Flume: Random Analysis (March-November)

(adapted from Jarad Niemi - Soilpad Analysis)

Jessica Nelson

2022-04-27

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, scipen = 999)

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] data.table_1.14.2  ggResidpanel_0.3.0 emmeans_1.7.2      forcats_0.5.1      stringr_1.4.0
##  [7] purrr_0.3.4        readr_2.1.2        tidyr_1.2.0        tibble_3.1.6       ggplot2_3.3.5
## [13] 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     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_2.0.2      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_1.38          fastmap_1.1.0       utf8_1.2.2          stringi_1.7.6
## [66] Rcpp_1.0.8          vctrs_0.3.8         DEoptimR_1.0-10     dbplyr_2.1.1        tidyselect_1.1.
## [71] xfun_0.30
```

Read in data

```r
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: 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.
```

```r
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.
```

```r
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.
```

```r
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.
```

```r
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’
```

```r
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")
```

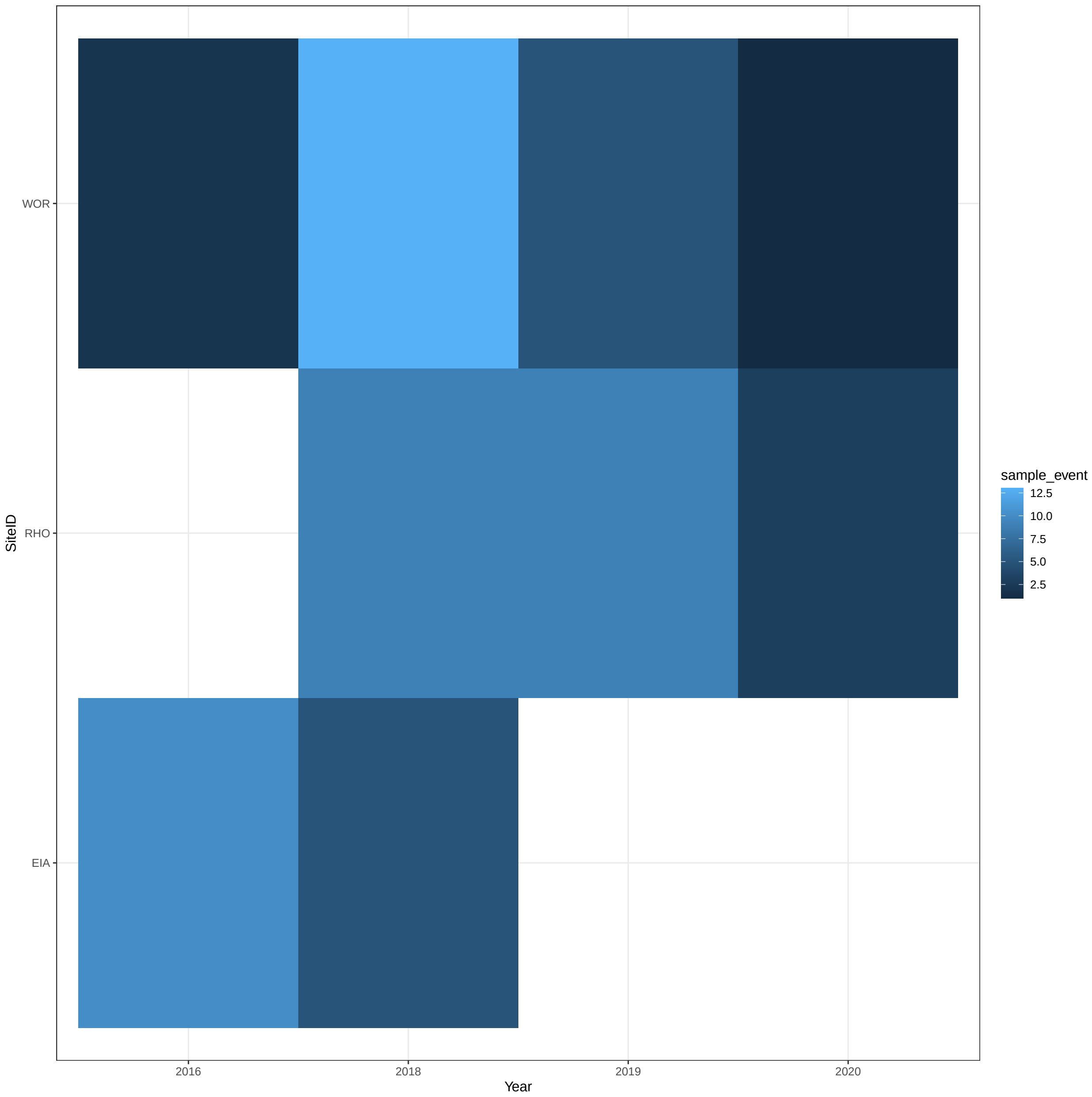
```
flumeR <- full_df %>%
  #filter(!is.na(ro_event)) %>%
    subset(random == 'R')
```

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.

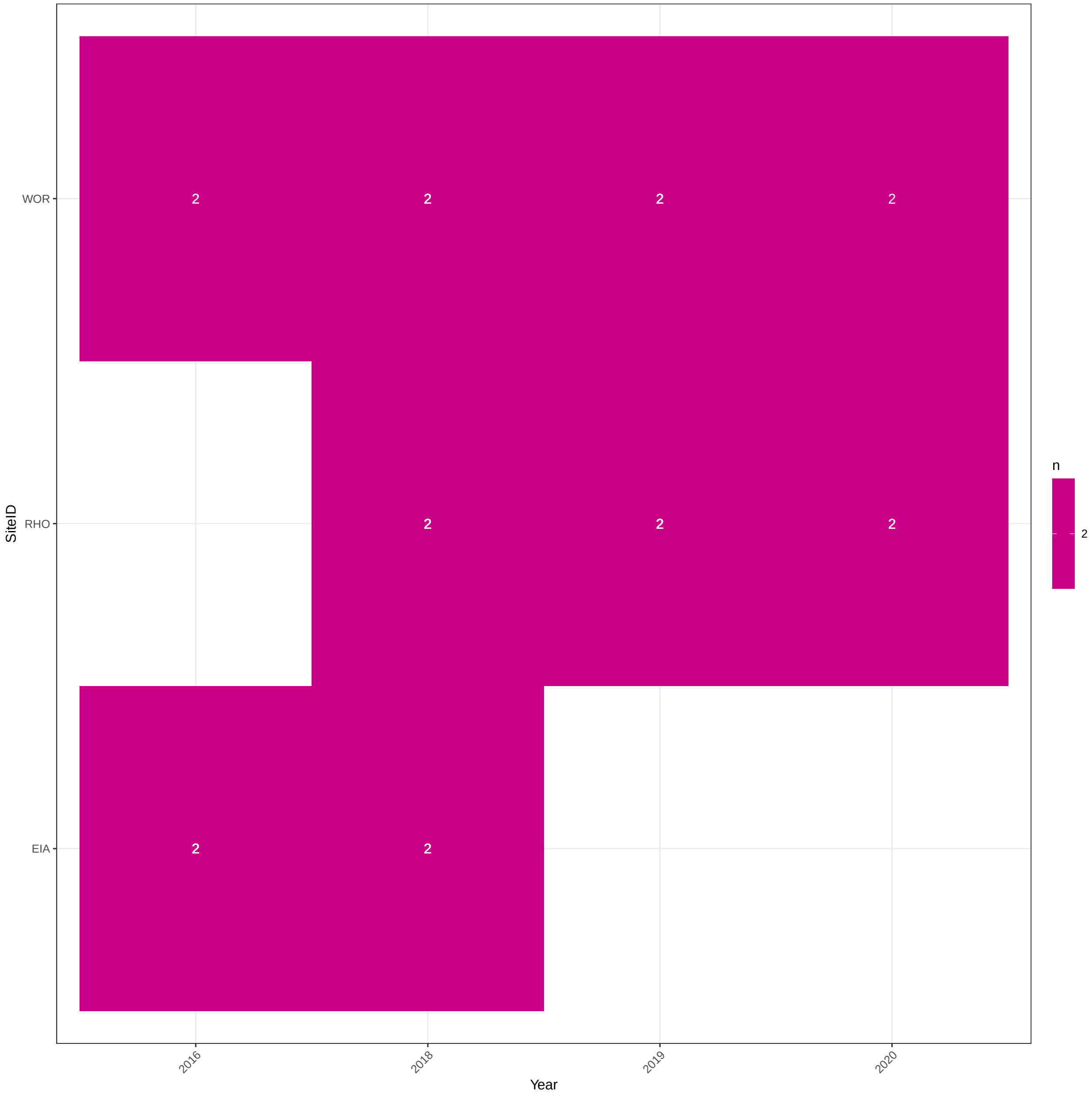
```
TSS_countsR <- flumeR %>%
  group_by(Year, SiteID, sample_event) %>%
  distinct() %>%
  summarize(n = n(), .groups = "drop")
```

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

Plot the number of observations for each combination.

