Flume: Full Analysis (March-November)

(adapted from Jarad Niemi - Soilpad Analysis)

Jessica Nelson

2022-05-02

Contents


```
knitr::opts_chunk$set(echo = TRUE,
                      cache = TRUE,
                      fig.width = 12,
                      fig.height = 12)

library("lme4")
```

```
## Loading required package: Matrix
```

```
library("lmerTest")
```

```
##
## Attaching package: 'lmerTest'

## The following object is masked from 'package:lme4':
##
##     lmer

## The following object is masked from 'package:stats':
##
##     step
```

```r
library("tidyverse"); theme_set(theme_bw())
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.8
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x tidyr::expand() masks Matrix::expand()
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x tidyr::pack()   masks Matrix::pack()
## x tidyr::unpack() masks Matrix::unpack()
```

```r
library("lattice")
library("emmeans")
library("ggResidpanel")
library("data.table")
```

```
##
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':
##
##     between, first, last

## The following object is masked from 'package:purrr':
##
##     transpose
```

```r
library("stringr")
library("ggplot2")
library("ggpmisc")
```

```
## Loading required package: ggpp

##
## Attaching package: 'ggpp'

## The following object is masked from 'package:ggplot2':
##
##     annotate
```

```r
options(width = 120)

dir.create("fig", showWarnings = FALSE)
```

```
sessionInfo()
```

```
## R version 4.1.3 (2022-03-10)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 17763)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United States.1252  LC_CTYPE=English_United States.1252    LC_MONETARY=Englis
## [4] LC_NUMERIC=C                           LC_TIME=English_United States.1252
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] ggpmisc_0.4.6      ggpp_0.4.4         data.table_1.14.2  ggResidpanel_0.3.0 emmeans_1.7.2
##  [7] forcats_0.5.1      stringr_1.4.0      dplyr_1.0.8        purrr_0.3.4        readr_2.1.2
## [13] tibble_3.1.6       ggplot2_3.3.5      tidyverse_1.3.1    lmerTest_3.1-3     lme4_1.1-28
##
## loaded via a namespace (and not attached):
##  [1] httr_1.4.2          viridisLite_0.4.0   jsonlite_1.8.0      splines_4.1.3       modelr_0.1.8
##  [6] assertthat_0.2.1    cellranger_1.1.0    robustbase_0.93-9   yaml_2.3.5          numDeriv_2016.8
## [11] pillar_1.7.0        backports_1.4.1     quantreg_5.88       glue_1.6.2          digest_0.6.29
## [16] rvest_1.0.2         minqa_1.2.4         colorspace_2.0-3    cowplot_1.1.1       htmltools_0.5.2
## [21] pkgconfig_2.0.3     broom_0.7.12        SparseM_1.81        haven_2.4.3         xtable_1.8-4
## [26] mvtnorm_1.1-3       scales_1.1.1        MatrixModels_0.5-0  tzdb_0.2.0          generics_0.1.2
## [31] ellipsis_0.3.2      withr_2.5.0         lazyeval_0.2.2      cli_3.2.0           magrittr_2.0.2
## [36] crayon_1.5.0        readxl_1.3.1        estimability_1.3    evaluate_0.15       fs_1.5.2
## [41] fansi_1.0.2         nlme_3.1-155        MASS_7.3-55         xml2_1.3.3          tools_4.1.3
## [46] hms_1.1.1           lifecycle_1.0.1     plotly_4.10.0       munsell_0.5.0       reprex_2.0.1
## [51] qqplotr_0.0.5       compiler_4.1.3      rlang_1.0.2         grid_4.1.3          nloptr_2.0.0
## [56] rstudioapi_0.13     htmlwidgets_1.5.4   rmarkdown_2.13      boot_1.3-28         gtable_0.3.0
## [61] DBI_1.1.2           R6_2.5.1            lubridate_1.8.0     knitr_1.38          fastmap_1.1.0
## [66] utf8_1.2.2          stringi_1.7.6       Rcpp_1.0.8          vctrs_0.3.8         DEoptimR_1.0-10
## [71] dbplyr_2.1.1        tidyselect_1.1.2    xfun_0.30
```

Read in data

```
library("tidyverse")

options(scipen = 999)

flume <- read_csv("../data/tidy/flume_event_data612_UPDATE.csv") %>%
  mutate(Year = factor(Year)) %>%
  subset(SiteID != 'MAR') %>%
  subset(subset=!(SiteID=="MCN" & Year == 2016)) %>%
  subset(subset=!(SiteID=="MCN" & Year == 2017)) %>%
  subset(subset=!(SiteID=="MCN" & Year == 2018)) %>%
  subset(subset=!(SiteID=="MCN" & Year == 2019)) %>%
  subset(subset=!(SiteID=="MCN" & Year == 2020)) %>%
```

```
  subset(subset=!(SiteID=="RHO" & Year == 2016)) %>%
  subset(subset=!(SiteID=="RHO" & Year == 2017)) %>%
  subset(subset=!(SiteID == "WOR" & Year == 2018 & precipitation == "NA"))
```

```
## Rows: 439 Columns: 23
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (8): SiteID, subtreatment, Treatment, sampleID, rf_event, random, crop, f_loc
## dbl (15): precipitation, rain_time, slope75, Lfactor, Sfactor, LSfactor, sample_event, ro_event, Yea
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
flume_sum <- flume %>%
  group_by(Treatment, Year, SiteID, sample_event, tss_sum, crop, slope75, LSfactor, Lfactor,Sfactor) %>%
  summarize(tss_load = tss_sum) %>%
  distinct()
```

```
## `summarise()` has grouped output by 'Treatment', 'Year', 'SiteID', 'sample_event', 'tss_sum', 'crop'
## 'LSfactor', 'Lfactor', 'Sfactor'. You can override using the `.groups` argument.
```

```
ppt_sum <- flume %>%
  group_by(Treatment, Year, SiteID, sample_event, crop, slope75, LSfactor, Lfactor,Sfactor) %>%
  #group_by(Treatment, Year, SiteID, sample_event, crop) %>%
  summarize(ppt_sum = sum(precipitation)) %>%
  ungroup() %>%
  filter(!duplicated(cbind(Year, SiteID, sample_event)))
```

```
## `summarise()` has grouped output by 'Treatment', 'Year', 'SiteID', 'sample_event', 'crop', 'slope75'
## 'Lfactor'. You can override using the `.groups` argument.
```

```
sampl_anova <- flume_sum %>%
  filter(!is.na(tss_sum)) %>%
  select(Year, SiteID, Treatment, sample_event, tss_sum, crop) %>%
  group_by(SiteID, Year, Treatment, sample_event, crop) %>%
    summarize(tss_load = sum(tss_sum)) %>%
  ungroup() %>%
  select(Year, SiteID, Treatment, sample_event, tss_load, crop) %>%
  pivot_wider(names_from = Treatment, values_from = tss_load)
```

```
## Adding missing grouping variables: `slope75`, `LSfactor`, `Lfactor`, `Sfactor`
## `summarise()` has grouped output by 'SiteID', 'Year', 'Treatment', 'sample_event'. You can override u
## argument.
```

```
pivot_sample <- sampl_anova %>%
  inner_join(ppt_sum,by=c("SiteID", "Year", "sample_event", "crop")) %>%
  filter(!is.na(strips)) %>%
  mutate(ln_ppt = log(ppt_sum)) %>%
  subset(select = -c(Treatment))
```

```
long_load <- pivot_sample %>%
  gather(Treatment, tss_load, control:strips) %>%
  arrange(Treatment, tss_load) %>%
  filter(!is.na(diff)) %>%
  select(SiteID, Treatment, Year, sample_event, tss_load, ppt_sum, crop)
```

```
## Warning in is.na(diff): is.na() applied to non-(list or vector) of type 'closure'
```

```
rf_ro_pivot <- long_load %>%
  mutate(random = (ifelse(SiteID == 'ARM', 'NR',
  ifelse(SiteID == 'EIA', 'R',
  ifelse(SiteID == 'MCN', 'R',
  ifelse(SiteID == 'HOE', 'NR',
  ifelse(SiteID == 'RHO', 'R',
  ifelse(SiteID == 'WHI', 'NR',
  ifelse(SiteID == 'WOR', 'R', 0)))))))))

long_load <- long_load %>%
  mutate(random = (ifelse(SiteID == 'ARM', 'NR',
  ifelse(SiteID == 'EIA', 'R',
  ifelse(SiteID == 'MCN', 'R',
  ifelse(SiteID == 'HOE', 'NR',
  ifelse(SiteID == 'RHO', 'R',
  ifelse(SiteID == 'WHI', 'NR',
  ifelse(SiteID == 'WOR', 'R', 0)))))))))
```

