A13766W1

SECOND PUBLIC EXAMINATION

Honour School of Mathematics and Statistics Part A: Paper A12

Simulation and Statistical Programming

TRINITY TERM 2019

Tuesday 25 June, 09:30–11:00

- *You may hand in answers to any number of questions.*
- *The best two answers will count towards the total mark for the paper.*
- *All questions are worth 25 marks.*
- *Begin the answer to each question in a new answer booklet.*
- *Indicate on the front page of the answer booklet which question you have answered in that booklet.*
- *Hand in your answers in numerical order.*
- *A booklet with the front cover sheet completed must be handed in even if no question has been attempted.*
- *Cross out all rough working and any working you do not want to be marked. If you have used separate answer booklets for rough work please cross through the front of each such answer booklet and attach these answer booklets at the back of your work.*

Do not turn this page until you are told that you may do so

1. (a) [13 marks] A student starts R and types in the following commands.

```
> f <- function(n=1) {
+   if (n==1) return(list(1))
+   y = f(n-1)
+   z = y[[length(y)]]
+   r = list(c(0,z)+c(z,0))
+   return(c(y,r))
+ }
> f(2)
[[1]]
[1] 1

[[2]]
[1] 1 1
> W<-f()
> X<-f(n<-4)
> Y<-f(n=5)[[5]]
> Z<-n
```

(i) Briefly explain the difference between a recursive and a loop-based implementation of an algorithm. Comment on what factors would lead you to choose one over the other for a given implementation.

> **Solution:** (3, B) In a recursive implementation the function implementing the algorithm calls itself. This generates a chain of copies of the function calling itself till the termination condition is met and the function passes back the results up the chain. In a loop based implementation the same sequence of operations in carried out in a single function environment by iteration using for example a "for" loop. Recursion is often fast but demanding of memory so this is a memory/speed trade-off.

(ii) What does the function f above do? Explain how it works.

> **Solution:** (3, S - pretend to be a computer and desk check it) The code computes Pascal's triangle returning a list with one field for each row of the table. This is done by recursing back from the input n to $n = 1$ where the answer is 1, computing the $k + 1$'th row from the current k'th row $z_1, ..., z_k$ by adding the vectors $(0, z_1, ..., z_k) + (z_1, ..., z_k, 0)$ for $k = 1, ..., n - 1$.

(iii) What are the types and values of `W`, `X`, `Y` and `Z`? Briefly justify each of your answers.

> **Solution:** (4, S) It follows from the previous answer that `W` is a list of length one with one field equal one, since there is no argument to `f`, so the default $n = 1$ is used and the function returns `list(1)`. For `X` we get the first four rows of Pascal's triangle, as fields in a list of length 4,
> `X[[1]] 1`
> `X[[2]] 1 1`
> `X[[3]] 1 2 1`
> `X[[4]] 1 3 3 1`
> and as a side-effect n will be set equal 4. For `Y` we get the fifth row of Pascal's

triangle, as a vector
`1 4 6 4 1`
and `Z` is equal four due to the earlier side-effect on n.

(iv) Re-implement the function `f()` in R using a for-loop.

Solution: (3, S/N - they havnt seen this specific example but they have turned recursive programmes into for loops)

```
g <-function(n=1) {
  #Pascal's triangle, n rows
  out<-list(z<-1) #first row, initialise list
  if (n>1) { #this test needed to get right output for n=1
    for (k in 2:n) {
      z<-c(0,z)+c(z,0) #same code as question, ie vectorised
      out[[k]]<-z
    }
  }
  return(out)
}
```

(b) [12 marks]

(i) Let p and q be two probability density functions on $\mathbb{R}$. Write down the rejection sampling algorithm targeting p with proposal q, stating any conditions imposed on p and q.

