GE Healthcare

Protocol

Dharmacon™ Bioinformatic Analysis of Pooled Lentiviral Library Screens

A guide for performing differential expression analysis of high-throughput sequencing data and calling hits resulting from pooled lentiviral shRNA, microRNA and sgRNA library screens.

Table of Contents

1 Legal disclaimers

SOFTWARE LICENSE TERMS. With respect to any software products incorporated in or forming a part of the software provided or described hereunder ("Software"), Dharmacon, now part of GE Healthcare. ("Seller") and you, the purchaser, licensee and/or end-user of the software ("Buyer") intend and agree that such software products are being licensed and not sold, and that the words "purchase", "sell" or similar or derivative words are understood and agreed to mean "license", and that the word "Buyer" or similar or derivative words are understood and agreed to mean "licensee". Notwithstanding anything to the contrary contained herein, Seller or its licensor, as the case may be, retains all rights and interest in software products provided hereunder.

Seller hereby grants to Buyer a royalty-free, non-exclusive, nontransferable license, without power to sublicense, to use software provided hereunder solely for Buyer's own internal business purposes and to use the related documentation solely for Buyer's own internal business purposes. This license terminates when Buyer's lawful possession of the hardware products provided hereunder ceases, unless earlier terminated as provided herein. Buyer agrees to hold in confidence and not to sell, transfer, license, loan or otherwise make available in any form to third parties the software products and related documentation provided hereunder. Buyer may not disassemble, decompile or reverse engineer, copy, modify, enhance or otherwise change or supplement the software products provided hereunder without Seller's prior written consent. Seller will be entitled to terminate this license if Buyer fails to comply with any term or condition herein. Buyer agrees, upon termination of this license, immediately to return to Seller all software products and related documentation provided hereunder and all copies and portions thereof.

Certain of the software products provided by Seller may be owned by one or more third parties and licensed to Seller. Accordingly, Seller and Buyer agree that such third parties retain ownership of and title to such software products. The warranty and indemnification provisions set forth herein shall not apply to software products owned by third parties and provided hereunder.

LIMITATIONS ON LIABILITY. NOTWITHSTANDING ANYTHING TO THE CONTRARY CONTAINED HEREIN, THE LIABILITY OF SELLER UNDER THESE TERMS AND CONDITIONS (WHETHER BY REASON OF BREACH OF CONTRACT, TORT, INDEMNIFICATION, OR OTHERWISE, BUT EXCLUDING LIABILITY OF SELLER FOR BREACH OF WARRANTY (THE SOLE REMEDY FOR WHICH WILL BE AS PROVIDED UNDER THE WARRANTY SECTION BELOW)) WILL NOT EXCEED AN AMOUNT EQUAL TO THE TOTAL PURCHASE PRICE PAID BY BUYER TO SELLER WITH RESPECT TO THE PRODUCT(S) GIVING RISE TO SUCH LIABILITY. NOTWITHSTANDING ANYTHING TO THE CONTRARY CONTAINED HEREIN, IN NO EVENT WILL SELLER BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL OR INCIDENTAL DAMAGES (INCLUDING WITHOUT LIMITATION DAMAGES FOR LOSS OF USE OF FACILITIES OR EQUIPMENT, LOSS OF REVENUE, LOSS OF DATA, LOSS OF PROFITS OR LOSS OF GOODWILL), REGARDLESS OF WHETHER SELLER (a) HAS BEEN INFORMED OF THE POSSIBILITY OF SUCH DAMAGES OR (b) IS NEGLIGENT.

WARRANTY AND DISCLAIMER OF WARRANTIES. The software is provided "AS IS" and should be used at the users own risk. No guarantee of analysis or results is provided or should be implied.

NO OTHER WARRANTIES, EXPRESS OR IMPLIED, ARE GRANTED, INCLUDING WITHOUT LIMITATION, IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR ANY PARTICULAR PURPOSE, OR NON INFRINGEMENT. BUYER'S EXCLUSIVE REMEDY FOR NON-CONFORMING PRODUCTS DURING THE WARRANTY PERIOD IS LIMITED TO REPAIR, REPLACEMENT OF OR REFUND FOR THE NON-CONFORMING PRODUCT(S) AT SELLER'S SOLE OPTION. THERE IS NO OBLIGATION TO REPAIR, REPLACE OR REFUND FOR PRODUCTS AS THE RESULT OF (I) ACCIDENT, DISASTER OR EVENT OF FORCE MAJEURE, (II) MISUSE, FAULT OR NEGLIGENCE OF OR BY BUYER, (III) USE OF THE PRODUCTS IN A MANNER FOR WHICH THEY WERE NOT DESIGNED, OR (IV) IMPROPER STORAGE AND HANDLING OF THE PRODUCTS.

