Demonstration of the Central Limit Theorem for sums of independent but not identically distributed random variables

Introduction

Suppose that $X_1, \cdots, X_n$ are independent random variables and $S = X_1 + \cdots + X_n$. In certain cases (for example, when $\{X_i\}$ have normal distributions or Poisson distributions), we can obtain the exact distribution of S but typically, we are forced to use approximations (normal, Poisson and others); these approximations typically require n to be reasonably large although they are often adequate for small values of n.

In this document, we will outline a method for computing exact the exact distribution of S when the summands $\{X_i\}$ take values on a finite set of non-negative integers. We can then use these exact distribution to examine the adequacy of normal approximations (motivated by the Central Limit Theorem) under various conditions.

The probability generating function

Suppose that X is a discrete random variable whose possible values are the integers $0, 1, 2, \cdots, \ell$ and define the probability mass function

$$f(x) = P(X = x) \quad \text{for } x = 0, 1, 2, \cdots, \ell.$$

Then we can define the probability generating function[1]

$$p(t) = E(t^X) = \sum_{x=0}^{\ell} f(x) t^x.$$

It is easy to see that $p(t)$ is a polynomial of order ℓ whose coefficients are the probabilities $f(0), \cdots, f(\ell)$; thus if we are given the probability generating function of a random variable (taking non-negative integer values) then we can determine the probability distribution of the random variable. In fact, if we know that $P(X \le \ell) = 1$ then it is sufficient to know the values of the probability generating function $p(t)$ at $\ell + 1$ values of t in order to recover the probabilities $f(0), \cdots, f(\ell)$.

Now suppose that we have n independent random variables $X_1, \cdots, X_n$, with probability mass functions $f_1(x), f_2(x), \cdots, f_n(x)$, whose possible values are $0, \cdots, \ell$; the probability generating function of X_i is

$$p_i(t) = \sum_{x=0}^{\ell} f_i(x) t^x.$$

[1] The probability generating function is related to the moment generating function $m(t) = E[\exp(tX)]$ of X by the relation $m(t) = p(\exp(t))$.

If $S = X_1 + \cdots + X_n$ then the possible values of S are the integers $0, 1, 2, \cdots, n\ell$ and the probability generating function of S is

$$p_{\mathsf{s}}(t) = E(t^S) = \prod_{i=1}^{n} p_i(t) = \sum_{x=0}^{n\ell} P(S = x)t^x.$$

Using the same argument as above, in order to evaluate the distribution of S, we need only evaluate its probability generating function at $n\ell + 1$ values of t and therefore we need only evaluate $p(t)$ at $n\ell + 1$ values of t. The key to efficiently computing the distribution of S lies in determining at which $n\ell + 1$ values of t we should evaluate $p_{\mathsf{s}}(t) = p_1(t) \times \cdots \times p_n(t)$.

It turns out that there is no difficulty in defining the probability generating functions $p_1(t), \cdots, p_n(t)$ and $p_{\mathsf{s}}(t)$ for complex-valued t of the form $\exp(\iota\theta) = \cos(\theta) + \iota \sin(\theta)$ (where $\iota^2 = -1$). In particular, we will evaluate $p_1(t), \cdots, p_n(t)$ at points $t_0, \cdots, t_{n\ell}$ where

$$t_j = \exp\left(-2\pi\iota \frac{j}{n\ell + 1}\right) = \cos\left(2\pi \frac{j}{n\ell + 1}\right) - \iota \sin\left(2\pi \frac{j}{n\ell + 1}\right). \tag{1}$$

The resulting sequence $p_i(t_0), \cdots, p_i(t_{n\ell})$ turns out to be the discrete Fourier transform of the probabilities $f_i(0), \cdots, f_i(n\ell)$ where $f_i(\ell + 1) = f_i(\ell + 2) = \cdots = f_i(n\ell) = 0$. After computing $p_{\mathsf{s}}(t_j) = p_1(t_j) \times \cdots \times p_n(t_j)$ for $j = 0, \cdots, n\ell$, we can recover the probability mass function of S by using computing the inverse discrete Fourier transform. The discrete Fourier transform and its inverse will be discussed in the next section.

