MAS115: R programming
Lecture 3: Some more pseudo-code and Monte Carlo estimation
Lab Class: `for` and `if` statements, input

Paul Blackwell

The University of Sheffield
School of Mathematics and Statistics

Aims

Introduce programming structures:

- Loops with the `for()` command
- Controlling flow with `if` statements
- Inputting values

In the lecture we are going to talk about some more general ideas involved in programming:

- Writing some more pseudo-code
- Monte Carlo Estimation

Will give some practical examples of problems and try and create some pseudo-code for the lab class.

Recap from last time: Pseudo-Code

- Programming is about splitting big problems up into little ones
- Often useful to do this without consideration of language
- Helps to understand the flow and what you need code
- Write out steps that you are going to follow sequentially
- You can then implement each part of your code in your chosen language
- Include FOR and IF statements to allow for repetition and conditional statements

Pseudo-Code Example

Consider a vector of numbers `x = (x[1], x[2], ..., x[n])`.
What does the following pseudo-code do?

```
INPUT x;
SET N = length(x);
CREATE y to be vector of length N;

FOR (i in 1:N)
   IF (x[i] < 0)
      THEN ASSIGN y[i] = -x[i];
      ELSE ASSIGN y[i] = x[i];
   ENDIF
ENDFOR

RETURN y;
```

Pseudo-Code Task 1: Sum of cubes

Write some pseudo-code which uses a FOR loop to find the sum (for a given N) of the cubed numbers

$$\sum_{i=1}^{N} i^3$$

Pseudo-Code Task 1: Sum of cubes

Write some pseudo-code which uses a FOR loop to find the sum (for a given N) of the cubed numbers

$$\sum_{i=1}^{N} i^3$$

```
INPUT N;
SET sumcubes = 0;
FOR (i in 1:N)
  DO sumcubes = sumcubes + i^3;
ENDFOR;
RETURN sumcubes;
```

Including BREAK and NEXT in pseudo-code

- May want to either interrupt the loop or skip to the next iteration
- `break` transfers control to the statement following the loop.
- `next` transfers control to the beginning of the next iteration.

What does the following do?

```
FOR (i in 1:10) {
  PRINT i;
  IF (i = 3)
    THEN BREAK;
  ENDIF;
ENDFOR;
}
```

Pseudo-code Task 2 - Finding prime numbers

Problem:

Given a natural number $n > 3$ find out if it is prime.

Method:

Initialise a flag variable `prime <- TRUE`. Then try dividing `n` by possible factors, keep flag variable as `TRUE` unless you find a number that divides `n` in which case you should change flag to `FALSE`. Otherwise, having tried all possible numbers we know number is prime.

Hint: I used a `FOR` loop with a `BREAK` statement

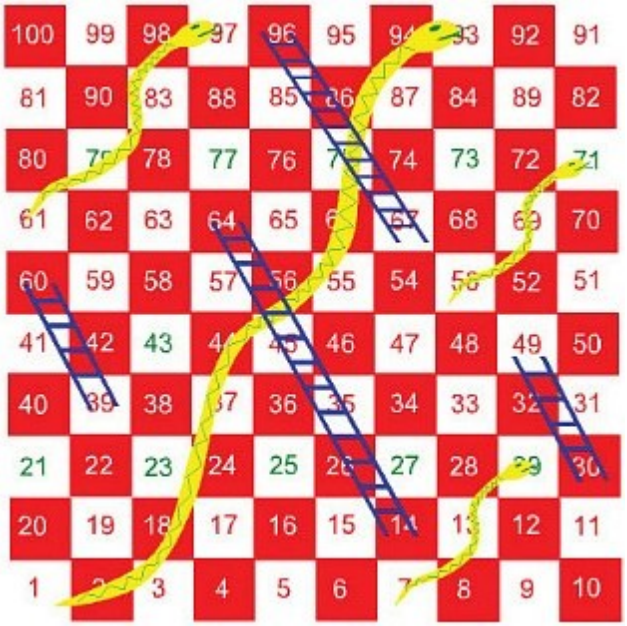
Pseudo-code Task 2 Solution

INITIALIZE variable `prime <- TRUE`
FOR (each natural number $i = 2, \ldots, \lfloor \sqrt{n} \rfloor$)
    IF n mod $i == 0$
        THEN set `prime <- FALSE`
            BREAK for loop
    ENDIF
ENDFOR
RETURN `prime`

Monte-Carlo Estimation

Question:

Suppose you are playing snakes and ladders. To finish playing you need to land exactly on the winning tile. If you roll more than the number needed to finish then you will bounce off the end tile and back however many moves you have left.



