Microbial growth analysis with the growthcurver R package

Andrew Tolonen

2024.02

Introduction

The goal of this script is to demonstrate methods for microbial growth analysis using the R package growthcurver. The growthcurver package is described in this publication and in this vignette. The growthcurver package fits the data to a logistic growth model, enabling calculation of a generation time and carrying capacity.

This analysis shows how to load growth data into growthcurver to calculate growth parameters. The outputs are plots of the growth and logisitic fit for each well (Fig 1), a summary of growth parameters (generation time, carrying capacity) for each treatment (Table 2), and a PCA on growth parameters to identify outlier wells (Fig2).

Setup, load data

```
library(growthcurver);
library(tidyverse);
library(readxl);
library(knitr)

mytheme = theme(axis.text.x = element_text(size = 4),
                axis.text.y = element_text(size = 4),
                axis.title.x = element_text(size = 12),
                axis.title.y = element_text(size = 12),
                strip.text.x = element_text(size = 8),
                legend.position = "none",
                aspect.ratio =1,
                panel.grid.minor=element_blank(), panel.grid.major=element_blank());

# load my growth data
datafile = "/home/tolonen/Github/actolonen/MicrobialGrowth/Growthcurver/2024.02.29_growthData.xlsx";
growthdata = read_excel(datafile, sheet = "Data", col_names = TRUE, skip=0);
growthdata = growthdata %>%
  rename(time = Hours) %>% # time col must be called "time"
  mutate_if(is.character, as.numeric); # make all data numeric

table1 = kable(growthdata[1:5, 1:10], caption = "Format of input growth data for growthcurver fit. Colu

table1
```

Table 1: Format of input growth data for growthcurver fit. Columns are time (hours) and OD600 of each well

time	A01	A02	A03	A04	A05	A06	A07	A08	A09
0.0	0.078	0.07	0.070	0.072	0.084	0.087	0.084	0.085	0.189
0.5	0.073	0.07	0.070	0.072	0.081	0.081	0.084	0.089	0.199
1.0	0.074	0.07	0.071	0.071	0.081	0.081	0.093	0.086	0.201
1.5	0.073	0.07	0.071	0.071	0.091	0.084	0.095	0.093	0.207
2.0	0.072	0.07	0.070	0.072	0.082	0.082	0.099	0.097	0.207

Calculate logistic fit, combine with growth data

```
# define well containing 'blank'
growthdata = growthdata %>%
  mutate(blank = A01);

# calc logistic fit for all wells: K / (1 + ((K - N0) / N0) * exp(-r * t))
# all wells are normalized to "blank"
gc_fit = SummarizeGrowthByPlate(growthdata, bg_correct = "blank");

# components of SummarizeGrowthByPlate object
my_k = gc_fit$k;  # carrying capacity
my_n0 = gc_fit$n0; # initial OD600
my_r = gc_fit$r; # intrinsic growth rate
my_genTime = gc_fit$t_gen; # generation time
my_aucL = gc_fit$auc_l; # AUC, integral of logistic eq
my_aucE = gc_fit$auc_e; # AUC, area of trapezoid
my_sigma = gc_fit$sigma; # goodnesss of fit of the parameters of the logistic equation (residual standa
my_note = gc_fit$note; # note if poor fit

# make data.frame of fit data
Well = names(growthdata) %>%
  tail(-1); # remove first elt (time)
fitdata = data.frame(Well, my_k, my_n0, my_r, my_genTime, my_sigma, my_note);

# convert growth data to tidy format
growthdatalong = growthdata %>%
  select(-blank) %>%
  pivot_longer(cols = A01:F12, names_to = "Well", values_to = "OD600");

# # add fit data to growthdata
growthdatalong = left_join(growthdatalong, fitdata, by="Well");
 growthdatalong = growthdatalong %>%
 mutate(my_fit = my_k / (1 + (((my_k - my_n0) / my_n0) * exp(1)^-(my_r * time))));

# add asterisks to well names if fit problem
growthdatalong = growthdatalong %>%
  mutate(fit_qual = if_else(my_note == "", true="", false="*")) %>%
  unite("Well", Well, fit_qual, sep = "");
```

plot growth and fit data

```
# plot growth (black) and fit (red) data for each well
growthplots = ggplot(growthdatalong, aes(x=time, y=OD600))+
 facet_wrap(~Well, ncol=12)+
 geom_line(size=0.2, color='black') +
 geom_line(aes(x=time, y=my_fit), color='red')+
 geom_point(size=0.25) +
 xlab("time") + ylab("OD(600)")+
 mytheme;

growthplots
```