If Seller determines that Products for which Buyer has requested warranty services are not covered by the warranty hereunder, Buyer shall pay or reimburse Seller for all costs of investigating and responding to such request at Seller's then prevailing time and materials rates. If Seller provides repair services or replacement parts that are not covered by this warranty, Buyer shall pay Seller therefor at Seller's then prevailing time and materials rates. ANY INSTALLATION, MAINTENANCE, REPAIR, SERVICE, RELOCATION OR ALTERATION TO OR OF, OR OTHER TAMPERING WITH, THE PRODUCTS PERFORMED BY ANY PERSON OR ENTITY OTHER THAN SELLER WITHOUT SELLER'S PRIOR WRITTEN APPROVAL, OR ANY USE OF REPLACEMENT PARTS NOT SUPPLIED BY SELLER, SHALL IMMEDIATELY VOID AND CANCEL ALL WARRANTIES WITH RESPECT TO THE AFFECTED PRODUCTS. THE OBLIGATIONS CREATED BY THIS WARRANTY STATEMENT TO REPAIR OR REPLACE A DEFECTIVE PRODUCT SHALL BE THE SOLE REMEDY OF BUYER IN THE EVENT OF A DEFECTIVE PRODUCT. SELLER DOES NOT WARRANT THAT THE PRODUCTS ARE ERROR-FREE OR WILL ACCOMPLISH ANY PARTICULAR RESULT.

EXPORT RESTRICTIONS. Buyer acknowledges that each Product and any related technology, including technical information supplied by Seller or contained in documents (collectively "Items"), is subject to export controls of the U.S. government. The export controls may include, but are not limited to, those of the Export Administration Regulations of the U.S. Department of Commerce (the "EAR"), which may restrict or require licenses for the export of Items from the United States and their re-export from other countries. Buyer will comply with the EAR and all other applicable laws, regulations, laws, treaties, and agreements relating to the export, re-export, and import of any Item. Buyer will not, without first obtaining the required license to do so from the appropriate U.S. government agency; (i) export or re-export any Item, or (ii) export, re-export, distribute or supply any Item to any restricted or embargoed country or to a person or entity whose privilege to participate in exports has been denied or restricted by the U.S. government. Buyer will cooperate fully with Seller in any official or unofficial audit or inspection related to applicable export or import control laws or regulations, and will indemnify and hold Seller harmless from, or in connection with, any violation of this Section by Buyer or its employees, consultants, agents, or customers.

2 Summary

This guide is intended to walk the user through the steps required to determine primary hits from the output of a Dharmacon™ Decode™ shRNA, SMARTvector™ shRNA, shMIMIC™ microRNA or Edit-R™ sgRNA Pooled Screening Libraries. This guide covers the simplest case, where the abundance of constructs is compared between two conditions, but should give the reader a good foundation for analyzing more complex experiments. All scripts referenced in this guide can be downloaded from dharmacon.gelifesciences.com/resource-library.

3 Intended audience

There are many open source tools available for the analysis of high-throughput sequencing data and for performing differential expression analysis. Most of these tools are built for the Linux environment and require proficiency in using command line tools. This guide is intended for those who are familiar with Linux and running programs from the command line.

4 Software requirements

The following programs need to be installed prior to using this guide. Please refer to the websites of each program for documentation of hardware and software prerequisites. Older versions may have slight changes in the command name and options, so we advise using these versions, or newer, if possible:

- R v(2.15.3) – The R statistical framework is used to perform the differential expression analysis and can be found at r-project.org.
- Bioconductor v(2.11.0) – This is a R package that is used to handle biological data and can be found at bioconductor.org.
- Bowtie2 v(2.1.0) – This is a short read aligner for processing the FASTQ files and can be found at bowtie-bio.sourceforge.net/bowtie2.
- DESeq v(1.10.1) – This is a R package built against Bioconductor, which performs the differential expression analysis and can be found at bioconductor.org/packages/release/bioc/html/DESeq.html.
- Python v(2.7.3) – This is an interpreter for the Python programming language and, in this case, 2.7.x is required as newer versions of the interpreter are not backwards-compatible. This guide will provide a couple of Python scripts for converting alignment files into a table file that can be parsed by DESeq. Python v(2.7.3) or later can be found at python.org.

