

# Non-life — Assignment NL1

Niels Keizer\* and Robert Jan Sopers†

September 15, 2016

## Generating multinormal and multi-student r.v.'s

### Q1

First we run the following script:

```
> set.seed(1)
> sum(duplicated(runif(1e6))) ## = 120
[1] 120
> sum(duplicated(rnorm(1e8))) ## = 0
[1] 0
```

The function `duplicated` returns a logical array where unique numbers are marked with 0 and duplicates are marked with 1 (the first occurrence of the number is marked with a 0). Summing this array thus gives the total number of duplicates. The uniform distribution gives 120 duplicates in a much smaller sample size than the normal distribution, which gives 0 duplicates.

In the assignment, the expected number of different numbers is derived to be

$$\mathbb{E}[N_n] = \frac{1 - f^n}{1 - f}$$

The number of duplicates is then given by  $n - \mathbb{E}[N_n]$ . We run the following script:

```
> m <- 2^32; n <- 1e6
> f <- 1 - 1/m
> num_dup_unif <- n - (1 - f^n)/(1 - f)
> num_dup_unif
[1] 116.3988
```

The expected result of 116.4 is quite close to the generated result. The outcome of 120 is therefore consistent with the assumption that `runif` produces different values uniformly.

---

\*Student number: 10910492

†Student number: 99999999

The resolution of `n_norm` is somewhere in the  $2^{50}$ 's. Directly calculating  $1 - f^n$  will give 1, because  $f$  is so close to 1. First, we use to approximation given in the assignment.

$$f^n = \left(1 - \frac{1}{m}\right)^n \approx 1 - \frac{n}{m} + \frac{n^2}{2m^2}$$

Inserting this into our equation for the number of different numbers gives

$$\frac{1 - f^n}{1 - f} \approx \frac{1 - 1 + \frac{n}{m} - \frac{n^2}{2m^2}}{1 - \left(1 - \frac{1}{m}\right)} = \frac{\frac{n}{m} - \frac{n^2}{2m^2}}{\frac{1}{m}} = n - \frac{n^2}{2m}$$

This results in the following equation for the expected number of duplicates.

$$n - \left(n - \frac{n^2}{2m}\right) = \frac{n^2}{2m}$$

Next we check in R if the number of duplicates is consistent with values for  $m$  of  $10^{15}$ ,  $10^{16}$ ,  $10^{17}$  or  $10^{18}$ , when  $n$  is  $10^8$

```
> n_norm <- 1e8
> m_norm <- c(1e15, 1e16, 1e17, 1e18)
> num_dup_norm <- n_norm^2 / (2 * m_norm)
> num_dup_norm
[1] 5.000 0.500 0.050 0.005
```

The obtained result seems to be consistent with a resolution of  $10^{16}$  or higher.

## Q2

The following code is executed in R.

```
> n <- 200; p <- 0.52
> x <- c(0, cumsum(2 * rbinom(n, 1, p) - 1))
> plot(x, type="l", lwd=1, ylab="state", xlab="step", main="1-dimensional random walk")
```

This gives the following biased random walk:

