Overview of the **htmf** Package

Laurent Gautier <laurent@cbs.dtu.dk>
<add your name here when adding content>

March 23, 2012

1 Introduction

```
> library(htmf)
```

The package **htmf** relies on other R packages:

```
[1] "reshape, ggplot2, Biobase, methods, stats"
```

Those must be available, or an error will occur when trying to attach **htmf**

2 Reading data

2.1 Biolector

The *biolector* system is recording the measurments in a CSV file.

The function `read_biolector` can be passed a filename or a connection and read the file's content into a `PlateRun` data structure.

Here we use a *gzipped* connection:

```
> fn <- system.file("exampleData", "biolector_data.csv.gz", package="htmf")
> # Create a gzip connection
> gzf <- gzfile(fn)
> pr <- read_biolector(gzf)
> close(gzf)
```

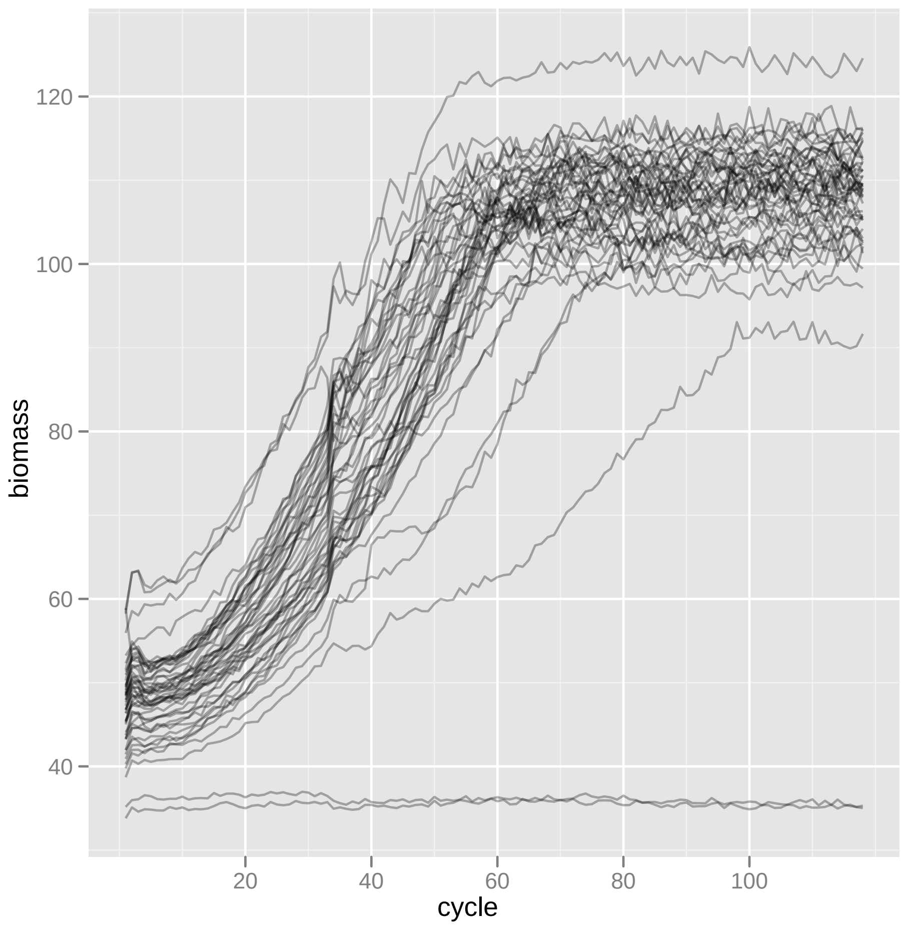
3 Plotting data

3.1 Default plots

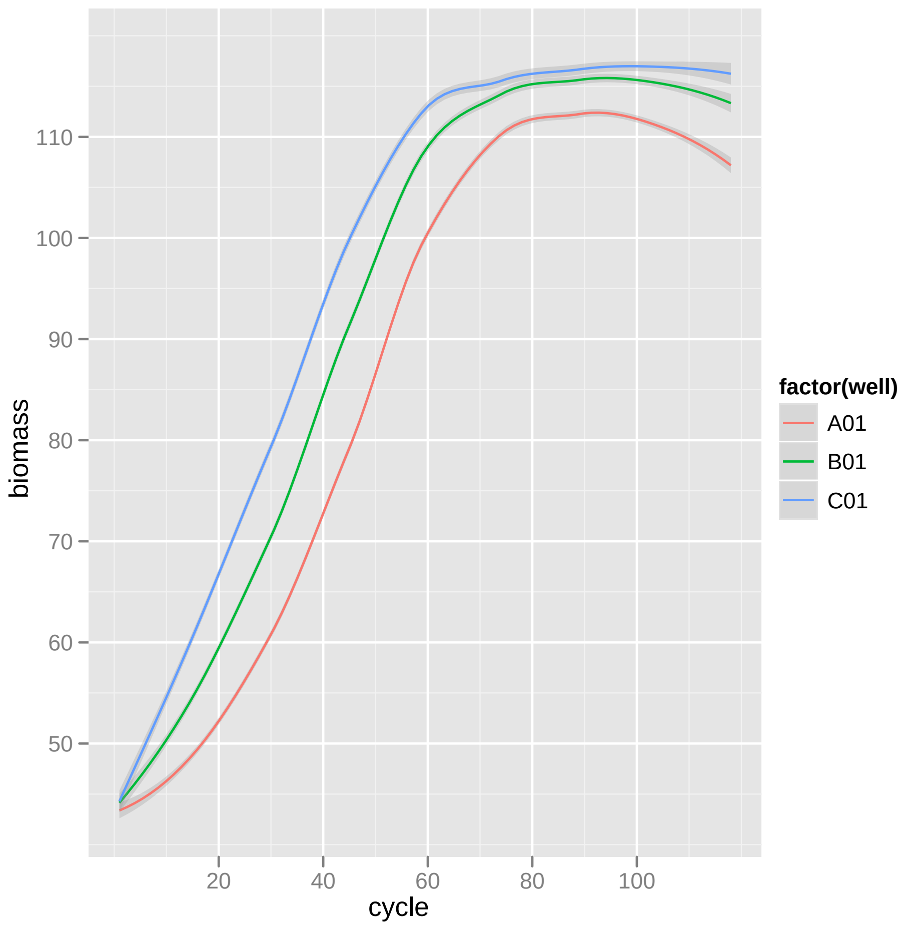
```
> # plot_biolector
```

3.2 Custom plots

```
> dataf_biomass <- melt(measure(pr$Biomass))
> names(dataf_biomass)[ncol(dataf_biomass)] <- "biomass"
> p <- ggplot(dataf_biomass) +
+      aes(x = cycle, y = biomass) +
+      geom_line(aes(group = well), alpha = 0.3)
> print(p)
>
```



```
> p <- ggplot(subset(dataf_biomass, well %in% c("A01","B01","C01"))) +
+      aes(x = cycle, y = biomass, col = factor(well)) +
+      geom_smooth(aes(group = well), alpha = 0.3)
> print(p)
>
```



4 Computing on the data

4.1 Adjusting the data

4.2 Fitting curves

As observed in earlier plots of the biomass data, a logistic curve seems to be able to represent our data. In an surprising move for fitting dose-respone or growth data, we choose the four-parameter logistic curve.

$$response = f(input) = A + \frac{B - A}{1 + \exp(\frac{xmid - input}{scal})}$$

with

A: lower asymptote

B: higher asymptote

xmid: point of inflection

scal: scale parameter

That function can be defined in R:

```
> fpl <- function(A, B, xmid, scal, x) A+(B-A)/(1+exp((xmid-x)/scal))
```

