Appendix B: growth inhibition

Jennifer Tran

2024-07-11

Load required packages and import data

```r
require("pacman")
```

```
## Loading required package: pacman
```

```r
p_load(data.table, dplyr, tidyr, purrr, DescTools)
```

I typically pre-format data from the Excel output of our Tecan MPlex. Here, I've appended about 26 plates worth of growth curves together for ease of importing into R. You can view a glimpse of the data here

```
## Rows: 10
## Columns: 10
## $ plate1_time          <dbl> 0.0, 894.1, 1788.1, 2682.1, 3576.2, 4470.2, 5364.~
## $ plate1_fos_256_1     <dbl> 0.0933, 0.0944, 0.0947, 0.0988, 0.1042, 0.1097, 0~
## $ plate1_fos_256_2     <dbl> 0.0877, 0.0964, 0.0981, 0.1000, 0.1048, 0.1131, 0~
## $ plate1_fos_256_3     <dbl> 0.0845, 0.0904, 0.0915, 0.0957, 0.0984, 0.1024, 0~
## $ plate1_fos_256_blank <dbl> 0.0831, 0.0831, 0.0871, 0.0787, 0.0806, 0.0851, 0~
## $ plate1_fos_128_1     <dbl> 0.0991, 0.0999, 0.1051, 0.1181, 0.1196, 0.1205, 0~
## $ plate1_fos_128_2     <dbl> 0.0970, 0.0958, 0.0979, 0.1283, 0.1145, 0.1139, 0~
## $ plate1_fos_128_3     <dbl> 0.0923, 0.0901, 0.0929, 0.0955, 0.1003, 0.1067, 0~
## $ plate1_fos_128_blank <dbl> 0.0830, 0.0825, 0.0796, 0.0796, 0.0793, 0.0810, 0~
## $ plate1_fos_64_1      <dbl> 0.0957, 0.0941, 0.0957, 0.0985, 0.1153, 0.1131, 0~
```

. . . but it requires some extra data wrangling as a result

```r
split_df_by_plate <- function(df) {
  plate_numbers <- unique(gsub("(_.*)|(_time)", "", names(df)))
  plate_dfs <- map(plate_numbers, function(plate) {
    df %>%
      select(matches(paste0(plate, "_"))) %>%
      rename_with(~ gsub(paste0(plate, "_"), "", .), -matches(paste0(plate, "_time"))) %>%
      rename(time = matches(paste0(plate, "_time")))
  })
  names(plate_dfs) <- plate_numbers
  plate_dfs
}
```

Function for subtracting off background

Subtract off background absorbance values from media only Media only controls here also include the antibiotic concentration as some antibiotics actually have additional noise at OD600/A600. In two cases, media only controls got contaminated towards the end of the growth curve. In those cases, controls from the antibiotic concentration one higher or lower were used.

```
subtract_blanks <- function(plate_df) {
  blank_cols <- grep("blank", names(plate_df), value = TRUE)
  plate_df <- plate_df %>%
    mutate(across(-time, ~ . - rowMeans(across(all_of(blank_cols)))))
  plate_df <- plate_df %>%
    select(-matches("blank"))
  plate_df
}
```

Function for calculating area under the curve

This uses the base AUC() function from the pracma package. It defaults to the trapezoidal method.

```
calculate_auc <- function(plate_df) {
  plate_df %>%
    pivot_longer(-time, names_to = c("antibiotic", "concentration", "replicate"), names_sep = "_") %>%
    group_by(antibiotic, concentration, replicate) %>%
    summarise(auc = AUC(time, value), .groups = 'drop')
}
```

Function for calculating percent inhibition

Each plate contains a replicate control with no antibiotic. Nearly every plate actually use a replicate control in the same column as the corresponding antibiotic sample.

Take the average for each antibiotic concentration across the three replicates. The standard deviation can give an idea of how much the growth curves varied between replicates.

```
calculate_percent_inhibition <- function(auc_df) {
  control_auc <- auc_df %>% filter(concentration == 0) %>% select(antibiotic, replicate, auc)
  auc_df <- auc_df %>%
    left_join(control_auc, by = c("antibiotic", "replicate"), suffix = c("", "_control")) %>%
    mutate(percent_inhibition = 100 * (1 - (auc / auc_control))) %>%
    select(-auc_control)
  auc_df
}

summarize_percent_inhibition <- function(inhibition_df, plate_number) {
  inhibition_df %>%
    group_by(antibiotic, concentration) %>%
    summarise(
      mean_percent_inhibition = mean(percent_inhibition),
      sd_percent_inhibition = sd(percent_inhibition),
      .groups = 'drop'
    ) %>%
```

```r
    mutate(plate_number = plate_number)
}
```

Process

```r
process_growth_curves <- function(df) {
  plate_dfs <- split_df_by_plate(df)
  inhibition_summaries <- map2_dfr(plate_dfs, names(plate_dfs), function(plate_df, plate_number) {
    plate_df <- subtract_blanks(plate_df)
    auc_df <- calculate_auc(plate_df)
    inhibition_df <- calculate_percent_inhibition(auc_df)
    summarize_percent_inhibition(inhibition_df, plate_number)
  })
  return(inhibition_summaries)
}

result_df <- process_growth_curves(df)
```

```
## Rows: 359
## Columns: 5
## $ antibiotic              <chr> "fos", "fos", "fos", "fos", "fos", "fos", "ami~
## $ concentration           <chr> "0", "128", "16", "256", "32", "64", "0", "1",~
## $ mean_percent_inhibition <dbl> 0.000000, 84.076541, 6.029997, 93.460875, 10.3~
## $ sd_percent_inhibition   <dbl> 0.00000000, 2.99277699, 1.19134659, 1.94497853~
## $ plate_number            <chr> "plate1", "plate1", "plate1", "plate1", "plate~
```

I like to clean it up by removing the no antibiotic controls. They're all zero percent inhibition because of how inhibition was calculated.

```r
cleaned_results <- result_df %>% filter(concentration != 0)
```