**CrisprCountsAnalysis (CCA) Method**

CCA is based on finding genes whose knockout effect is the greatest between the test and control samples as well as the knockout of the gene’s ability to kill a cell in the test samples. We also apply purely non-parametric statistics, as the distribution of counts and foldchanges often do not follow a probabilistic distribution, thus robust in the presence of outliers. Implementation of CCA was based on MolBioLib [1, 2] and includes the Mann Whitney U test from ALGLIB C++ [3].

The input of CCA is a matrix of samples versus sgRNAs where the entries are the raw number of counts of sgRNA barcodes seen in that sample. CCA’s steps for computing the CCA score by which the genes are

1. Normalize the count file so that each sample’s count over all sgRNAs is 10 million.
2. Remove all sgRNAs where the T0 count is less than 30, avoiding false positives due to low read counts.
3. Compute a depletion matrix of samples versus sgRNAs where the  
     
   depletion = 1 – (count at final time)/(count at initial time) = 1 – foldchange   
   1. The depletion is such that it is maximum, 1, if the test sample has no viable cells at the final time. The depletion may be negative if there is proliferation of cells at the final time.
4. By default, we limit the minimum value of the depletion of all control samples to 0.
   1. Doing otherwise may create false positive hits of, for example,n tumor suppressor genes where they proliferate greatly when knocked out in the control samples.
5. For a given gene, let the vector of depletion values over all test samples be denoted **t** and over all control samples be denoted **c**. For vector **v**, let Q3(**v**) be the third quantile of **v**. The CCA score for that gene is  
     
   Score = { A\*median(**t**) + B\*Q3(**t**) + C\*(median(**t**) – median(**c**)) + D\*(Q3(**t**) – Q3(**c**)) } \*  
    { 1 – (likelihood **t** < non-essential)E } \* { 1 – (likelihood **c** > essential)F } \*   
    { 1 – (likelihood **t** = **c**)G } \* { 1 – (likelihood **t** < **c**)H }  
     
   where A2, B0.017, C0.02, D=1, E8.8, F0.35, G7.1, and H0.22. Likelihoods are computed using Mann-Whitney U test where the inequality is tested by taking either the right or left tail and the equality is tested by taking both tails. For comparison with essential and non-essential genes, we use the set of genes by T. Hart [4, 5]. For essential genes, we use depletion values of all samples of all sgRNAs associated with an essential gene.
   1. For chemogenomic screens, typically A0.49, B4.0, C0.8, D=1, E0.3, F0, G1.3, and H0.58.
6. For isogenic screens, we subtract 10000 to all genes whose median(**t**) is less than zero. This is because for pharmaceutical purposes, we are only looking for genes that impact proliferations negatively.

The top 3000 CCA scores are modeled using a beta distribution fitted using the fitdistrplus package [6, 7] in R. Taking the top genes with p < 0.05, we stratify them into 4 Jenks classes [8] using the classInt package [9] in R.

The values of the parameters, A through H except D, were determined by using a derivative-free optimiziation method, BiteOpt [10], to minimize  
  
where numInTopN is the number of positive control synthetic lethal genes found in the top N genes as ranked by CCA’s scoring method over all training screens that have positive controls. D is always set to 1, as the other variables, A, B, and C, may be scaled.

[1] T. K. Ohsumi and M. L. Borowsky, MolBioLib: a C++11 framework for rapid development and deployment of bioinformatics tasks, Bioinformatics, Volume 28, Issue 19, 1 October 2012, 2412-2416.

[2] T. K. Ohsumi, MolBioLib ([sourceforge.net/projects/molbiolib](http://sourceforge.net/projects/molbiolib)).

[3] S. Bochkanov, ALGLIB ([www.alglib.net](http://www.alglib.net)).

[4] T. Hart and J. Moffat, BAGEL: a computational framework for identifying essential genes from pooled library screens, BMC Bioinformatics, 17 (2016).

[5] T. Hart, BAGEL v2 ([github.com/hart-lab/bagel](http://github.com/hart-lab/bagel)).

[6] M.-L. Delignette-Muller and C. Dutang, fitdistrplus : An R Package for Fitting Distributions. Journal of Statistical Software, 64(4), 2015, 1-34.

[7] M.-L. Delignette-Muller and C. Dutang, fitdistrplus ([cran.r-project.org/web/packages/fitdistrplus/index.html](http://cran.r-project.org/web/packages/fitdistrplus/index.html)).

[8] G. F. Jenks, The Data Model Concept in Statistical Mapping, International Yearbook of Cartography, 7, 1967, 186-190.

[9] R. Bivand, classInt ([cran.r-project.org/web/packages/classInt/index.html](http://cran.r-project.org/web/packages/classInt/index.html)).

[10] A. Vaneev, biteOpt ([github.com/avaneev/biteopt](http://github.com/avaneev/biteopt)).