Flume: Full Analysis

Jessica Nelson

2022-03-23

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("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")


options(width = 120)

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

```r
sessionInfo()
```

```
## R version 4.1.3 (2022-03-10)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 19044)
##
## 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] broom_0.7.12       rstatix_0.7.0      ggpubr_0.4.0       data.table_1.14.2  ggResidpanel_0.3.0
##  [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] nlme_3.1-155        fs_1.5.2            lubridate_1.8.0     httr_1.4.2          numDeriv_2016.8
##  [6] tools_4.1.3         backports_1.4.1     utf8_1.2.2          R6_2.5.1            DBI_1.1.2
## [11] lazyeval_0.2.2      colorspace_2.0-3    withr_2.5.0         tidyselect_1.1.2    compiler_4.1.3
## [16] cli_3.2.0           rvest_1.0.2         xml2_1.3.3          plotly_4.10.0       scales_1.1.1
## [21] DEoptimR_1.0-10     mvtnorm_1.1-3       robustbase_0.93-9   digest_0.6.29       minqa_1.2.4
## [26] rmarkdown_2.13      qqplotr_0.0.5       pkgconfig_2.0.3     htmltools_0.5.2     dbplyr_2.1.1
## [31] fastmap_1.1.0       htmlwidgets_1.5.4   rlang_1.0.2         readxl_1.3.1        rstudioapi_0.13
## [36] generics_0.1.2      jsonlite_1.8.0      car_3.0-12          magrittr_2.0.1      Rcpp_1.0.8.3
## [41] munsell_0.5.0       fansi_1.0.2         abind_1.4-5         lifecycle_1.0.1     stringi_1.7.6
## [46] yaml_2.3.5          carData_3.0-5       MASS_7.3-55         grid_4.1.3          crayon_1.5.0
## [51] lattice_0.20-45     haven_2.4.3         cowplot_1.1.1       splines_4.1.3       hms_1.1.1
## [56] knitr_1.37          pillar_1.7.0        boot_1.3-28         estimability_1.3    ggsignif_0.6.3
## [61] reprex_2.0.1        glue_1.6.2          evaluate_0.15       modelr_0.1.8        vctrs_0.3.8
## [66] nloptr_2.0.0        tzdb_0.2.0          cellranger_1.1.0    gtable_0.3.0        assertthat_0.2.
## [71] xfun_0.30           xtable_1.8-4        viridisLite_0.4.0   ellipsis_0.3.2
```

Read in data

```
library("tidyverse")

flume <- read_csv("../data/tidy/flume_event_data612_UPDATE.csv") %>%
  mutate(Year = factor(Year)) %>%
  subset(subtreatment != 'grass strip') %>%
  subset(SiteID != 'MCN') %>%
  subset(subset=!(SiteID=="RHO" & Year == 2016)) %>%
  subset(subset=!(SiteID=="RHO" & Year == 2017))
```

```
## Rows: 432 Columns: 19
## -- Column specification ----------------------------------------------------------------
## Delimiter: ","
## chr  (7): SiteID, subtreatment, Treatment, sampleID, random, crop, f_loc
## dbl (12): precipitation, rain_time, rf_event, sample_event, ro_event, Year, flow_time, flow, tss_sum
##
## 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) %>%
  summarize(tss_load = tss_sum,
            ln_tss_load = log(tss_load+0.000198)) %>%
  distinct()
```

```
## `summarise()` has grouped output by 'Treatment', 'Year', 'SiteID', 'sample_event', 'tss_sum', 'crop'
## using the `.groups` argument.
```

```
ppt_sum <- flume %>%
  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'. You can override u
## argument.
```

```
sample_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)
```

```
## `summarise()` has grouped output by 'SiteID', 'Year', 'Treatment', 'sample_event'. You can override u
## argument.
```

```
pivot_sample <- sample_anova %>%
  inner_join(ppt_sum,by=c("SiteID", "Year", "sample_event", "crop")) %>%
  filter(!is.na(strips)) %>%
  mutate(ln_ppt = log(ppt_sum),
         diff = strips-control,
         ln_diff = log(abs(diff)+0.00165),
         ln_ctl = log(control+0.005322915),
         ln_trt = log(strips+0.0104)) %>%
  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, diff, ppt_sum, crop)
```

```
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 == 'MAR', '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',
```

```r
  ifelse(SiteID == 'MCN', 'R',
  ifelse(SiteID == 'HOE', 'NR',
  ifelse(SiteID == 'MAR', 'NR',
  ifelse(SiteID == 'RHO', 'R',
  ifelse(SiteID == 'WHI', 'NR',
  ifelse(SiteID == 'WOR', 'R', 0))))))))))
```

```r
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),
         ln_tss_load = log(tss_load+0.005322915),
         Treatment = Treatment.x) %>%
  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)
```