```r
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
```

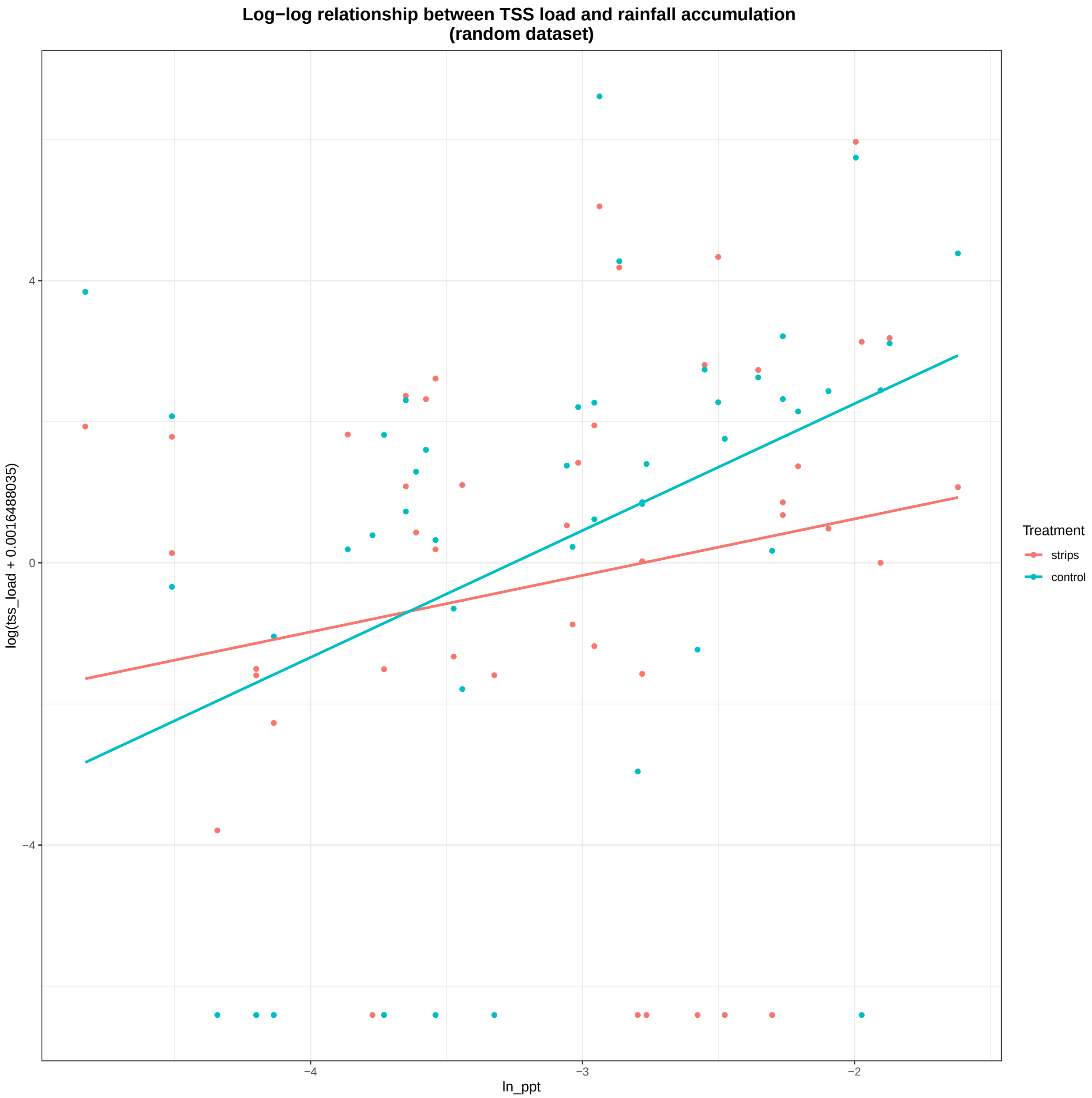


