The *cellGrowth* package

Julien Gagneur, Andreas Neudecker

1st March 2012

1 Introduction

Growth is a key cellular phenotype relevant in areas ranging from microbiology to cancer biology. Hence, quantitative measures of growth must be accurately estimated. Historically, many parametric models have been proposed, reviewed in ([1]). However, in our experience, growth curves rarely follow these idealistic behaviours. In practice, non-parametric models, in which curves are simply smoothed to reduce noise in the data, do a better job for capturing all possible behaviours.

This package provides fitting growth curves in non-parametric (local regression) and parametric models. It determines the maximum growth rate, e.g. generations per time unit, and the maximum of growths. It comes with neat plotting functions, automatic bandwidth selection for non-parametric models and handles data coming in well plate format.

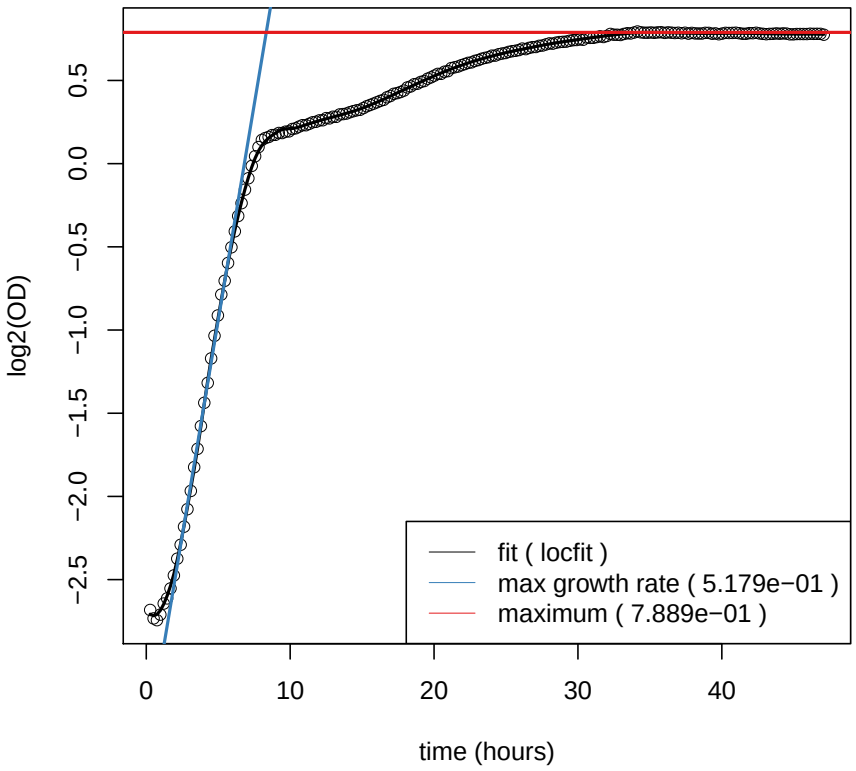
This vignette demonstrates the key features of `cellGrowth`, i.e fitting one growth curve, handling multiple machine runs for plates coming in 96-well plate format and automatic bandwidth selection. The last section describes how to handle custom data formats.

2 Fitting of one curve

We start with fitting one curve with local polynomial fitting, provided by the package `locfit`. The data comes from a 96-well plate with measurement every 15 minutes for a bit less than two days. We below load the whole data and fit a growth curve for the well F2 and display the fit. We convert time from seconds into hours.

```
> library(cellGrowth)

> examplePath = system.file("extdata", package="cellGrowth")
> dat = readYeastGrower(file.path(examplePath,"Plate2_YPFruc.txt"))
> fit = fitCellGrowth(
         x=dat$time,
         z=log2(dat$OD[[which(getWellIdsTecan(dat) == "F02")]])
         )
> plot(fit, scaleX=1/(60*60), xlab="time (hours)")
```



