ELISAtools Package Vignette

Feng, Feng

2021-01-20

1. Overview

Enzyme-linked immunosorbent assays (ELISA) are probably the most frequently used assays in the biomedical field. In most ELISAs, a standard curve provides the mathematical relationship between known antigen concentrations and signal generation so that samples with unknown antigen levels can be quantified. For single use or short-term projects, lot-to-lot variability between ELISA kits is either not relevant or is manageable, but long-term projects over months or years present challenges for quality data management when multiple research ELISA kit lots are utilized. We develop this R package to do regular ELISA data analyses. Mostly importantly we also develop a mathematical solution with a sufficiently generalized approach to adjust/normalize the batch affects caused by the lot-to-lot variability between ELISA kits. The program calculates a Shift factor ("S") that can be applied retrospectively to every ELISA plate's standard curve, and the biomarker concentrations for that plate are adjusted accordingly (Feng et al. 2018).

2. Theory and Model

The four- and five-parameter logistic functions (4pl and 5pl) are good models for ELISA data. The 5pl has the form of

$$Y = a + \frac{d-a}{(1+e^{\frac{xmid-x}{scal}})^g}$$

Y is the signal intensity of assay; x is the log- transformed concentration of analytes; a, d, $xmid$, $scal$ and g are the five parameters. When g takes a value of 1, the 5pl becomes the 4pl.

To analyze data of an individual ELISA assay, either the 4pl or the 5pl is fitted based on the standard data

to estimate the parameters, and then measured ODs are reverse-looked up on the fitted equation to obtain the concentrations of analytes in unknown samples.

In case of analyzing data from multiple assays, batch effects have to be modelled and corrected in order to compare these data. Many factors, either biological or technical, could possibly impact the reliability and reproducibility of immunoassays and lead to batch effects. Here we only model and adjust the one caused by the lot-to-lot variability between ELISA kits. The model first assumes the lot-to-lot variability mainly comes from the inaccurate quantitation of concentrations of standard analytes, which can expressed as

$$c = c_0 \cdot N$$

or in a logarithmic format

$$log(c) = log(c_0) + log(N)$$

c_0 and c are the real and provided concentration of standard analytes by the manufacturer, respectively, and N is the fold difference between them. Rewrite the above log-transformed equation by replacing $log(x)$, $log(x_0)$ and $log(N)$ with x, x_0 and S

$$x_0 = x - S$$

Therefore the statistical model of the standard curves can be written as

$$Y_{ijk} = a + \frac{d-a}{(1+e^{\frac{xmid_i - x_{ij}}{scal}})^g} + \epsilon_{ijk}$$

$$i = 1, 2....n; j = 1, 2....m; k = 1, 2....l$$

m, n and l are total number of batches, standards and measurements, respectively. From the equation, we can see that the parameters, a, d, $scal$, and g are constant to all the batches. But the parameter $xmid$, the inflexion point of the function, is not and it can be expressed as

$$xmid_i = xmid + S_i$$

Again, S_i, the Shifting factor, is the log fold difference between the known concentration and the true one. As a results, the model indicates that the standard curves of the same batch differ from each other as a result of random errors of measurements, while difference between curves from different batches is ascribed to inaccurate quantitation of analyte concentrations.

To do the batch normalization/correction, the analyte concentrations in unknown samples are first estimated based on unadjusted standard curves and then the Shifting factor S of the batch is applied to obtain the final quantities,

$$(x_{ij})_{adj} = \hat{x}_{ij} - S_i$$

$\hat{x}_{ij}$ and S_i are the unadjusted concentration of analytes in the unknown sample and the shifting factor of the batch, respectively. $(x_{ij})_{adj}$ is the batch normalized/corrected concentration. One things to mention is that practically it might not be possible to know the accurate concentrations of analytes in the standard samples. Therefore, we need to designate one batch as the reference, in which the analyte concentrations of standard samples are treated as accurate, and estimate the Shifting factor S of all other batches relative to it (Feng et al. 2018).