```r
##ggsave("fig/soilpad_counts_no_diversion.png", g, width = 12, height = 12)
```

Data visualization

```
hR <- ggplot(flumeR, aes(x=ln_ppt, y=log(tss_load+0.0016488035), 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+0.0016488035) ~
                             Treatment*ln_ppt +
                            Year*Treatment +

                            (1 | SiteID) +
                             (1 | SiteID:Treatment) + #removed due to singular fit
                            #(1|Year:sample_event) + #removed due to singular fit
                            (1|SiteID:Year:sample_event),

                          data = flumeR)
```

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

```
summary(mR_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) + (1 | SiteID:Treatment) + (1 | SiteID:Year:sample_event)
##    Data: flumeR
##
## REML criterion at convergence: 442.9
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.3330 -0.2755  0.0976  0.5767  1.6519
##
## Random effects:
##  Groups                   Name        Variance      Std.Dev.
##  SiteID:Year:sample_event (Intercept) 1.68471560340 1.2979659
##  SiteID:Treatment         (Intercept) 0.92000686186 0.9591699
##  SiteID                   (Intercept) 0.00000003641 0.0001908
##  Residual                             6.61355397368 2.5716831
## Number of obs: 94, groups:  SiteID:Year:sample_event, 47; SiteID:Treatment, 6; SiteID, 3
##
## Fixed effects:
##                           Estimate Std. Error      df t value Pr(>|t|)
## (Intercept)                 1.3692     2.0321 57.6362   0.674  0.50316
## Treatmentcontrol            3.0598     2.5967 34.9507   1.178  0.24661
## ln_ppt                      1.2678     0.5421 77.7591   2.339  0.02191 *
## Year2018                    1.6304     1.2160 62.4417   1.341  0.18483
## Year2019                    5.4784     1.3792 36.6739   3.972  0.00032 ***
## Year2020                    1.3201     1.8467 62.6079   0.715  0.47736
## Treatmentcontrol:ln_ppt     0.6843     0.6847 40.0067   0.999  0.32361
## Treatmentcontrol:Year2018   0.4755     1.5517 39.3406   0.306  0.76088
## Treatmentcontrol:Year2019  -2.0192     1.7731 29.2098  -1.139  0.26404
## Treatmentcontrol:Year2020  -1.7740     2.3564 39.3880  -0.753  0.45602
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Correlation of Fixed Effects:
##             (Intr) Trtmnt ln_ppt Yr2018 Yr2019 Yr2020 Trtm:_ T:Y2018 T:Y2019
## Trtmntcntrl -0.639
## ln_ppt       0.837 -0.522
## Year2018    -0.463  0.295 -0.088
## Year2019    -0.281  0.183  0.116  0.656
## Year2020    -0.324  0.208 -0.047  0.509  0.506
## Trtmntcnt:_ -0.528  0.826 -0.632  0.055 -0.075  0.029
## Trtmn:Y2018  0.295 -0.462  0.054 -0.638 -0.427 -0.332 -0.085
## Trtmn:Y2019  0.182 -0.285 -0.073 -0.424 -0.643 -0.331  0.116  0.664
## Trtmn:Y2020  0.208 -0.326  0.029 -0.332 -0.333 -0.638 -0.045  0.520   0.519
## optimizer (nloptwrap) convergence code: 0 (OK)
## boundary (singular) fit: see help(’isSingular’)
```

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

