Hydrolytic Extracellular Enzyme Activity Calculations

Hava Blair

June 25, 2020

Contents

1 Setup

1.1 Templates

Start by filling out your template files and metadata file.

You need one completed **template file** for each plate, in CSV format. There is a template in the project folder. There are templates for both **Plate A** and **Plate B** with plate layouts pre-filled for you (check to make sure they match what you did!). The template is in the project folder, file name is "template_plate_a_hydrolytic.csv" for A plates and "template_plate_b_hydrolytic.csv"

You also need one completed **metadata file** (data for all plates you are analyzing with this script can be included in the same file). The template is in the project folder, file name "template_metadata.csv"

Name your template files with the ID of your plate. This will be added as its own column by plater so that you can identify which plate the data is coming from.

Files used with this version of the script should contain the following four blocks of information:

–"template" = standard, blank, and sample IDs. Empty wells (no sample, standard, or check) may be coded as ".", "0" (zero), "NA", or left blank.

–"data" = raw fluorescence data from the plate reader

–"bad_wells" = identifies any wells that have known problems (ex. pipetting errors). May be coded as "bad", "Bad", "x", or "X". Anything with NA (blank) will be kept as "good"

– "conc_uM" = identifies the concentration of the substrate or MUB used in the well. The concentration should be in umol/L (uM). Examples: sub_300 = 300 uM enzyme substrate. mub_1.25 = 1.25 uM MUB standard.

1.2 File organization

The files you want to process must all be in the **plate-templates subfolder**, and that folder must not contain any other files R will attempt to read in all files in the designated folder.

Now you are ready to read in your files. If you get an error, check whether you have any extra CSV files in the designated folder.

1.3 Read in plate templates & metadata

```
# read in completed template files from subfolder
file.names <- dir("./plater-templates/")
file.paths <- paste0("./plater-templates/", file.names)
plates <- read_plates(file.paths)
colnames(plates) <- c("plate", "wells", "id", "fluor", "bad_wells",
    "conc_uM")
unique(plates$plate)  # To check that all plates read in correctly
```

```
##  [1] "mg_2006" "mg_2008" "mg_2014" "mg_2016" "mg_2022" "mg_2024" "mg_2050"
##  [8] "mg_2051" "mg_2082" "mg_2083" "mg_2090" "mg_2092" "plate_b"
```

```r
# read in plate metadata
plate_metadata <- read.csv("./eea_metadata_mgtest.csv", stringsAsFactors = FALSE)

colnames(plate_metadata) <- c("plate", "ph_buffer", "moist_soil_mass_g",
    "vol_buffer_ml", "substrates", "tin", "tin_moist", "tin_dry",
    "time_soil_added", "time_naoh_added", "time_plate_read")
```

2 Metadata calculations

```r
# parse the time columns so we can do math on them
plate_metadata <- plate_metadata %>% mutate(time_soil_added = parse_hm(time_soil_added),
    time_naoh_added = parse_hm(time_soil_added), time_plate_read = parse_hm(time_plate_read))

# calculate incubation time and convert from seconds to hours
plate_metadata <- plate_metadata %>% mutate(inc_time_hr = as.numeric(((time_plate_read -
    time_soil_added)/60/60)))

# calculate soil moisture content and dry soil equivalent
plate_metadata <- plate_metadata %>% mutate(mc_soil_moist = tin_moist -
    tin, mc_soil_dry = tin_dry - tin, soil_water_content = (mc_soil_moist -
    mc_soil_dry)/mc_soil_dry, soil_ov_dry_eq_g = moist_soil_mass_g -
    (moist_soil_mass_g * soil_water_content))
```

3 Initial Quality control

3.1 Remove empty wells and known bad wells

```r
# Remove empty wells
no_missing <- subset(plates, !is.na(id) & id != 0 & id != ".")

# Remove bad wells
no_bad <- subset(no_missing, is.na(bad_wells))

# Create a dataframe with the details about which wells were
# removed as 'bad'.  Can write this to CSV now, or wait until
# after later QC checks to compile a list of samples that
# need to be redone.
bad_list <- subset(no_missing, !is.na(bad_wells))

# print a message about how many wells were removed
wells_removed <- nrow(plates) - nrow(no_bad)

glue("Removed {wells_removed} wells that were missing or bad")
```

```
## Removed 193 wells that were missing or bad
```

```
# filter out plate b here - it will be used for the emission
# coefficient calculation
plate_b <- no_bad %>% filter(plate == "plate_b")

# clean_data contains all of the plate a data
clean_data <- no_bad %>% filter(plate != "plate_b")

# nest data so we can keep our calculations organized nicely
# by plate
clean_nested <- clean_data %>% group_by(plate) %>% nest()
```

4 Standards calculations

4.1 B Plate: MUB standard calculations (for emission coefficient calculation)

```
# exract MUB concentrations from the conc_uM column and drop
# the empty 'bad_wells' column
plate_b <- plate_b %>% mutate(conc_uM = as.numeric(str_replace(conc_uM,
    "mub_", ""))) %>% select(-bad_wells)

# calculate the mean fluorescence across technical replicates
# for each MUB concentration
plate_b_means <- plate_b %>% group_by(id) %>% filter(str_detect(id,
    "MUB")) %>% summarise(mub_std_fluor = mean(fluor), n = n())

# calculate the plate blank (buffer only)
plate_b_blank <- plate_b %>% filter(str_detect(id, "Buf")) %>%
    summarise(fluor_blank = mean(fluor), n = n())

fluor_blank <- plate_b_blank$fluor_blank

# subtract the fluorescence of the plate blank (id = Buf)
# from the mean fluorescence of the standards
plate_b_means <- plate_b_means %>% mutate(plate_blank = fluor_blank,
    corr_fluor = mub_std_fluor - plate_blank, mub_conc_uM = as.numeric(str_replace(id,
        "MUB", "")))

plate_b_means
```

```
## # A tibble: 4 x 6
##   id       mub_std_fluor     n plate_blank corr_fluor mub_conc_uM
##   <chr>            <dbl> <int>       <dbl>      <dbl>       <dbl>
## 1 MUB0.16          1486.     4        66.1      1420.       0.16
## 2 MUB0.625         5891.     4        66.1      5825.       0.625
## 3 MUB1.25         11928.     4        66.1     11862.       1.25
## 4 MUB2.5          22628      4        66.1     22562.       2.5
```

4.2 A Plates: Calculate plate blank and homogenate blank

```
# function to calculate plate blanks (Buffer only)
get_plate_blank <- function(data) {
    buf <- data %>% filter(id == "Buf")
    mean(buf$fluor)
}

# apply function to nested df
with_a_blanks <- clean_nested %>% mutate(plate_blank = map_dbl(data,
    get_plate_blank))

# function to calculate the fluor for HOMogenate BLanks
# (hombl = buffer + homogenate)
get_hombl <- function(data) {
    buf_soil <- data %>% filter(str_detect(id, "Hombl"))
    mean(buf_soil$fluor)
}

# apply function to nested df
calc_hombl <- with_a_blanks %>% mutate(hombl = map_dbl(data,
    get_hombl))

head(calc_hombl)
```

```
## # A tibble: 6 x 4
## # Groups:   plate [6]
##   plate   data              plate_blank hombl
##   <chr>   <list>                  <dbl> <dbl>
## 1 mg_2006 <tibble [80 x 5]>        61.2  113.
## 2 mg_2008 <tibble [80 x 5]>        60.8  116.
## 3 mg_2014 <tibble [80 x 5]>        68.6  131.
## 4 mg_2016 <tibble [80 x 5]>        59.2  125
## 5 mg_2022 <tibble [80 x 5]>        64.4  121.
## 6 mg_2024 <tibble [80 x 5]>        58.6  128.
```

4.3 Plate A: Calculate mean fluor of the MUB standard + soil wells (for quench coeff calculation)

```
# function to average the MUB+soil (quench) standard wells to
# get mean fluorescence for each MUB concentration

quench_std_fun <- function(data, hombl) {
    data %>% select(wells, id, fluor, conc_uM) %>% filter(str_detect(id,
        "quench")) %>% mutate(conc_uM = as.numeric(str_replace(conc_uM,
        "mub_", ""))) %>% group_by(conc_uM) %>% summarise(quench_fluor = mean(fluor),
        n = n(), hombl = hombl, corr_fluor = quench_fluor - hombl)  #also corrects for hom blank here
}

# apply function to nested df
```

```
quench_nested <- calc_hombl %>% mutate(quench_std_values = map2(data,
    hombl, quench_std_fun))

head(quench_nested)
```

```
## # A tibble: 6 x 5
## # Groups:   plate [6]
##   plate   data              plate_blank hombl quench_std_values
##   <chr>   <list>                  <dbl> <dbl> <list>
## 1 mg_2006 <tibble [80 x 5]>        61.2  113. <tibble [4 x 5]>
## 2 mg_2008 <tibble [80 x 5]>        60.8  116. <tibble [4 x 5]>
## 3 mg_2014 <tibble [80 x 5]>        68.6  131. <tibble [4 x 5]>
## 4 mg_2016 <tibble [80 x 5]>        59.2  125  <tibble [4 x 5]>
## 5 mg_2022 <tibble [80 x 5]>        64.4  121. <tibble [4 x 5]>
## 6 mg_2024 <tibble [80 x 5]>        58.6  128. <tibble [4 x 5]>
```

5 Linear models

5.1 Linear model functions

```
# function to run a linear model x = MUB concentration, y =
# mean fluorescence

lm_mod_ftn <- function(df) {
    lm(corr_fluor ~ conc_uM, data = df)
}

# functions to extract linear model details calculated above
# into a nicer format for putting in our graphs

b_fun <- function(mod) {
    coefficients(mod)[[1]]
}

slope_fun <- function(mod) {
    coefficients(mod)[[2]]
}

r_sq_fun <- function(mod) {
    summary(mod)[["r.squared"]]
}

# leaving this max fluorescence function out for now because
# I'm not sure we need it.  max_fluor_fun <- function(data){
# max(data$corr_std_mean) }
```

