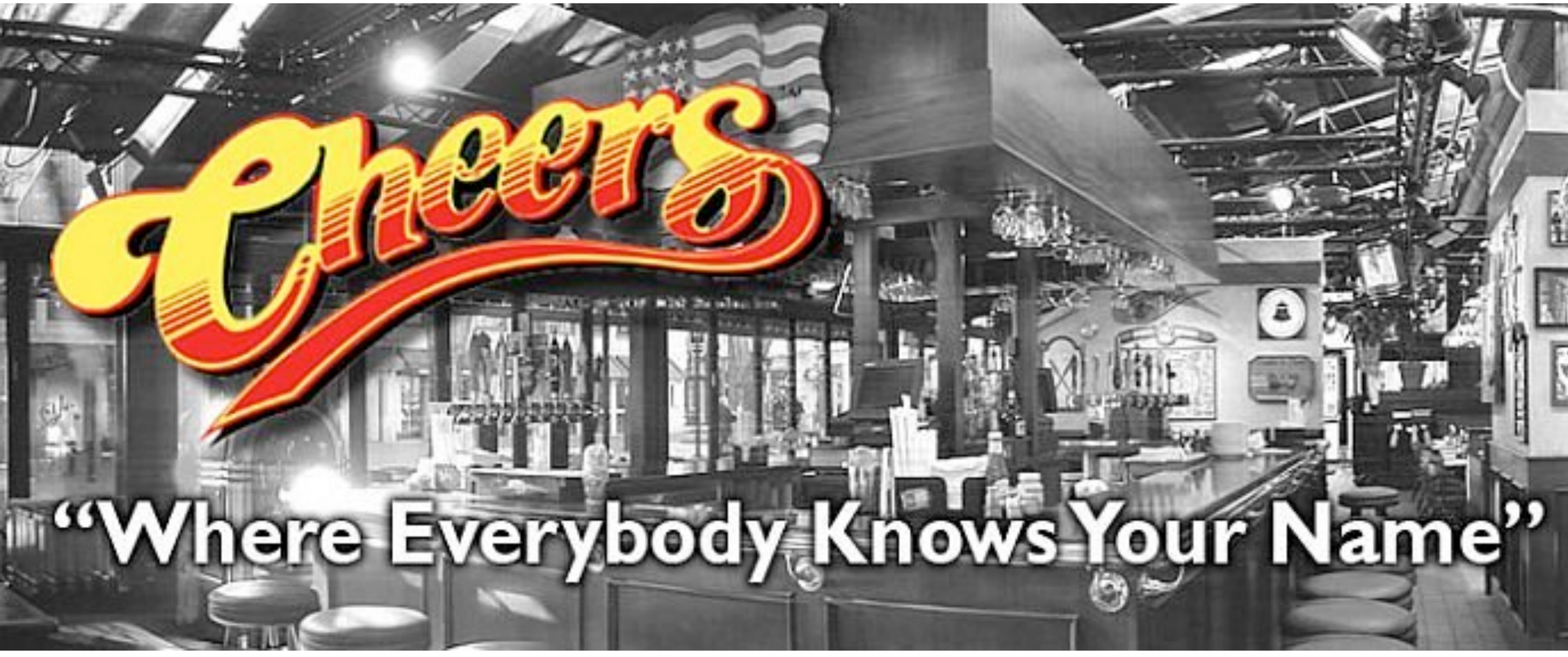
naming things

prepared by Jenny Bryan for

Reproducible Science Workshop

Names matter



NO

myabstract.docx
Joe's Filenames Use Spaces and Punctuation.xlsx
figure 1.png
fig 2.png
JW7d^(2sl@deletethisandyourcareerisoverWx2*.txt

YES

2014-06-08_abstract-for-sla.docx
joes-filenames-are-getting-better.xlsx
fig01_scatterplot-talk-length-vs-interest.png
fig02_histogram-talk-attendance.png
1986-01-28_raw-data-from-challenger-o-rings.txt

three principles for (file) names

machine readable

human readable

plays well with default ordering

awesome file names :)