```
## boundary (singular) fit: see help(’isSingular’)
## 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 + 0.0016488035) ~ ln_ppt + Year + (1 | SiteID) +
##     (1 | SiteID:Treatment) + (1 | SiteID:Year:sample_event)
##    Data: flumeR
##
## REML criterion at convergence: 459
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.7727 -0.2975  0.2827  0.6125  1.2923
##
## Random effects:
##  Groups                   Name        Variance Std.Dev.
##  SiteID:Year:sample_event (Intercept) 1.5229   1.2341
##  SiteID:Treatment         (Intercept) 0.1891   0.4348
##  SiteID                   (Intercept) 0.1996   0.4468
##  Residual                             6.9877   2.6434
## Number of obs: 94, groups:  SiteID:Year:sample_event, 47; SiteID:Treatment, 6; SiteID, 3
##
## Fixed effects:
##             Estimate Std. Error      df t value Pr(>|t|)
## (Intercept)   2.9831     1.5417 31.9123   1.935 0.061913 .
## ln_ppt        1.6154     0.4210 40.5295   3.837 0.000427 ***
## Year2018      1.8045     0.9208 25.1042   1.960 0.061231 .
## Year2019      4.3648     1.0243 11.4463   4.261 0.001226 **
## Year2020      0.3368     1.3986 25.2126   0.241 0.811665
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##          (Intr) ln_ppt Yr2018 Yr2019
## ln_ppt    0.858
## Year2018 -0.463 -0.095
## Year2019 -0.269  0.117  0.633
## Year2020 -0.318 -0.053  0.483  0.472
```

```
##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
##  strips / control 0.397 0.580 Inf    0.0226      6.96
##
## Year = 2018:
##  contrast         ratio    SE  df asymp.LCL asymp.UCL
##  strips / control 0.247 0.291 Inf    0.0244      2.49
##
## Year = 2019:
##  contrast         ratio    SE  df asymp.LCL asymp.UCL
##  strips / control 2.988 3.984 Inf    0.2191     40.75
##
## Year = 2020:
##  contrast         ratio    SE  df asymp.LCL asymp.UCL
##  strips / control 2.338 4.766 Inf    0.0430    127.02
##
## 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
##  strips       0.617 0.463 Inf     0.141      2.68
##  control      0.679 0.510 Inf     0.155      2.95
##
## 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 0.909 0.918 Inf     0.126      6.58
##
## 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
```

```
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    0.118 0.100 Inf    0.0214     0.617
##  2018    0.772 0.516 Inf    0.2075     2.860
##  2019   10.420 7.935 Inf    2.3417    46.344
##  2020    0.183 0.224 Inf    0.0154     1.989
##
## 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_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
##  strips / control 1.660 1.950 Inf   0.16587     16.61
##
## ln_ppt = -3:
##  contrast         ratio    SE  df asymp.LCL asymp.UCL
##  strips / control 0.837 0.848 Inf   0.11491      6.10
##
## ln_ppt = -2:
```

```
##  contrast         ratio    SE  df asymp.LCL asymp.UCL
##  strips / control 0.422 0.536 Inf   0.03512      5.08
##
## ln_ppt = -1:
##  contrast         ratio    SE  df asymp.LCL asymp.UCL
##  strips / control 0.213 0.377 Inf   0.00664      6.84
##
## ln_ppt = -0.25:
##  contrast         ratio    SE  df asymp.LCL asymp.UCL
##  strips / control 0.128 0.282 Inf   0.00167      9.71
##
## 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
```

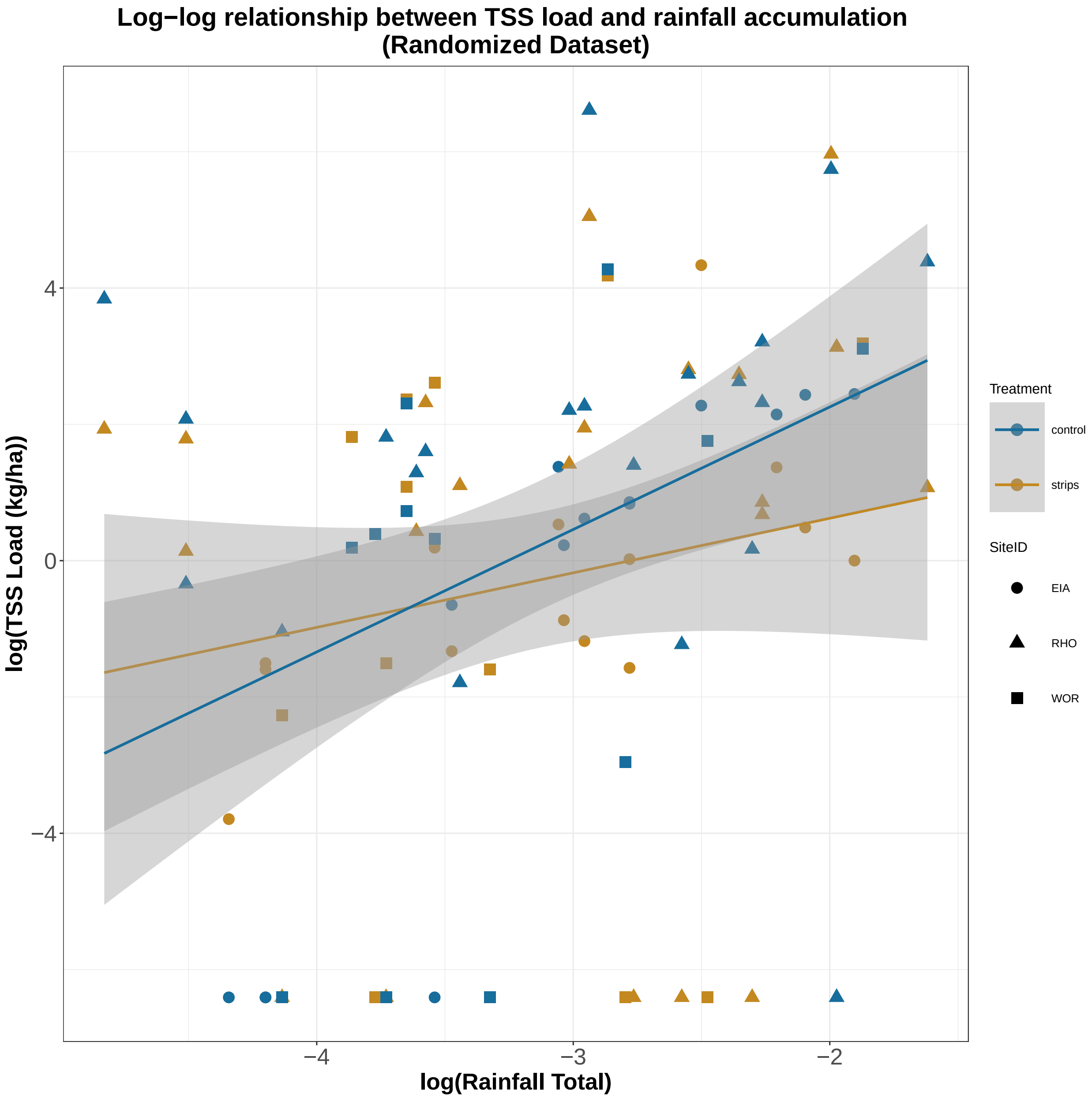
```
#cropR      = emmeans(mR_flume, pairwise ~ Treatment|crop,
#                    type = "response",
#                    lmer.df = "asymptotic")

