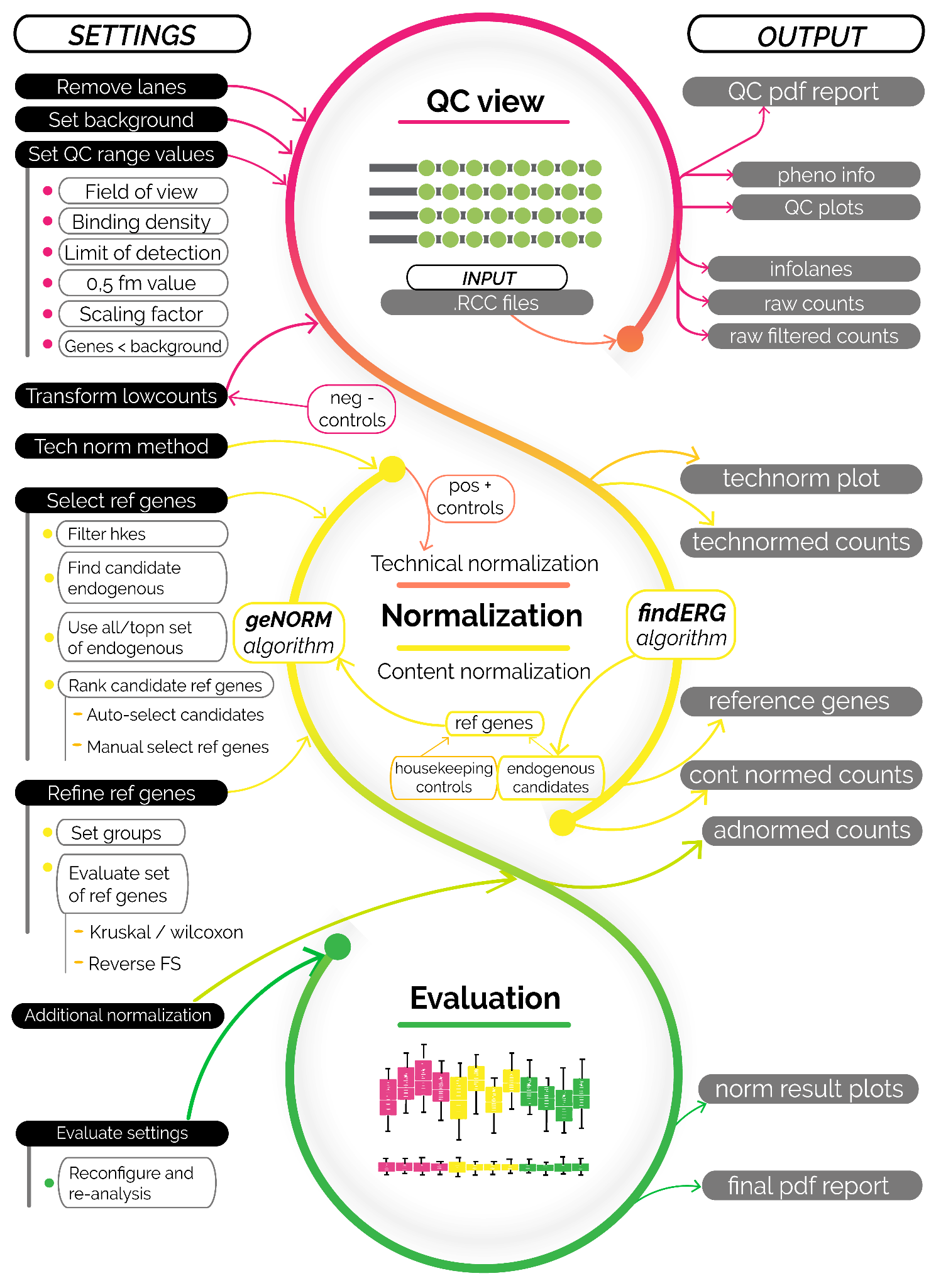
USER GUIDE



Please cite:

[Tal cual esto lo otro, info de la publicación]