5.2 A Plates: Linear model calcs for quench std curves

```
# calculate linear model for quench (A plates)

quench_lm_calcs <- quench_nested %>% mutate(quench_lm = map(quench_std_values,
    lm_mod_ftn))

# Extract linear models details from the homogenate control
# linear model
quench_lm_details <- quench_lm_calcs %>% mutate(intcpt_quench = map_dbl(quench_lm,
    b_fun), slope_quench = map_dbl(quench_lm, slope_fun), r_squared_quench = map_dbl(quench_lm,
    r_sq_fun))

# nest the linear model details in a dataframe
nest_quench_stats <- quench_lm_details %>% nest(lm_stats_quench = c(intcpt_quench,
    slope_quench, r_squared_quench))

quench_lm_details %>% select(plate, intcpt_quench, slope_quench,
    r_squared_quench)
```

```
## # A tibble: 12 x 4
## # Groups:   plate [12]
##    plate   intcpt_quench slope_quench r_squared_quench
##    <chr>           <dbl>        <dbl>            <dbl>
##  1 mg_2006          42.0        3847.            1.00
##  2 mg_2008          56.0        3164.            0.998
##  3 mg_2014          15.4        4345.            1.00
##  4 mg_2016          76.6        3174.            1.00
##  5 mg_2022          58.0        2773.            1.00
##  6 mg_2024          27.1        2576.            1.00
##  7 mg_2050          32.5        2974.            1.00
##  8 mg_2051         104.         2910.            0.998
##  9 mg_2082          53.5        3152.            0.999
## 10 mg_2083         -36.5        3125.            1.00
## 11 mg_2090          21.3        4015.            1.00
## 12 mg_2092          69.3        3741.            1.00
```

5.3 B Plate: Linear model calculations

```
# fix name of mub concentration column so it works with the
# lm function
plate_b_rename <- plate_b_means %>% rename(conc_uM = mub_conc_uM)

# apply linear model ftn to plate b data
lm_plate_b <- lm_mod_ftn(plate_b_rename)

# save linear model details
intcpt_emis <- b_fun(lm_plate_b)
slope_emis <- slope_fun(lm_plate_b)
rsq_emis <- r_sq_fun(lm_plate_b)

# add linear model details to plate b dataframe
lm_stats_emis <- plate_b_rename %>% mutate(intcpt_emis = intcpt_emis,
```

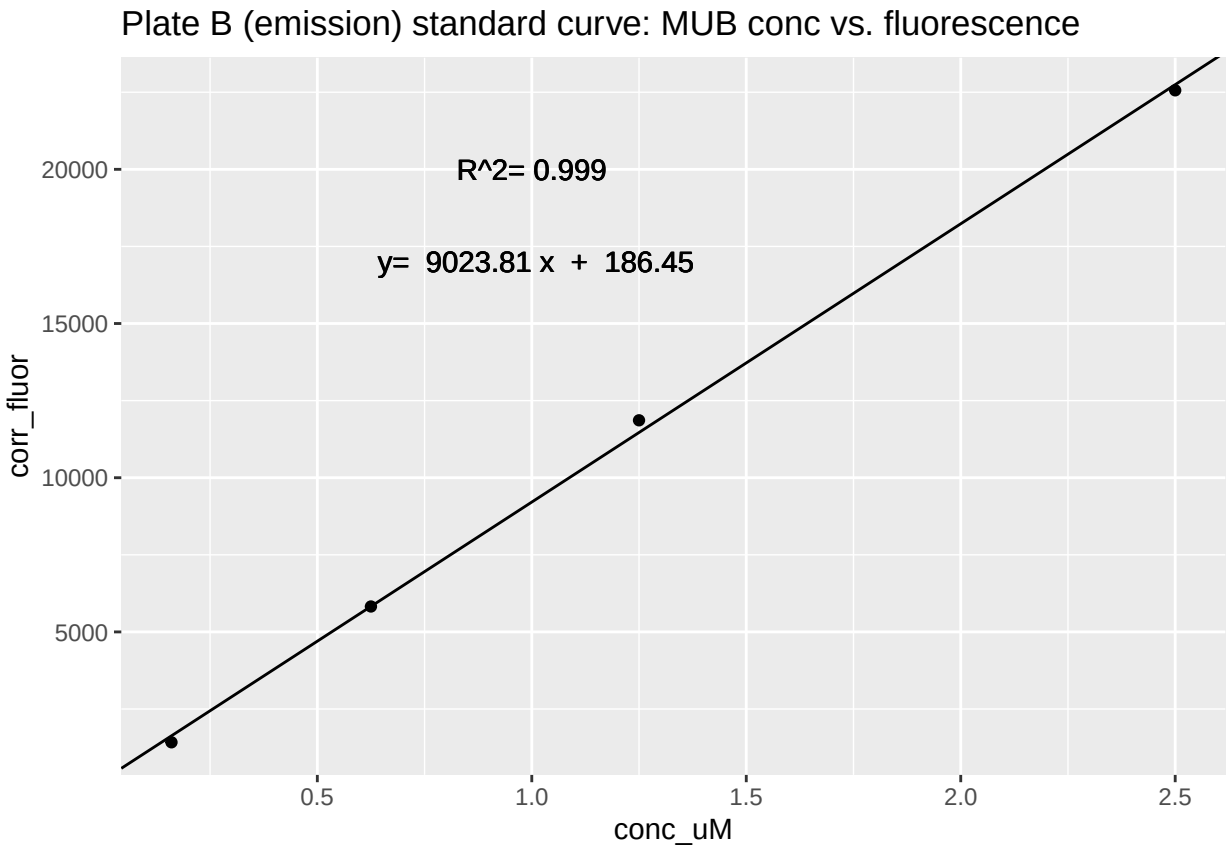
```
    slope_emis = slope_emis, rsq_emis = rsq_emis)

head(lm_stats_emis)
```

```
## # A tibble: 4 x 9
##   id    mub_std_fluor     n plate_blank corr_fluor conc_uM intcpt_emis
##   <chr>         <dbl> <int>       <dbl>      <dbl>   <dbl>       <dbl>
## 1 MUB0~         1486.     4        66.1      1420.   0.16         186.
## 2 MUB0~         5891.     4        66.1      5825.   0.625        186.
## 3 MUB1~        11928.     4        66.1     11862.   1.25         186.
## 4 MUB2~        22628      4        66.1     22562.   2.5          186.
## # ... with 2 more variables: slope_emis <dbl>, rsq_emis <dbl>
```

6 Standard curve graphs

6.1 B Plate: Plot standard curve (emission)

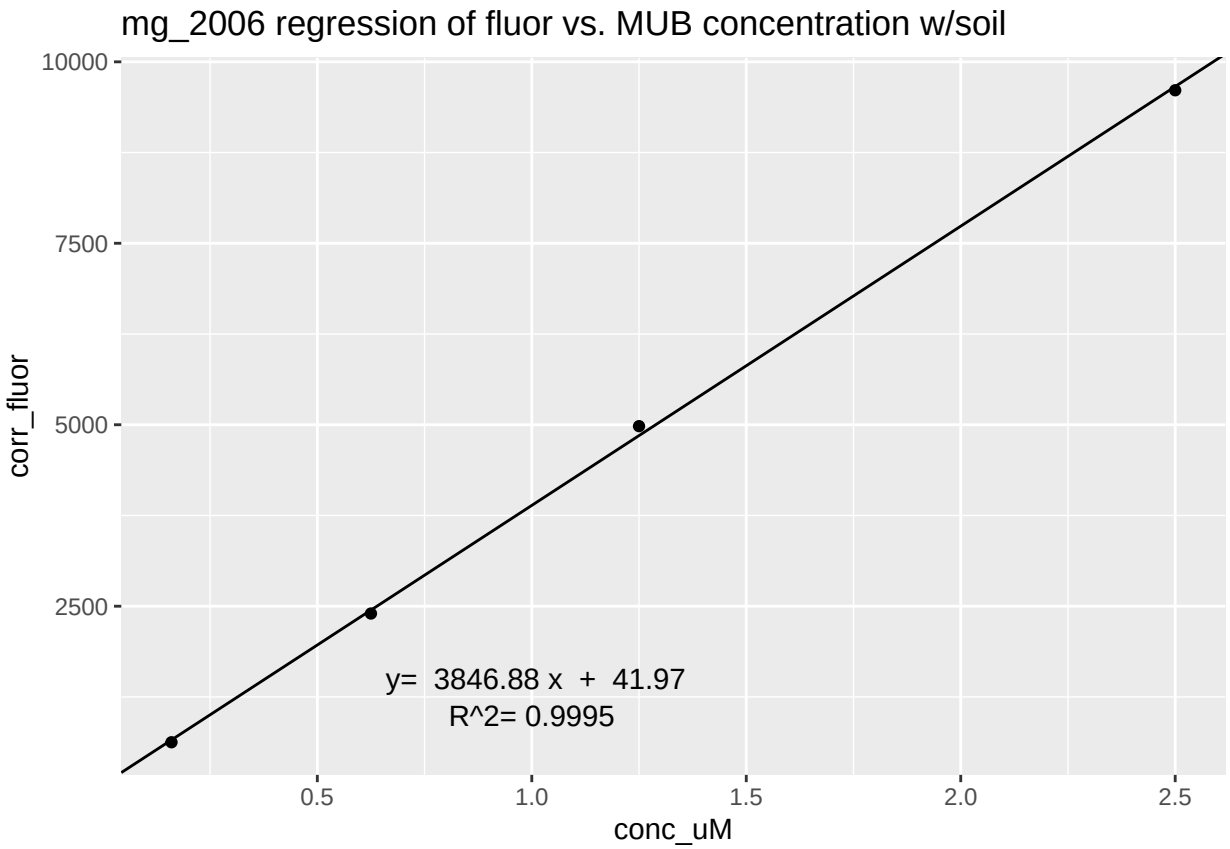
```
## consider writing this into a function so that multiple B
## plates (from multiple runs) could be processed together.

b_plot <- ggplot(lm_stats_emis) + geom_point(aes(x = conc_uM,
    y = corr_fluor)) + geom_abline(aes(slope = slope_emis, intercept = intcpt_emis)) +
    geom_text(data = lm_stats_emis, aes(x = 1, y = 20000, label = paste("R^2=",
        round(rsq_emis, digits = 3))), inherit.aes = FALSE) +
    geom_text(aes(x = 1, y = 17000, label = paste(" y= ", round(slope_emis,
        digits = 2), "x", " + ", round(intcpt_emis, digits = 2))),
        inherit.aes = FALSE) + labs(title = glue("Plate B (emission) standard curve: MUB conc vs. fluor

b_plot
```