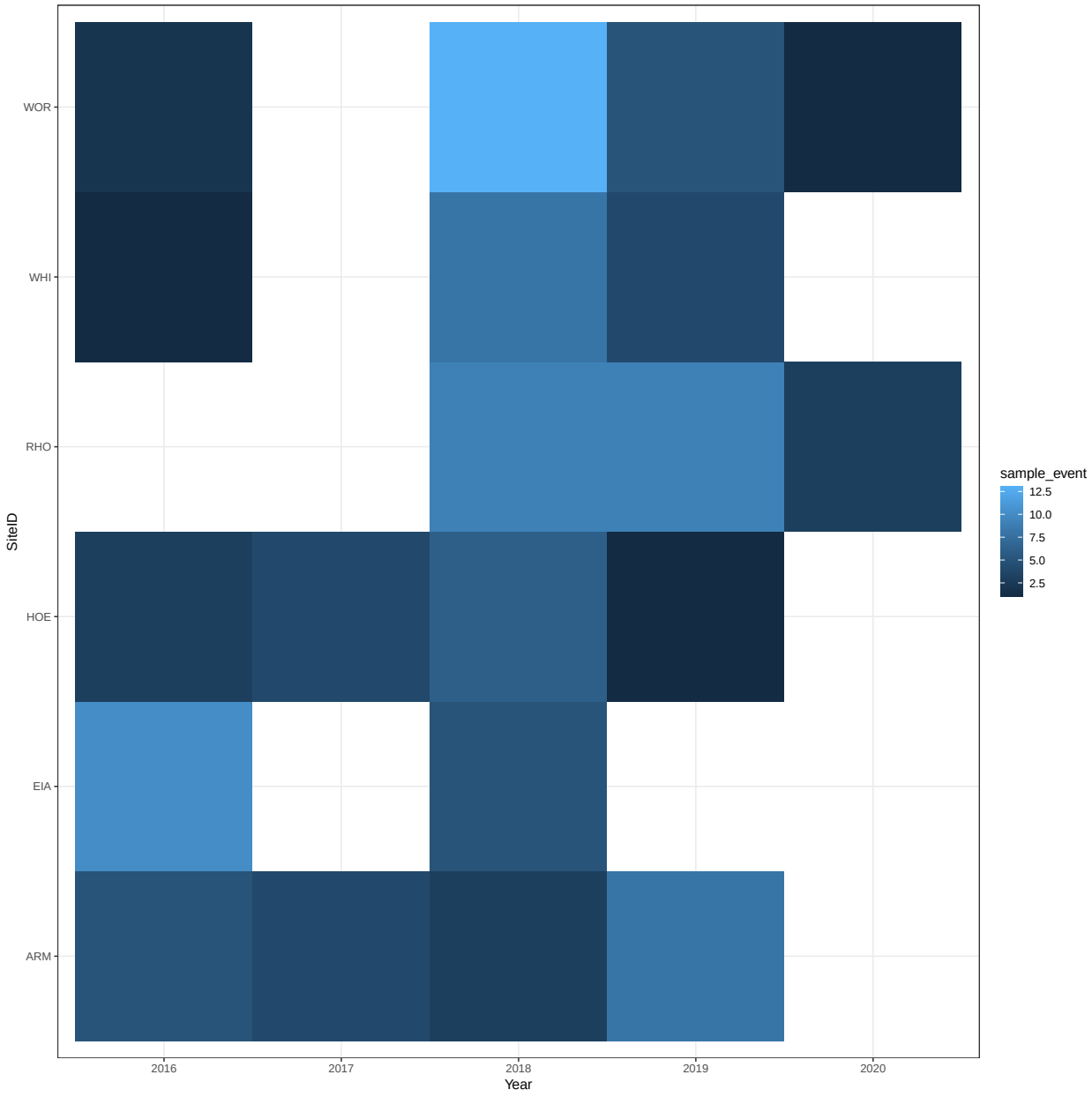
```
full_df <- rf_ro_pivot %>%
  inner_join(ppt_sum,by=c("SiteID", "Year", "sample_event","crop")) %>%
  drop_na(tss_load) %>%
  mutate(ppt_sum = ppt_sum.x,
         ln_ppt = log(ppt_sum.x),
         Treatment = factor(Treatment.x, levels=c('strips','control'))) %>%
  subset(select = -c(Treatment.y, Treatment.x, ppt_sum.x, ppt_sum.y)) %>%
  arrange(Year, SiteID, Treatment, sample_event)

save(full_df, file = "full_df.RData")

#write.csv(full_df,"D:/ISU/ResearchProject/flume_analysis/data/tidy/full_df.csv", row.names = FALSE)
```

```
load("full_df.RData")
```

Exploratory analysis

Site-year with sample event

```
site_year_rfevent <- full_df %>%
  select(SiteID, Year, sample_event) %>%
  unique()

ggplot(site_year_rfevent, aes(Year, SiteID, fill=sample_event)) +
  geom_tile()
```



