Exercise sheet 2

Prepare the below such that you are able to discuss it on Wednesday 10th March

Exercise 1 Continuous time stochastic SIR model

The simple continuous time stochastic SIR model was in the lecture formulated as a stochastic process with piecewise constant conditional intensity function (CIF).

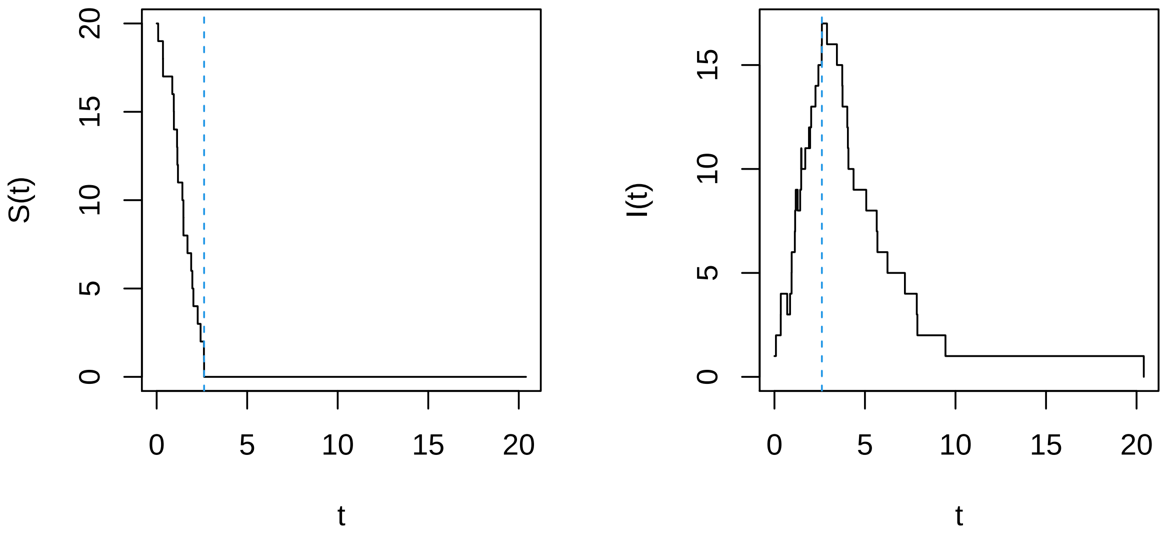
a) Write an R function which creates one realization of this model with inputs $\theta = (\beta, \gamma)'$ and $(s(0), i(0))'$. The function should return a matrix with 3 columns where each row contains the event time and the number of susceptibles at that event time.
Solution:

```r
# Function for converting missings to infinite values
NA2Inf <- function(x) {x[is.na(x)] <- Inf; return(x)}

simulate_sir <- function(theta, si0) {
  beta <- theta[1]
  gamma <- theta[2]
  t <- 0

  # Results should be given in a three-dimensional matrix
  # with columns t, S(t), I(t)
  res <- matrix(c(t, si0), ncol = 3)
  colnames(res) <- c("t", "s", "i")

  # If I(t) > 0 continue
  while(tail(res, n = 1)[, 3] > 0) { # Check for above requirement
    sit <- tail(res, n = 1)[, 2 : 3]
    # Intensity of the two competing events F
    cif <- c("S->I" = beta * sit[1] * sit[2],
             "I->R" = gamma * sit[2]) # Transitions
    # Simulate waiting times from an exponential distribution and
    # apply the function that makes missing infinite
    waiting_time <- NA2Inf(rexp(2, rate = cif))

    if (waiting_time[1] < waiting_time[2]) { #S->I event occurs
      t <- t + waiting_time[1]
      sit <- c(sit[1] - 1, sit[2] + 1)
    } else {
      t <- t + waiting_time[2]
      sit <- c(sit[1], sit[2] - 1)
    }
    res <- rbind(res, c(t, sit))
  }
  return(as.data.frame(res))
}
```

b) Test your function from the previous question with the inputs values $\theta = (0.1, 0.3)'$ and

$(s(0), i(0))' = (20, 1)'$.
(*Hint*: A graphical representation may help visualise the results)
Solution:

```r
# Use the inputs provided
set.seed(20210303)
traj <- simulate_sir(theta = c(beta = 0.1, gamma = 0.3), si0 = c(20, 1))
# Plot the results
par(mfrow = c(1, 2))
plot(traj$t, traj$s, type = "s", ylab = "S(t)", xlab = "t")
abline(v = traj$t[min(which(traj$s == 0))], col = 4, lty = 2)
# Add line for first instance where S(t) = 0
plot(traj$t, traj$i, type = "s", ylab = "I(t)", xlab = "t")
abline(v = traj$t[min(which(traj$s == 0))], col = 4, lty = 2)
```



```r
# Add line for first instance where S(t) = 0
```

The curves for changes to the susceptible and infected compartments look as would be expected from a simple SIR model (Susceptibles are decreasing S-shape and I is bell-shaped). The peak of I is around the time where S has depleated and I continues to decrease thereafter (this makes sense as there are no more people available to be infected). The curve for the R component would similarly be expected to grow from around this time point.