WORKFLOW DESCRIPTION

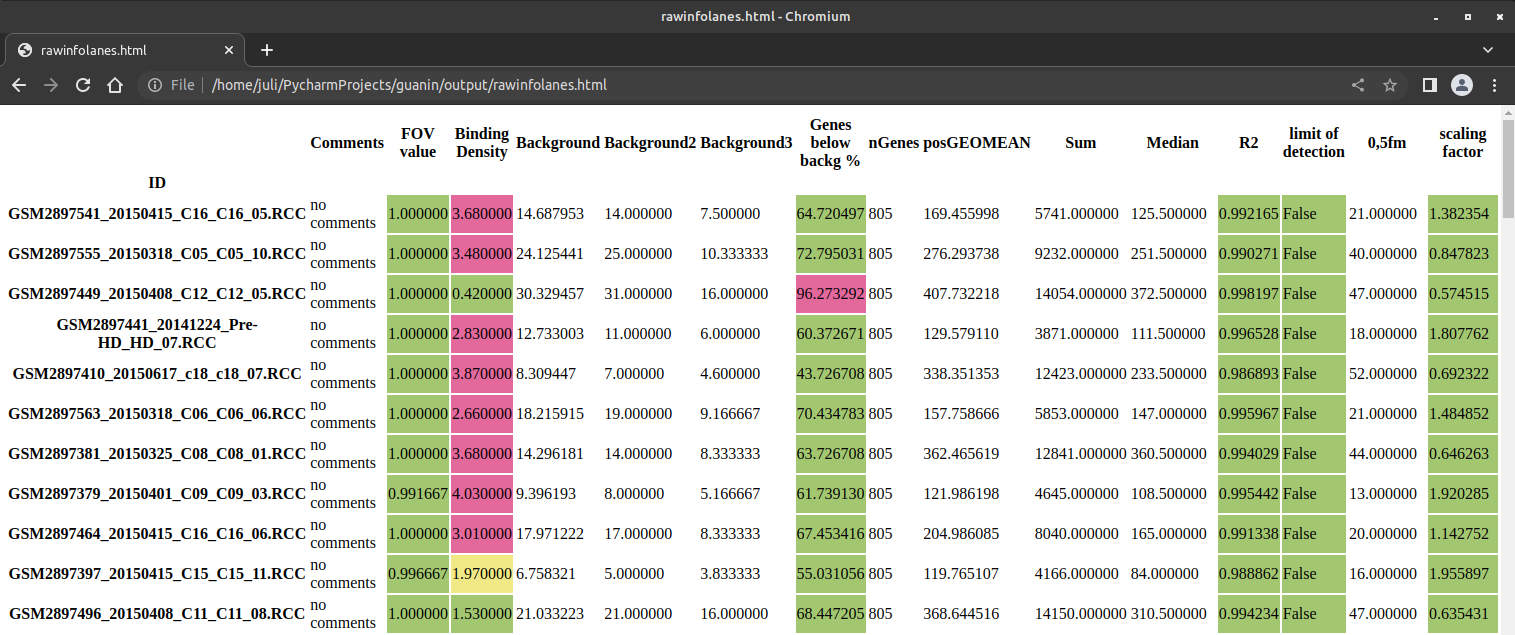
**LOADING DATA…**

Input for normalization are raw .RCC files. No other Nanostring files need to be provided, neither preprocessing with nSolver or other software is required.

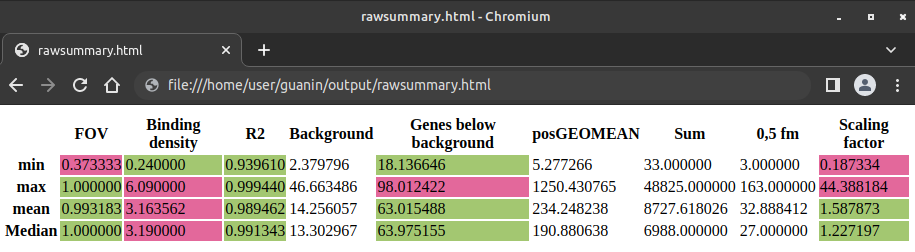
Sample info [condition/group] can be provided to refine content normalization. Although it’s recommended, it’s not mandatory. This sample info .csv file can be created using a text editor or spreadsheet software such as Excel. It should contain 2 columns: “SAMPLE” (that needs to match sample identification option (sample ID or file name) and “GROUP”.

