Week 1 - CCLE Annotations

Your name here

June 12th, 2020

Introduction

The **Cancer Cell Line Encyclopedia** is a ***massive*** collection of data that covers more than 1400 cancer cell lines (and still growing!). It contains various types of data, of which only some are listed below:

- RNAseq (gene expression profile)
- RPPA (reverse-phase protein arrays - protein production profile)
- Fusion/translocation events (chromosomal/gene level rearrangements)
- miRNA (micro-RNA expression)
- Mutations (germline - present in all cells, heritable; somatic - not heritable)
- Copy number (chromosome/gene level amplifications and deletions)
- RRBS (reduced-representation bisulfite sequencing - methylation profiles)

The human genome has somewhere between 17,000-30,000 genes (depending on who you ask), so some of these datasets can have around 1,400 x 17,000 different points of data!

How many points of data is that exactly? Let's calculate it! Remember that we can treat R like a calculator:

```r
number_of_celllines <- 1400 # assign the value 1400 to the variable "number_of_celllines"
number_of_genes <- 17000 # do the same thing for 17000 to "number_of_genes"

number_of_datapoints <- number_of_genes * number_of_celllines # create a new variable called number_of_
number_of_datapoints # show our result!
```

```
## [1] 23800000
```

So a single chunk of data can have approximately 23800000 different values.

That's a LOT of data! But before we even begin to dive into the actual data itself, we need to learn information about the data.

Getting data about the data

To start off, we're going to take a look at information about the cell lines in CCLE. Each of the cell lines listed here are used in the wet lab. For example, you may have heard the story of HeLa cells, which were obtained from a woman named Henrietta Lacks. If you have some free time, then I highly recommend reading The Immortal Life of Henrietta Lacks!

Like HeLa cells, each of the cell lines in CCLE was obtained from real patients, isolated from tumor biopsies, and immortalized for research use. Each of has a story to tell about how cancer arose, how it grew, and hopefully, how we can stop it from growing.

The first thing we want to do is find the file that contains the meta-data we want for the Cancer Cell Line Encyclopedia. I have placed that file for you in your working directory (`Cell_lines_annotations_20181226.txt`) that contains the data we want to look at. To start, let's load this file and see what kind of data we're working with.

```r
CCLE_metadata <- read.delim("Cell_lines_annotations_20181226.txt") # use the read.delim function to rea

nrow(CCLE_metadata) # how many rows does our data have?
ncol(CCLE_metadata) # how many columns does our data have?

CCLE_metadata[1:5,1:10] # what does our data actually look like?
# Here, I'm getting the first 5 rows (1:5), and the first 10 columns (1:10), just to see - try increasi
```

CCLE_ID	depMapID	Name	Pathology	Site_Primary
DMS53_LUNG	ACH-000698	DMS 53	primary	lung
SW1116_LARGE_INTESTINE	ACH-000489	SW1116	primary	large_intestine
NCIH1694_LUNG	ACH-000431	NCI-H1694	metastasis	lung
P3HR1_HAEMATOPOIETIC_AND_LYMPHOID_TISSUE	ACH-000707	P3HR-1	metastasis	haematopoietic_
HUT78_HAEMATOPOIETIC_AND_LYMPHOID_TISSUE	ACH-000509	HuT 78	primary	haematopoietic_

So according to this file, there are 1,461 rows! Each row should correspond to a cell line, and we can see that the first 3 columns show 3 different naming or ID conventions for each cell line. The first is the CCLE ID, which just takes the name of the cell line and stitches it to the type of tissue it came from. For example, A549_LUNG is a lung cancer cell line that is conventionally called A549.

For convenience, the `type` is also present in it's own column, and may have additional details in the `type_refined` column.

(P.S., you can also find HELA_CERVIX, and the data associated with the HeLa cancer cell line)

When we want to communicate our findings to the broader public, we want to use the conventional names like "A549" and "HELA", instead of "A549_LUNG" or "HELA_CERVIX". We can use looping, either by creating a `for` loop, or by using the `-apply` functions on the CCLE ID column to isolate the name we want to use.

Let's try first with a `for` loop:

```r
common_names <- character() # create a new empty vector to store the new names in
for(id in CCLE_metadata$CCLE_ID){
  split_id <- strsplit(x = id, split = "_") # This is the strsplit function, which splits strings by a 

  # Note that strsplit returns a list of length 1, containing a vector of the string split by whatever 
  # To access the common cell line name (which is the 1st word), we need to take the first element of t
  common_names[id] <- split_id[[1]][1] # save that element as the "id"th element of common_names.
}

common_names[1:5] # Check to see if these look correct
```

```
##                               DMS53_LUNG
##                                  "DMS53"
##                   SW1116_LARGE_INTESTINE
##                                 "SW1116"
##                            NCIH1694_LUNG
##                               "NCIH1694"
## P3HR1_HAEMATOPOIETIC_AND_LYMPHOID_TISSUE
##                                  "P3HR1"
## HUT78_HAEMATOPOIETIC_AND_LYMPHOID_TISSUE
##                                  "HUT78"
```

HOT TIP: Remember that we loaded the CCLE metadata as a data frame. This means we can access a column of the data frame using the `$` operator - so for accessing the `type` column, I wrote `CCLE_metadata$type`.