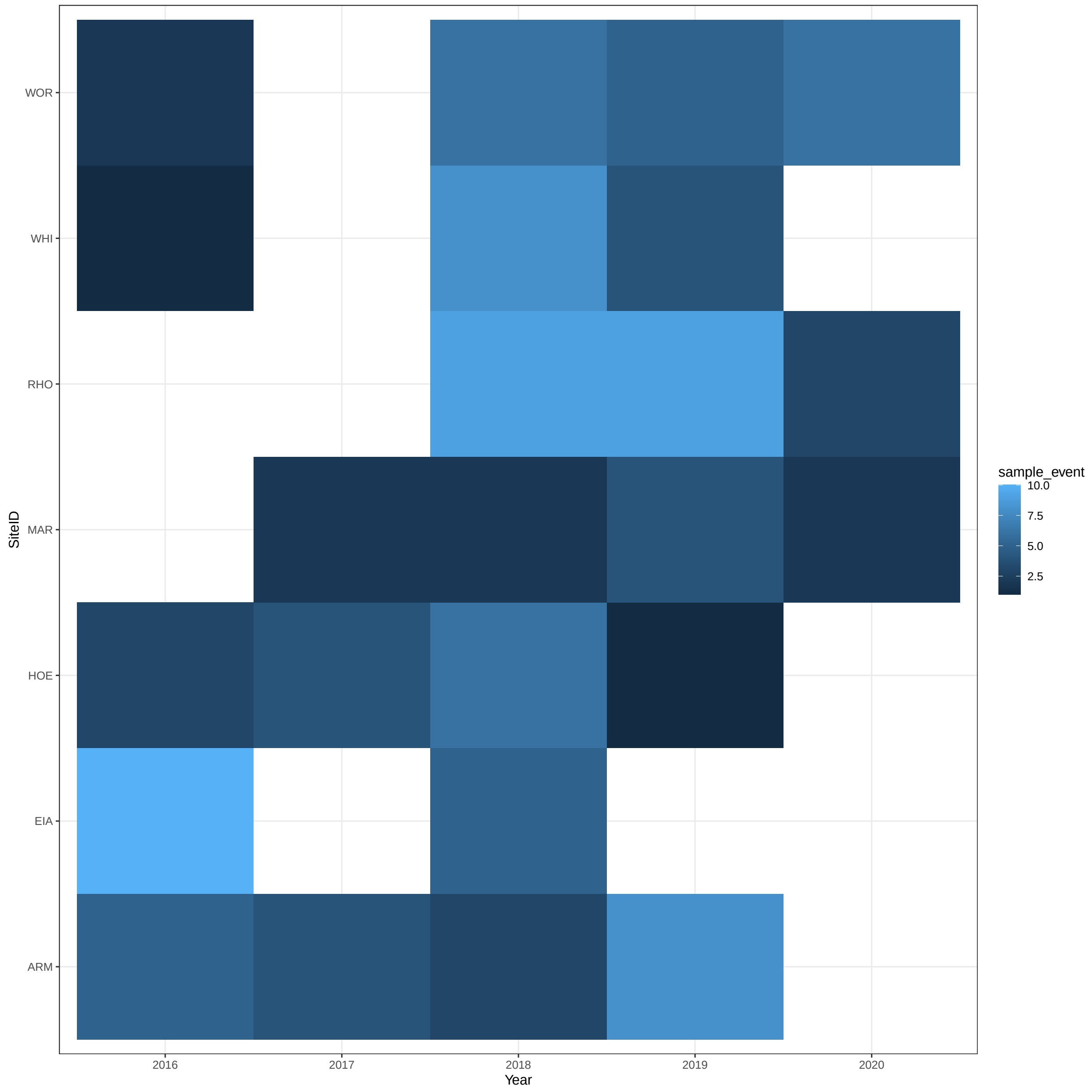
```r
load("full_df.RData")
```

Exploratory analysis

Site-year with rainfall event

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

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



Number of samples

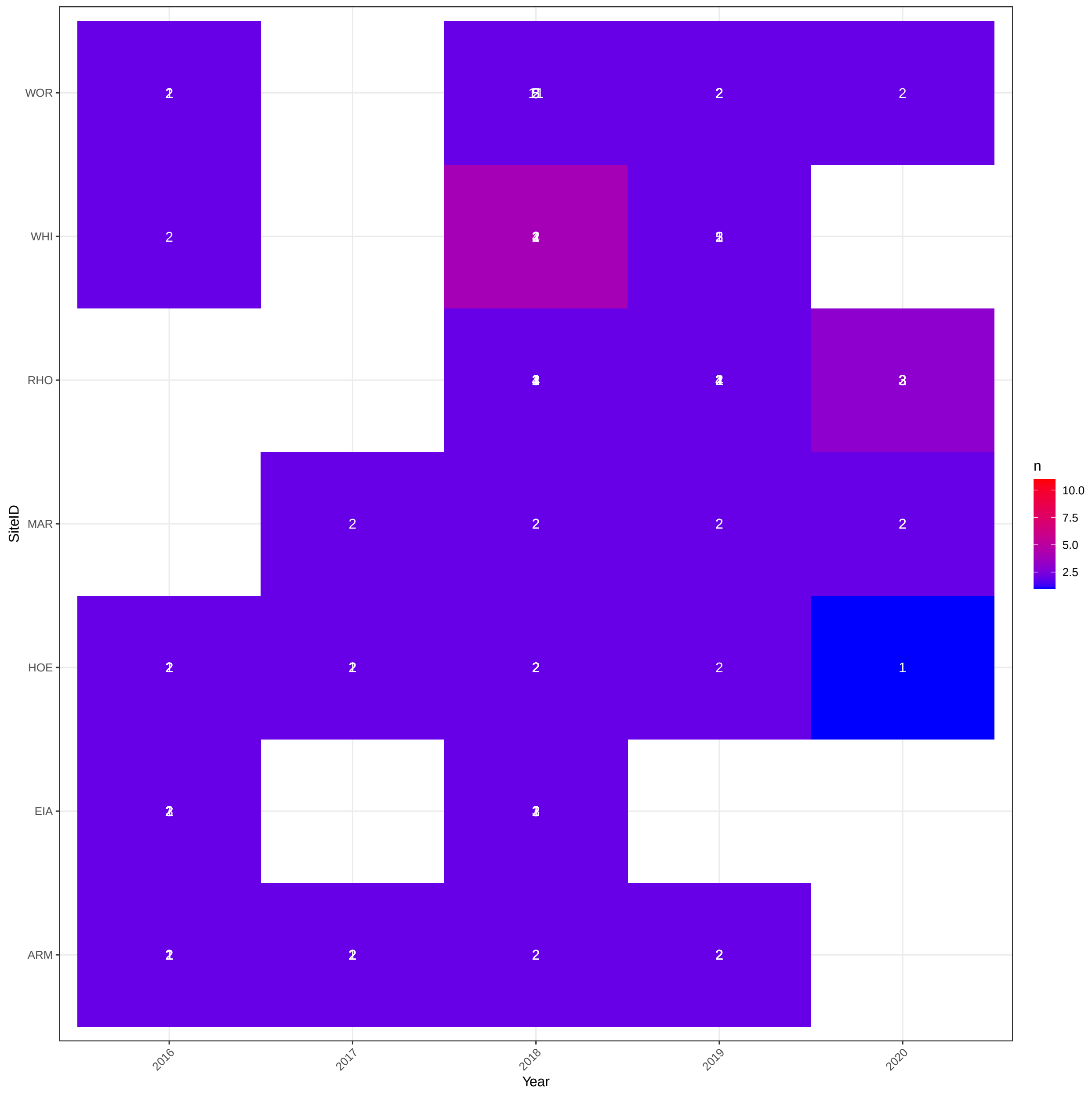
Calculate the number of observations for each treatment-position-year-site-time combination.

```
TSS_counts <- flume %>%
  group_by(Year, SiteID, rf_event) %>%
  distinct() %>%
  summarize(n = n(), .groups = "drop")
```

Plot the number of observations for each combination.

```r
g <- ggplot(TSS_counts, aes(x = Year, y= SiteID, fill = n)) +
  geom_tile() +
  geom_text(aes(label = n), color = "white") +
  scale_fill_gradient(low = "blue", high = "red") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

g
```

