`fluodilution` package: Interpretation of Fluorescence Dilution Experiments

Hadrien Chauvin, Kathryn Watson, Vahid Shahrezaei, David Holden, and Sophie Helaine

Imperial College London

March 13, 2018

Fluorescence dilution, or FD (and also known as proliferation assay), is a technique used to get information concerning cell growth at the single-cell level, both in immunology (Lyons and Parish, 1994) and microbiology (Helaine et al., 2014, 2010; Roostalu et al., 2008).

FD makes use of a fluorescent reporter, such as a dye like CFSE or a fluorescence protein (e.g., GFP) expressed by a plasmid. In both cases, the fluorophore is present in fixed quantities in progenitor cells/bacteria: therefore, whenever a cell/bacterium divides, the fluorescence is approximately halved. The distribution of fluorescence in the population can then be recorded over time using flow cytometry. Augmented by cell counts / Colony Forming Units (Chauvin et al., 2016) or the staining of dying/dead cells (Hyrien et al., 2010), the resulting dataset can be used to infer a set of "biological constants" such as the growth/death rates or their variability. The **fluodilution** package provides generic functions to manipulate FD data and find these parameters.

In this vignette, we will show how to fit one of the experiments of Helaine et al. (2010). In this experiment, primary BMM extracted from C57 B1/6 mice were infected by wild-type 12023s *Salmonella* Typhimurium harbouring the pFCcGi plasmid, a plasmid with a constitutive reporter helping in bacterial recognition and an inducible reporter used for fluorescence dilution. The data was acquired on a LSRFortessa cytometer (DB). Raw data is available upon request.

First, we load the **fluodilution** package and the dataset. This dataset, provided along with the package, is an `fd_data` object.

```
library(fluodilution)
data(FdSTyphimuriumWTC57)
invisible(summary(FdSTyphimuriumWTC57))
```

`fd_data` objects are used by **fluodilution** to represent FD datasets. They also contain, as `attributes`, various default parameters to shorten calls to functions such as `cutoff` or `fd_model` (see below). Here, our dataset contains CFU counts in log10 units and fluorescence histograms. It is possible to plot an overview of this dataset:

```
plot(FdSTyphimuriumWTC57, type=c("hist", "N"))
```

The quality of FD data is lower when fluorescence is lower, as both autofluorescence and stochastic partitioning quick in (Chauvin et al. 2016). Therefore, it is good practice to gate away low fluorescence using `cutoff`. `cutoff` returns an `fd_data` object with the leftmost part of the histograms, below the cutoff points, removed. As a consequence, the proportions do not sum to one anymore. It is possible to jauge the extent of the cutoff by calling `plot` again (and in general `plot` can be used to output many other diagnostic plots).

```
# plot(FdSTyphimuriumWTC57, type="cutoff")
dat <- cutoff(FdSTyphimuriumWTC57)
```

Next, we must construct the FD model with which to interpret the experiment. This model specifies a number of parameters to be estimated. In our framework, the model is hierarchical: first, a Finite Mixture Model (FMM) translates fluorescence into proportions of cells that have divided a given number of times (collectively giving "division histograms"), followed by a proliferation model that explains those proportions in terms of biological activity (division, death, migration). The `fd_model` function acts as a glue between those two sub-models.

```
mdl <- fd_model(fmm="gaussian", proliferation="branching", data=dat)
```

Here, we specified a Gaussian FMM and a branching process as the proliferation model. We could also have chosen an FMM that corrects for autofluorescence (`fmm="af"`) or both autofluorescence and stochastic partitioning of fluorescent molecules into daughter cells upon division (`fmm="af_bp"`). These additional FMM are described at length in Chauvin et al. (2016). In terms of proliferation models, a Cyton model (Hawkins et al., 2007) could also have been used (`pro="cyton"`).

The "default" values of the parameters can be accessed *via* `start` (they can be used as starting values during optimization). The boundary for the acceptable values of these parameters is given by `lower` and `upper`.

```
cbind(Start = start(mdl), Lower = lower(mdl), Upper = upper(mdl))
```

```
##                   Start Lower Upper
## fmm.c0             5.00  0.00  10.0
## pro.One.res0       0.20  0.00   1.0
## pro.One.res        0.20  0.00   1.0
## pro.One.p          0.50  0.10   0.9
## pro.One.p0         0.49  0.10   0.9
## pro.One.g.mm       5.00  1.00  20.0
## pro.One.g.delta    1.00  0.01  10.0
## pro.One.g.ss       0.50  0.10   0.5
## pro.One.g0.mm      5.00  1.00  20.0
## pro.One.g0.delta   1.00  0.01  10.0
## pro.One.g0.ss      0.50  0.10   0.5
## pro.One.f.mm       5.00  1.00  20.0
## pro.One.f.delta    0.20  0.01  10.0
## pro.One.f.ss       0.50  0.10   0.5
## pro.One.f0.mm      5.00  1.00  20.0
## pro.One.f0.delta   0.20  0.01  10.0
## pro.One.f0.ss      0.50  0.10   0.5
```