5 FASTQ files from high-throughput sequencing

The starting point for this analysis is the FASTQ sequence files from the high-throughput sequencing instrument. These files will be provided to you by your sequencing service provider. Copy the FASTQ files to your local Linux machine and decompress them if necessary. You should be provided with several FASTQ files, each file pertaining to an individual sample. Occasionally these files are split into several files for a single sample, but your sequencing service provider should make it clear which FASTQ file(s) pertain to each sample. The intermediate files generated during the analysis described here require additional disk space. It is recommended that you have twice the disk space of your raw uncompressed FASTQ files. For instance, if your FASTQ files take up 20 gigabytes total, it is recommended that you have 40 gigabytes of free disk space.

6 Align the FASTQ files to reference shRNA, microRNA or sgRNA (construct) FASTA files

In order to measure the relative quantity of construct integration events in a sample, you must align the sequencing reads to a reference file. The number of alignments to each individual construct will be an estimate of the relative abundance of that construct in your sample. The FASTA file(s) containing construct sequences (construct FASTA file(s)) will be provided with each Dharmacon pooled lentiviral library shipment.

A. Create a Bowtie2 index

To create a Bowtie2 index from the construct FASTA file provided by Dharmacon, copy the FASTA file to the directory where the FASTQ files were placed. Run the bowtie2-build command on the FASTA file to obtain a set of Bowtie2 index files. You can do this by issuing the following command (where reference_list.fasta is the construct FASTA file):

```
bowtie2-build reference_list.fasta reference_list
```

This will generate several files that contain the index information. The prefix will be the second argument of the above command, in this case reference_list. The suffix will be a number or "rev" followed by .bt2.

B. Align using Bowtie2

Once Bowtie2 index files have been created, the FASTQ high-throughput sequencing reads can be aligned against the construct FASTA file using Bowtie2. For each FASTQ file (or set of files), run the following command (see the Batch Process section (6.C.) for running several samples . For pooled lentiviral shRNA and microRNA library screens trim 28 bases from the 3' end of each read ("-3 28"):

```
bowtie2 -p 4 -3 28 -x reference_list -U ReferenceRep1.fastq -S ReferenceRep1.sam
```

For pooled lentiviral sgRNA library screens trim 31 bases from the 3' end of each read ("-3 31"):

```
bowtie2 -p 4 -3 31 -x reference_list -U ReferenceRep1.fastq -S ReferenceRep1.sam
```

If you have multiple FASTQ files for a single sample you can append them to the end of the command as follows:

```
bowtie2 -p 4 -3 28 -x reference_list -U ReferenceRep1A.fastq,ReferenceRep1B.
fastq,ReferenceRep1C.fastq -S ReferenceRep1.sam
```

The following is an explanation of the Bowtie2 options used above, but you can modify these parameters to fit your specific application:

- -p 4: will run Bowtie2 with 4 threads. Change this to the number of cores your local machine has. For example, if you are using a dual core machine, specify -p 2.
- -3 28: will trim off 28 bases from the 3' end of your read. For pooled lentiviral shRNA and microRNA library screens only the first 22 bases of the read are needed to uniquely identify a construct. Here, 28 was determined by subtracting 22 from a read length of 50 bases. For pooled lentiviral sgRNA library screens only the first 19 bases of the read are needed to uniquely identify a construct, so 31 bases should be trimmed from a read length of 50 bases. For any other read length, you will have to determine the number of bases to trim from the 3' end of your read to obtain 22 or 19 bases of sequence.
- -x reference_list: This is the prefix used when creating the Bowtie2 index. This is the prefix to the files that have the extension .bt2. Only supply the prefix as a parameter, not the entire file name.
- -U ReferenceRep1A-C.fastq: This is the FASTQ file(s) that represent a single sample, which can be one or many FASTQ files, depending on how your high-throughput sequencing service provider prefers to supply sample data.
- -S ReferenceRep1.sam: This is the output file name for writing Bowtie2 output in Sequence Alignment/Map (SAM) format. Use a file name that will appropriately differentiate between your samples. This file will later be converted into a format that can be used as the input for the differential expression analysis program DESeq.