If “sample ID” field in the .RCC files have been declared and you trust this information (no duplicates, etc), “sample ID” is recommended to be used as sample identifier. Else, filename is default parameter as it is foolproof.

→OUTPUT: rawinfolanes.csv (sample information)



→OUTPUT: rawsummary (samples summary).



**PRELIMINARY QC INSPECTION…**

- Background determination:

Negative controls are included in nanostring panels in order to set a threshold of non-expressed genes: the background. It can be more or less restrictive, depending on the characteristics of the experiment.

Default background is calculated as the mean of the negative controls + twice the standard deviation. Max of negative controls or mean of negative controls can be used. Also, Guanin implements a new method of selection alternative negative controls, useful in the case there is a problem with predefined negative controls (i.e: they are expressed). In this case, a alternative background is calculated from low-expressed genes among the endogenous. Aditionally, background can be set manually.

- Background correction:

Once background is set, there are several options to handle values below background (low counts):

1. Assimilate to background: Sets all values < background as equal to background.
2. Subtract background: Sets all values as value - background, assigning 0 value to genes expressed equally or lower than background level. (default)
3. Skip: Ignores background correction by not performing any correction.

- Sample inspection:

Samples with QC abnormaliites can be a) flagged or b) removed from the analysis. QC flag values can be set:

1. % of genes below background: A big amount of genes being expressed below background can relate problems with the sample. By default, Guanin flags samples that have more than 80% of their genes less expressed than the background. Lower % values can refine more strictly.

Samples with FOV, BD, linearity or scaling factor values below or above recommended levels can be related with errors:

1. Field of view: Default values: [0.75 - 1]
2. Binding density: Default values: [0.1 - 1.8]
3. Linearity: Default values: [0.75 - 1]
4. Scaling factor: Default values: [0.3 - 3]

Additionaly, samples can be manually selected to remove from the analysis. SampleIDs or filenameIDs should not contain spaces, and files manually selected to remove are input separated by spaces (“sample1 sample2 sample3”).

Default values for preliminary QC inspection need to be set for plotting and calculations.

Then, parameters can be modified in order to re-run QC and refine QC thresholds.

→OUTPUT\_files:

rawcounts.csv (raw matrix counts of endogenous genes)

rawfcounts.csv (raw matrix counts of filtered samples, endogenous genes)

dfhkecounts.csv (raw matrix counts of housekeeping genes)

posnegcounts.csv (raw matrix counts of positive and negative controls)

→OUTPUT\_reports: QC inspection pdf in output/reports (ejemplo adjunto)

Summary infolanes (filtered lanes info)

QCflags.txt (info about what samples have been flagged/discarded and why)

**TECHNICAL NORMALIZATION**

In order to perform technical normalization replicates with known concentration are used. This positive controls are used to calculate a lane-specific scaling factor that can be derived from:

1. posgeomean of positive controls (default)
2. summation of positive controls
3. median of positive controls

Note: \*Although Nanostring nCounter performs first background correction and after that technical normalization, other tools that throw better normalization results apply background correction over technically normalized data, Guanin has proved to obtain better normalization results with this procedure too. For this reason, although for the user this is a conceptually posterior process, QC inspection and technical normalization are performed together.

[esto no lo explica ninguna otra herramienta… tampoco tenemos “datos” para justificarlo (los demás tampoco lo justifican…) igual sobra esta info?]

→OUTPUT\_files:

tnormcounts.csv (matrix counts after technical normalization)

**CONTENT NORMALIZATION**

Choosing appropiate housekeeping genes is crucial for normalization. That’s why content normalization can be performed using:

1. Default panel housekeeping genes (filtered or not)
2. Default housekeeping + best endogenous candidate reference genes
3. All endogenous genes
4. Manual selection of genes

As housekeeping genes are supposed to have stable high expression on every sample, it is recommended to discard any of them if it is lowly expressed at any sample. Default value for exclussion is set as 50, but higher values are encouraged.

Including most promising endogenous genes that can be used as housekeeping is a Guanin unique feature, that uses findERG algorithm. It finds among endogenous genes the most stably and high expressed among all lanes.

