Flume: Random Analysis

Jessica Nelson

2022-03-24

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
```

```
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()
```

```
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
```

```
library("stringr")

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 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] data.table_1.14.2  ggResidpanel_0.3.0 emmeans_1.7.2      forcats_0.5.1      stringr_1.4.0      c
##  [7] purrr_0.3.4        readr_2.1.2        tidyr_1.2.0        tibble_3.1.6       ggplot2_3.3.5      t
## [13] lmerTest_3.1-3     lme4_1.1-28        Matrix_1.4-1
##
## 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     lattice_0.20-45     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        haven_2.4.3         xtable_1.8-4        mvtnorm_1.1-3
## [26] scales_1.1.1        tzdb_0.2.0          generics_0.1.2      ellipsis_0.3.2      withr_2.5.0
## [31] lazyeval_0.2.2      cli_3.2.0           magrittr_2.0.1      crayon_1.5.0        readxl_1.3.1
## [36] estimability_1.3    evaluate_0.15       fs_1.5.2            fansi_1.0.2         nlme_3.1-155
## [41] MASS_7.3-55         xml2_1.3.3          tools_4.1.3         hms_1.1.1           lifecycle_1.0.1
## [46] plotly_4.10.0       munsell_0.5.0       reprex_2.0.1        qqplotr_0.0.5       compiler_4.1.3
## [51] rlang_1.0.2         grid_4.1.3          nloptr_2.0.0        rstudioapi_0.13     htmlwidgets_1.5
## [56] rmarkdown_2.13      boot_1.3-28         gtable_0.3.0        DBI_1.1.2           R6_2.5.1
## [61] lubridate_1.8.0     knitr_1.37          fastmap_1.1.0       utf8_1.2.2          stringi_1.7.6
## [66] Rcpp_1.0.8.3        vctrs_0.3.8         DEoptimR_1.0-10     dbplyr_2.1.1        tidyselect_1.1.1
## [71] xfun_0.30
```

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+0.0053),
         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',
  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)))))))))))
```

```
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+0.0053),
         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)
```

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

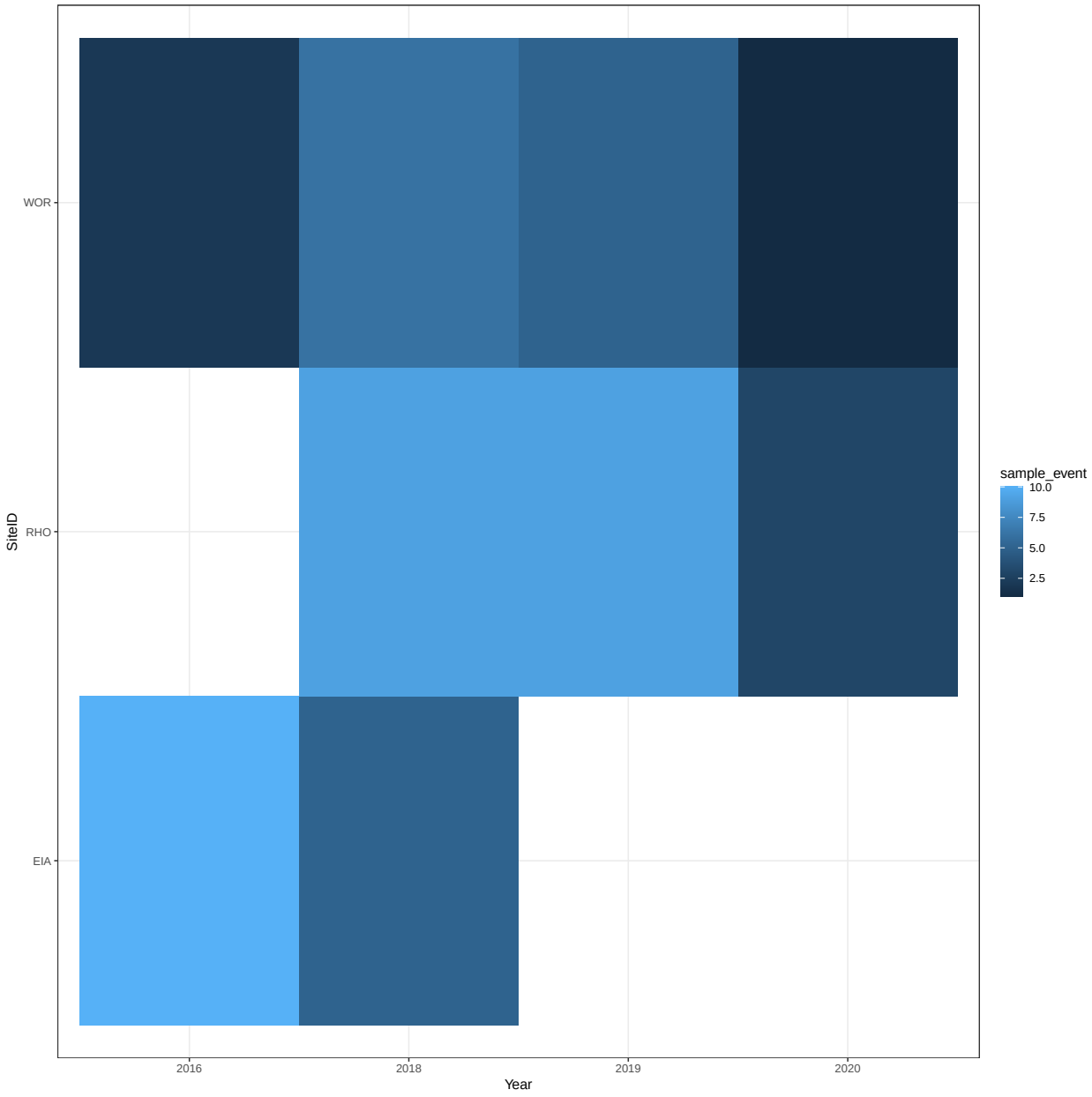
```
flumeR <- full_df %>%
  #filter(!is.na(ro_event)) %>%
    subset(random == 'R')
```

