Chapter 5 - Monte Carlo Simulation 1

1 Aims of the Chapter

By the end of this chapter you should know

- how to set the starting point of the sequence of random numbers using `set.seed`
- what Monte Carlo simulations are
- how to use Monte Carlo simulations to calculate quantities of interest
- how to quantify the uncertainty in Monte Carlo simulations
- the role of the Central Limit Theorem in Monte Carlo simulation
- what a sampling distribution is

2 What are Monte Carlo Methods?

Monte Carlo methods is the generic term for methods that involve simulation based on (pseudo) random numbers. Usually we use the random numbers to simulate realisations from a probability distribution. This is what `rnorm` does. We will just refer to this process as **simulation.** Performing simulations allows you to solve a wide range of problems. These problems fall into two classes:

1. those that are not solvable in an analytic way
2. those that potentially could be solved analytically but simulating is just much easier

In case 1 you have no choice but to resort to simulation. In case 2 you need to decide whether to use simulation or try to solve it analytically. Using simulation is an extremely powerful tool but you need to consider how many simulations to perform. Simulations can take a long time to run so performing enough to get reliable results (uncertainties you are prepared to live with) can take a prohibitive amount of time in complex simulations.

3 The `set.seed` function

Because we are going to use random numbers in our simulations we will get a different result every time we use `rnorm` or any other function that uses random numbers. This can make it difficult if you are trying to get the same result each time so that you can check your code. So

```
rnorm(1)
```

```
## [1] -0.4516211
```

```
rnorm(1)
```

```
## [1] 0.7694792
```

will give different results to

```
rnorm(1)
```

```
## [1] -1.141834
```

```
rnorm(1)
```

```
## [1] -0.9409708
```

If we want to get the same numbers every time you ask R to generate a sequence of random numbers then you need to tell it where to start the sequence (the start position is called the **seed**). You can specify the seed of the random number generator using the `set.seed` function

```
set.seed(676481256)
rnorm(1)
```

```
## [1] 1.669423
```

```
rnorm(1)
```

```
## [1] -1.96153
```

```
set.seed(676481256)
rnorm(1)
```

```
## [1] 1.669423
```

```
rnorm(1)
```

```
## [1] -1.96153
```

Generally there is no need to specify the seed but in some situations in can be helpful. For example if you are sharing your code with a colleague and want to know what numbers `rnorm` is going to generate so that you can discuss them with the colleague. If you don’t set the seed you have no way of knowing what numbers `rnorm` will simulate. I hardly ever use set.seed() but it is occasionally very useful.

4 Example 1 - calculating cumulative probabilites via simulation

Suppose a random variable X has a cumulative distribution function (CDF) $F_X(X = x)$. Suppose we want to calculate $F_X(X = 1)$:

- if there is a closed-form expression for $F_X(X = x)$ (i.e. one we can substitute a value x into) then we can just evaluate $F_X(X = 1)$ directly
- if there is no closed-form expression (as in the normal or gamma distributions) R may still have a function for calculating the probability (e.g. `pnorm` or `pgamma`).
- if neither of these is true then we need to rely entirely on simulation

How can we use simulation to estimate $F_X(X = 1)$? It should seem intuitive that the answer is to simulate realisations of the random variable X and calculate the proportion of the realisations that are ≤ 1. Statistics students taking Computational Inference will get a formal justification of this (but for the moment let us assume this has a theoretical justification).

Let's consider the normal distribution as an example of a random variable where there is no closed form CDF but where R can calculate the probability. We will check that the simulation method gives approximately the same answer as `pnorm` .

