SOFTWARE REQUIREMENTS SPECIFICATIONS

FOR

OH MY GENES

Prepared by:

GROUP NAME: STARTREK

ASHLY TAFADZWA DHANI - 201632120150

HOZZA TAMIM MUSTAPHA - 201632120160

TUTOR: HUI LAN

ZHEJIANG NORMAL UNIVERSITY

INTRODUCTION

This document is a software requirement specification(SRS) which describes the Little Hill Lab's initial requirements for an online application (Oh My Genes) which allows our scientists to upload gene expression files and quickly get differentially expressed genes.

PURPOSE AND PERSPECTIVE

The aim of this document is identifying differentially expressed genes given a gene expression file containing two cell samples. Therefor Intended beneficiaries for this project are biologists in the Lab (potentially worldwide). This SRS is basis for agreement between scientists and biologists about the product to be developed. Through this document, the workload needed for development, validation and verification will ease. To be specific, this document is going to describe functionality, external interfaces, performance, attributes and the design constraints of the system which is going to be developed.

SCOPE AND DOMAIN KNOWLEDGE

This project is intended for making use of today's popular technology Web technologies to create web application that will be used by scientists to identify differentially expressed genes. Currently, there are lots of online Apps serving the same purpose, both open-source and commercial, in the market. Usually they provide lots of extensive features to biologists to ease the process of uploading gene expression files to get differentially expressed genes in an efficient way. Specifically, the online application will involve; Control sample which is a cell sample prepared in its normal condition.
Also, the treatment sample which is a cell sample treated by special chemicals, or in which some genes are altered. Then also the differentially expressed genes which is the genes which have significantly different expression levels between two samples. Then also we have up-regulation which is a gene which is said to be up-regulated if it has higher expression in treatment than in control.
logFC is the log fold change of gene expression. log_2 [T/C], where T is the gene expression level from a treatment sample, while C is the gene expression level from a control sample.

FUNCTIONALITY

To protect the privacy of users, the data submitted by one user is not visible to any other user. The front-end operates within dynamically loaded web pages arranged in a step-wise fashion. It permits viewing on different devices through the reactivity of the Flask framework. The application is eventually intended for the biologists. It will be deployed to web site and all users of the product will access by use of the Flask Framework. Website will have a simple user interface where users can operate all the provided functionality by upload and go. However, this site will be only a part of a larger system. There will be cloud server where all the user data is kept and all the execution is done. Website will only be the interface for the user data and the execution of provided functionalities. All the data however will be retrievable through the flask database however Because database connections encapsulate a transaction, you will need to make sure that only one request at a time uses the connection through the given account of the given user.

OVERVIEW

The web application has a simple interface with a single button [Upload and GO]. Our scientists upload a plain text file containing gene expression levels from two samples, representing two experimental conditions. Accepting the file, the software will return a table of differentially expressed genes and a scatter plot of these genes whose X-axis is control and Y-axis is treatment. If an invalid gene expression is given, the web application returns a page informing the user to provide the correct format.
We are going to focus on describing the system in terms of product perspective, product functions, user characteristics, assumptions and dependencies on the following sections of this document. Next, we will address specific requirements of the system, which will enclose external interface requirements, requirements of the system, performance requirements, and other requirements.

Input

A valid submitted gene expression file will be having the following format. It is a TAB-delimited, plain text file with three columns as below. The file contains an optional head line, followed by each gene's expression in a control sample (e.g., ControlSample) and in a treatment sample (e.g., KnockOutSample).