3. A Short Example

1. Installation

Package can be installed from R CRAN

```
install.packages("ELISAtools")
```

To get the latest source code, please go to https://github.com/BULQI/ELISAtools [https://github.com/BULQI/ELISAtools]

2. Loading *ELISAtools* and importing data

After installation, the package can be loaded by

```
    library(ELISAtools)
```

```
#> Loading required package: R2HTML
```

```
#> Loading required package: stringi
```

```
#> Loading required package: minpack.lm
```

To read the data into software, we rely on 4 sets of files, design file, annotation file, standard concentration file and OD data file.

1. Standard concentration file,

   this is a table containing the concentrations for the standards. It contains only two columns, ID and Std.

ID	Std
s1	3000
s2	1500
..	..

   It doesn't matter how the users name the columns. The software will only read the first two columns. The rest will be ignored. However, it does matter how the users name the standards. The software expects the standard samples to be names as sxx, a lower case letter "s" followed by digits. The digits could be single or many. The IDs in this sample will be cross referenced by the annotation file. So please keep the IDs/names of standards in consistence.

2. Annotation file:

   This file is the map for the ELISA plate. It is a 96-well plate format (8x12 grids). It basically tells the software which OD data are for which samples when reading the plate. It contains the sample names in the plate wells. It accepts samples names with no fixed rules, but does expect the name of standards with a strict format (see above).

1	2	3	4	5	6	7	8	9	10	11
A	s1	s1	s1							
B	s2	s2	s2	T-01	T-01	T01	T-02			
..	..	..	..	..	..	..	..	..	..	..

3. OD data file:

   This is the actually OD input. It is a exported text file from the original ".sdf" file (a file format by the microplate scanner by the Molecular Devices). It contains the OD readings plus other analysis content done by the plate reader. Only the OD data will be read. The OD data are arranged in a format of

96 well plate. The ODs will be assigned the names by referencing the above annotation file. A trunk of the OD file is shown below,

#BLOCKS= 4

Plate: Plate1 1.3 TimeFormat Endpoint Absorbance Raw FALSE 1 2 450 620 1 12 96 1 8

Temperature(¡C) A1 A2 A3 A4 A5 A6 A7 A8

24.6 2.2416 2.6971 2.4565 0.0647 0.0611 0.0591 0.0635 0.0592 0.0698 0.0629

24.6 0.0372 0.0363 0.039 0.0384 0.036 0.0352 0.0383 0.0371 0.0386 0.0369

~End

4. Design file:

   This is the file to be fed directly into the package. The software will read the data correctly by the information contained in this file. It is in a table format (tab-delimitted text file). Each row of the table is about one run of ELISA reading, and each reading could be made up of one or multiple ELISA plate. Multiple ELISA readings (runs) become one batch, depending on the standard lot number for example. Each run (row) might have the same or different annotation or standard files. Of course, each run (row) should have different OD files. The package comes with a set of sample input files. The design file can be viewed by the following r code.

```
dir_file<-system.file("extdata", package="ELISAtools")
tbl.design<-read.table(file.path(dir_file,"design.txt"), header=T,sep="\t")
print(tbl.design)
```

```
#>   ExpID            FileName  Batch Num_Plate      Date   AnnotationFile
#> 1  Exp1         Assay_2.txt Batch1         1 9/18/2009 AnExp_2plate.txt
#> 2  Exp1   Assay_3_and_4.txt Batch1         2 9/18/2009 AnExp_2plate.txt
#> 3  Exp1 Assay_11_and_12.txt Batch2         2 9/18/2009 AnExp_2plate.txt
#>      Std_Conc Dir_Annotation Dir_StdConc
#> 1 stdConc.txt             NA          NA
#> 2 stdConc.txt             NA          NA
#> 3 stdConc.txt             NA          NA
```

Two notes about the input files: 1) Standard, annotation and OD data files are arranged into different sections, each of which is for one elisa plate. The section/plate is denoted by lines starting with "Plate:. . . "

and "~End". The content within these two lines are read and the rest will be discarded. Each run could have many plates. Please keep the files in consistent about number of plates; 2) it is not important how the files named as long as they are specified correctly.