The discrete Fourier transform

The discrete Fourier transform (DFT) of a sequence of numbers $x_0, \cdots, x_{m-1}$ is a sequence of complex numbers $\widehat{x}_0, \cdots, \widehat{x}_{m-1}$ defined by

$$\widehat{x}_j = \sum_{k=0}^{m-1} \exp\left(-2\pi\iota \frac{j}{m} k\right) x_k = \sum_{k=0}^{m-1} x_k \cos\left(2\pi \frac{j}{m} k\right) - \iota \sum_{k=0}^{m-1} x_k \sin\left(2\pi \frac{j}{m} k\right). \tag{2}$$

Given the DFT, it is possible to recover the original sequence by the inversion formula

$$x_j = \frac{1}{m} \sum_{k=0}^{m-1} \exp\left(2\pi\iota \frac{j}{m} k\right) \widehat{x}_k \quad (j = 0, \cdots, m - 1) \tag{3}$$

The DFT (and its inverse) can be computed using an algorithm called the Fast Fourier Transform (FFT), which can be remarkably computationally efficient. (It is most efficient when m is a product of small prime numbers, the best case being when $m = 2^r$.)

Assessing normal approximations using DFTs

Under various conditions on the summands $X_1, \cdots, X_n$, the distribution of $S = X_1 + \cdots + X_n$ will be approximately normal with mean $E(S)$ and variance $\mathrm{Var}(S)$. Roughly speaking, normal approximations will hold for sums of bounded random variables provided that none of the summands contributes too much of the variance of S, that is,

$$\max_{1\le i\le n}\frac{\mathrm{Var}(X_i)}{\mathrm{Var}(S)}=\max_{1\le i\le n}\frac{\mathrm{Var}(X_i)}{\mathrm{Var}(X_1)+\cdots+\mathrm{Var}(X_n)} \tag{4}$$

is not too large. (For random variables that are not bounded, more sophisticated conditions based on higher moments of X_i are needed; for example, setting $\mu_i = E(X_i)$, a normal approximation for the distribution of S will be valid if

$$\frac{E[|X_1-\mu_1|^3]+\cdots+E[|X_n-\mu_n|^3]}{\{\mathrm{Var}(S)\}^{3/2}}$$

is not too large.)

In this section, we will use the DFT to compute the exact probability distribution of $S = X_1 + \cdots + X_n$ where $X_1, \cdots, X_n$ are independent random variables (taking values $0, 1, \cdots, \ell$) with probability mass functions $f_1(x), \cdots, f_n(x)$. As noted above, the possible values of S are $0, 1, \cdots, n\ell$. The distribution of S can be determined by first computing the DFTs of the sequences of probabilities $\{f_i(0), f_i(1), \cdots, f_i(n\ell)\}$ (for $i = 1, \cdots, n$ and setting $f_i(\ell + 1) = \cdots = f_i(n\ell) = 0$) taking an n-fold product of the DFTs, and then taking the inverse DFT of this product to yield the distribution of S.

More precisely, the steps of the algorithm are as follows:

1. Compute the DFT (defined in (2)) of $f_i(0), \cdots, f_i(n\ell)$ (for $i = 1, \cdots, n$):

   $$\widehat{f}_i(j)=\sum_{k=0}^{n\ell}\exp\left(-2\pi\iota\frac{j}{n\ell+1}k\right)f_i(k)=p_i(t_j)$$

   where t_j is defined in (1).

2. Take n-fold products:

   $$\widehat{f}_{\mathsf{s}}(j)=\prod_{i=1}^{n}\widehat{f}_i(j)=p_{\mathsf{s}}(t_j)$$

3. Take the inverse DFT (defined in (3)) of $\widehat{f}_{\mathsf{s}}(0), \cdots, \widehat{f}_{\mathsf{s}}(n\ell)$:

   $$f_{\mathsf{s}}(j)=P(S=j)=\frac{1}{n\ell+1}\sum_{k=0}^{n\ell}\exp\left(2\pi\iota\frac{j}{n\ell+1}k\right)\widehat{f}_{\mathsf{s}}(k)$$

   for $j = 0, \cdots, n\ell$.