We can also access columns using the `[]`, so I could have also written `CCLE_metadata[,"type"]`. I could also access a specific column number, by writing `CCLE_metadata[,5]`.

That seemed to work - now we have a vector of the common names to use when we talk to bench biologists! Okay, now let's try again, this time with the -apply functions.

```
# Using the apply function, we can isolate each row or column at a time and apply a function to its con
# Note that MARGIN = 1 means we're looping over rows - if we set MARGIN = 2, we could loop over columns
# We also define a function to apply over the contents of each row, which I put into a variable I named
common_ids_lapply <- apply(X = CCLE_metadata, MARGIN = 1, FUN = function(row_x){
  id <- row_x["CCLE_ID"] # Get the CCLE_ID of row_x
  split_id <- strsplit(id, split = "_") # Use the same strsplit function as above.
  return(split_id[[1]][1]) # Return the first element of the first element (see above).
})

# Let's try again using the sapply function, which loops over a vector. To this, we need to give just t

common_ids_sapply <- sapply(X = CCLE_metadata$CCLE_ID, USE.NAMES = F, FUN = function(id){
  split_id <- strsplit(id, split = "_") # Use the same strsplit function as above.
  return(split_id[[1]][1]) # Return the first element of the first element (see above).
})
# Side note: I set "USE.NAMES" to FALSE to compare to "apply" which does not use names, but we could se

all.equal(common_ids_lapply, common_ids_sapply)
```

```
## [1] TRUE
```

There's a lot of other very interesting data here. Let's take a look at some of the column names to get a sense of what else we can find out about these cell lines.

```
colnames(x = CCLE_metadata)
```

Column names of CCLE metadata
CCLE_ID
depMapID
Name
Pathology
Site_Primary
Site_Subtype1
Site_Subtype2
Site_Subtype3
Histology
Hist_Subtype1
Hist_Subtype2
Hist_Subtype3
Gender
Life_Stage
Age
Race
Geo_Loc
inferred_ethnicity
Site_Of_Finding
Disease
Annotation_Source
Original.Source.of.Cell.Line
Characteristics

Column names of CCLE metadata
Growth.Medium
Supplements
Freezing.Medium
Doubling.Time.from.Vendor
Doubling.Time.Calculated.hrs
type
type_refined
PATHOLOGIST_ANNOTATION
mutRate
tcga_code

As background, the `Pathology` (and `site_subtypes`) and `Histology` (and `Hist_subtypes`) are annotations provided by the pathologist after they took the patient's tumor out. We also have information about each patient, including:

- `Gender`
- `Age`
- `Race`
- `Geo_Loc` (where was the patient sample obtained?)
- `inferred_ethnicity` (based on genetic analysis of the cell line)

There's also information about how the cells were grown in the lab. This includes:

- `Original.Source.of.Cell.Line` (where did the CCLE get their stock of cell line from?)
- `Characteristics` (is it an adherent cell line that sticks to the plate, or does it grow suspended in fluid like a blood cell?)
- `Growth.Medium` (what media does the cell grow in?)
- `Supplements` (what additional nutrients does the cell need to grow?)
- `Freezing.Medium` (what media was the cell stored in?)
- `Doubling.Time.From.Vendor` (according to the source, how long does it take for the cells to replicate?)
- `Doubling.Time.Calculated.Hours` (how many hours did the cells actually take to double?)

Diving into the metadata

Now that we have a pretty good understanding of what kinds of data we have, let's take a look at the data itself.

The first thing we should do is understand how some of the values look. `R` has some useful functions for summarizing data, some of which I'm going to demonstrate here:

The `table` function gives us a frequency table for a categorical variable (how many times does each value occur in the data?)

```r
# First, let's look at the different tissue/tumor types we're working with.
table(CCLE_metadata$type)
```

type	freq
AML	37
B-cell_ALL	12
bile_duct	8
breast	60
chondrosarcoma	4
CML	15

type	freq
colorectal	63
endometrium	28
esophagus	26
Ewings_Sarcoma	12
giant_cell_tumour	3
glioma	65
kidney	37
leukemia_other	5
liver	28
lung_NSC	135
lung_small_cell	53
lymphoma_Burkitt	11
lymphoma_DLBCL	18
lymphoma_Hodgkin	13
lymphoma_other	28
medulloblastoma	4
melanoma	63
meningioma	3
mesothelioma	11
multiple_myeloma	29
neuroblastoma	17
osteosarcoma	10
other	4
ovary	55
pancreas	46
prostate	8
soft_tissue	20
stomach	39
T-cell_ALL	16
thyroid	12
upper_aerodigestive	33
urinary_tract	28