As a standard panel includes 12 housekeeping genes, a number of endogenous candidates is encouraged to be included between 4 and 12 (default 6). This endogenous candidates enter with housekeeping in a evaluation pool and, depending on the results of the final reference genes selection, can be reasonable to re-run including more or less (if housekeeping are bad and all endogenous are chosen over them, for example, more endogenous could be included).

For this evaluation of candidate reference genes, geNorm algorithm is used, retrieving a ranked list of best candidate reference genes, and calculating the optimal number to use.

Indeed, a brand new content normalization approach is provided, using ponderated weights of every reference gene based on its ranking geNorm value.

In this way, several combination of parametrizations can be used and refined:

1. What genes to include in the pool as candidate ref genes?
   1. Housekeeping only
   2. Housekeeping + n best endogenous (default)
2. Wich genes from the pool will be selected?
   1. geNorm n and gene names intelligent selection (default)
   2. genorm n and gene names intelligent ponderated selection
   3. Top n best from genorm ranking
3. Avoid geNorm calculations, use…
   1. All endogenous genes
   2. Top n most expressed genes
   3. Manual selection of reference genes

Once reference genes are chosen, Guanin allows to perform a [exclusive feature] additional filtering in order to ensure they are not differencially expressed between groups.

If groups are set, they can be flagged or filtered if Kruskal-Wallis or Wilcoxon’s tests reveal that they are significantly differentially expressed among groups. In the case of Wilcoxon’s tests, Guanin performs every pair of group comparisons (if groups > 2).

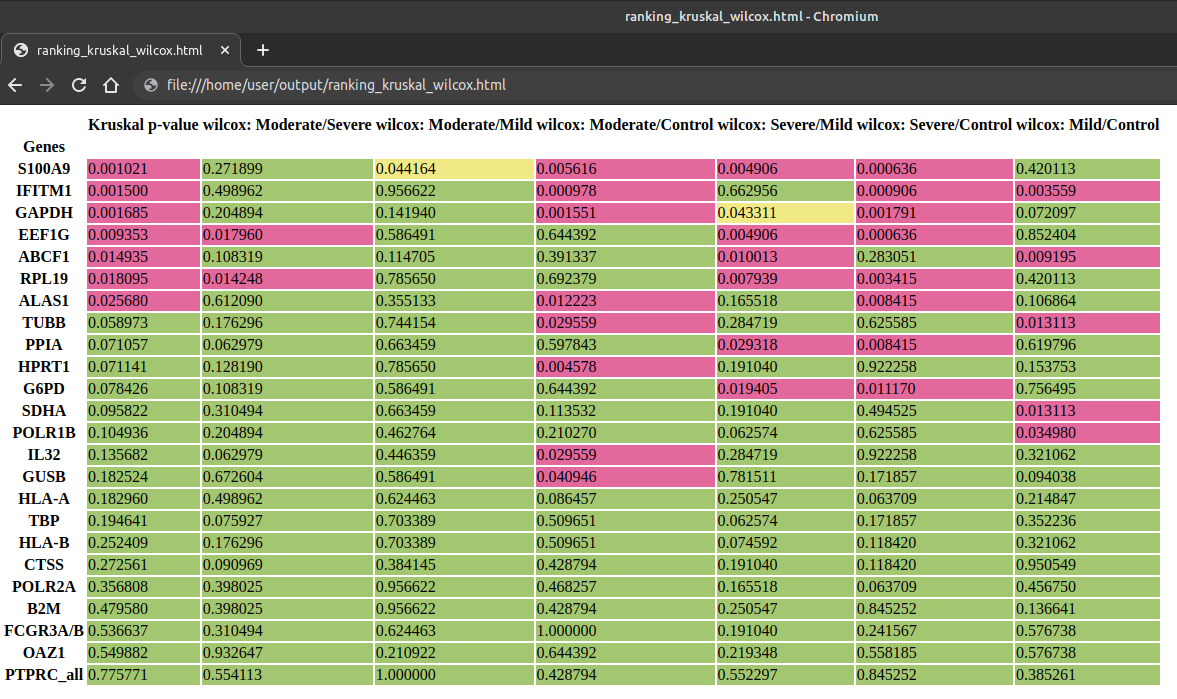
Additionally, a (only informative) reverse feature selection ranking can be shown in order to dilucidate what reference genes are more significantly (and how much) revealing a relation with group predicting. Can be interesting to interpret wich genes are the ones the machine learning algorithm considers the most representative and discards them the last, and with what accuracy the algorithm can predict to wich groups belongs a sample from one or a group of reference genes. This accuracy should be close to 1/j, where j is the number of groups declared in our experiment, and approximately could be warning that something wrong is happening if accuracy is closer to 2\*(1/j) than to 1/j.