#confint(cropR)
```

```
h <- ggplot(flumeR, aes(x=log(ppt_sum), y=log(tss_load+0.0016488035), color=Treatment), inherit.aes = FA
  scale_color_manual(values = c("control" = "#176D9C",
                                "strips" = "#C38820")) +
  geom_point(aes(shape=SiteID), size=4) +
  #scale_y_log10() +
  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(Randomized 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))

h
```

```
## ‘geom_smooth()‘ using formula ’y ~ x’
```



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

Check assumptions

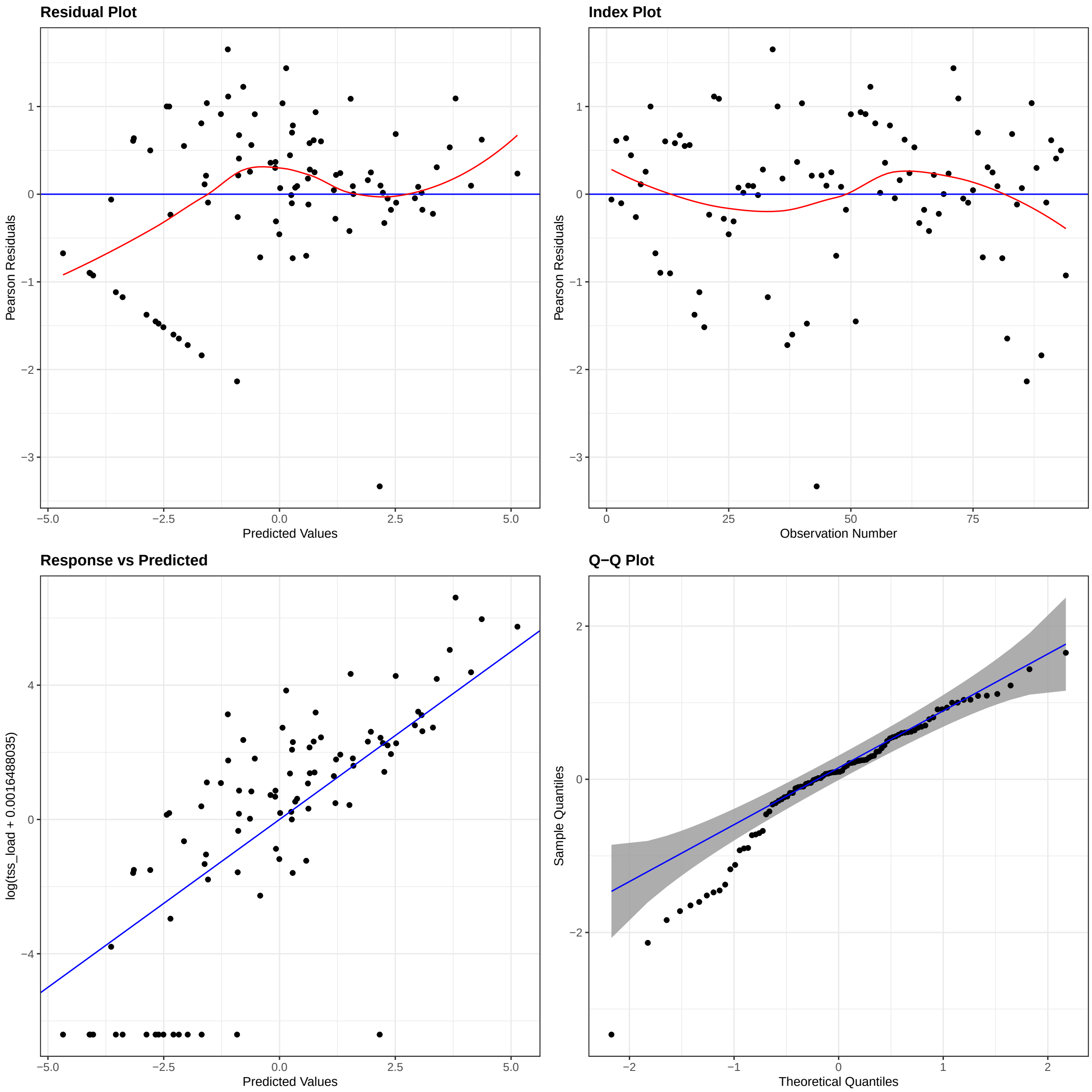
There are two possible models:

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

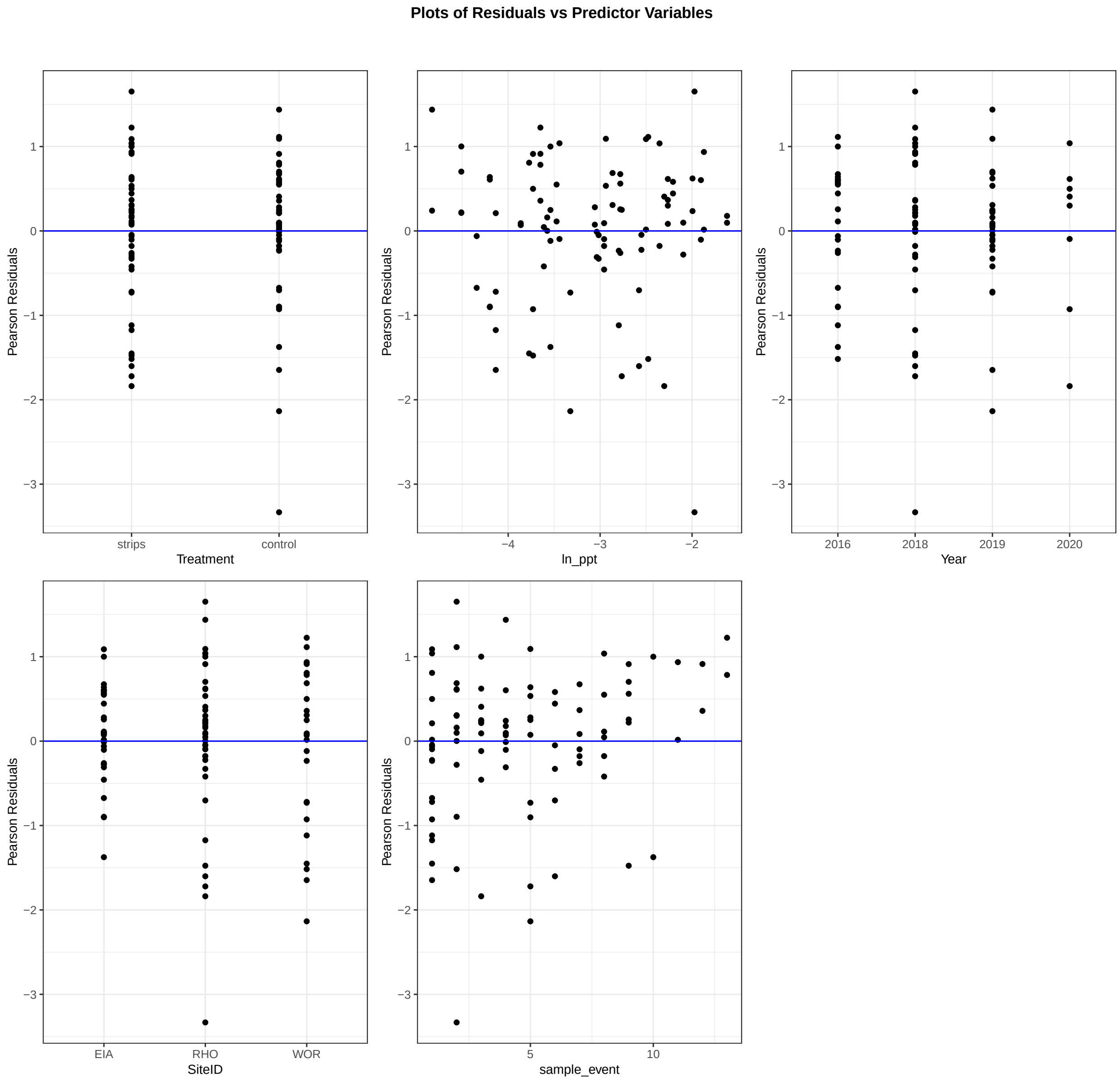
Full model design

