Babraham Bioinformatics

Exercises: Introduction to R

Version 0.13

Licence

Exercise 1: Simple Calculations

- Use R to calculate the following:
  - 31 * 78
  - 697 / 41
- Assign the value of 39 to `x`
- Assign the value of 22 to `y`
- Make `z` the value of `x - y`
- Display the value of `z` in the console
- Calculate the square root of 2345, and perform a log2 transformation on the result.

Exercise 2: Working with Vectors

- Create a vector called `vec1` containing the numbers 2,5,8,12 and 16
- Use x:y notation to make a second vector called `vec2` containing the numbers 5 to 9
- Subtract `vec2` from `vec1` and look at the result
- Use `seq()` to make a vector of 100 values starting at 2 and increasing by 3 each time
- Extract the values at positions 5,10,15 and 20 in the vector of 100 values you made
- Extract the values at positions 10 to 30 in the vector of 100 values you made

Exercise 3: Lists and Data Frames

- Enter the following into a vector with the name `'mouse.colour'`. Remember to surround each piece of text with quotes.

  purple
  red
  yellow
  brown
- Display the 2nd element in the vector (red) in the console.
- Enter the following into a vector with the name `‘mouse.weight’`:

  23
  21
  18
  26
- Join the 2 vectors together to make a data frame named `mouse.info` with 2 columns and 4 rows.
- Display the data frame in the console.
- Display just row 3 in the console
- Display just column 1 in the console
- Display the item of data in row 4, column 1.

Exercise 4: Reading in data from a file

Set your working directory to where the data files are stored. Make sure that the folder of data files has been unzipped. e.g. `setwd("D:/Data_folder")`

4a

- Read the file 'small_file.txt' into a new data structure. This is a tab delimited file so you should use `read.delim()`.
- View the data set to check that it has imported correctly.

4b

- Read the file 'Child_Variants.csv' into a new data structure. This is a comma separated file so you should use `read.csv()`.
- View the data set to check that it has imported correctly.
- Display row 11.
- Calculate the `mean` of the column named `MutantReadPercent.`

Exercise 5: Filtering

5a

- Create a filtered version of the small_file dataset which only includes rows where the length is < 65. Save this with a new name.
- Create a further filtered dataset from the output of the previous point only including rows where the Category is 'B' (you'll need to do an exact text match using two equals signs). Save this with a new name.

5b

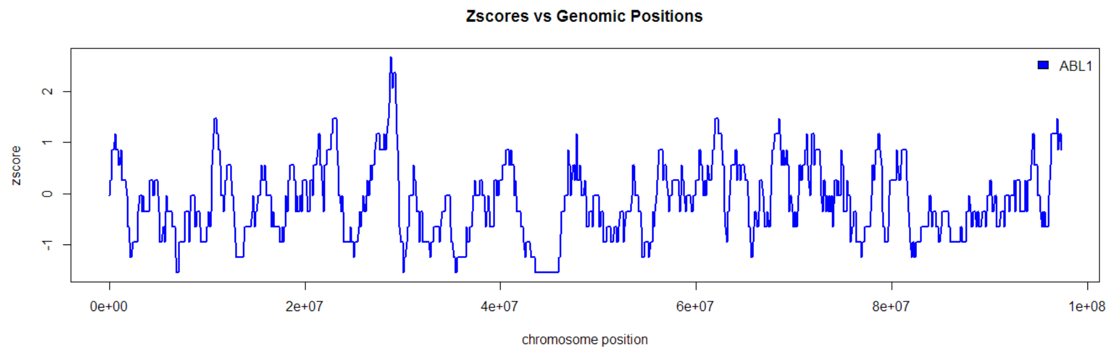
- Create a filtered version of the child variants dataset which only includes rows where the `MutantReadPercent` is >=70. Save this under a new name.
- Create a further filtered dataset from the output of the previous point for mutations where the `REF` column is 'C' (you'll need to do an exact text match using two equals signs). Save this under a new name.
- From the last filtered list calculate the number of lines for mutations to T, G, and A and see if there is a preference for one of these mutations. You will need to use an exact match filter against the text in the `ALT` column. To get the count you can simply `sum()` the boolean array values (true counts as 1, false counts as 0). Create a vector called `'mutation.counts'` of the counts for the different C mutation frequencies.

Exercise 6: Histograms, Boxplots and Barplots

- In the original child variants dataset draw a histogram of the `MutantReadPercent` column, try increasing the number of categories (breaks) to 50.
- Plot a boxplot of the `MutantReadPercent` values from both the original child variants and the same column from the filtered dataset you made in Exercise 5 (`MutantReadPercent` >=70). Check that the distributions look different.
- Plot the results of the vector created in Exercise 5 (`'mutation.counts'`) as a `barplot`. Use the `names.arg` function to show which mutation is which and use the `rainbow()` function to give the bars different colours.

[Optional] Exercise 7: Scatterplots and Line graphs

- Read in the file `'neutrophils.csv'`. This is a comma-delimited file so use `read.csv()`.
- Create a `boxplot` of the 4 samples and put a suitable title on the plot (you should be able to pass the entire data frame to `boxplot` since you're plotting all of the data there).
- Use the `colMeans()` function to calculate the mean of each dataset. Note: the data contains missing values (NA) so look at the help file for `colMeans` to find out how to ignore these. Plot the means as a `barplot`.
- Read in the file `'brain_bodyweight.txt'`. This is a tab delimited file so you can use `read.delim()`. The first column contains species names, not data, so use `row.names=1` to set these up correctly in your data frame.
- Log transform the data (base 2).
- Create a scatterplot with default parameters with the log transformed data.
- Create the same scatterplot but experiment with some parameters. Have a look through at the plot help page by using `?plot`. More parameters can be found by using `?par`.
- Read in the file `'chr_data.txt'`.
- Remove the first column (you can simply select the second and third columns and then overwrite the original variable)
- Create a line graph like the one below. The process will be:
  - Use `plot()` to plot the genome position vs the ABL1 dataset. You will need to set the `type` parameter to `l` (lower case L) to get this to be a line graph.
  - Use `legend` to add in a legend to the plot, and `title` to add a title to the plot (you could also do this by adding a main parameter to the original plot function)


Zscores vs Genomic Positions
ABL1
zscore
0e+00
2e+07
4e+07
6e+07
8e+07
1e+08
chromosome position