Adaptive Randomization

November 15, 2018

Review of the Analysis of Covariance

Cochran and Cox [CC57] (Chapt 3) present the analysis of covariance using an experiment on the effects of fumigants on soil nematodes.

> The object of the experiment was to measure the effectiveness of 4 soil fumigants in keeping down the number of eelworms in soil.
>
> ...
>
> Each fumigant was tested both in a single and a double dose. These comprise 8 distinct treatments. The control (no fumigant) was the ninth treatment. Normally the replication would have consisted of 9 plots, but in this case 4 control plots were placed in each "replicate," which actually contained 12 plots.
>
> ...
>
> After harvest, a sample of 400 grams of soil was taken from each plot and the number of eelworm cysts counted for each sample. The data are thos labelled "second count" in table 3.1.
>
> ...
>
> In the eelworm experiment, where the plots are grouped into blocks of 12 plots each, any differences from block to block in the severity of eelworm infestation were eliminated in this way. On the other hean, differences in infestation *from plot to plot within the same block* are not controlled by the design and do contribute to experimental errors, since treatments were assigned at random to plots within each block. Accordingly, before the fumigantes were applied, samples were taken in order to estimate the natural infestation in each plot.

For simplicity, in this analysis we label each of the 4 control plots per block as distinct treatments; this allows us to model the experiment as an RCB of 12 treatments. We will later rerandomize the block and denoting each control as unique will avoid complications.

The data were entered into an ARM trial file and exported to R in the following format:

```
> cochran.arm.dat <- data.frame(
+   plot=as.factor(c(301, 603, 702, 1101, 304, 601, 902, 1203, 103, 604,
+                    904, 1103, 101, 404, 804, 1201, 204, 402, 903, 1102,
+                    202, 502, 704, 1003, 303, 401, 801, 1004, 203, 503,
+                    701, 1202, 104, 501, 803, 1104, 102, 403, 703, 1204,
+                    302, 504, 802, 1001, 201, 602, 901, 1002)),
+   treatment=as.factor(c(1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 4,
+                         5, 5, 5, 5, 6, 6, 6, 6, 7, 7, 7, 7, 8, 8, 8, 8,
+                         9, 9, 9, 9, 10, 10, 10, 10, 11, 11, 11, 11, 12, 12, 12, 12)),
+   replicate=as.factor(c(1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4,
+                         1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4,
```

```
+                         1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4)),
+   y=c(3, 3, 4, 5, 3, 3, 6, 6, 1, 3, 6, 5, 1, 1, 5, 6, 2, 1, 6, 5, 2, 2,
+       4, 4, 3, 1, 5, 4, 2, 2, 4, 6, 1, 2, 5, 5, 1, 1, 4, 6, 3, 2, 5, 4, 2, 3, 6, 4),
+   x=c(1, 7, 2, 5, 4, 5, 2, 7, 3, 8, 4, 7, 1, 8, 4, 5, 4, 6, 3, 6, 2, 6, 4, 7, 3, 5,
+       1, 8, 3, 7, 1, 6, 4, 5, 3, 8, 2, 7, 3, 8, 2, 8, 2, 5, 1, 6, 1, 6),
+   assessment2=c(268, 132, 256, 236, 408, 292, 280, 142, 415, 454, 386, 176,
+                 365, 298, 379, 199, 222, 114, 398, 332, 561, 92, 304, 308,
+                 433, 80, 194, 221, 311, 28, 372, 166, 338, 254, 421, 137,
+                 563, 268, 708, 590, 505, 106, 219, 356, 363, 352, 466, 212),
+   assessment1=c(124, 109, 230, 107, 222, 193, 283, 80, 107, 153, 212, 41,
+                 162, 48, 263, 95, 42, 19, 252, 89, 193, 9, 145, 42, 194, 23,
+                 138, 62, 128, 17, 282, 127, 67, 29, 197, 74, 191, 44,
+                 216, 134, 211, 19, 100, 88, 102, 209, 269, 25))
```

We'll review the analysis of covariance using an ARM data format. I used `base.dat` as a template for different randomizations, and comparend algorithms by replacing the base data with different examples of ANCOVA. These have been moved to a later section, but I'll keep the `arm.dat`/`base.dat` convention in place.

```
> base.dat <- cochran.arm.dat
> arm.dat <- base.dat
```

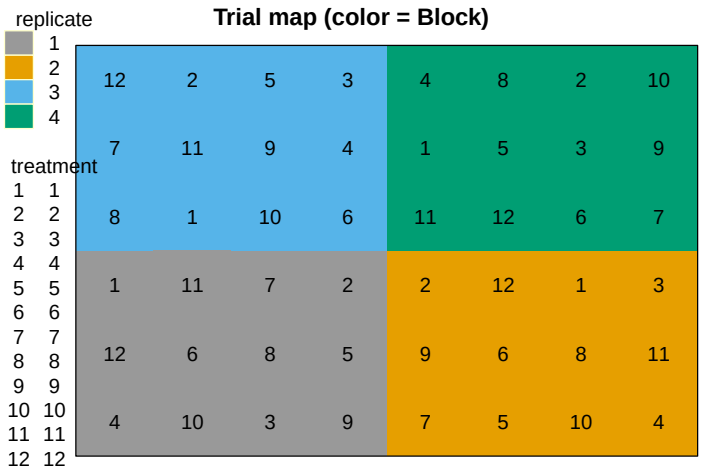
Later on, we'll need the number of treatments and replicates.

```
> reps <- length(levels(arm.dat$replicate))
> treatments <- length(levels(arm.dat$treatment))
```

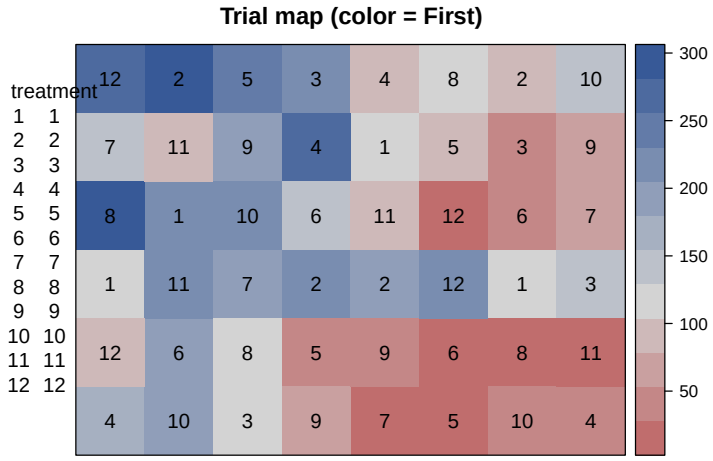
Trial Maps

Consider that the blocks for this experiment do not adequately capture spatial variation of the first eelworm measurements.

```
> library(desplot)
> desplot(replicate~x*y, data=arm.dat, cex=1, text=treatment,
+         main="Trial map (color = Block)", col.regions=cbPalette)
```




```
> desplot(assessment1~x*y, data=arm.dat, cex=1, text=treatment,
+         main="Trial map (color = First)")
```



Models

Cochran and Cox present an analysis of covariance that extends the designed experiment to include simple linear regression of the covariate.

```
> anova(lm(assessment2 ~ replicate + treatment,data=arm.dat))

Analysis of Variance Table

Response: assessment2
           Df      Sum Sq     Mean Sq F value     Pr(>F)
```

```
replicate  3 289426.5000 96475.50000 8.18632 0.00032762 ***
treatment 11 313234.1667 28475.83333 2.41628 0.02478343 *
Residuals 33 388904.0000 11784.96970
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1
```

```
> anova(lm(assessment2 ~ replicate + treatment + assessment1, data=arm.dat))
```

```
Analysis of Variance Table

Response: assessment2
            Df      Sum Sq      Mean Sq  F value     Pr(>F)
replicate    3 289426.5000  96475.50000 19.87811 1.8509e-07 ***
treatment   11 313234.1667  28475.83333  5.86725 4.0182e-05 ***
assessment1  1 233596.6698 233596.66984 48.13098 7.4151e-08 ***
Residuals   32 155307.3302   4853.35407
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1
```