In case of individual genes, this predictive ability will be spotted by Kruskal or Wilcoxon filtering, so no relevant results should be thrown. But it is in case of additive effect of a combination of genes that results in predictive vinculating effects with the output group when we can spot problems with our reference genes selection.

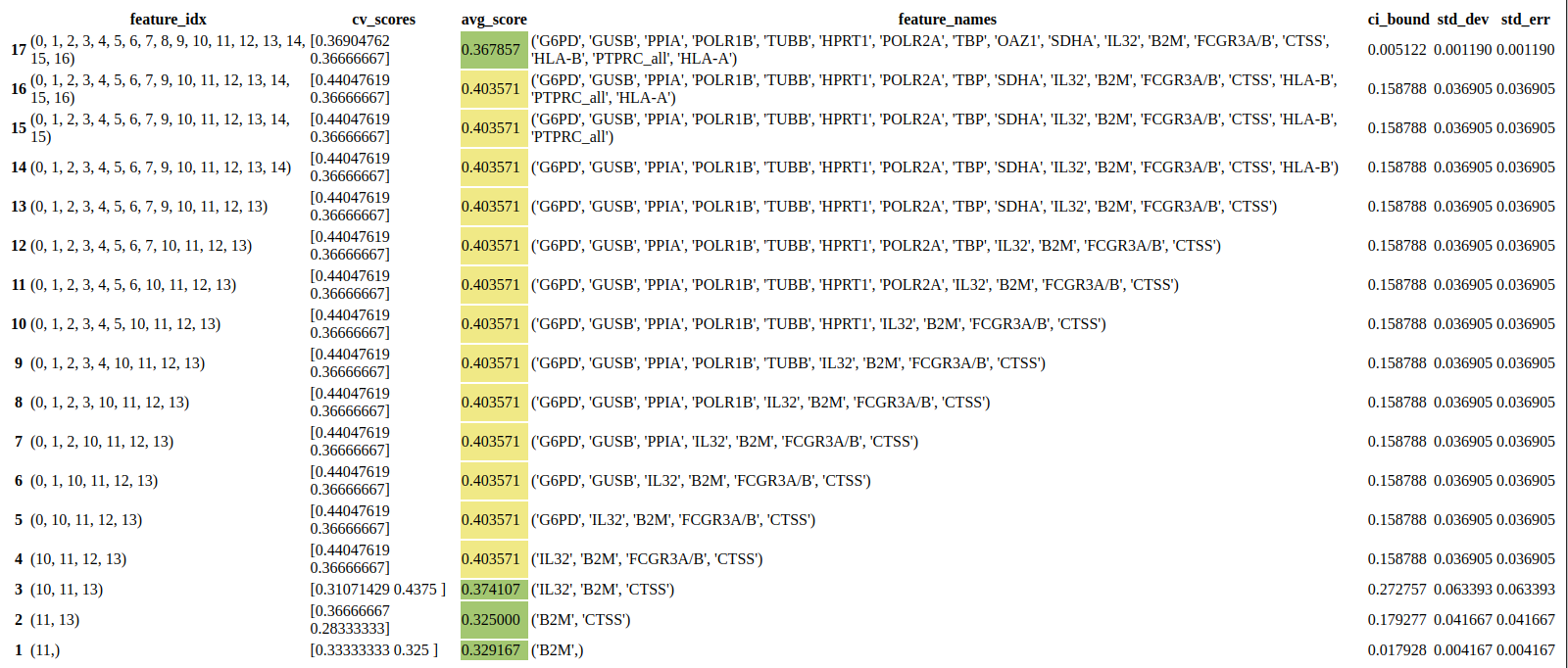
[Esto es un poco liada conceptualmente, creo que es un buen punto pero no sé yo si estoy sabiendo expresarlo bien pa que se entienda de una sin enrollarme mucho]

