Seminar 2: Week 36

I. Dates for Easter

Western Easter is celebrated on the first Sunday after the Paschal full moon, which is the first full moon on or after 21 March (a fixed approximation of the March equinox). This gives a date between March 22 and April 25. There are various algorithms to calculate the day of Easter for a given year. Here we'll explore an algorithm based on the work of Gauss, but revised several times.[1]

We want to find the date for Easter in year Y. Let *mod* and *div* denote the modulus (remained of integer division) and result of integer division. We define the following variables

Variable	Expression
a	`Y mod 19`
b	`Y div 100`
c	`Y mod 100`
d	`b div 4`
e	`b mod 4`
g	`(13+8b) div 25`
h	`(15+19a+b-d-g) mod 30`
i	`c div 4`
k	`c mod 4`
l	`(32+2e+2i-h-k) mod 7`
m	`(a+11h+19l) div 433`

Now the month can be computed as `(90+h+l-7m) div 25` and the day as `(19+h+l-7m+33*month) mod 32`.

1. Write a script where you initially provide a year Y and end up with variable month and day indicating the day of Easter. Verify that you find that Easter day last year (2021) was on April 4th and on April 17th this year.
2. Transform your script into a function that gives the day of Easter as the number of days after March 21st.
3. Write a loop and find all the Easter days between year 2000 and 2100. Show the distribution of days in a histogram.
4. Optional: Run the loop over the very long run, e.g. between year 2000 and 5000 or an even longer time span, and find the long run distribution of Easter days. What does the distribution look like?

[1] See https://en.wikipedia.org/wiki/Date_of_Easter for details

II. The Fibonacci numbers

The Fibonacci numbers is a sequence of numbers F_n where $F_1 = F_2 = 1$ and $F_n = F_{n-2} + F_{n-1}$ for $n > 2$ (i.e. each number is the sum of the two preceding numbers).

1. Write a loop to compute the 100 first Fibonacci numbers
2. It is also possible to find the Fibonacci numbers recursively. Consider a function `fibo(n)`, which gives the n'th Fibonacci number. Construct it so that it returns `fibo(n-1)+fibo(n-2)` unless n<3, in which case it returns 1.
3. Compare the speed of the two algorithms. Why is the recursive algorithm so much slower?

III. The Lotka-Volterra model

The Lotka-Volterra model describes the size of the population of two species, one predator and one prey. Here we use the example of rabbits (prey) and foxes (predators). In year t there are $R(t)$ rabbits and $F(t)$ foxes. The population of rabbits increase by a factor a due to reproduction, but a fraction is eaten by foxes, depending on the number of rabbits and foxes and a factor b, yielding the equation

$$R(t+1) = (1+a)R(t) - bR(t)F(t).$$

The population of foxes decays by a factor c, but also grows with a rate depending on the amount of food (rabbits) per fox and a parameter d, yielding the equation

$$F(t+1) = (1-c)F(t) + dR(t)F(t)$$

We want to simulate this model for $T = 200$ periods using the parameters $a = 0.07, b = 0.002, c = 0.2, d = 0.0025$. We start with $R(1) = 80$ and $F(1) = 20$.

1. Start by creating two vectors R and F (empty or zeros), both with length T. Set the initial values for period 1.
2. Create a loop over $t = 2, \dots, T$ to add the new number of rabbits and foxes for each period.
3. Plot the population of rabbits and foxes over time, preferably in the same diagram.