2013-06-26_BRAFWTNEGASSAY_Plasmid-Cellline-100-1MutantFraction_H01.csv
2013-06-26_BRAFWTNEGASSAY_Plasmid-Cellline-100-1MutantFraction_H02.csv
2013-06-26_BRAFWTNEGASSAY_Plasmid-Cellline-100-1MutantFraction_H03.csv
2013-06-26_BRAFWTNEGASSAY_Plasmid-Cellline-100-1MutantFraction_platefile.csv
2014-02-26_BRAFWTNEGASSAY_FFPEDNA-CRC-1-41_A01.csv
2014-02-26_BRAFWTNEGASSAY_FFPEDNA-CRC-1-41_A02.csv
2014-02-26_BRAFWTNEGASSAY_FFPEDNA-CRC-1-41_A03.csv
2014-02-26_BRAFWTNEGASSAY_FFPEDNA-CRC-1-41_A04.csv

“machine readable”

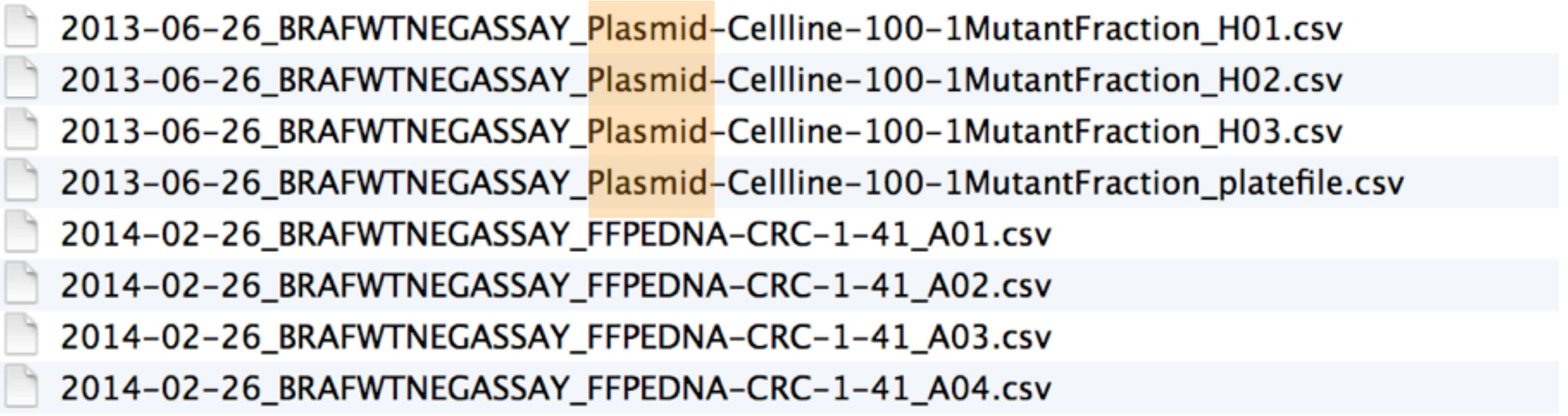
regular expression and globbing friendly