Exploratory analysis

Site-year with rainfall event

```
site_year_rfeventR <- flumeR %>%
  select(SiteID, Year, sample_event) %>%
  unique()

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



Number of samples

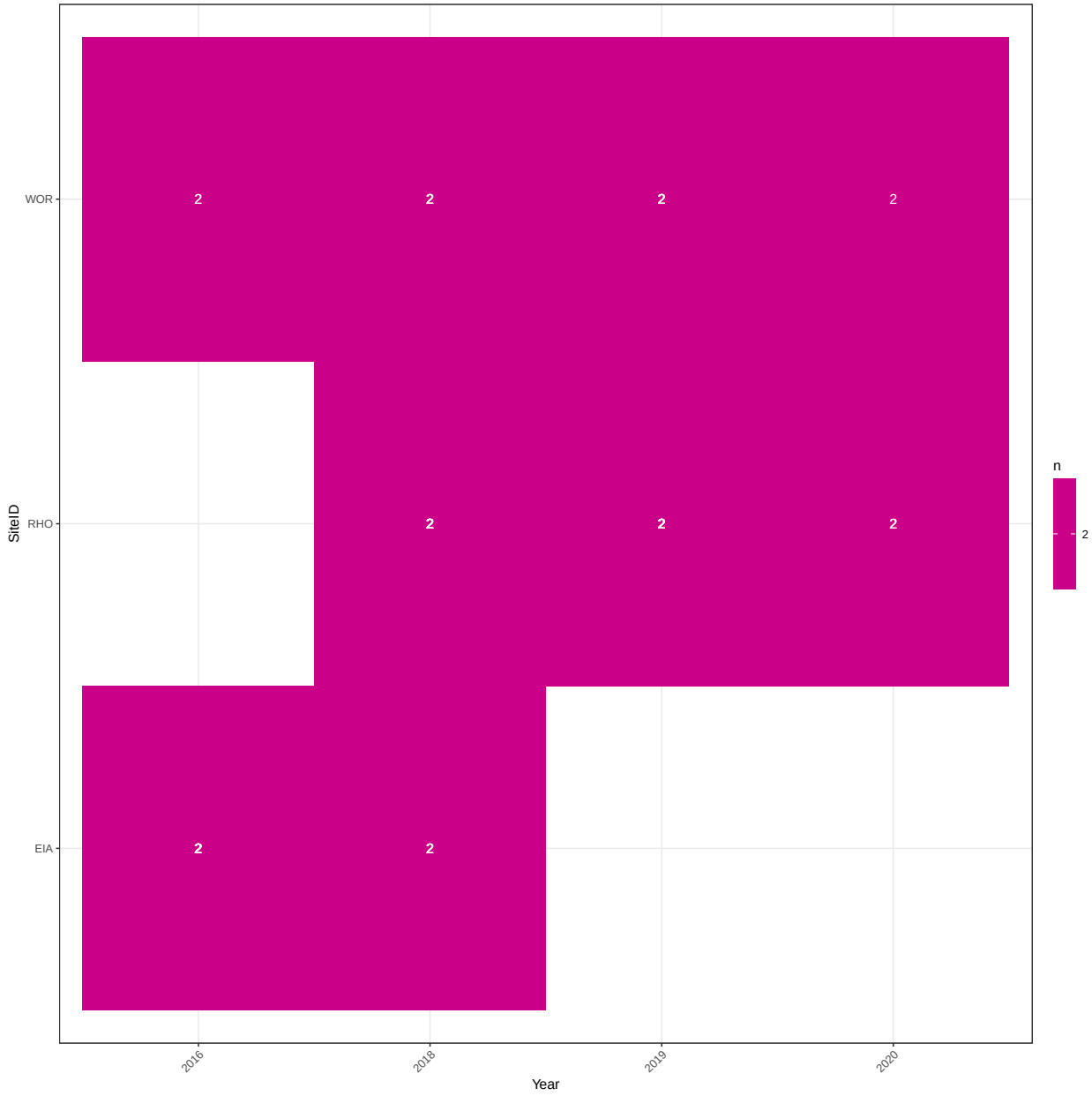
Calculate the number of observations for each treatment-position-year-site-time combination.