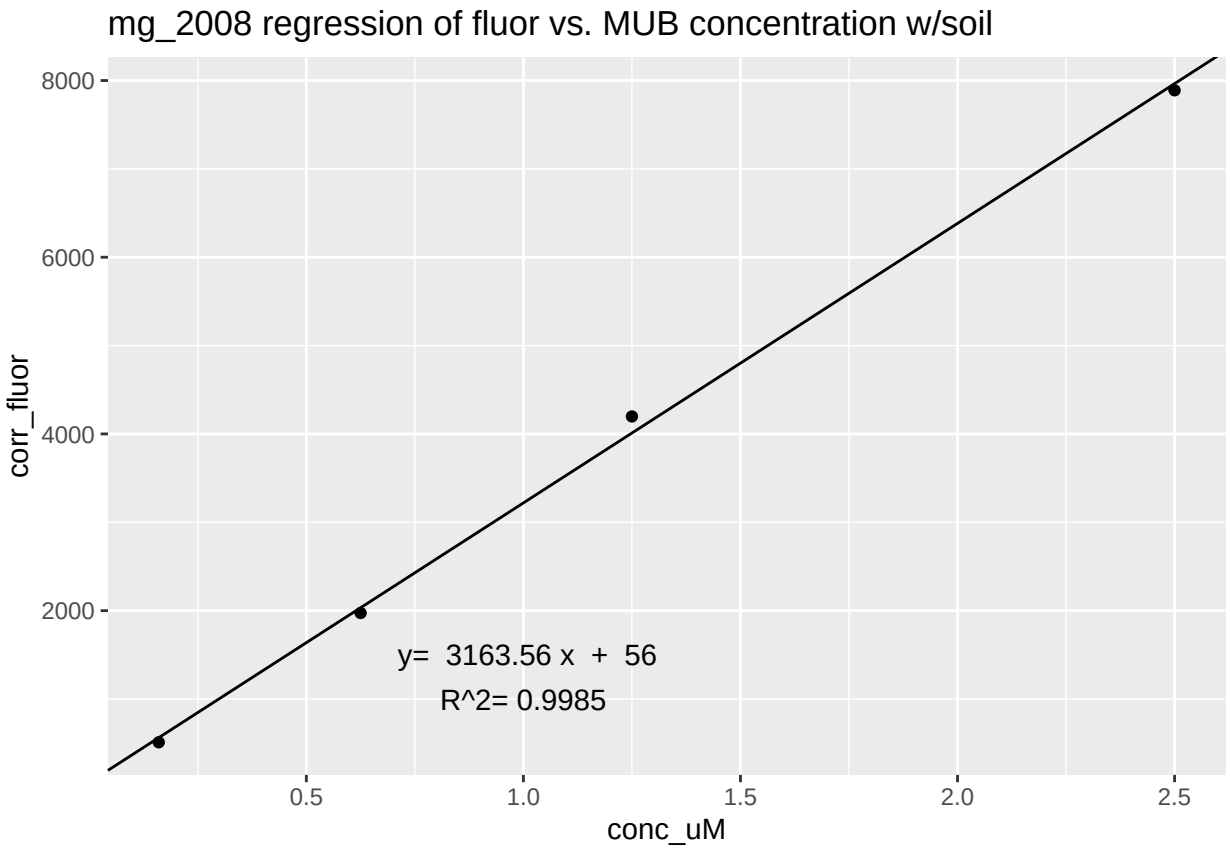
6.2 A Plate: plot quench standard curves

```
# plotting function
plot_homog_fun <- function(quench_std_values, lm_stats_quench,
    plate) {

    g <- ggplot() + geom_point(data = quench_std_values, aes(x = conc_uM,
        y = corr_fluor)) + geom_abline(data = lm_stats_quench,
        aes(slope = slope_quench, intercept = intcpt_quench)) +
        geom_text(data = lm_stats_quench, aes(x = 1, y = 1000,
            label = paste("R^2=", round(r_squared_quench, digits = 4))),
            inherit.aes = FALSE) + geom_text(data = lm_stats_quench,
        aes(x = 1, y = 1500, label = paste(" y= ", round(slope_quench,
            digits = 2), "x", " + ", round(intcpt_quench, digits = 2))),
        inherit.aes = FALSE) + labs(title = glue("{plate} regression of fluor vs. MUB concentration w/s
    return(g)
}

# run the plotting function for each plate and store the
# plots in our nested dataframe
homog_plots <- nest_quench_stats %>% mutate(plot_homog_control = pmap(list(quench_std_values,
    lm_stats_quench, plate), plot_homog_fun))

homog_plots$plot_homog_control
```

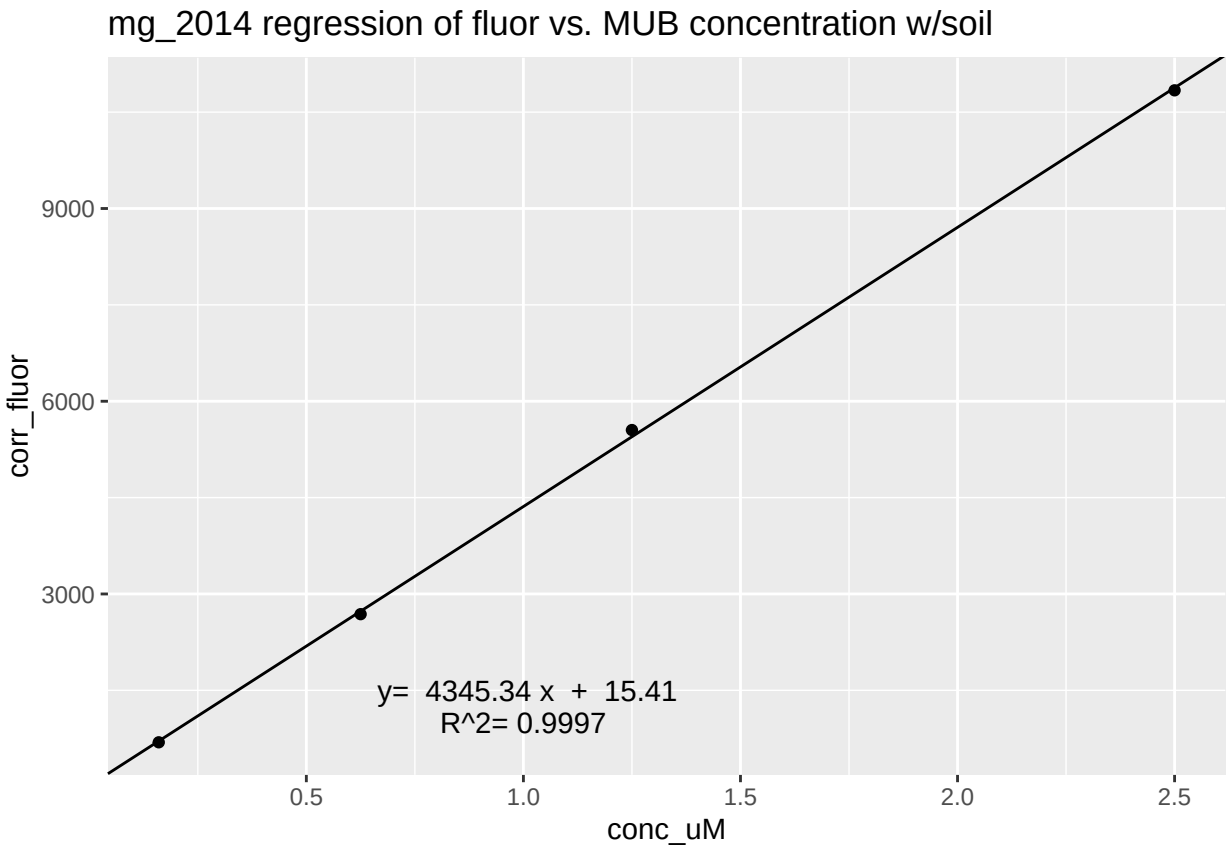
```
## [[1]]
```


mg_2006 regression of fluor vs. MUB concentration w/soil
corr_fluor
conc_uM
10000
7500
5000
2500
0.5
1.0
1.5
2.0
2.5
y= 3846.88 x + 41.97
R^2= 0.9995

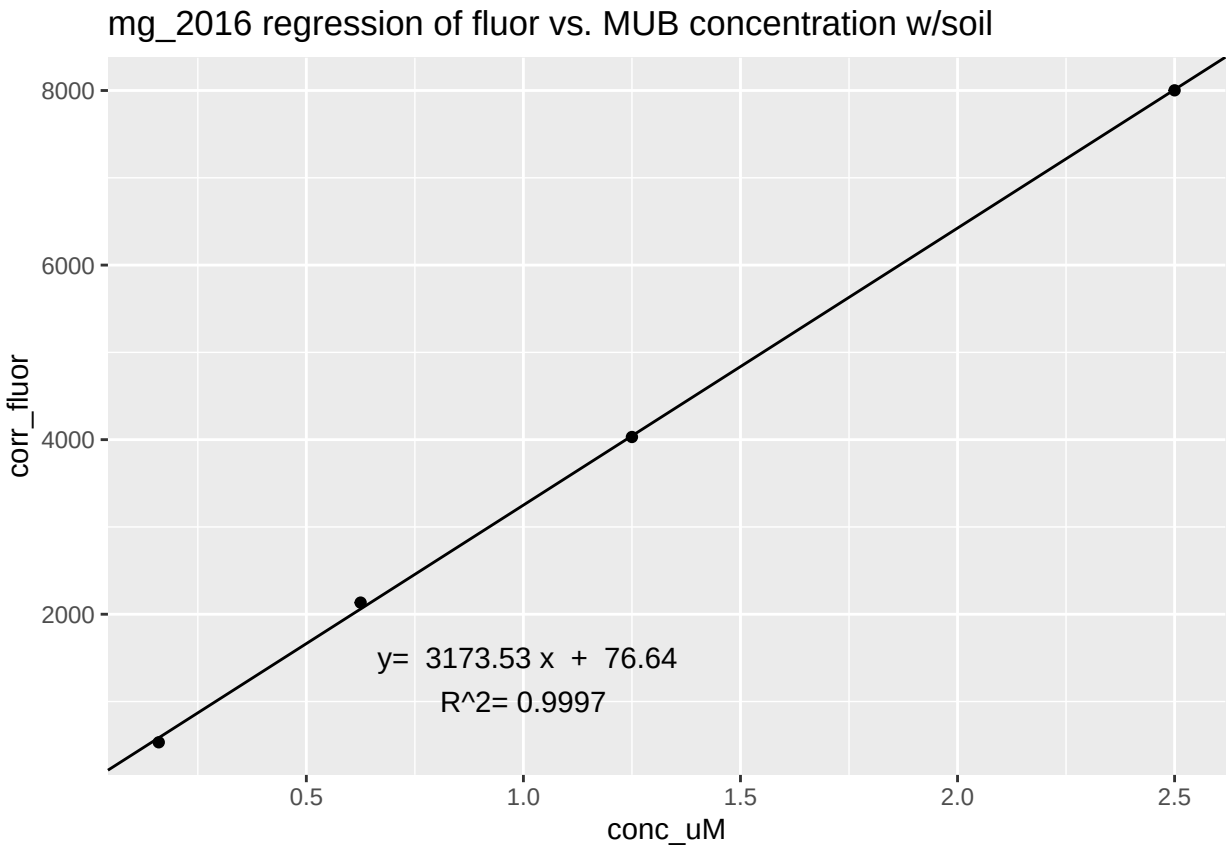
```
##
## [[2]]
```


mg_2008 regression of fluor vs. MUB concentration w/soil
corr_fluor
8000
6000
4000
2000
0.5
1.0
1.5
2.0
2.5
conc_uM
y= 3163.56 x + 56
R^2= 0.9985