We will also want to consider testing if the response is quadratic with regard to the covariate, and if there is a treatment by covariate interaction.

```
> anova(lm(assessment2 ~ replicate + treatment + assessment1 + I(assessment1^2),data=arm.dat
```

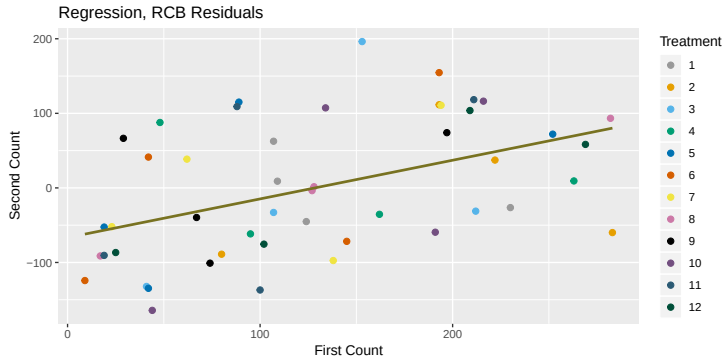
```
Analysis of Variance Table

Response: assessment2
                 Df       Sum Sq      Mean Sq  F value     Pr(>F)
replicate         3 289426.50000  96475.50000 20.05202 2.0593e-07 ***
treatment        11 313234.16667  28475.83333  5.91858 4.2920e-05 ***
assessment1       1 233596.66984 233596.66984 48.55207 8.0897e-08 ***
I(assessment1^2)  1   6158.24199   6158.24199  1.27996    0.26658
Residuals        31 149149.08817   4811.26091
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1
```

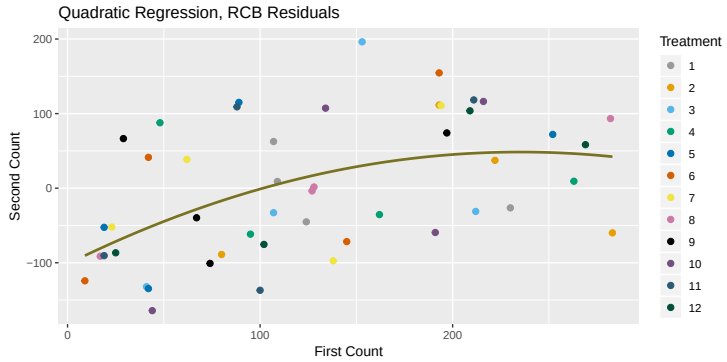
```
> anova(lm(assessment2 ~ replicate + treatment + assessment1 + treatment:assessment1,data=ar
```

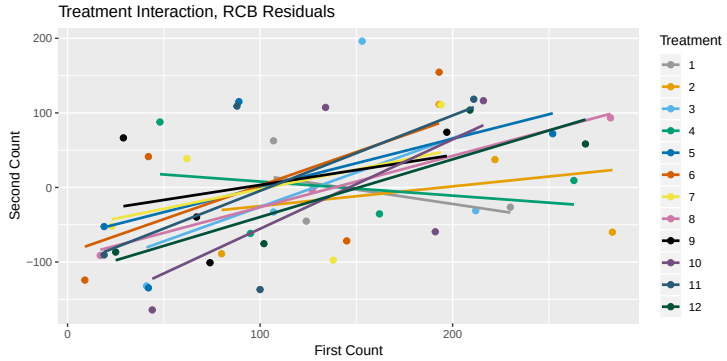
```
Analysis of Variance Table

Response: assessment2
                     Df       Sum Sq      Mean Sq  F value     Pr(>F)
replicate             3 289426.50000  96475.50000 16.42166 1.0032e-05 ***
treatment            11 313234.16667  28475.83333  4.84704 0.00095291 ***
assessment1           1 233596.66984 233596.66984 39.76185 2.9746e-06 ***
treatment:assessment1 11  31934.53626   2903.13966  0.49416 0.88600912
Residuals            21 123372.79390   5874.89495
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1
```

To illustrate the differences, plot RCB residuals against the different models:

```
> arm.dat$RCBResidual <- resid(lm(assessment2 ~ replicate + treatment,data=arm.dat))
```

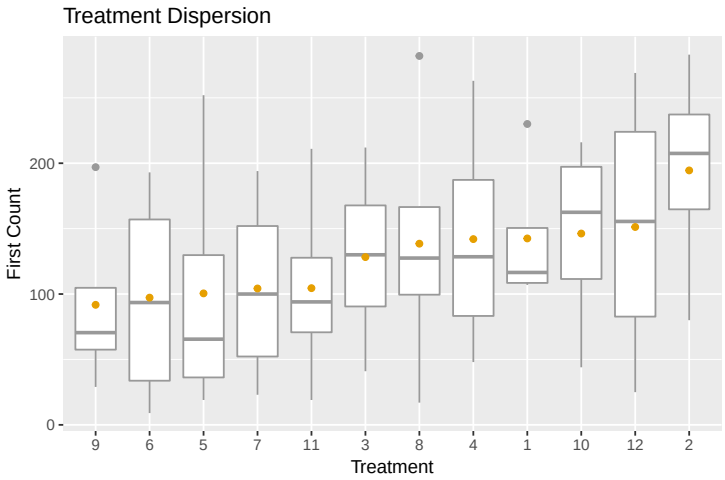






Treatments Disperson over Covariates

We plot covariate versus treatments, sorted by covariate average.



Two features to look for:

- Range of ovariate means should be small to minimize starting bias for treatments
- Standard deviations should be uniform and large to provide each treatment with an appropriate sample over covariates.

Randomizations

What does an optimal randomization look like? We'll create multiple randomizations and compare measures of covariate dispersion over treatments.

To calculate D-optimality, we want to normalize covariates to a scale of 0-1.

```
> arm.dat$cov <- arm.dat$assessment1/max(arm.dat$assessment1)
```

Simple function to compute $|X^t X|$ from a linear model. We don't actually need to solve the model, we just need `x` matrices.

```
> optimality <- function(lm,contrast=NULL) {
+   if(is.null(lm$x)) {
+     lm <- update(lm,x=TRUE)
+   }
+   X <- lm$x
+   I <- t(X) %*% X
+   return(list(D=det(I),A=sum(diag(I))))
+ }
```

2000 seems to be an appropriate number of randomizations to test.

```
> simulations <- 2000
> set.seed(simulations)
```

Create data structures to hold optimality measures.

```
> randomization.label <- paste(simulations," Randomizations")
> rcb.designs <- vector("list", length = simulations)
> rcb.designs.dat <- data.frame(
+   Design = rep(0,simulations),
+   MaxMeanDif = rep(0,simulations),
+   MaxSDDif = rep(0,simulations),
+   MeanSD = rep(0,simulations),
+   MomentNorm = rep(0,simulations),
+   ANCOVA = rep(0,simulations),
+   Regression = rep(0,simulations)
+ )
> var.designs.dat <- rcb.designs.dat
> var.designs <- rcb.designs
```

Execute randomizations. We'll compare analysis for simple regression of response versus covariate and a proper RCB analysis of covariance model.

```
> for (s in 1:simulations) {
+   if(s!=1) {
+     #rerandomize RCB treatments
+     for(blk in levels(arm.dat$replicate)) {
+       arm.dat$treatment[arm.dat$replicate==blk] <- levels(arm.dat$treatment)[sample(1:trea
+     }
+   }
+
+   arm.dat$treatment <- as.factor(arm.dat$treatment)
+   rcb.designs[[s]] <- arm.dat
+   #mean and standard deviation for covariate among treatments
+   cov.means <- tapply(arm.dat$cov,list(arm.dat$treatment),mean)
+   cov.sds <- tapply(arm.dat$cov,list(arm.dat$treatment),sd)
+
+   rcb.designs.dat$MaxMeanDif[s] <- max(cov.means)-min(cov.means)
+   rcb.designs.dat$MaxSDDif[s] <- max(cov.sds)-min(cov.sds)
+   rcb.designs.dat$MeanSD[s] <- mean(cov.sds)
+   rcb.designs.dat$MomentNorm[s] <- sqrt(sum(rcb.designs.dat$MaxMeanDif[s]^2+rcb.designs.da
+
+   #fit linear models
+  rcb.lm <- lm(assessment1 ~ replicate + treatment, data = arm.dat, method="model.frame", m
+  ancova.lm <- lm(assessment1 ~ replicate + treatment + cov, data = arm.dat, model=FALSE, x
+  rcb.regression.lm <- lm(assessment1 ~ cov +  treatment, data = arm.dat, model=FALSE, x =
+
+   opt.rcb.ancova <- optimality(ancova.lm)
+   opt.rcb.regression <- optimality(rcb.regression.lm)
+
+   # randomize by covariate
```