- avoid spaces, punctuation, accented characters, case sensitivity

easy to compute on

- deliberate use of delimiters

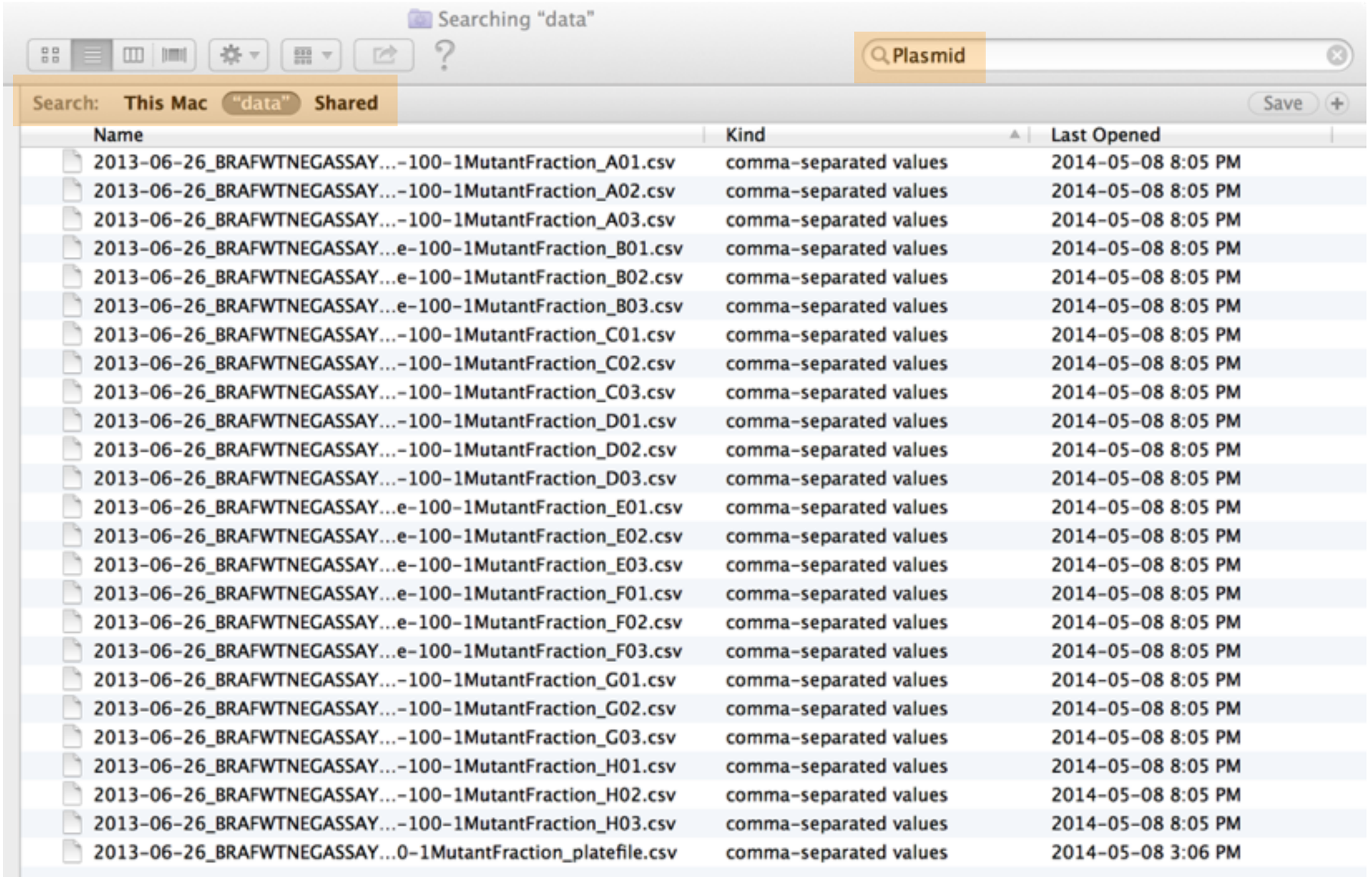
Excerpt of complete file listing:



Example of **globbing** to narrow file listing:

```
Jennifers-MacBook-Pro-3:2014-03-21 jenny$ ls *Plasmid*
2013-06-26_BRAFWTNEGASSAY_Plasmid-Cellline-100-1MutantFraction_A01.csv
2013-06-26_BRAFWTNEGASSAY_Plasmid-Cellline-100-1MutantFraction_A02.csv
2013-06-26_BRAFWTNEGASSAY_Plasmid-Cellline-100-1MutantFraction_A03.csv
2013-06-26_BRAFWTNEGASSAY_Plasmid-Cellline-100-1MutantFraction_B01.csv
....
2013-06-26_BRAFWTNEGASSAY_Plasmid-Cellline-100-1MutantFraction_H03.csv
2013-06-26_BRAFWTNEGASSAY_Plasmid-Cellline-100-1MutantFraction_platefile.csv
```

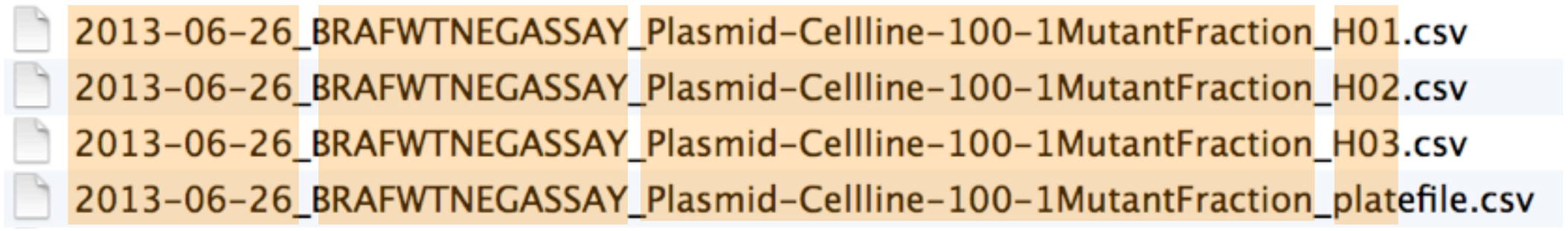
Same using Mac OS Finder search facilities:



Same using R's ability to narrow file list by regex:

```
> list.files(pattern = "Plasmid") %>% head
[1] "2013-06-26_BRAFWTNEGASSAY_Plasmid-Cellline-100-1MutantFraction_A01.csv"
[2] "2013-06-26_BRAFWTNEGASSAY_Plasmid-Cellline-100-1MutantFraction_A02.csv"
[3] "2013-06-26_BRAFWTNEGASSAY_Plasmid-Cellline-100-1MutantFraction_A03.csv"
[4] "2013-06-26_BRAFWTNEGASSAY_Plasmid-Cellline-100-1MutantFraction_B01.csv"
[5] "2013-06-26_BRAFWTNEGASSAY_Plasmid-Cellline-100-1MutantFraction_B02.csv"
[6] "2013-06-26_BRAFWTNEGASSAY_Plasmid-Cellline-100-1MutantFraction_B03.csv"
```

Deliberate use of "_" and "-" allows us to recover meta-data from the filenames.

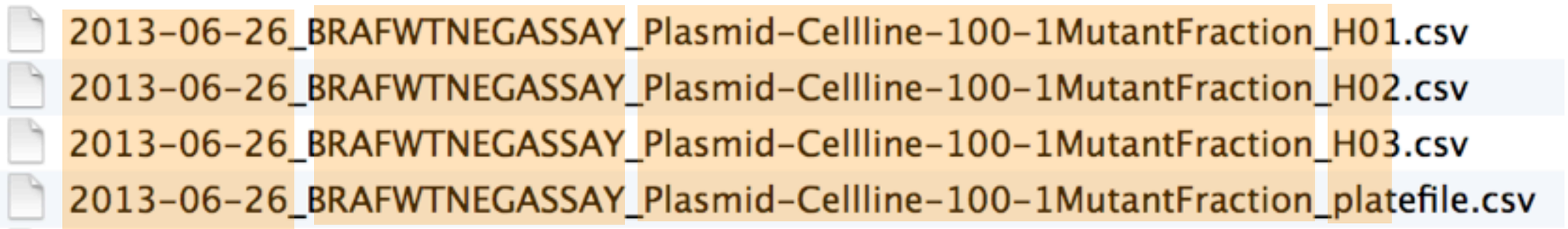



```
> flist <- list.files(pattern = "Plasmid") %>% head

> stringr::str_split_fixed(flist, "[_\\.]", 5)
     [,1]         [,2]              [,3]                                  [,4]  [,5]
[1,] "2013-06-26" "BRAFWTNEGASSAY" "Plasmid-Cellline-100-1MutantFraction" "A01" "csv"
[2,] "2013-06-26" "BRAFWTNEGASSAY" "Plasmid-Cellline-100-1MutantFraction" "A02" "csv"
[3,] "2013-06-26" "BRAFWTNEGASSAY" "Plasmid-Cellline-100-1MutantFraction" "A03" "csv"
[4,] "2013-06-26" "BRAFWTNEGASSAY" "Plasmid-Cellline-100-1MutantFraction" "B01" "csv"
[5,] "2013-06-26" "BRAFWTNEGASSAY" "Plasmid-Cellline-100-1MutantFraction" "B02" "csv"
[6,] "2013-06-26" "BRAFWTNEGASSAY" "Plasmid-Cellline-100-1MutantFraction" "B03" "csv"
```