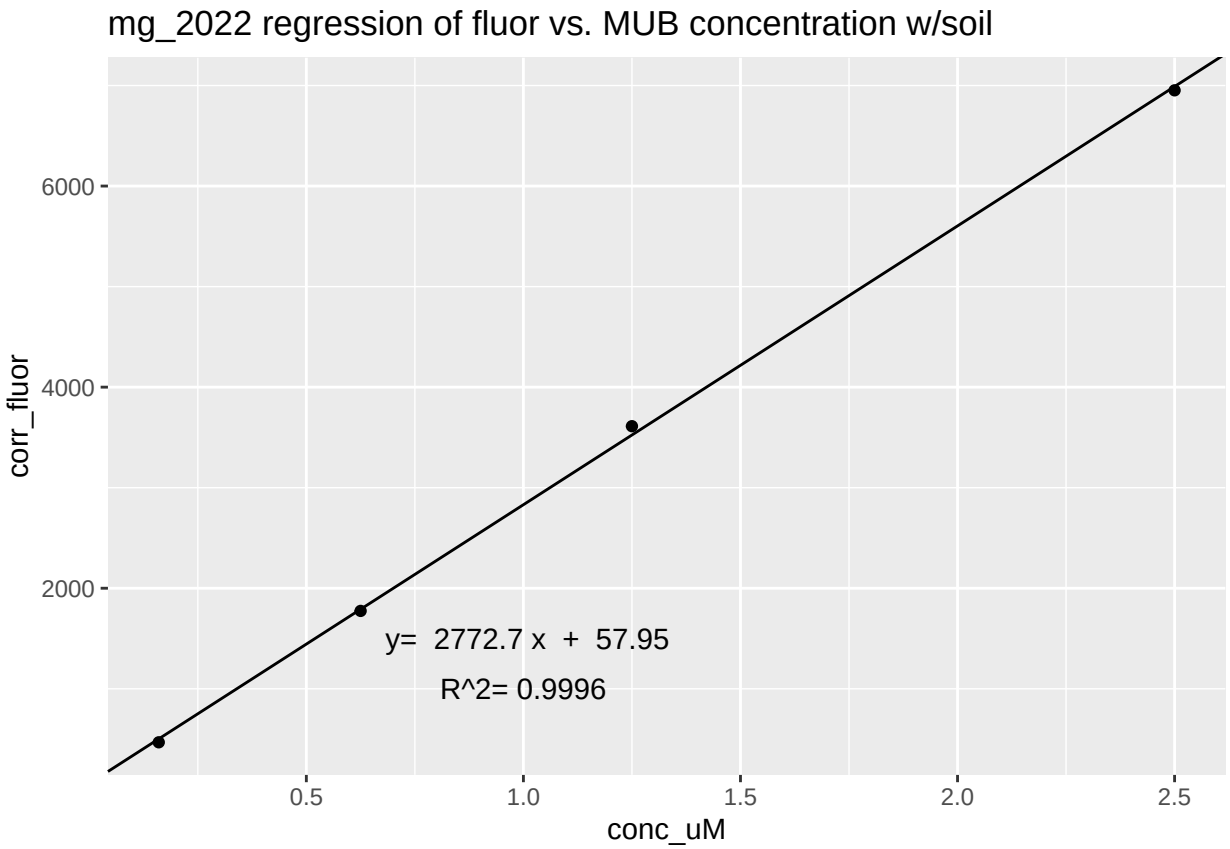
```
##
## [[3]]
```


mg_2014 regression of fluor vs. MUB concentration w/soil
corr_fluor
9000
6000
3000
0.5
1.0
1.5
2.0
2.5
conc_uM
y= 4345.34 x + 15.41
R^2= 0.9997

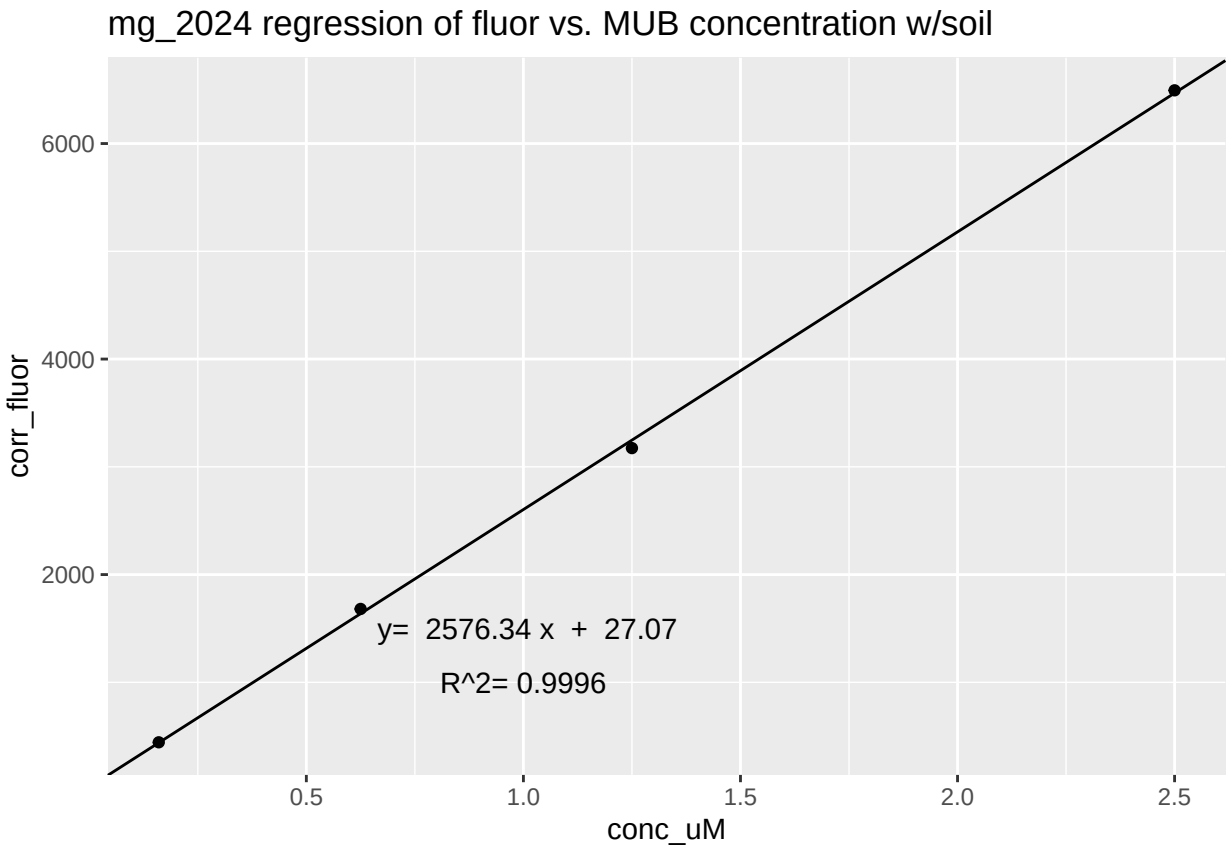
```
##
## [[4]]
```


mg_2016 regression of fluor vs. MUB concentration w/soil
corr_fluor
8000
6000
4000
2000
0.5
1.0
1.5
2.0
2.5
conc_uM
y= 3173.53 x + 76.64
R^2= 0.9997