Generally, the options outlined above are sufficient for aligning the reads. For a more detailed look at the available Bowtie2 options, you can issue the following command:

```
bowtie2 -h
```

Repeat the Bowtie2 command for each sample in your analysis. The following section describes how to automatically run multiple samples using a bash script.

C. Batch process

Since it may take several hours to process each sample, we recommend creating a bash script to run all the samples in series (refer to Bowtie2 documentation for running time estimates). For example, if you have four test samples to analyze, two biological replicates for the reference sample and two biological replicates for the experimental sample, you could create a file that looks like this:

```
#!/bin/bash
#My lentiviral pooled screening bash script
echo ReferenceRep1.fastq
bowtie2 -p 4 -3 28 -x reference_list -U ReferenceRep1.fastq -S ReferenceRep1.sam
echo ReferenceRep2.fastq
bowtie2 -p 4 -3 28 -x reference_list -U ReferenceRep2.fastq -S ReferenceRep2.sam
echo ExperimentRep1.fastq
bowtie2 -p 4 -3 28 -x reference_list -U ExperimentRep1.fastq -S ExperimentRep1.sam
echo ExperimentRep2.fastq
bowtie2 -p 4 -3 28 -x reference_list -U ExperimentRep2.fastq -S ExperimentRep2.sam
```

You can download this script to your local directory from: dharmacon.gelifesciences.com/uploadedFiles/Resources/alignAll.sh. After downloading, the script can be modified to match your number of samples, sample file names, index prefix and read length. The following command can then be run:

```
nohup bash alignAll.sh &
```

The nohup command will ensure that the process will not be terminated even if you log out of the terminal. The ampersand (&) at the end will place the command in the background. The echo lines will help you determine which FASTQ file is being processed when parsing the nohup.out file. This file will appear in the directory from which you run the nohup command. The nohup.out file contains all the output written to stderr and stdout by the Bowtie2 process and the bash script, which includes some alignment summaries that are useful for troubleshooting. To determine if the Bowtie2 process is still running, you can use the top command.

D. Alignment summaries

If you use the bash script approach for running multiple samples (recommended), then there will be useful alignment summaries automatically written to the nohup.out file. If running individually, the summaries will be printed to stderr when each alignment is complete.

Each summary displays: 1) the number of reads processed; 2) the number of unpaired reads (should be same as number of reads processed); 3) the number of reads that aligned 0 times (failed to align); 4) the number of reads successfully aligned exactly 1 time or > 1 times, and the overall alignment rate. The following is an example of what the nohup.out file looks like:

```
ReferenceRep1.fastq
6458981 reads; of these:
  6458981 (100.00%) were unpaired; of these:
    958019 (14.83%) aligned 0 times
    5487858 (84.96%) aligned exactly 1 time
    13104 (0.20%) aligned >1 times
85.17% overall alignment rate
ReferenceRep2.fastq
8292012 reads; of these:
  8292012 (100.00%) were unpaired; of these:
    1237541 (14.92%) aligned 0 times
    7033710 (84.83%) aligned exactly 1 time
    20761 (0.25%) aligned >1 times
85.08% overall alignment rate
ExperimentRep1.fastq
6474817 reads; of these:
  6474817 (100.00%) were unpaired; of these:
    1044282 (16.13%) aligned 0 times
    5418676 (83.69%) aligned exactly 1 time
    11859 (0.18%) aligned >1 times
83.87% overall alignment rate
ExperimentRep2.fastq
9794566 reads; of these:
  9794566 (100.00%) were unpaired; of these:
    1537929 (15.70%) aligned 0 times
    8237661 (84.10%) aligned exactly 1 time
    18976 (0.19%) aligned >1 times
84.30% overall alignment rate
```

If the number of successful alignments is < 70%, make sure you are using the correct reference construct FASTA file, and ensure that the -3 option is set correctly for your read length (refer to section 6.B.). If you are certain that you are using the correct construct FASTA file and the correct -3 option, then contact your sequencing service provider to ensure that there were no problems with the sequencing.