Let's consider $X \sim N(0, 1)$. To calculate $F_X(X = 1)$ we can use

```
pnorm(q = 1, mean = 0, sd = 1)
```

```
## [1] 0.8413447
```

How do we estimate this using 100 simulations?

```
normal_realisations <- rnorm(100, mean = 0, sd = 1)
(proportion_less_than_1 <- sum(normal_realisations <= 1)) / 100
```

```
## [1] 0.8
```

5 How many within-set simulations

The first question is how many within-set simulations? If we run our code using 100 within-set simulations several times we get a distribution of probabilities

```
normal_realisations <- rnorm(100, mean = 0, sd = 1)
(proportion_less_than_1 <- sum(normal_realisations <= 1)) / 100
```

```
## [1] 0.84
```

```
normal_realisations <- rnorm(100, mean = 0, sd = 1)
(proportion_less_than_1 <- sum(normal_realisations <= 1)) / 100
```

```
## [1] 0.85
```

```
normal_realisations <- rnorm(100, mean = 0, sd = 1)
(proportion_less_than_1 <- sum(normal_realisations <= 1)) / 100
```

```
## [1] 0.9
```

Let's repeat this whole process 5000 times and look at the histogram of the probability estimates we get. We'll put this all in a function so we can change the number of realisations simulated in each set.

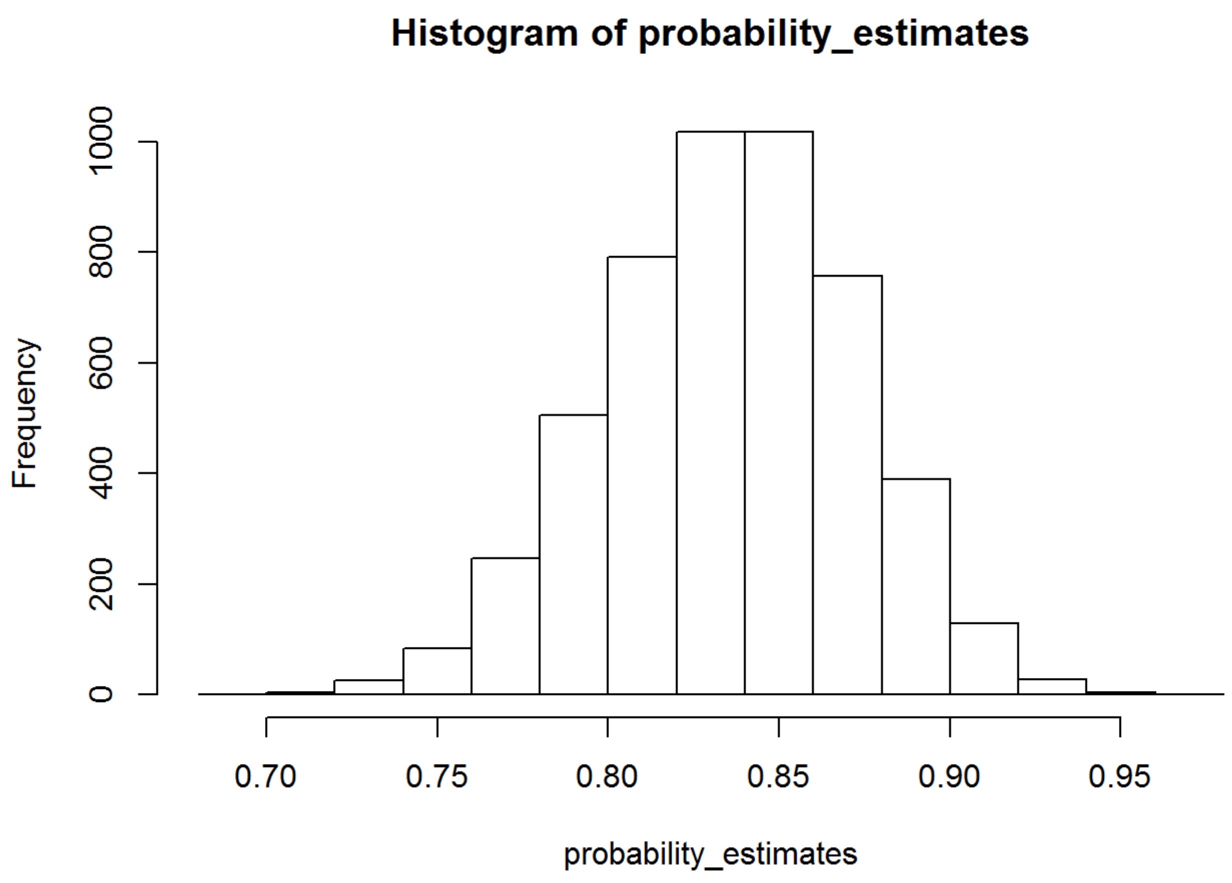
```
calculate_norm_prob <- function(num_per_set){
  normal_realisations <- rnorm(n = num_per_set, mean = 0, sd = 1)
  proportion_less_than_1 <- sum(normal_realisations <= 1) / num_per_set
}
```