The fit object also contains the maximum growth rate, e.g. gerenerations per time unit, the maximum of growth and the datapoint where the maximum growth or the maximum is reached.

```
> attributes(fit)[c(3,4,5,6)]

$maxGrowthRate
[1] 0.0001438498

$pointOfMaxGrowthRate
[1] 17

$max
[1] 0.7888636

$pointOfMax
[1] 150
```

3 Experimental design with multiple machine runs

An experiment is a set of output files from different machine runs, each one on a specific plate. The experiment design is described by two further, tab-separated, files: a machine run file and a plate layout file. An example of a machine run file is provided

```
> mr_file = read.delim(file.path(examplePath,"machineRun.txt"))
> mr_file

  directory          filename      plate  media  use
1         . Plate1_YPMalt.txt nonfloc_P1 YPMalt TRUE
2         . Plate2_YPMalt.txt nonfloc_P2 YPMalt TRUE
3         . Plate1_YPFruc.txt nonfloc_P1 YPFruc TRUE
4         . Plate2_YPFruc.txt nonfloc_P2 YPFruc TRUE
```

It has `directory`, `filename` and `plate` for mandatory columns.

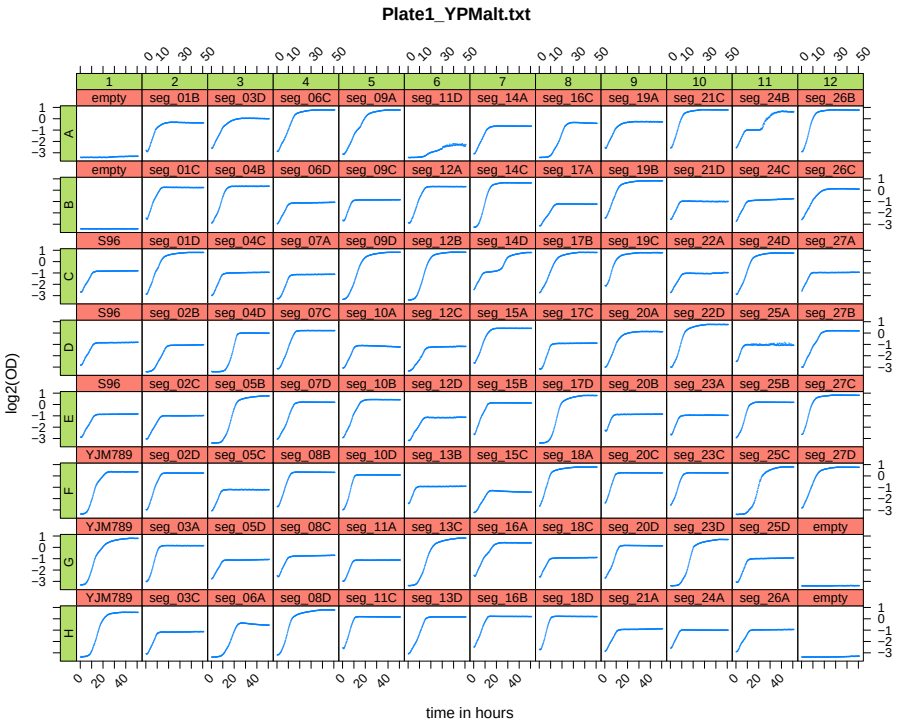
We also provide the companion layout file

```
> pl_file = read.delim(file.path(examplePath,"plateLayout.txt"))
> head(pl_file)

       plate well  strain background
1 nonfloc_P1  A01   empty      empty
2 nonfloc_P1  A02 seg_01B        SxY
3 nonfloc_P1  A03 seg_03D        SxY
4 nonfloc_P1  A04 seg_06C        SxY
5 nonfloc_P1  A05 seg_09A        SxY
6 nonfloc_P1  A06 seg_11D        SxY
```

It has `plate` and `well` for mandatory columns. `wellDataFrame` combines these two files into one single object of class well, essentially a data frame. The generic plotting function for this datatype plots a given plate using the function `plotPlate`.

```
> well = wellDataFrame(
                 file.path(examplePath,"plateLayout.txt"),
                 file.path(examplePath,"machineRun.txt")
                 )
> plot(well,labelColumn="strain",scaleX=1/3600,xlab="time in hours")
```