date assay sample set well

This happens to be R but also possible in the shell, Python, etc.




```
> flist <- list.files(pattern = "Plasmid") %>% head

> stringr::str_split_fixed(flist, "[_\\.]", 5)
     [,1]         [,2]              [,3]                                [,4]  [,5]
[1,] "2013-06-26" "BRAFWTNEGASSAY" "Plasmid-Cellline-100-1MutantFraction" "A01" "csv"
[2,] "2013-06-26" "BRAFWTNEGASSAY" "Plasmid-Cellline-100-1MutantFraction" "A02" "csv"
[3,] "2013-06-26" "BRAFWTNEGASSAY" "Plasmid-Cellline-100-1MutantFraction" "A03" "csv"
[4,] "2013-06-26" "BRAFWTNEGASSAY" "Plasmid-Cellline-100-1MutantFraction" "B01" "csv"
[5,] "2013-06-26" "BRAFWTNEGASSAY" "Plasmid-Cellline-100-1MutantFraction" "B02" "csv"
[6,] "2013-06-26" "BRAFWTNEGASSAY" "Plasmid-Cellline-100-1MutantFraction" "B03" "csv"
```

“_” underscore used to delimit units of meta-data I want later

“-” hyphen used to delimit words so my eyes don’t bleed

“machine readable”

easy to search for files later

easy to narrow file lists based on names

easy to extract info from file names, e.g. by splitting

new to regular expressions and globbing? be kind to yourself and avoid

- spaces in file names
- punctuation
- accented characters
- different files named “foo” and “Foo”

“human readable”

name contains info on ***content***

connects to concept of a ***slug*** from semantic URLs

“human readable”

```
Jennifers-MacBook-Pro-3:analysis jenny$ ls -1
```

```
01_marshal-data.md                          01.md
01_marshal-data.r                           01.r
02_pre-dea-filtering.md                     02.md
02_pre-dea-filtering.r                      02.r
03_dea-with-limma-voom.md                   03.md
03_dea-with-limma-voom.r                    03.r
04_explore-dea-results.md                   04.md
04_explore-dea-results.r                    04.r
90_limma-model-term-name-fiasco.md          90.md
90_limma-model-term-name-fiasco.r           90.r
Makefile                                    Makefile
figure                                      figure
helper01_load-counts.r                      helper01.r
helper02_load-exp-des.r                     helper02.r
helper03_load-focus-statinf.r               helper03.r
helper04_extract-and-tidy.r                 helper04.r
tmp.txt                                     tmp.txt
```

Which set of file(name)s do you want at 3a.m. before a deadline?

“human readable”

```
01_marshal-data.r

02_pre-dea-filtering.r

03_dea-with-limma-voom.r

04_explore-dea-results.r
```

embrace the **slug**

```
90_limma-model-term-name-fiasco.r

helper01_load-counts.r

helper02_load-exp-des.r

helper03_load-focus-statinf.r

helper04_extract-and-tidy.r
```

“human readable”

easy to figure out what the heck something is, based on its name

“plays well with default ordering”

put something numeric first

use the ISO 8601 standard for dates

left pad other numbers with zeros

“plays well with default ordering”