Exercise 2 Reed-Frost model

The dynamics of the Reed-Frost model can be described by the discrete-time Markov chain

$$Y_{t+1}|x_t, y_t \sim \text{Bin}(x_t, 1-(1-w)^{y_t}),$$
$$X_{t+1} = X_t - Y_{t+1},$$

with $x_0 = n$ susceptible and $y_0 = m$ infectious individuals initially and w is the probability of an infectious contact between susceptible individual i and infectious individual j in interval $(t, t+1]$. The Reed-Frost model relates to the Kermack-McKendrick SIR formulation of models since Y describes changes to the I compartment and X describes changes to the S compartment.

a) Create an R function which takes inputs `w`, `x0`, and `y0` which simulates one instance of a Reed-Frost model with probability w and initial values $x_0 =$ `x0` and $y_0 =$ `y0`. The function should return a `matrix` with two columns such that each row contains (x_t, y_t) for $t = 0, 1, 2, \ldots$. (*Hint*: You may want to use `rbind`)
Solution:

```r
reed_frost <- function(w, x0, y0) {
  XY <- matrix(c(x0, y0), ncol = 2) # Initial values
  t <- 1 # Initialise step in loop
  while (all(XY[t, ] > 0)) { # Stopping criterion for while-loop
    theta <- 1 - (1 - w) ^ XY[t, 2] # Parameter for binomial distribution
    ytp1 <- rbinom(n = 1, size = XY[t, 1], prob = theta) # Calculate Y_{t + 1}
    xtp1 <- XY[t, 1] - ytp1 # Calculate X_{t + 1} with Y_{t + 1} from above
    XY <- rbind(XY, c(xtp1, ytp1)) # Add to matrix object
    t <- t + 1 # Increase step
  }
  return(XY)
}
```

b) Test this function with the probability $w = 0.15$ and initialisation $x_0 = 20, y_0 = 1$. (*Hint*: You may want to set a seed such that you can use the output elsewhere)
Solution:

```r
set.seed(20210224)
reed_frost(w = 0.15, x0 = 20, y0 = 1)

##      [,1] [,2]
## [1,]   20    1
## [2,]   18    2
## [3,]   11    7
## [4,]    5    6
## [5,]    3    2
## [6,]    2    1
## [7,]    2    0
```