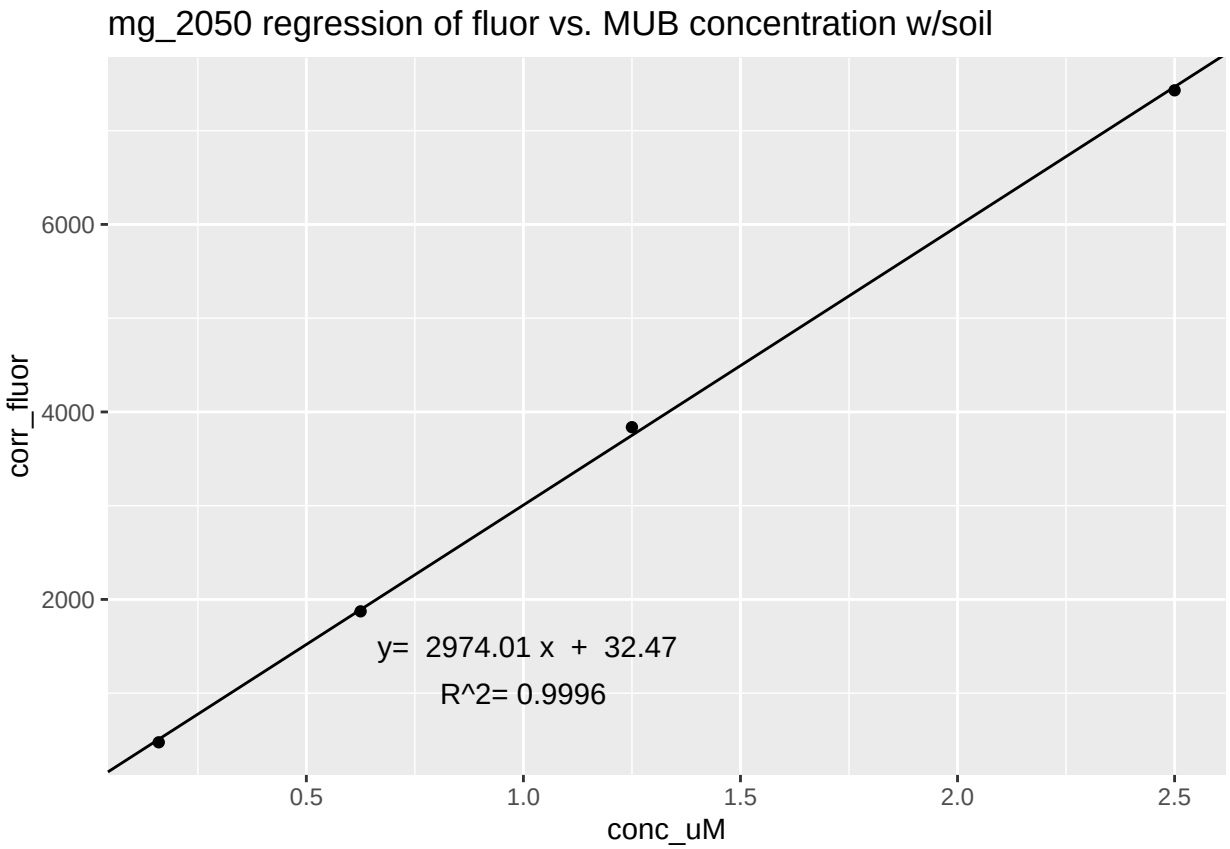
```
##
## [[5]]
```


mg_2022 regression of fluor vs. MUB concentration w/soil
corr_fluor
6000
4000
2000
0.5
1.0
1.5
2.0
2.5
conc_uM
y= 2772.7 x + 57.95
R^2= 0.9996

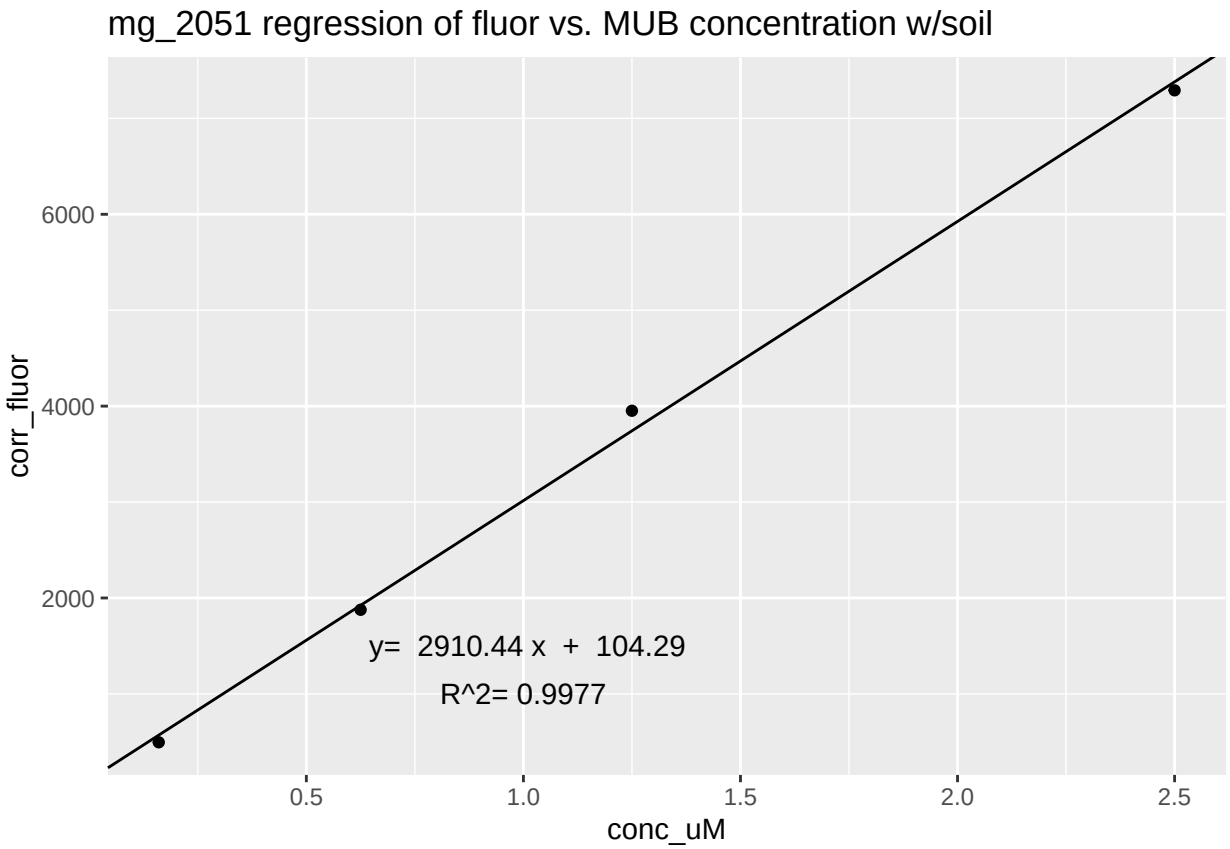
```
##
## [[6]]
```


mg_2024 regression of fluor vs. MUB concentration w/soil
corr_fluor
6000
4000
2000
0.5
1.0
1.5
2.0
2.5
conc_uM
y= 2576.34 x + 27.07
R^2= 0.9996

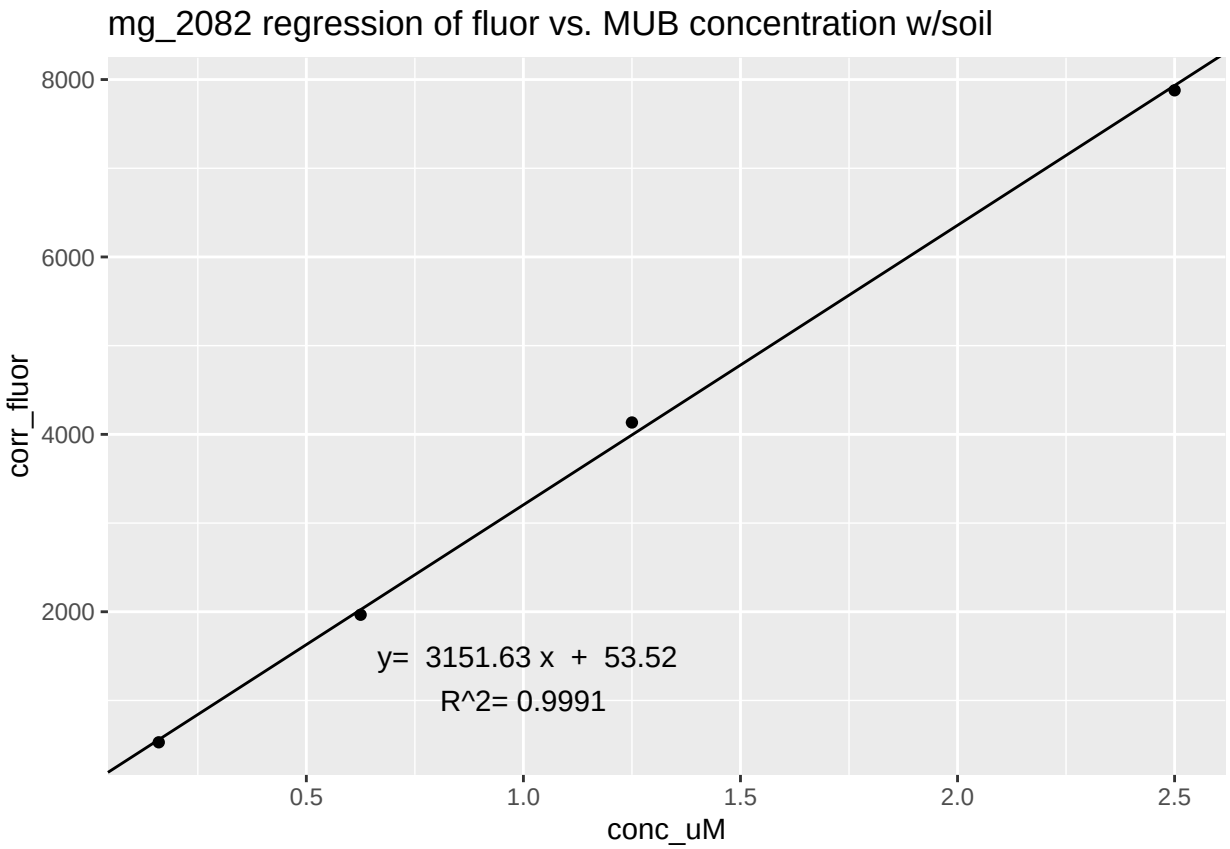
```
##
## [[7]]
```


mg_2050 regression of fluor vs. MUB concentration w/soil
corr_fluor
6000
4000
2000
y= 2974.01 x + 32.47
R^2= 0.9996
0.5
1.0
1.5
2.0
2.5
conc_uM