You can use the function `fitCellGrowths` to fit multiple growth curves at once.

```
> fits <- fitCellGrowths(well)
```

```
treating 4 unique tecan files.
treating file /tmp/RtmpHidljn/Rinst74aa1d806787/cellGrowth/extdata/./Plate1_YPMalt.txt
treating file /tmp/RtmpHidljn/Rinst74aa1d806787/cellGrowth/extdata/./Plate1_YPFruc.txt
treating file /tmp/RtmpHidljn/Rinst74aa1d806787/cellGrowth/extdata/./Plate2_YPMalt.txt
treating file /tmp/RtmpHidljn/Rinst74aa1d806787/cellGrowth/extdata/./Plate2_YPFruc.txt
```

It returns a data frame with maximum growth rate, maximum and the time points at which the maximum growth rate and the maximum is reached.

```
> head(fits)
```

```
  maxGrowthRate pointOfMaxGrowthRate         max pointOfMax
1  3.340948e-06                   103 -3.30494108        200
2  4.932709e-06                   132 -3.08932379        150
3  8.028939e-05                    19 -0.30745870        106
4  1.421942e-04                    20  0.25841063        200
5  5.883646e-05                    12  0.04323751        131
6  1.331913e-04                    20  0.19517028        190
```

4 Automatic bandwidth selection

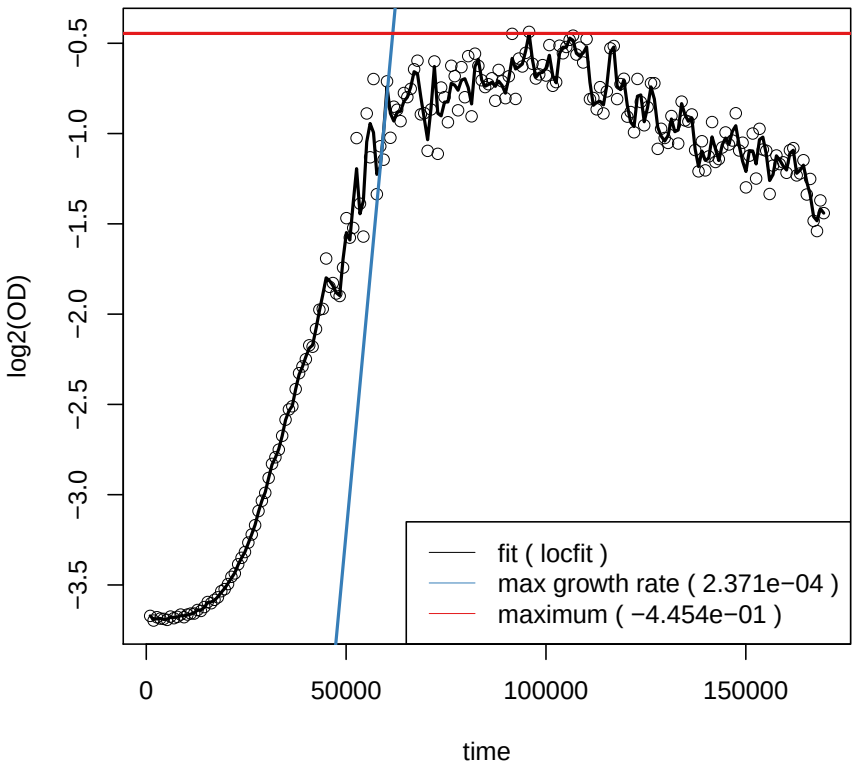
Local polynomial fitting, as most smoothing procedures, depends on a bandwidth parameter. The larger the bandwidth, the smoother the fit. Too large

bandwidth underestimate growth rates whereas too small ones tend to be sensitive to noise in the data. bandwidthCV() uses cross-validation to automatically select a bandwidth which gives good prediction on left out data as well as robust estimate of growth rate parameters.

```
> ## Not run:
> #        bw <- bandwidthCV(
> #                well,
> #                bandwidths=seq(0.5*3600,10*3600, length.out=30)
> #        )
> ## End(Not run)
```