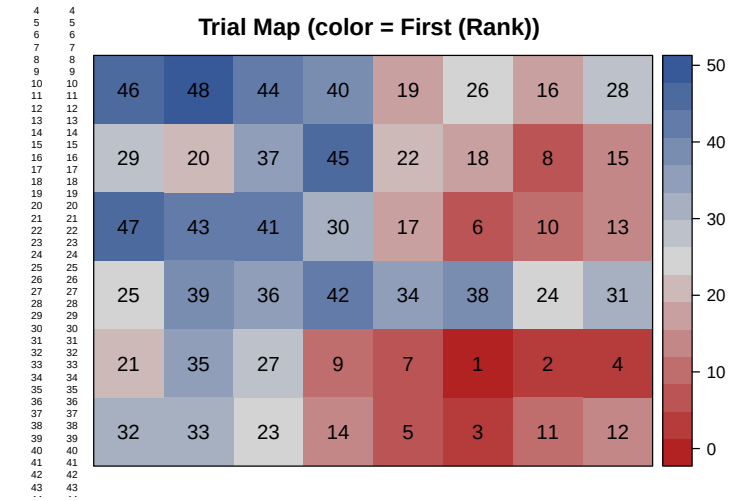
```
+   arm.dat$Rank <- rank(arm.dat$cov,ties.method = c("random"))
+   arm.dat$VariableBlock <- ceiling(arm.dat$Rank/treatments)
+   for(blk in 1:max(arm.dat$VariableBlock)) {
+     arm.dat$VariableTreatment[arm.dat$VariableBlock==blk] <- sample(1:treatments)
+   }
+
+   arm.dat$VariableBlock <- as.factor(arm.dat$VariableBlock)
+   arm.dat$VariableTreatment <- as.factor(arm.dat$VariableTreatment)
+   var.designs[[s]] <- arm.dat
+   cov.means <- tapply(arm.dat$cov,list(arm.dat$VariableTreatment),mean)
+   cov.sds <- tapply(arm.dat$cov,list(arm.dat$VariableTreatment),sd)
+
+   var.designs.dat$MaxMeanDif[s] <- max(cov.means)-min(cov.means)
+   var.designs.dat$MaxSDDif[s] <- max(cov.sds)-min(cov.sds)
+   var.designs.dat$MeanSD[s] <- mean(cov.sds)
+   var.designs.dat$MomentNorm[s] <- sqrt(sum(var.designs.dat$MaxMeanDif[s]^2+var.designs.da
+
+   var.ancova.lm <- lm(assessment1 ~ VariableBlock + VariableTreatment + cov, data = arm.da
+   var.regression.lm <- lm(assessment1 ~ cov +  VariableTreatment, data = arm.dat, model=FA
+
+   opt.var.regression <- optimality(var.regression.lm)
+   opt.var.ancova <- optimality(var.ancova.lm)
+
+   #calculate optimality from information
+   rcb.designs.dat$ANCOVA[s] = opt.rcb.ancova$D
+   rcb.designs.dat$Regression[s] = opt.rcb.regression$D
+   var.designs.dat$ANCOVA[s] = opt.var.ancova$D
+   var.designs.dat$Regression[s]= opt.var.regression$D
+
+ }
```

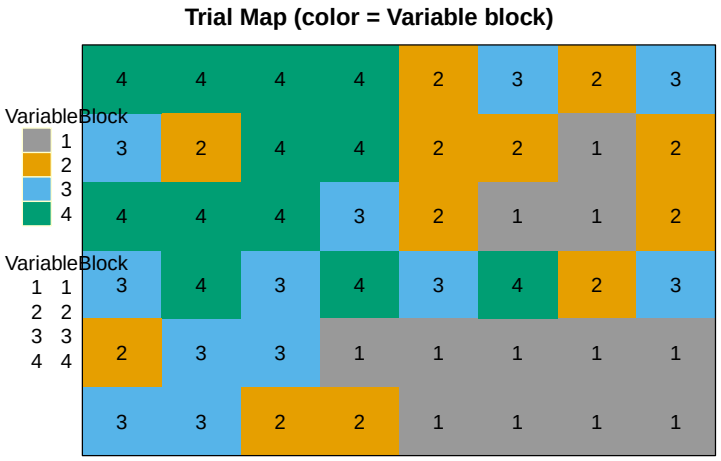
Combined the randomizations and label for graphs.

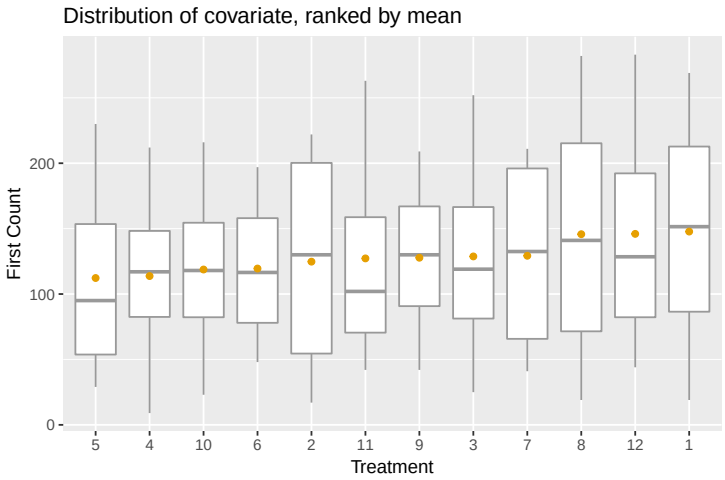
```
> rcb.designs.dat$Blocking <- 'RCB'
> var.designs.dat$Blocking <- 'Variable'
> combined.dat <- rbind(rcb.designs.dat,var.designs.dat)
```

We'll show the trial map for the an arbitrary variable blocked randomization.

```
> example.var <- var.designs[[3]]
```


Trial Map (color = First (Rank))
46 48 44 40 19 26 16 28
29 20 37 45 22 18 8 15
47 43 41 30 17 6 10 13
25 39 36 42 34 38 24 31
21 35 27 9 7 1 2 4
32 33 23 14 5 3 11 12
50
40
30
20
10
0


Trial Map (color = Variable block)
VariableBlock
1
2
3
4
4 4 4 4 2 3 2 3
3 2 4 4 2 2 1 2
4 4 4 3 2 1 1 2
3 4 3 4 3 4 2 3
2 3 3 1 1 1 1 1
3 3 2 2 1 1 1 1



Select Most and Least Optimal Randomizations

We select from the various RCB and variable blocked designs the most and least optimal. We'll use this to illustrate optimal and sub-optimal designs. We'll leave space in the first set to include the original randomization and example adaptive randomizations.

```
> most.rcb.designs <- c(which.max(rcb.designs.dat$Regression)[1],
+                 which.max(rcb.designs.dat$ANCOVA)[1], 1, 1)
> most.var.designs <- c(which.max(var.designs.dat$Regression)[1],
+                 which.max(var.designs.dat$ANCOVA)[1])
> least.rcb.designs <- c(which.min(rcb.designs.dat$Regression)[1],
+                 which.min(rcb.designs.dat$ANCOVA)[1])
> least.var.designs <- c(which.min(var.designs.dat$Regression)[1],
+                 which.min(var.designs.dat$ANCOVA)[1])
```

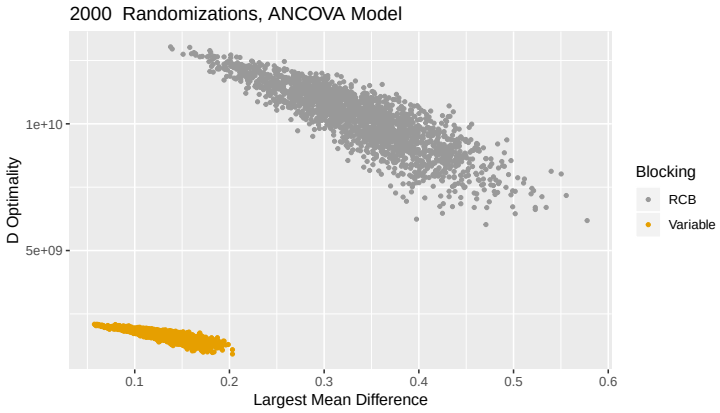
We'll use this data structure to add points to plots comparing randomizations.