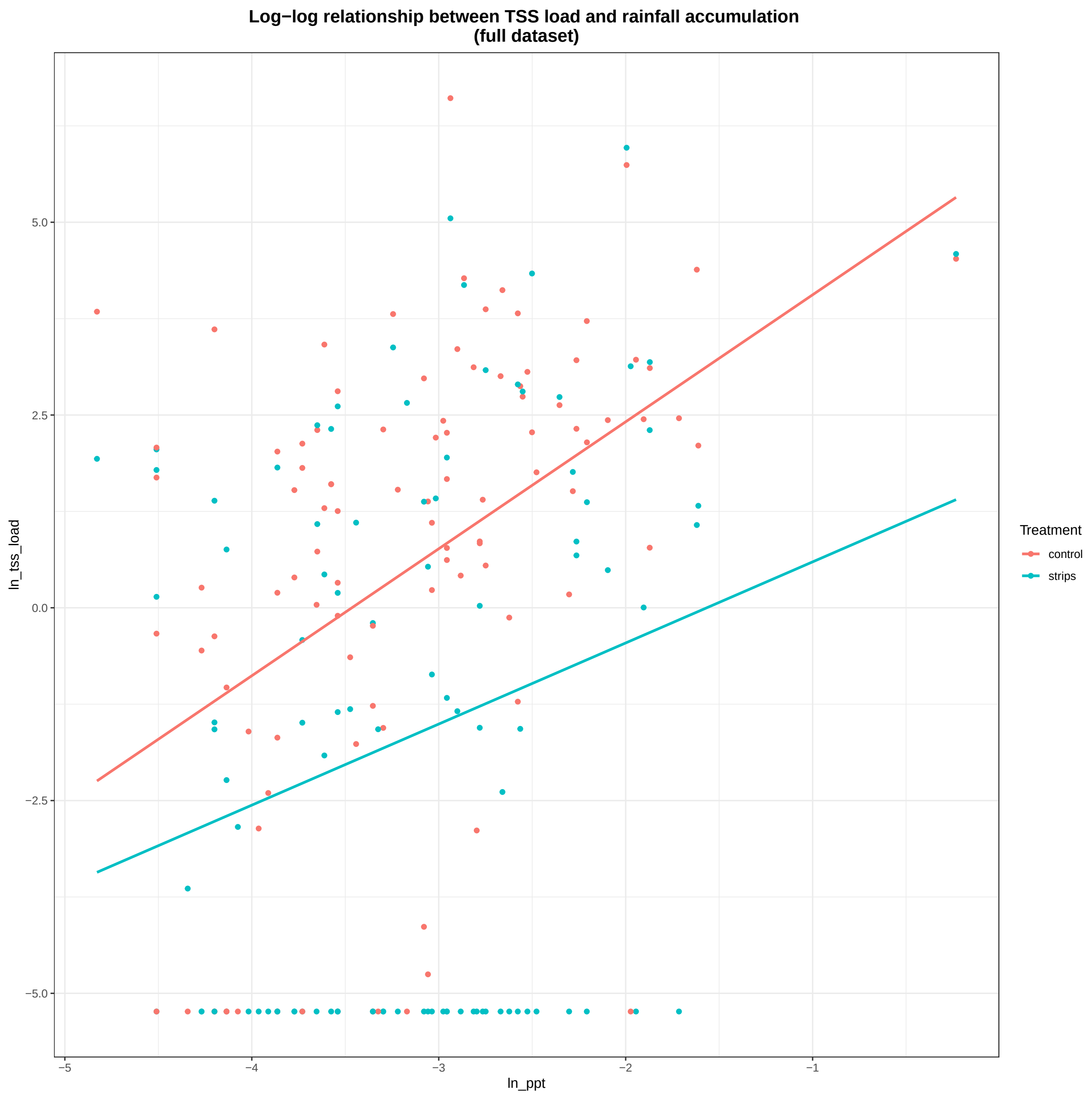


Data visualization

```r
h <- ggplot(full_df, aes(x=ln_ppt, y=ln_tss_load, color=Treatment), inherit.aes = FALSE) +
  geom_point() +
  geom_smooth(method=lm, se=FALSE, 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))

h
```

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



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

```
#{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+0.005322915) ~
                            (1 | SiteID) +
                            (1 | SiteID:Treatment) +
                           #(1|SiteID:Treatment:sample_event) + removed due to singular fit

                            Treatment*ln_ppt +
                           Treatment*crop +

                           Year,
                         data = full_df)
```

```
## boundary (singular) fit: see help('isSingular')
```

```
summary(m_flume)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method ['lmerModLmerTest']
## Formula: log(tss_load + 0.005322915) ~ (1 | SiteID) + (1 | SiteID:Treatment) +
##     Treatment * ln_ppt + Treatment * crop + Year
##    Data: full_df
##
## REML criterion at convergence: 965.3
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.3778 -0.5772  0.0148  0.6231  2.4739
##
## Random effects:
##  Groups           Name        Variance  Std.Dev.
##  SiteID:Treatment (Intercept) 2.785e+00 1.669e+00
##  SiteID           (Intercept) 5.469e-09 7.396e-05
##  Residual                     6.292e+00 2.508e+00
## Number of obs: 204, groups:  SiteID:Treatment, 14; SiteID, 7
##
```

```
## Fixed effects:
##                              Estimate Std. Error       df t value Pr(>|t|)
## (Intercept)                    5.0023     1.2967 100.4190   3.858 0.000202 ***
## Treatmentstrips               -4.7733     1.7434  89.0443  -2.738 0.007465 **
## ln_ppt                         1.6719     0.3141 184.6241   5.323 2.93e-07 ***
## cropsoybean                   -1.3309     0.6687 192.8648  -1.990 0.047968 *
## Year2017                       1.3665     0.7684 189.8879   1.778 0.076928 .
## Year2018                       1.1635     0.5474 188.9205   2.126 0.034827 *
## Year2019                       2.0982     0.5994 191.2516   3.501 0.000578 ***
## Year2020                      -0.4797     0.9251 192.7128  -0.518 0.604715
## Treatmentstrips:ln_ppt        -0.6660     0.4426 184.7193  -1.505 0.134094
## Treatmentstrips:cropsoybean    1.5244     0.9361 191.5921   1.629 0.105062
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) Trtmnt ln_ppt crpsyb Yr2017 Yr2018 Yr2019 Yr2020 Trtm:_
## Trtmntstrps -0.672
## ln_ppt       0.759 -0.572
## cropsoybean -0.226  0.140 -0.012
## Year2017    -0.163  0.000  0.047  0.021
## Year2018    -0.300  0.000  0.019  0.126  0.453
## Year2019    -0.249  0.000  0.069  0.116  0.448  0.685
## Year2020    -0.196  0.000  0.017  0.054  0.293  0.475  0.478
## Trtmntstr:_ -0.546  0.812 -0.705  0.012  0.000  0.000  0.000  0.000
## Trtmntstrp:  0.134 -0.200  0.012 -0.700  0.000  0.000  0.000  0.000 -0.017
## optimizer (nloptwrap) convergence code: 0 (OK)
## boundary (singular) fit: see help('isSingular')
```