Solution: (Bookwork, 3 marks) Suppose q is such that $p/q \leqslant M$ for some $M > 0$. Then the rejection sampling algorithm is

1. Simulate $Y \sim q$ and $U \sim \text{Unif}[0, 1]$.
2. If $U \leqslant \frac{p(Y)}{Mq(Y)}$ then return Y.
3. Else goto 1.

(ii) Let $\beta_1 > 0$, $\beta_2 > 0$ and $\gamma > 0$ be parameters and for $x \in \mathbb{R}$ let

$$\widetilde{p}(x) = xe^{-\beta_1 x}\mathbf{1}_{x\geqslant 0} - xe^{\beta_2 x}\mathbf{1}_{x\leqslant 0}$$

and $\widetilde{q}(x) = e^{-\gamma|x|}$. Compute Z_p and Z_q so that $p = \widetilde{p}/Z_p$ and $q = \widetilde{q}/Z_q$ are probability densities.
Hint: the density of a Gamma(α, β) random variable is $g(x) = \frac{\beta^\alpha}{\Gamma(\alpha)}x^{\alpha-1}e^{-\beta x}, x \geqslant 0$.

Solution: (Bookwork/similar: 3 marks) We recognize the form of a $\Gamma(2, \beta_{1,2})$ respectively, so using the hint and $\Gamma(2) = 1$ we have that

$$Z_p = \int_{-\infty}^{\infty} \widetilde{p}(x)dx = \frac{1}{\beta_1^2} + \frac{1}{\beta_2^2}.$$

Likewise,

$$Z_q = \int_{-\infty}^{\infty} \widetilde{q}(x)dx = 2\int_0^{\infty} e^{-\gamma x}dx = \frac{2}{\gamma}.$$

$Z_p = \frac{1}{\beta_1^2} + \frac{1}{\beta_2^2}$ and $Z_q = \frac{2}{\gamma}$

(iii) Give a condition on γ so that we can use q as the proposal in a rejection sampling algorithm targeting p. What is the expected number of proposals needed to produce one simulated sample?

Solution: (Similar. 4 marks) We need $\gamma < \min(\beta_1, \beta_2)$ since otherwise

$$\lim_{x \to \pm\infty} \widetilde{p}/\widetilde{q}(x) = \infty$$

(take the limit in $+\infty$ or $-\infty$ according to which of β_1 or β_2 is the smallest. Suppose $\beta_1 > \beta_2$, then the max

$$\frac{d}{dx}\frac{\widetilde{p}(x)}{\widetilde{q}(x)} = e^{-(\beta_1-\gamma)x}[1 - (\beta_1 - \gamma)x]\mathbf{1}_{x \geqslant 0} - e^{-(\beta_2-\gamma)x}[1 + (\beta_2 - \gamma)x]\mathbf{1}_{x<0}$$

which cancels at $(\beta_1 - \gamma)^{-1}$ and $-(\beta_2 - \gamma)^{-1}$ both corresponds to local maxima. The largest one is attained at $(\beta_1 - \gamma)^{-1}$ and a direct calculation shows that the value of this maximum $\widetilde{M} = (\beta_1 - \gamma)^{-1}e^{-1}$. Then $M = \frac{Z_q}{Z_p}\widetilde{M} = (\beta_1 - \gamma)^{-1}e^{-1}\frac{2}{\frac{\gamma}{\beta_1^2} + \frac{\gamma}{\beta_2^2}}$ is the mean number of simulations needed.

(iv) Can we use a Gaussian proposal instead of q?

Solution: (Bookwork/similar: 2 marks) No since the Gaussian has thiner tail. So impossible to bound. To see why, the Gaussian density is proportional to $\widetilde{f}(x) = e^{-x^2/2}$ so that

$$\lim_{x \to \pm\infty} \widetilde{p}(x)/\widetilde{f}(x) = \lim_{x} e^{-\beta_{1,2}|x| + x^2/2} = \infty.$$

2. (a) [12 marks]

(i) Describe the inversion method for simulating a random variable X whose cumulative distribution function is $F(x) = P(X \leqslant x)$.