```
TSS_countsR <- flumeR %>%
  group_by(Year, SiteID, sample_event) %>%
  distinct() %>%
  summarize(n = n(), .groups = "drop")
```

Plot the number of observations for each combination.

```
g <- ggplot(TSS_countsR, 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
```

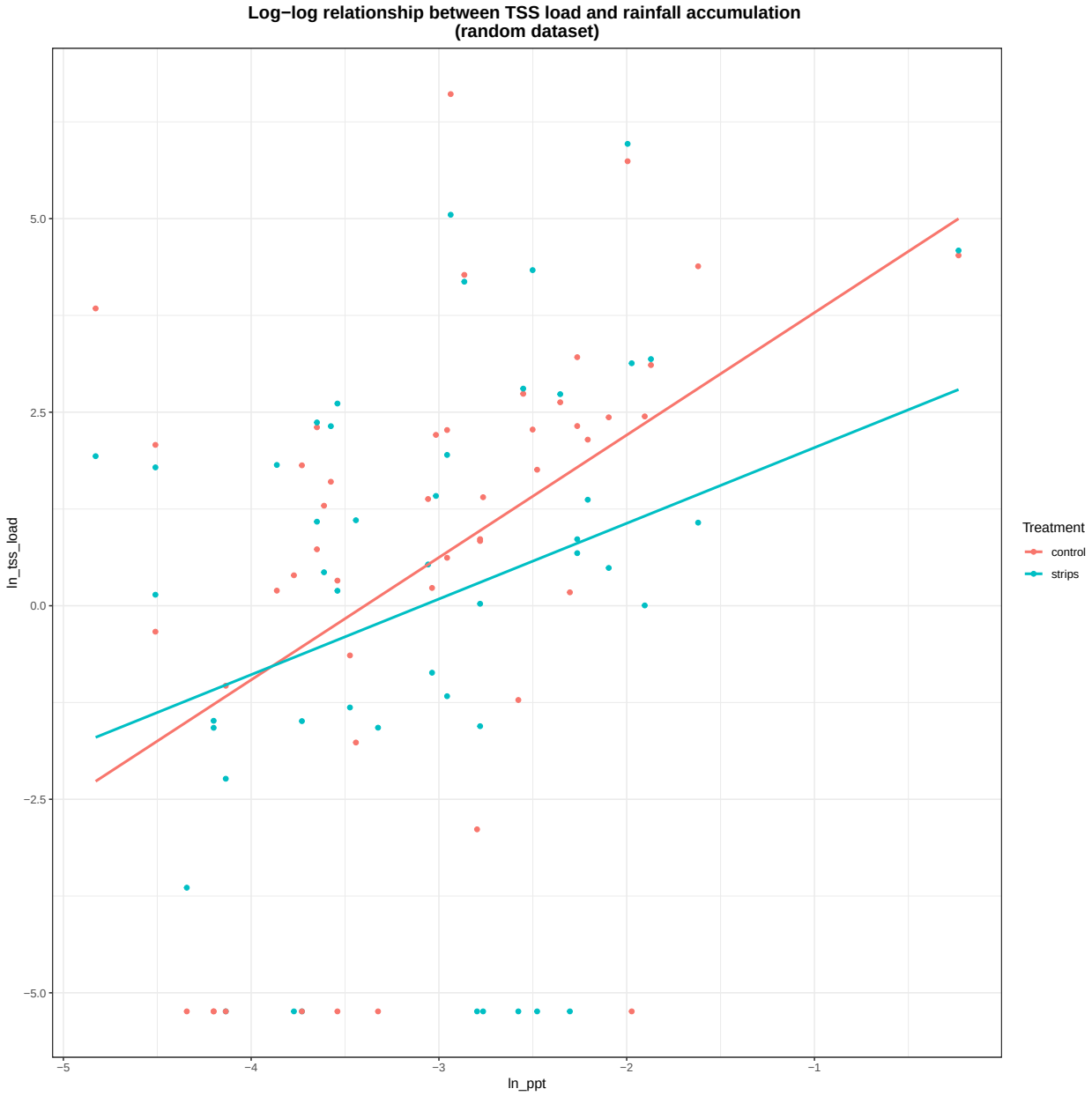