OUTPUT\_files: refgenes.csv (count matrix of chosen reference genes)

rnormcounts.csv (content-normalized count matrix)

OUTPUT\_reports: ranking\_kruskal\_wilcox.csv (p values of association of reference gene expression with groups. Values > 0.05 for all association are encouraged)

metrics\_reverse\_feature\_selection.csv (predictability of groups by association of several reference genes. Values similar to 1/groups are encouraged. High accuracy prediction with few combination of genes may indicate bad combination of reference genes



**ADDITIONAL NORMALIZATION…**

Aditional normalization can be useful in the case we need our data in an specific format, such as in a range to 0-1, etc. For that, it can be implemented:

1. quantile normalization
2. standarization

OUTPUT\_files: adnormcounts.csv (count matrix of additionaly normalized data)

**EVALUATION OF NORMALIZATION**

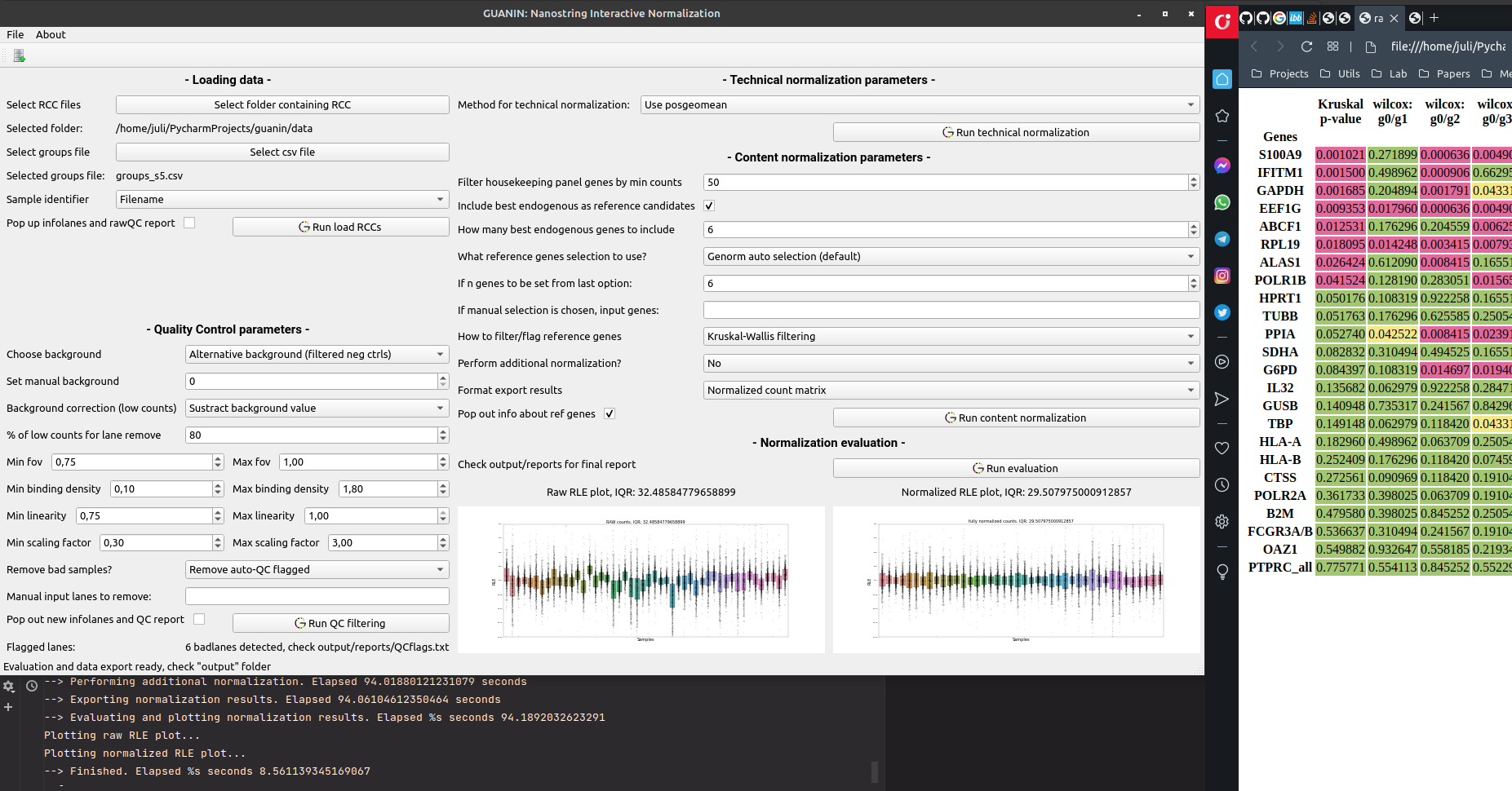
In order to assess if normalization is offering reasonable results, two measures can be used that make use of relative log expression.