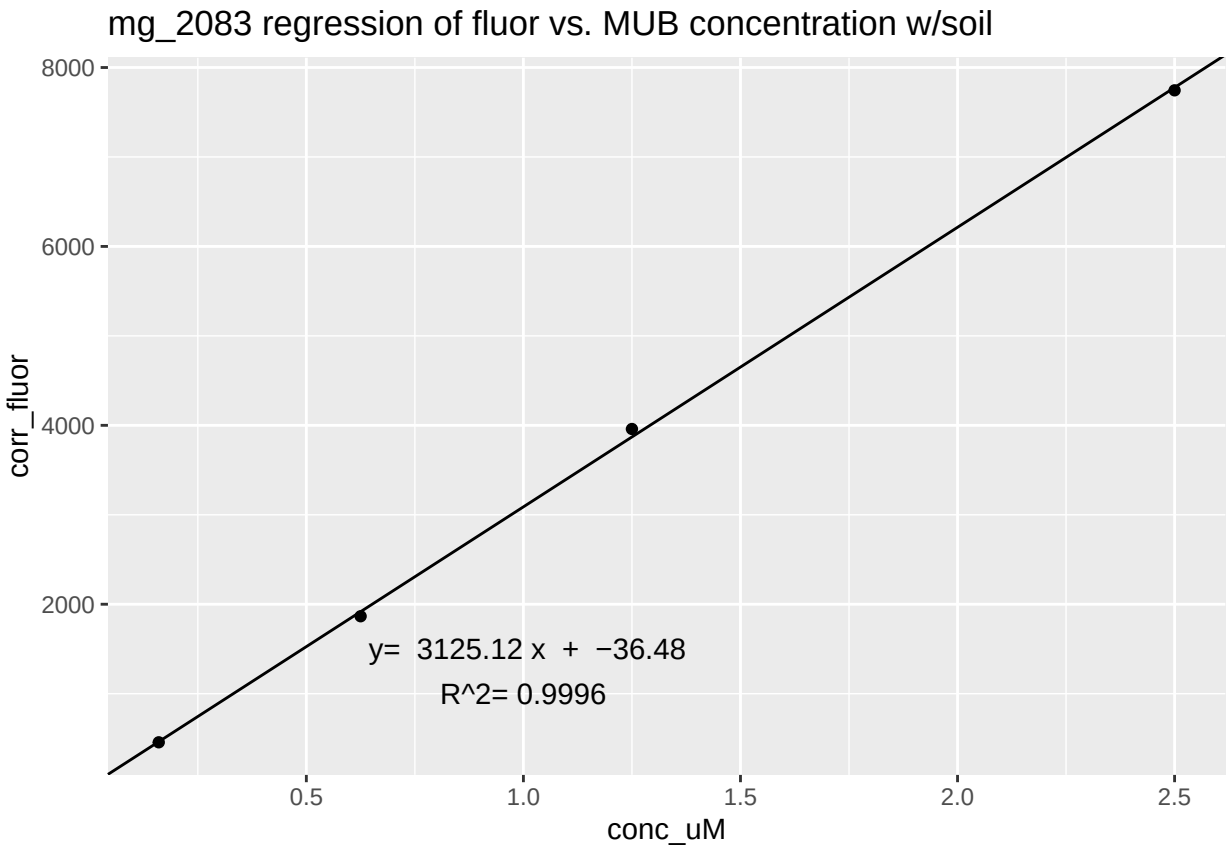
```
##
## [[8]]
```


mg_2051 regression of fluor vs. MUB concentration w/soil
corr_fluor
6000
4000
2000
y= 2910.44 x + 104.29
R^2= 0.9977
0.5
1.0
1.5
2.0
2.5
conc_uM

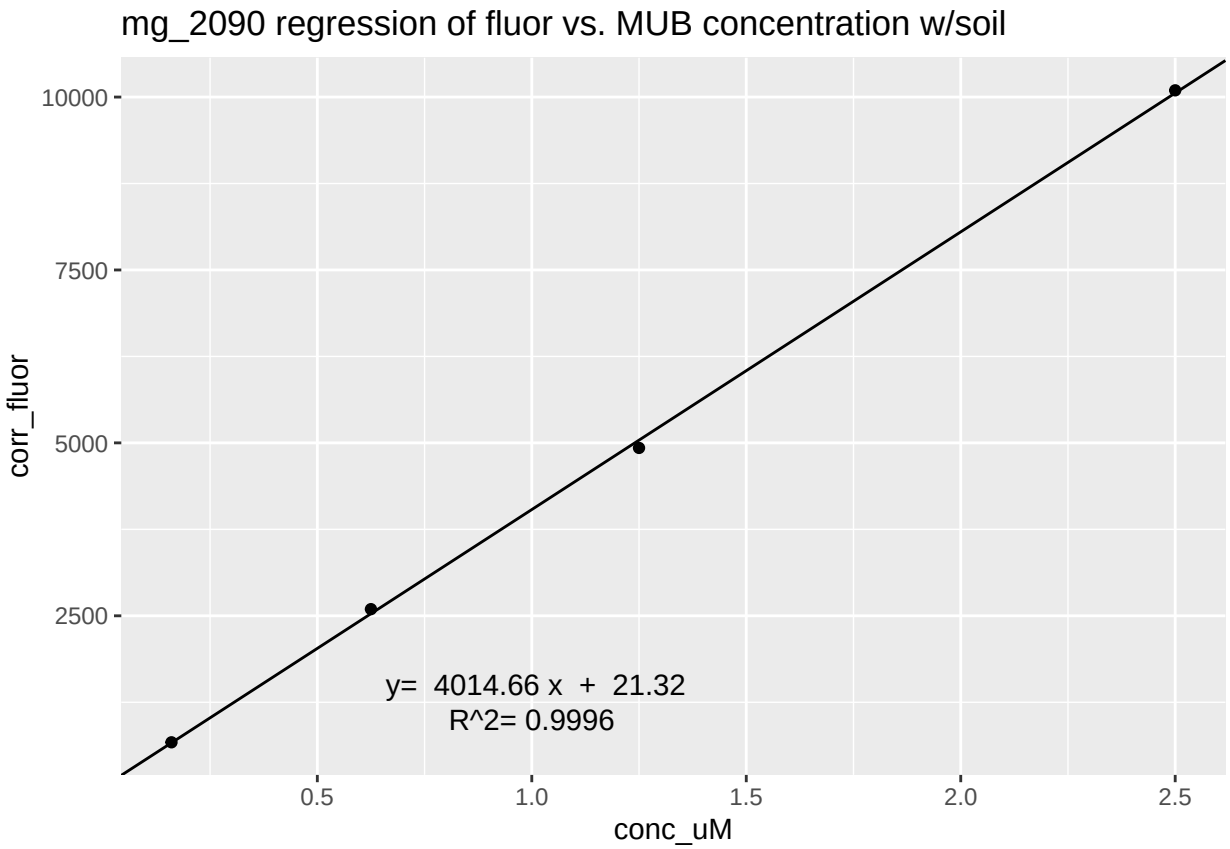
```
##
## [[9]]
```


mg_2082 regression of fluor vs. MUB concentration w/soil
corr_fluor
8000
6000
4000
2000
0.5
1.0
1.5
2.0
2.5
conc_uM
y= 3151.63 x + 53.52
R^2= 0.9991

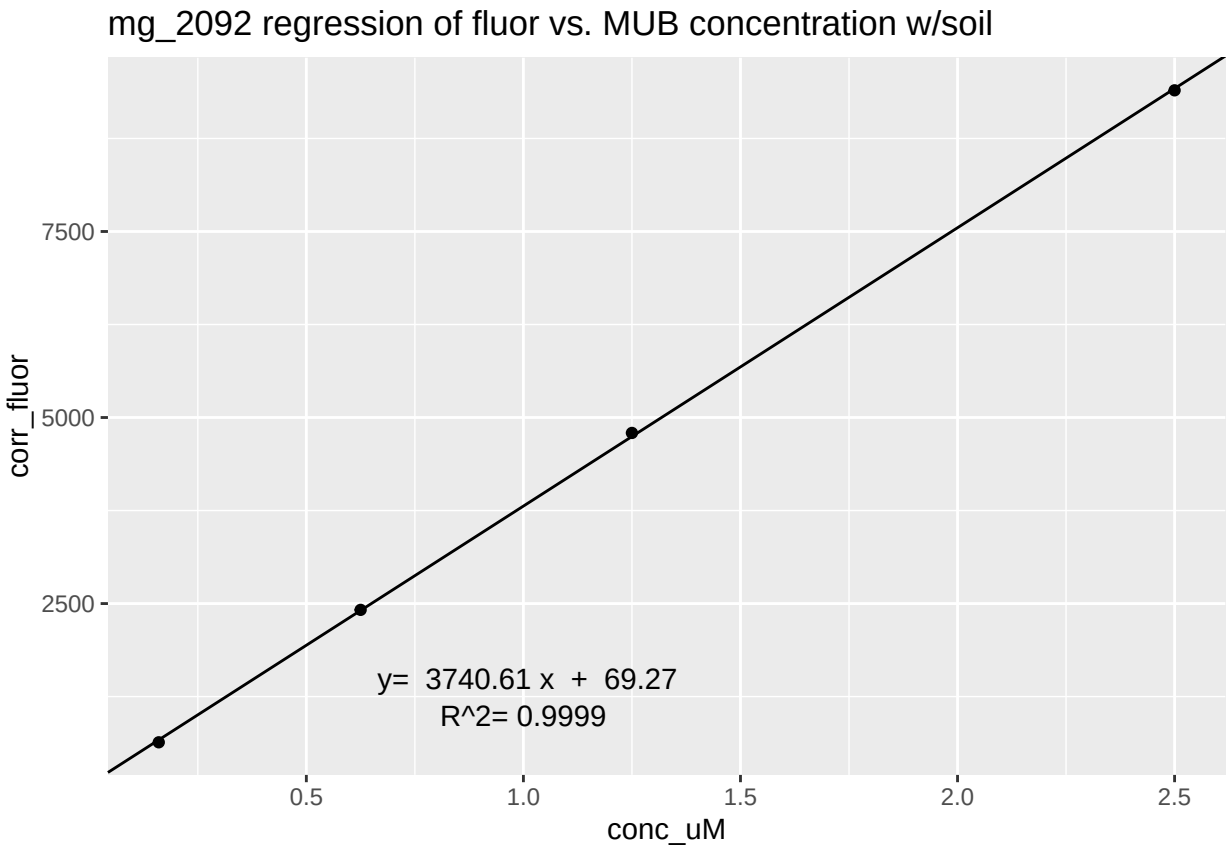
```
##
## [[10]]
```


mg_2083 regression of fluor vs. MUB concentration w/soil
8000
6000
4000
2000
corr_fluor
y= 3125.12 x + −36.48
R^2= 0.9996
0.5
1.0
1.5
2.0
2.5
conc_uM

```
##
## [[11]]
```


mg_2090 regression of fluor vs. MUB concentration w/soil
corr_fluor
10000
7500
5000
2500
y= 4014.66 x + 21.32
R^2= 0.9996
0.5
1.0
1.5
2.0
2.5
conc_uM

```
##
## [[12]]
```



7 Coefficient calculations

7.1 Calculate emission coefficient

The emission coefficient is the slope (m) from plate b (fluorescence vs MUB conc NO SOIL) divided by the assay volume

Pay attention to units! **slope (m) units** from the standard curves = fluor/uM = fluor/ (umol/L) = fluor L / umol = fluor mL / nmol

assay volume units are 250uL = 0.00025 L = 0.250 mL

emission coefficient units are fluorescence/nmol.

```
assay_vol_ml <- 0.25

# just grabbing the first element of slope_emis because the
# slope is the same for all 4 rows in this df ( we only have
# 1 B plate). In future consider writing a function that will
# return this for multiple B plates

slope_emis <- lm_stats_emis$slope_emis[1]

# calculate slope of the emission curve and the emisson
# coefficient
```

```
emis_calc <- homog_plots %>% mutate(slope_emis = slope_emis[1],
    emis_coeff = slope_emis/assay_vol_ml)
```

7.2 Calculate the quench coefficient

The quench coefficient is calculated as the slope of the quench standard curve divided by the slope of the buffer control (emission) standard curve.