Data visualization

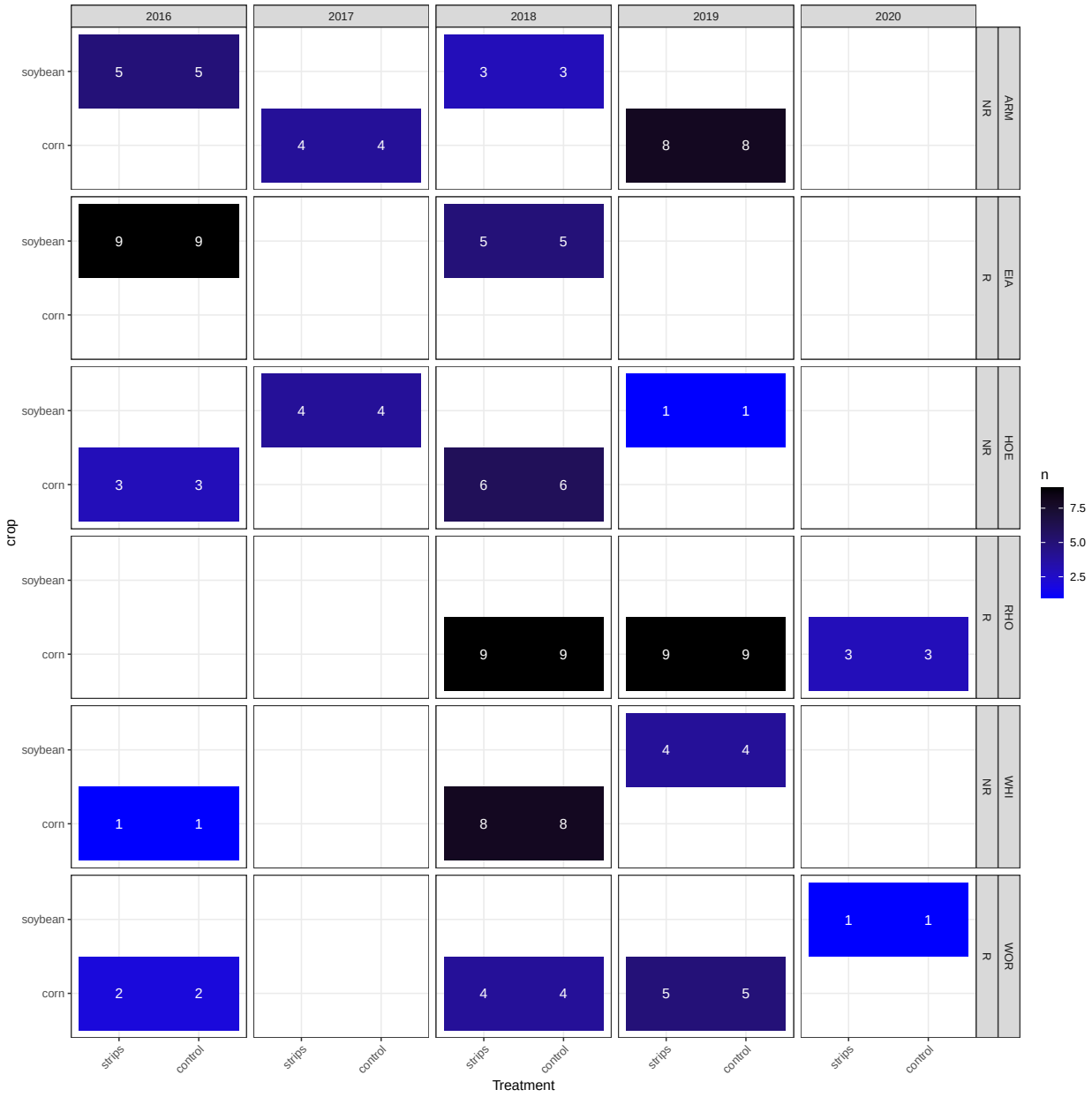
Number of samples

Calculate the number of observations for each treatment-position-year-site-time combination.

```
TSS_counts <- full_df %>%
  group_by(Year, SiteID, Treatment, crop, random) %>%
  distinct() %>%
  summarize(n = n(), .groups = "drop")
```

Plot the number of observations for each combination.

```
g = ggplot(TSS_counts, aes(x = Treatment, y = crop, fill = n)) +
  geom_tile() +
  geom_text(aes(label = n), color = "white") +
  facet_grid(SiteID + random ~ Year) +
  scale_fill_gradient(low = "blue", high = "black") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

g
```

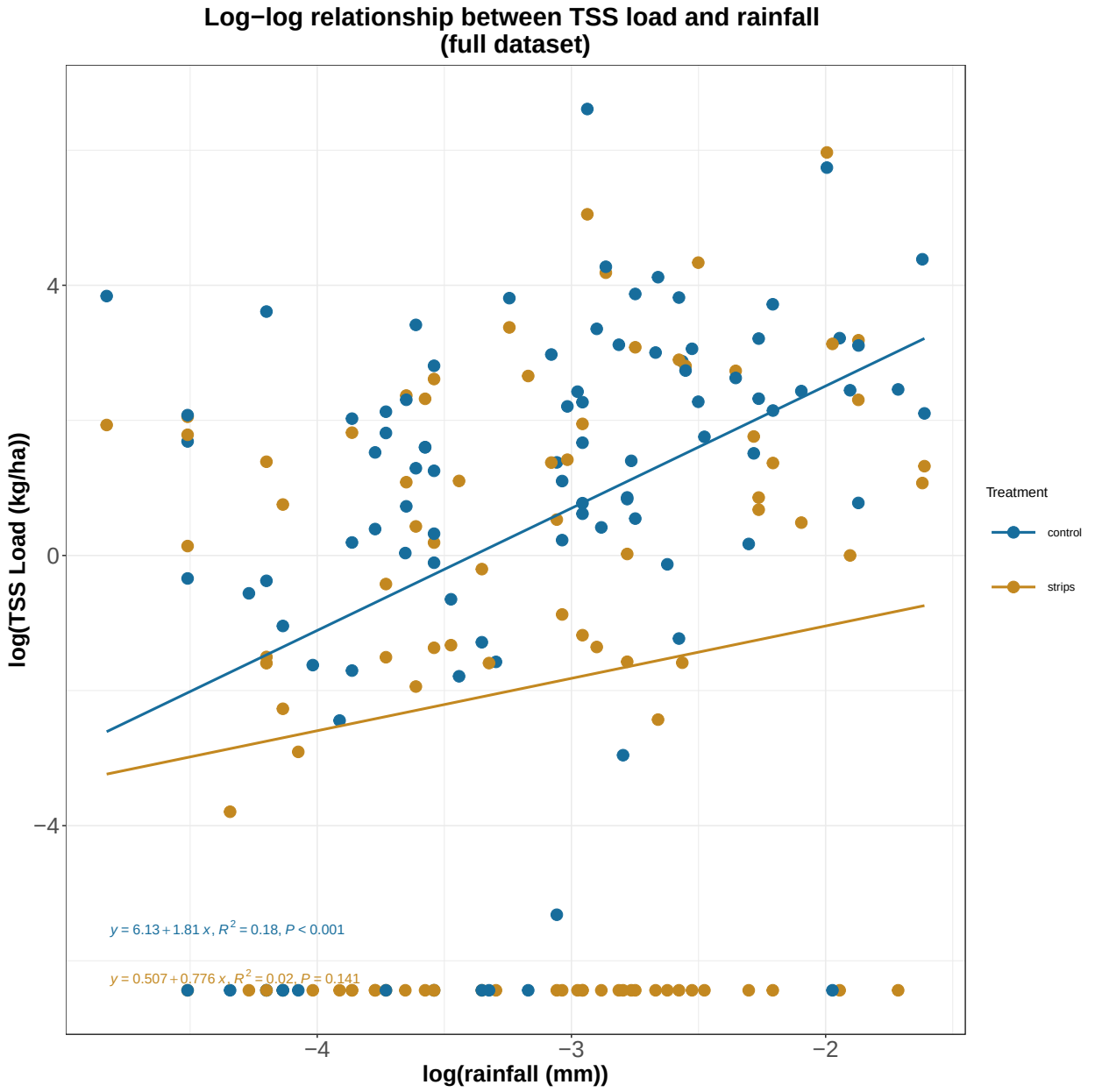



```
# {r, dependson="load"} #  # h <- ggplot(full_df, aes(x=log(ppt_sum), y=log(tss_load+0.0016488035), color=Treatment)) + #   labs(x="Day of year",y="Totall number of cells")+ #   geom_point(shape = 16, size=3) + #   geom_abline(aes(intercept=`(Intercept)`, slope=ln_ppt), as.data.frame(t(fixef(m_flu + #   #scale_y_log10() + #   scale_color_manual(values = c("control" = "#176D9C", # "strips" = "#C38820")) + #   xlab("log(rainfall total)") + #   ylab("log(TSS Load (kg/ha))")
```

```
+ #   #geom_smooth(method=lm, se=TRUE, fullrange=TRUE) + #   ggtitle("Log-log relationship
between TSS load and rainfall accumulation \n(full dataset)") + #   theme(plot.title =
element_text(size=14, face="bold",hjust = 0.5), #         legend.key.size = unit(3,"line"))
+ #   theme(plot.title = element_text(size=20, face="bold", hjust=0.5), #         axis.title.x
= element_text(size=18, face="bold"), #         axis.title.y = element_text(size=18,
face="bold"), #         axis.text.x = element_text(size=18), #         axis.text.y =
element_text(size=18)) + #   xlim(-5, -1) #    # h #  # #ggsave("fig/randReg_ppt_load1trend.png",
h, width = 12, height = 12) #  #
```

```
h <- ggplot(full_df, aes(x=log(ppt_sum), y=log(tss_load+0.0016), color=Treatment), inherit.aes = FALSE)
 scale_color_manual(values = c("control" = "#176D9C",
                               "strips" = "#C38820")) +
 geom_point(size=4) +
 xlab("log(rainfall (mm))") +
 ylab("log(TSS Load (kg/ha))") +
 geom_smooth(method=lm, se=FALSE, fullrange=TRUE) +
 ggtitle("Log-log relationship between TSS load and rainfall \n(full dataset)") +
 theme(plot.title = element_text(size=14, face="bold",hjust = 0.5),
       legend.key.size = unit(3,"line")) +
 theme(plot.title = element_text(size=20, face="bold", hjust=0.5),
       axis.title.x = element_text(size=18, face="bold"),
       axis.title.y = element_text(size=18, face="bold"),
       axis.text.x = element_text(size=18),
       axis.text.y = element_text(size=18)) +
 #xlim(-5, -1.5) +
 stat_poly_eq(formula = y ~ x,
              aes(label = paste(..eq.label.., ..rr.label.., ..p.value.label.., sep = "*`,`~")),
              parse = TRUE,
              label.x.npc = "left",
              label.y.npc = "bottom",
              vstep = 0.05) #+
 #scale_y_log10()

h
```

```
## `geom_smooth()` using formula 'y ~ x'
```