> **Solution:** (B:3) The generalized inverse of F is $F^{-1}(u) = \inf\{x : F(x) \geqslant u\}$. Let U be a Unif([0, 1]) random varaible, then $X = F^{-1}(U)$ has distribution F. This works for the continuous and discrete case.

(ii) Let M be a random variable with Geometric distribution

$$\pi(M = k) = (1-p)^{k-1}p, \quad k = 1, 2, \ldots$$

where $p \in (0, 1)$ is a parameter. Conditionally on M, let X be an integer valued random variable uniformly distributed on $\{1, \ldots, M\}$

$$P(X = x|M = m) = \frac{1}{m}\mathbf{1}_{x\in\{1,\ldots,m\}}.$$

Write a pseudo-code algorithm simulating M using the method of inversion.

> **Solution:** (S: 3 marks) Write $q = 1 - p$. M is discrete and writing F for its cdf we have that F^{-1} is the smallest x satisfying $x \geqslant \log(1-u)/\log(q)$ given by
>
> $$F^{-1}(u) = \lceil \frac{\log(1-u)}{\log q} \rceil.$$
>
> (Replacing $1 - u$ with u is admissible but not necessary).
> The pseudocode is thus:
>
> Step 1: simulate U Unif([0, 1]),
>
> Step 2: Return $x = \lceil \frac{\log(1-U)}{\log q} \rceil$
>
> Some students may think they need to do
>
> Step 1: simulate U Unif([0, 1]),
>
> Step 2:
> 1. $i = 0$, $F = 0$
> 2. While $F_i < U$ then $i = i + 1$ and $F = F + q^{i-1}p$
> 3. return i
>
> This is longer but acceptable.

(iii) Write an R function implementing the algorithm you wrote in part (a)(ii). Your function should take as input a value of p and return a single random draw from a Geometric distribution with parameter p.

> **Solution:** (6, S) It is likely some students will answer the Q above slightly differently - with a while loop. That would translate into a slightly more complicated implementation than that given below.
>
> ```
> rg<-function(p) {
>   #return a sample from Geom(p)
>   if (p<0 | p>1) stop('0<=p<=1')
>   u=runif(1)
> ```

```
  d=log(1-u)/log(1-p)
  return(ceiling(d))
}
```

(b) [13 marks] Suppose we observe a realisation $X = x^*$ of the random variable X from part (a). Consider the following rejection sampling algorithm.

Step 1 Simulate $m \sim \pi(\cdot)$ and $y \sim P(\cdot|m)$.

Step 2 If $y = x^*$ then return m. If not go to Step 1.

(i) Consider one pass through steps 1 and 2. What is the probability that m is rejected at Step 2, as a function of p and x^*? Show that, for any fixed x^*, this probability tends to one as p becomes small.
Hint: What would be the rejection probability if m was fixed?

Solution: : (New: 4 marks) Let us call ρ the rejection probability. Then if we simulate $m < x^*$ we are sure to reject. If we simulate $m \geqslant x^*$ the probability of rejecting is then $(1 - 1/m)$. Therefore, summing over all possible values of m, we have

$$\rho(p) = \sum_{n=1}^{x^*-1} q^{n-1}p + \sum_{n=x^*}^{\infty} q^{n-1}p(1 - \frac{1}{n}) = 1 - p \sum_{n=x^*}^{\infty} \frac{q^{n-1}}{n}$$

The second term is finite, $\sum_{n=x^*}^{\infty} \frac{q^{n-1}}{n} \leqslant \sum_{n=1}^{\infty} q^{n-1}$, so $\rho(p) \to 1$ as $p \to 0$.

(ii) Suppose we run the algorithm and the return value is Z. Show that Z is distributed according to the posterior distribution $\pi(\cdot|X = x^*)$.