We can then use the `replicate` function to repeat this process as many times as we like

```
set.seed(564564654)
probability_estimates <- replicate(n = 5000, calculate_norm_prob(num_per_set = 100))
quantile(probability_estimates, probs = c(0.025, 0.975))
```

```
##  2.5% 97.5%
##  0.77  0.91
```

```
hist(probability_estimates)
```



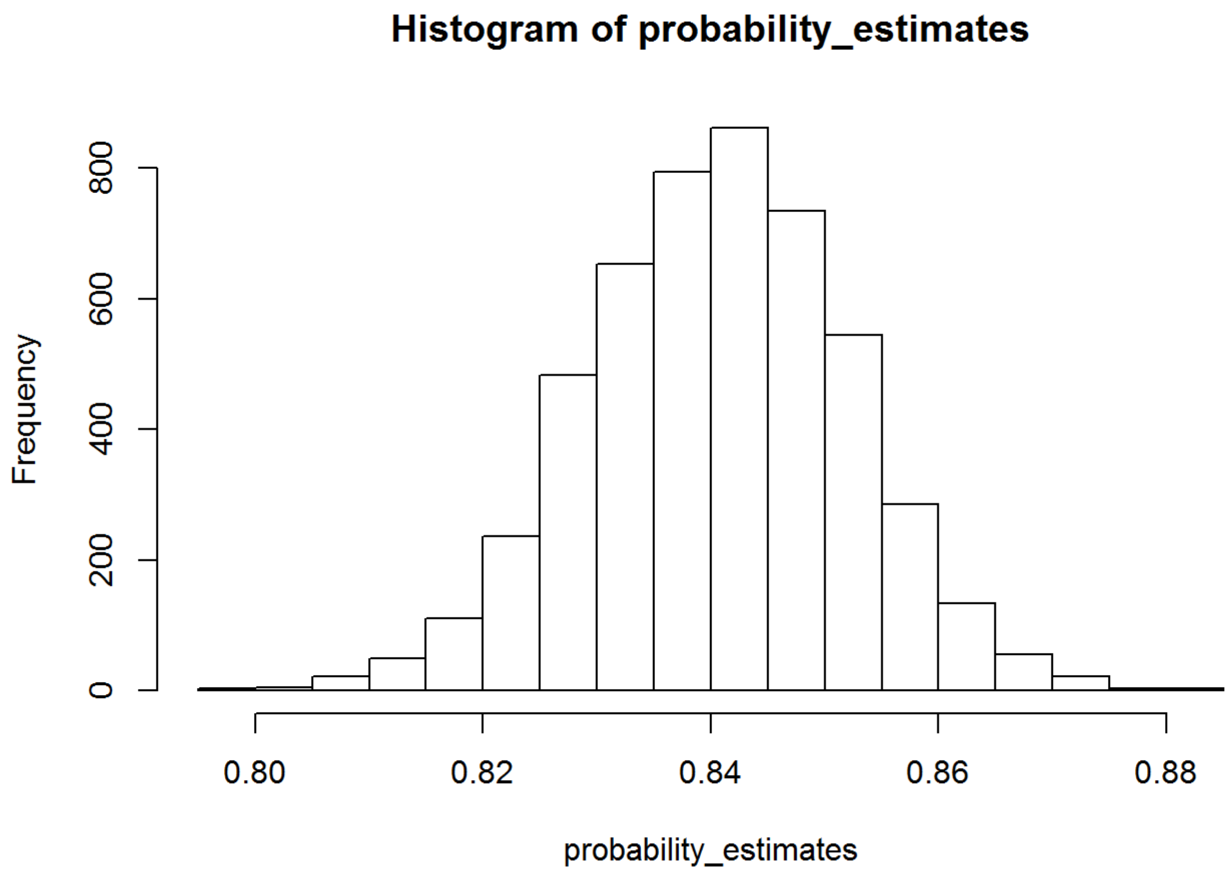
The histogram clearly shows the uncertainty in the probability estimate. We know the true value is 0.8413447 to 7 d.p. but using only 100 realisations per set we see that 95% of our estimates are in the interval (0.77, 0.91). Note that 0.77 and 0.91 don't even agree to 1 d.p.