The first order derivative can be used to compute the slope at the inflection point:

$$\frac{d}{dx}f = \frac{B - A}{scal(1 + \exp(\frac{m-x}{scal}))^2}$$

The package **nlme** is used for the fitting, and to demonstrate how to do it only one growth curve is worked on.

```
> library(nlme)
> dataf_wells <- split(dataf_biomass, dataf_biomass$well)
> params <- getInitial(biomass ~ SSfpl(cycle, 110, 0, 0, 0),
+                       data=dataf_wells[[1]])
> params
```

```
      110          0          0          0
 46.219825 110.889552  44.368340   8.776391
```

```
> A <- params[1]; B <- params[2]
> xmid <- params[3]; scal <- params[4]
> fit <- nls(biomass ~ SSfpl(cycle, A, B, xmid, scal),
+            alg="plinear", data=dataf_wells[[1]])
> fit
```

```
Nonlinear regression model
  model:  biomass ~ SSfpl(cycle, A, B, xmid, scal)
   data:  dataf_wells[[1]]
      A       B    xmid    scal    .lin
 46.220 110.890  44.368   8.776   1.000
 residual sum-of-squares: 377.9

Number of iterations to convergence: 0
Achieved convergence tolerance: 2.852e-14
```

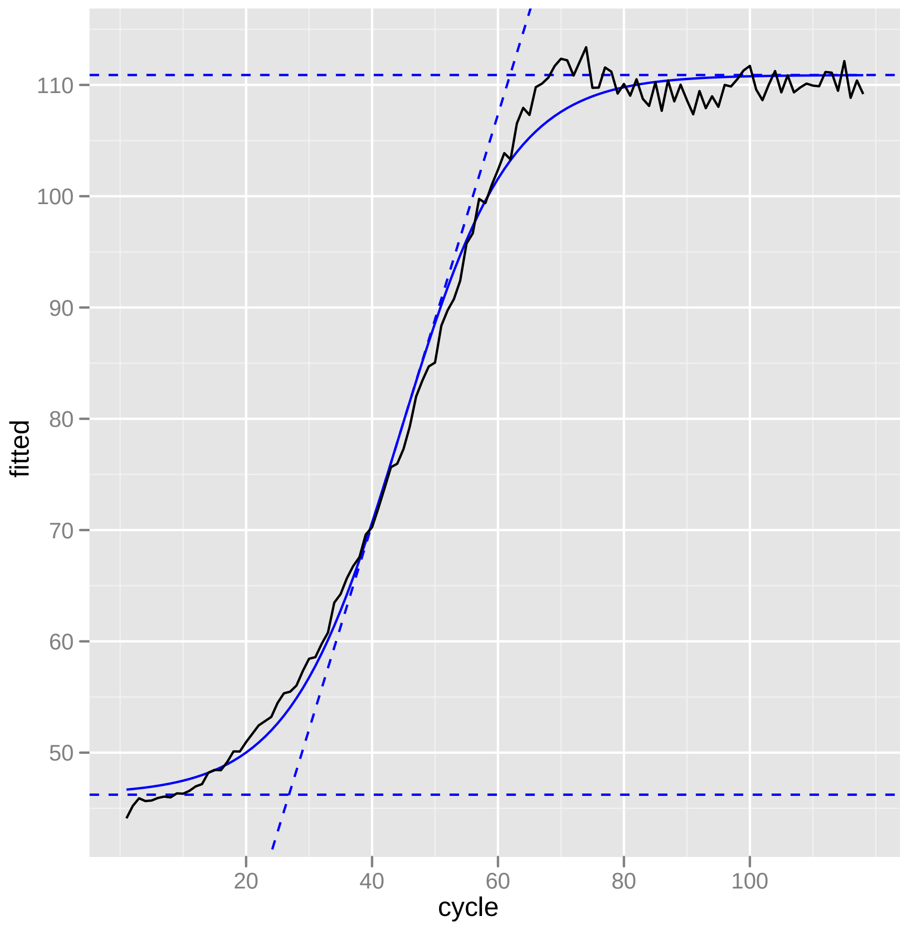
The estimated parameters can be related to physical or biological representations with

