Flume: Full Analysis (March-November)

(adapted from Jarad Niemi - Soilpad Analysis)

Jessica Nelson

2022-04-24

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")


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 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] data.table_1.14.2  ggResidpanel_0.3.0 emmeans_1.7.2      lattice_0.20-45    forcats_0.5.1
##  [7] dplyr_1.0.8        purrr_0.3.4        readr_2.1.2        tidyr_1.2.0        tibble_3.1.6
## [13] tidyverse_1.3.1    lmerTest_3.1-3     lme4_1.1-28        Matrix_1.4-0
##
## 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     glue_1.6.2          digest_0.6.29       rvest_1.0.2
## [16] minqa_1.2.4         colorspace_2.0-3    cowplot_1.1.1       htmltools_0.5.2     pkgconfig_2.0.3
## [21] broom_0.7.12        haven_2.4.3         xtable_1.8-4        mvtnorm_1.1-3       scales_1.1.1
## [26] tzdb_0.2.0          generics_0.1.2      ellipsis_0.3.2      withr_2.5.0         lazyeval_0.2.2
## [31] cli_3.2.0           magrittr_2.0.2      crayon_1.5.0        readxl_1.3.1        estimability_1.3
## [36] evaluate_0.15       fs_1.5.2            fansi_1.0.2         nlme_3.1-155        MASS_7.3-55
## [41] xml2_1.3.3          tools_4.1.3         hms_1.1.1           lifecycle_1.0.1     plotly_4.10.0
## [46] munsell_0.5.0       reprex_2.0.1        qqplotr_0.0.5       compiler_4.1.3      rlang_1.0.2
## [51] grid_4.1.3          nloptr_2.0.0        rstudioapi_0.13     htmlwidgets_1.5.4   rmarkdown_2.13
## [56] boot_1.3-28         gtable_0.3.0        DBI_1.1.2           R6_2.5.1            lubridate_1.8.0
## [61] knitr_1.38          fastmap_1.1.0       utf8_1.2.2          stringi_1.7.6       Rcpp_1.0.8
## [66] vctrs_0.3.8         DEoptimR_1.0-10     dbplyr_2.1.1        tidyselect_1.1.2    xfun_0.30
```

Read in data

```
library("tidyverse")

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: 21
## -- Column specification --------------------------------------------------------------------------
## Delimiter: ","
## chr  (8): SiteID, subtreatment, Treatment, sampleID, rf_event, random, crop, f_loc
## dbl (13): precipitation, rain_time, slope75, Lsfactor, sample_event, ro_event, Year, flow_time, flow
##
## 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) %>%
  summarize(tss_load = tss_sum) %>%
  distinct()
```

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

```
ppt_sum <- flume %>%
  group_by(Treatment, Year, SiteID, sample_event, crop, Lsfactor, slope75) %>%
  #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', 'Lsfactor
## 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`
## `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, slope75, Lsfactor)
```

```
## 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)))))))))
```

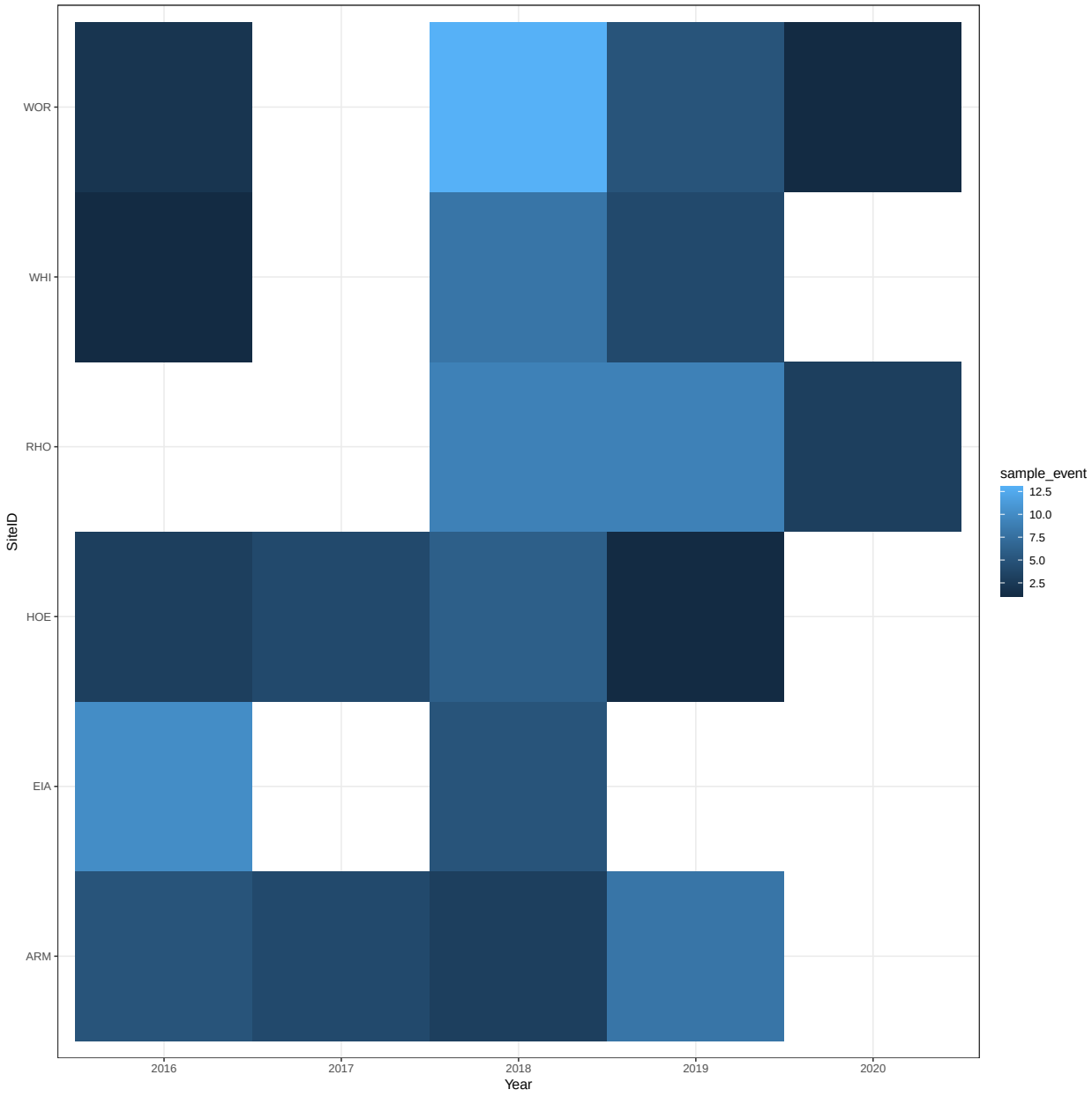
```
full_df <- rf_ro_pivot %>%
  inner_join(ppt_sum,by=c("SiteID", "Year", "sample_event","crop")) %>%
  drop_na(tss_load) %>%
  mutate(ppt_sum = ppt_sum.x,
         ln_ppt = log(ppt_sum.x),
         slope75 = slope75.x,
         LSfactor = Lsfactor.x,
         Treatment = factor(Treatment.x, levels=c('strips','control'))) %>%
  subset(select = -c(Treatment.y, Treatment.x, ppt_sum.x, ppt_sum.y, Lsfactor.x, Lsfactor.y, slope75.x,
  arrange(Year, SiteID, Treatment, sample_event)