Add more information about what the quench coefficient actually is...

```
coeff_calcs <- emis_calc %>% mutate(quench_coeff = map2_dbl(lm_stats_quench,
    slope_emis, ~as.numeric(.x[["slope_quench"]])/.y))
```

8 Assay well calculations

8.1 A Plates: Mean of technical reps for assay & substrate control wells

```
# Calculate the mean fluor for the assay wells & substrate
# control wells (note using the un-nested dataframe here)

assay_sub_blnks <- clean_data %>% filter(str_detect(id, "assay") |
    str_detect(id, "sub-blank")) %>% group_by(plate, id) %>%
    summarise(fluor_mean = mean(fluor), n = n(), sd = sd(fluor,
        na.rm = T), cv = sd(fluor, na.rm = T)/mean(fluor) * 100)

# format dataframe 'wide' so that substrate controls are in
# columns adjacent to assay wells for each enzyme
wide <- assay_sub_blnks %>% pivot_wider(id_cols = c(plate, id),
    names_from = id, values_from = fluor_mean, names_prefix = "mean_fluor_",
    names_sep = "_")

# grab relevant coeffs from nested dataframe
coeffs <- coeff_calcs %>% select(plate, plate_blank, hombl, emis_coeff,
    quench_coeff)

# join coeffs to assay/substrate well calculations
wide_all <- left_join(wide, coeffs, by = "plate")
```

8.2 Calculate net fluorescence

Net fluorescence = ((assay-homogenate control) / quench coefficient) - (substrate control-plate blank)

```
# net fluorescence calculation for each enzyme
net_fluor_calcs <- wide_all %>%
mutate(BG_net_fluor = (((`mean_fluor_BG-assay` - hombl)/quench_coeff) -
    (`mean_fluor_BG-sub-blank` - plate_blank)), Cello_net_fluor = (((`mean_fluor_Cello-assay` -
    hombl)/quench_coeff) - (`mean_fluor_Cello-sub-blank` - plate_blank)),
    NAG_net_fluor = (((`mean_fluor_NAG-assay` - hombl)/quench_coeff) -
```

```
        (`mean_fluor_NAG-sub-blank` - plate_blank)), P_net_fluor = (((`mean_fluor_P-assay` -
        hombl)/quench_coeff) - (`mean_fluor_P-sub-blank` - plate_blank)))
```

8.3 Calculate enzyme activity

Activity(nmol/gˆ-1 hˆ-1) = [Net fluor x buffer vol (mL)] / [emis coeff x homogenate vol(mL) x time(h) x ode soil mass mass(g)]

Activity units are nmol / gˆ-1 hrˆ-1

```
# buffer vol is the volume of buffer used to make the soil
# slurries. In the regular Gutknecht lab protocol, this is 50
# mL
buffer_vol_ml <- 50

# vol of homogenate added to the assay wells in mL
homogenate_vol_ml <- 0.2

# bring in plate metadata (incubation time, ov dry soil eq,
# etc)
calcs_with_meta <- left_join(net_fluor_calcs, plate_metadata,
    by = "plate")

# activity calculation for each enzyme
activity_calcs <- calcs_with_meta %>%
mutate(BG_activity = (BG_net_fluor * buffer_vol_ml)/(emis_coeff *
    homogenate_vol_ml * inc_time_hr * soil_ov_dry_eq_g), Cello_activity = (Cello_net_fluor *
    buffer_vol_ml)/(emis_coeff * homogenate_vol_ml * inc_time_hr *
    soil_ov_dry_eq_g), NAG_activity = (NAG_net_fluor * buffer_vol_ml)/(emis_coeff *
    homogenate_vol_ml * inc_time_hr * soil_ov_dry_eq_g), P_activity = (P_net_fluor *
    buffer_vol_ml)/(emis_coeff * homogenate_vol_ml * inc_time_hr *
    soil_ov_dry_eq_g))
```

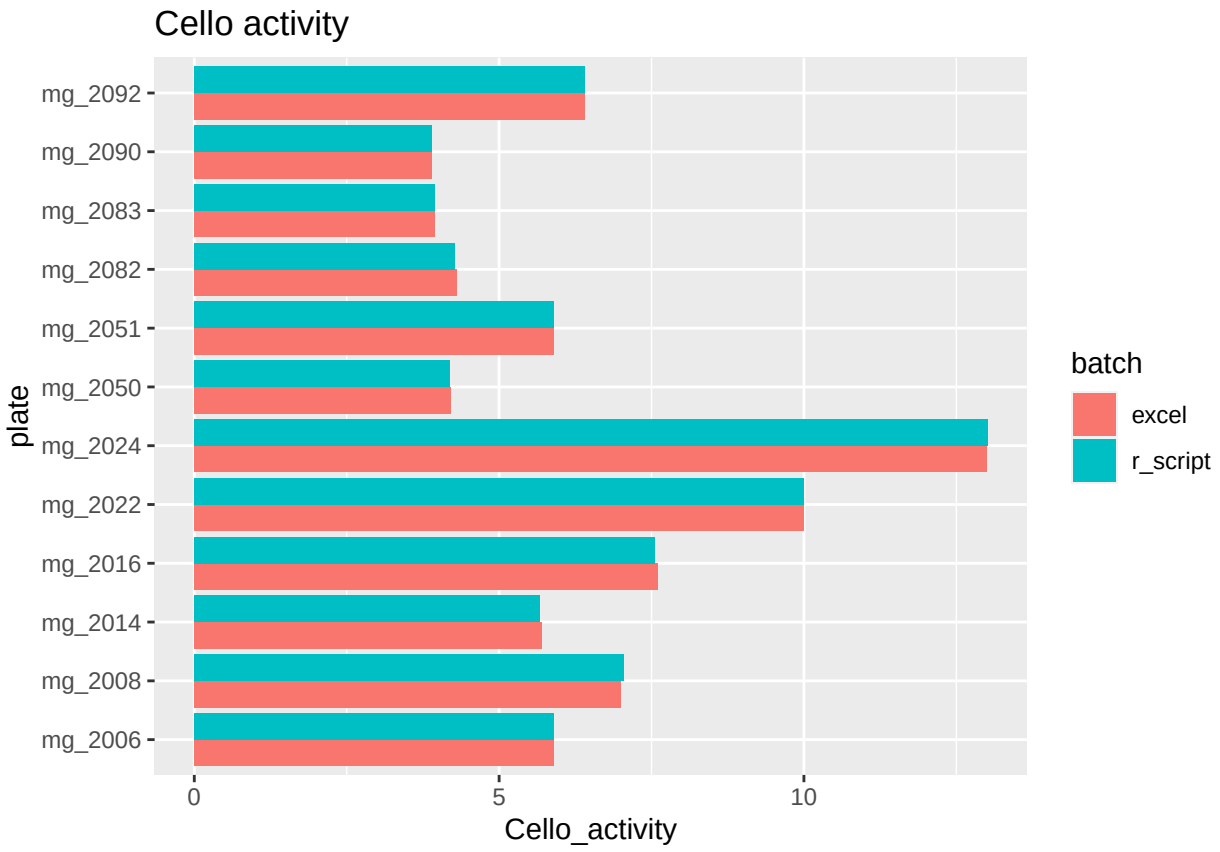
9 Compare R vs. Excel results

I used the "HB_Hydrolygic enzyme calculation template_12 samples" in the Gutknecht shared drive to do the "Excel" calculations listed here. I had to a couple changes to the calculations so they aligned with my R script:

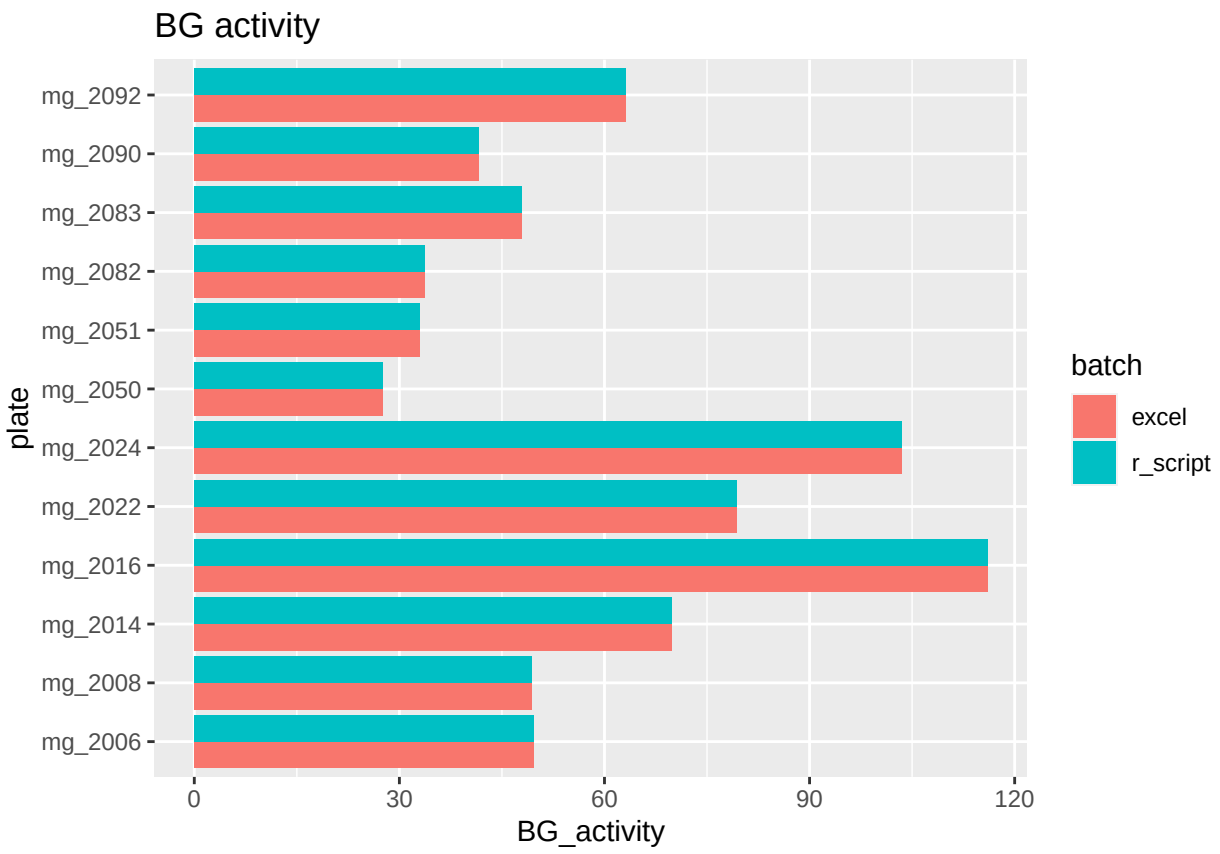
Emission coefficient: In the Excel sheet, emission coefficient is calculated as the slope of the MUB standard curve. German et al. 2011 specify that emission coefficient is this slope **divided by the assay volume**. I changed the excel spreadsheet to account for this, and also removed the assay volume term from the **final activity calculation** so we weren't accounting for it twice. The math would have worked out either way, but I wanted to be able to compare coefficients.

```
# results from R calculations
script_test <- activity_calcs %>% select(plate, BG_activity,
    P_activity, NAG_activity, Cello_activity)

# Load in test results from excel spreadsheet calcs
test1 <- read.csv("./excel_test1_mg.csv", stringsAsFactors = FALSE)
```