7 Differential expression analysis

Once the Bowtie2 program has completed aligning the FASTQ sequencing reads against the construct FASTA file, you should have a file with the suffix .sam for each of your samples. In the simplest case you will be comparing the construct abundance of two conditions (such as Reference versus Experimental).

A. Convert Bowtie2 files into DESeq input

Before you can analyze the relative construct abundance between any two experimental conditions (i.e., analysis of differential expression), the Bowtie2 output must be converted into a format that is compatible with DESeq. This requires converting and combining the several Bowtie2 hit files into a single input file for DESeq.

The following is a Python script that can be used to convert each .sam file into a .count file, which can be useful to quickly inspect the number of alignments per construct.

```
""" Count number of FASTQ reads that align to each reference using Bowtie2. """

from argparse import ArgumentParser
from os.path import abspath

parser = ArgumentParser(description="Count Bowtie2 hits.")
parser.add_argument("-i", "--input_filepath", help="Input SAM-formatted file",
                    type=file, required=True)
parser.add_argument("-o", "--output_filepath", help="Output filepath",
                    type=str, required=True)

def calculateHitHash(hits_file):
    """ Returns a dictionary containing alignment hits """

    results = dict()
    for line in hits_file:
        cols = line.split('\t')
        if line.startswith('@') or len(cols) < 1:
            continue

        accession = cols[2]
        if accession in results:
            results[accession] += 1
        else:
            results[accession] = 1

    return results

def main():
    args = parser.parse_args()
    hits_file = args.input_filepath
    output_filepath = abspath(args.output_filepath)

    results = calculateHitHash(hits_file)

    with open(output_filepath, 'w') as output_file:
        for accession, count in results.iteritems():
            if (count > 0 and accession != '*' ):
                output_file.write("%s,%s\n" % (accession, str(count)))

if __name__ == "__main__":
    main()
```

Download the script to your local directory from: dharmacon.gelifesciences.com/uploadedFiles/Resources/countBowtie2Hits.py. The script takes input from the "-i" input file and writes the output to the supplied "-o" file. Run the following command for each .sam file that you have.

```
python countBowtie2Hits.py -i ReferenceRep1.sam -o ReferenceRep1.counts
```

At the end of this step, you should have a file that looks like this for each sample:

```
Reference_343689,346
Reference_221360,911
Reference_171978,1298
Reference_171972,396
Reference_315993,800
Reference_315990,806
Reference_220372,19
```

The first column is the Reference ID (used as a key for each of the constructs). The second column is the number of alignments for that construct. The following script will take all the .count files you produce and aggregate them into a single file for use with DESeq.

```
""" Combines count files into format that can be used by R for statistical
analysis. Specifically, the ability to identify significant differentially
expressed references with DESeq. """

from argparse import ArgumentParser

parser = ArgumentParser(description="Combines count files in R-table format.")
parser.add_argument("-i", "--input_filepaths",
                    help="Comma-separated list of input count files",
                    type=str, required=True)
parser.add_argument("-o", "--output_filepath", help="Output filepath",
                    type=str, required=True)

def main():
    args = parser.parse_args()
    files = args.input_filepaths.strip().split(',')
    output_filepath = args.output_filepath

    # parse count files into a single dictionary object
    reference_ids = dict()
    file_num = 1
    for filename in files:
        with open(filename) as clusterFile:
            for line in clusterFile:
                cols = line.strip().split(",")
                reference = cols[0]
                hits = float(cols[1])
                if reference in reference_ids:
                    if file_num in reference_ids[reference]:
                        reference_ids[reference][file_num] += hits
                    else:
                        reference_ids[reference][file_num] = hits
                else:
                    reference_ids[reference] = dict()
                    reference_ids[reference][file_num] = hits
            file_num += 1

    # write the combined count file
    with open(output_filepath, 'w') as output_file:
        output_file.write("%s\n" % ",".join(files))
        for k, v in reference_ids.iteritems():
            j = 1
            row = [k]
            while j < file_num:
                if j in v:
                    row.append(str(v[j]))
                else:
                    row.append(str(0))
                j += 1
            output_file.write("%s\n" % ",".join(row))

if __name__ == "__main__":
    main()
```

Download the script to your local directory from: dharmacon.gelifesciences.com/uploadedFiles/Resources/createRTable.py.