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

write.csv(full_df,"../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

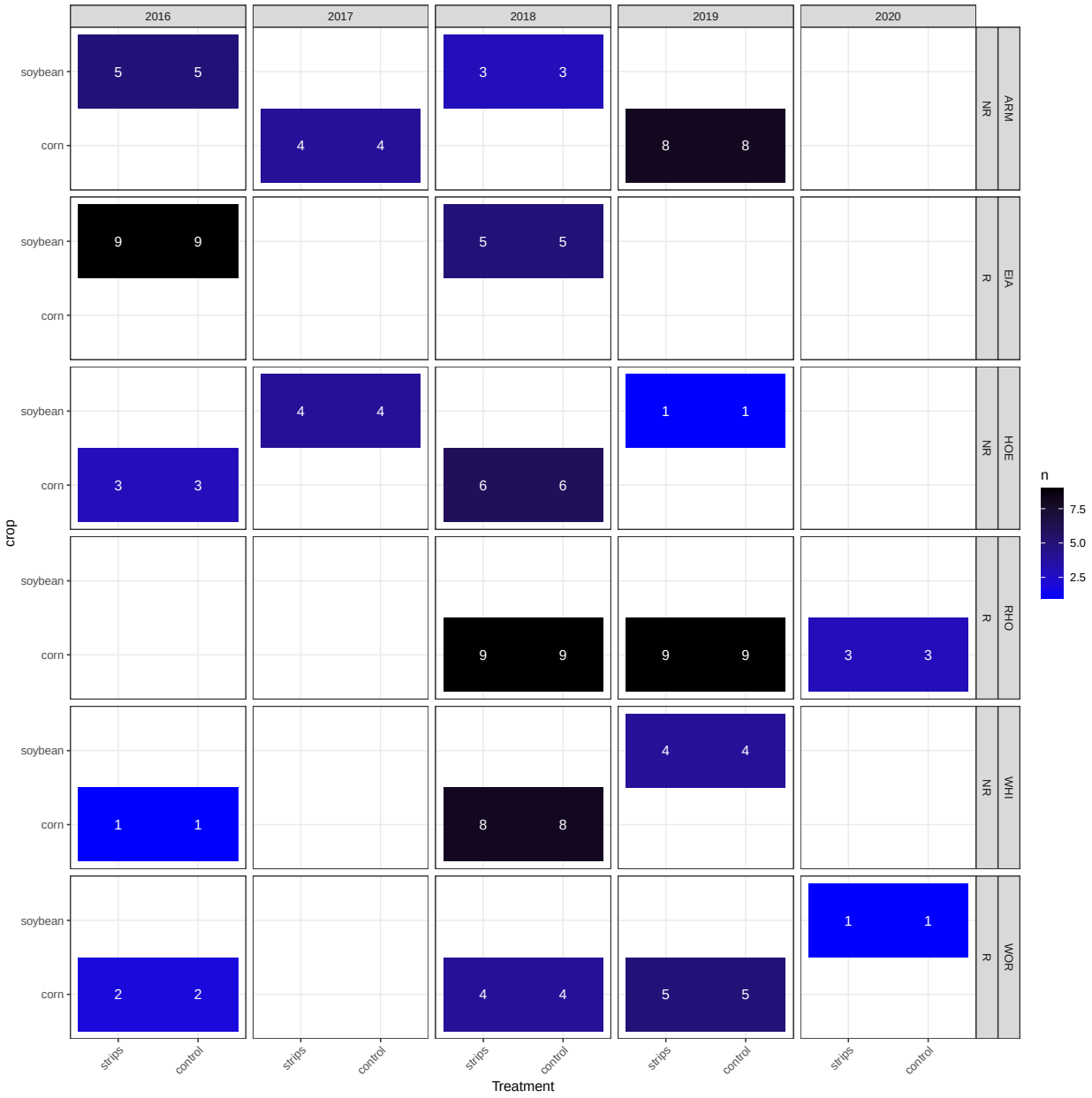
Number of samples

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.

