Code Challenges for applied bioinformatics

Forschungszentrum Jülich, IBG-4

General guidelines

Coding language is English, that is variables and documentation must be in English. All code must be documented. Each challenge must have a concise and comprehensive README file giving clear installation and execution instructions. All code must be delivered executable. So please provide a command line call to invoke your solution and verify the result. Assume a *nix operating system. Code delivered without a command line call will *not* be considered a solution of the challenge. If not specified explicitly, you may choose any toolset and programming language of your preference as long as they are freely available. You must at least solve two of the three challenges and challenge two must be among them. Please solve the challenges yourself and provide only your original code.

Challenge One - Basic algorithms

Assume a directed acyclic graph (DAG) in which nodes can have multiple parents and there is a single unique root node. The *minimum* number of edges connecting any node *n* with the DAG's root node *r* is considered the depth *d*(*n*) of node *n*. Because a node can have many parents, there might be several routes, of varying number of edges, from the node to the root *r*. Provide a test (see Test Driven Development) that executes your solution and verifies it works as expected. Then write the function that returns the depth of a node given the DAG and test it with your test. Assume the DAG, i.e. the input to your function, to be represented as an adjacency matrix. Further input arguments are the node of interest, for which we want to know its depth, and the root node, meaning that the root node is known so that your algorithm does not need to find it. Use any programming language you like to solve this challenge. You will receive extra points if you name at least two of the three standard algorithms that solve this challenge - or the other one(s) apart from the algorithm you applied in your solution.
Alternatively, you can receive extra points if you implement the solution in either Brainfuck or Assembler, but because you will be considered not totally sane we will not invite you to dinner.

Challenge Two - Applied bioinformatics

A defining analysis in bioinformatics is searching for similar sequences for a set of query nucleotide or amino acid sequences e.g. using Blast. Reciprocal Best Hits (RBH) are a common proxy for orthology in comparative genomics (see this article). Essentially, a RBH is found when the proteins encoded by two genes, each in a different genome, find each other as the best

scoring match in the other genome. NCBI's BLAST is the software most usually used for the sequence comparisons necessary to find RBHs.
Download any two Nostoc bacterial genomes from NCBI, use one complete genome (the “reference”) and of the other only a small fraction e.g. one hundred genes (the “subject”). Write a simple pipeline that identifies the RBH for the “subject” genes. You can use Blast, Blat, Diamond or any other sequence similarity search tool.
You will receive extra points if you use shell commands like `cut`, `sort`, `awk`, and `comm` to identify the RBH from the results of the sequence similarity searches.

Challenge Three - Data visualization and interpretation

Genetic diversity within species is often structured geographically, reflecting both current ecological and historical drivers of genetic differentiation. Thus population structure, and admixture among populations, should be accounted for when searching for functional diversity within species. In this link you can find the Q-output and sample metadata of an Admixture (a popular software to infer population structure) analysis, performed on maize and teosinte samples. In the table stored in file “`maize.3.Q`”, each row represents a sample, in the same order as in the also provided “`maize_samples.txt`” metadata file. Considering only the maize samples, plot the Q estimates and the altitude at which each sample was collected. You can make the plot and order the samples however you deem appropriate. Besides the code and plot, provide a short biological interpretation (max 50 words) of the admixture analysis and plotted results.

Enjoy your code challenges and thank you very much for your interest, time, and effort!