Please see the example data input files at package installed folder. The install folder could be obtained with the following R commands,

```
system.file("extdata", package="ELISAtools");
```

```
#> [1] "/tmp/user/1000/RtmpmUk4Q4/Rinst1086298b518b/ELISAtools/extdata"
```

To read the input files, please generate the files specified above and then call the command of "loadData".

```
dir_file<-system.file("extdata", package="ELISAtools")


batches<-loadData(file.path(dir_file,"design.txt"))
```

```
#> Reading Data for Batch: 1 -- Batch1
#>  Experiment: 1 -- Exp1
#>  Experiment: 2 -- Exp1
#> Reading Data for Batch: 2 -- Batch2
#>  Experiment: 1 -- Exp1
#> Done!!!
```

```
#batch data object
names(batches)
```

```
#> [1] "Batch1" "Batch2"
```

```
length(batches)
```

```
#> [1] 2
```

```
#visualize the raw day without analyses
plotAlignData(batches);
```

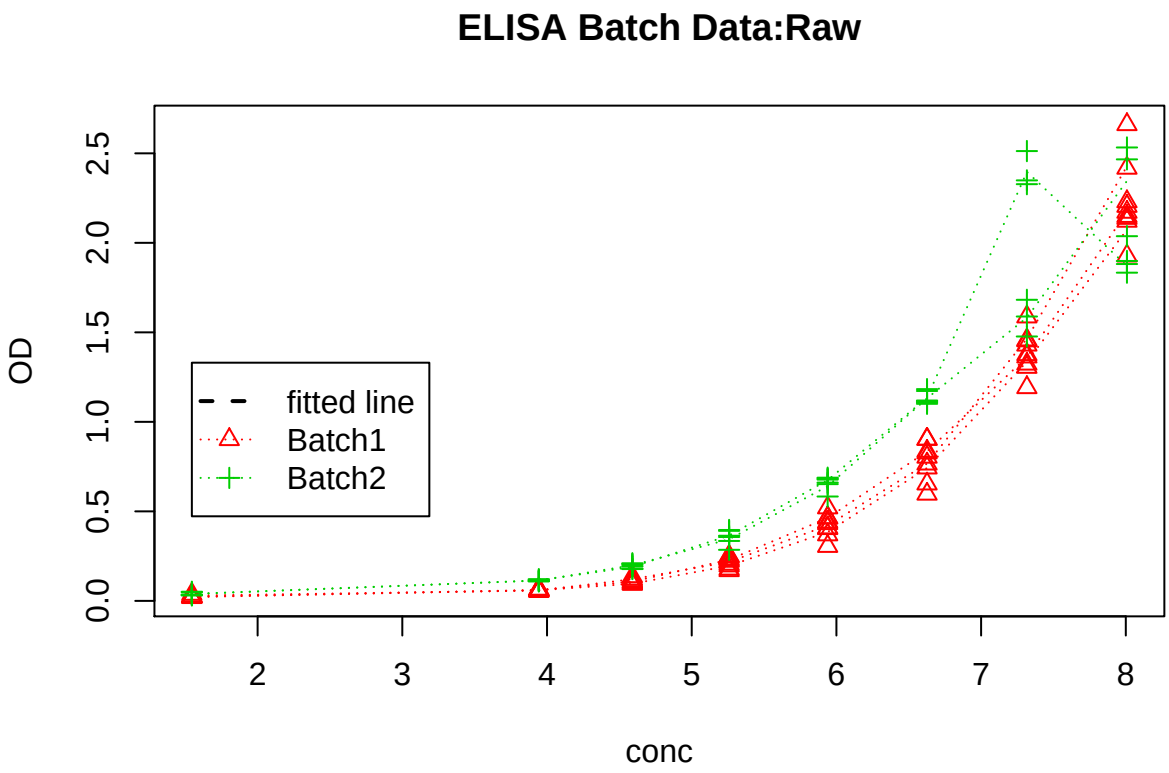

Figure 1: Raw batch data

```
#> NULL
```

The above code will read in data for two batches of ELISA experiments, and each contains one or two runs, which include one or two elisa plates. We also plotted the raw data using the *plotAlignData()* (see Figure 1).