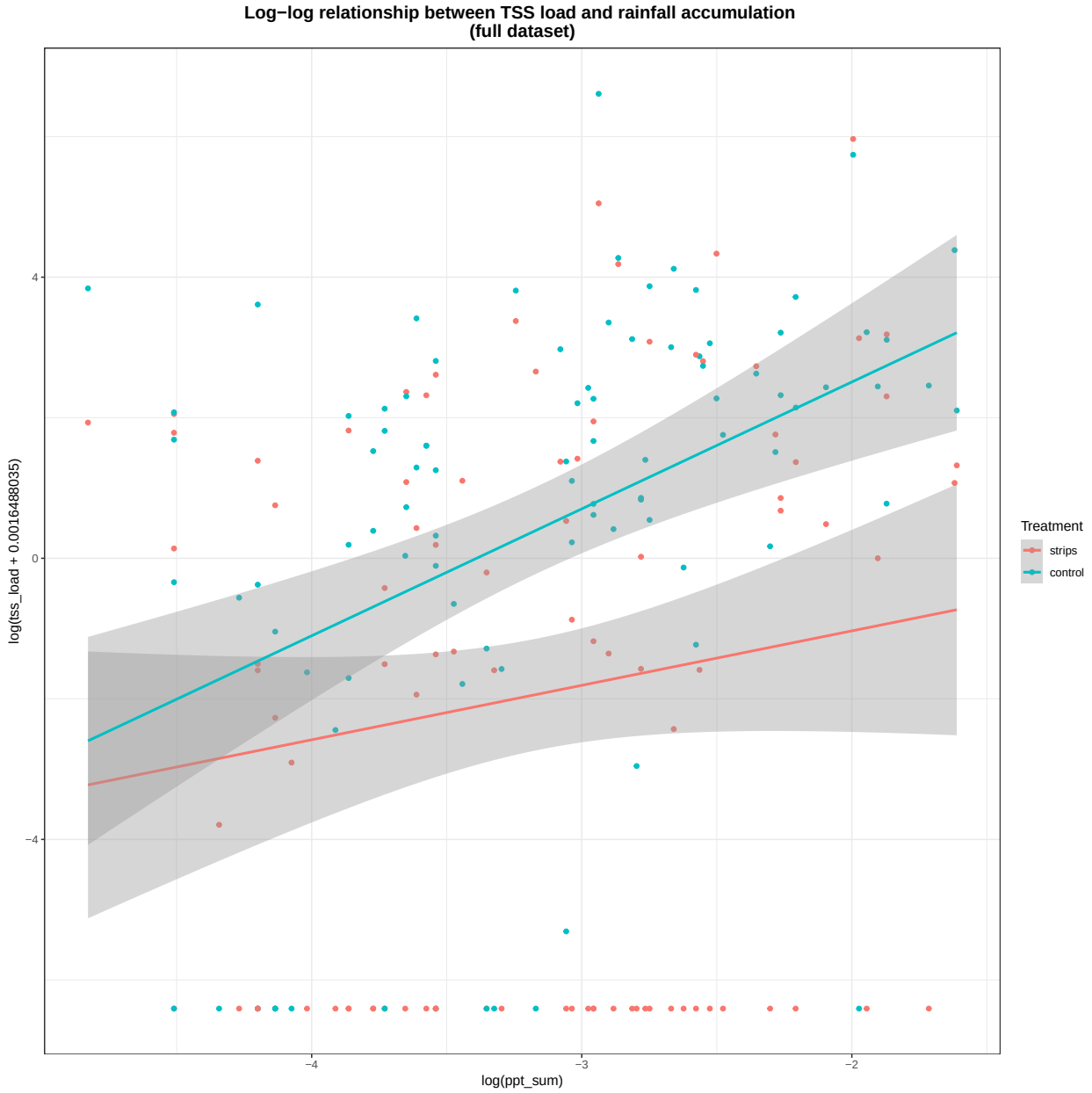
```
g = ggplot(TSS_counts, aes(x = Treatment, y = crop, fill = n)) +
  geom_tile() +
  geom_text(aes(label = n), color = "white") +
  facet_grid(SiteID + random ~ Year) +
  scale_fill_gradient(low = "blue", high = "black") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

g
```




```
h <- ggplot(full_df, aes(x=log(ppt_sum), y=log(tss_load+0.0016488035), color=Treatment), inherit.aes = F
  geom_point() +
  #scale_y_log10() +
  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))

h
```

