R basics and transcriptome data analysis

The aim of this methods course will be a discrete introduction to the analysis of transcriptome data using an in-house developed analysis tool based on the programming language R. Besides, this course should also give you the possibility to learn the basics of R programming being able to use it also for other tasks during your studies. Results of the downstream analysis may vary by using different software, internet databases, or due to your emphasis. There will be a consultation hour on Monday, Wednesday, and Friday between 12:00-12:30 when you can ask specific questions.

https://uni-bonn.zoom-x.de/j/67310913960?pwd=bWlialZOR2tHd3BZNXBYWW1RL3ZqZz09

During this course, you will use different databases and tools. **Please inform yourself about the databases and the basic idea of the tools.**

Course materials and folder for protocol upload can be found on sciebo:

https://uni-bonn.sciebo.de/s/MpMjvyBFJUtBUIn

Good luck!

1. **R basics**
   a. Go to the **swirl** website: https://swirlstats.com/, which we will use to learn the basics in R programming. **Take your time with it**.
      i. Go to: https://swirlstats.com/students.html and follow the instructions to install **R**, **RStudio,** and how to use **swirl**.
      ii. Start the course “R Programming”.
      iii. Do lessons 1-10 and 15.

2. **Application of R:**
   a. Use RStudio to solve the following tasks:
      i. Calculate the sum of “4,2,5,7,9”?
      ii. Assign the word “bioinformatics” to the variable z.
      iii. Generate a vector with 5 random numbers, name it “myvector” and assign a name to each vector element.
      iv. Visualize “myvector” using a barplot.
      v. Visualize “myvector” additionally using the ggplot2 library by 3 different plots.
   b. Answer the R-questions: simple tasks (3rd page)
   c. Read in the table “data.txt”.
      i. How many columns and rows does it have?
      ii. What is shown in columns and rows?
      iii. Visualize the columns of “data.txt” by boxplots.
   d. Perform R-assigments: advanced tasks (4th page)

3. **Guided bulk transcriptome analysis:**
   Perform a bulk transcriptome analysis using own written pipeline.

   1. **Load Data**
      a. Import data.txt (raw counts) and annotation.txt (metadata).
      b. Ensure data integrity and explore the structure.
   2. **Quality Control (QC)**
      a. Create boxplots for raw counts to visualize distribution per sample.
      b. Perform PCA on raw counts to assess global variance and detect outliers.
   3. **Normalization**
      a. Normalize raw counts using TMM (edgeR).
      b. Evaluate normalization with boxplots and PCA of normalized data.
   4. **Differential Expression Analysis**
      a. Identify differentially expressed genes using a t-test for IFNg vs baseline and IL4 vs baseline.
      b. Separate results into upregulated and downregulated genes.
   5. **Visualization**
      a. Generate heatmaps of significant genes identified from the t-test results.
   6. **GO Enrichment**
      a. Perform GO enrichment analysis for upregulated and downregulated genes.
      b. Visualize results with bar plots of enriched biological processes.

   7. **<u>Bonus 1: Linear Model Analysis</u>**
      a. Identify significant genes using a linear model for a vs untreated and b vs untreated.
      b. Visualize results with heatmaps and include enriched GO terms.
   8. **<u>Bonus 2: Batch Correction</u>**
      a. Perform batch correction (e.g., using sva or limma) to remove batch effects.
      b. Re-evaluate differential expression analysis and compare results before and after correction.

4. **Advanced analysis of the transcriptome data results:**
   a. **Visualize the gene structure of one of your chosen DE genes from the previous taks and determine the transcript variability.**
   The extensive NCBI database or another genetic online-atlas should give you all necessary information, like genomic context and regions, as well as a number of transcripts and products.

   b. **Perform one gene ontology enrichment analysis (GOEA) of the differential expressed genes from the previous taks.**
      1. Use g:profiler and get familiar with the output it
      2. Use an another alternative to perform a GOEA

Simple tasks:

1. We have a vector x:

   ```
   x <- c(4,6,5,7,10,9,4,15)
   ```

   What happens and why: x < 7

2. We have two vectors p and q:

   ```
   p <- c (3, 5, 6, 8)
   q <- c (3, 3, 3)
   ```

   What happens and why: p+q

3. We have the following vectors:

   ```
   Age <- c(22, 25, 18, 20)
   Name <- c("James", "Mathew", "Olivia", "Stella")
   Gender <- c("M", "M", "F", "F")
   ```

   What is the R-code for getting the following output:

   ```
   ##   Age   Name Gender
   ## 1  22  James      M
   ## 2  25 Mathew      M
   ```

4. We have the vector x: sum(is.na(data_name))

   ```
   x <- c(34, 56, 55, 87, NA, 4, 77, NA, 21, NA, 39)
   ```

   Which R-statement will count the number of NA values in x?

5. Use the seq() function to generate the sequence 9,18,27,36,45 and assign it to the vector z

6. Create a vector with values 1,2,3,4 using 3 different ways

x[!is.na(x) & x > 0], so it is not NA, and greater than 0

Advanced tasks:

1. For Loop
    a. Create a for loop that iterates from 1 to 10 and prints the current iteration number.
    b. Use an if statement inside the for loop to check if the current iteration number is even or odd. If the number is even, print "even" and if the number is odd, print "odd".
2. While loop
    a. Create a while loop that starts at 0 and adds 2 to the current iteration number until the sum is greater than 20. Print the current iteration number at each step of the loop.
3. lapply
    a. Use lapply() to square each number of a **list** of 4 numbers.

4. sapply
    a. Use sapply() to square each number of a **vector** of 4 numbers.
    b. Use sapply() to square each number of a **previous list** of 4 numbers.

5. Function
    a. Create a function called math_operation that takes two arguments: a number and a string.
    b. Inside the function, use the switch() function to check the string input and perform a mathematical operation on the number. If the string is "add", add 10 to the number, if the string is "subtract", subtract 5 from the number, if the string is "multiply", multiply the number by 2, and if the string is "divide", divide the number by 3.
    c. Return the result of the operation