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

```
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
## boundary (singular) fit: see help('isSingular')
```

```
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 + 0.005322915) ~ (1 | SiteID) + (1 | SiteID:Treatment) +      Treatment + ln_pp
##    Data: full_df
##
## REML criterion at convergence: 973.8
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.1861 -0.6003  0.0621  0.6738  2.3623
##
## Random effects:
##  Groups           Name        Variance Std.Dev.
##  SiteID:Treatment (Intercept) 2.675    1.636
##  SiteID           (Intercept) 0.000    0.000
##  Residual                     6.415    2.533
## Number of obs: 204, groups:  SiteID:Treatment, 14; SiteID, 7
##
## Fixed effects:
##                  Estimate Std. Error       df t value Pr(>|t|)
## (Intercept)        3.3971     1.0457  57.2941   3.249 0.001942 **
## Treatmentstrips   -2.1234     0.9497  11.6541  -2.236 0.045760 *
## ln_ppt             1.3379     0.2250 187.4208   5.946 1.32e-08 ***
## Year2017           1.3846     0.7750 193.4757   1.787 0.075546 .
## Year2018           1.2787     0.5436 195.0865   2.352 0.019651 *
## Year2019           2.2130     0.5964 196.7014   3.711 0.000269 ***
## Year2020          -0.4016     0.9304 196.5760  -0.432 0.666478
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##              (Intr) Trtmnt ln_ppt Yr2017 Yr2018 Yr2019
## Trtmntstrps -0.454
## ln_ppt       0.658  0.000
## Year2017    -0.196  0.000  0.067
## Year2018    -0.340  0.000  0.029  0.455
## Year2019    -0.277  0.000  0.100  0.449  0.675
## Year2020    -0.228  0.000  0.026  0.291  0.470  0.473
## optimizer (nloptwrap) convergence code: 0 (OK)
## boundary (singular) fit: see help('isSingular')
```

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

```
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
##  control / strips  6.73 6.57 Inf     0.995      45.6
##
## Year = 2017:
##  contrast         ratio   SE  df asymp.LCL asymp.UCL
##  control / strips  6.73 6.57 Inf     0.995      45.6
##
## Year = 2018:
##  contrast         ratio   SE  df asymp.LCL asymp.UCL
##  control / strips  6.73 6.57 Inf     0.995      45.6
##
## Year = 2019:
##  contrast         ratio   SE  df asymp.LCL asymp.UCL
##  control / strips  6.73 6.57 Inf     0.995      45.6
##
## Year = 2020:
##  contrast         ratio   SE  df asymp.LCL asymp.UCL
##  control / strips  6.73 6.57 Inf     0.995      45.6
##
## 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
##  control      0.885 0.6249 Inf    0.2198     3.518
##  strips       0.127 0.0928 Inf    0.0281     0.518
##
## 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 + 0.005) scale
##
## $contrasts
##  contrast         ratio   SE  df asymp.LCL asymp.UCL
##  control / strips  6.73 6.57 Inf     0.995      45.6
##
## 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")
confint(year)
```

```
##  Year response     SE  df asymp.LCL asymp.UCL
##  2016   0.1444 0.0946 Inf    0.0381     0.511
##  2017   0.5818 0.4563 Inf    0.1227     2.688
##  2018   0.4739 0.2615 Inf    0.1592     1.391
##  2019   1.2150 0.7074 Inf    0.3865     3.796
##  2020   0.0873 0.0821 Inf    0.0110     0.521
##
## 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 + 0.005) scale
```

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

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

```
## ln_ppt = -4:
##  contrast         ratio    SE  df asymp.LCL asymp.UCL
##  control / strips  3.85  4.00 Inf      0.50      29.6
##
## ln_ppt = -3:
##  contrast         ratio    SE  df asymp.LCL asymp.UCL
##  control / strips  7.49  7.33 Inf      1.10      51.0
##
## ln_ppt = -2:
##  contrast         ratio    SE  df asymp.LCL asymp.UCL
##  control / strips 14.57 16.13 Inf      1.67     127.5
##
## 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
```