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



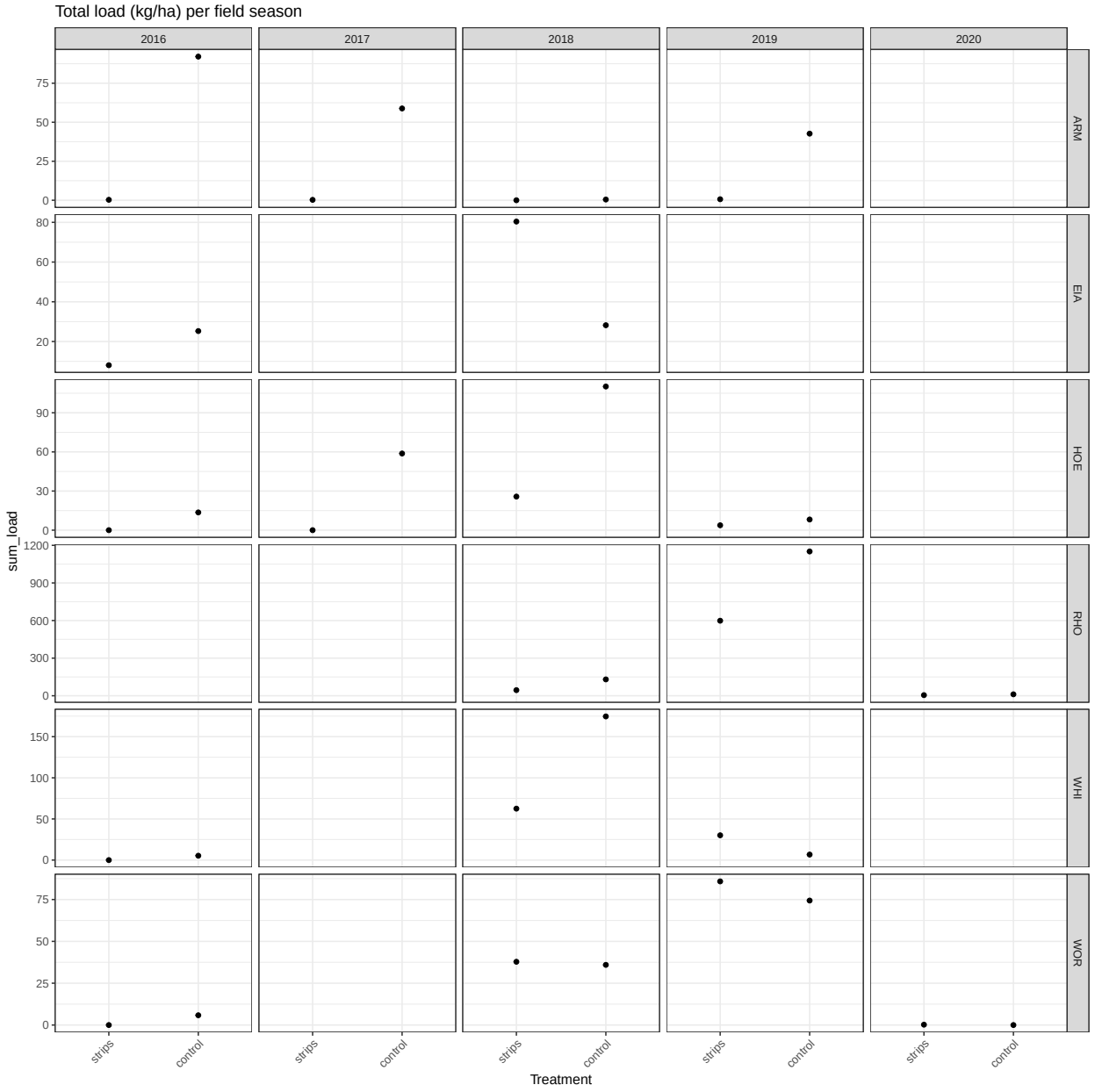

```
#ggsave("fig/randReg_ppt_load.png", h, width = 12, height = 12)
```

```
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,
            #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")
```

```
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 +

                           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 + Year * Treatment +
##     (1 | SiteID:Treatment) + (1 | Year:sample_event) + (1 | SiteID:Year:sample_event)
##    Data: full_df
##
## REML criterion at convergence: 920.9
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.4385 -0.5283  0.0774  0.5389  1.8626
##
## Random effects:
##  Groups                    Name        Variance Std.Dev.
##  SiteID:Year:sample_event (Intercept) 1.5373   1.2399
##  Year:sample_event        (Intercept) 0.3959   0.6292
##  SiteID:Treatment         (Intercept) 3.0827   1.7558
##  Residual                             6.4680   2.5432
## 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)                 -0.6119     1.5786  97.0780  -0.388 0.699144
## Treatmentcontrol             5.8358     2.0101  64.7217   2.903 0.005046 **
## ln_ppt                       0.9341     0.3954 160.1333   2.363 0.019351 *
## Year2017                     0.4611     1.3404 104.7098   0.344 0.731509
## Year2018                     1.9302     0.9297  68.7092   2.076 0.041616 *
## Year2019                     3.6123     1.0140  77.3950   3.563 0.000633 ***
## Year2020                     0.9842     1.7520 136.4887   0.562 0.575186
## Treatmentcontrol:ln_ppt      0.8935     0.4927  83.0996   1.813 0.073368 .
## Treatmentcontrol:Year2017    1.6650     1.6271  86.1177   1.023 0.309036
## Treatmentcontrol:Year2018   -0.7665     1.1073  86.9266  -0.692 0.490596
```

```
## Treatmentcontrol:Year2019  -2.3875     1.2163  88.4991  -1.963 0.052787 .
## Treatmentcontrol:Year2020  -2.9438     2.1529  86.5865  -1.367 0.175039
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) Trtmnt ln_ppt Yr2017 Yr2018 Yr2019 Yr2020 Trtm:_ T:Y2017 T:Y2018 T:Y2019
## Trtmntcntrl -0.637
## ln_ppt       0.759 -0.463
## Year2017    -0.181  0.098  0.089
## Year2018    -0.343  0.189  0.051  0.426
## Year2019    -0.293  0.168  0.093  0.420  0.649
## Year2020    -0.233  0.133 -0.011  0.222  0.409  0.413
## Trtmntcnt:_ -0.473  0.744 -0.623 -0.055 -0.036 -0.053  0.005
## Trtmn:Y2017  0.103 -0.162 -0.056 -0.607 -0.251 -0.248 -0.130  0.091
## Trtmn:Y2018  0.203 -0.318 -0.037 -0.256 -0.596 -0.395 -0.251  0.060  0.421
## Trtmn:Y2019  0.178 -0.280 -0.055 -0.251 -0.392 -0.600 -0.256  0.088  0.413   0.658
## Trtmn:Y2020  0.138 -0.217  0.005 -0.128 -0.244 -0.250 -0.614 -0.008  0.211   0.409   0.417
```

