Scientific Programming Assignment 1

MPhil in Computational Biology

October 23, 2016

If there are errors found, I will update the assignment on the web at
`http://github.com/sje30/rpc2016`

Due date: 2016-11-01 23:45

Please submit your report to the Moodle website as a single .pdf file. Name your file `spa1_XXX.pdf`, where XXX is your CRSid. (For example, I would save my file as `spa1_sje30.pdf`.).

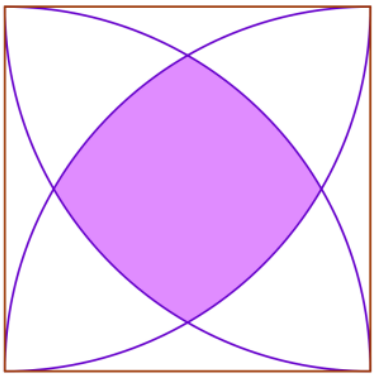
Put a copy of your code (but nothing else) in the appendix of your report. To include R code in latex, see the example code at: `http://www.damtp.cam.ac.uk/user/sje30/teaching/r/rlistings`.

Your report must be a maximum of ten pages, excluding the appendix. This course work will consist of 30% towards your overall mark for this module.

1 Curved squares [10 marks]

[Source `https://nrich.maths.org/6328`]

A square of side length 1 has an arc of radius 1 drawn from each of its corners, as in the following diagram. The arcs intersect inside the square at four points, to create the shaded region:



(a) Draw the above plot as closely as you can (colours do not need to match) in R using base graphics. [5 marks]

(b) What is the area of the largest square that can be completely contained within the shaded region? Draw the square on top of the plot. [2 marks]

(c) Estimate the area of the shaded region using Monte-Carlo method (e.g. `http://mathfaculty.fullerton.edu/mathews/n2003/montecarlopimod.html`). How close is your answer to the analytical solution? [3 marks]

2 Cryptarithms [10 marks]

[See `http://www.logicville.com/cryptarithm.htm` for background reading. Note that leading digits in each number will not be zero.]

(a) Write a function that solves the following maths problem: [5 marks]

```
 A B
 * C
 ---
 D E
+F G
 ---
 H I
```

where the symbols A–I correspond to distinct digits, 0-9. Show all solutions.

(b) Write a function that takes one argument, a string, and returns all corresponding solutions. Demonstrate it working on the following three cases: [5 marks]

- `"send + more = money"`
- `"snow + rain = sleet"`
- **(Advanced:)** `"one + two + two + three + three = eleven"`

To solve this problem you might find the following function useful.

```
## ---- permutations
## Taken from: http://stackoverflow.com/questions/11095992
permutations <- function(n){
    if(n==1){
        return(matrix(1))
```

```
    } else {
        sp <- permutations(n-1)
        p <- nrow(sp)
        A <- matrix(nrow=n*p,ncol=n)
        for(i in 1:n){
            A[(i-1)*p+1:p,] <- cbind(i,sp+(sp>=i))
        }
        return(A)
    }
}
```

For the advanced section, you might find the following tip useful:

```
eval(parse(text="rnorm(3)"))

## [1] -0.4411564  0.7935214  0.3105203
```

3 Word processing [10 marks]

Read in the file from the following location:
`http://www.damtp.cam.ac.uk/user/sje30/teaching/r/usr-share-dict-words` This is a dictionary that comes with Ubuntu linux. Each line contains one word.

1. How many words are in the database? [1 mark]
2. How many words contain non-ASCII characters? (Remove these words from the rest of the analysis.) [1 mark]
3. List the words that do not contain a vowel. How many are there? [1 mark]
4. List all the palindromes in the dictionary. [2 marks]
5. How many words contain an apostrophe (')? Remove these words from the rest of the analysis. [1 mark]
6. Estimate the probability distribution of the twenty six letters (treat upper and lower case as the same) of the alphabet using this database. [2 marks]
7. Compute the bigram probablility matrix for this database, assuming that each word is preceded and followed by one space. (For example, the word "dog" would generate four bigrams: " d", "do", "og", "g "). Draw your bigram matrix in a similar way to Figure 2.2 from David Mackay's [excellent!] textbook: `http://www.inference.phy.cam.ac.uk/itprnn/book.pdf`, using area of each square to represent frequency.

   Which is the most frequent bigram, and with what probability? [2 marks]

Hints: remember that *scan()* and friends can read data directly from a web page. Read the R help page for *regexp* and *grep()*. On unix, the command *man ascii* will list all ASCII characters.