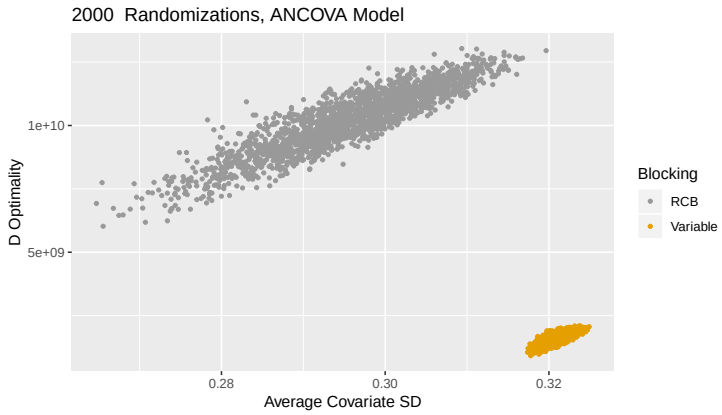
```
> reference.rcb.dat <- subset(rcb.designs.dat[most.rcb.designs,])
> reference.rcb.dat$Criteria <- c("Regression",
+                                 "ANCOVA",
+                                "Original",
+                                "Adaptive"
+                                )
> reference.post.dat <- subset(var.designs.dat[most.var.designs,])
> reference.post.dat$Criteria <- reference.rcb.dat$Criteria[c(1,2)]
> reference.dat <- rbind(reference.rcb.dat,reference.post.dat)
```

Plot D-optimality for both regression and ANCOVA against treatment means and treatment dispersions


2000 Randomizations, Regression Model
6.5e+07
6.0e+07
5.5e+07
5.0e+07
D Optimality
0.1
0.2
0.3
0.4
0.5
0.6
Largest Mean Difference
Blocking
RCB
Variable


2000 Randomizations, Regression Model
6.5e+07
6.0e+07
5.5e+07
5.0e+07
D Optimality
0.28
0.30
0.32
Average Covariate SD
Blocking
RCB
Variable

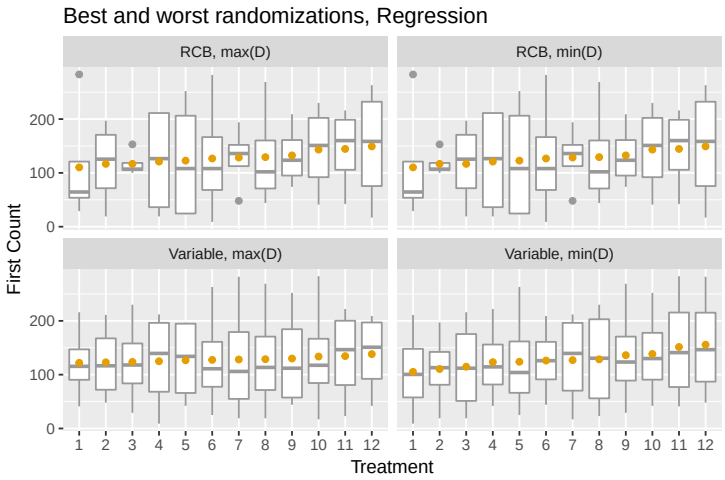

2000 Randomizations, ANCOVA Model
1e+10
5e+09
D Optimality
0.1
0.2
0.3
0.4
0.5
0.6
Largest Mean Difference
Blocking
RCB
Variable



Optimality for Regression

Select the most and least optimal designs with regard to covariate regression.

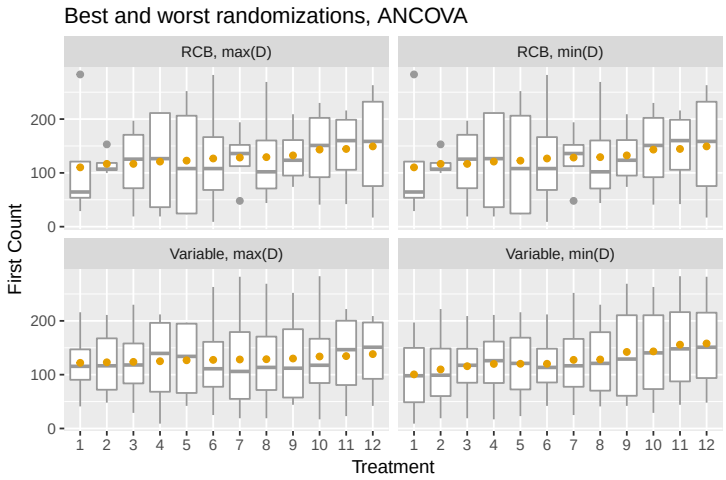
```
> best.var <- var.designs[[most.var.designs[1]]]
> CovRank <- rank(tapply(best.var$assessment1,list(best.var$VariableTreatment),mean),ties.me
> best.var$CovRank <- CovRank[best.var$VariableTreatment]
> best.var$CovRank <- as.factor(best.var$CovRank)
> least.var <- var.designs[[least.rcb.designs[1]]]
> CovRank <- rank(tapply(least.var$assessment1,list(least.var$VariableTreatment),mean),ties.
> least.var$CovRank <- CovRank[least.var$VariableTreatment]
> least.var$CovRank <- as.factor(least.var$CovRank)
> best.var$Design <- 'Variable, max(D)'
> least.var$Design <- 'Variable, min(D)'
> best.rcb <- rcb.designs[[most.rcb.designs[1]]]
> CovRank <- rank(tapply(best.rcb$assessment1,list(best.rcb$treatment),mean),ties.method = c
> best.rcb$CovRank <- CovRank[best.rcb$treatment]
> best.rcb$CovRank <- as.factor(best.rcb$CovRank)
> least.rcb <- rcb.designs[[most.rcb.designs[1]]]
> CovRank <- rank(tapply(least.rcb$assessment1,list(least.rcb$treatment),mean),ties.method =
> least.rcb$CovRank <- CovRank[least.rcb$treatment]
> least.rcb$CovRank <- as.factor(least.rcb$CovRank)
> best.rcb$Design <- 'RCB, max(D)'
> least.rcb$Design <- 'RCB, min(D)'
> comp.randomizatons <- rbind(best.var,least.var,best.rcb,least.rcb)
```