How long do you expect it will take you to finish?

Monte-Carlo Estimation

Answer:

Working this out is hard but computer simulation can help us:

- The number of rolls we need is a random variable X
- We want to know $\mu = E[X]$
- If we could play the game lots of times then we would observe $X_1, \ldots, X_n$
- Could estimate

$$\hat{\mu} = \bar{X} = \frac{1}{n} \sum_{i=1}^{n} X_i$$

- The Central Limit Theorem tells us that (under some conditions)

$$\bar{X} \to N\left(\mu, \frac{\sigma^2}{n}\right).$$

We don't want to play the game loads of times but we can get our computer to simulate games for us

Monte-Carlo Estimation: Estimating π

- Seen this idea already
- Georges throwing his toothpick onto his notepad
- Calculated $P(\text{Toothpick cross a line}) = p = \frac{1}{\pi}$
- Throw lots of hypothetical toothpicks and see how many cross

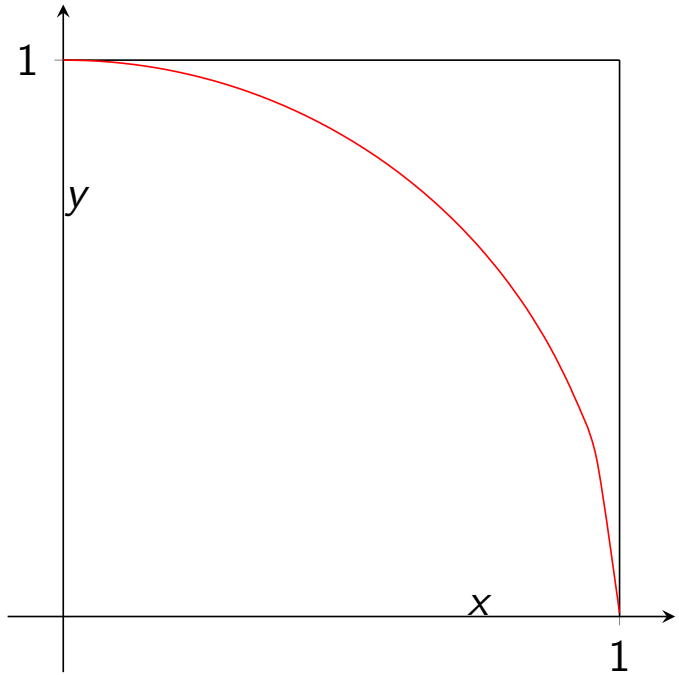
```
INPUT N;
SET cross = 0;

FOR (i in 1:N)
   SAMPLE d from UNIF[0,2];
   SAMPLE theta from UNIF[0, pi/2];
   IF (d < cos(theta))
      THEN cross = cross + 1;
   ENDIF
ENDFOR

RETURN (N/cross);
```

Estimating π again - Class Task

Consider the unit quarter-disc in the first quadrant, sitting inside the unit square



- If you threw a point uniformly at random onto square, what is probability it would lie in the disc?
- Write pseudo-code which generates N points (x, y) uniformly at random in the unit square and hence estimate π.

Solution - single loop

```
INPUT N;
SET n = 0;

FOR (i in 1:N)
    SAMPLE x from UNIF[0,1];
    SAMPLE y from UNIF[0,1];
    IF (x^2+y^2 < 1)
        THEN n = n + 1;
    ENDIF
ENDFOR

RETURN (4*n/N);
```

Solution - towards vectorization in R

```
INPUT N;
SET n = 0;

SAMPLE N VALUES x[1]...x[N] from UNIF[0,1];
SAMPLE N VALUES y[1]...y[N] from UNIF[0,1];
FOR (i in 1:N)
   SET r[i] = x[i]^2+y[i]^2
ENDFOR
FOR (i in 1:N)
   IF (r[i] < 1)
      THEN n = n + 1;
   ENDIF
ENDFOR

RETURN (4*n/N);
```

Conclusion

- How to perform loops in R to iterate statements.
- Using the `if` control statement to perform different tasks dependent upon the value of a `condition`.
- Writing pseudo-code to help you break down seemingly long and difficult questions

In the lab class today we will be learning the detail about these ideas and implementing the pseudo-code we have created.