Solution: : (N/B: 3 marks - they have to enter a bookwork proof half way through) Let us call Z the random variable which represents the value produced by this algorithm. Then, by partitioning on the numbers of trials necessary to obtain an output,

$$\begin{aligned} P(Z = z) &= \pi(z)\mathbb{P}(X = x^*|M = z)[1 + \rho + \rho^2 + \ldots] \\ &\propto \pi(z)\mathbb{P}(X = x^*|M = z) \\ &\propto P(M = z|X = x^*) \end{aligned}$$

(iii) Write an R function implementing the rejection sampler in part (b). Your function should take as input p, x^* and a positive integer N and return as output N independent samples from the posterior $\pi(\cdot|X = x^*)$. Call the function you wrote in part (a)(iii) for Step 1.

Solution: (6, S - they have written rejection algorithms in R. Also they have the algorithm given)

```
rpost<-function(p,X,N) {
  #sample posterior for M|Y
  Ms=numeric(N)
  for (i in 1:N) {
    finished=FALSE
    while (!finished) {
```

```
      m=rg(p)          #sample prior
      y=sample(1:M,1)  #sample synthetic data*
      finished=(y==X)  #stop if sim data matches real data*
    }
    Ms[i]=m             #save sample
  }
  return(Ms)
}
```

3. (a) [14 marks]

(i) Let $p(x)$ be a probability mass function on Ω with $p(x) > 0$ for all $x \in \Omega$. Write down the Metropolis-Hastings algorithm simulating a Markov chain with proposal distribution $q(x'|x)$ and invariant distribution p.

Solution: (B:5)

1. Set the initial state $Y_0 = x$ for some $x \in \Omega$
2. For $t = 1, 2, \ldots, n-1$, assume $Y_{t-1} = x_{t-1}$. Simulate $Y_t' \sim q(\cdot|x_{t-1})$ and $U_t \sim U[0,1]$.
3. If $U_t \leqslant \alpha(Y_t'|x_{t-1})$ set $Y_t = Y_t'$. Otherwise set $Y_t = x_{t-1}$.

Here $\alpha(y|x) = \min\{1, \frac{p(y)q(x|y)}{p(x)q(y|x)}\}$.

(ii) Show that the chain generated by your Metropolis-Hastings algorithm is reversible.

Solution: (B:3) It suffices to check the detailed balance condition.

$$
\begin{aligned}
p(x)P_{x,y} &= p(x)q(y|x)\alpha(y|x) \\
&= p(x)q(y|x)\min\{1, \frac{p(y)q(x|y)}{p(x)q(y|x)} \\
&= \min\{p(x)q(y|x), p(y)q(x|y)\} \\
&= p(y)q(x|y)\min\{\frac{p(x)q(y|x)}{p(y)q(x|y)}, 1\} \\
&= p(y0P_{y,x}.
\end{aligned}
$$

For $i = 1, ..., n$, let $X_i \in \{-1, 1\}$ be n random variables. Let $\Omega = \{-1, 1\}^n$. Let $X = (X_1, ..., X_n)$ follow the 1-dimensional Ising model distribution in which $P(X = x) = \pi(x)$ for $x = (x_1, \ldots, x_n) \in \Omega$ with

$$\pi(x) = \exp(\theta \textstyle\sum_{i=1}^{n-1} x_i x_{i+1})/Z_\theta$$

where Z_θ is a normalising constant.

(iii) Consider the following proposal for x' given x: pick $i \sim U\{1, ..., n\}$ uniformly at random and set $x_i' = -x_i$ and $x_j' = x_j$ for $j \neq i$ (flip the sign of the ith coordinate). Let $q(x'|x)$ be the transition probability for this proposal. Calculate the acceptance probability in the Metropolis-Hastings algorithm targeting the one-dimensional Ising probability distribution $\pi(x)$, $x \in \Omega$, given above.

Solution: (S:6 - the harder 2D case was done in bookwork) Since $q(x'|x) = q(x|x') = 1/n$ if they differ at one site, and $q(x'|x) = q(x|x') = 0$ otherwise, the acceptance probability is

$$\alpha(x'|x) = \min\left\{1, \frac{\pi(x')}{\pi(x)}\right\}.$$

The factors of Z_θ will cancel. In the exponential,

$$\theta \sum_{i=1}^{n-1} x'_i x'_{i+1} - \theta \sum_{i=1}^{n-1} x_i x_{i+1} = \theta(x_{i-1}(x'_i - x_i) + (x'_i - x_i)x_{i+1}),$$

This is $\theta(x'_i - x_i)(x_{i-1} + x_{i+1})$ unless $i = 1$ when we have $\theta(x'_1 - x_1)x_2$ or $i = n$ when it is $\theta(x'_n - x_n)x_{n-1}$. So

$$\alpha(x'|x) = \begin{cases} \min(1, \exp(\theta(x'_i - x_i)(x_{i-1} + x_{i+1})) & \text{if } 1 < i < n \\ \min(1, \exp(\theta(x'_1 - x_1)x_2) & \text{if } i = 1 \\ \min(1, \exp(\theta(x'_n - x_n)x_{n-1}) & \text{if } i = n. \end{cases}$$

(b) [11 marks] Suppose now that given $x \in \Omega$, we have $Y_i \sim N(x_i, 1)$, $i = 1, ..., n$, are independent and normally distributed with mean x_i and variance 1. Let $Y = (Y_1, ..., Y_n)$ and suppose $Y = y$ with $y \in \mathbb{R}^n$.

(i) We now consider Bayesian inference for $X|Y = y$, using the Ising model prior $\pi(x)$ given above. Assume that the Ising model parameter θ is known.
Write down the conditional density $f(y|x)$ of Y given $x \in \Omega$ and the posterior distribution $\pi(x|y)$ for $X|Y = y$.

Solution: (S:4)

$$f(y|x) = (2/pi)^{-n/2} \exp(-\sum_{i=1}^{n} (y_i - x_i)^2/2).$$

Since the posterior is proportional to the likelihood times the prior we have

$$\pi(x|y) \propto \exp\left(-\frac{\sum_{i=1}^n (y_i - x_i)^2}{2} + \theta \sum_{i=1}^{n-1} x_i x_j \right).$$

We dont need the normalising constant because it will cancel in the Hasting ratio. It would be the sum of this over $x \in \Omega$.

(ii) Give a Metropolis-Hastings Algorithm targeting the posterior $\pi(x|y)$ using the proposal transition probabilities $q(x'|x)$ defined in (a)(iii) above.

Solution: (N:7) As before q cancels in the calculation of α. The whole setup is similar to the previous case, but the likelihood now enters.

$$\alpha(x'|x) = \min\left\{1, \frac{\pi(x')f(x'|y)}{\pi(x)f(x|y)}\right\}.$$

Now $f(y|x')/f(y|x) = \exp(-(y_i - x'_i)^2/2 + (y_i - x_i)^2/2)$ and this simplifies to give $\exp(y_i(x'_i - x_i))$ so

$$\alpha(x'|x) = \begin{cases} \min(1, \exp(\theta(x'_i - x_i)(x_{i-1} + x_{i+1} + y_i)) & \text{if } 1 < i < n \\ \min(1, \exp(\theta(x'_1 - x_1)(x_2 + y_1)) & \text{if } i = 1 \\ \min(1, \exp(\theta(x'_n - x_n)(x_{n-1} + y_n)) & \text{if } i = n. \end{cases}$$

The algorithm starts with $X_0 = x_0$ (for eg $x_0 = (1, 1, ..., 1)$) then for $t = 1, 2, ..., T$

1 Choose $i \sim U\{1, ..., n\}$. Set $x' \leftarrow x$ and $x'_i \leftarrow -x_i$.
2 If $U \leqslant \alpha(x'|x)$ (with α above) accept x' and set $X_t = x'$ else $X_t = x$.