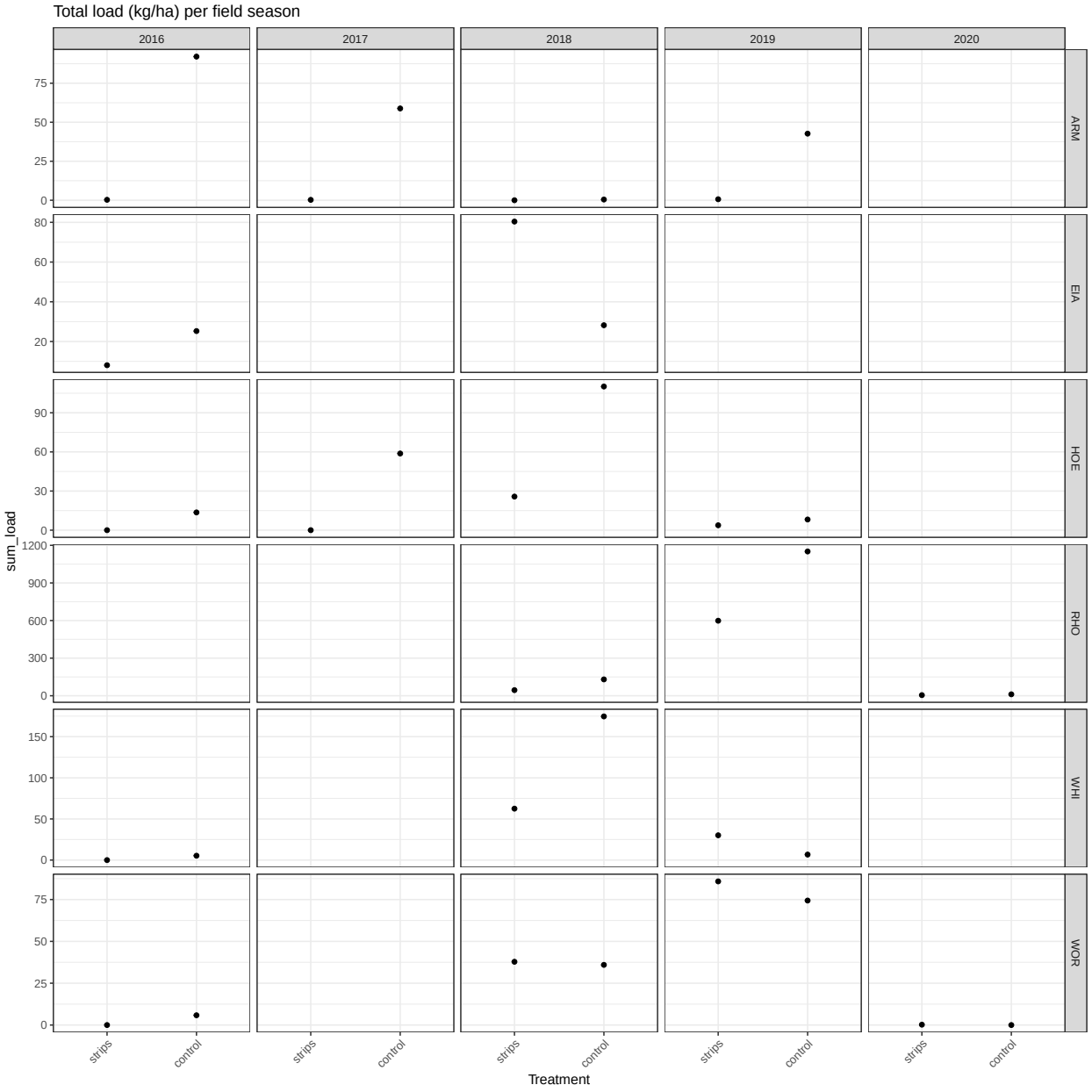

```
# ggsave("fig/randReg_sfactorbase10.png", h, width = 12, height = 12)
```

```
load_sum <- full_df %>%
  group_by(Year, SiteID, Treatment, crop) %>%
  summarize(sum_load             = sum(tss_load, na.rm = TRUE),
            log_load = mean(log(sum_load), na.rm = TRUE),
            n                    = n(),
            .groups = "drop")


ppt_sum <- full_df %>%
  group_by(Year, SiteID) %>%
  summarize(sum_ppt              = sum(ppt_sum, na.rm = TRUE)*1000,
```

```r
            #log_load = mean(log(sum_load), na.rm = TRUE),
            n                    = n(),
            .groups = "drop")

TSS_sum <- full_df %>%
  group_by(Year, SiteID, Treatment) %>%
  distinct() %>%
  summarize(tss_sum = sum(tss_load), .groups = "drop")
```

```r
g <- ggplot(load_sum,
           aes(x = Treatment,
               y = sum_load)) +
  geom_point() +
  geom_line() +
  facet_grid(SiteID ~ Year, scales = "free_y") +
  #scale_y_log10() +
  labs(title ="Total load (kg/ha) per field season") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
g
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust the group aesthetic?
```




```
#ggsave("fig/wp_per_day_plot.png", g)
```

Average isn't realistic

```
#{r, dependson="create_sediment"} #pivot_sample %>%  #  anova_test(ln_trt ~ ln_ppt*ln_ctl)
## purr https://stackoverflow.com/questions/50702152/compare-models-via-anova-with-purrr-or-dplyr
## anova() and may need an linear model built up. #
```

Main Analyses

There are three main analyses of interest:

- confirmatory, design-based analysis

- exploratory, covariate analysis
- relationship of sediment flow to sediment loss