Here is an example of running the analysis with four samples. The order in which the samples are listed is extremely important for this command. All replicates of the first condition should be listed before all replicates of the second condition. This ensures that the differential expression analysis is performed between experimental samples, not biological replicates.

```
python createRTable.py -i ReferenceRep1.counts,ReferenceRep2.counts,ExperimentRep1.
counts,ExperimentRep2.counts -o combinedCounts.rtable
```

If you have more than two replicates for each sample, you should add them as well. Please be sure that you list all the count files associated with one condition before you add the second condition. For example, if you have three replicates for each condition, the command would look as follows:

```
python createRTable.py -i ReferenceRep1.counts,ReferenceRep2.counts,ReferenceRep3.
counts,ExperimentRep1.counts,ExperimentRep2.counts,ExperimentRep3.counts -o combinedCounts.rtable
```

The resulting .rtable file will show all the conditions in a table. Here is an example:

```
ReferenceRep1.counts,ReferenceRep2.counts,ExperimentRep1.counts,ExperimentRep2.counts
Reference_343689,346.0,232.0,341.0,601.0
Reference_221360,911.0,1734.0,2394.0,3655.0
Reference_171978,1298.0,2452.0,3315.0,3845.0
Reference_171972,396.0,563.0,414.0,360.0
Reference_315993,800.0,1292.0,471.0,761.0
Reference_315990,806.0,1684.0,418.0,2577.0
```

B. Run DESeq on .rtable files

Now you should have a single .rtable file that represents the relative abundance of each construct in your samples. The DESeq program can now be run within R to compute the differential expression of each construct, which will identify the primary hits of the screen.

Note: analyzing more than two conditions at the same time is beyond the scope of this guide; please refer to the DESeq documentation for more advanced types of analysis.

The following is a convenience script for automatically running DESeq in R. Please review the DESeq documentation for explanations of each of these options and how to modify this data analysis.

```
# Determine significant Differential Expression hits
args = commandArgs(trailingOnly=TRUE)

hitCounts = args[1]
fdr = as.numeric(args[2])

library(DESeq)
counts = read.table(hitCounts, sep=",")

# CHANGE THE FOLLOWING LINE TO MATCH YOUR CONDITIONS.
conds = c("reference", "reference", "experiment", "experiment")

cds = newCountDataSet(counts, conds)
cds = estimateSizeFactors(cds)
cds = estimateDispersions(cds, fitType="local", sharingMode='fit-only')
DEResult = nbinomTest(cds, "reference", "experiment")

DEResult = DEResult[order(DEResult$padj),]
significantHits = DEResult[DEResult$padj < fdr,]
significantHits = significantHits[order(significantHits$padj),]

write.table(DEResult, paste(hitCounts, ".results", sep=""), sep=",")
write.table(significantHits, paste(hitCounts,".sig.results", sep=""), sep=",")

# This function plots the MA-plot
plotDE <- function(DEResult, fdr){
    plot(DEResult$baseMean, DEResult$log2FoldChange, cex=.3, yaxt="n", xaxt="n",
      xlab="Mean of Normalized Counts",
      ylab=as.expression(bquote(Log[2]~"Fold Change")), log="x", pch=20,
          col=ifelse(DEResult$padj < fdr & DEResult$log2FoldChange > 0, "red",
                     ifelse(DEResult$padj < fdr & DEResult$log2FoldChange < 0,
                            "blue", "black"))
          )
    axis(1, las=0)
    axis(2, las=2)
}

pdf(paste(hitCounts, ".pdf", sep=""))
plotDE(DEResult ,fdr)
dev.off()
```

Download the script to your local directory from: dharmacon.gelifesciences.com/uploadedFiles/Resources/runDESeq.r. You may have to modify the following condition to reflect the number of replicates you used to create the .rtable:

```
conds = c("reference", "reference", "experiment", "experiment")
```

The above statement will work for two replicates of each condition. The labels for the conditions are arbitrary, but each replicate must be labeled identically and coincide with the sample order in your .rtable file. For instance, if there are three replicates for each condition, you would replace that line with:

```
conds = c("reference", "reference", "reference", "experiment", "experiment", "experiment")
```

Now that you saved and updated the runDESeq.r script, you can run the following command using the .rtable file you generated earlier:

```
R CMD BATCH --vanilla '--args combinedCounts.rtable .05' -f runDESeq.r
```

combinedCounts.rtable is the .rtable file created in the previous steps. The second argument, in this case .05, represents the desired False Discovery Rate as estimated by Benjamini-Hochberg multiple-test correction. You can change the False Discovery Rate to match your tolerance for false positives. The default of .05 is very conservative. Any construct with an adjusted p-value lower or equal to this value will be written to a separate file which will represent all the significant hits based on this threshold.