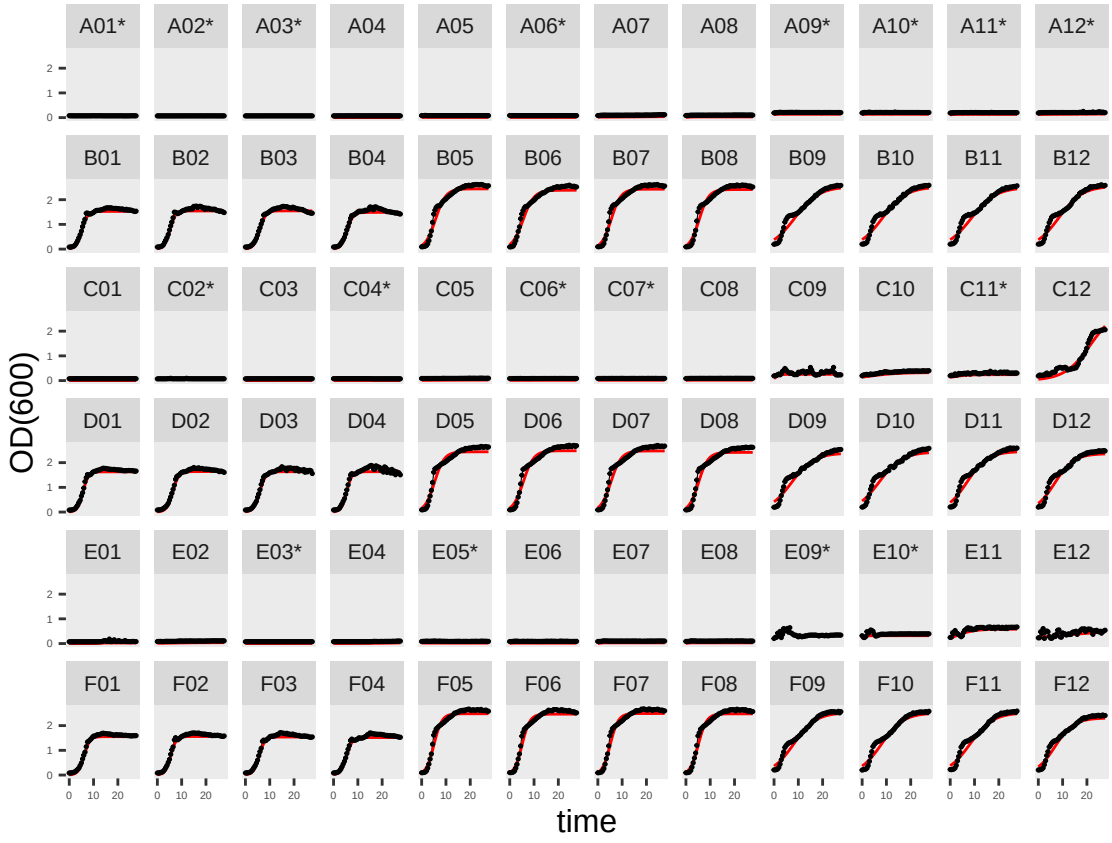

Fig 1 Plots of growth data (black) and logistic fit (red) for each well. Well names with asterisks were flagged as poor fits using the logistic model.

```
# output a growth summary table: Well, Treatment, Generation time, max cell yield.

# load my plate map data
datafile = "/home/tolonen/Github/actolonen/MicrobialGrowth/Growthcurver/2024.02.29_growthData.xlsx";
platemap = read_excel(datafile, sheet = "informations", col_names = TRUE, skip=30);

summarydata = left_join(fitdata, platemap, by="Well") %>%
  select(Well, Treatment, my_genTime, my_k) %>%
  rename(Generation_time = my_genTime) %>%
  rename(Carrying_Capacity = my_k) %>%
  mutate(Generation_time = round(Generation_time, digits = 2)) %>%
```

```
  mutate(Carrying_Capacity = round(Carrying_Capacity, digits = 2));

summarydata_stats = summarydata %>%
  group_by(Treatment) %>%
  mutate(Generation_Time_mean = mean(Generation_time)) %>%
  mutate(Generation_Time_sd = sd(Generation_time)) %>%
  mutate(Carrying_Capacity_mean = mean(Carrying_Capacity)) %>%
  select(Treatment, Generation_Time_mean, Generation_Time_sd, Carrying_Capacity_mean) %>%
  distinct();

table2 = kable(summarydata_stats, caption = "Table 2: treatment, generation time, and carrying capcity
table2
```