What happens if we increase the number of within-set realisations to 1000?

```
set.seed(676346766)
probability_estimates <- replicate(n = 5000, calculate_norm_prob(num_per_set = 1000))
quantile(probability_estimates, probs = c(0.025, 0.975))
```

```
##  2.5% 97.5%
## 0.818 0.863
```

```
hist(probability_estimates)
```



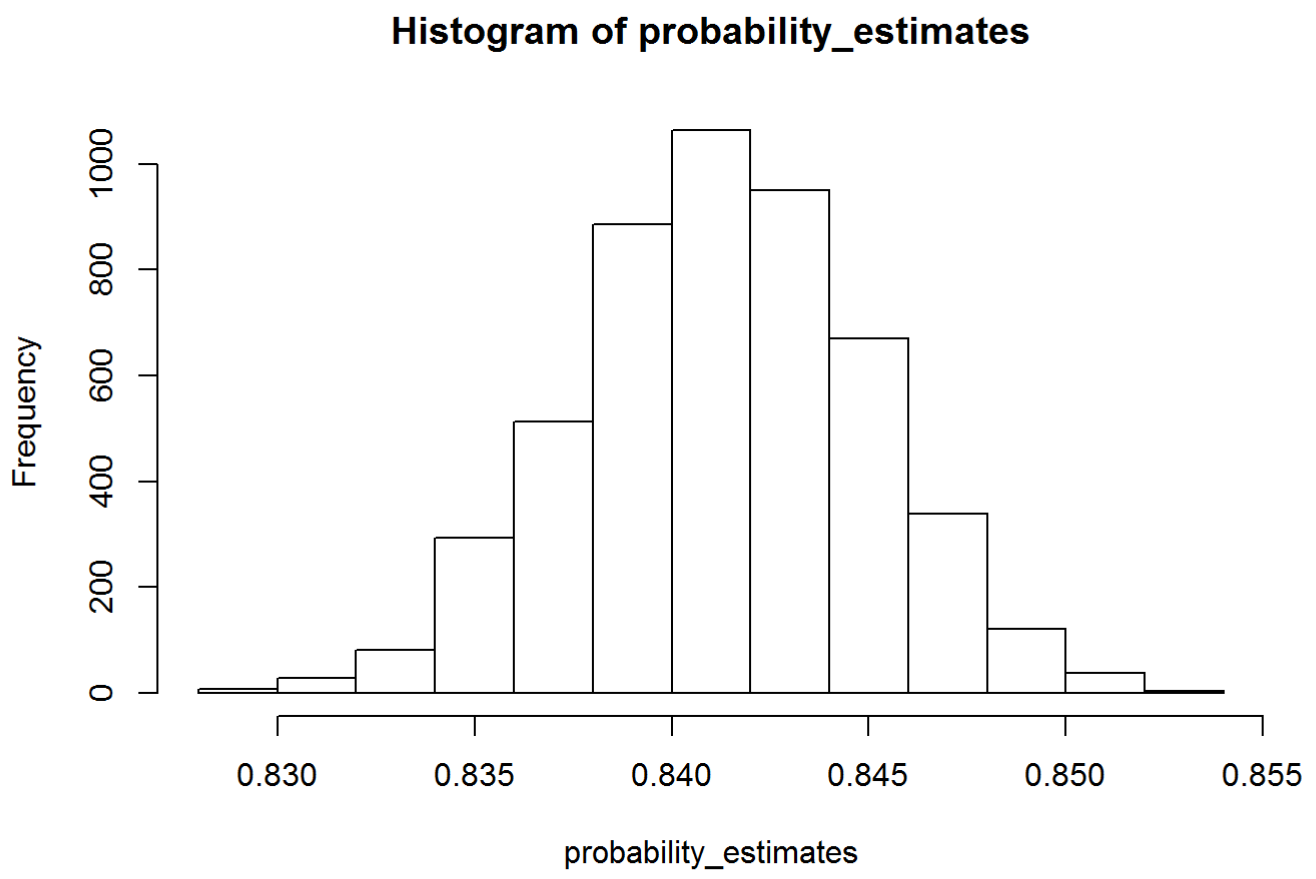
Increasing to 1000 realisations per set we see the range of estimates has decreased (there is less uncertainty). Now 95% of our estimates are in the interval (0.818, 0.863)