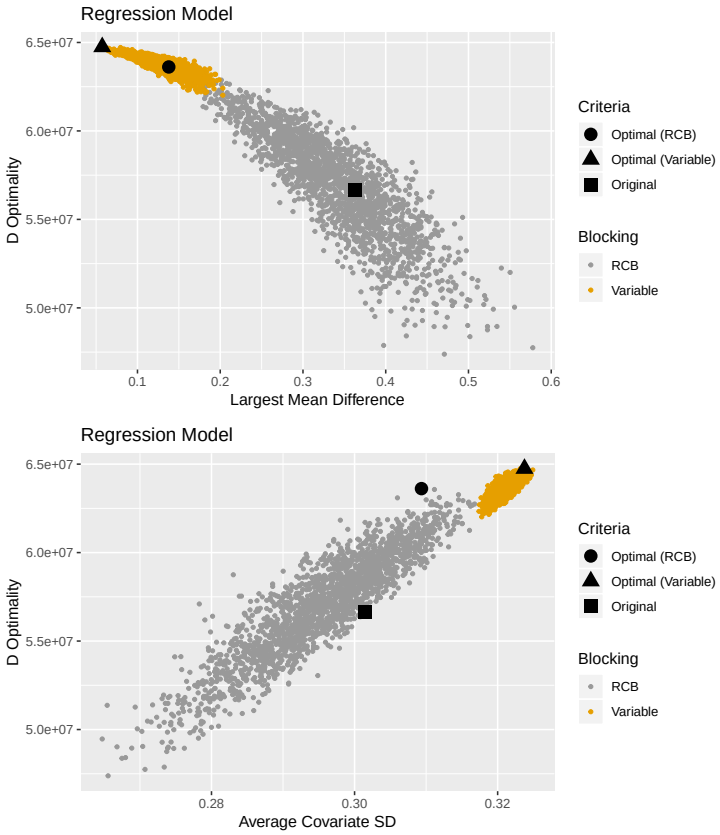
Optimality for ANCOVA

Select the most and least optimal designs with regard to analysis of covariance.

```
> example.var <- var.designs[[1]]
> best.var <- var.designs[[most.var.designs[2]]]
> CovRank <- rank(tapply(best.var$cov,list(best.var$VariableTreatment),mean),ties.method = c
> best.var$CovRank <- CovRank[best.var$VariableTreatment]
> best.var$CovRank <- as.factor(best.var$CovRank)
> least.var <- var.designs[[least.var.designs[2]]]
> CovRank <- rank(tapply(least.var$cov,list(least.var$VariableTreatment),mean),ties.method =
> least.var$CovRank <- CovRank[least.var$VariableTreatment]
> least.var$CovRank <- as.factor(least.var$CovRank)
> best.var$Design <- 'Variable, max(D)'
> least.var$Design <- 'Variable, min(D)'
> best.rcb <- rcb.designs[[most.rcb.designs[2]]]
> CovRank <- rank(tapply(best.rcb$cov,list(best.rcb$treatment),mean),ties.method = c("random
> best.rcb$CovRank <- CovRank[best.rcb$treatment]
> best.rcb$CovRank <- as.factor(best.rcb$CovRank)
> least.rcb <- rcb.designs[[most.rcb.designs[2]]]
> CovRank <- rank(tapply(least.rcb$cov,list(least.rcb$treatment),mean),ties.method = c("rand
> least.rcb$CovRank <- CovRank[least.rcb$treatment]
> least.rcb$CovRank <- as.factor(least.rcb$CovRank)
> best.rcb$Design <- 'RCB, max(D)'
> least.rcb$Design <- 'RCB, min(D)'
> comp.randomizatons <- rbind(best.var,least.var,best.rcb,least.rcb)
```



Add points to the graphs representing the original randomization and the optimal regression model.



Variables as Columns

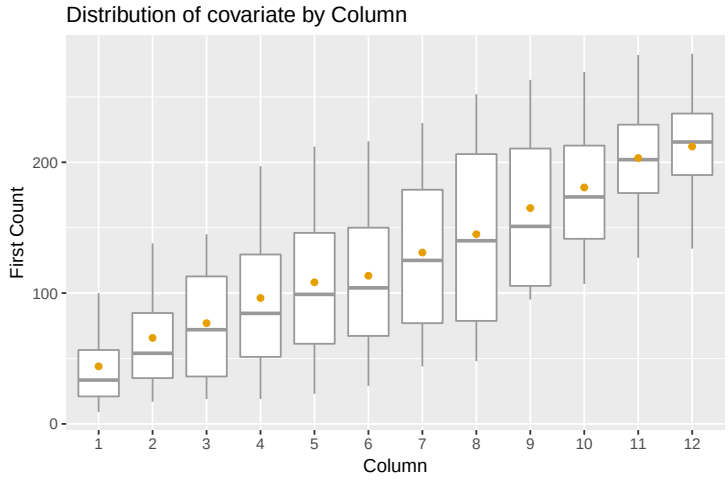
This design was randomized as a 12x12 Latin square, and we keep the first 4 rows. These rows are the complete blocks, and columns are used to identify covariate ranks within complete blocks.

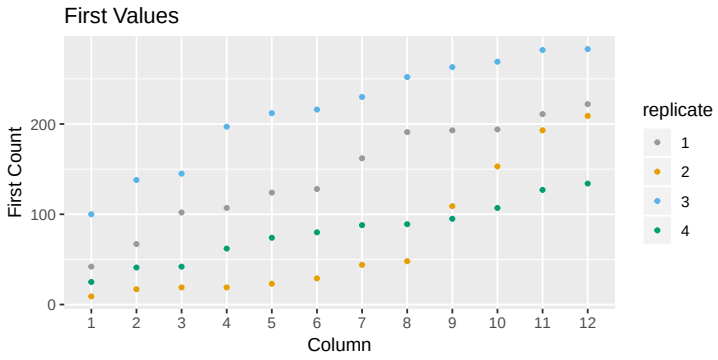
```
> latin.dat <- data.frame(
+ plot=as.factor(c(907, 1012, 1105, 1208, 910, 1002, 1111, 1209, 906, 1001, 1104, 1212, 903,
+ treatment=as.factor(c(1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 6, 6, 6,
+ replicate=as.factor(c(1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3,
+ column=c(7,12,5,8,10,2,11,9,6,1,4,12,3,7,8,2,5,10,9,1,11,6,12,7,4,3,2,10,8,5,1,6,9,11,7,3,
+ y=c(1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4, 1,
+ x=c(7, 12, 5, 8, 10, 2, 11, 9, 6, 1, 4, 12, 3, 7, 8, 2, 5, 10, 9, 1, 11, 6, 12, 7, 4, 3, 2
+ assessment1=c(124, 109, 230, 107, 222, 193, 283, 80, 107, 153, 212, 41, 162, 48, 263, 95,
+ assessment2=c(268, 132, 256, 236, 408, 292, 280, 142, 415, 454, 386, 176, 365, 298, 379, 1
+ )
> row.names(latin.dat) <- latin.dat$plot
> replicates <- length(levels(latin.dat$replicate))
> treatments <- length(levels(latin.dat$treatment))
> latin.dat$cov <- latin.dat$assessment1/max(latin.dat$assessment1)
> latin.dat$AdaptiveTreatment <- 0
> latin.dat$AdaptiveRank <- 0
> #
> for(r in 1:replicates) {
+   rows <- latin.dat$replicate==r
+   #get the treatment column numbers
+   current.columns <- latin.dat$column[rows]
+   current.treatments <- latin.dat$treatment[rows]
+
+   #get covariate and ranks
+   current.covariates <- latin.dat$assessment1[rows]
+   current.rank <- rank(current.covariates,ties.method = "random")
+
+   #track the adaptive rank
+   latin.dat$AdaptiveRank[rows] <- current.rank
+
+   #order covariates
+   current.order <- order(current.covariates)
+   current.plots <- latin.dat$plot[rows]
+
+   #map column order to plot ordered by covariates
+   latin.dat[as.character(current.plots[current.order]),'AdaptiveTreatment'] <- current.tre
+
+ }
```

Calculate optimality for the example column variable blocks.