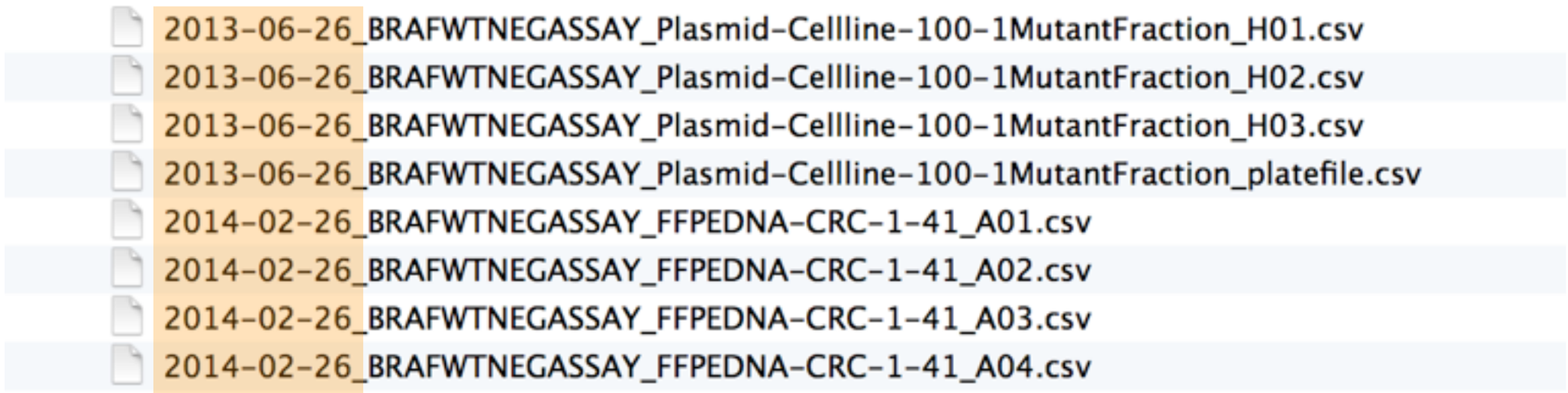
chronological order

2013-06-26_BRAFWTNEGASSAY_Plasmid-Cellline-100-1MutantFraction_H01.csv
2013-06-26_BRAFWTNEGASSAY_Plasmid-Cellline-100-1MutantFraction_H02.csv
2013-06-26_BRAFWTNEGASSAY_Plasmid-Cellline-100-1MutantFraction_H03.csv
2013-06-26_BRAFWTNEGASSAY_Plasmid-Cellline-100-1MutantFraction_platefile.csv
2014-02-26_BRAFWTNEGASSAY_FFPEDNA-CRC-1-41_A01.csv
2014-02-26_BRAFWTNEGASSAY_FFPEDNA-CRC-1-41_A02.csv
2014-02-26_BRAFWTNEGASSAY_FFPEDNA-CRC-1-41_A03.csv
2014-02-26_BRAFWTNEGASSAY_FFPEDNA-CRC-1-41_A04.csv

logical order

```
01_marshal-data.r
02_pre-dea-filtering.r
03_dea-with-limma-voom.r
04_explore-dea-results.r
90_limma-model-term-name-fiasco.r
helper01_load-counts.r
helper02_load-exp-des.r
helper03_load-focus-statinf.r
helper04_extract-and-tidy.r
```

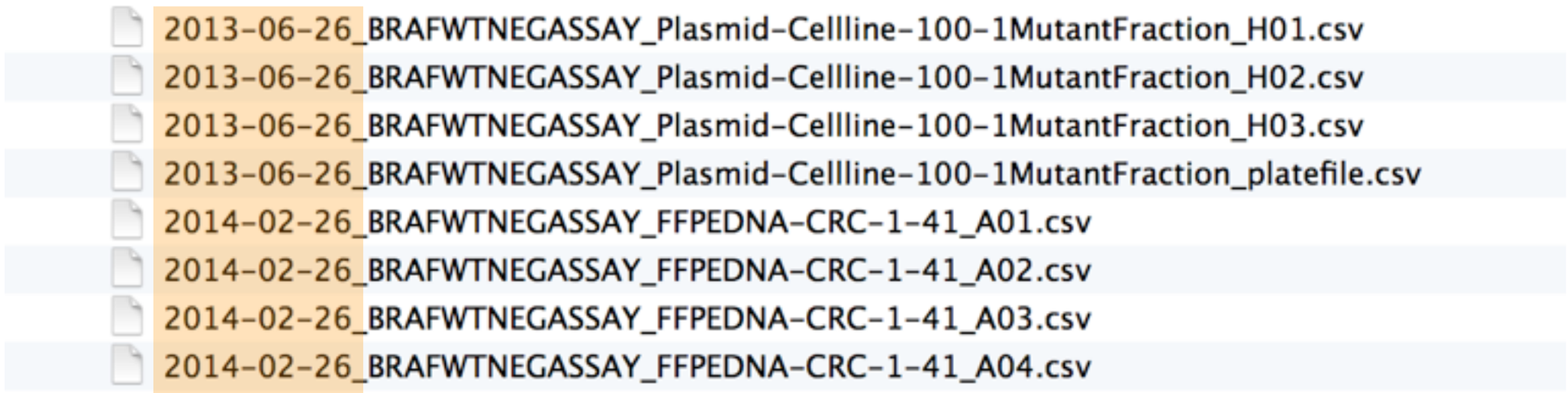
“plays well with default ordering”




```
01_marshal-data.r
02_pre-dea-filtering.r
03_dea-with-limma-voom.r
04_explore-dea-results.r
90_limma-model-term-name-fiasco.r
helper01_load-counts.r
helper02_load-exp-des.r
helper03_load-focus-statinf.r
helper04_extract-and-tidy.r
```

put something numeric first

“plays well with default ordering”



use the ISO 8601 standard for dates

YYYY-MM-DD

PUBLIC SERVICE ANNOUNCEMENT:

OUR DIFFERENT WAYS OF WRITING DATES AS NUMBERS CAN LEAD TO ONLINE CONFUSION. THAT'S WHY IN 1988 ISO SET A GLOBAL STANDARD NUMERIC DATE FORMAT.