The algorithm above can be implemented quite simply in R using the R function `fft` to compute both the DFT and its inverse. The R function given below takes an $n \times (\ell + 1)$ matrix of probabilities with the values in the i-th row corresponding to the distribution of X_i. The output includes the vector of probabilities giving $f_{\mathsf{s}}(x)$ for $x = 0, 1, \cdots, n\ell$ as well as the mean and variance of S.

```
dist.sum <- function(probs) {
        k <- ncol(probs)
        n <- nrow(probs) # number of summands
        x <- c(0:(n*(k-1))) # range of the sum of n random variables
        dft <- 1
        m <- 0
        v <- 0
        for (i in 1:n) {
           p <- as.vector(probs[i,])
# now pad the vector with zeros so that it has the same length as x
           p <- c(p,rep(0,length(x)-k))
# update mean and variance
           m <- m + sum(p*x)
           v <- v + sum(p*(x-sum(x*p))^2)
# take the DFT of p and multiply by previous value of dft
           dft <- dft*fft(p)
           }
# take the inverse DFT of dft to obtain the distribution of S
        p.sum <- fft(dft,inv=T)/length(x)
# the probability distribution of the sum will be contained in
# the real components of probs.fft.inv; in theory, the imaginary
# component is exactly 0 but computationally the imaginary component
# persists due to rounding error
        p.sum <- Re(p.sum)
        output <- list(x=x,probs=p.sum,mean=m,var=v)
        output
        }
```

The output of the command `r <- dist.sum(probs)` is an object `r` consisting of four components, `r$x`, `r$probs`, `r$mean`, and `r$var`, which contain, respectively, the possible values of S, the the corresponding probabilities for the distribution of S, the mean of S, and the variance of S.

Example 1. Suppose that $X_1, \cdots, X_{20}$ are independent random variables with

$$f_i(x) = P(X_i = x) = \frac{1}{i+1} \quad \text{for } x = 0, \cdots, i$$

Note that the distributions of X_i are all uniform with means ranging from 0.5 (for $i = 1$) to 10 (for $i = 20$) and variances ranging from 0.25 to 36.67. The following R code illustrates the use of the function `dist.sum` to determine the distribution of $S = X_1 + \cdots X_{20}$:

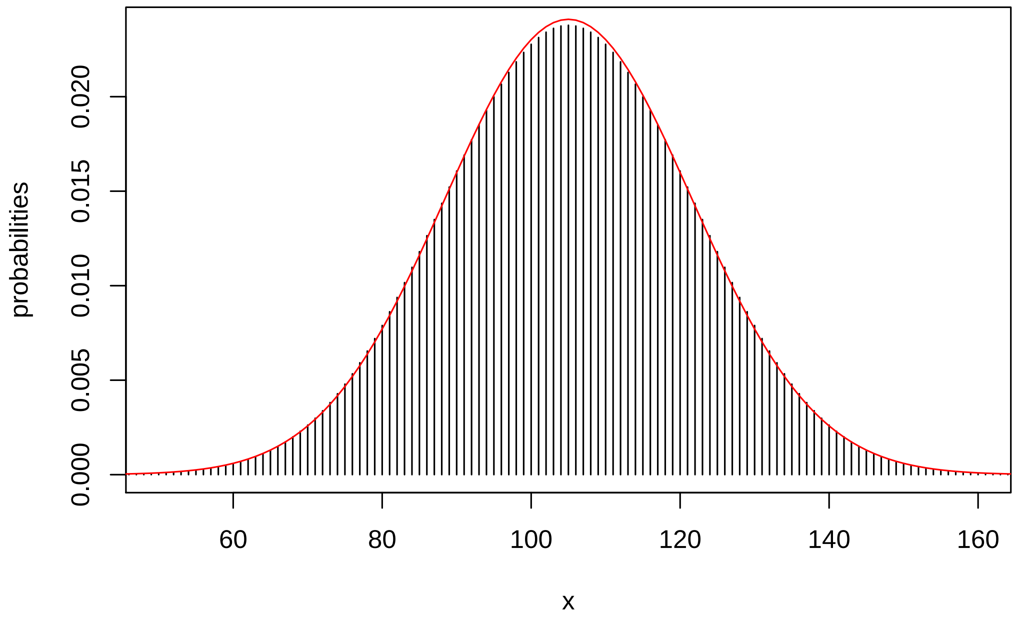

Figure 1: Exact probability distribution (lines) of S in Example 1 with its normal approximation (solid curve).

```
> # compute matrix of probabilities
>  probs <- NULL
> for (i in 1:20) {
+    probs <- rbind(probs,c(rep(1/(i+1),i+1),rep(0,20-i)))
+ }
> r <- dist.sum(probs)
> plot(r$x,r$probs,type="h",xlab="x",ylab="probabilities",xlim=c(50,160))
> lines(r$x,dnorm(r$x,r$mean,sqrt(r$var)),col="red")
```

The distribution of S with its normal approximation (with mean $E(S) = 105$ and variance $\text{Var}(S) = 274.17$) is shown in Figure 1; the normal approximation appears to be very good in this case. The key here is the fact that no summand X_i contributes too much to the variance of S; the ratio in (4) for this example is 0.13.

