TD09_solutions

December 9, 2014

1 Travaux pratiques Python - statistiques et graphiques

```
In [1]: import os ## Operating system library

        ## Get home directory
        home_dir = os.path.expanduser('~')
        print("Home directory = " + home_dir)

        ## Specify the data directory
        course_dir = os.path.join(home_dir, "jgb53d-bd-prog")
        cds_dir = os.path.join(course_dir, "data", "gene_tables")
        print("Data directory = " + cds_dir)

        ## List all files in the data directory
        data_files = os.listdir(cds_dir)
        print(data_files)

        ## Specify the full path of the organism-specific CDS files
        ecoli_cds_file = os.path.join(cds_dir,  "Escherichia_coli_K_12_substr__MG1655_uid57779_cds.tab")
        print("E.coli CDS file : " + ecoli_cds_file)
        plasmodium_cds_file = os.path.join(cds_dir,  "Plasmodium_falciparum_cds.tab")
        print("P.falciparum CDS file: " + plasmodium_cds_file)
```

```
Home directory = /Users/jvanheld
Data directory = /Users/jvanheld/jgb53d-bd-prog/data/gene_tables
['Escherichia_coli_K_12_substr__MG1655_uid57779_cds.tab', 'Plasmodium_falciparum_cds.tab']
E.coli CDS file : /Users/jvanheld/jgb53d-bd-prog/data/gene_tables/Escherichia_coli_K_12_substr__MG1655_uid
P.falciparum CDS file: /Users/jvanheld/jgb53d-bd-prog/data/gene_tables/Plasmodium_falciparum_cds.tab
```

1.1 Lecture des coordonnées à partir d'un fichier tabulaire

Ecrivez une méthode qui prend en entrée un nom de fichier et un numéro de colonne, lit le contenu de ce fichier et retourne une liste contenant les valeurs(supposées entières) de la colonne indiquée.

Note: La première ligne du fichier ne contient pas de données, il s'agit d'une ligne d'en-tête, qui indique le contenu de chaque colonne. Vous pouvez simplement passercette ligne.

```
In [2]: def read_column(file_name, column):
            """
            Extract the content of a specified column from a tab-delimited file.

            Attributes:
                file_name -- name of a tab-delimited text file
                column -- number of the column (first column is 1)
```

```
    Returns:
        values -- a list of integer values parsed from the specified column of the input file
    """
    ## Check that the column attribute is an integer (column number)
    if (type(column) != int):
        raise Exception("read_column() error: invalid column number: should be an integer")

    ## Check that the column attribute is at least 1
    if (column < 1):
        raise Exception("read_column() error: invalid column number: should be >= 1")

    ## Check that file name is not an empty string
    if (file_name == ""):
        raise Exception ("read_column() error: file name cannot be empty")

    ## Open the file in read mode
    file = open(file_name, "r")

    l = 0 ## Initialize a line counter
    values = [] ## Create a list to store the values
    column_header = "" ## Header of the selected column
    for line in file:
        l += 1
        fields = line.split(sep="\t") ## Split the line by tab
        if (l == 1):
            ## Report the column content from the header line (for information)
            column_header = fields[column-1]
        else: ## parse all subsequent lines
            value = int(fields[column-1])
            values.append(value)
    print("Extracted " + str(len(values)) + " values from column " + str(column) + "\t" + colum
    return(values)
```

Utilisez cette méthode pour extraire les positions de début et de fin des CDS d'Escherichia coli, que vous stockerez dans des listes dénommées start_positions et end_positions, respectivement.

```
In [3]: start_positions = read_column(ecoli_cds_file, 2)
        end_positions = read_column(ecoli_cds_file, 3)
```

```
Extracted 4319 values from column 2        start_pos
Extracted 4319 values from column 3        end_pos
```

1.2 Statistics on gene lengths

```
In [4]: cds_lengths = [] ## Initialize and empty list to store gene lengths
        for i in range(len(start_positions)):
            cds_lengths.append(end_positions[i] - start_positions[i] +1)

        ## Check the 5 first gene length values
        print (cds_lengths[:5])

        ## Compute the min and max values step by step
        n = len(cds_lengths) ## Number of CDS is the length of the list
        min_len = cds_lengths[0] ## Initialize the minimal length with the first gene
```

```
        max_len = cds_lengths[0] ## Initialize the maximal length with the first gene
        sum_len = 0
        mean_len = 0
        for len in cds_lengths:
            if len > max_len: ## Update max if required
                max_len = len
            if len < min_len: ## Update min if required
                min_len = len
            sum_len += len ## Update length sum
        mean_len = sum_len / n ## Compute the mean

        ## Print the result
        print ("Number of CDS\t" + str(n))
        print ("Min length\t" + str(min_len))
        print ("Max length\t" + str(max_len))
        print ("Mean length\t" + str(mean_len))
```

```
[66, 2463, 933, 1287, 297]
Number of CDS        4319
Min length        45
Max length        21837
Mean length        956.6899745311415
```

```
In [5]: import statistics
```

```
        ## Print the result
        print ("Number of CDS\t" + str(n))
        print ("Min length\t" + str(min(cds_lengths)))
        print ("Max length\t" + str(max(cds_lengths)))
        print ("Mean length\t" + str(statistics.mean(cds_lengths)))
        print ("Median length\t" + str(statistics.median(cds_lengths)))
```

```
Number of CDS        4319
Min length        45
Max length        21837
Mean length        956.6899745311415
Median length        837
```

```
In [5]:
```

```
In [5]:
```

```
In []:
```