Confirmatory, design-based analysis

Treatment effect

```
m_flume <- lmerTest::lmer(log(tss_load+0.0016488035) ~
                            Treatment*ln_ppt +
                           Treatment*crop +
                           Year*Treatment +

                           #(1 | SiteID) + #removed due to singular fit
                            (1 | SiteID:Treatment) +
                           (1|Year:sample_event) + #consider this and below with SiteID
                           (1|SiteID:Year:sample_event),

                         data = full_df)

summary(m_flume)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method ['lmerModLmerTest']
## Formula: log(tss_load + 0.0016488035) ~ Treatment * ln_ppt + Treatment *
##     crop + Year * Treatment + (1 | SiteID:Treatment) + (1 | Year:sample_event) +      (1 | SiteID:Yea
##    Data: full_df
##
## REML criterion at convergence: 914
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.4445 -0.5003  0.0822  0.5487  1.9205
##
## Random effects:
##  Groups                    Name        Variance Std.Dev.
##  SiteID:Year:sample_event (Intercept) 1.6534   1.2859
##  Year:sample_event        (Intercept) 0.3946   0.6282
##  SiteID:Treatment         (Intercept) 2.9090   1.7056
##  Residual                             6.3011   2.5102
## Number of obs: 188, groups:  SiteID:Year:sample_event, 94; Year:sample_event, 38; SiteID:Treatment,
##
## Fixed effects:
##                               Estimate Std. Error       df t value Pr(>|t|)
## (Intercept)                    -0.9303     1.6242 108.3155  -0.573 0.567992
## Treatmentcontrol                6.8732     2.0451  69.1950   3.361 0.001268 **
## ln_ppt                          0.9291     0.3941 157.4461   2.357 0.019643 *
## cropsoybean                     0.6396     0.8133 148.0779   0.786 0.432872
## Year2017                        0.4205     1.3354 102.8844   0.315 0.753466
## Year2018                        2.0676     0.9502  68.2671   2.176 0.033013 *
## Year2019                        3.7002     1.0233  75.1924   3.616 0.000539 ***
## Year2020                        0.9475     1.7455 133.9956   0.543 0.588135
## Treatmentcontrol:ln_ppt         0.9093     0.4863  82.3246   1.870 0.065067 .
## Treatmentcontrol:cropsoybean   -2.0153     1.0270  90.4437  -1.962 0.052798 .
```

```
## Treatmentcontrol:Year2017      1.7158     1.6055  85.4024   1.069 0.288222
## Treatmentcontrol:Year2018     -1.2396     1.1188  84.0391  -1.108 0.271010
## Treatmentcontrol:Year2019     -2.7269     1.2125  85.8722  -2.249 0.027066 *
## Treatmentcontrol:Year2020     -2.8790     2.1243  85.9608  -1.355 0.178890
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

##
## Correlation matrix not shown by default, as p = 14 > 12.
## Use print(x, correlation=TRUE)  or
##     vcov(x)        if you need it
```

```
m_flume_step <- step(m_flume, reduce.random = FALSE, alpha.fixed = 0.1)
m_flume_model <- get_model(m_flume_step)
summary(m_flume_model)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method ['lmerModLmerTest']
## Formula: log(tss_load + 0.0016488035) ~ Treatment * ln_ppt + Treatment *
##     crop + Year * Treatment + (1 | SiteID:Treatment) + (1 | Year:sample_event) +      (1 | SiteID:Yea
##    Data: full_df
##
## REML criterion at convergence: 914
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.4445 -0.5003  0.0822  0.5487  1.9205
##
## Random effects:
##  Groups                    Name        Variance Std.Dev.
##  SiteID:Year:sample_event (Intercept) 1.6534   1.2859
##  Year:sample_event        (Intercept) 0.3946   0.6282
##  SiteID:Treatment         (Intercept) 2.9090   1.7056
##  Residual                             6.3011   2.5102
## Number of obs: 188, groups:  SiteID:Year:sample_event, 94; Year:sample_event, 38; SiteID:Treatment,
##
## Fixed effects:
##                              Estimate Std. Error       df t value Pr(>|t|)
## (Intercept)                   -0.9303     1.6242 108.3155  -0.573 0.567992
## Treatmentcontrol               6.8732     2.0451  69.1950   3.361 0.001268 **
## ln_ppt                         0.9291     0.3941 157.4461   2.357 0.019643 *
## cropsoybean                    0.6396     0.8133 148.0779   0.786 0.432872
## Year2017                       0.4205     1.3354 102.8844   0.315 0.753466
## Year2018                       2.0676     0.9502  68.2671   2.176 0.033013 *
## Year2019                       3.7002     1.0233  75.1924   3.616 0.000539 ***
## Year2020                       0.9475     1.7455 133.9956   0.543 0.588135
## Treatmentcontrol:ln_ppt        0.9093     0.4863  82.3246   1.870 0.065067 .
## Treatmentcontrol:cropsoybean  -2.0153     1.0270  90.4437  -1.962 0.052798 .
## Treatmentcontrol:Year2017      1.7158     1.6055  85.4024   1.069 0.288222
## Treatmentcontrol:Year2018     -1.2396     1.1188  84.0391  -1.108 0.271010
## Treatmentcontrol:Year2019     -2.7269     1.2125  85.8722  -2.249 0.027066 *
## Treatmentcontrol:Year2020     -2.8790     2.1243  85.9608  -1.355 0.178890
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Correlation matrix not shown by default, as p = 14 > 12.
## Use print(x, correlation=TRUE)  or
##     vcov(x)        if you need it
```

```
##https://campus.datacamp.com/courses/hierarchical-and-mixed-effects-models-in-r/linear-mixed-effect-mo
```

#“‘{r design_step_model, dependson = “design_model”} #emmip(m_flume, ln_ppt ~ Treatment | Year)

##https://campus.datacamp.com/courses/hierarchical-and-mixed-effects-models-in-r/linear-mixed-effect-models?ex=7 #“‘

```
trt_yr = emmeans(m_flume, pairwise ~ Treatment|Year,
                    type = "response",
                    lmer.df = "asymptotic")
confint(trt_yr)$contrasts
```

```
## Year = 2016:
##  contrast           ratio     SE  df asymp.LCL asymp.UCL
##  strips / control 0.05016 0.0660 Inf  0.003805     0.661
##
## Year = 2017:
##  contrast           ratio     SE  df asymp.LCL asymp.UCL
##  strips / control 0.00902 0.0152 Inf  0.000333     0.244
##
## Year = 2018:
##  contrast           ratio     SE  df asymp.LCL asymp.UCL
##  strips / control 0.17328 0.2057 Inf  0.016917     1.775
##
## Year = 2019:
##  contrast           ratio     SE  df asymp.LCL asymp.UCL
##  strips / control 0.76681 0.9542 Inf  0.066905     8.789
##
## Year = 2020:
##  contrast           ratio     SE  df asymp.LCL asymp.UCL
##  strips / control 0.89275 1.8887 Inf  0.014122    56.438
##
## Results are averaged over the levels of: crop
## Degrees-of-freedom method: asymptotic
## Confidence level used: 0.95
## Intervals are back-transformed from the log scale
```

```
trt = emmeans(m_flume, pairwise ~ Treatment,
                    type = "response",
                    lmer.df = "asymptotic")
```

```
## NOTE: Results may be misleading due to involvement in interactions
```

```
confint(trt)
```

```
## $emmeans
##  Treatment response     SE  df asymp.LCL asymp.UCL
```

```
##  strips       0.119 0.0986 Inf    0.0224     0.598
##  control      0.857 0.7042 Inf    0.1704     4.283
##
## Results are averaged over the levels of: crop, Year
## Degrees-of-freedom method: asymptotic
## Confidence level used: 0.95
## Intervals are back-transformed from the log(mu + 0.002) scale
##
## $contrasts
##  contrast         ratio    SE  df asymp.LCL asymp.UCL
##  strips / control  0.14 0.156 Inf    0.0158      1.24
##
## Results are averaged over the levels of: crop, Year
## Degrees-of-freedom method: asymptotic
## Confidence level used: 0.95
## Intervals are back-transformed from the log scale
```

```
year = emmeans(m_flume, ~ Year,
                    type = "response",
                    lmer.df = "asymptotic")
```

```
## NOTE: Results may be misleading due to involvement in interactions
```