THIS IS ***THE*** CORRECT WAY TO WRITE NUMERIC DATES:

2013-02-27

THE FOLLOWING FORMATS ARE THEREFORE DISCOURAGED:

02/27/2013 02/27/13 27/02/2013 27/02/13

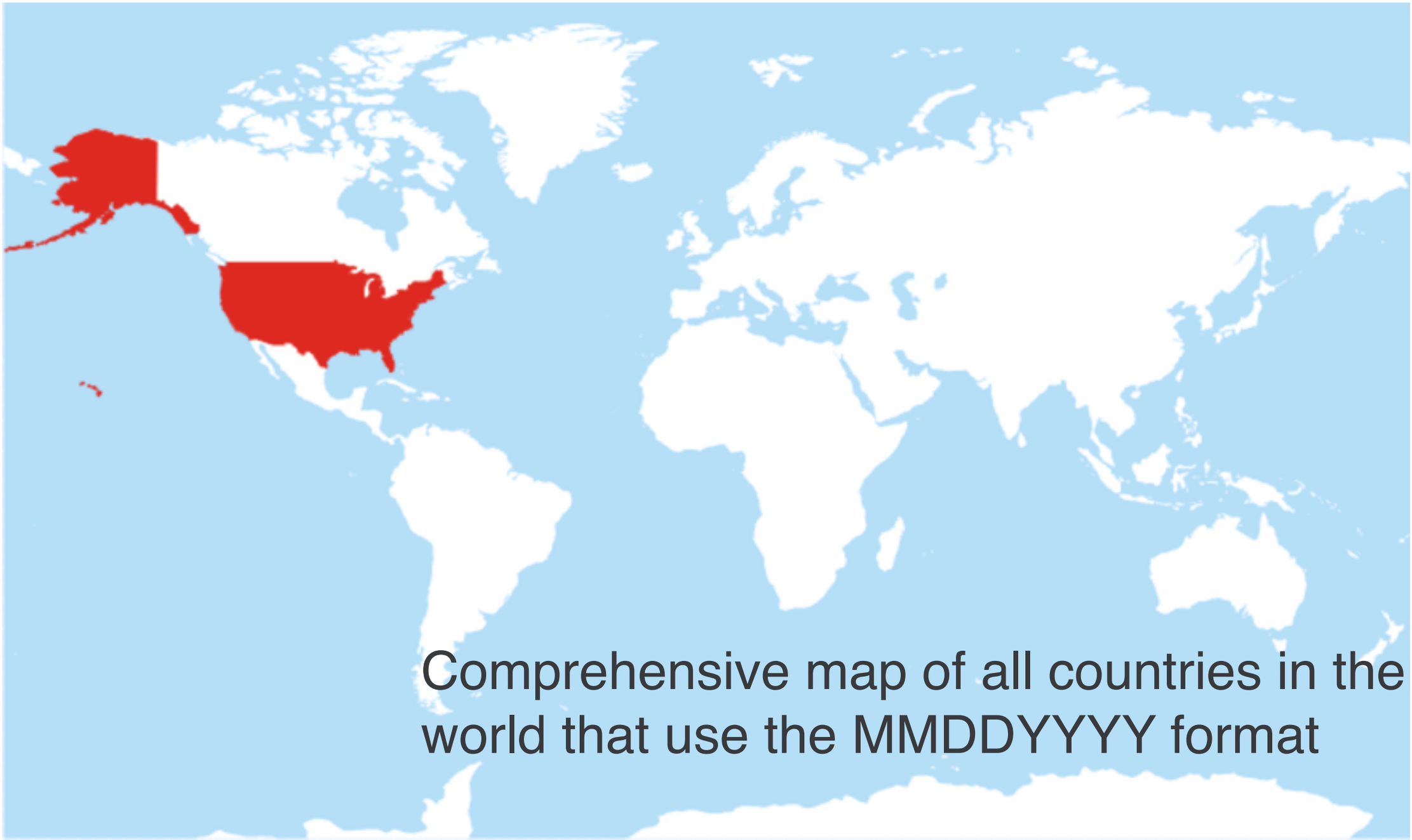
20130227 2013.02.27 27.02.13 27-02-13

27.2.13 2013. II. 27. $27\frac{1}{2}$-13 2013.158904109

MMXIII-II-XXVII MMXIII $\frac{\text{LVII}}{\text{CCCLXV}}$ 1330300800

$((3+3)\times(111+1)-1)\times 3/3-1/3^3$ ~~2013~~ Hisssss

10/11011/1101 02/27/20/13 $\overset{2}{0}\overset{3}{1}\overset{1}{2}\overset{4}{3}\overset{67}{7}\overset{5}{}\overset{}{}\overset{8}{}$



https://twitter.com/donohoe/status/597876118688026624

left pad other numbers with zeros

```
01_marshal-data.r
02_pre-dea-filtering.r
03_dea-with-limma-voom.r
04_explore-dea-results.r
90_limma-model-term-name-fiasco.r
helper01_load-counts.r
helper02_load-exp-des.r
helper03_load-focus-statinf.r
