A collection of scripts to query DepMap data base

Data and queries

We use five data files from DepMap for mutations, copy number, mRNA expression, CRISPR effect, shRNA effect for *genes* and *cell lines*. These files are updated on quarterly basis, so we start with a directory with file versions we want to use.

Our scripts create two types of tables:

1. For a set G of genes and set C of cell lines, rows for cell lines, columns for genes and "results" as entries.
    a. For copy number, mRNA expression, CRISPR effect, shRNA effect the results are as in the original data files but abbreviated to 7 characters – higher precision is not needed
    b. For mutations, counts of deleterious and non-deleterious mutations. More informative stat would be loss-of-function and no-loss-of-function mutation, but that seems to require a different model for each gene.
2. For a set G of genes and set C of cell lines report selected info on all mutations. That allows to work on the loss-of-function issue
3. We output numbers abbreviated to 7 characters and use "NA" when data is missing, for mutations we output "_" separated pair of integers (one or two digits)

This collection of scripts uses awk: efficiency is sufficient, the results are computed 1-2 minutes and awk is very stable. The code runs both in MacOS (always installed) and in Linux using gawk (easy to install).
Issue: in Linux, we need to use zcat instead of gzcat, test Linux version.

Preliminary processing

DepMap releases updates of their data files every quarter, to be up-to-date one should process those files. For our queries, two files have to be converted. Our scripts use large data files in compressed form, `filename.gz` (`sample_info` file is small).

1. Converting a data file from .csv to .tab
   If fields in .csv file are enclosed in quotes and contain command, using awk to process them requires extra code, so it is more efficient (time, code size) to convert such file to tab separated.
   `awk -f csv2tab filename.csv > filename.tab`
   We used it only for `sample_info` where quotes and commas in fields.
2. Changing format of shRNA effect files.

in DepMap, files for CRISPR effect, gene copy number, and gene expression have
rows for cell lines with DepMap IDs (ACH-0…)
columns for genes lines
but the file for siRNA effect, "D2_combined_gene_dep_scores.csv, has
rows for genes
columns for cell lines with CCLE names (name "_" lineage)
For uniformity, we convert D2_… file to the first format

```
awk -f transpose_D2.awk D2_combined_gene_dep_scores.csv >
D2_transposed.csv
```

Issue. One can make queries faster and storage smaller by reducing the number of digits stored, large throughput measurements do not require more than 7 characters given the actual accuracy of measurements. For now, we use 7 character numbers in the output.
Issue. We should convert CCLE_expression.csv from logarithmic entries to actual values which already have limited precision, and modify the query script that performs that conversions.

Queries

For mRNA expression, CRISPR effect, shRNA effect and copy number we want to create tables that we use in biology projects. Thus we have a script that extracts table for

a. a single gene, or a file with a list of genes or all genes
b. a file with a list of cell lines
c. a type of data (one of four)

In the case of lists, we want rows and columns ordered exactly like in the lists, sometimes it makes a significant difference.

Finding identifiers of cell lines

The first task is to "identify the identifiers". There identifiers used by biologists have aliases and inconsistencies in capitalization and the use of non-alphanumeric characters. In big majority of cases so-called "stripped name", with non-alphanumeric characters removed and all alphabetic characters converted to upper case, properly identifies the cell line, sometimes we need to check "aliases" and in few cases we have multiple possibilities. Some cell lines with published data are not present in DepMap data base.

Therefore given a list of identifiers L, one cell line name per line, each in the 1st column, we run

```
awk -f FIND.awk L > L_samples.tsv
```

The output has DepMap identifier in 1st column, input identifier in 2nd column, and "primary disease" in 3rd column. For identifiers with no matches or multiple matches we have a report at the end.

Issue. Sometimes gene names are not HUGO but aliases, and not all genes have data. We may have a simple script or even simpler, a table, that tells which gene names have data, and we can check gene aliases if we have a name with no data. More ambitious is to search for HUGO symbols that correspond to an alias.

Mutation query

`awk -v sample=sfil -v genes=gfil -f report_mut.awk > output.tsv`

sfil is the name of a file in the format produced by `FIND.awk`,
gfil is the name of a file with gene name in the first column, lines starting with `!` are ignored
if we want one gene only, we can use `-v gene=gname`
Issue. Allow *gname* to be a comma separated list of genes, practical when we query several genes.

Other queries

`awk -v samples -v genes -v d=datatype -f ccle_2.awk > output.tsv`

datatype is
`CRI` for CRISPR gene effect
`RNA` for shRNA gene effect
`exp` for mRNA gene expression
null for gene copy number (in this case, -v d= can be omitted)