Example 2. Again we assume that $X_1, \cdots, X_{20}$ are independent random variables with

$$f_i(x) = P(X_i = x) = 1/2 \quad \text{for } x = i-1, i \text{ and } i = 1, \cdots, 19$$

and

$$f_{20}(x) = P(X_{20} = x) = \frac{1}{21} \quad \text{for } x = 0, 1, \cdots, 20.$$

Again we will use the function `dist.sum` to compute the distribution of $S = X_1 + \cdots + X_{20}$.

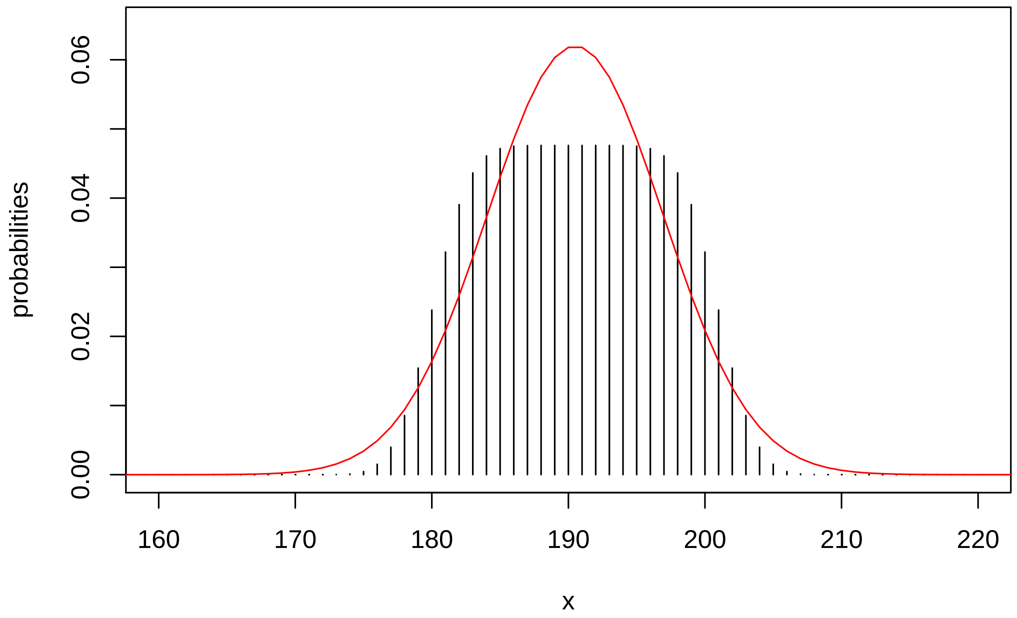

Figure 2: Exact probability distribution (lines) of S in Example 2 with its normal approximation (solid curve).

```
> # compute matrix of probabilities
>  probs <- c(1/2,1/2,rep(0,19))
> for (i in 2:19) {
+    probs <- rbind(probs,c(rep(0,i-1),1/2,1/2,rep(0,20-i)))
+ }
> probs <- rbind(probs,rep(1/21,21))
> r <- dist.sum(probs)
> plot(r$x,r$probs,type="h",xlab="x",ylab="probabilities",xlim=c(160,220),
+  ylim=c(0,0.065))
> lines(r$x,dnorm(r$x,r$mean,sqrt(r$var)),col="red")
```

Figure 2 shows the distribution of S compared to a normal density with mean $E(S) = 190.5$ and variance $\mathrm{Var}(S) = 41.42$; clearly, the normal approximation is not at all adequate here. In this particular example, the variance of S is dominated by the random variable X_{20} (whose variance is 36.67), which contributes 88.5% of the variance of S, which violates our heuristic for normal approximations based on the ratio (4).