Table 2: Table 2: treatment, generation time, and carrying capcity of each well

Treatment	Generation_Time_mean	Generation_Time_sd	Carrying_Capacity_mean
MS + ClbP-17	1.0375	2.0750000	0.0000
MM29 + Clbp	127.1850	204.5578064	29007.6300
CMG + ClbP-17	6.3200	11.8203920	0.1425
MS + ClbP-17 + ECO.001	0.8125	0.0350000	1.5275
MM29 + ClbP-17 + ECO.001	1.4300	0.0909212	2.4125
CMG + ClbP-17 + ECO.001	3.4200	0.0605530	2.5200
MS + BCECF	1.6975	3.2889145	0.0000
MM29 + BCECF	4.7200	5.8233724	0.0175
CMG + BCECF	2.7575	1.6316939	0.9450
MS + BCECF + ECO.001	0.8050	0.0525991	1.6325
MM29 + BCECF + ECO.001	1.4200	0.0588784	2.4525
CMG + BCECF + ECO.001	3.0225	0.2530316	2.4075
MS + ClbP-17 + BCECF	5.9025	4.6463346	20873.5600
MM29 + ClbP-17 + BCECF	2.2700	3.9533867	0.0250
CMG + ClbP-17 + BCECF	3.6825	5.3782796	0.3700
MS + ClbP-17 + BCECF + ECO.001	0.8050	0.0238048	1.5425
MM29 + ClbP-17 + BCECF + ECO.001	1.1275	0.0330404	2.4775
CMG + ClbP-17 + BCECF + ECO.001	3.0550	0.2626151	2.4675
NA	0.0000	NA	0.0000

```
# focus on wells that contained microbial cells
pca_data = gc_fit %>%
  as_data_frame() %>%
  rename(Well = sample);

pca_gc_out = left_join(pca_data, platemap, by="Well");
pca_gc_out = pca_gc_out %>%
  filter(grepl("ECO.001", Treatment)) %>%
  select(-Treatment);

# Prepare the gc_out data for the PCA
rownames(pca_gc_out) = pca_gc_out$Well;
```

```
# Do the PCA on all growth paramters
pca.res = prcomp(pca_gc_out %>%
                   select(k:sigma), center=TRUE, scale=TRUE);

# Plot the results
plotpca = as_data_frame(list(PC1=pca.res$x[,1],
                   PC2=pca.res$x[,2],
                   samples = pca_gc_out$Well));

pcaplot = ggplot(plotpca, aes(x=PC1,y=PC2, label=samples)) +
  geom_text(size = 3);

pcaplot
```

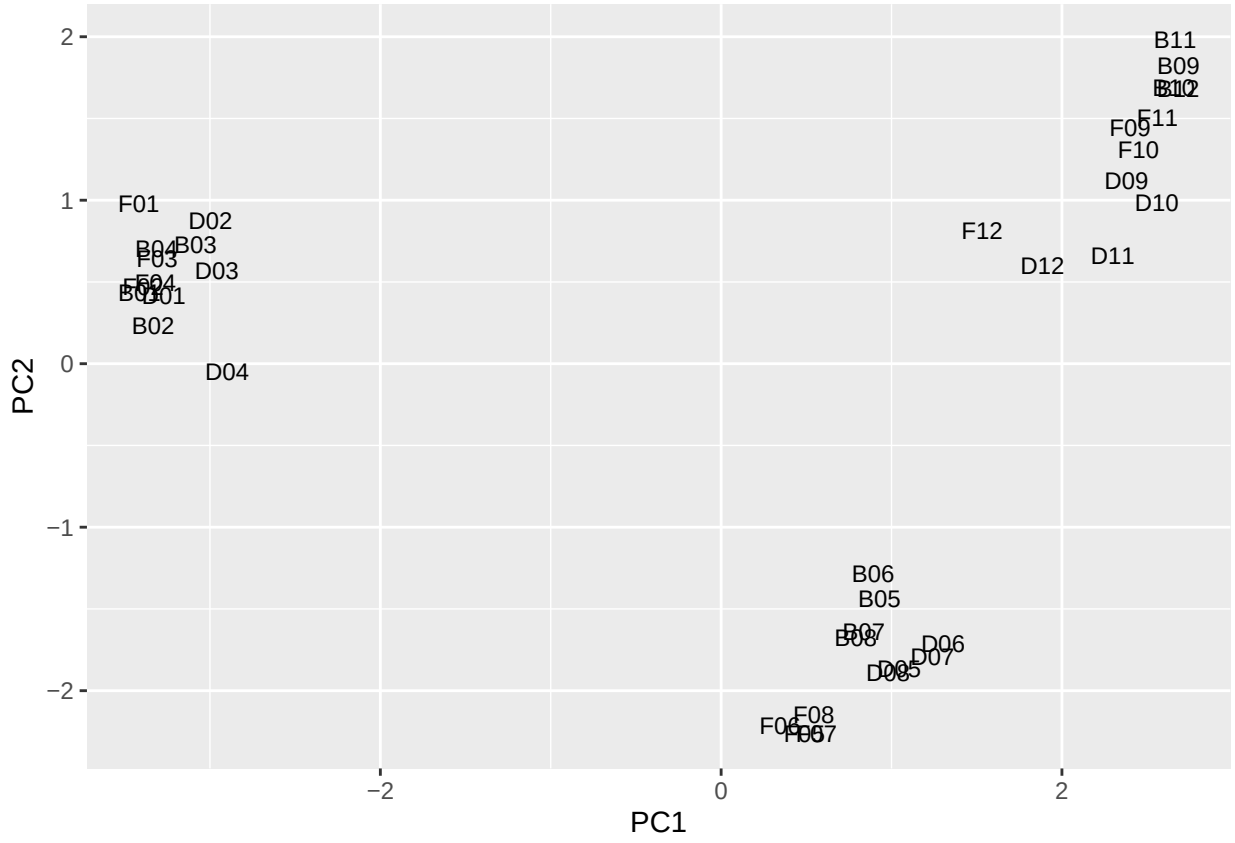

Fig 2 Principal component analysis on growth parameters to identify outlier wells.