1. (a) Give a Metropolis-Hastings algorithm with a stationary Gamma probability density function,
\[ \pi(x) \propto x^{\alpha-1} \exp(-\beta x), \quad x > 0 \]
with parameters \( \alpha, \beta > 0 \). Use the proposal distribution \( Y \sim \text{Exp}(\beta) \).
(b) Write an R function implementing your MCMC algorithm. Your function should take as input values for \( \alpha \) and \( \beta \) and a number \( n \) of steps and return as output a realization \( X_1, X_2, ..., X_n \) of a Markov chain targeting \( \pi \). State briefly how you checked your code.

2. MCMC for Bayesian inference (first two parts were an exam Q in 2009)
(a) Let \( X \sim \text{Binomial}(n, r) \) be a binomial random variable with \( n \) trials and success probability \( r \). Let \( \pi(x; n, r) \) be the pmf of \( X \). Give a Metropolis-Hastings Markov chain Monte Carlo algorithm with stationary pmf \( \pi(x; n, r) \).
(b) Suppose the success probability for \( X \) is random, with \( \Pr(R = r) = p(r) \) given by
\[ p(r) = \begin{cases} \quad r & \text{for } r \in \{1/2, 1/4, 1/8, \ldots\}, \\ \quad 0 & \text{otherwise}. \end{cases} \]
An observed value \( X = x \) of the Binomial variable in part (a) is generated by simulating \( R \sim p \) to get \( R = r^* \) say, and then \( X \sim \text{Binomial}(n, r^*) \) as before. Specify a Metropolis-Hastings Markov chain Monte Carlo algorithm simulating a Markov chain, \( (R_t)_{t=0,1,2,...} \) with equilibrium probability mass function \( R_t \overset{d}{\rightarrow} p(r|x) \) where
\[ p(r|x) \propto \pi(x; n, r)p(r) \]
is called the posterior distribution for \( r \) given data \( x \).
(c) Write an R function implementing your MH MCMC algorithm with target distribution \( p(r|x) \). Suppose \( n = 10 \) and we observe \( x = 0 \). Run your MCMC algorithm and estimate the mode of \( p(r|x) \) over values of \( r \).

3. Let \( X \) be an \( n \times p \) matrix of fixed covariates with \( n > p \), and suppose that \( X \) has full column rank \( p \).
(a) Explain why the \( p \times p \) matrix \( X^TX \) is invertible.
Consider the linear model given by
\[ Y_i = \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} + \varepsilon_i, \]
where \( \varepsilon_i \sim N(0, \sigma^2) \).
(b) Write down the distribution of \( Y_i \), and use it to write out the log-likelihood for \( \beta = (\beta_1, \ldots, \beta_p) \).
(c) Show that the MLE is equivalent to minimising the sum of squares:
\[ R(\beta) = \sum_{i=1}^{n} (Y_i - \beta_1 x_{i1} - \cdots - \beta_p x_{ip})^2. \]
(d) By differentiating and writing the problem as a system of linear equations, show that the MLE is \( \hat{\beta} = (X^T X)^{-1} X^T Y. \)

4. Consider the linear model \( Y = X\beta + \epsilon \) where \( Y \) is a vector of \( n \) observations, \( X \) is an \( n \times p \) matrix with each column containing a different explanatory variable and \( \epsilon \) is a vector of \( n \) independent normal random errors with mean zero and unknown variance \( \sigma^2 \). The maximum likelihood estimator for \( \beta \) is \( \hat{\beta} = (X^T X)^{-1} X^T Y. \)

The sample variance is \( s^2 = \frac{1}{n-p} \| X\hat{\beta} - Y \|^2 \) where \( p \) is the length of \( \beta \). The standard error for \( \beta \) is \( \text{se}(\hat{\beta}_i) = s \sqrt{[(X^T X)^{-1}]_{ii}} \).

(a) The trees data give Girth, Height and Volume measurements for 31 trees. Fit the model \( Y_i = \beta_1 + x_i^{\text{height}} \beta_2 + x_i^{\text{girth}} \beta_3 + \epsilon_i \) using the R commands

\[
> \text{data(trees)} \\
> \text{summary(lm(Volume ~ Girth + Height, data=trees))}
\]

and briefly interpret the output.

(b) Write a function of your own (using `solve()` or your solution to question 3, not `lm()`) to fit a linear model. Your function should take the length 31 vector `trees$Volume` and the \( 31 \times 3 \) matrix \( X = \text{cbind}(1, trees$Girth, trees$Height) \) as input and return estimates of \( \beta \), the residual standard error \( s \), and the standard errors of each \( \beta_i \). Check your output against the corresponding results from the `summary(lm())` output in (a).

5. Here is an algorithm to compute the QR factorisation of an \( n \times p \) matrix \( A \) with \( p \leq n \). That is, it returns an \( n \times p \) orthogonal matrix \( Q \) and a \( p \times p \) upper triangular matrix \( R \) such that \( A = QR \).

Let \( |v| \) denote the Euclidean norm of a vector \( v \). Let \( A_{[a:b]} \) denote the matrix formed from the columns \( a, a+1, \ldots, b \) of \( A \).

1. Create \( n \times p \) matrix \( Q \) and \( p \times p \) matrix \( R \).
2. Set \( Q_{[1,1]} = A_{[1,1]}/|A_{[1,1]}| \) and \( R_{11} = |A_{[1,1]}| \).
3. If \( p = 1 \) then we are done; return \( Q \) and \( R \).
4. Otherwise (i.e. if \( p > 1 \)), set \( R_{[1,2:p]} = Q_{[1,1]}^T A_{[1,2:p]} \) and \( R_{[2:p,1]} = 0 \).
5. Set \( A' = A_{[2:p]} - Q_{[1,1]} R_{[1,2:p]} \).
   \[\text{[Notice that } Q_{[1,1]} R_{[1,2:p]} \text{ is an outer product of an } n \text{ component column vector and a } (p-1) \text{ component row vector, so } A' \text{ is a new } n \times (p-1) \text{ matrix. Either make use of the } \texttt{outer()} \text{ command or, if you use } \texttt{I} \text{ be careful to use the } \texttt{drop} \text{ argument when forming these sub-matrices.]}\]
6. Compute the QR factorisation of \( A' \) (so \( A' = Q'R' \) say).
7. Set \( Q_{[2:p]} = Q' \) and \( R_{[2:p,2:p]} = R' \) and return \( Q \) and \( R \).
(a) Implement this algorithm as a recursive function in R. Your function should take as input an $n \times p$ matrix $A$ and return two matrices $Q$ and $R$ as a list. State briefly how you checked your function was correct.

(b) Using your QR function, and the R command `backsolve()`, give a least squares solution to the over-determined system

$$X \beta = Y$$

where $X$ and $Y$ take their values from the `trees` data in question 4.