```
##ggsave("fig/soilpad_counts_no_diversion.png", g, width = 12, height = 12)
```

Data visualization

```
hR <- ggplot(flumeR, aes(x=ln_ppt, y=ln_tss_load, color=Treatment)) +
  geom_point() +
  geom_smooth(method=lm, se=FALSE, fullrange=TRUE) +
  ggtitle("Log-log relationship between TSS load and rainfall accumulation \n(random dataset)") +
  theme(plot.title = element_text(size=14, face="bold",hjust = 0.5))
hR
```

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



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

Main Analyses

There are three main analyses of interest:

- confirmatory, design-based analysis
- exploratory, covariate analysis
- relationship of sediment flow to sediment loss

```
#mR_flume <- lmerTest::lmer(log(tss_load+0.005322915) ~
#                             #(1 | SiteID) +
#                             (1 | SiteID:Treatment) +
#                             Treatment*ln_ppt +
#                            Year,
#                           data = flumeR)

mR_flume <- lmerTest::lmer(log(tss_load+0.0053) ~
                             (1 | SiteID) +
                             (1 | SiteID:Treatment) + #removed due to singular fit
                           #(1|SiteID:Treatment:sample_event) + #removed due to singular fit

                             Treatment*ln_ppt +
                            Treatment*crop +

                            Year*Treatment,
                           data = flumeR)

summary(mR_flume)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method ['lmerModLmerTest']
## Formula: log(tss_load + 0.0053) ~ (1 | SiteID) + (1 | SiteID:Treatment) +
##     Treatment * ln_ppt + Treatment * crop + Year * Treatment
##    Data: flumeR
##
## REML criterion at convergence: 423.6
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.2099 -0.5126  0.1453  0.6884  1.8065
##
## Random effects:
##  Groups           Name        Variance Std.Dev.
##  SiteID:Treatment (Intercept) 0.03228  0.1797
##  SiteID           (Intercept) 0.09050  0.3008
##  Residual                     6.34844  2.5196
## Number of obs: 96, groups:  SiteID:Treatment, 6; SiteID, 3
##
## Fixed effects:
##                             Estimate Std. Error      df t value Pr(>|t|)
## (Intercept)                   4.2895     1.7016 77.3718   2.521 0.013763 *
```

```
## Treatmentstrips              -3.1942     2.3920 71.0758  -1.335 0.186016
## ln_ppt                        1.7182     0.4265 82.0947   4.029 0.000125 ***
## cropsoybean                  -0.4542     1.0950  8.1133  -0.415 0.689064
## Year2018                      1.8146     1.1173 83.7093   1.624 0.108113
## Year2019                      3.1661     1.3332 83.6471   2.375 0.019841 *
## Year2020                     -0.3167     1.5929 80.9147  -0.199 0.842906
## Treatmentstrips:ln_ppt       -0.2696     0.6031 82.0590  -0.447 0.655997
## Treatmentstrips:cropsoybean   2.6364     1.4806  4.8660   1.781 0.136699
## Treatmentstrips:Year2018      0.3234     1.5779 82.7664   0.205 0.838099
## Treatmentstrips:Year2019      2.6979     1.8829 82.8362   1.433 0.155675
## Treatmentstrips:Year2020      1.6559     2.2419 73.0920   0.739 0.462511
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) Trtmnt ln_ppt crpsyb Yr2018 Yr2019 Yr2020 Trtm:_ Trtmn: T:Y2018 T:Y2019
## Trtmntstrps -0.703
## ln_ppt       0.736 -0.524
## cropsoybean -0.399  0.276  0.108
## Year2018    -0.618  0.440 -0.089  0.454
## Year2019    -0.452  0.322  0.149  0.589  0.697
## Year2020    -0.403  0.288 -0.020  0.307  0.518  0.505
## Trtmntstr:_ -0.521  0.741 -0.707 -0.074  0.063 -0.106  0.014
## Trtmntstrp:  0.287 -0.408 -0.077 -0.676 -0.342 -0.442 -0.244  0.109
## Trtmn:Y2018  0.438 -0.623  0.063 -0.327 -0.706 -0.492 -0.364 -0.089  0.484
## Trtmn:Y2019  0.320 -0.456 -0.106 -0.424 -0.492 -0.706 -0.354  0.150  0.626  0.696
## Trtmn:Y2020  0.288 -0.410  0.014 -0.235 -0.365 -0.355 -0.704 -0.019  0.347  0.517   0.503
```