If we want to be more certain we can increase the number of within-set realisations further to say 10,000

```
set.seed(9894822)
probability_estimates <- replicate(n = 5000, calculate_norm_prob(num_per_set = 10000))
quantile(probability_estimates, probs = c(0.025, 0.975))
```

```
##   2.5%  97.5%
## 0.8342 0.8484
```

```
hist(probability_estimates)
```



We are now really starting to reduce the uncertainty in the probability estimates. Now 95% of our probability estimates are in the interval (0.8342, 0.8484). If want to reduce the uncertainty further we need more than 10,000 realisations per set. Note that the number of d.p. the answer is given to is completely determined by the number of realisations per set. When the number of realisations per set is 1000 the estimates are all fractions with 1000 in the denominator (and hence are given to 3 d.p.).

If we want to be fairly confident that our answer is correct to 2 d.p then we really need more realisations per set. Let's try 100,000

```
set.seed(9894822)
probability_estimates <- replicate(n = 5000, calculate_norm_prob(num_per_set = 100000
))
quantile(probability_estimates, probs = c(0.025, 0.975))
```

```
##   2.5%  97.5%
## 0.8391 0.8437
```

Now 95% of our estimates are in the interval (0.8391, 0.8437) which are both 0.84 to 2 d.p. Note that I had to use `set.seed` to guarantee these numbers. We could also ask what proportion of our estimates give 0.84 to 2 d.p.

```
sum(probability_estimates < 0.845 & probability_estimates >= 0.835 ) / 5000
```

```
## [1] 0.9984
```

So over 99.8 % of our 5000 estimates give an answer of 0.84 to 2 d.p. We can be very confident it is the correct answer (providing our code is correct !!!) to 2 d.p.

6 Exercises A

1. Suppose the random variable X has a Poisson distribution with an expected value of 4. Write a function to sample n within-set realisations of X and calculate the proportion of realisations between 1 and 5 **inclusive**.
2. Use the `replicate` function to run your function in Question 1) 1000 times with $n = 10,\ 100 \text{ and } 1000$. Plot the histogram and describe the 2.5th and 97.5th percentile of the probabilities you obtain in each case.
3. Repeat Question 2) but only run your function $m = 5$ times each for $n = 10,\ 100 \text{ and } 1000$. How useful are the results of this question compared to the results of Question 2)?