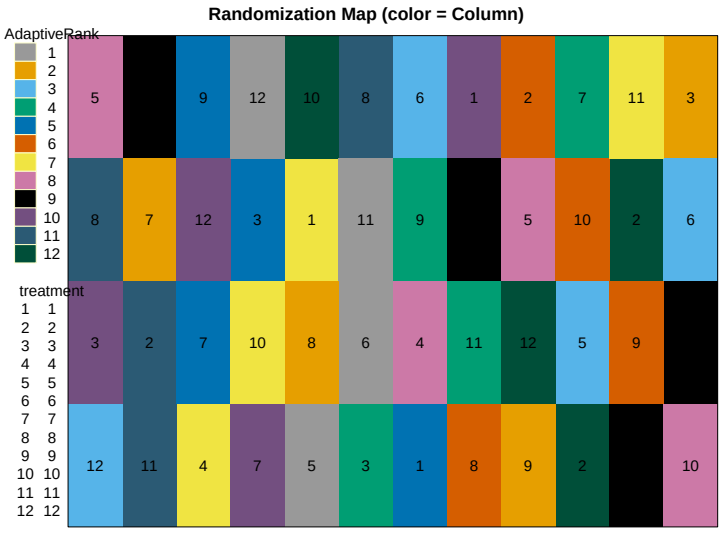
```
> latin.dat$AdaptiveTreatment <- as.factor(latin.dat$AdaptiveTreatment)
> model1 <- optimality(lm(assessment2 ~ AdaptiveTreatment + cov,data=latin.dat,x=TRUE))
> model2 <- optimality(lm(assessment2 ~ replicate +  AdaptiveTreatment + cov,data=latin.dat,
> adaptive <- which(reference.rcb.dat$Criteria == "Adaptive")
> reference.dat$Regression[adaptive] <- model1$D
> reference.dat$ANCOVA[adaptive] <- model2$D
> cov.means <- tapply(latin.dat$cov,list(latin.dat$AdaptiveTreatment),mean)
> cov.sds <- tapply(latin.dat$cov,list(latin.dat$AdaptiveTreatment),sd)
> reference.dat$MaxMeanDif[adaptive] <- max(cov.means)-min(cov.means)
> reference.dat$MaxSDDif[adaptive] <- max(cov.sds)-min(cov.sds)
> reference.dat$MeanSD[adaptive] <- mean(cov.sds)
> reference.dat$MomentNorm[adaptive] <- sqrt(sum(reference.dat$MaxMeanDif[adaptive]^2+refere
```

Plot the optimality of the previous randomizations, adding a point for the column variable block example.






```
> desplot(AdaptiveRank~x*y, data=latin.dat, cex=1, text=treatment,
+         main="Randomization Map (color = Column)", col.regions=cbPalette)
```



Adaptive Design

While column variable blocks improves treatment dispersion, we don't find maximally optimal designs. Since D-otpimal designs tend to minimize the spread of covariate means over treatments, while maximizing covariate standard deviation, we develop an adaptive algorithm that alternates between minimizing means and maximiming deviances.

Step 1

Start with first replicate, assign treatments by rank.

```
> arm.dat <- base.dat
> replicates <- length(levels(arm.dat$replicate))
> treatments <- length(levels(arm.dat$treatment))
> arm.dat$cov <- arm.dat$assessment1/max(arm.dat$assessment1)
> arm.dat$AdaptiveTreatment <- 0
> arm.dat$CovRank <- 0
> #for(blk in 1:max(arm.dat$VariableBlock)) {
> randomization <-  1:treatments
> rows <- arm.dat$replicate==1
> cov1 <- arm.dat$cov[rows]
> assigned <- arm.dat$AdaptiveTreatment
> rank1 <- rank(cov1,ties.method = "random")
> randomization[rank1]

 [1]  5 12  4  7  1  9 10  6  2  8 11  3

> arm.dat$AdaptiveTreatment[rows] <- rank1
> arm.dat$CovRank[rows] <- rank1
> arm.dat$Step <- 1
> steps.dat <- arm.dat
```

Step 2

Reverse treatment order by rank, relative to step 1.

```
> rows <- arm.dat$replicate==2
> cov2 <- arm.dat$cov[rows]
> rank2 <- rank(cov2,ties.method = "random")
> cov2[rank2]

 [1] 0.1024734982332 0.0671378091873 0.1554770318021 0.0600706713781
 [5] 0.1696113074205 0.3851590106007 0.0671378091873 0.6819787985866
 [9] 0.0318021201413 0.0812720848057 0.5406360424028 0.7385159010601

> rank2

 [1]  9 11 10  8  4  1  5  2  6  7  3 12

> rank2[treatments:1]

 [1] 12  3  7  6  2  5  1  4  8 10 11  9

> (treatments:1)[rank2]

 [1]  4  2  3  5  9 12  8 11  7  6 10  1

> arm.dat$AdaptiveTreatment[rows] <- (treatments:1)[rank2]
> arm.dat$CovRank[rows] <- rank2
> arm.dat$Step <- 2
> steps.dat <- rbind(arm.dat,steps.dat)
```

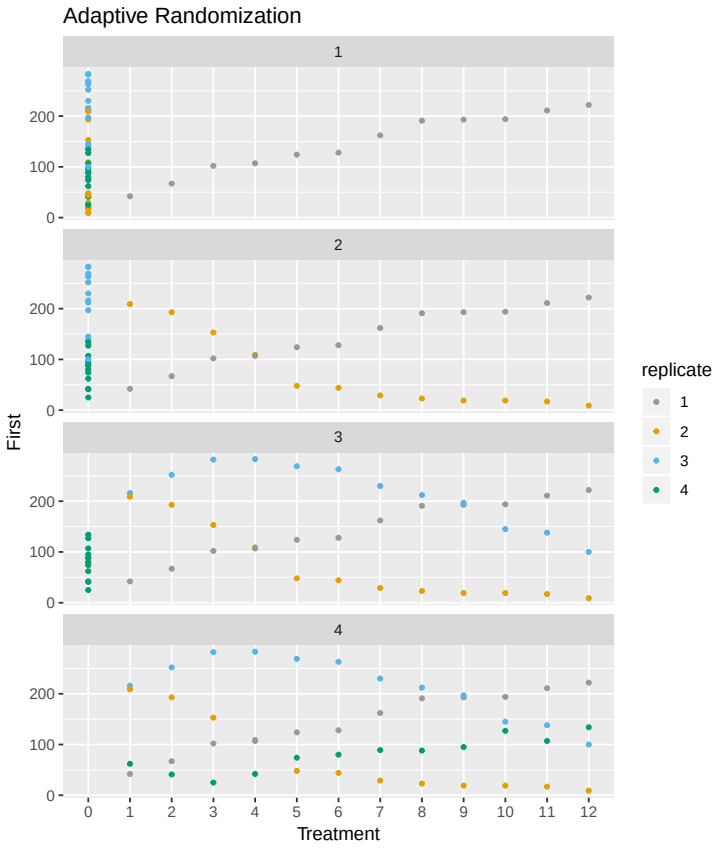
Steps 3 and 4

Iterate over remaining replicates. If the step is odd numbered, then treatments with the smallest range should get the most extreme values; otherwise, treatments with the largest mean deviations get the plots that will normalize covariate means.

Using plot numbers as row names simplifies matching covariates to treatments.

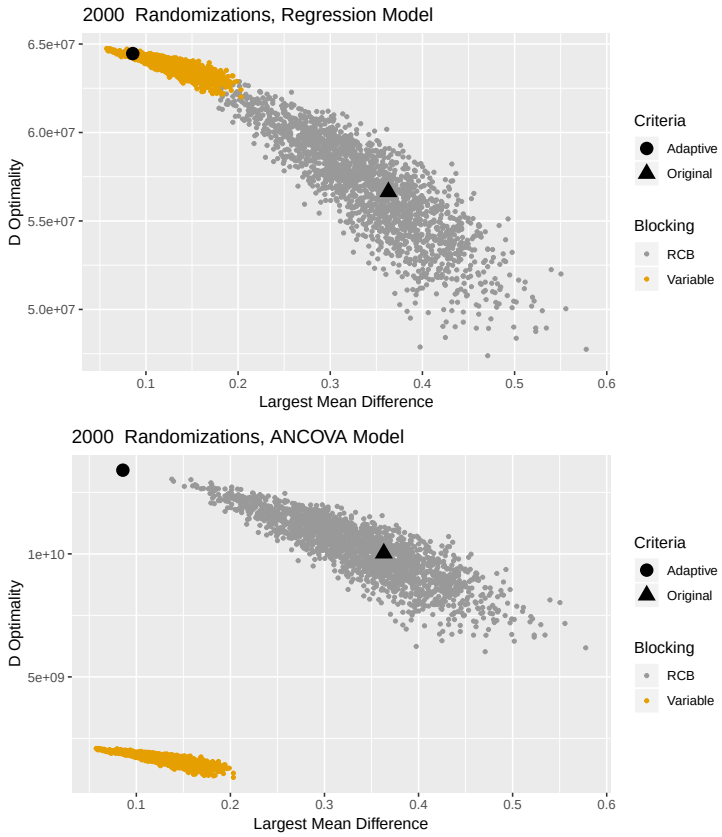
```
> row.names(arm.dat) <- arm.dat$plot

> for(r in 3:replicates) {
+   assigned.dat <- subset(arm.dat,arm.dat$AdaptiveTreatment>0)
+   current.sd <- tapply(assigned.dat$assessment1,list(assigned.dat$AdaptiveTreatment),sd)
+   current.mean <- tapply(assigned.dat$assessment1,list(assigned.dat$AdaptiveTreatment),mea
+
+   rows <- arm.dat$replicate==r
+   cov <- arm.dat$cov[rows]
+   plots <- arm.dat$plot[rows]
+
+   plot.order <- order(cov - mean(assigned.dat$assessment1))
+
+   sd.order <- order(current.sd)
+   mean.order <- order(current.mean)
+
+   if(r%%2==1) {
+     arm.dat[as.character(plots[plot.order[(treatments:1)]]),'AdaptiveTreatment'] <- (1:tre
+   } else {
+     arm.dat[as.character(plots[plot.order[(treatments:1)]]),'AdaptiveTreatment'] <- (1:tre
+   }
+   arm.dat$Step <- r
+   steps.dat <- rbind(arm.dat,steps.dat)
+ }
```

We can visualize the process by plotting covariate against treatment at each step.