```
mR_flume_step <- step(mR_flume, reduce.random = FALSE, alpha.fixed = 0.1)
```

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

```
mR_flume_model <- get_model(mR_flume_step)
summary(mR_flume_model)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method ['lmerModLmerTest']
## Formula: log(tss_load + 0.0053) ~ (1 | SiteID) + (1 | SiteID:Treatment) +      ln_ppt + Year
##    Data: flumeR
##
## REML criterion at convergence: 444.6
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.8406 -0.4395  0.2485  0.6888  1.5735
##
## Random effects:
##  Groups           Name        Variance Std.Dev.
##  SiteID:Treatment (Intercept) 0.06663  0.2581
##  SiteID           (Intercept) 0.17565  0.4191
##  Residual                     6.36232  2.5224
## Number of obs: 96, groups:  SiteID:Treatment, 6; SiteID, 3
##
## Fixed effects:
```

```
##             Estimate Std. Error      df t value Pr(>|t|)
## (Intercept)   3.1443     1.1365 42.3085   2.767 0.008362 **
## ln_ppt        1.5515     0.3002 87.1442   5.168 1.49e-06 ***
## Year2018      1.6770     0.7347 35.0354   2.283 0.028634 *
## Year2019      4.0534     0.8216 14.2027   4.934 0.000211 ***
## Year2020      0.2703     1.1186 35.5078   0.242 0.810428
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##          (Intr) ln_ppt Yr2018 Yr2019
## ln_ppt    0.827
## Year2018 -0.535 -0.142
## Year2019 -0.313  0.109  0.636
## Year2020 -0.336 -0.047  0.491  0.479
```

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

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

```
## Year = 2016:
##  contrast         ratio    SE  df asymp.LCL asymp.UCL
##  control / strips 2.860 3.375 Inf    0.2831     28.89
##
## Year = 2018:
##  contrast         ratio    SE  df asymp.LCL asymp.UCL
##  control / strips 2.070 1.883 Inf    0.3480     12.31
##
## Year = 2019:
##  contrast         ratio    SE  df asymp.LCL asymp.UCL
##  control / strips 0.193 0.239 Inf    0.0170      2.19
##
## Year = 2020:
##  contrast         ratio    SE  df asymp.LCL asymp.UCL
##  control / strips 0.546 0.997 Inf    0.0152     19.58
##
## 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
```

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

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

```
confint(trtR)
```

```
## $emmeans
##  Treatment response    SE  df asymp.LCL asymp.UCL
##  control      0.965 0.493 Inf     0.354      2.62
##  strips       1.087 0.555 Inf     0.399      2.95
##
## 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 0.888 0.597 Inf     0.238      3.32
##
## 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
```

```
yearR   = emmeans(mR_flume, ~ Year,
                  type = "response",
                  lmer.df = "asymptotic")
```

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

```
confint(yearR)
```

```
##  Year response     SE  df asymp.LCL asymp.UCL
##  2016    0.174  0.111 Inf    0.0478     0.598
##  2018    1.285  0.632 Inf    0.4886     3.367
##  2019   16.339 10.558 Inf    4.6025    57.967
##  2020    0.293  0.278 Inf    0.0425     1.854
##
## 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_pptR  = emmeans(mR_flume, pairwise ~ Treatment|ln_ppt,
                  at=list(ln_ppt=c(-4,-3,-2,-1,-0.25)),
                  type = "response",
                  lmer.df = "asymptotic")