6.1 How many sets of realisations

Consider the questions in Exercises A. We were really interested in the **distribution** (or at least the range) of the estimates of the probabilities. The distribution tells us about the variability in our estimate. In question 3) we only obtained 5 sets of realisations. We really couldn't tell what the distribution was with only 5 values. In question 2) we used 1000 sets of realisations. This gives us a much better idea of the distribution.

Key points:

1. Note the distinction between:
   - n, the number of realisations **per set** used to estimate the quantity of interest
   - m, the number of **sets** of realisations used to determine the distribution of the estimates
2. When performing Monte Carlo simulations you should use a high value of m (i.e. as many **sets** of simulations as you can within a reasonable time), or at least do enough that you are confident of the distribution of the random variable of interest.

Often n is determined by the question or experiment buy you usually **do** have control over m (computational time usually being the issue)

7 Sampling distributions (simulation uncertainty)

The histograms that we have been constructing represent attempts to construct the sampling distributions of our estimator of $F_X(X = 1)$ by direct simulation, choosing a large value of m. We can sometimes use the Central Limit Theorem (CLT) to derive this sampling distribution analytically. This can save a lot of time for complex simulations and may be your only option if the code takes weeks to run. Let's look at the sampling distribution of our estimator of $F_X(X = 1)$ formally.

Let's consider again $X \sim N(0, 1)$. We estimate $F_X(X = 1)$ by simulating realisations of the random variable X and calculating the proportion of the realisations that are ≤ 1. What is the sampling distribution of this estimator? MAS6024 should not worry too much about the details here but should be able to use the result in equation (3).

- Let X_i represent the ith realisation of X.
- Then $Pr(X_i \leq 1) = F_X(X = 1) = p$ say.
- Let $Y_i = 1$ if $X_i \leq 1$ and $Y_i = 0$ otherwise.
- I.e. $Y_i \sim Ber(p)$ (Bernoulli distribution).
- $\mathbb{E}(Y_i) = p$ from properties of the Bernoulli distribution.
- $Var(Y_i) = p(1 - p)$ from properties of the Bernoulli distribution.

We are estimating $F_X(X = 1)$ using the estimator $\hat{F} = \frac{1}{n}\sum_{i=1}^{n} y_i$ where y_i is a realisation of Y_i and n is the number of within-set realisations. The Y_i are independent and identically distributed so for large n, the Central Limit Theorem gives

$$\sum_{i=1}^{n} y_i \sim N(n\mathbb{E}(Y_i), nVar(Y_i))$$
$$\hat{F} = \frac{1}{n}\sum_{i=1}^{n} y_i \sim N\left(\mathbb{E}(Y_i), \frac{Var(Y_i)}{n}\right)$$
$$\hat{F} \sim N\left(p, \frac{p(1-p)}{n}\right) \tag{7.1}$$

where $\hat{F}$ means the estimator of F. The variance in equation (7.1) confirms our previous observation that the uncertainty in the estimate of $F_X(X=1)$ decreases as the number of realisations per set (n) increases. In fact the variance of the estimate halves if you double n.

Let's check this theoretical result with what we observed with 5000 sets of 100,000 realisations. Our true value of p is 0.841, n is 100,000 so using equation (7.1)

$$\hat{F} \sim N(0.841, 1.334838e-06)$$

So a central 95% probability interval region is

$$\begin{aligned}&(0.841 - 1.96 \times \sqrt{1.334838e-06}, \quad 0.841 + 1.96 \times \sqrt{1.334838e-06})\\ =&(0.8391, \quad 0.8436)\end{aligned}$$

Which is in very close agreement with the 2.5th and 97.5th percentiles of the distribution of our estimsted probabilities (0.8391 and 0.8437 respectively)