Relative log expression are useful for visualizing unwanted variation.

1. RLE plots, comparing pre-normalization and post-normalization. Narrow boxplots mean less unknown expression differences.
2. IQR, that can be used to numerically compare different normalization parametrizations that could suit our experiment.

OUTPUT\_reports: norm\_report.pdf (showing genorm results for reference genes chosen, RLE plots and IQR).

https://i.imgur.com/TBTcTnm.png



**CLI COMMANDS HELP:**

- Argument: '-f', '--folder', *type*=*str*, *default*= pathlib.Path.cwd() / 'examples/d1\_COV\_GSE183071', *help*='relative folder where RCC set is located. **-** Argument: '-minf', '--minfov', *type*=*float*, *default*=0.75, *help*='set manually min fov for QC' **-** Argument: '-maxf', '--maxfov', *type*=*float*, *default*=1, *help*='set manually max fov for QC' **-** Argument: '-minbd', '--minbd', *type*=*float*, *default*=0.1, *help*='set manually min binding density for QC' **-** Argument: '-maxbd', '--maxbd', *type*=*float*, *default*=1.8, *help*='set manually max binding density for QC'  **-** Argument: '-minlin', '--minlin', *type*=*float*, *default*=0.75, *help*='set manually min linearity for QC' **-** Argument: '-maxlin', '--maxlin', *type*=*float*, *default*=1, *help*='set manually max linearity for QC' **-** Argument: '-minscaf', '--minscalingfactor', *type*=*float*, *default*=0.3, *help*='set manually min scaling factor for QC' **-** Argument: '-maxscaf', '--maxscalingfactor', *type*=*float*, *default*=3, *help*='set manually max scaling factor for QC' **-** Argument: '-swbrrq', '--showbrowserrawqc', *type*=*bool*, *default*=False, *help*='pops up infolanes and qc summary'  **-** Argument: '-swbrq', '--showbrowserqc', *type*=*bool*, *default*=False, *help*='pops up infolanes and qc summary') **-** Argument: '-swbrcn', '--showbrowsercnorm', *type*=*bool*, *default*=False, *help*='pops up infolanes and qc summary') **-** Argument: '-lc', '--lowcounts', *type*=*str*, *default*='sustract', *choices*=['skip', 'asim', 'sustract'], *help*='what to do with counts below background?') **-** Argument: '-mi', '--modeid', *type*=*str*, *default*='filename', *choices*=['sampleID','filename', 'id+filename'], *help*='choose sample identifier. sampleID: optimal if assigned in rccs. filenames: easier to be unique. id+filename: care with group assignment coherence') **-** Argument: '-mv', '--modeview', *type*=*str*, *default*='view', *choices*=['justrun', 'view'], *help*='choose if plot graphs or just run calculations') **-** Argument: '-tnm', '--tecnormeth', *type*=*str*, *default*='posgeomean', *choices*=['posgeomean','Sum', 'Median', 'regression'], *help*='choose method for technical normalization') **-** Argument: '-reg', '--refendgenes', *type*=*str*, *default*= 'endhkes', *choices*=['hkes', 'endhkes'], *help*='choose refgenes, housekeeping, or hkes and endogenous') **-** Argument: '-re', '--remove', *type*=*str*, *nargs*='+', *default*=None, *help*='lanes to be removed from the analysis') **-** Argument: '-bg', '--background', *type*=*str*, *default*= 'Background', *choices*=['Background', 'Background2', 'Background3', 'Backgroundalt'], *help*='choose background: b1=meancneg+(2\*std), b2=maxcneg, b3=meancneg, balt=') **-** Argument: '-pbb', '--pbelowbackground', *type*=*int*, *default*=85, *help*='if more than %bb genes