3. Model fitting and unknown concentration estimation

After reading the data input, the model fitting can be run by *runFit()* function. Either 5-parameter or 4-parameter logistic function can be chosen for fitting. Unknown sample concentration is estimated by calling *predictAll()* function. This function estimate both the unknown centration based on the standards on the same plate and will also correct for the batch effects, if there more than one batches specified. The following code will run the analysis based on the sample input in the package folder.

```
#make a guess for the initial value of the parameters,
#the other two parameters a and d will be estimated based on data.
#Sometimes it is important to make the initial values
#closed to the true ones
pars<-c(7.2,0.05, 0.015)
names(pars)<-c("xmid", "scal", "g")

#do fitting. model will be written into data set.
batches<-runFit(pars=pars,  batches=batches, refBatch.ID=1  )
```

```
#> It.    0, RSS =    150.543, Par. =       7.2      0.05     0.015  0.0719858 -0.0266009   0.561701
#> It.    1, RSS =    29.8934, Par. =   9.18304    2.42055  0.732203 -0.00877949    -0.1111    0.7529
#> It.    2, RSS =    19.8829, Par. =   8.27077    1.82651  0.827047   0.233295  -0.217177   0.810197
#> It.    3, RSS =    13.7017, Par. =   6.66284   0.527611  0.934275  0.0055074 -0.0481826   0.298624
#> It.    4, RSS =    2.90591, Par. =   7.09358   0.745189   1.00764  0.0150432 -0.0895312   0.464597
#> It.    5, RSS =    2.60539, Par. =   7.37122   0.639172  0.812993  0.0781559 -0.0940255   0.504719
#> It.    6, RSS =    2.57826, Par. =   7.41157   0.613732  0.774134  0.0791265 -0.0896965   0.512471
#> It.    7, RSS =    2.54194, Par. =   7.50216   0.583111  0.695468  0.0794811 -0.0886957   0.514304
#> It.    8, RSS =    2.50628, Par. =    7.6931   0.514954  0.538393  0.0807765 -0.0872921   0.514611
#> It.    9, RSS =    2.48391, Par. =    7.7604   0.477694  0.491376  0.0848433 -0.0858748   0.521245
#> It.   10, RSS =    2.48286, Par. =   7.75941   0.474778  0.490348  0.0876899 -0.0848607   0.527894
```

```
#> It.   11, RSS =    2.48279, Par. =    7.76256   0.472863   0.487708   0.087354 -0.0847152   0.528743
#> It.   12, RSS =    2.48276, Par. =    7.76668   0.470838    0.48448   0.087488 -0.0846484    0.52904
#> It.   13, RSS =    2.48276, Par. =    7.76676   0.470675   0.484357  0.0876237 -0.0846153   0.529405
#> It.   14, RSS =    2.48276, Par. =    7.76708   0.470504   0.484103  0.0876123 -0.0846073   0.529445
#> It.   15, RSS =    2.48276, Par. =     7.7671   0.470483   0.484081   0.087625 -0.0846041   0.529477
```

```
#now call to do predications based on the model.
batches<-predictAll(batches);
```

4. Visualizing and reporting results

To visualiz the fitting and S factor corretion of batch effects, we can call two plotting functoins, *plotBatchData()* and *plotAlignData()*. The former simply plots the batch data without doing batch effect correction and the latter with the batch effector correction.

The following code chunk plots Batch 1 ELISA data with the 5-parameter fitting, but no S factor shifting/normalization (Figure 2).

```
#plotting batch 1 data for visualizing
plotBatchData(batches[[1]])
```

```
#> NULL
```

The next chunk of code plots both batches together after the batch effects adjusted (Figure 3). Comparing Figure 1 with Figure 3, we can clearly see the batch effects and the correction.

```
#plotting batch data with adjustment for visualizing
plotAlignData(batches)
```

```
#> NULL
```

The batch data can also be saved to a text file with the fitting results and S factors.

