Homework #1 STA410H1F/2102H1F

due Wednesday October 4, 2017

Instructions: Solutions to problems 1 and 2 are to be submitted on Blackboard (PDF files strongly preferred). You are strongly encouraged to do problems 3 and 4 but these are not to be submitted for grading.

1. Catastrophic cancellation in the subtraction $x - y$ can occur when x and y are subject to round-off error. Specifically, if $\mathrm{fl}(x) = x(1+u)$ and $\mathrm{fl}(y) = y(1+v)$ then

$$\mathrm{fl}(x) - \mathrm{fl}(y) = x - y + (xu - yv)$$

where the absolute error $|xu - yv|$ can be very large if both x and y are large; in some cases, this error may swamp the object we are trying to compute, namely $x - y$, particularly if $|x - y|$ is relatively small compared to $|x|$ and $|y|$. For example, if we compute the sample variance using the right hand side of the identity

$$\sum_{i=1}^{n}(x_i - \bar{x})^2 = \sum_{i=1}^{n} x_i^2 - \frac{1}{n}\left(\sum_{i=1}^{n} x_i\right)^2, \tag{1}$$

a combination of round-off errors from the summations and catastrophic cancellation in the subtraction may result in the computation of a negative sample variance! (In older versions of Microsoft Excel, certain statistical calculations were prone to this unpleasant phenomenon.) In this problem, we will consider two algorithms for computing the sample variances that avoid this catastrophic cancellation. Both are "one pass" algorithms, in the sense that we only need to cycle once through the data (as is the case if we use the right hand side of (1)); to use the left hand side of (1), we need two passes since we need to first compute $\bar{x}$ before computing the sum on the left hand side of (1). In parts (a) and (b) below, define $\bar{x}_k$ be the sample mean of $x_1, \cdots, x_k$ and note that

$$\bar{x}_{k+1} = \frac{k}{k+1}\bar{x}_k + \frac{1}{k+1}x_{k+1}$$

with $\bar{x} = \bar{x}_n$.

(a) Show that $\displaystyle\sum_{i=1}^{n}(x_i - \bar{x})^2$ can be computed using the recursion

$$\sum_{i=1}^{k+1}(x_i - \bar{x}_{k+1})^2 = \sum_{i=1}^{k}(x_i - \bar{x}_k)^2 + \frac{k}{k+1}(x_{k+1} - \bar{x}_k)^2$$

for $k = 1, \cdots, n-1$. (This is known as West's algorithm.)

(b) A somewhat simpler one-pass method replaces $\bar{x}$ by some estimate x_0 and then corrects for the error in estimation. Specifically, if x_0 is an arbitrary number, show that

$$\sum_{i=1}^{n}(x_i - \bar{x})^2 = \sum_{i=1}^{n}(x_i - x_0)^2 - n(x_0 - \bar{x})^2.$$

(c) The key in using the formula in part (b) is to choose x_0 to avoid catastrophic cancellation, that is, x_0 should be close to $\bar{x}$. How might you choose x_0 (without first computing $\bar{x}$) to minimize the possibility of catastrophic cancellation? Ideally, x_0 should be calculated using $o(n)$ operations.

(An interesting paper on computational algorithms for computing the variance is "Algorithms for computing the sample variance: analysis and recommendations" by Chan, Golub, and LeVeque; this paper is available on Blackboard.)

2. Suppose that $X_1, X_2, \cdots$ is an infinite sequence of independent identically distributed random variables with some distribution function F and N is a Poisson distribution with mean λ that is independent of the sequence $\{X_i\}$. Then we can define a *compound Poisson* random variable Y by

$$Y = \sum_{i=1}^{N} X_i$$

where $Y = 0$ if $N = 0$. This distribution arises naturally in risk theory and insurance – for example, if N represents the number of claims and $X_1, X_2, \cdots$ the amounts paid for each claim then Y is the total sum paid. For the purposes of risk management, it is useful to know the distribution of Y, particularly its tail.
(a) Suppose that $\{X_i\}$ are discrete integer-valued random variables with probability generating function $\phi(s) = E(s^{X_i})$. Show that the probability generating function of Y is $g(\phi(s))$ where $g(s)$ is the probability generating function of N, which is given by

$$g(s) = \exp(-\lambda(1 - s)).$$

(b) The distribution of Y can be approximated using the Fast Fourier Transform. Assume that the random variables $\{X_i\}$ have a distribution $p(x)$ on the integers $0, 1, \cdots, \ell$. The complication is that, unlike the distribution of $S = X_1 + \cdots + X_n$, the distribution of Y is not concentrated on a finite set of integers. Therefore, we need to find an integer M such that $P(Y \geq M)$ is smaller than some pre-determined threshold ϵ; M will depend on the Poisson mean λ as well as the integer ℓ (or more precisely, the distribution $p(x)$). (Also to optimize the FFT algorithm, M should be a power of 2 although this isn't absolutely necessary unless M is very large.) Show that if $P(N \geq m) \leq \epsilon$ then $P(Y \geq m\ell) \leq \epsilon$ and so we can take $M \geq m\ell$.
(c) The bound M determined in part (b) is typically very conservative and can be decreased substantially. One approach to determining a better bound is based on the probability generating function of Y derived in part (a) and Markov's inequality. Specifically, if $s > 1$ we have

$$P(Y \geq M) = P(s^Y \geq s^M) \leq \frac{E(s^Y)}{s^M} = \frac{\exp(-\lambda(1 - \phi(s)))}{s^M}.$$