```
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 + Year * Treatment +
##     (1 | SiteID:Treatment) + (1 | Year:sample_event) + (1 | SiteID:Year:sample_event)
##    Data: full_df
##
## REML criterion at convergence: 920.9
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.4385 -0.5283  0.0774  0.5389  1.8626
##
## Random effects:
##  Groups                    Name        Variance Std.Dev.
##  SiteID:Year:sample_event (Intercept) 1.5373   1.2399
##  Year:sample_event        (Intercept) 0.3959   0.6292
##  SiteID:Treatment         (Intercept) 3.0827   1.7558
##  Residual                             6.4680   2.5432
## 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)                 -0.6119     1.5786  97.0780  -0.388 0.699144
## Treatmentcontrol             5.8358     2.0101  64.7217   2.903 0.005046 **
## ln_ppt                       0.9341     0.3954 160.1333   2.363 0.019351 *
## Year2017                     0.4611     1.3404 104.7098   0.344 0.731509
## Year2018                     1.9302     0.9297  68.7092   2.076 0.041616 *
## Year2019                     3.6123     1.0140  77.3950   3.563 0.000633 ***
## Year2020                     0.9842     1.7520 136.4887   0.562 0.575186
## Treatmentcontrol:ln_ppt      0.8935     0.4927  83.0996   1.813 0.073368 .
## Treatmentcontrol:Year2017    1.6650     1.6271  86.1177   1.023 0.309036
## Treatmentcontrol:Year2018   -0.7665     1.1073  86.9266  -0.692 0.490596
```

```
## Treatmentcontrol:Year2019  -2.3875     1.2163  88.4991  -1.963 0.052787 .
## Treatmentcontrol:Year2020  -2.9438     2.1529  86.5865  -1.367 0.175039
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) Trtmnt ln_ppt Yr2017 Yr2018 Yr2019 Yr2020 Trtm:_ T:Y2017 T:Y2018 T:Y2019
## Trtmntcntrl -0.637
## ln_ppt       0.759 -0.463
## Year2017    -0.181  0.098  0.089
## Year2018    -0.343  0.189  0.051  0.426
## Year2019    -0.293  0.168  0.093  0.420  0.649
## Year2020    -0.233  0.133 -0.011  0.222  0.409  0.413
## Trtmntcnt:_ -0.473  0.744 -0.623 -0.055 -0.036 -0.053  0.005
## Trtmn:Y2017  0.103 -0.162 -0.056 -0.607 -0.251 -0.248 -0.130  0.091
## Trtmn:Y2018  0.203 -0.318 -0.037 -0.256 -0.596 -0.395 -0.251  0.060  0.421
## Trtmn:Y2019  0.178 -0.280 -0.055 -0.251 -0.392 -0.600 -0.256  0.088  0.413   0.658
## Trtmn:Y2020  0.138 -0.217  0.005 -0.128 -0.244 -0.250 -0.614 -0.008  0.211   0.409   0.417
```