```
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'
```

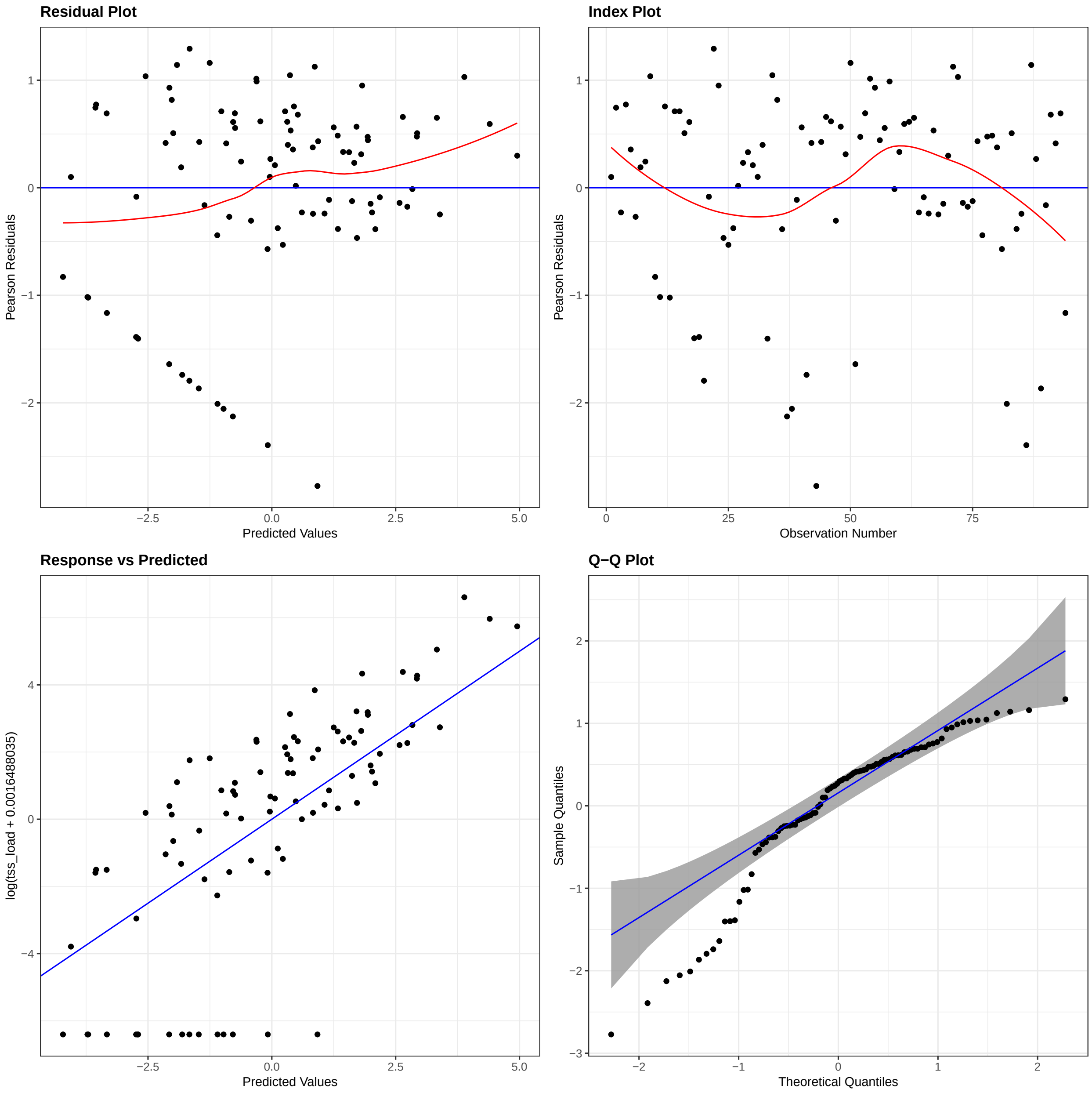


```
resid_xpanel(mR_flume)
```



```r
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'
```

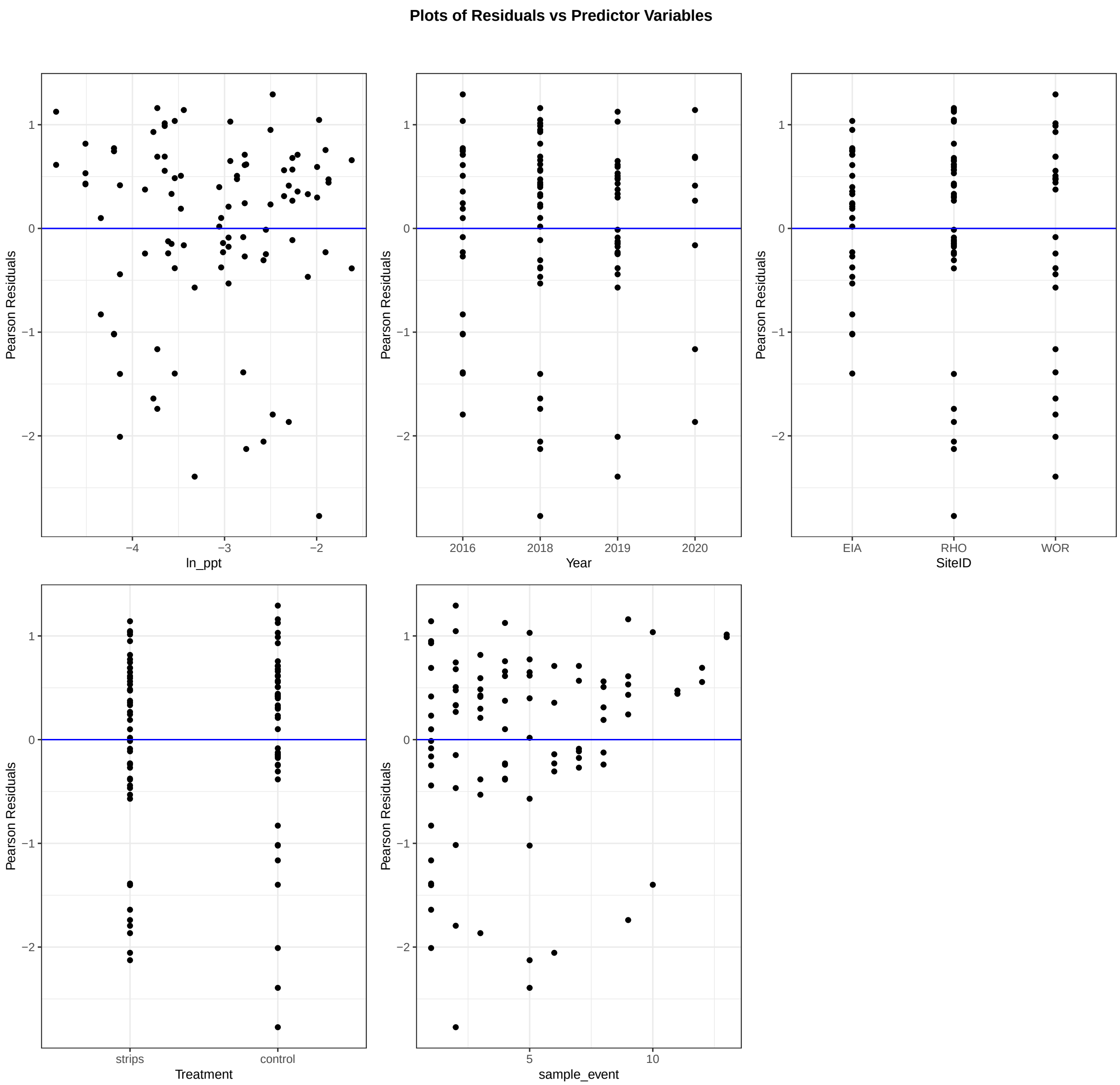

Residual Plot

Index Plot

Response vs Predicted

Q–Q Plot