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

```
## ln_ppt = -4:
##  contrast         ratio    SE  df asymp.LCL asymp.UCL
##  control / strips 0.689 0.592 Inf    0.1282      3.71
##
```

```
## ln_ppt = -3:
##  contrast         ratio    SE  df asymp.LCL asymp.UCL
##  control / strips 0.903 0.609 Inf    0.2406      3.39
##
## ln_ppt = -2:
##  contrast         ratio    SE  df asymp.LCL asymp.UCL
##  control / strips 1.182 1.122 Inf    0.1840      7.60
##
## ln_ppt = -1:
##  contrast         ratio    SE  df asymp.LCL asymp.UCL
##  control / strips 1.548 2.229 Inf    0.0920     26.04
##
## ln_ppt = -0.25:
##  contrast         ratio    SE  df asymp.LCL asymp.UCL
##  control / strips 1.895 3.510 Inf    0.0502     71.50
##
## 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
```

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

confint(cropR)$contrasts
```

```
## crop = corn:
##  contrast         ratio    SE  df asymp.LCL asymp.UCL
##  control / strips 3.319 2.653 Inf    0.6930     15.90
##
## crop = soybean:
##  contrast         ratio    SE  df asymp.LCL asymp.UCL
##  control / strips 0.238 0.277 Inf    0.0242      2.34
##
## 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

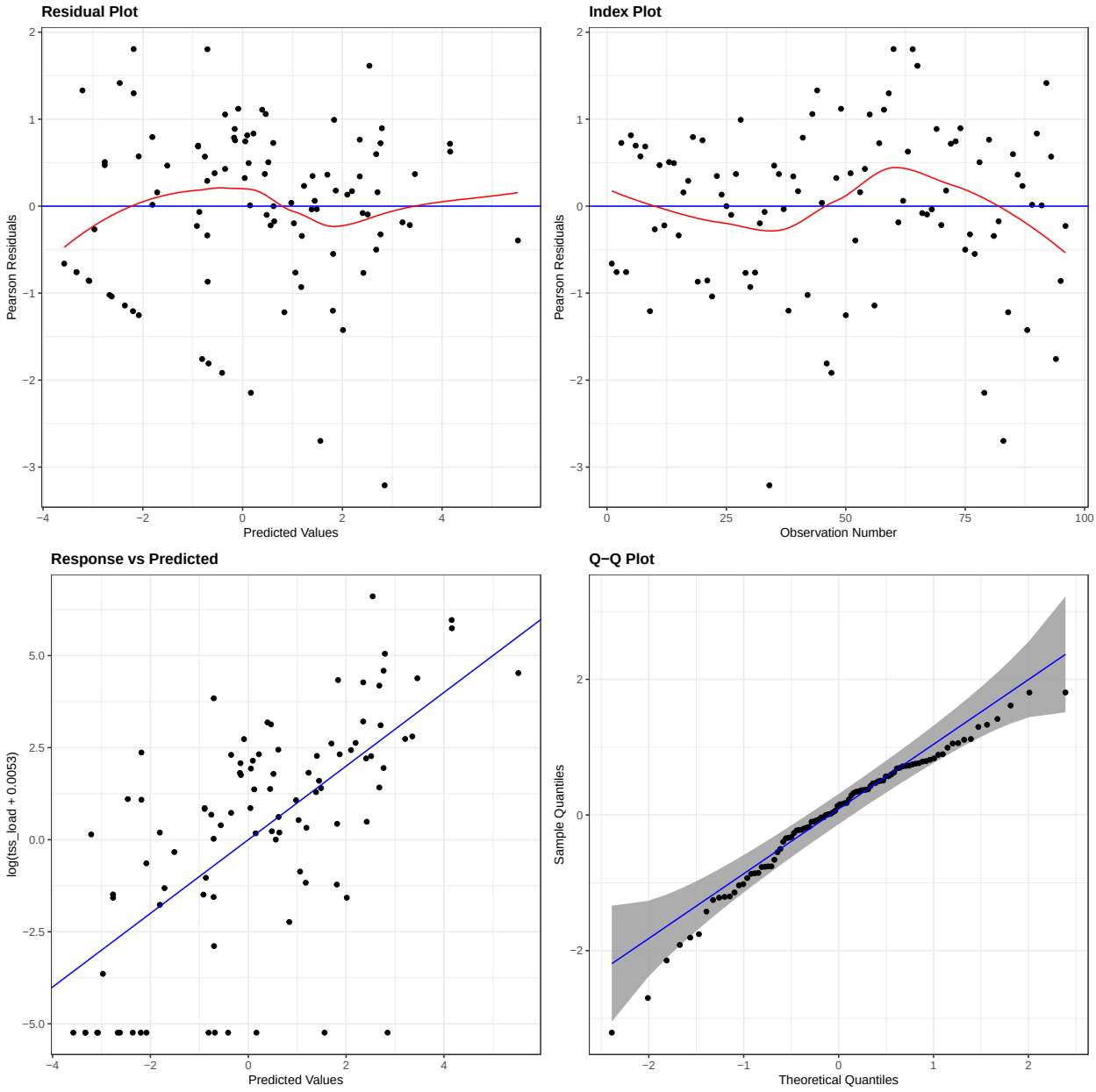
There are two possible models:

- mR_flume: full model design, design-based analysis
- mR_flume_model: model design selected based on backward step selection

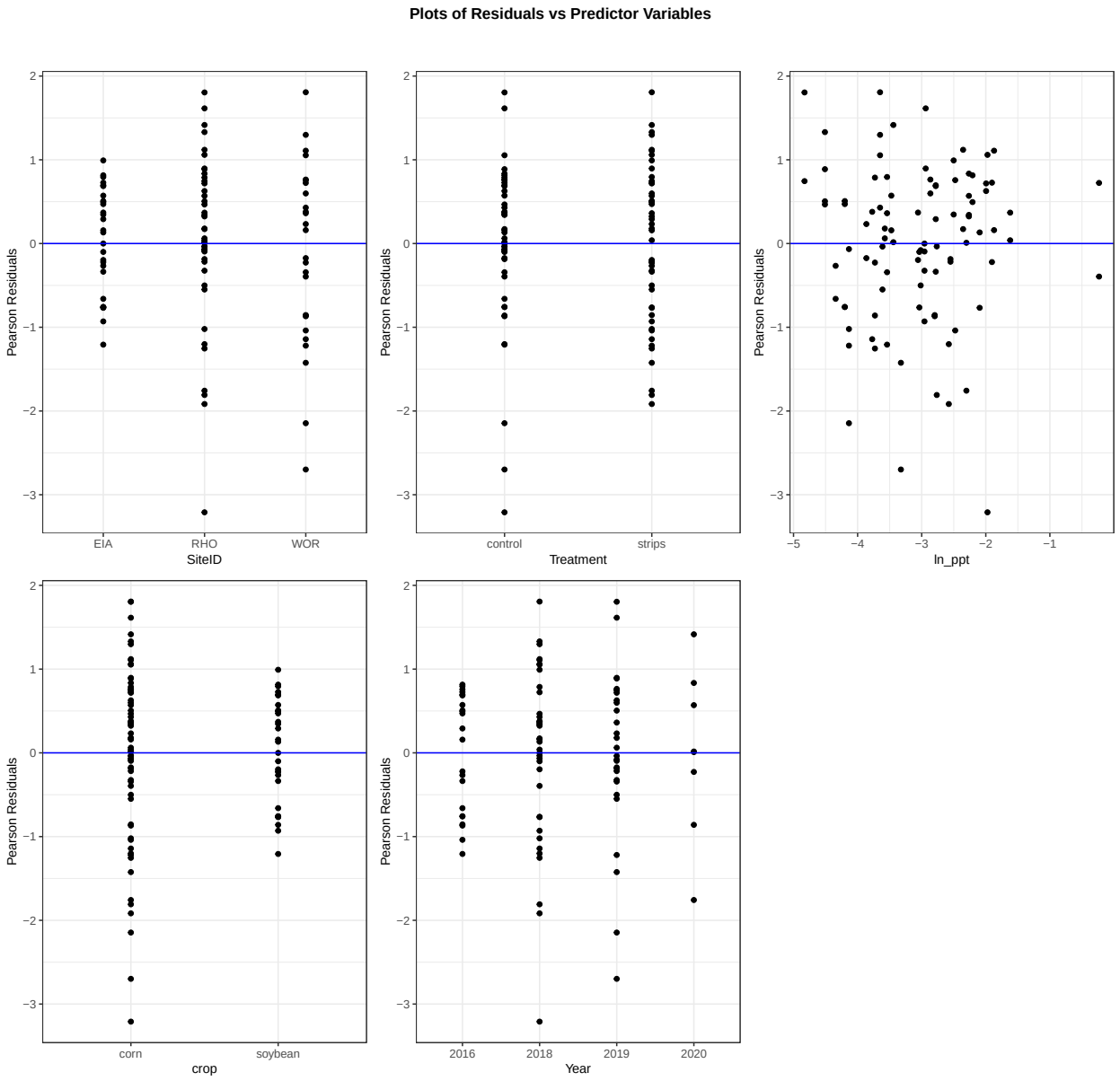
Full model design

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