We can also use the `table` function to compare two variables - so if I wanted to look at both gender and ethnicity, I could do the following:

```
table(CCLE_metadata$Gender, CCLE_metadata$inferred_ethnicity)
```

	African_american	Asian	Caucasian
	7	56	73
female	29	102	268
male	25	148	329
null	0	0	0

Note that there are blank values in `Gender` (which is why there's an extra row). We'll definitely talk more about this on Friday, but for now, I hope you see how useful the `table` function can be!

Another useful function is `summary`, which can give us important summary statistics about numeric variables. One interesting numerical variable here is mutation rate (or `mutRate`). See below for the output of `summary`

```r
summary(object = CCLE_metadata$mutRate)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's 
##   53.25  102.92  134.96  192.92  178.86 3119.62     497
```

This means that, on average, the mutation rate for each cell line is 192.92. Don't worry about exactly what that means just yet, we'll cover it on Friday!

One important thing to see is the number of `NA` values. These are values that are missing from the data, and are handled specifically by `R` as missing values. When we calculate the mean and median by themselves in R, we need to make sure we omit NA values - otherwise, it's like multiplying by 0, all the other numbers end up becoming NA also. To see what I *mean*, let's try just taking the mean without omitting NA, using the `mean` function.

```r
mean(CCLE_metadata$mutRate) # welp, that didn't work - should you take the mean without removing NA? na
```

```
## [1] NA
```

```r
mean(CCLE_metadata$mutRate, na.rm = T) # na.rm means "remove NAs for this calculation" - much better!
```

```
## [1] 192.9194
```

We can also use other summary statistics - for example, `sd` gives standard deviation, `var` gives variance, and there are others that we'll learn later on.

HOT TIP: If you don't already know, you can pull up the help page for any function by putting `?` before the function and running it. For example, `?mean` pulls up the help page for the mean function. You can read here what other arguments to functions are (like `na.rm`) and what their default values are (`na.rm` by default is `FALSE`, so NA values are NOT removed when you run `mean` by itself).

Narrowing down to one type of cancer

Just as an example, let's say we wanted to learn more about breast cancer. We would need to take a subset of the data. So let's do that here:

```r
is_breast_cell <- CCLE_metadata$type == "breast" # if the type is breast, give me TRUE. otherwise, give
which_is_breast_cell <- which(is_breast_cell) # "which" is a function that gives the index of each TRUE

which_is_breast_cell
```

```
##  [1]   12   55   65   66   71   81   91   92  101  173  225  259  260  278  292
## [16]  303  304  310  311  320  327  330  331  336  356  548  551  595  599  613
## [31]  630  645  680  684  728  756  757  761  764  766  812  860  880  892  897
## [46]  898  899  911  941  954  955  956  959  964  966  984 1008 1028 1035 1044
```

If that wasn't clear, what we want to do is only identify which cell lines have a `type` that is exactly "breast", so we use the `==` boolean equality comparison to check if each value in the `type` column is "breast" or not. If it is, we get a TRUE, and if it's not, we get a FALSE. That gives us a single logical vector containing TRUEs and FALSEs for each value in `type`.

Then, we use `which` to give us the index (or position) of each TRUE. From this, we know that the 12th value is a TRUE, which means that row 12 in `CCLE_metadata` data frame is a breast cancer cell line. We can check that here:

```r
CCLE_metadata[12,] # give us the 12th row
```

	CCLE_ID	depMapID	Name	Pathology	Site_Primary	Site_Subtype1	Site_Subtype2	Site
12	HCC2157_BREAST	ACH-000691	HCC2157	primary	breast	NS	NS	NS

Yep! The 12th cell line is `HCC2157_BREAST`, which is indeed a breast cancer cell line.

Okay, so now that we have all of the indices (plural of index, not indexes!) of the breast cell lines, I'm curious - which breast cancer cell line has the highest mutation rate?

```
CCLE_metadata_breast <- CCLE_metadata[which_is_breast_cell,] # make a new subsetted dataframe that only

max(CCLE_metadata_breast$mutRate, na.rm = T) # The highest mutation rate - don't forget to remove NAs!
```

```
## [1] 486.8665
```

```
which_breast_highest_mutRate <- which.max(CCLE_metadata_breast$mutRate) # Get the index using the which

CCLE_metadata_breast$CCLE_ID[which_breast_highest_mutRate] # Get the cell line ID with the highest muta
```

```
## [1] "HCC1569_BREAST"
```

QUESTIONS

Create a new R chunk or click the "Insert -> R" near the top right of the script-writing panel, and write code to solve the following questions.