7.1 Sampling distributions in practice

In practice, equation (7.1) is not helpful because we don't know the value of p (p is the quantity we are trying to estimate); if we knew its value we wouldn't need to estimate it. When we don't know p we can get a confidence interval for it by replacing p with its estimate ($\hat{p}$) in the variance in equation (7.1). This gives

$$\hat{F} \sim N\left(p, \frac{\hat{p}(1-\hat{p})}{n}\right) \tag{7.2}$$

We can rearrange this to give a 95% confidence interval for p as

$$\left(\hat{F} - 1.96\sqrt{\frac{\hat{p}(1-\hat{p})}{n}}, \hat{F} + 1.96\sqrt{\frac{\hat{p}(1-\hat{p})}{n}}\right) \tag{7.3}$$

8 Exercises B

1. Calculate the actual value of the probability in Question 2, Exercises A using the `ppois` function.
2. Use the result in equation (7.1) to verify the result for the 2.5th and 97.5th percentile of the estimated probabilities in Question 2, Exercises A, with $n = 1000$.
3. A box contains 8 dice

- 3 of the dice are 6 sided
- 3 are 8 sided
- 2 are 12 sided

The sides of a die with n sides are numbered 1,2, … ,n. A person is blindfolded and picks a die at random and rolls it. They replace the die and pick another at random and roll it. They continue this process until they have rolled 5 dice in total. Assume the dice rolls are independent. What is the probability the total score over the five rolls is 36 or more?

4.

- A rectangular pond has sides of length 5 and 4 metres.
- The pond is initially empty.
- 10 lily plants are simultaneously thrown into the pond in such a way that every lily is equally likely to land anywhere in the pond.
- A lily does not move once it has landed.
- Each lily grows a single circular leaf.
- If a leaf reaches the side of the pond it continues growing and the leaf hangs over the edge of the pond.
- If two leaves meet they continue growing over each other.
- After 6 months the radius of each lily leaf is independently and identically uniformly distributed between 25 and 100 cm.

Write a function to investigate the probability that after 6 months the leaf of a **randomly selected** lily does not overlap any of the others? Your function should allow the number of lilies to be a function argument with suitable default value.

5. A human resources manager in a large company has collected data on the length of time his staff take on their lunch, morning and afternoon breaks. They cannot monitor **all** the individual employees so only have data on all the employees as a whole.

Based on the data collected the manager has decided to model the time (in minutes) taken over

- lunch as a random variable with a $N(30, 100)$ distribution
- morning coffee break as exponential distribution with expected values of 20 mins
- afternoon coffee break as exponential distribution with expected values of 10 minutes

For the rest of this question assume that the times taken over the three breaks are independent.

- Are the manager's choice of probability distributions sensible?
- Is the assumption of independence realistic?
- By using Monte Carlo simulation and using the manager's choice of probability distributions above, draw a histogram of the probability distribution of the **total combined** time taken over lunch and both coffee breaks by the employees in the company.
- How you could determine quantitatively whether the number of sets of realisations you have used is enough to get a reliable distribution?

The manager wants to know what proportion of his employees that take more than 90 minutes over the three breaks. The manager thinks the proposed probability distributions for the times may not be reliable so decides instead to monitor 100 employees and individually record the total time taken over the three breaks.

Suppose there are 10,000 employees in the company and 10% of them actually take more than 90 minutes over the three breaks (but the manager doesn't know this).

- Simulate the probability distribution of the proportion of the 100 employees recorded that take more than 90 minutes.
- If the manager observes that 8 out of the 100 take more than 90 minutes, what confidence interval would they derive for the proportion of employees in the company that take more than 90 minutes, assuming they used the CLT?
- Suppose the manager observes that 2 out of 25 employees take more than 90 minutes. The CLT might **not** be valid here since 25 is rather small for an asymptotic result to be used. What confidence interval would the manager calculate if they did choose to use the CLT? Why is this interval inappropriate?

9 References