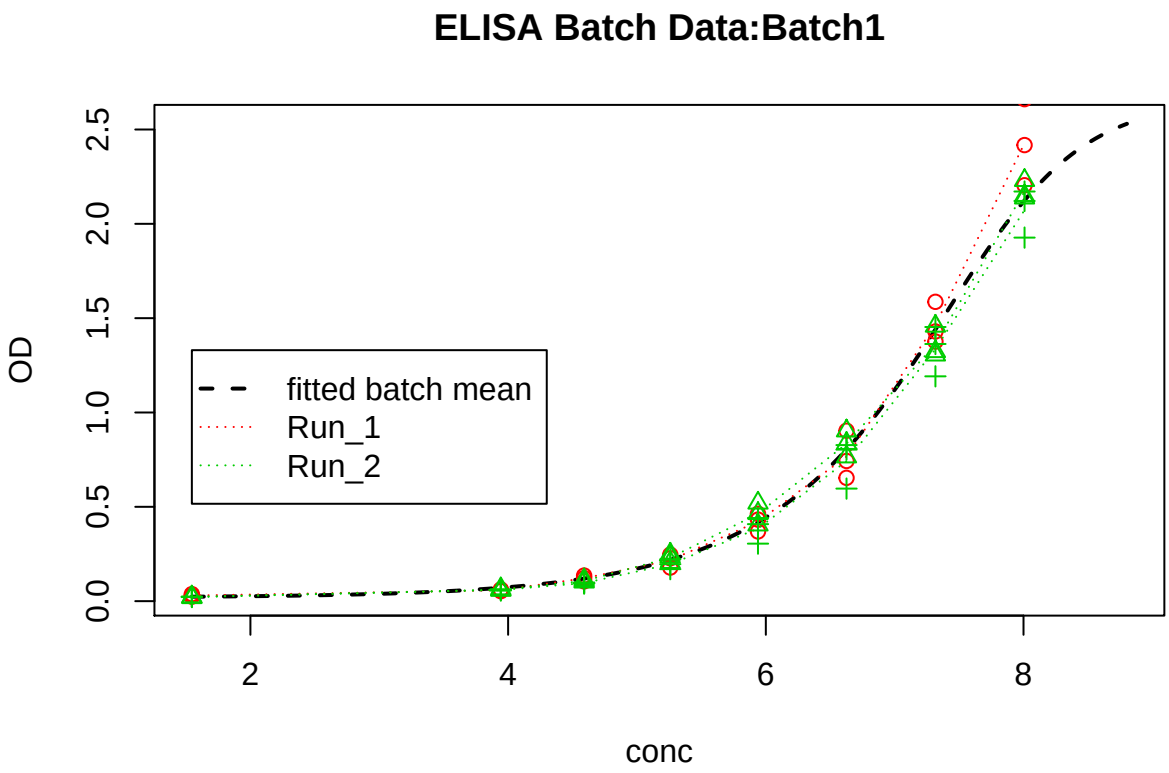

Figure 2: Batch 1 ELISA data with 5-parameter fitting (no S factor adjustment)