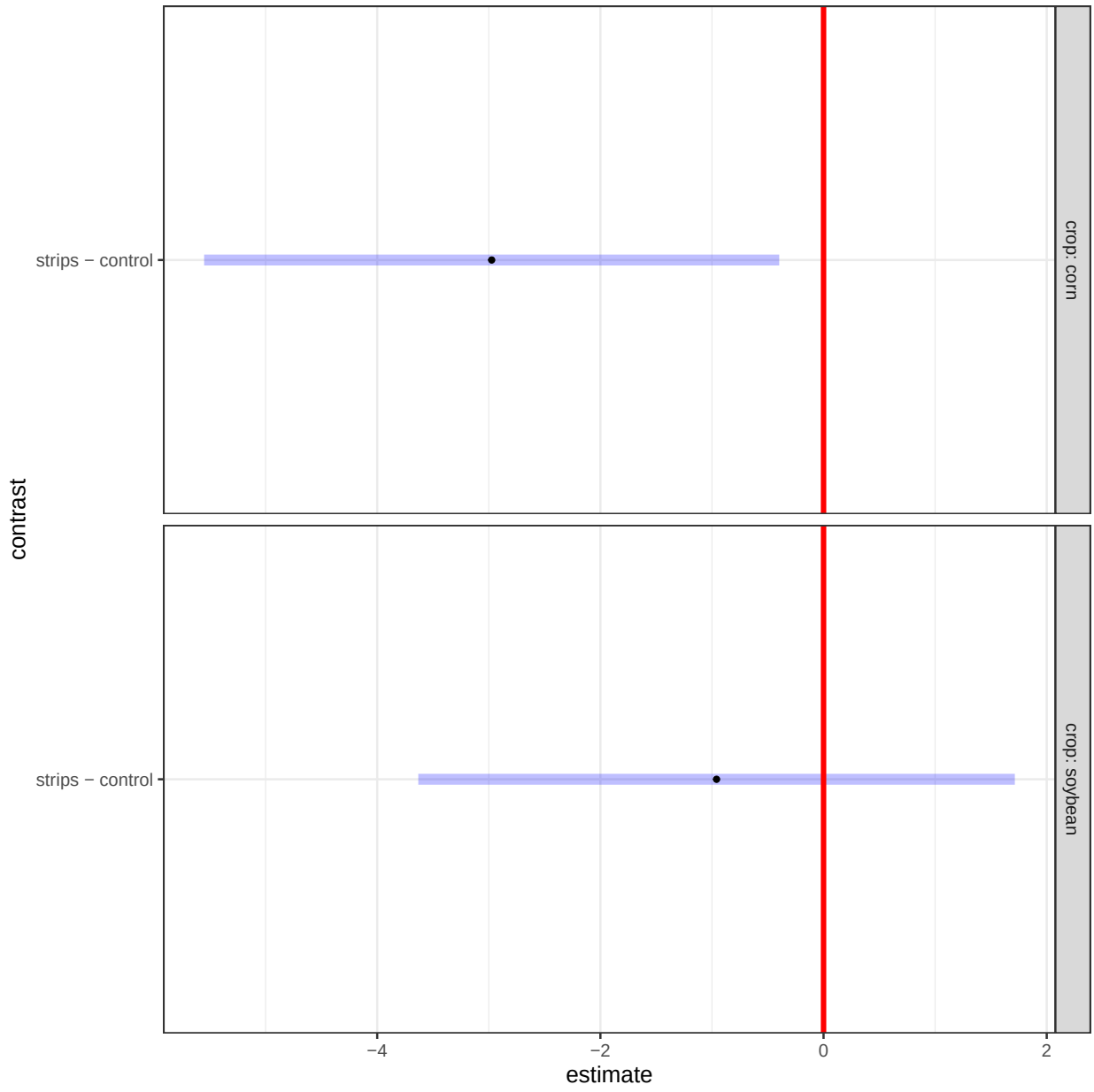
```
confint(year)
```

```
##  Year response    SE  df asymp.LCL asymp.UCL
##  2016   0.1271 0.099 Inf   0.02686      0.58
##  2017   0.4607 0.473 Inf   0.06065      3.43
##  2018   0.5460 0.369 Inf   0.14437      2.05
##  2019   1.3306 0.958 Inf   0.32374      5.45
##  2020   0.0771 0.103 Inf   0.00437      1.03
##
## Results are averaged over the levels of: Treatment, crop
## Degrees-of-freedom method: asymptotic
## Confidence level used: 0.95
## Intervals are back-transformed from the log(mu + 0.002) scale
```

```
trt_ppt = emmeans(m_flume, pairwise ~ Treatment|ln_ppt,
                    #at=list(ln_ppt=c(-4,-3.1,-3,-2)),
                    type = "response",
                    lmer.df = "asymptotic", adjust='scheffe')

confint(trt_ppt)$contrasts ## exp. the values
```

```
## ln_ppt = -3.15964530584514:
##  contrast         ratio    SE  df asymp.LCL asymp.UCL
##  strips / control  0.14 0.156 Inf    0.0158      1.24
##
## Results are averaged over the levels of: crop, Year
## Degrees-of-freedom method: asymptotic
## Confidence level used: 0.95
## Intervals are back-transformed from the log scale
```

```
plot(pairs(emmeans(m_flume, 'Treatment', 'crop'), adjust="scheffe")) +
  theme_bw(base_size=18) +
  geom_vline(xintercept=0, size=2, color="red")
```