1. Now that we know how many cell lines and columns there are, can you calculate the number of data points in the metadata file?

Hint: Remember that R is a calculator!

2. A software was used to infer the ethnicity of the cell line from genomic data. Which ethnicities are most/least represented in CCLE? Compare that to the reported race of the patient. Can you think about what the consequences of this might be? What about the software that is used to infer ethnicity?

Hint: Use the `table` function to answer this question.

3. Cancer is often thought of as a disease of aging, but in truth, it affects people of all ages. What are some of the summary statistics about the ages of the patients from whom the CCLE was derived? BONUS: Can you identify which cancers predominantly affect people under age 18?

Hint: Use the `summary` function to answer this question.

4. Growing cells in a dish requires that you understand how frequently the cell population doubles. Cells that fully occupy the space of the plate need to be "passaged", or transferred from one plate to another to allow the cells more room to grow. The "doubling time" has been recorded for all CCLE cell lines, and we can use it as a proxy for how fast the tumor might have been growing in the patient. What is the average doubling time (use calculated hours) for colorectal cancer cell lines? Which cell line has the fastest doubling time?

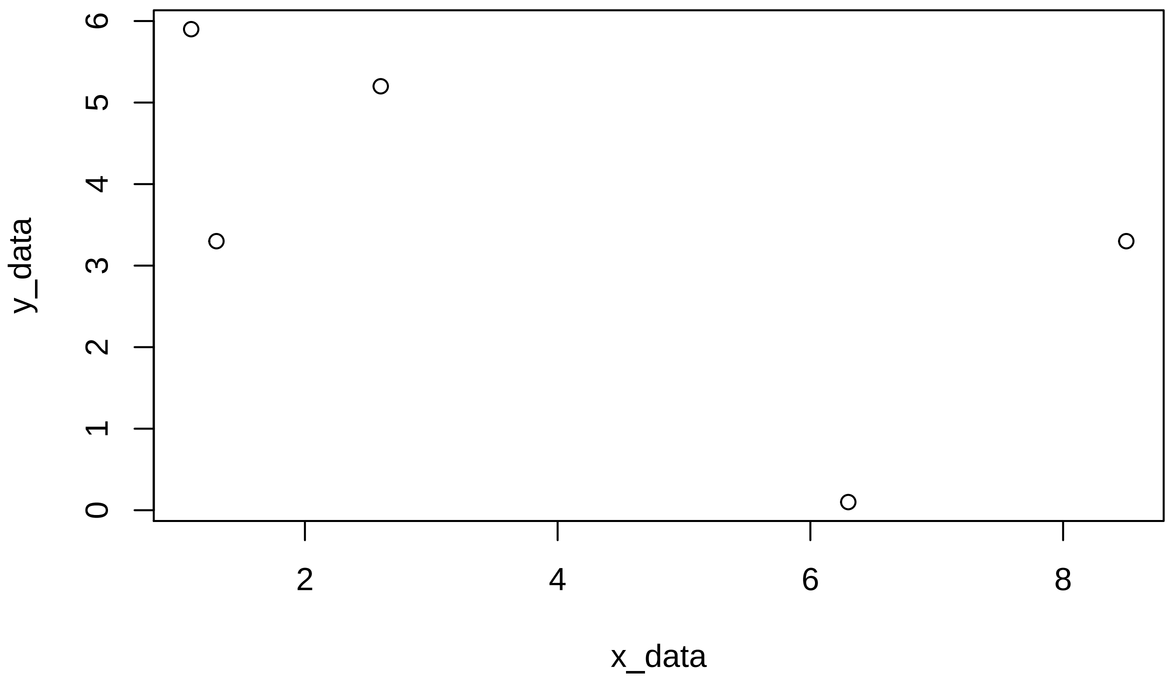
Hint: You'll need to subset for colorectal cancer cell lines.

5. Create a plot showing the relationship between doubling time and mutation rate. Extra credit for anyone who can use the `ggplot2` package (load using `library(ggplot2)`) to create their plot, and/or who can color their plot by cancer type. BONUS: Are these variables correlated?

Hint: The base `plot` function works by taking in data for the x-axis and y-axis as follows:

```
x_data <- c(1.3, 2.6, 6.3, 8.5, 1.1)
y_data <- c(3.3, 5.2, 0.1, 3.3, 5.9)

plot(x_data, y_data)
```



Extra hint: The `qplot` function of ggplot2 also works similarly!

BONUS hint: To learn more about the correlation function in `R`, try typing `?cor` to read the help page.