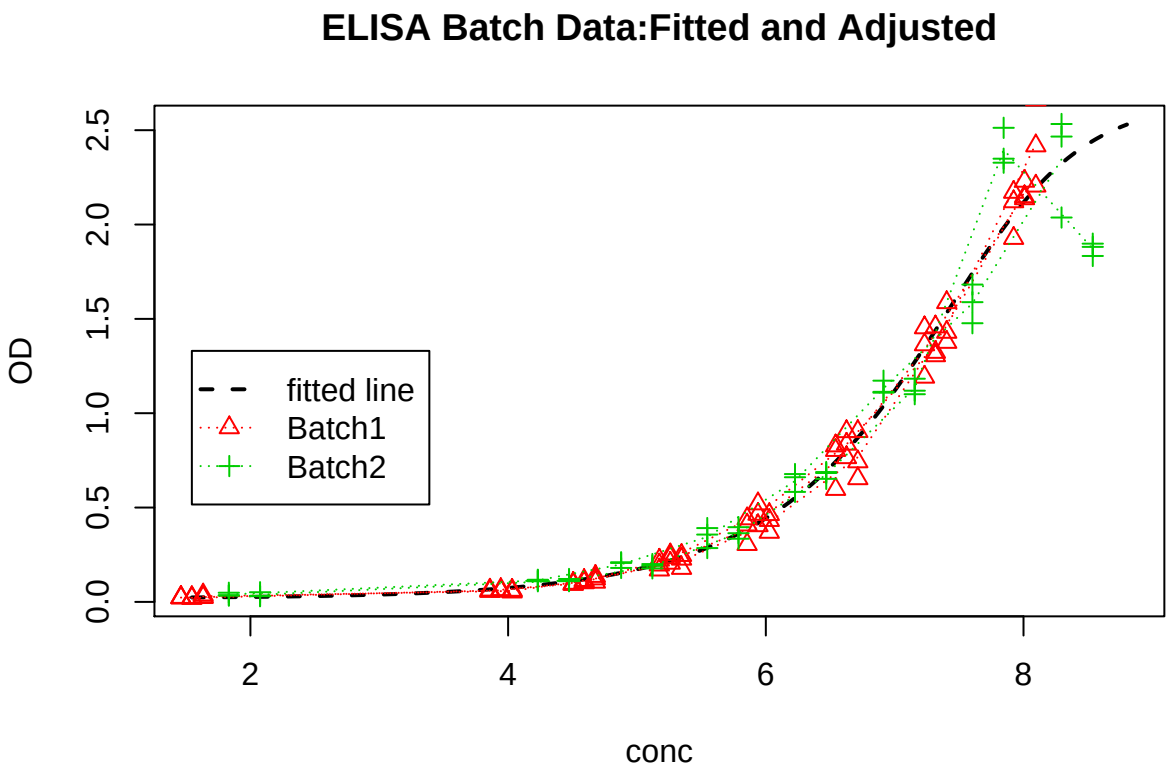

Figure 3: ELISA batch data with 5-parameter fitting and S factor adjustment

```
#save the batch data with fitting results and S factors
fn<-file.path(tempdir(),"batchAnalysis.txt");
saveDataText(batches, fn);
```

We also develop a function *reportHtml()*, which will generate a comprehensive report of the fitting and batch adjusting results in a html format and it also include some brief QC for the fitting. Run the following, (PLEASE DO remember to specify a directory, where you have the write permission, to write the report. The *ELISAtools* package *extdata* directory is most likely not writable for regular users.)

```
#reporting. need to specify a directory that is writable!!!
reportHtml(batches, file.dir=tempdir())
```

An html report will be generated and a text format file is also saved to the disk. In this case, the html report and the data file have the default names, *report.html* and *report.txt*.

5. Save/load the batch data in R data set

The ELISA data can be saved in R data set and the previously saved data can be load/read to combine with the current data. The example R code are listed as below,

```
  #now saving the combine data.
  saveDB(batches, file.path(tempdir(),"elisa_tool1.rds"));
  batches.old<-loadDB(file.path(tempdir(), "elisa_tool1.rds"));
```

We can also combine the two ELISA batch data into one to analyze together and save for the future use.

```
  #now suppose want to join/combine the two batches, old and new
  batches.com<-combineData(batches.old, batches);
  reportHtml(batches.com,file.dir=tempdir())
```

One thing to mention is that we need to re-run the fitting after combining the two batches of data.

4. Session Info

The version number of R and packages loaded for generating the vignette were

```r
sessionInfo()
```

```
#> R version 3.6.3 (2020-02-29)
#> Platform: x86_64-pc-linux-gnu (64-bit)
#> Running under: Debian GNU/Linux 10 (buster)
#>
#> Matrix products: default
#> BLAS:   /usr/lib/x86_64-linux-gnu/atlas/libblas.so.3.10.3
#> LAPACK: /usr/lib/x86_64-linux-gnu/atlas/liblapack.so.3.10.3
#>
#> locale:
#>  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
#>  [3] LC_TIME=en_US.UTF-8        LC_COLLATE=C
#>  [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
#>  [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
#>  [9] LC_ADDRESS=C               LC_TELEPHONE=C
#> [11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
#>
#> attached base packages:
#> [1] stats     graphics  grDevices utils     datasets  methods   base
#>
#> other attached packages:
#> [1] ELISAtools_0.1.5 minpack.lm_1.2-1 stringi_1.5.3    R2HTML_2.3.2
#>
#> loaded via a namespace (and not attached):
#>  [1] compiler_3.6.3  magrittr_2.0.1  htmltools_0.5.1 tools_3.6.3
#>  [5] yaml_2.2.1      rmarkdown_2.6   highr_0.8       knitr_1.30
#>  [9] stringr_1.4.0   digest_0.6.27   xfun_0.20       rlang_0.4.10
#> [13] evaluate_0.14
```

5. Reference and Citation

Feng, Feng, Morgan Thompson, Beena Thomas, and Elizabeth Duffy. 2018. "A Computational Solution to Improve Biomarker Reproducibility During Long-Term Projects." *BioRxiv*. Cold Spring Harbor Laboratory. https://doi.org/10.1101/483800.