```
##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 0.0492 0.0661 Inf  0.003518     0.687
##
## Year = 2017:
##  contrast          ratio     SE  df asymp.LCL asymp.UCL
##  strips / control 0.0093 0.0160 Inf  0.000322     0.268
##
## Year = 2018:
##  contrast          ratio     SE  df asymp.LCL asymp.UCL
##  strips / control 0.1058 0.1258 Inf  0.010286     1.088
##
## Year = 2019:
##  contrast          ratio     SE  df asymp.LCL asymp.UCL
##  strips / control 0.5352 0.6742 Inf  0.045302     6.322
##
## Year = 2020:
##  contrast          ratio     SE  df asymp.LCL asymp.UCL
##  strips / control 0.9334 2.0079 Inf  0.013770    63.267
##
## 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       0.113 0.0958 Inf    0.0207     0.588
##  control      0.960 0.8032 Inf    0.1853     4.942
##
## 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(mu + 0.002) scale
##
## $contrasts
##  contrast         ratio    SE  df asymp.LCL asymp.UCL
##  strips / control 0.119 0.136 Inf    0.0128      1.11
##
## 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
```

```
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   0.1262 0.0995 Inf   0.02619     0.586
##  2017   0.4645 0.4793 Inf   0.06048     3.496
##  2018   0.5989 0.3996 Inf   0.16134     2.211
##  2019   1.4341 1.0300 Inf   0.35028     5.856
##  2020   0.0769 0.1032 Inf   0.00432     1.031
##
## Results are averaged over the levels of: Treatment
## 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,-3.5,-3,-2)),
                    type = "response",
```

```
                    lmer.df = "asymptotic")

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

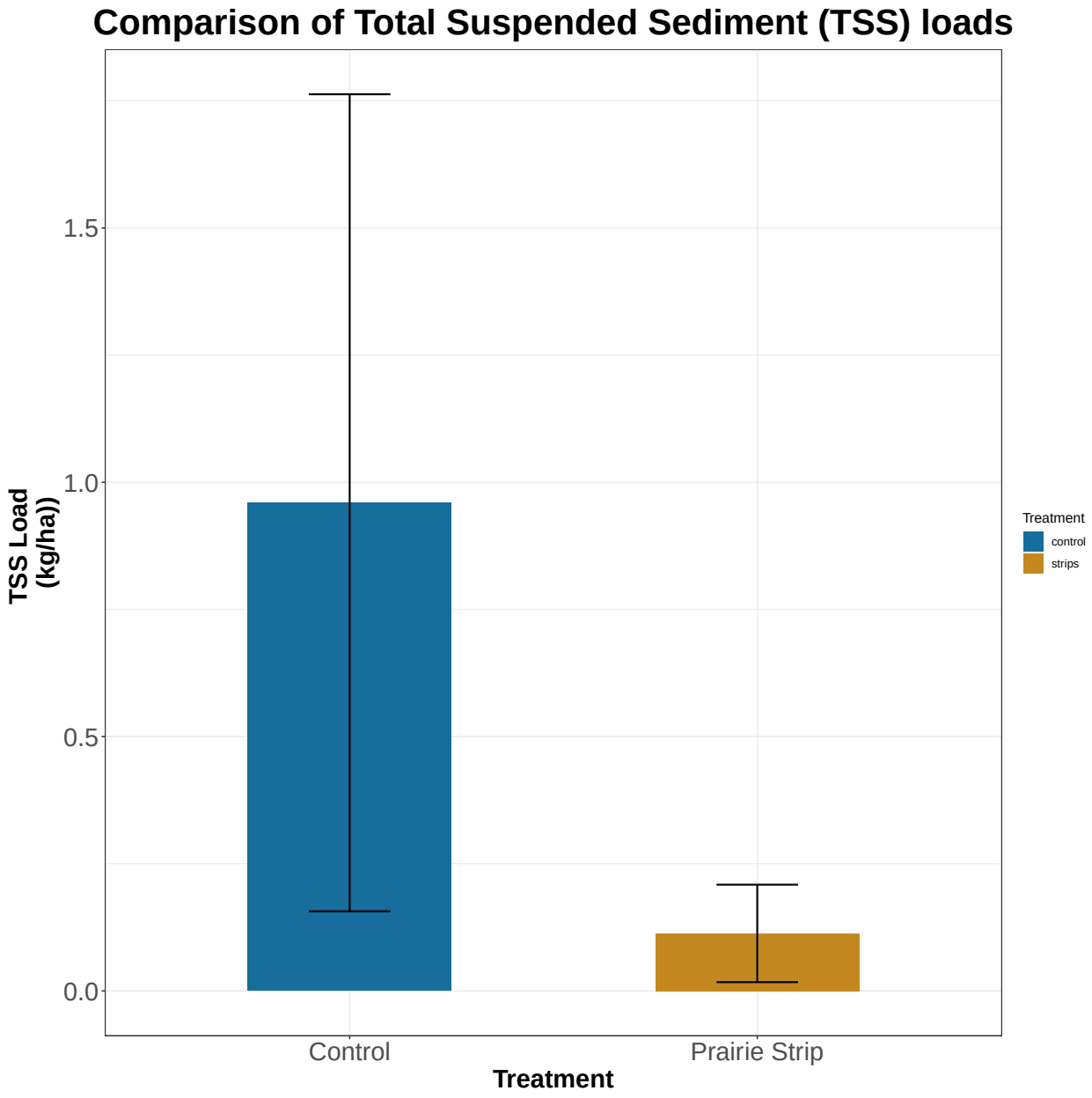
```
## ln_ppt = -4:
##  contrast          ratio    SE  df asymp.LCL asymp.UCL
##  strips / control 0.2528 0.307 Inf   0.02343     2.727
##
## ln_ppt = -3.5:
##  contrast          ratio    SE  df asymp.LCL asymp.UCL
##  strips / control 0.1617 0.186 Inf   0.01689     1.548
##
## ln_ppt = -3:
##  contrast          ratio    SE  df asymp.LCL asymp.UCL
##  strips / control 0.1034 0.118 Inf   0.01101     0.971
##
## ln_ppt = -2:
##  contrast          ratio    SE  df asymp.LCL asymp.UCL
##  strips / control 0.0423 0.054 Inf   0.00348     0.515
##
## 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
```