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

confint(crop)$contrasts
```

```
## crop = corn:
##  contrast         ratio    SE  df asymp.LCL asymp.UCL
##  control / strips 14.43 14.68 Inf     1.962     106.1
##
## crop = soybean:
##  contrast         ratio    SE  df asymp.LCL asymp.UCL
##  control / strips  3.14  3.59 Inf     0.335      29.5
##
## Results are averaged over the levels of: Year
## Degrees-of-freedom method: asymptotic
## Confidence level used: 0.95
## Intervals are back-transformed from the log scale
```

Check assumptions

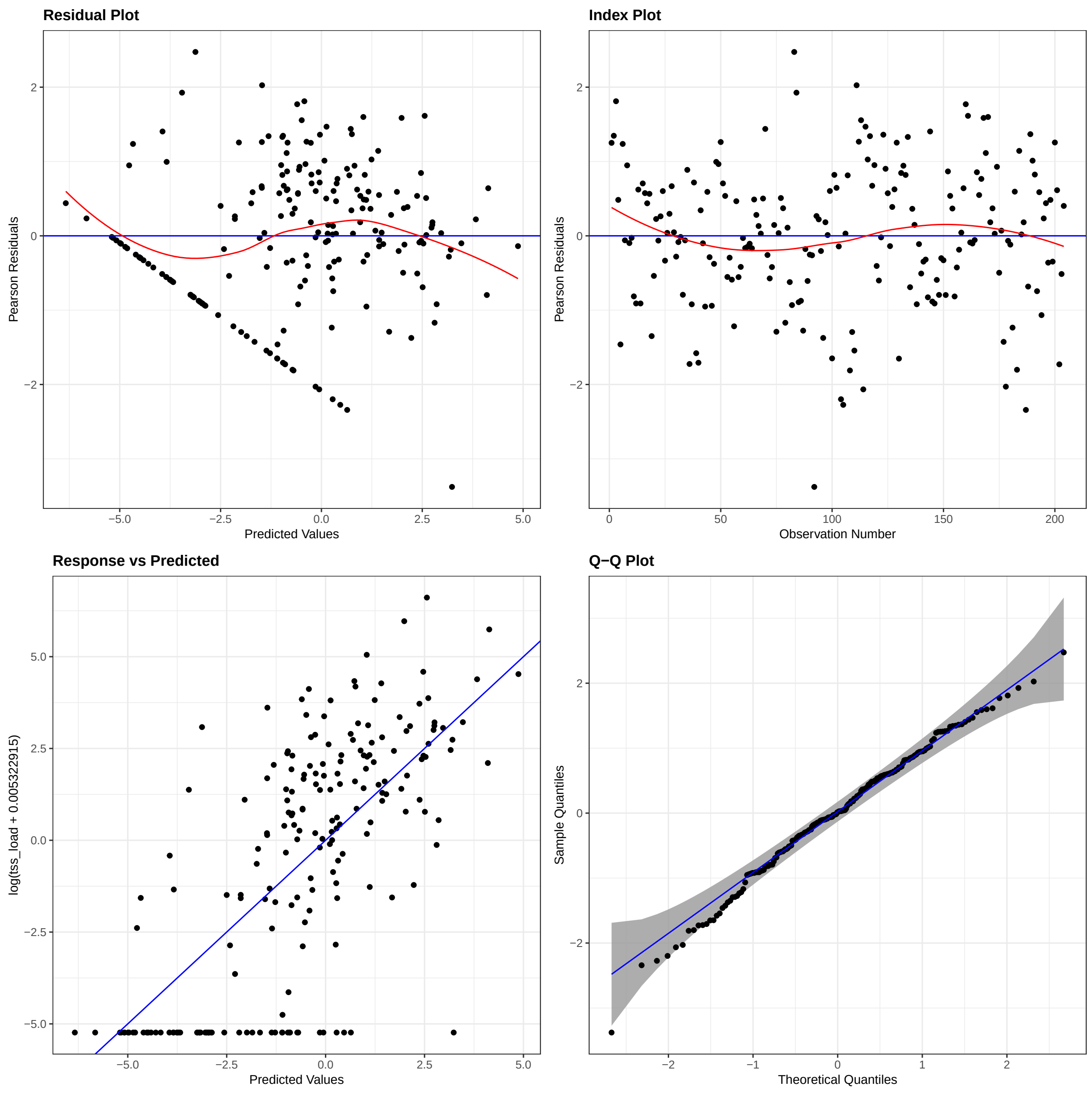
There are two possible models:

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

Full model design

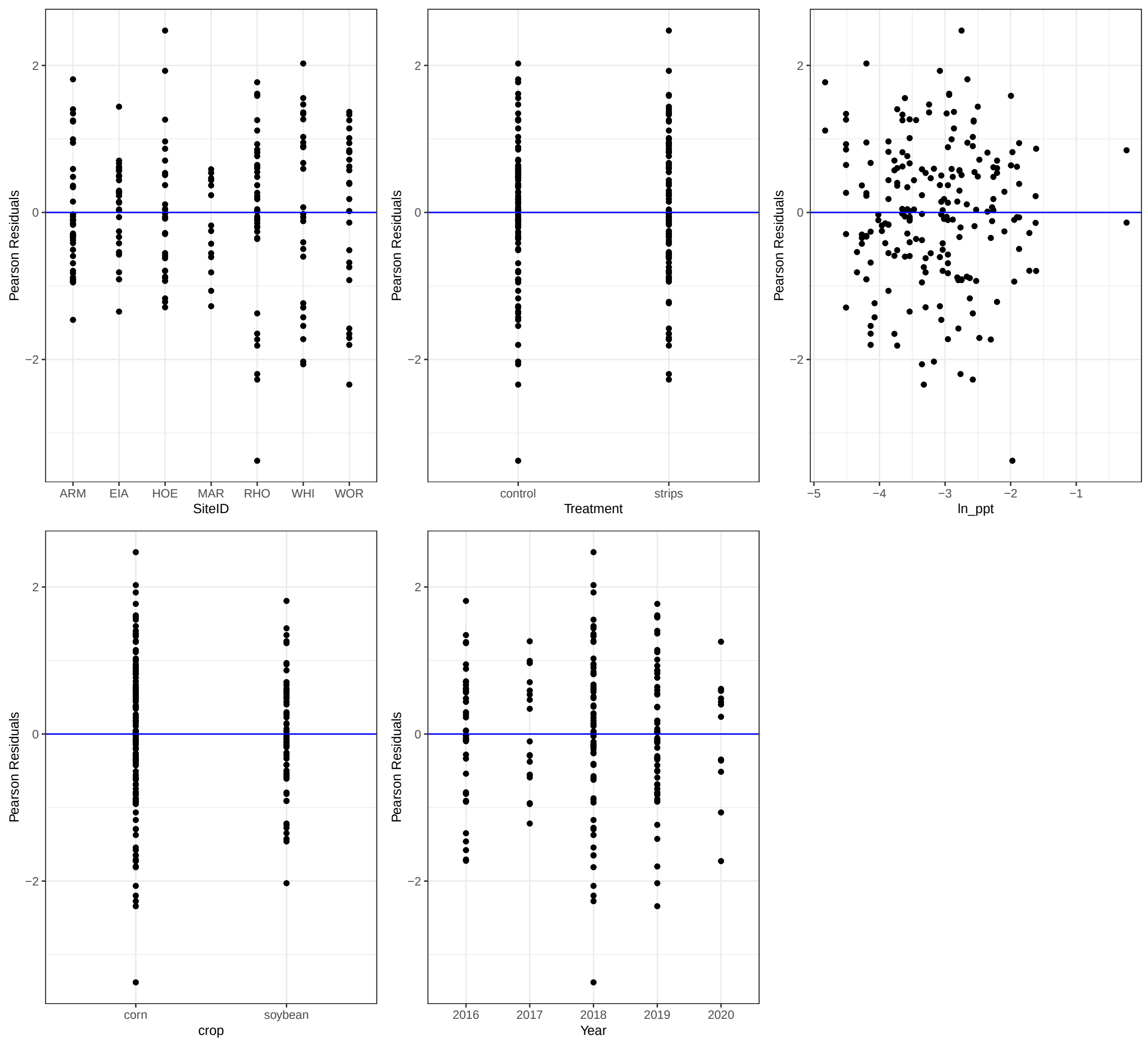
```
resid_panel(m_flume,
            plots = c("resid","index","yvp","qq"),
            smoother = TRUE, qqbands = TRUE)
```

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