We see that by default the model features a large amount of parameters. It is unlikely that so many parameters can be optimized all at once, even with sophisticated gloabl search algorithms. Therefore, we have to constrain the parameters to some specific values. This can be achieved by specifying a set of constraints using a notation we developed for this package. (However, this notation can be used to constrain any nonlinear model in general.)

```
# nolint start
cstr <- ~
    pro:all:(f0/f/g0/g):{
        ss <- 0.5
    } + pro:all:{
        res <- 0
    } + fmm:{
        c0 <- 0
    } + ~pro:all:((f0/g0):{
        delta <- 1
    } + (f/g):{
        delta <- 0.01
    }) + pro:all:f0:{
```

```
        mm <- .L2$f$mm
    }
# nolint end
mdl <- update(mdl, data=dat, addcstr=cstr)
```

According to the above constraints, the distributions of times to death and times to first division are required to be exponential. As to the times to subsequent division, they have to follow a Dirac distribution (fixed-length cell cycles). Also, the mean times to division are always the same but the times to death could be different before and after the first division. We also do not allow the presence of nongrowing bacteria after the first division.

This model, along with the dataset, can be used with, e.g., `nls` to perform a nonlinear least-square regression. The least square formula can be retrived with `fd_formula` (`mgen` is the maximum number of generation to fit):

```
fd_formula("mdl", mgen=10)
```

```
## y ~ fd_predict_mat(mdl)(data.frame(Category = Category, Time = Time,
##     Timepoint = Timepoint, a = a, b = b, Inoculum = Inoculum,
##     Weight = Weight, Type = Type), cbind(pro.One.res0, pro.One.p,
##     pro.One.p0, pro.One.g.mm, pro.One.g0.mm, pro.One.f.mm), mgen = 10)
```

Because the problem still remains complex and it is difficult to determine appropriate starting values beforehand, local optimization as provided by `nls` is unlikely to succeed. Therefore, the user is advised to perform a global search using the "contributed" function `nlsSA` (it is not formally part of the package as it deals with nonlinear regression in general and not FD *per se*). This function behaves as if `nls` was called. `nlsSA.R` provides other such variants of `nls`.

```
source(system.file("contrib", "nlsSA.R", package="fluodilution"))
# Put maxit=100 to actually run the optimization
fit <- nlsSA(
    fd_formula("mdl", mgen = 10),
    data = dat,
    start = start(mdl),
    lower = lower(mdl),
    upper = upper(mdl),
    trace = T,
    control = list(simple.function = T, maxit = 1)
)
```

As the object `fit` is an `nls` object, methods such as `coef`, `vcov`, `predict` or `summary` can be used. Additionally, it is possible to "unconstrain" the model to get back to a full list of parameters:

```
coef(fit)
```

```
##  pro.One.res0     pro.One.p    pro.One.p0  pro.One.g.mm
##          0.20          0.50          0.49          5.00
## pro.One.g0.mm  pro.One.f.mm
##          5.00          5.00
```

```
unlist(relist(coef(fit), mdl))
```

```
##           fmm.c0     pro.One.res0      pro.One.res
##            0.000            0.098            0.000
##        pro.One.p       pro.One.p0     pro.One.g.mm
##            0.500            0.392            5.000
##  pro.One.g.delta     pro.One.g.ss    pro.One.g0.mm
##            0.010            0.500            5.000
## pro.One.g0.delta    pro.One.g0.ss     pro.One.f.mm
##            1.000            0.500            5.000
```

```
##  pro.One.f.delta     pro.One.f.ss    pro.One.f0.mm
##            0.010            0.500            5.000
## pro.One.f0.delta    pro.One.f0.ss
##            1.000            0.500
```

In the example above, we have performed an ordinary least-square regression (OLS). It is possible to improve on statistical efficiency by performing a mixed-effects regression. The OLS coefficients can be used as starting values for a call to `nlme` (package **nlme**).

The package features other functions to support and enrich the workflow above.

References

E D Hawkins, M L Turner, M R Dowling, C van Gend, and P D Hodgkin. A model of immune regulation as a consequence of randomized lymphocyte division and death times. *Proceedings of the National Academy of Sciences of the United States of America*, 104(12):5032–7, mar 2007.

Sophie Helaine, Jessica A Thompson, Kathryn G Watson, Mei Liu, Cliona Boyle, and David W Holden. Dynamics of intracellular bacterial replication at the single cell level. *Proceedings of the National Academy of Sciences of the United States of America*, 107(8):3746–51, feb 2010.

Sophie Helaine, Angela M Cheverton, Kathryn G Watson, Laura M Faure, Sophie A Matthews, and David W Holden. Internalization of Salmonella by macrophages induces formation of nonreplicating persisters. *Science (New York, N.Y.)*, 343(6167):204–8, jan 2014.

Ollivier Hyrien, Rui Chen, and Martin S Zand. An age-dependent branching process model for the analysis of CFSE-labeling experiments. *Biology direct*, 5:41, jan 2010.

A.Bruce Lyons and Christopher R. Parish. Determination of lymphocyte division by flow cytometry. *Journal of Immunological Methods*, 171(1):131–137, may 1994.

Johanna Roostalu, Arvi Jõers, Hannes Luidalepp, Niilo Kaldalu, and Tanel Tenson. Cell division in Escherichia coli cultures monitored at single cell resolution. *BMC microbiology*, 8:68, jan 2008.