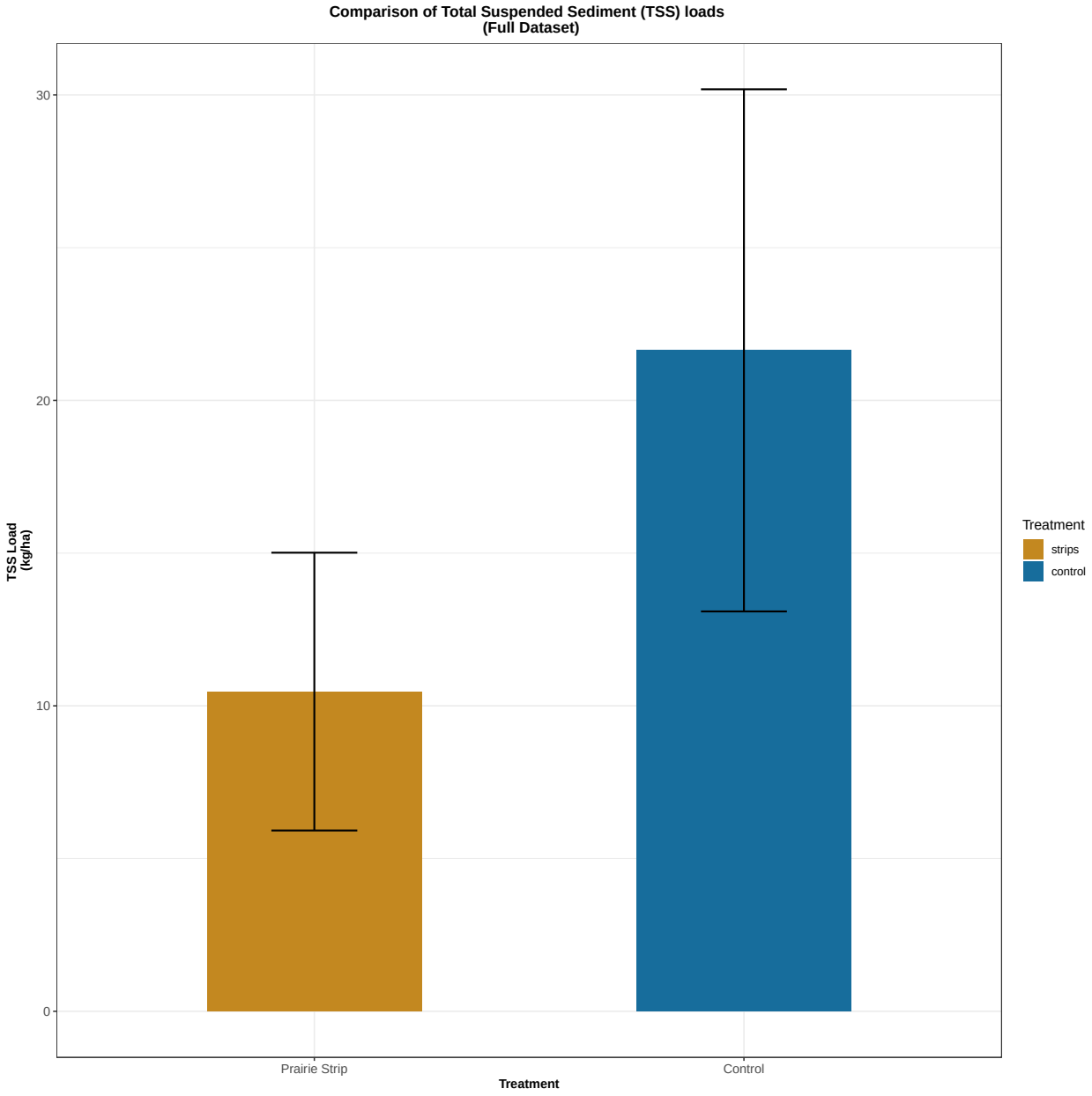

```
#crop      = emmeans(m_flume, pairwise ~ Treatment|crop,
#                    type = "response",
#                    lmer.df = "asymptotic")

#confint(crop)$contrasts
```

```
# trt <- as.data.frame(trt)
#
# k <- trt %>%
```

```
#   filter(contrast != "strips - control")
#
# trt_plot <- k %>%
#   ggplot(aes(x=Treatment, y=response, fill=Treatment))+
#   geom_bar(width = 0.5, position = position_dodge(), stat="summary") +
#   geom_errorbar(aes(ymin = (response-SE), ymax = (response+SE)),
#             width = 0.2,
#             linetype = "solid",
#             position = position_dodge(width = 0.5),
#             color="black", size=0.7) +
#   scale_fill_manual(values = c("control" = "#176D9C",
#                                "strips" = "#C38820")) +
#   ggtitle("Comparison of Total Suspended Sediment (TSS) loads \n (Full Dataset)") +
#   xlab("Treatment") +
#   ylab("TSS Load \n(kg/ha)") +
#   theme(plot.title = element_text(size=20, face="bold", hjust=0.5),
#         axis.title.x = element_text(size=18, face="bold"),
#         axis.title.y = element_text(size=18, face="bold"),
#         axis.text.x = element_text(size=18),
#         axis.text.y = element_text(size=18)) +
#   scale_x_discrete(labels= c("Control", "Prairie Strip"))
#
# trt_plot

d <- full_df

group <- full_df %>%
  group_by(Treatment) %>%
  summarize(n = n(),
            mean = mean(tss_load),
            sd = sd(tss_load),
            .groups = "drop") %>%
  mutate(se = sd / sqrt(n),
         lb = mean + qt(0.025, df = n-1)*se,
         ub = mean - qt(0.025, df = n-1)*se)

trt_plot <- group %>%
  ggplot(aes(x=Treatment, y=mean, fill=Treatment))+
  geom_bar(width = 0.5, position = position_dodge(), stat="summary") +
  geom_errorbar(aes(ymin = (mean-se), ymax = (mean+se)),
                width = 0.2,
                linetype = "solid",
                position = position_dodge(width = 0.5),
                color="black", size=0.7) +
  scale_fill_manual(values = c("strips" = "#C38820", "control" = "#176D9C")) +
  ggtitle("Comparison of Total Suspended Sediment (TSS) loads \n (Full Dataset)") +
  xlab("Treatment") +
  ylab("TSS Load \n(kg/ha)") +
  theme(plot.title = element_text(size=12, face="bold", hjust=0.5),
        axis.title.x = element_text(size=10, face="bold"),
        axis.title.y = element_text(size=10, face="bold"),
        axis.text.x = element_text(size=10),
        axis.text.y = element_text(size=10)) +
```

```
  scale_x_discrete(labels= c("Prairie Strip", "Control"))

trt_plot
```

```
## No summary function supplied, defaulting to `mean_se()`
```




```
#ggsave("E:/ISU/Project/SoilMove/data/statistics/flume_analysis/code/fig/flume_plot.png", trt_plot)
```

Check assumptions

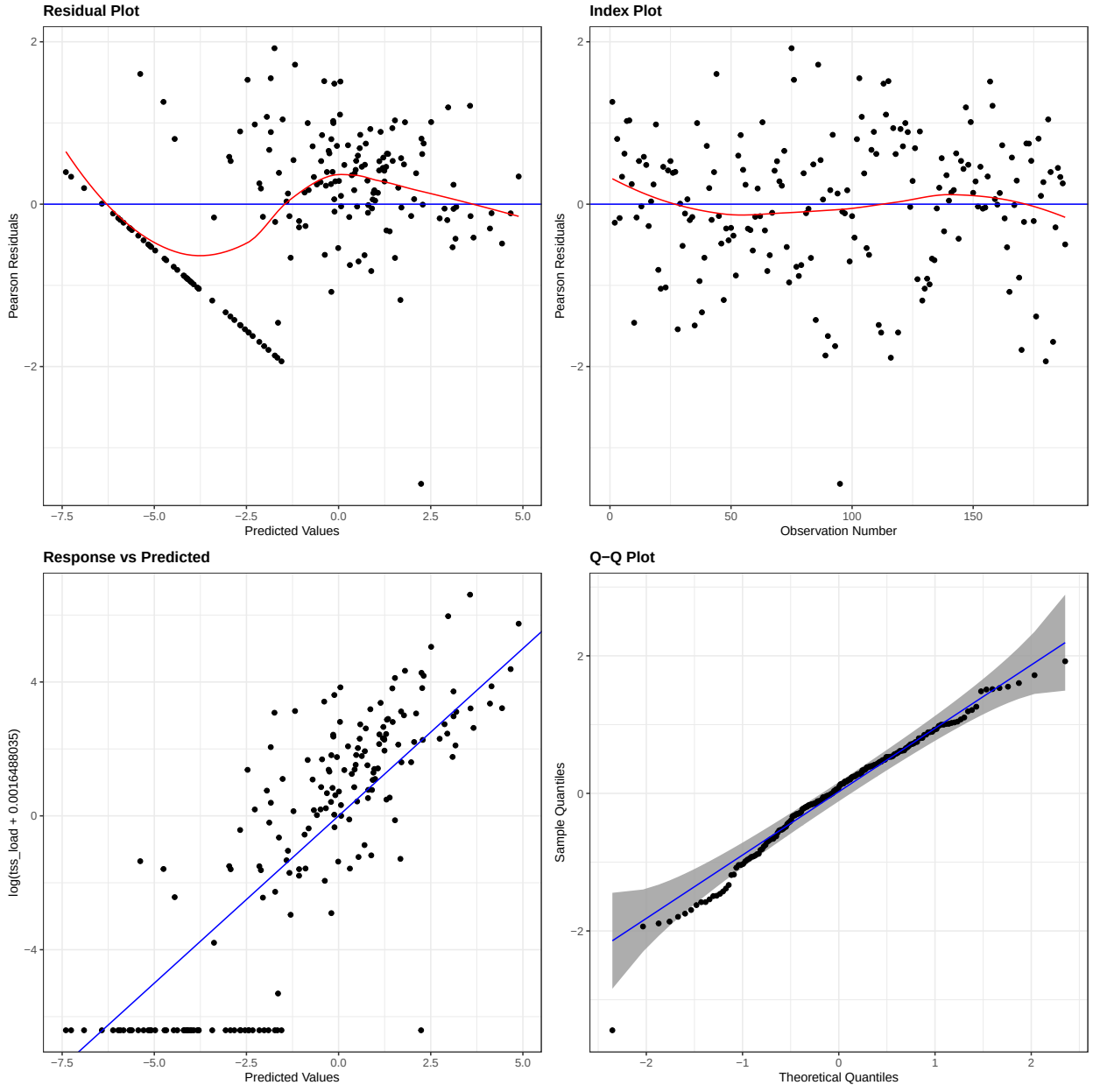
There are two possible models:

- m_flume: full model design, design-based analysis
- m_flume_model: model design selected based on backward step selection

