \text{for } n \geq 0.$$

Using the information about the offspring distribution and the observed data, we can compute the posterior distribution of λ in the following way:

$$\begin{aligned}
&\pi(\lambda|k_0...k_4) \\
&\propto_\lambda \pi(k_0...k_4|\lambda)\pi(\lambda) \\
&\propto_\lambda e^{-\lambda}\lambda^k\frac{1}{k} = e^{-\lambda}\lambda^{k-1} \\
&\propto_\lambda e^{-2\lambda}(e^{-\lambda}\lambda)(e^{-\lambda}\lambda^4)(e^{-\lambda}\lambda^6) = e^{-3\lambda}\lambda^{11}.
\end{aligned}$$

Moreover, based on the information in the information we have about the offspring distribution, we can also look to the *Poisson - Gamma conjugacy*. In other words:

$$\begin{aligned}
&\lambda|k_0 \sim \text{Gamma}(0+1, 0+1) \\
&\lambda|k_0, k_1 \sim \text{Gamma}(1+1, 1+1) \\
&... \\
&\lambda|k_0, ..., k_4 \sim \text{Gamma}(16, 5)
\end{aligned}$$

$$\pi(\lambda|k) \propto_\lambda \text{Gamma}(\lambda, 16, 5).$$

In conclusion we can say that the posterior distribution for λ, using the information provided, is computed with $\text{Gamma}(\lambda, 16, 5)$.

Related function in R:

```r
# posterior for lambda, derived from question 2a.
posterior_lambda <- function(lam, data) {
  return (dgamma(lam, sum(data), length(data)));
}

# computed posterior for the observed data
# (Z0, Z1, Z2, Z3, Z4) = (1, 1, 2, 5, 7)
posterior_Z0_Z4 <- function(lam) {
  return (posterior_lambda(lam, data_Z0_Z4));
};
```

Question (b)

To write a function that outputs the probability that a branching process will become extinct, we need to derive the **probability generation function** (pgf) first. The pgf for the Poisson distribution is computed as follows:

$$G(s) = E(s^Z) = \sum_{k=0}^{\infty} s^k P(Z = k) = \sum_{k=0}^{\infty} s^k e^{-\lambda} \frac{\lambda^k}{k!} = e^{-\lambda} \sum_{k=0}^{\infty} \frac{(s\lambda)^k}{k!} = e^{-\lambda} e^{\lambda s} = e^{\lambda(s-1)}.$$

Using the derived pgf, we then wrote a function in R that takes as input a value for λ and outputs the probability that this branching process will become extinct.

```r
extinction_prob <- function(lam) {
  pgf <- function(s) { exp(lam * (s - 1)) }

  # Define the function to find a root for (pgf(s) - s)
  f_to_minimize <- function(s) { pgf(s) - s }

  # Choose the interval to search for a root
  interval <- if (lam <= 1) c(0, 1) else c(0, 0.999)

  # Use uniroot to find the root of the function on the interval.
  # The root is the fixed point which corresponds to the extinction
  # probability
  root <- uniroot(f_to_minimize, interval)$root;

  return(root)
}
```

Question (c)

To compute the probability of extinction taking the uncertainty in λ into account one method is to integrate over all possible values of the parameter, weighting each possible value of λ by its probability given the data.

This implies writing an integral representing this probability in terms of:

- the posterior of λ (that is, the probability of seeing a value of λ, given the already observed data), obtained from question (a);
- the probability of extinction given a specific value of λ, using the function derived from question (b).

Let's denote the extinction probability for a given λ as $E(\lambda)$. The extinction probability considering the uncertainty of λ can be represented as follows:

$$\int_0^{\infty} E(\lambda)\pi(\lambda|Z_0...Z_4)\, d\lambda.$$

The integral is over the entire parameter space of λ, which is $[0, \infty)$ for a Poisson distribution. It calculates the expected value of the extinction probability, with respect to the posterior distribution of the parameter from question (a).

With the following code, instead, the integral above is computed using numerical integration.

```r
# question 2c: extinction probability of a branching process
# taking the uncertainty of lambda into account, in terms of
# the already computed posterior distribution and the extinction
extinction_prob_2c <- function() {
  return(integrate(Vectorize(function(lam) { extinction_prob(lam) * posterior_Z0_Z4(lam) }),
      0, Inf)$value);
}
cat("Probability of extinction, using numerical integration: ", extinction_prob_2c(), "\n");
```

```
Probability of extinction, using numerical integration: 0.07161584
```

Question (d)

The results from the numerical integration in c) and the simulation in d) are similar and look reasonable. This should indicate that our calculations are correct. Results can be seen below the code.

```r
# simulation of extinction probability (question 2d)
# for a branching process with Poisson offspring distribution
# and parameter lambda. The lambda value will be sampled
# from the same posterior distribution computed in 2a, to
# keep the uncertainty of lambda and the observed data
# into account.
extinction_prob_simulation_bayes <- function(n_sims, gens) {
  simlist <- replicate(n_sims, branch(gens, sample_posterior())[11]);
  return (sum(simlist==0)/n_sims);
}
cat("Probability of extinction, using simulations: ", extinction_prob_simulation_bayes(5000,
    10), "\n");

Probability of extinction, using simulations: 0.0728
Probability of extinction, using numerical integration: 0.07161584
```


```
Probability of extinction, using simulations: 0.0728
Probability of extinction, using numerical integration: 0.07161584
```


Question (e)

The likelihood, the function of λ which for each λ gives the probability of the observations, is the following:

$$L(Z_0...Z_4) = \pi(Z_0...Z_4|\lambda).$$

Note that is the same function used to derive the posterior of λ in question (a).

The code below instead calculates the Maximum Likelihood Estimate (MLE) value for λ, maximizing the function above. Then, It calculates the probability of extinction given that λ.

```r
# question 2e: maximum likelihood estimation
#Calculate likelihood of data given a specific lambda
likelihood_data_Z0_Z4 <- function(lam) {
```

```
  return (prod(dpois(data_Z0_Z4, lam)));
}
#Find the lambda resulting in the greatest likelyhood of reciving the given datapoints
maximum_likelihood_lambda <- function(data) {
  return (optimize(Vectorize(likelihood_data_Z0_Z4), c(0, 100), maximum = TRUE)$maximum);
}
cat("Maximum likelihood estimate of lambda: ", maximum_likelihood_lambda(data_Z0_Z4), "\n");
cat("Probability of extinction, using maximum likelihood estimate: ",
    extinction_prob(maximum_likelihood_lambda(data_Z0_Z4)), "\n");
```

```
Maximum likelihood estimate of lambda: 3.200017
Probability of extinction, using the maximum likelihood estimate: 0.04744449
```