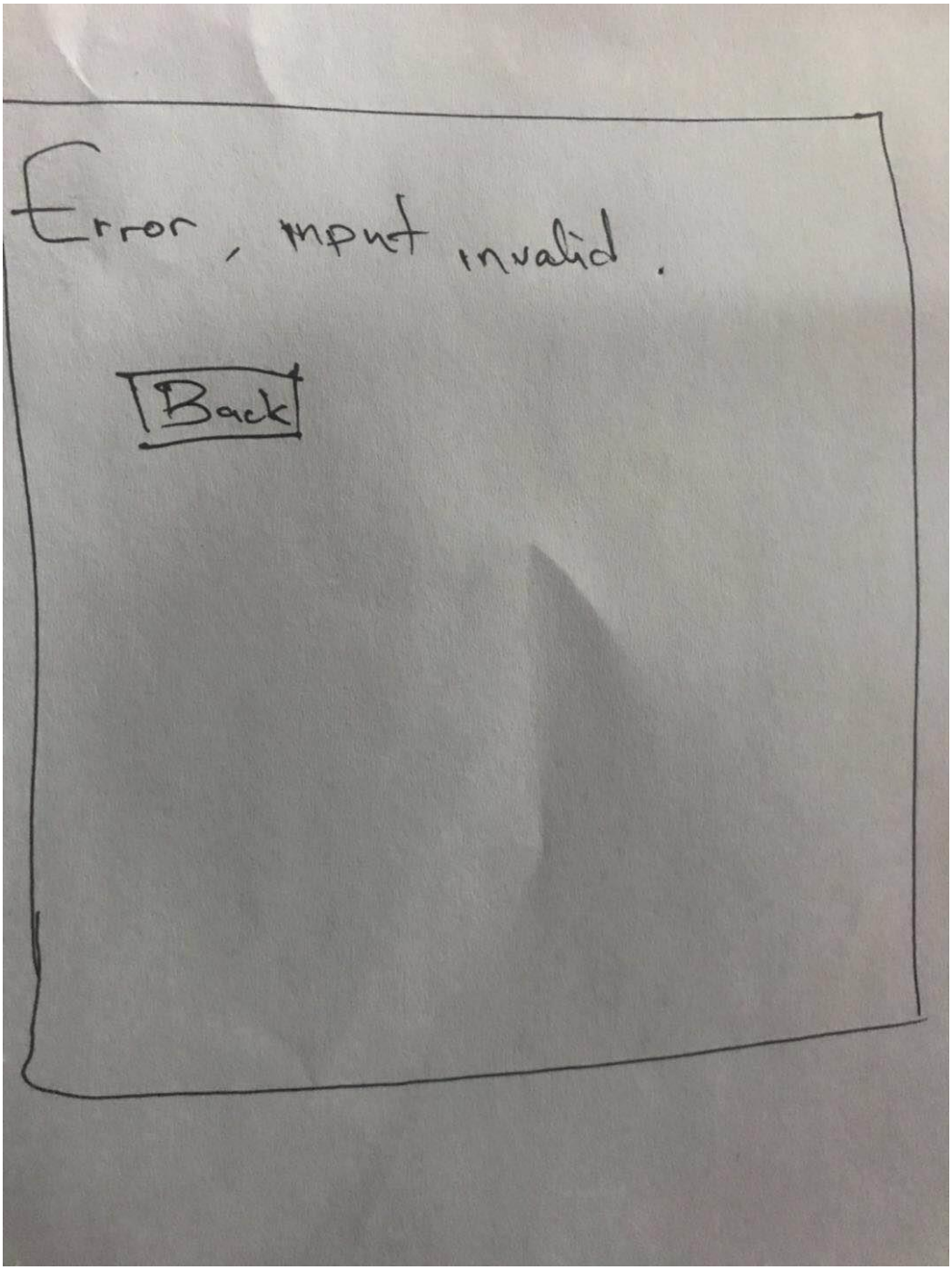
gene_id	ControlSample	KnockOutSample
AT1G01010	1.198558083	2.036161827
AT1G01020	13.75736234	13.370796
AT1G01030	0.833779536	0.203616183
AT1G01040	9.58846466	7.126566394
AT1G01046	0	0
AT1G01050	23.81482799	21.10821094
AT1G01060	0.625334652	1.221697096
AT1G01070	1.719670292	0.950208853
AT1G01080	28.34850421	25.24840665
AT1G01090	58.26034505	42.96301455
AT1G01100	1066.508249	1308.030358
AT1G01110	2.709783491	1.425313279