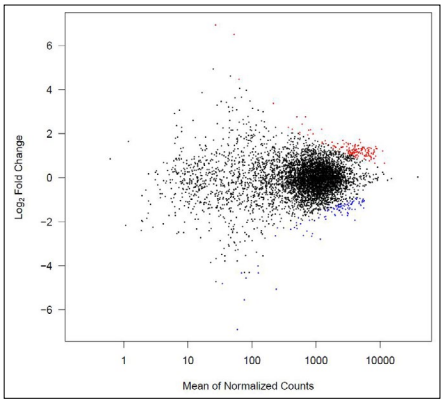
The output of DESeq will write two text files and a PDF file. The two text files will have the extension .results. For example, given the input file name of combinedCounts.rtable, output files with the names combinedCounts.rtable.results and combinedCounts.rtable.sig.results will be generated.

The file with the extension .sig.results will contain only the constructs that have significant differential expression values based on the False Discovery Rate you provide to the runDESeq.r script. Using the unique identifier assigned to each construct in the .sig.results file, you can look up the targeted genes (in the case of an shRNA or sgRNA screen) or expressed microRNAs (in the case of a shMIMIC microRNA screen) in the Excel-based data file provided with each Dharmacon pooled lentiviral library shipment. The .results file will have the output of all the differential expression tests between the two conditions including the significant ones. Please refer to the DESeq manual for interpreting the .results files. Here is an example of how the files appear:

```
"id","baseMean","baseMeanA","baseMeanB","foldChange","log2FoldChange","pval","padj"
"84","Reference_153307",8621.81673939894,4851.58351495858,12392.0499638393,2.55422789809382,
1.35288725361601,1.69438575931572e-10,4.09913480062145e-07
"855","Reference_62931",4039.29690413751,6394.35620343482,1684.23760484019,0.263394398318861,
-1.92470343115905,2.19871346359098e-10,4.09913480062145e-07
"2424","Reference_29460",8550.22640315312,4659.75446525429,12440.6983410519,2.66981842794865,
1.41674162880218,1.65203281409974e-10,4.09913480062145e-07
"1411","Reference_71792",4765.36757636804,2181.8922364658,7348.84291627028,3.36810535069039,
1.75193726504965,3.81130491962204e-10,5.02422910367094e-07
```

- The first column is a unique integer assigned to each construct by DESeq and it does not have a corresponding label in the header.
- "id" is the reference ID, a unique identifier for a given shRNA, microRNA or sgRNA construct.
- "baseMean" is the normalized mean of the normalized counts for both conditions.
- "baseMeanA" is the mean of the normalized counts of the reference sample.
- "baseMeanB" is the mean of the normalized counts of the experimental sample.
- "foldChange" is the fold-change difference in construct abundance between two samples.
- "log2FoldChange is Log2 of the fold-change.
- "pval" is the unadjusted p-value of the differential expression statistical test.
- "padj" is the Benjamini-Hochberg multiple test corrected p-value.

An MA plot of the analyzed data will be generated as a PDF file and can be viewed with any PDF viewer (e.g., Adobe Acrobat™, gv (Ghostview™), Evince™, etc.). The following is an example MA plot:



The red points represent constructs that were significantly enriched based on the False Discovery Rate parameter, while the blue points represent constructs that were significantly depleted based on the False Discovery Rate parameter.

8 Conclusions

At the end of this analysis you should have a list of constructs that cause a significant change in the population of cells between one condition and another. Refer to the relevant Dharmacon pooled lentiviral library technical manual for hit follow up recommendations.

Orders can be placed at:
gelifesciences.com/dharmacon

Customer Support: cs.dharmacon@ge.com
Technical Support: ts.dharmacon@ge.com or
1.800.235.9880; 303.604.9499 if you have any questions.

V2-0715