helper04_extract-and-tidy.r
```

if you don't left pad, you get this:

```
10_final-figs-for-publication.R
1_data-cleaning.R
2_fit-model.R
```

which is just sad

“plays well with default ordering”

put something numeric first

use the ISO 8601 standard for dates

left pad other numbers with zeros

three principles for (file) names

machine readable

human readable

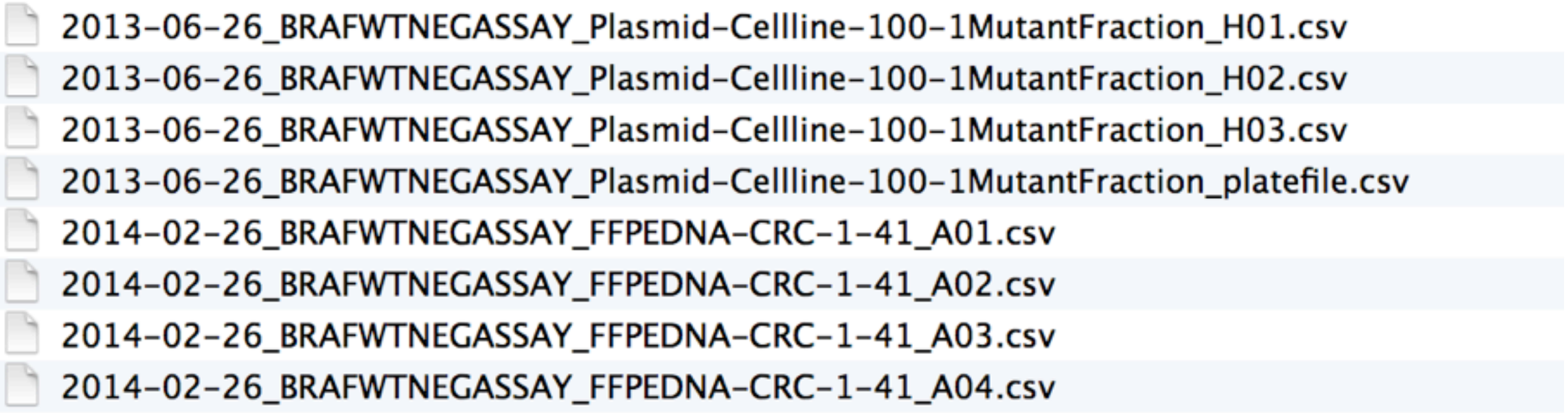
plays well with default ordering

three principles for (file) names

easy to implement NOW

payoffs accumulate as your skills evolve and projects get more complex

go forth and use awesome file names :)




```
01_marshal-data.r
02_pre-dea-filtering.r
03_dea-with-limma-voom.r
04_explore-dea-results.r
90_limma-model-term-name-fiasco.r
helper01_load-counts.r
helper02_load-exp-des.r
helper03_load-focus-statinf.r
helper04_extract-and-tidy.r
```