OUTPUT

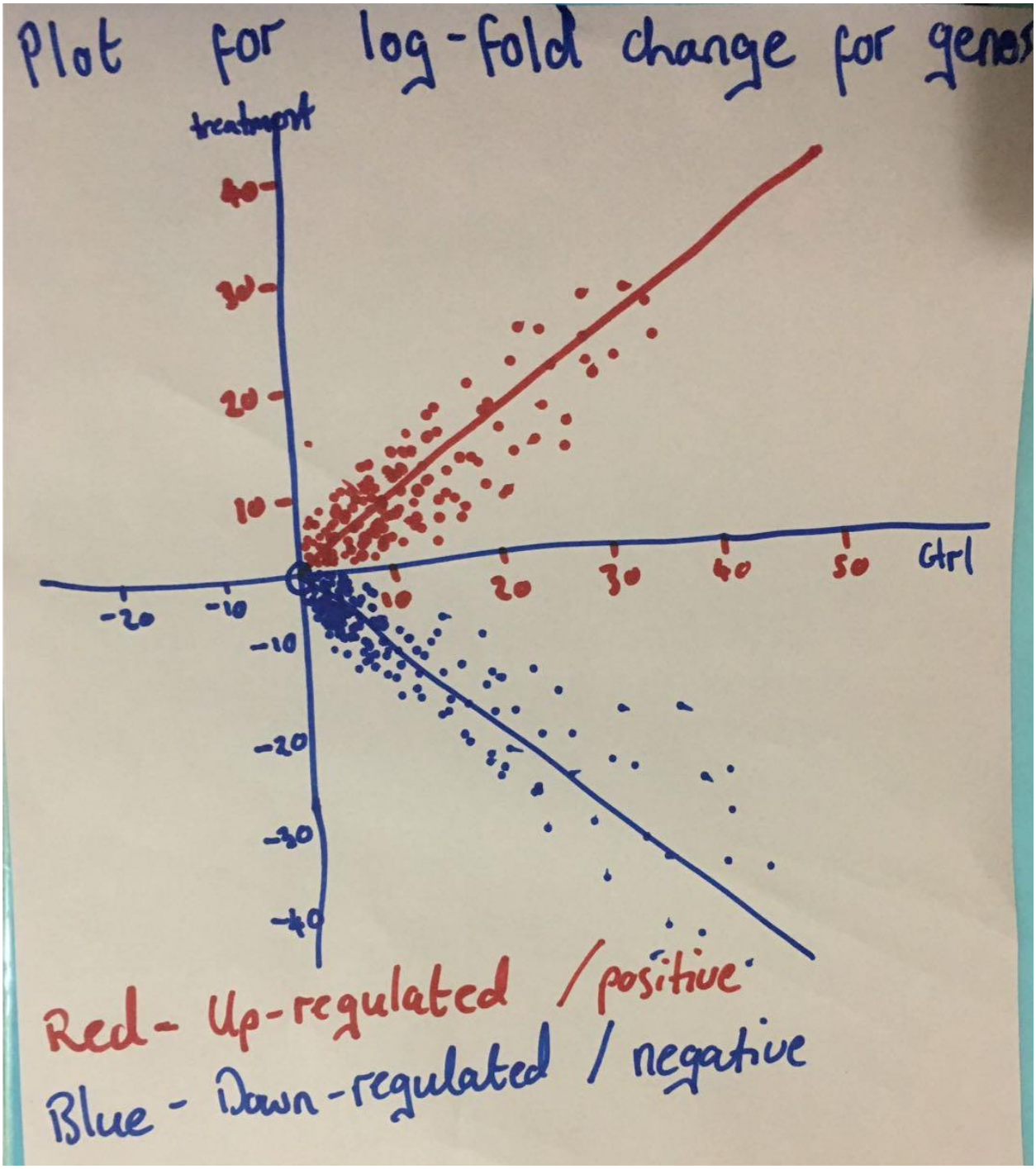
The web application displays a table and a scatter plot given a gene expression file. The table contains a list of differentially expressed genes with the following format. This output is only available if the input format is correct according to the designed requirements of the flask framework else it will bring an error. The Log_2[FC] is calculated by $\log_2$ of control sample divided by treatment sample. If Both the control and Treatment are Zero then the LOG FC is infinity.