```
#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") +
  xlab("Treatment") +
  ylab("TSS Load \n(kg/ha))") +
  theme(plot.title = element_text(size=28, face="bold", hjust=0.5),
        axis.title.x = element_text(size=20, face="bold"),
        axis.title.y = element_text(size=20, face="bold"),
```

```
        axis.text.x = element_text(size=20),
        axis.text.y = element_text(size=20)) +
  scale_x_discrete(labels= c("Control", "Prairie Strip"))

trt_plot
```

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



Check assumptions

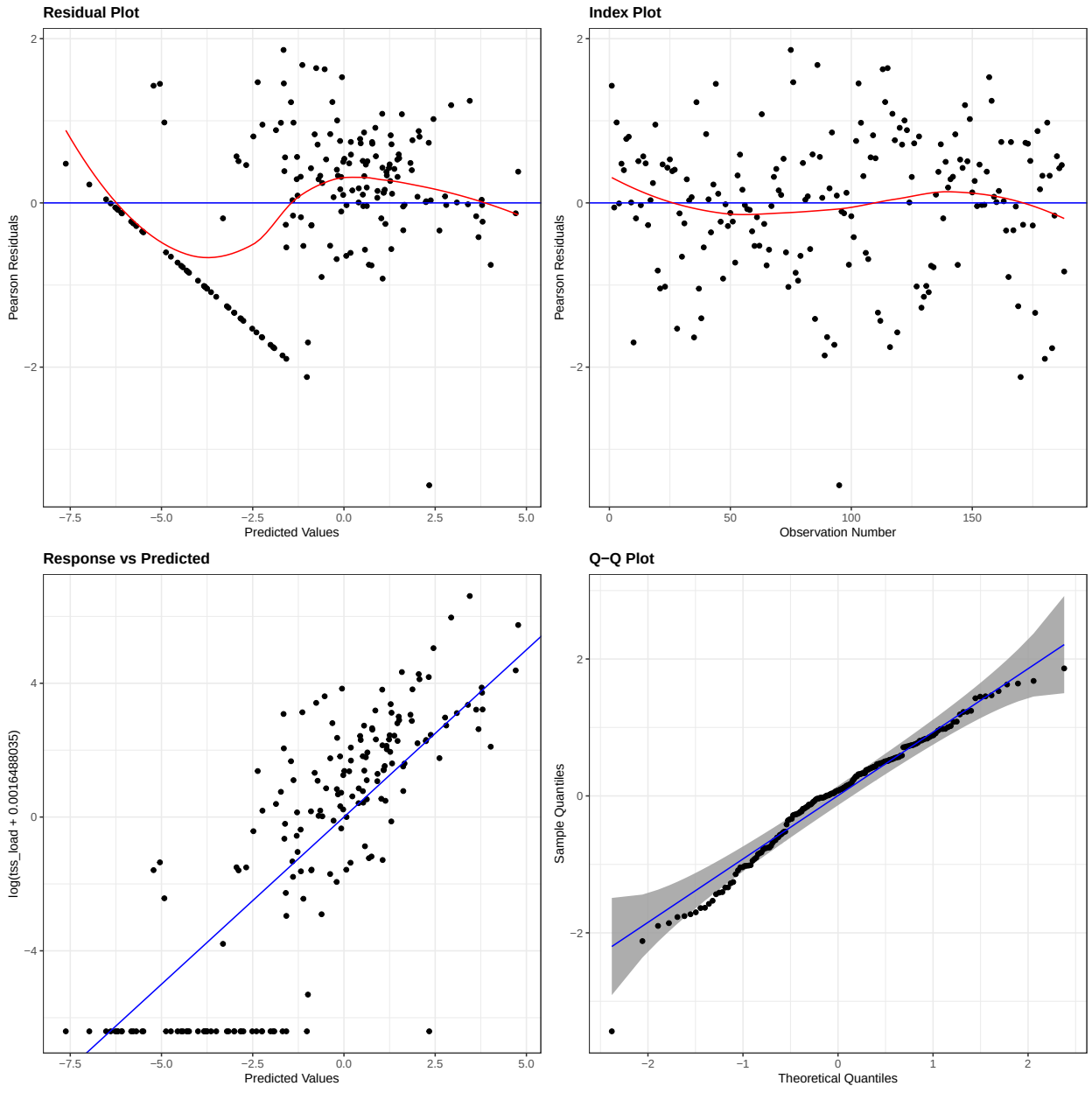
There are two possible models:

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

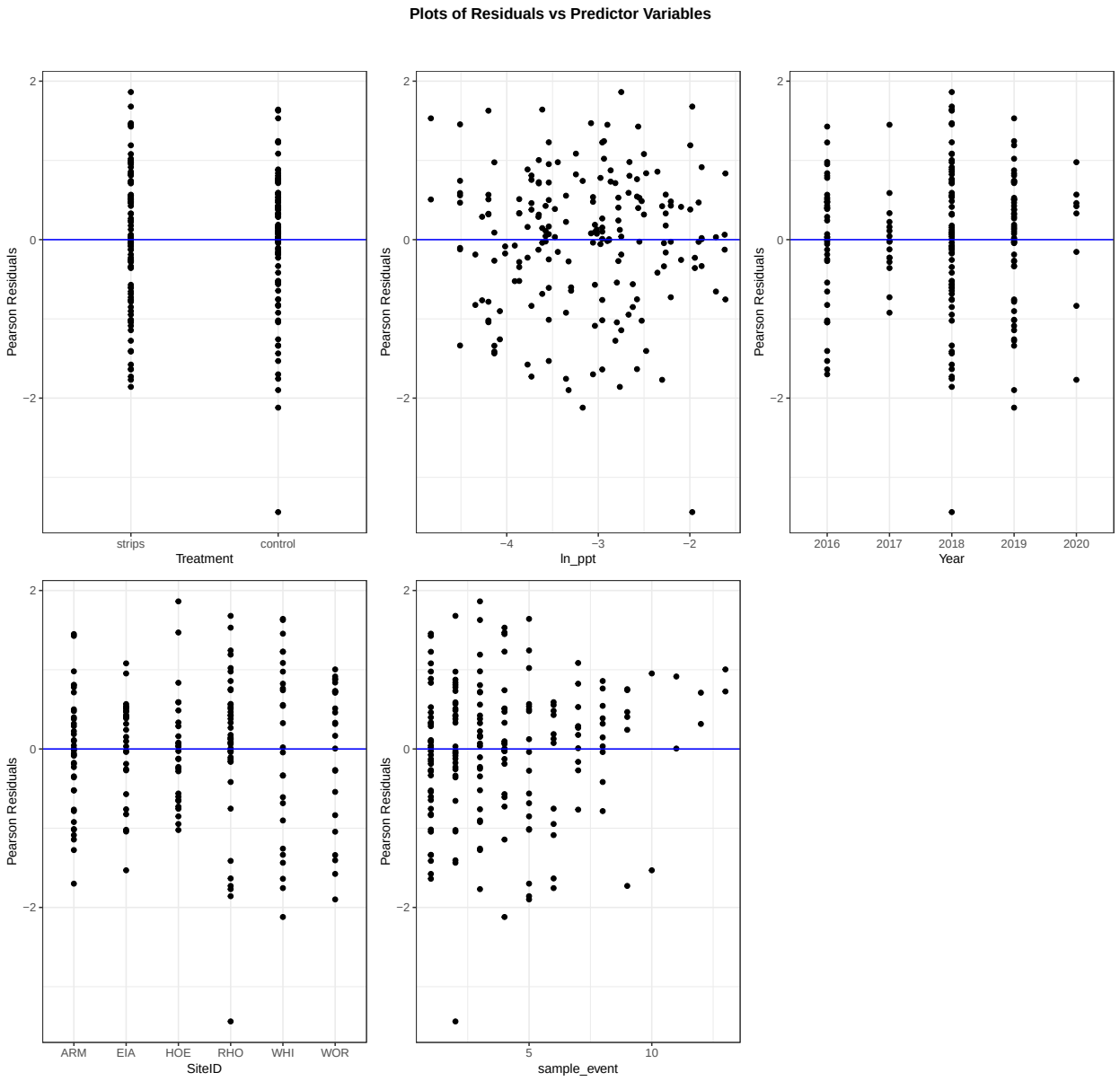
Full model design