A: amount of cells inoculated

B: amount of cells at the end of the fermentation

slope at the inflection point: maximum growth

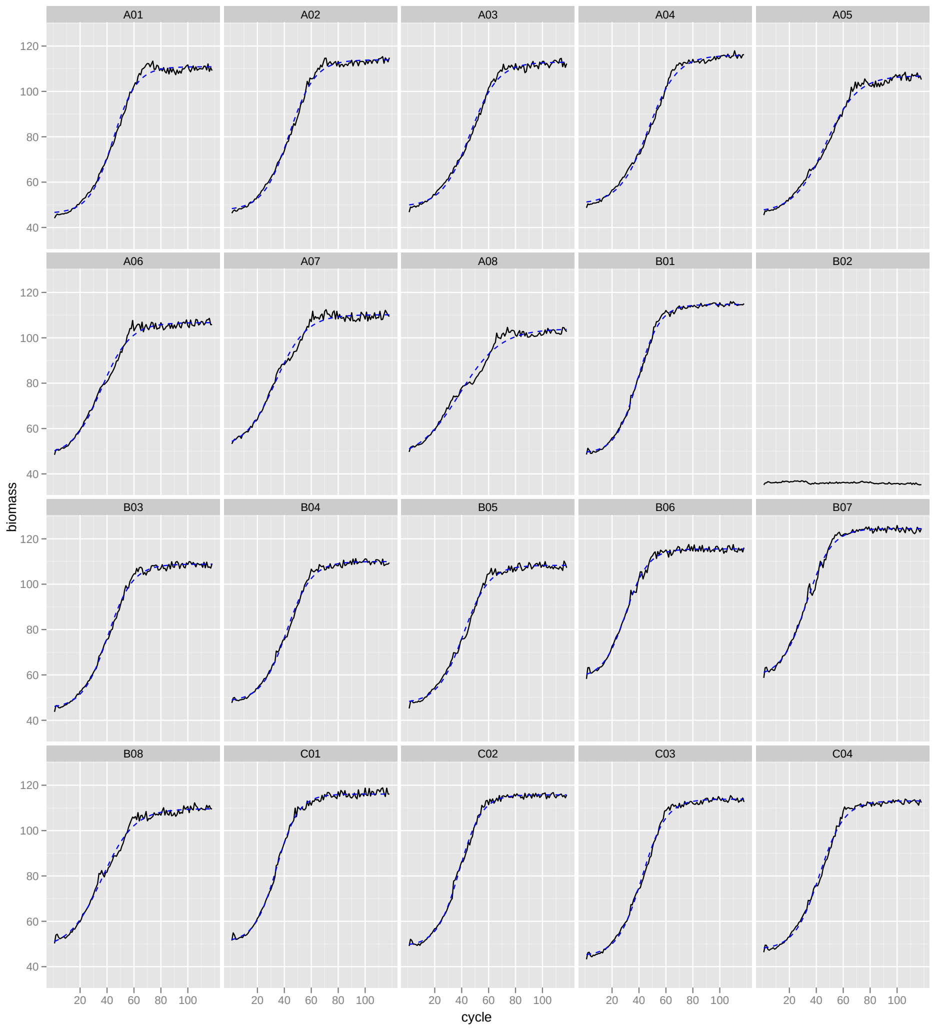
as shown on the figure below.



This process can be automated, and parameters of growth fitted without the user's assistance.

```
> for (i in 1:20) {
+
+   params <- tryCatch(getInitial(biomass ~ SSfpl(cycle, 110, 0, 0, 0),
+                                  data=dataf_wells[[i]]),
+                      error = function(e) NULL)
+   if (is.null(params)) {
+     dataf_wells[[i]]$fitted <- rep(NA, length = nrow(dataf_wells[[i]]))
+   } else {
+     A <- params[1]; B <- params[2]
+     xmid <- params[3]; scal <- params[4]
+     fit <- nls(biomass ~ SSfpl(cycle, A, B, xmid, scal), alg="plinear",
+                data=dataf_wells[[i]])
+
+     tmp <- coef(fit)
+     dataf_wells[[i]]$fitted <- fpl(tmp[1], tmp[2], tmp[3], tmp[4],
+                                     dataf_wells[[i]]$cycle)
+   }
+ }
> dataf_ten <- do.call("rbind", dataf_wells[1:20])
> p <- ggplot(dataf_ten) +
+      geom_line(aes(x = cycle, y = biomass), col="black") +
+      geom_line(aes(x = cycle, y = fitted), col="blue", linetype = 2) +
+      facet_wrap( ~ well)
> print(p)
```

```
>
```



A01
A02
A03
A04
A05
A06
A07
A08
B01
B02
B03
B04
B05
B06
B07
B08
C01
C02
C03
C04
biomass
cycle