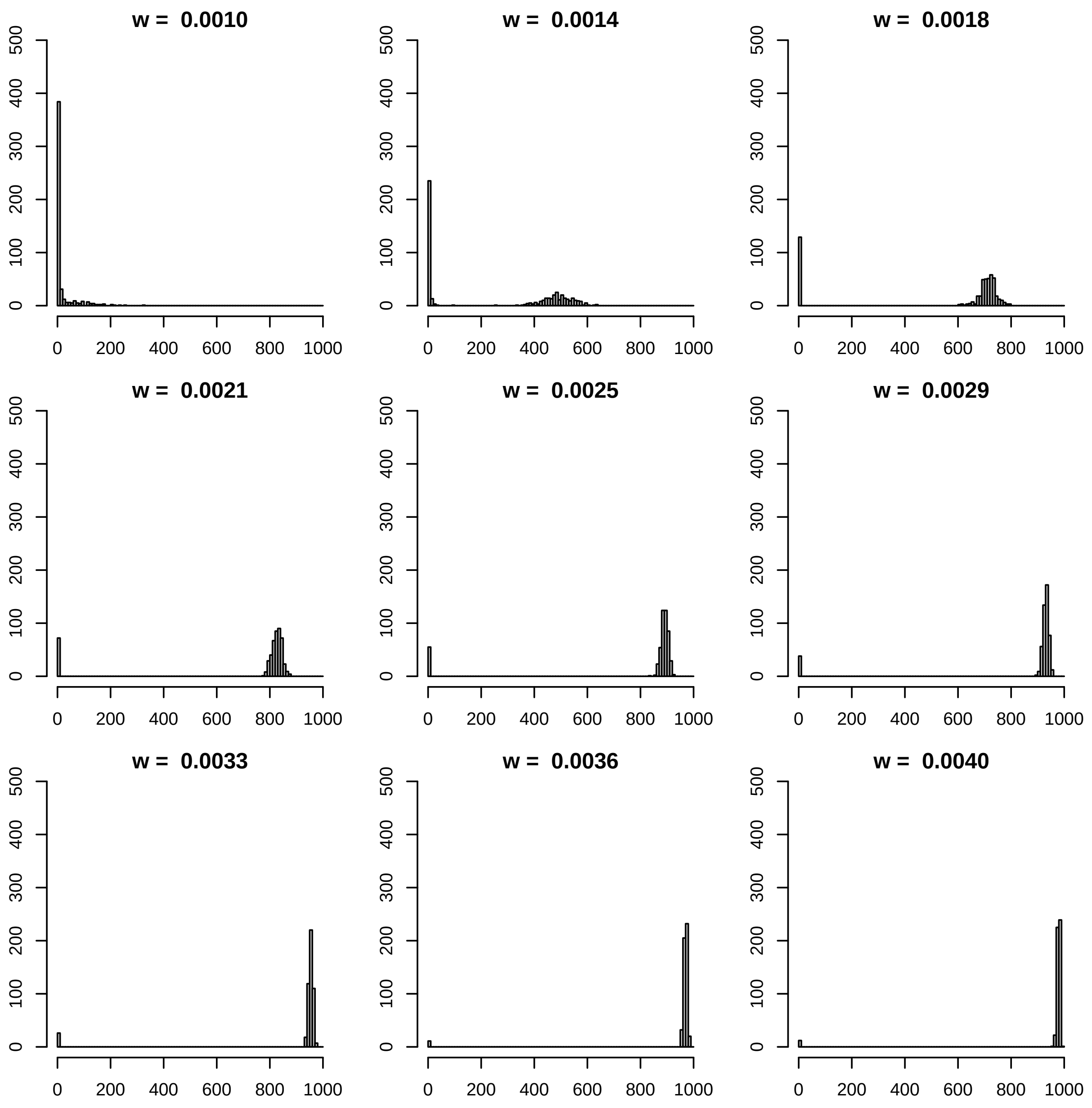
c) Recall the final size of the epidemic is $Z = Y_1 + Y_2 + Y_3 + \ldots$. Given the output of your function from question 1 a), construct another R function which takes the output of the previous function as input and calculates Z. Calculate this for the values given in question 1 b).
Solution:

```r
final_size <- function(mat) {
  return(sum(mat[- 1, # To ensure summing from Y_1 not Y_0
                 2, # Interested in second column (the Y_i's)
                 drop = FALSE]))
}
set.seed(20210224)
final_size(reed_frost(w = 0.15, x0 = 20, y0 = 1))

## [1] 18
```

d) Now consider $x_0 = 1000$ and $y = 1$. Describe how the final size Z relates to w. (*Hint*: A graphical representation may help visualise what is going on)

Solution:

```r
x0 <- 1000
par(mfrow = c(3, 3), mar = c(3, 3, 1, 1))
for (w in seq(0.001, 0.004, length = 9)) {
  hist(
    replicate(500, final_size(reed_frost(w = w, x0 = x0, y0 = 1))),
    # Calculate 500 instances of the Reed-Frost model with
    # x = 1000, y = 1
    # for various w
    breaks = seq(0, x0, by = 10),
    main = paste("w = ", sprintf("%.4f", w)), # Add descriptive titles
    xlab = "Z",
    freq = TRUE, ylim = c(0, 500))
}
```



We see that when w is smaller, i.e. the probability of an infectious contact decreases, the final size of the epidemic also decreases.

e) Given the values

	t_0	t_1	t_2	t_3	t_4	t_5
x_t	20	18	12	4	2	0
y_t	1	2	6	8	2	2

estimate w and its associated 95% Wald confidence interval using maximum likelihood estimation.
(*Hint*: You may want to use a parameter transformation for the probability parameter, e.g. based on the inverse cdf of the logistic distribution as well as `optim`)
Solution: Recall that the Wald CI is given by α

```
l <- function(w_logit, x, y) {
  if (length(x) != length(y)) { stop("x and y need to be the same length") }
  K <- length(x)
  w <- plogis(w_logit)
  theta <- 1 - (1 - w) ^ y
  #Compute loglik
  return(sum(dbinom(y[- 1], size = x[- K], prob = theta[- K], log = TRUE)))
}
mle <- optim(par = 0, fn = l, method = "BFGS",
             x = c(20, 18, 12, 4, 2, 0),
             y = c(1, 2, 6, 8, 2, 2),
             control = list(fnscale = - 1), hessian = TRUE)
#Maximum likelihood estimator
plogis(mle$par)

## [1] 0.1574987

#95% confidence interval
c(plogis(mle$par - qnorm(0.975) * sqrt( - 1 / mle$hess)),
  plogis(mle$par + qnorm(0.975) * sqrt( - 1 / mle$hess)))

## [1] 0.1027789 0.2337612
```