```
resid_xpanel(mR_flume_model)
```



Plots of Residuals vs Predictor Variables

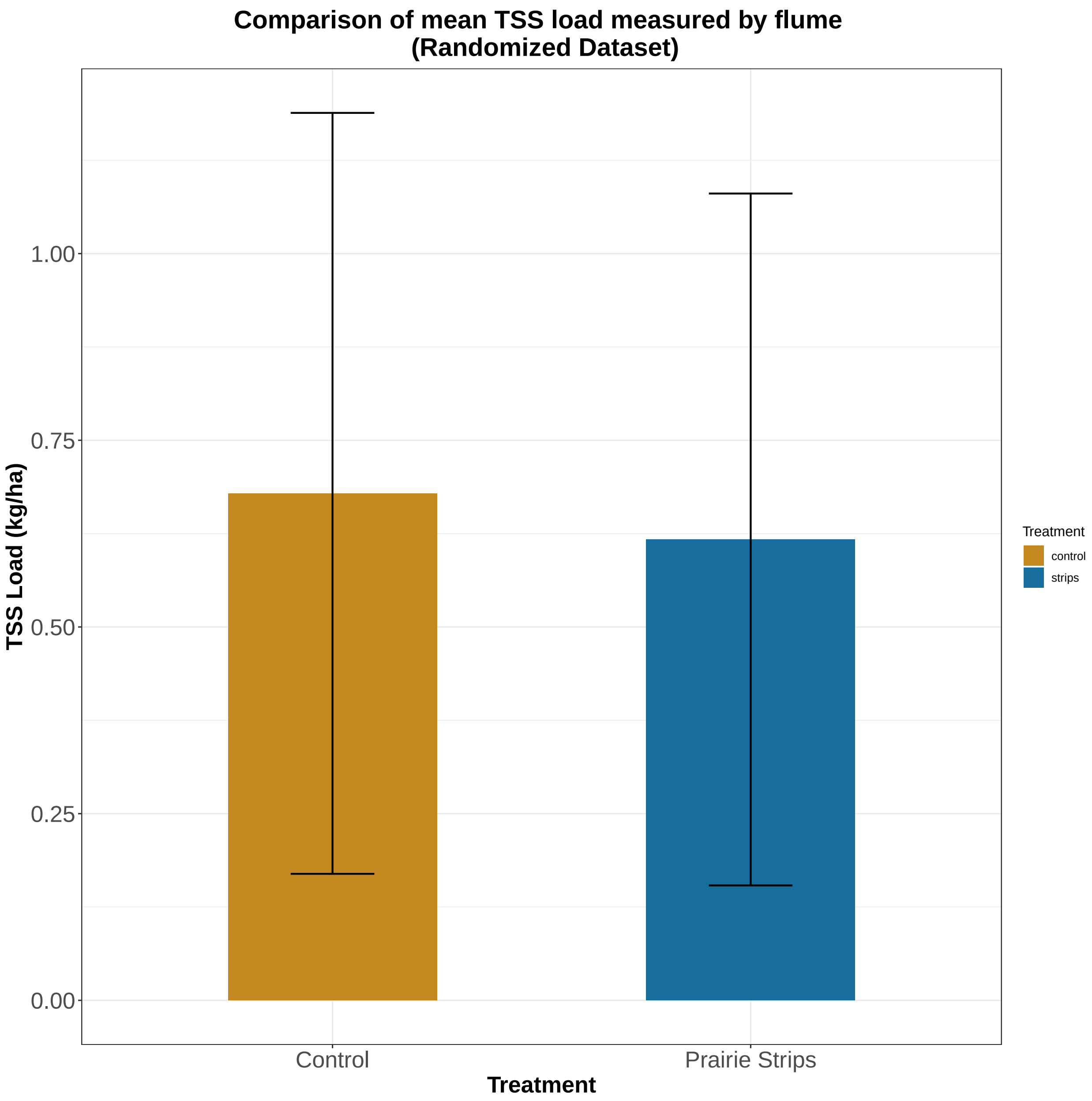
```
trtR <- as.data.frame(trtR)

jR <- trtR %>%
  filter(Treatment != ".")

trtR_plot <- jR %>%
  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('#C38820','#176D9C')) +
  #ggtitle("Comparison of mean TSS load measured by flume \n across sampling events (2016-2020)") +
  ggtitle("Comparison of mean TSS load measured by flume \n (Randomized Dataset)") +
  xlab("Treatment") +
  ylab("TSS Load (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 Strips"))
  #geom_text(aes(label=round(Score,2)), position=position_dodge(width=0.5),  vjust=-0.25) +
  #geom_text(aes(y = lb, label = lb), position=position_dodge(width=0.5), vjust=2)

trtR_plot
```

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


Comparison of mean TSS load measured by flume
(Randomized Dataset)
TSS Load (kg/ha)
1.00
0.75
0.50
0.25
0.00
Control
Prairie Strips
Treatment
Treatment
control
strips