Now calculate optimality for the proposed randomization and add to randomization plots.

```
> arm.dat$AdaptiveTreatment <- as.factor(arm.dat$AdaptiveTreatment)
> model1 <- optimality(lm(assessment2 ~ AdaptiveTreatment + cov,data=arm.dat,x=TRUE))
> model2 <- optimality(lm(assessment2 ~ replicate +  AdaptiveTreatment + cov,data=arm.dat,x=
> adaptive <- which(reference.rcb.dat$Criteria == "Adaptive")
> reference.dat$Regression[adaptive] <- model1$D
> reference.dat$ANCOVA[adaptive] <- model2$D
> cov.means <- tapply(arm.dat$cov,list(arm.dat$AdaptiveTreatment),mean)
> cov.sds <- tapply(arm.dat$cov,list(arm.dat$AdaptiveTreatment),sd)
> reference.dat$MaxMeanDif[adaptive] <- max(cov.means)-min(cov.means)
> reference.dat$MaxSDDif[adaptive] <- max(cov.sds)-min(cov.sds)
> reference.dat$MeanSD[adaptive] <- mean(cov.sds)
> reference.dat$MomentNorm[adaptive] <- sqrt(sum(reference.dat$MaxMeanDif[adaptive]^2+refere
```



Additional Examples

These are some additional data sets that were used to compare different randomization algorithms.

Change commented code to produce randomizations for different data sets. Data are from [MJ02], [ST60], [GG84] and [Kue00].

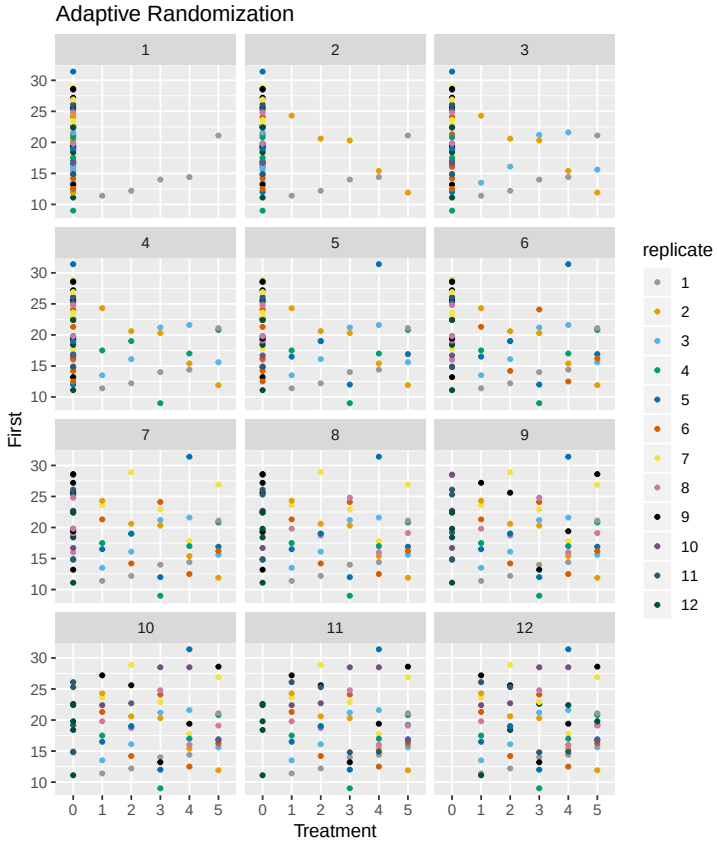
```
> #base.dat <- milliken3.15.arm.dat
> #base.dat <- steel17.10.dat
> #base.dat <- gomez.10.4.arm.dat
> #base.dat <- cochran.arm.dat
> base.dat <- milliken10.2.arm.dat
> #base.dat <- kuehl17.3.arm.dat
> arm.dat <- base.dat
> arm.dat$cov <- arm.dat$assessment1/max(arm.dat$assessment1)

> replicates <- length(levels(arm.dat$replicate))
> treatments <- length(levels(arm.dat$treatment))
```

```
> rcb.designs.dat <- data.frame(
+   Design = rep(0,simulations),
+   MaxMeanDif = rep(0,simulations),
+   MaxSDDif = rep(0,simulations),
+   MeanSD = rep(0,simulations),
+   MomentNorm = rep(0,simulations),
+   ANCOVA = rep(0,simulations),
+   Regression = rep(0,simulations)
+ )
> var.designs.dat <- rcb.designs.dat
> var.designs <- rcb.designs

> for (s in 1:simulations) {
+   if(s!=1) {
+     #rerandomize RCB treatments
+     for(blk in levels(arm.dat$replicate)) {
+       arm.dat$treatment[arm.dat$replicate==blk] <- levels(arm.dat$treatment)[sample(1:trea
+     }
+   }
+
+   arm.dat$treatment <- as.factor(arm.dat$treatment)
+   rcb.designs[[s]] <- arm.dat
+   #mean and standard deviation for covariate among treatments
+   cov.means <- tapply(arm.dat$cov,list(arm.dat$treatment),mean)
+   cov.sds <- tapply(arm.dat$cov,list(arm.dat$treatment),sd)
+
+   rcb.designs.dat$MaxMeanDif[s] <- max(cov.means)-min(cov.means)
+   rcb.designs.dat$MaxSDDif[s] <- max(cov.sds)-min(cov.sds)
+   rcb.designs.dat$MeanSD[s] <- mean(cov.sds)
+   rcb.designs.dat$MomentNorm[s] <- sqrt(sum(rcb.designs.dat$MaxMeanDif[s]^2+rcb.designs.da
+
+   #fit linear models
+   rcb.lm <- lm(assessment1 ~ replicate + treatment, data = arm.dat, model=FALSE, x = TRUE)
+   rcb.ancova.lm <- lm(assessment1 ~ replicate + treatment + cov, data = arm.dat, model=FAL
+
+   rcb.regression.lm <- lm(assessment1 ~ cov +  treatment, data = arm.dat, model=FALSE, x =
+
+   opt.rcb.ancova <- optimality(rcb.ancova.lm)
+   opt.rcb.regression <- optimality(rcb.regression.lm)
+
+   # randomize by covariate
+   arm.dat$Rank <- rank(arm.dat$cov,ties.method = c("random"))
+   arm.dat$VariableBlock <- ceiling(arm.dat$Rank/treatments)
+   for(blk in 1:max(arm.dat$VariableBlock)) {
+     arm.dat$VariableTreatment[arm.dat$VariableBlock==blk] <- sample(1:treatments)
+   }
```

```
+
+   arm.dat$VariableBlock <- as.factor(arm.dat$VariableBlock)
+   arm.dat$VariableTreatment <- as.factor(arm.dat$VariableTreatment)
+   var.designs[[s]] <- arm.dat
+   cov.means <- tapply(arm.dat$cov,list(arm.dat$VariableTreatment),mean)
+   cov.sds <- tapply(arm.dat$cov,list(arm.dat$VariableTreatment),sd)
+
+   var.designs.dat$MaxMeanDif[s] <- max(cov.means)-min(cov.means)
+   var.designs.dat$MaxSDDif[s] <- max(cov.sds)-min(cov.sds)
+   var.designs.dat$MeanSD[s] <- mean(cov.sds)
+   var.designs.dat$MomentNorm[s] <- sqrt(sum(var.designs.dat$MaxMeanDif[s]^2+var.designs.da
+
+   var.ancova.lm <- lm(assessment1 ~ VariableBlock + VariableTreatment + cov, data = arm.da
+   var.regression.lm <- lm(assessment1 ~ cov +  VariableTreatment, data = arm.dat, model=FA
+
+   opt.var.regression <- optimality(var.regression.lm)
+   opt.var.ancova <- optimality(var.ancova.lm)
+
+   #calculate optimality from information
+   rcb.designs.dat$ANCOVA[s] = opt.rcb.ancova$D
+   rcb.designs.dat$Regression[s] = opt.rcb.regression$D
+   var.designs.dat$ANCOVA[s] = opt.var.ancova$D
+   var.designs.dat$Regression[s]= opt.var.regression$D
+
+ }
> rcb.designs.dat$Blocking <- 'RCB'
> var.designs.dat$Blocking <- 'Variable'
> combined.dat <- rbind(rcb.designs.dat,var.designs.dat)

> arm.dat$AdaptiveTreatment <- 0
> arm.dat$CovRank <- 0
> randomization <-  1:treatments
> rows <- arm.dat$replicate==1
> cov1 <- arm.dat$cov[rows]
> assigned <- arm.dat$AdaptiveTreatment
> rank1 <- rank(cov1,ties.method = "random")
> randomization[rank1]

[1] 3 1 4 2 5

> arm.dat$AdaptiveTreatment[rows] <- rank1
> arm.dat$CovRank[rows] <- rank1
> arm.dat$Step <- 1
> steps.dat <- arm.dat
> rows <- arm.dat$replicate==2
> cov2 <- arm.dat$cov[rows]
```