```
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'
```



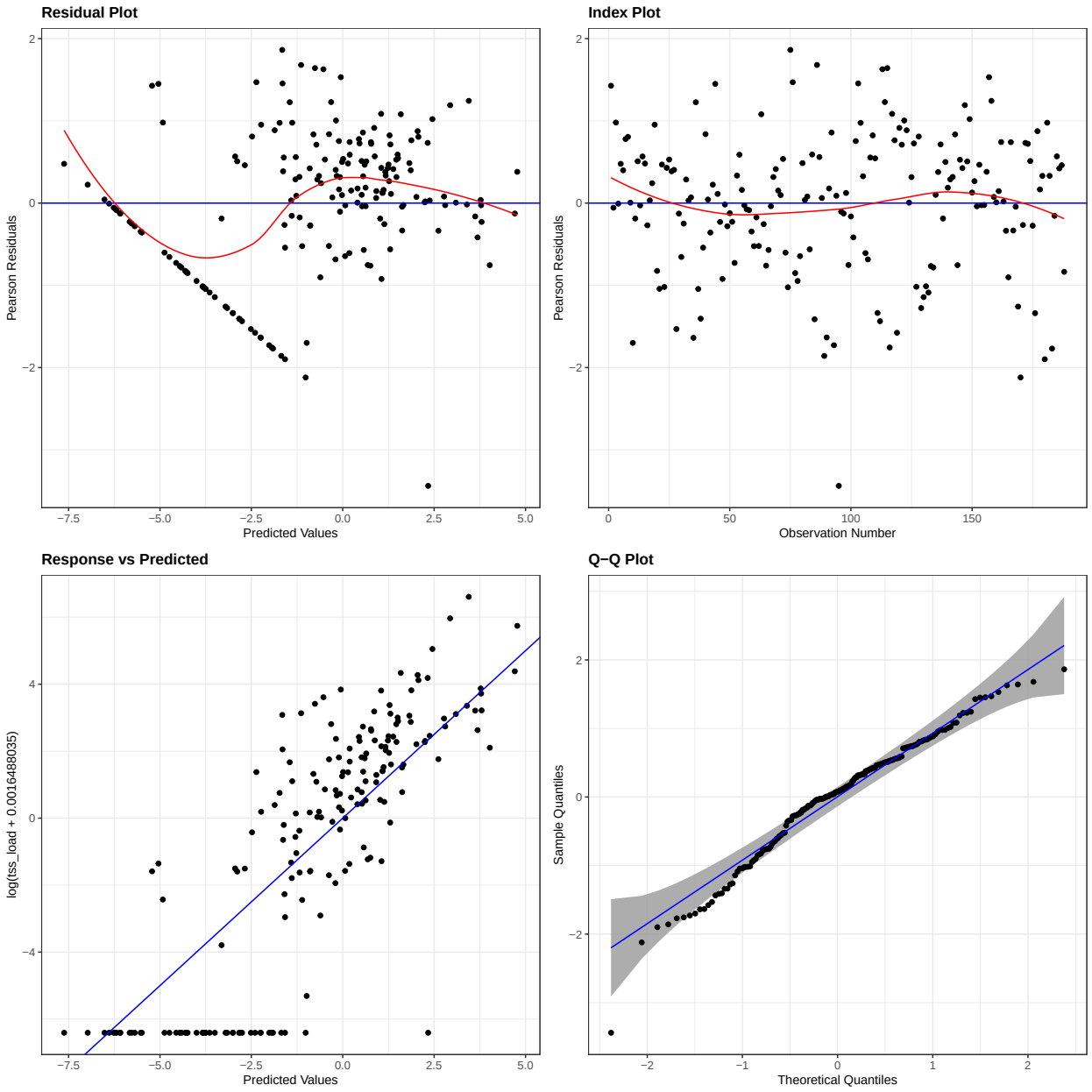
```
resid_xpanel(m_flume)
```



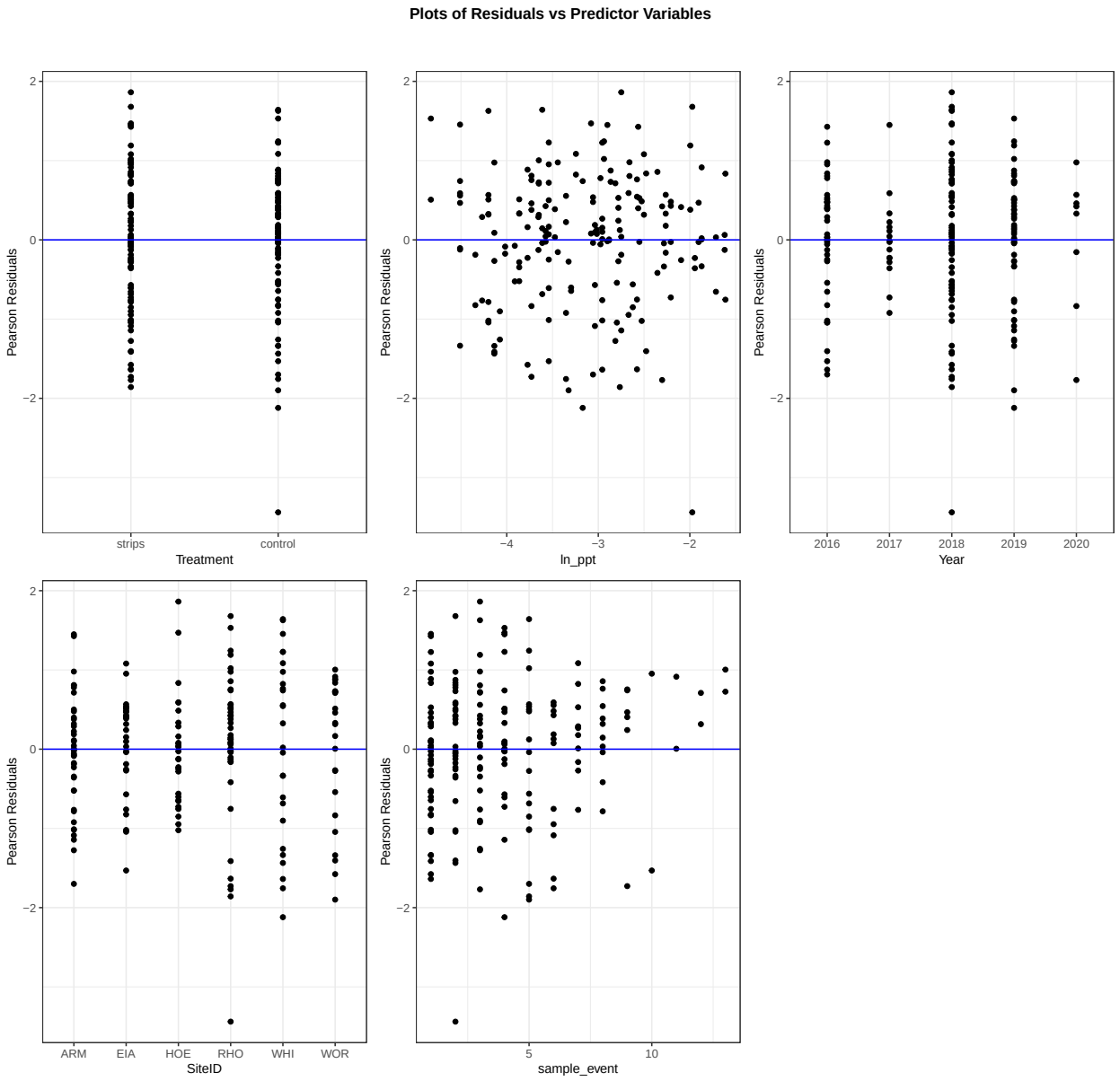
Selected model design

```
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
Pearson Residuals
Treatment
strips
control
ln_ppt
Year
2016
2017
2018
2019
2020
SiteID
ARM
EIA
HOE
RHO
WHI
WOR
sample_event
5
10