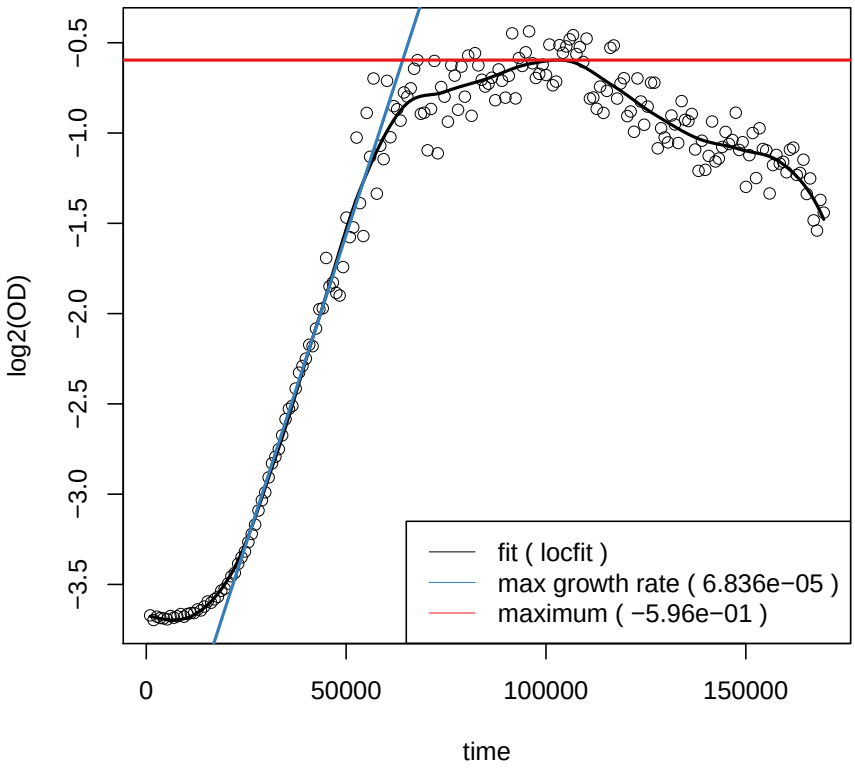
This call returns a list with the "optimal" bandwidth and data from the cross-validation, e.g. the squared error of the different bandwidths. Here you can see a plot of a fit with a too-low bandwidth

```
> fit_small = fitCellGrowth(
          x=dat$time,
          z=log2(dat$OD[[which(getWellIdsTecan(dat) == "E09")]]),
          locfit.h=1800
          )
> plot(fit_small)
```



and one with the output from bandwidthCV

```
> fit_big = fitCellGrowth(
          x=dat$time,
          z=log2(dat$OD[[which(getWellIdsTecan(dat) == "E09")]]),
          locfit.h=24000
          )
> plot(fit_big)
```



5 How do I use my own data format?

Data may come in any format and not necessarily from a well plate setup. Store your data in tab-separated file and load them into a data frame. Then call fitGrowthCurve(), as shown:

```
> own_file = read.delim(file.path(examplePath,"customDataFormat.txt"))
> head(own_file)

  time     od
1  980 0.1036
2 1827 0.1018
```

```
3 2673 0.1029
4 3520 0.1026
5 4366 0.1046
6 5215 0.1070

> x = own_file[[1]]
> z = own_file[[2]]
> fit = fitCellGrowth(x,z)
> attr(fit,"maxGrowth")

[1] 4.976027e-05

> attr(fit,"pointOfMaxGrowth")

[1] 33
```

References

[1] Zwietering MH, Jongenburger I, Rombouts FM, van 't Riet K. Modeling of the Bacterial Growth Curve *Applied and environmental biology* 56(6):1875-81

[2] Kelly LA, Gibson G, Gettinby G, Donachie W, Low JC The use of dummy data points when fitting bacterial growth curves *IMA Journal of Mathematics Applied in Medicine and Biology* 16(2):155-70