```
resid_xpanel(m_flume)
```

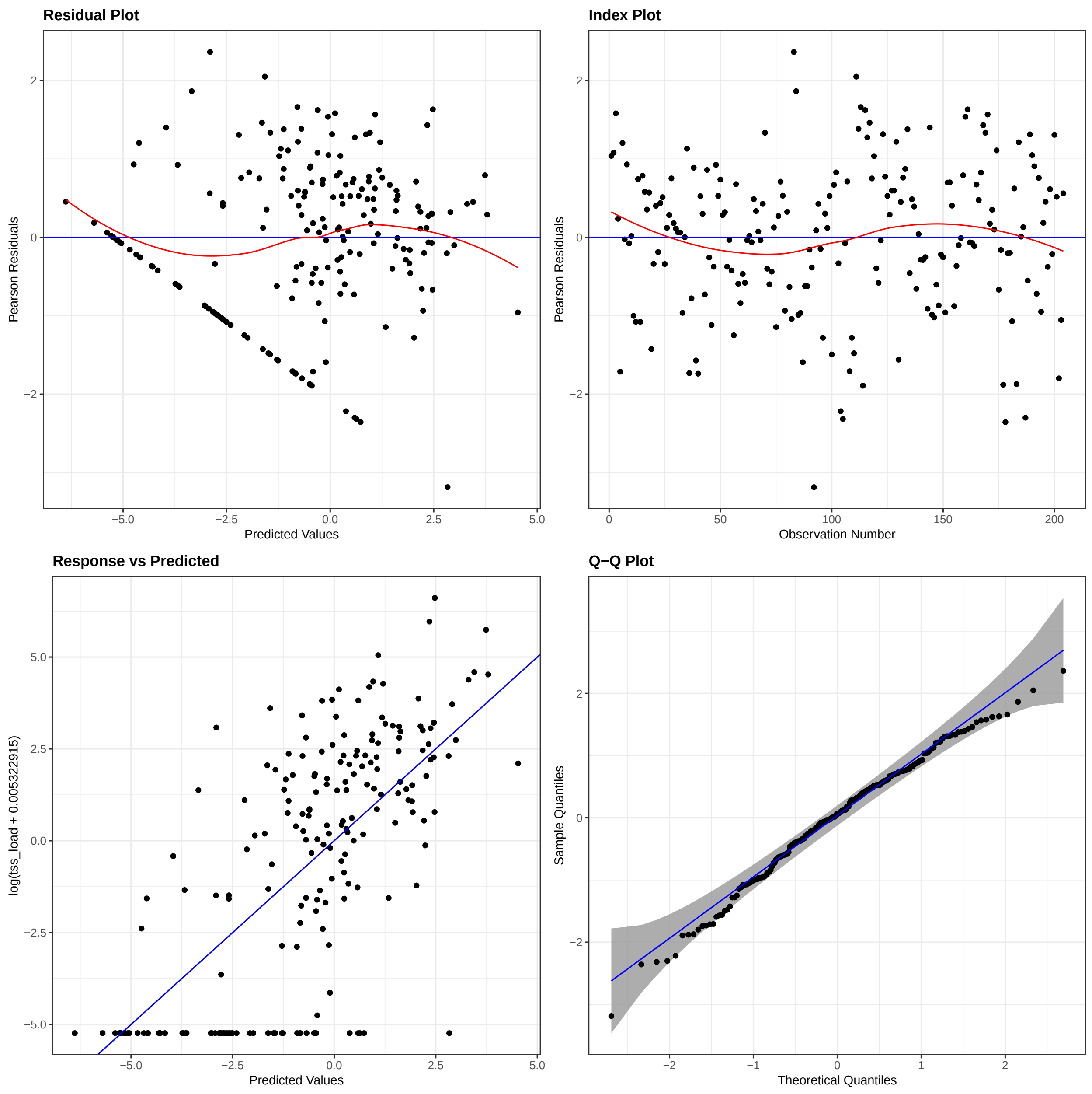
Plots of Residuals vs Predictor Variables



Selected model design

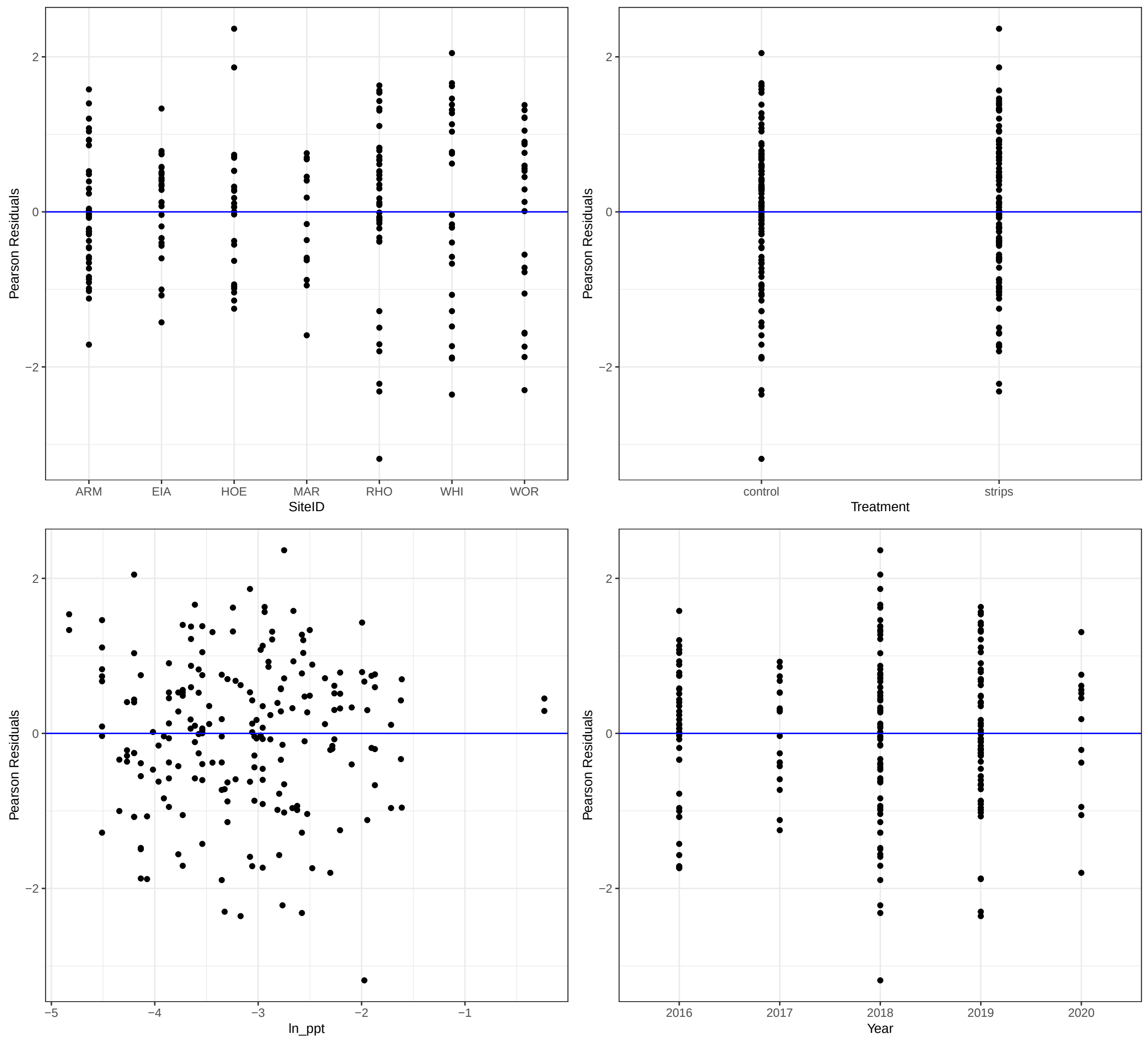
```r
resid_panel(m_flume_model,
            plots = c("resid","index","yvp","qq"),
            smoother = TRUE, qqbands = TRUE)
```

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



```
resid_xpanel(m_flume_model)
```

Plots of Residuals vs Predictor Variables


Pearson Residuals
ARM
EIA
HOE
MAR
RHO
WHI
WOR
SiteID
control
strips
Treatment
ln_ppt
Year