gene_id	ControlSample	TREAT SAMPLE	LOG_2[FC]
AT1G01010	1.198558083	2.036161827	0.76
AT1G01020	13.75736234	13.370796	-0.04
AT1G01030	0.833779536	0.203616183	-2.03
AT1G01040	9.58846466	7.126566394	-0.42
AT1G01046	0	0	-infinity
AT1G01050	23.81482799	21.10821094	-0.17
AT1G01060	0.625334652	1.221697096	0.97
AT1G01070	1.719670292	0.950208853	-0.86
AT1G01080	28.34850421	25.24840665	-0.17
AT1G01090	58.26034505	42.96301455	-0.44
AT1G01100	1066.508249	1308.030358	0.29
AT1G01110	2.709783491	1.425313279	-0.93

However the system is designed to bring an error message that will inform the user that the input placed is invalid and thus also include a button that will allow the user to go back and either fix the particular error or rewrite the input all together.



Plot for LOGFC ON GENE EXPRESSION



Response Time

To find out the response time we need to test the application. In order to test the application, we add a second module (flaskr_tests.py) and create a unittest skeleton. The code in the **setUp()** method creates a new test client and initializes a new database. This function is called before each individual test function is run. To delete the database after the test, we close the file and remove it from the filesystem in the **tearDown()** method. Additionally during setup the TESTING config flag is activated. What it does is disable the error catching during request handling so that you get better error reports when performing test requests against the application. This test client will give us a simple interface to the application. We can trigger test requests to the application, and the client will also keep track of cookies for us. This application because it doesn't involve too much URL therefore the response time is less than 5 seconds. Sample Response;

```
HTTP/1.1 200 OK


{
   "num_results": 8,
   "total_pages": 3,
   "page": 2,
   "objects": [{"id": 1, "name": "Jeffrey", "age": 24}, ...]
 }
```

Complications

`glmTreat` implements a test for differential expression relative to a minimum required fold-change threshold. Instead of testing for genes which have log-fold-changes different from zero, it tests whether the log2-fold-change is greater than `lfc` in absolute value. `glmTreat` is analogous to the TREAT approach developed by McCarthy and Smyth (2009) for microarrays.

`glmTreat` detects whether `glmfit` was produced by `glmFit` or `glmQLFit`. In the former case, it conducts a modified likelihood ratio test (LRT) against the fold-change threshold. In the latter case, it conducts a quasi-likelihood (QL) F-test against the threshold.

If `lfc=0`, then `glmTreat` is equivalent to `glmLRT` or `glmQLFTest`, depending on whether likelihood or quasi-likelihood is being used.

If there is no shrinkage on log-fold-changes, i.e., fitting glms with `prior.count=0`, then `unshrunk.logFC` and `logFC` are essentially the same. Hence they are merged into one column of `logFC` in `table`. Note that `glmTreat` constructs test statistics using `unshrunk.logFC` rather than `logFC`.

`glmTreat` with positive `lfc` gives larger p-values than would be obtained with `lfc=0`.
If `null="worst.case"`, then `glmTreat` conducts a test closely analogous to the `treat` function in the limma package. This conducts a test if which the null hypothesis puts the true logFC on the boundary of the `[-lfc,lfc]` interval closest to the observed logFC.
If `null="interval"`, then the null hypotheses assumes an interval of possible values for the true logFC. This approach is somewhat less conservative. In subsequent results, a positive log $_2$-fold-change (logFC) will indicate a gene up-regulated in lactating results of tests whereas a negative logFC will indicate a gene more highly expressed in the test. The value of the FC threshold can be varied depending on the dataset. In the presence of a huge number of DE genes, a relatively large FC threshold may be appropriate to narrow down the search to genes of interest. In the absence of DE genes, on the other hand, a small or even no FC threshold shall be used. If the threshold level is set to zero, then `glmTreat` becomes equivalent to `glmQLFTest` in the workflow shown here. In general, using `glmTreat` to reduce the number of DE genes is better than simply reducing the FDR cutoff, because `glmTreat` prioritizes genes with larger changes that are likely to be more biologically significant. `glmTreat` can also be used with *edgeR* pipelines other than quasi-likelihood, although we don't demonstrate that here.

Tentative timeline

User Requirements Document: 3 April 2018

Requirements Q&A and elicitation: 10 April 2018

Software Requirements Specification: 17 April 2018

Acceptance test: 1 June 2018

Deployment date: 15 June 2018