are below background, sample gets removed from analysis') **-** Argument: '-mbg', '--manualbackground', *type*=*float*, *default*=None, *help*='set manually background') **-** Argument: '-crg', '--chooserefgenes', *type*=*str*, *nargs*='+', *default* = None, *help* = 'list of strings like. choose manualy reference genes to use over decided-by-program ones') **-** Argument: '-fgv', '--filtergroupvariation', *type*=*str*, *default*='filterkrus', *choices*=['filterkrus', 'filterwilcox', 'flagkrus', 'flagwilcox', 'nofilter'], *help*='¿filter or flag preselected ref genes by significative group-driven differences? needs groups to be declared') **-** Argument: '-fsn', '--featureselectionneighbors', *type*=*float*, *default*=4, *help*='number of neighbors for feature selection analysis of refgenes. Recommended 3-6') **-** Argument: '-g', '--groups', *type*=*str*, *default*='yes', *choices*=['yes','no'], *help*='defining groups for kruskal/wilcox/fs analysis?') **-** Argument: '-ne', '--numend', *type*=*int*, *default*=6, *help*='number of endogenous tofind by ERgene to include in analysis to check viability as refgenes') **-** Argument: '-ar', '--autorename', *type*=*str*, *default*='off', *choices*=['on', 'off'], *help*='turn on when sample IDs are not unique, be careful on sample identification detail') **-** Argument: '-cn', '--contnorm', *type*=*str*, *default*='refgenes', *choices*=['ponderaterefgenes', 'refgenes', 'all', 'topn']) **-** Argument: '-an', '--adnormalization', *type*=*str*, *default*='no', *choices*=['no', 'standarization', 'quantile'], *help*='perform additional normalization? standarization and quantile normalization available') **-** Argument: '-tn', '--topngenestocontnorm', *type*=*int*, *default*=100, *help*='set n genes to compute for calculating norm factor from top n expressed endogenous genes') **-** Argument: '-mch', '--mincounthkes', *type*=*int*, *default*=80, *help*='set n min counts to filter hkes candidate as refgenes') **-** Argument: '-nrg', '--nrefgenes', *type*=*int*, *default*=None, *help*='set n refgenes to use, overwriting geNorm calculation') **-** Argument: '-lr', '--laneremover', *type*=*str*, *default*='yes', *choices*=['yes', 'no'], *help*='option to perform analysis with all lanes if set to no') **-** Argument: '-grn', '--groupsinrnormgenes', *type*=*str*, *default*='no', *choices*=['yes', 'no'], *help*='want groups to be specified in last column of rnormgenes dataframe?') **-** Argument: '-lo', '--logarizedoutput', *type*=*str*, *default*='10', *choices*=['2', '10', 'no'], *help*='want normed output to be logarized? in what logbase?') **-** Argument: '-le', '--logarizeforeval', *type*=*str*, *default*='10', *choices*=['2', '10', 'no'], *help*= 'logarithm base for RLE calculations') **-** Argument: '-gf', '--groupsfile', *type*=*str*, *default*='examples/groups\_d1\_COV\_GSE183071.csv', *help*='enter file name where groups are defined') **-** Argument: '-st', '--start\_time', *type*=*float*, *default* = time.time()) **-** Argument: '-cs', '--current\_state', *type*=*str*, *default*='Ready') **-** Argument: '-ftl', '--firsttransformlowcounts', *type*=*bool*, *default*=True) **-** Argument: '-of', '--outputfolder', *type*=*str*, *default*= tempfile.gettempdir() + '/guanin\_output') **-** Argument: '-sll', '--showlastlog', *type*=*bool*, *default* = False)