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

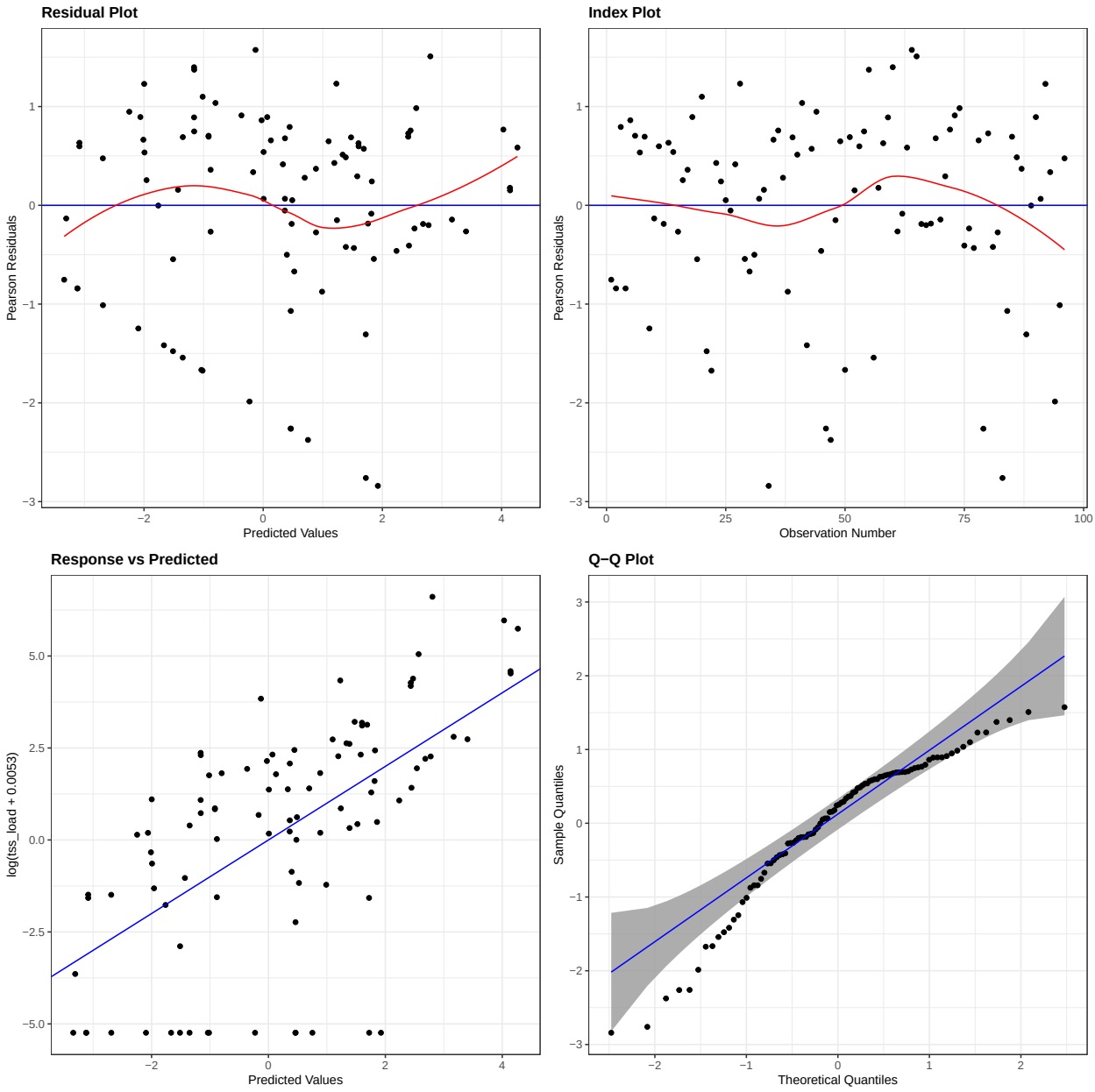



```
resid_xpanel(mR_flume)
```




```
resid_panel(mR_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(mR_flume_model)
```

Plots of Residuals vs Predictor Variables