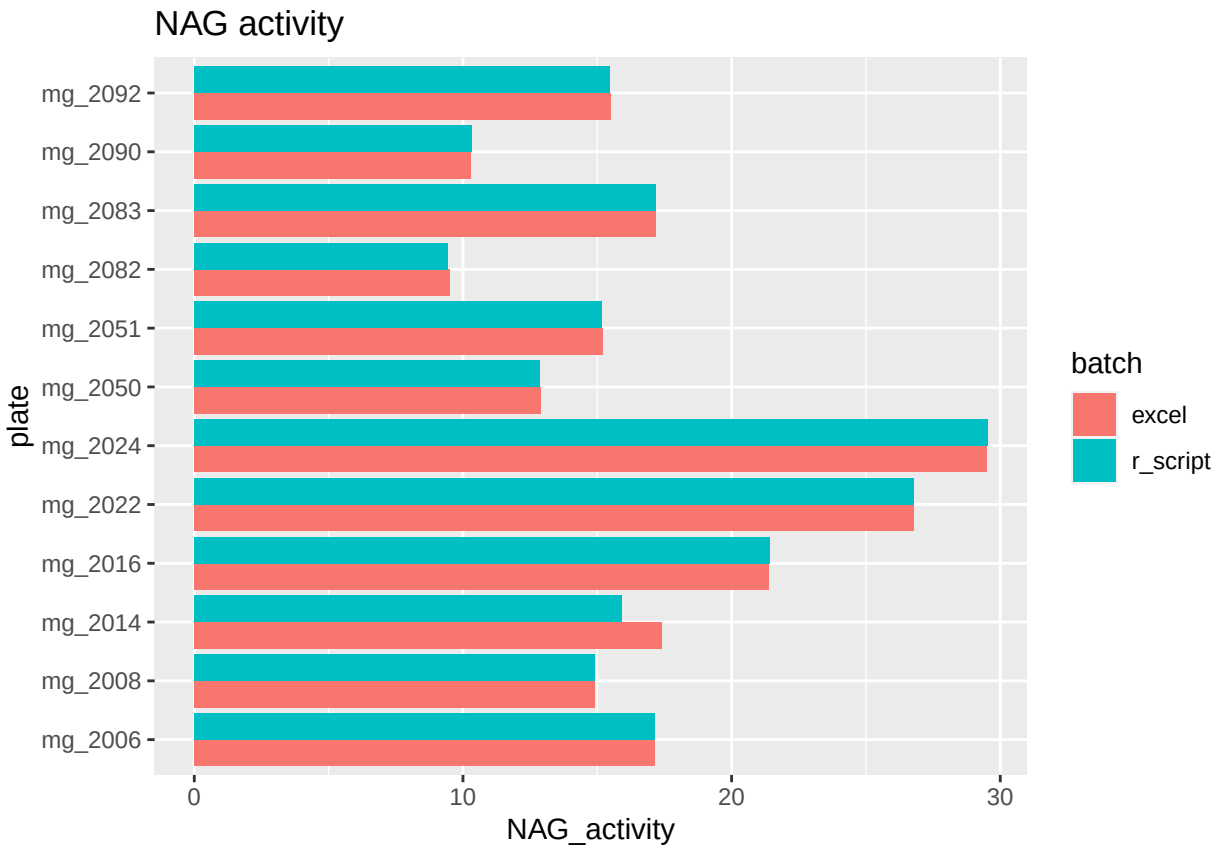
```r
colnames(test1) <- c("plate", "BG_activity", "P_activity", "NAG_activity",
    "Cello_activity")

# create columns to specify data provenance
test1$batch = "excel"
script_test$batch = "r_script"

# convert to dataframe
test1 <- as.data.frame(test1)
script_test <- as.data.frame(script_test)

# merge calcs from excel & R script into one dataframe
merge_test <- rbind(test1, script_test)

# bar plots to compare calcs
cello_plot <- ggplot() + geom_col(data = merge_test, aes(x = plate,
    y = Cello_activity, group = batch, fill = batch), position = "dodge") +
    coord_flip() + labs(title = "Cello activity")

bg_plot <- ggplot() + geom_col(data = merge_test, aes(x = plate,
    y = BG_activity, group = batch, fill = batch), position = "dodge") +
    coord_flip() + labs(title = "BG activity")

nag_plot <- ggplot() + geom_col(data = merge_test, aes(x = plate,
    y = NAG_activity, group = batch, fill = batch), position = "dodge") +
    coord_flip() + labs(title = "NAG activity")

p_plot <- ggplot() + geom_col(data = merge_test, aes(x = plate,
    y = P_activity, group = batch, fill = batch), position = "dodge") +
    coord_flip() + labs(title = "P activity")

cello_plot
```


Cello activity
mg_2092
mg_2090
mg_2083
mg_2082
mg_2051
mg_2050
mg_2024
mg_2022
mg_2016
mg_2014
mg_2008
mg_2006
plate
0
5
10
Cello_activity
batch
excel
r_script

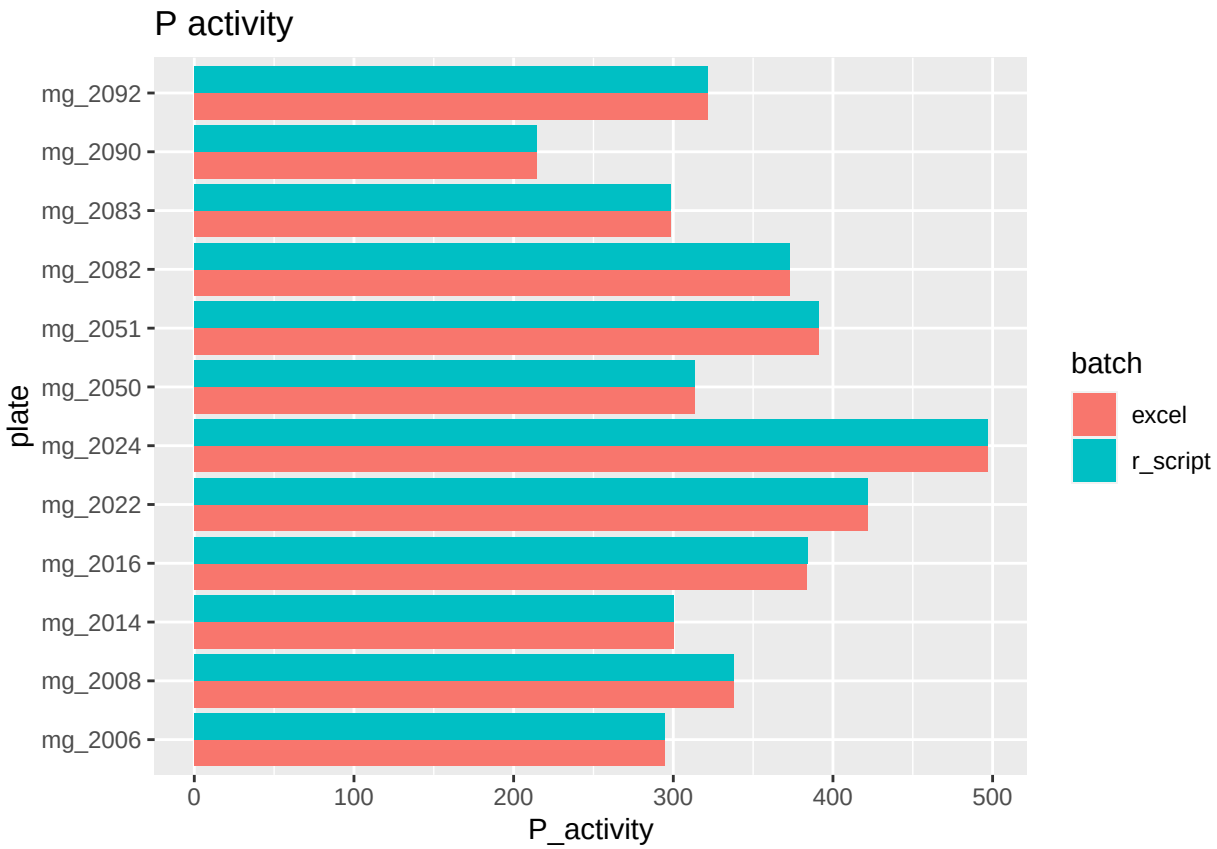
```
bg_plot
```


BG activity
mg_2092
mg_2090
mg_2083
mg_2082
mg_2051
mg_2050
mg_2024
mg_2022
mg_2016
mg_2014
mg_2008
mg_2006
plate
0
30
60
90
120
BG_activity
batch
excel
r_script

```
nag_plot
```


NAG activity
mg_2092
mg_2090
mg_2083
mg_2082
mg_2051
mg_2050
mg_2024
mg_2022
mg_2016
mg_2014
mg_2008
mg_2006
plate
0
10
20
30
NAG_activity
batch
excel
r_script

```
p_plot
```


P activity
mg_2092
mg_2090
mg_2083
mg_2082
mg_2051
mg_2050
mg_2024
mg_2022
mg_2016
mg_2014
mg_2008
mg_2006
plate
0
100
200
300
400
500
P_activity
batch
excel
r_script