Monte Carlo simulation of the birthday problem

The birthday problem

In class, we considered a very simple version of the birthday problem: If we have a group of n people, what is the probability that at least two share the same birthday? To compute this probability, we assumed that the n people represented a random sample from the population (or more precisely, their birthdays represent a random sample) and that birthdays are equally distributed over the 365 days of a non-leap year (so that we assume that no one is born on February 29). Under these assumptions, we can compute the probability as follows:

$$P(\text{at least one match}) = 1 - P(\text{no matches}) = 1 - \frac{364 \times 363 \times \cdots \times (366 - n)}{365^{n-1}}.$$

Using this formula, we can compute the somewhat surprising result that $P(\text{at least one match})$ exceeds 0.5 when $n \geq 23$.

If we assume unequal probabilities for birthdays, then the calculations for the probabilities of a match become much more difficult although some good approximations are available; one such approximation will be described later. It is also possible to use simulation to obtain good estimates for the probabilities of a match and we will pursue that approach in the next section.

While the birthday problem may appear to be merely a curiosity, it turns out to be similar to other more substantial problems. For example, in forensic science we may be interested in the probability that two seemingly unrelated people in a large population share the same DNA profile; in spite of the astronomical number of standard DNA profiles (which far exceeds the population of the earth), it is not uncommon for two (or more) unrelated people to share the same DNA profile. See the paper by Curran[1] for more details.

Monte Carlo simulation

Suppose that birthdays are distributed over the 366 possible days of the year with probabilities $p_1, p_2, \cdots, p_{366}$. To estimate the probability of at least one match in a group of n people, we can carry out the following "Monte Carlo"[2] experiment:

0. Set `nmatch := 0`.

1. Repeat steps (a) and (b) `nrep` times.

   (a) Draw a sample of size n (with replacement) from the set $\{1, 2, \cdots, 366\}$ using the probabilities $p_1, \cdots, p_{366}$.

[1] Curran, J. (2010) Are DNA profiles as rare as we think? Or can we trust DNA statistics? *Significance.* **7**, 62–66.

[2] This term appears to have been coined in the 1940s by John von Neumann, Stanislaw Ulam, and Nicolas Metropolis in reference to the Monte Carlo casino.

(b) If there is a match, set `nmatch := nmatch + 1`

2. P(at least one match) is estimated by `nmatch/nrep`.

The R function given on the following page implements this Monte Carlo experiment. We can actually estimate the probabilities for sizes from 2 up to n by looking at the first (say) j elements of the sample of size n. The function uses `nrep=10000` as its default value; this guarantees with a high degree of certainty that our probability estimates will be accurate to within 0.01 of the actual values. To get more accurate estimates, we can increase `nrep` at the expense of increasing the computation time; to gain an extra decimal place of accuracy, we need to increase `nrep` by a factor of 100.

```
birthday <- function(probs,n=50,nrep=10000) {
              m <- length(probs)
              index <- c(1:m)
              nmatch <- rep(0,n) # vector of n zeroes
              for (i in 1:nrep) {
# draw a sample of size n (with replacement) from the vector index
# with probabilities in probs
                x <- sample(index,size=n,replace=T,prob=probs)
                for (j in 2:n) {
                  xj <- unique(x[1:j]) # unique elements of x[1:j]
# if the length of xj is less than j there is a match for all sizes from
# j up to n and therefore we can exit the inner "for" loop
                  if (length(xj)<j) {
                      nmatch[j:n] <- nmatch[j:n]+1 # update nmatch
                      break # exit the inner "for" loop
                      }
                 }
               }
               prob.match <- nmatch/nrep # estimated probabilities
               prob.match # return the probabilities as the output
               }
```

For the purposes of illustration, we will use this function on two birthday probability distributions,

(i) $p_1 = p_2 = \cdots = p_{365} = 1/365, p_{366} = 0$ (a uniform distribution ignoring February 29),

(ii) $p_i = i/67161$ for $i = 1, \cdots, 366$ (a very non-uniform distribution),

using the following R code:

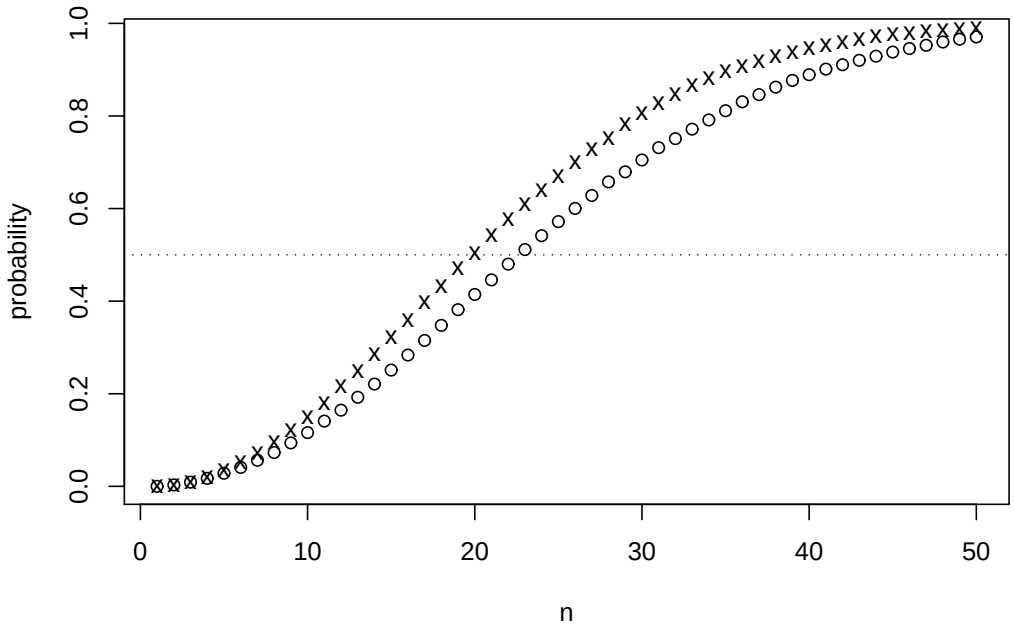

Figure 1: Estimated probabilities of at least one match for a uniform birthday distribution (o) and a non-uniform birthday distribution (x).

```
> probs0 <- rep(1/365,365) # 1/365 repeated 365 times
> r0 <- birthday(probs0,n=50,nrep=10000)
> probs1 <- c(1:366)/67161
> r1 <- birthday(probs1,n=50,nrep=10000)
> plot(c(1:50),r0,xlab="n",ylab="probability")
> points(c(1:50),r1,pch="x")
> abline(h=0.5,lty=3) # horizontal dotted line at 0.5
```

A plot of the probabilities (for the two distributions) for $n \leq 50$ is given in Figure 1. Note that the probabilities of a match are greater for the non-uniform distribution. Intuitively, this makes some – people whose birthday is more common are more likely to find a match and people with less common birthdays are less prevalent in the sample. More precisely, the probability that two randomly chosen people share the same birthday is $\sum_j p_j^2$, which is minimized when $\{p_j\}$ are all equal.

Poisson approximation

As mentioned earlier, it is possible to approximate analytically the probability that at least two people in a group of n share the same birthday in the case where the distribution of birthdays is non-uniform. One approximation is based on the fact that when

(a) n is not too large and

(b) the distribution of birthdays (that is, the probabilities $\{p_j\}$) is not too eccentric

then the number of pairs of people sharing the same birthday has a probability distribution that is approximately Poisson. (For now, this fact is not too important – we will discuss the Poisson distribution later in the term.) In particular, under the (very vague) assumptions (a) and (b),

$$P(\text{at least one match}) \approx 1 - \exp(-\lambda(n))$$

where

$$\lambda(n) = \frac{n(n-1)}{2} \sum_j p_j^2.$$

Note that $\lambda(n)$ has a very simple interpretation, which can be seen by looking at its two factors; the number of pairs in a group of n is $n(n-1)/2$ and the probability that two randomly chosen people share the same birthday is $\sum_j p_j^2$. Thus $\lambda(n)$ (and therefore $\{1 - \exp(-\lambda(n))\}$) is minimized when the $\{p_j\}$ are equal.

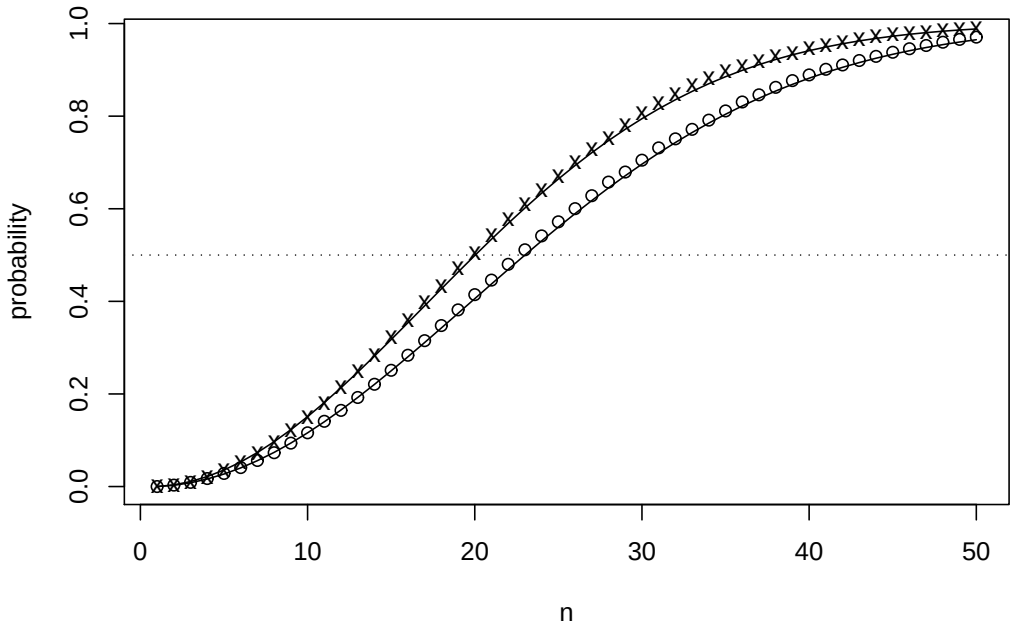

Figure 2: Estimated probabilities of at least one match for a uniform birthday distribution (o) and a non-uniform birthday distribution (x) with their respective Poisson approximations (solid lines).

Figure 2 shows the Poisson approximations for the distributions used in Figure 1; the approximations here seem to be quite adequate. The following R code was used to generate this plot:

```
> plot(c(1:50),r0,xlab="n",ylab="probability")
> points(c(1:50),r1,pch="x")
> abline(h=0.5,lty=3)
> lambda0 <- c(1:50)*c(0:49)*sum(probs0^2)/2
> lambda1 <- c(1:50)*c(0:49)*sum(probs1^2)/2
> lines(c(1:50),1-exp(-lambda0))
> lines(c(1:50),1-exp(-lambda1))
```