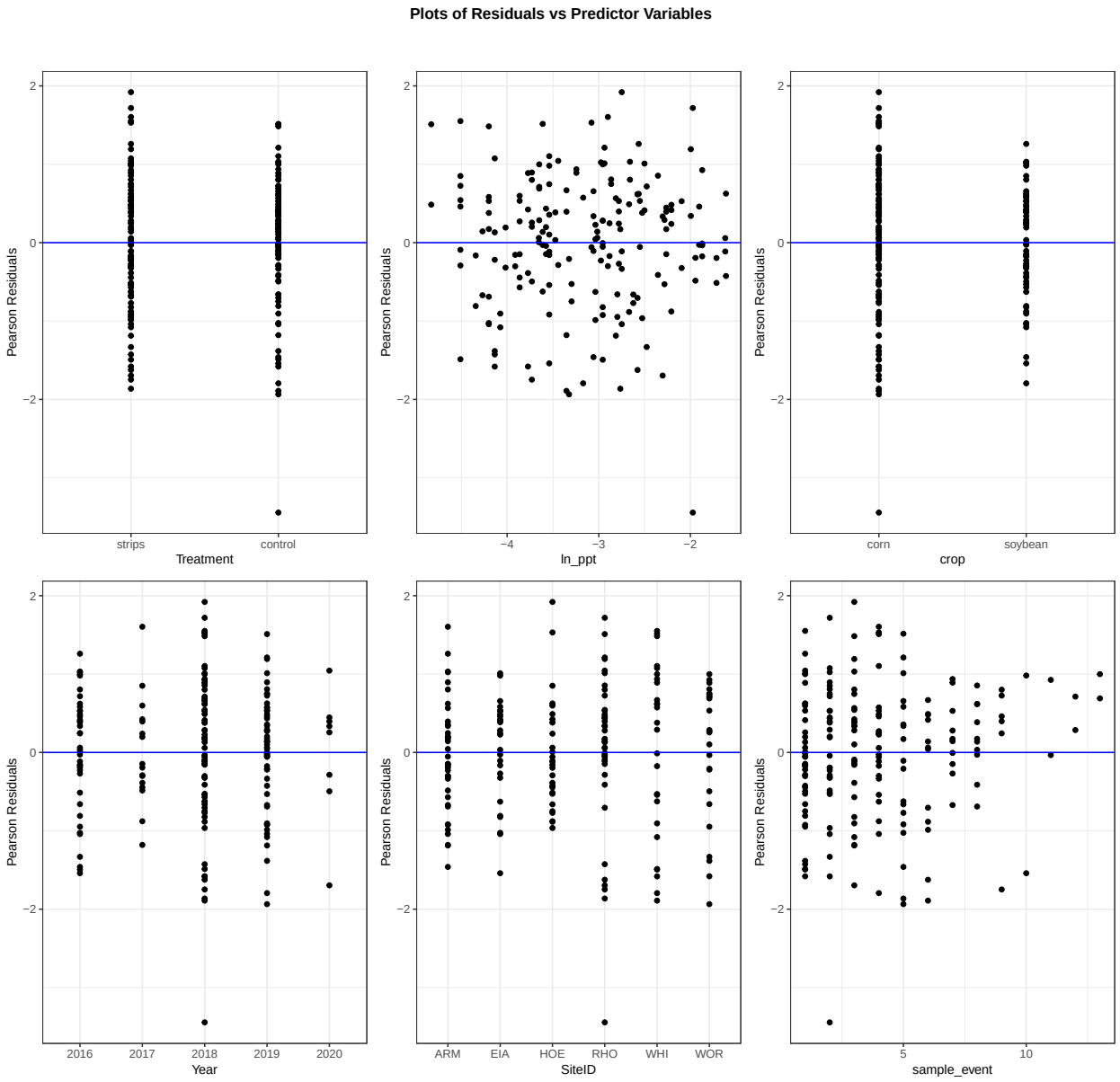
Full model design

```
resid_panel(m_flume,
            plots = c("resid","index","yvp","qq"),
            smoother = TRUE, qqbands = TRUE)
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```



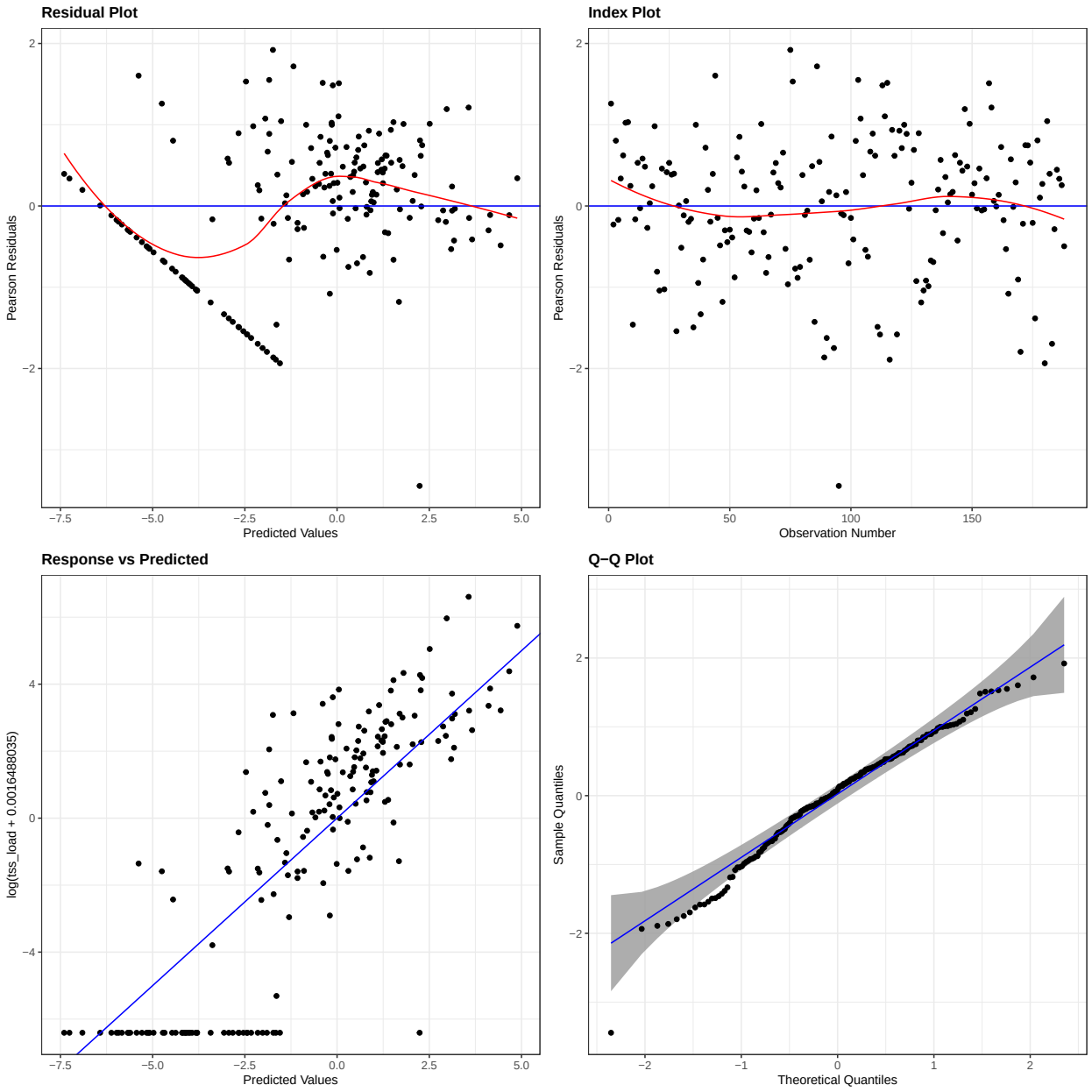
```
resid_xpanel(m_flume)
```



Selected model design

```
resid_panel(m_flume_model,
            plots = c("resid","index","yvp","qq"),
            smoother = TRUE, qqbands = TRUE)
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```




```
resid_xpanel(m_flume_model)
```

Plots of Residuals vs Predictor Variables