Use this fact to show that for $P(Y \geq M) < \epsilon$, we can take

$$M = \inf_{s>1} \frac{-\ln(\epsilon) - \lambda(1 - \phi(s))}{\ln(s)}.$$

(d) Given M (which depends on ϵ), the algorithm for determining the distribution of Y goes as follows:

1. Evaluate the DFT of $\{p(x) : x = 0, \cdots, M-1\}$:

$$\widehat{p}(j) = \sum_{x=0}^{M-1} \exp\left(-2\pi\iota \frac{j}{M} x\right) p(x)$$

   where $p(\ell+1) = \cdots = p(M-1) = 0$.

2. Evaluate $g(\widehat{p}(j)) = \exp(-\lambda(1-\widehat{p}(j)))$ for $j = 0, \cdots, M-1$.

3. Evaluate $P(Y = y)$ by computing the inverse FFT:

$$P(Y = y) = \frac{1}{M} \sum_{j=0}^{M-1} \exp\left(2\pi\iota \frac{y}{M} j\right) g(\widehat{p}(j))$$

Write an R function to implement this algorithm where M is determined using the method in part (c) with $\epsilon = 10^{-5}$. Use this function to evaluate the distribution of Y in the case where $\lambda = 7$ and the distribution of $\{X_i\}$ is

$$p(x) = P(X_i = x) = \frac{4 - |3-x|}{16} \quad \text{for } x = 0, \cdots, 6.$$

(You do not need to evaluate the bound M from part (c) with great precision; for example, a simple approach is to take a discrete set of points $\mathcal{S} = \{1 < s_1 < s_2 < \cdots < s_k\}$ and define

$$M = \min_{s \in \mathcal{S}} \frac{-\ln(\epsilon) - \lambda(1-\phi(s))}{\ln(s)}$$

where $\delta = s_{i+1} - s_i$ and s_k are determined graphically (that is, by plotting the appropriate function) so that you are convinced that the value of M is close to the actual infimum.

On Blackboard, there is a paper by Embrechts and Frei "Panjer recursion versus FFT for compound distributions", which (not surprisingly) compares the FFT approach to alternative approach (Panjer recursion). This paper includes some R code for the FFT algorithm described above and discusses some modifications to the basic FFT algorithm – feel free to use this R code as a template for your function. (Feel free to explore some of the other aspects of this paper for your own edification.)

Supplemental problems (not to hand in):

3. An alternative to the Quicksort algorithm is Mergesort (see p. 122 of the text). The Mergesort algorithm works by merging pairs of ordered lists. If the two lists contain r and s elements, respectively, then the number of comparisons needed to merge the two lists into single list lies between $\min(r, s)$ and $r + s - 1$.

Define $C(n)$ to be the (random) number of comparisons in the Mergesort algorithm. It can be shown that the expected number of comparisons, $E[C(n)]$ satisfies the recursive formula

$$E[C(n)] = E(Z_n) + E[C(\lfloor n/2 \rfloor)] + E[C(\lceil n/2 \rceil)]$$

where $\lfloor n/2 \rfloor = \lceil n/2 \rceil = n/2$ if n is even with $\lfloor n/2 \rfloor = (n-1)/2$, $\lceil n/2 \rceil = (n+1)/2$ if n is odd, and Z_n is a random variable whose distribution is

$$P(Z_n = k) = \frac{\binom{k-1}{\lfloor n/2 \rfloor - 1} + \binom{k-1}{\lceil n/2 \rceil - 1}}{\binom{n}{\lfloor n/2 \rfloor}} \quad \text{for } \lfloor n/2 \rfloor \le k \le n-1$$

and whose expected value is

$$E(Z_n) = \lfloor n/2 \rfloor \lceil n/2 \rceil \left(\frac{1}{\lfloor n/2 \rfloor + 1} + \frac{1}{\lceil n/2 \rceil + 1} \right)$$

(Z_n represents the number of comparisons needed to merge two ordered lists of lengths $\lfloor n/2 \rfloor$ and $\lceil n/2 \rceil$, respectively.)

(a) In class, we derived the following recursive formula for $E[C(n)]$ for the Quicksort algorithm:

$$E[C(n)] = n - 1 + \frac{2}{n} \sum_{k=1}^{n} E[C(k-1)]$$

where $E[C(0)] = E[C(1)] = 0$. Using R, compute $E[C(n)]$ for $n = 1, \cdots, 1000$ and compare the average performance of Quicksort to that of Mergesort.

(b) Calculate the worst case behaviour of $C(n)$ for Mergesort.

4. Problem 3.6 of the text. As the hint suggests, prove the result for $n = p^k$ and then show that it holds for n between p^k and p^{k+1}. Use the result to show that the worst case behaviour of $C(n)$ for Mergesort is $O(n \ln(n))$.