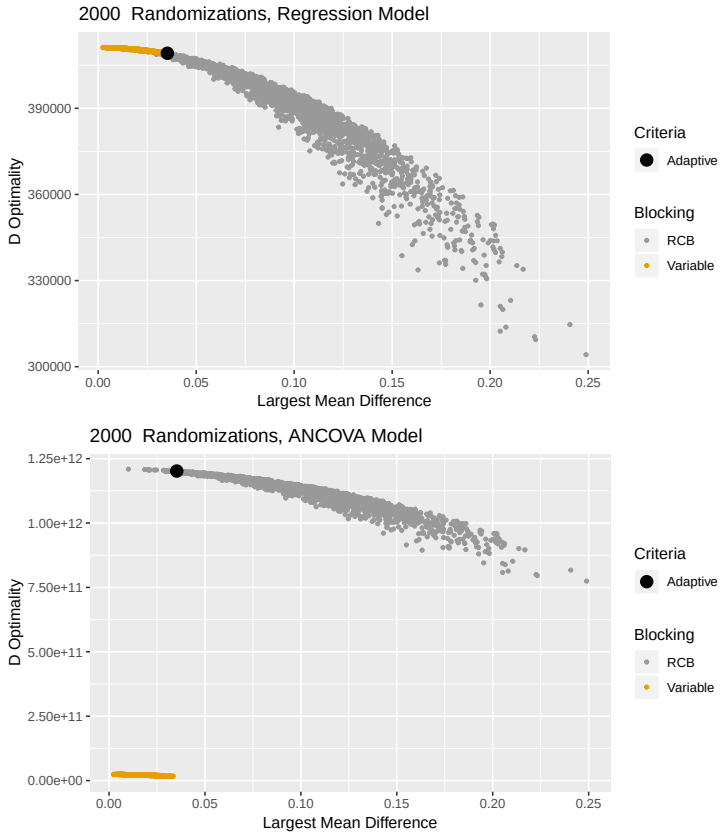
```
> rank2 <- rank(cov2,ties.method = "random")
> cov2[rank2]

[1] 0.656050955414 0.378980891720 0.646496815287 0.490445859873 0.773885350318

> rank2

[1] 2 4 3 1 5

> rank2[treatments:1]

[1] 5 1 3 4 2

> (treatments:1)[rank2]

[1] 4 2 3 5 1

> arm.dat$AdaptiveTreatment[rows] <- (treatments:1)[rank2]
> arm.dat$CovRank[rows] <- rank2
> arm.dat$Step <- 2
> steps.dat <- rbind(arm.dat,steps.dat)
> row.names(arm.dat) <- arm.dat$plot
> for(r in 3:replicates) {
+   assigned.dat <- subset(arm.dat,arm.dat$AdaptiveTreatment>0)
+   current.sd <- tapply(assigned.dat$assessment1,list(assigned.dat$AdaptiveTreatment),sd)
+   current.mean <- tapply(assigned.dat$assessment1,list(assigned.dat$AdaptiveTreatment),mea
+
+   rows <- arm.dat$replicate==r
+   cov <- arm.dat$cov[rows]
+   plots <- arm.dat$plot[rows]
+
+   plot.order <- order(cov - mean(assigned.dat$assessment1))
+
+   sd.order <- order(current.sd)
+   mean.order <- order(current.mean)
+
+   if(r%%2==1) {
+     arm.dat[as.character(plots[plot.order[(treatments:1)]]),'AdaptiveTreatment'] <- (1:tre
+   } else {
+     arm.dat[as.character(plots[plot.order[(treatments:1)]]),'AdaptiveTreatment'] <- (1:tre
+
+   }
+   arm.dat$Step <- r
+   steps.dat <- rbind(arm.dat,steps.dat)
+ }
> arm.dat$AdaptiveTreatment <- as.factor(arm.dat$AdaptiveTreatment)
> model1 <- optimality(lm(assessment2 ~ AdaptiveTreatment + cov,data=arm.dat,x=TRUE))
```

```
> model2 <- optimality(lm(assessment2 ~ replicate +  AdaptiveTreatment + cov,data=arm.dat,x=
> adaptive <- which(reference.rcb.dat$Criteria == "Adaptive")
> reference.dat$Regression[adaptive] <- model1$D
> reference.dat$ANCOVA[adaptive] <- model2$D
> cov.means <- tapply(arm.dat$cov,list(arm.dat$AdaptiveTreatment),mean)
> cov.sds <- tapply(arm.dat$cov,list(arm.dat$AdaptiveTreatment),sd)
> reference.dat$MaxMeanDif[adaptive] <- max(cov.means)-min(cov.means)
> reference.dat$MaxSDDif[adaptive] <- max(cov.sds)-min(cov.sds)
> reference.dat$MeanSD[adaptive] <- mean(cov.sds)
> reference.dat$MomentNorm[adaptive] <- sqrt(sum(reference.dat$MaxMeanDif[adaptive]^2+refere
```




```
> arm.dat$AdaptiveTreatment <- as.factor(arm.dat$AdaptiveTreatment)
> model1 <- optimality(lm(assessment2 ~ AdaptiveTreatment + cov,data=arm.dat,x=TRUE))
> model2 <- optimality(lm(assessment2 ~ replicate +  AdaptiveTreatment + cov,data=arm.dat,x=
> #model3 <- optimality(lm(assessment2 ~ cov +  replicate + AdaptiveTreatment + AdaptiveTrea
>
> adaptive <- which(reference.rcb.dat$Criteria == "Adaptive")
> reference.dat$Regression[adaptive] <- model1$D
```

```
> reference.dat$ANCOVA[adaptive] <- model2$D
> cov.means <- tapply(arm.dat$cov,list(arm.dat$AdaptiveTreatment),mean)
> cov.sds <- tapply(arm.dat$cov,list(arm.dat$AdaptiveTreatment),sd)
> reference.dat$MaxMeanDif[adaptive] <- max(cov.means)-min(cov.means)
> reference.dat$MaxSDDif[adaptive] <- max(cov.sds)-min(cov.sds)
> reference.dat$MeanSD[adaptive] <- mean(cov.sds)
> reference.dat$MomentNorm[adaptive] <- sqrt(sum(reference.dat$MaxMeanDif[adaptive]^2+refere
```



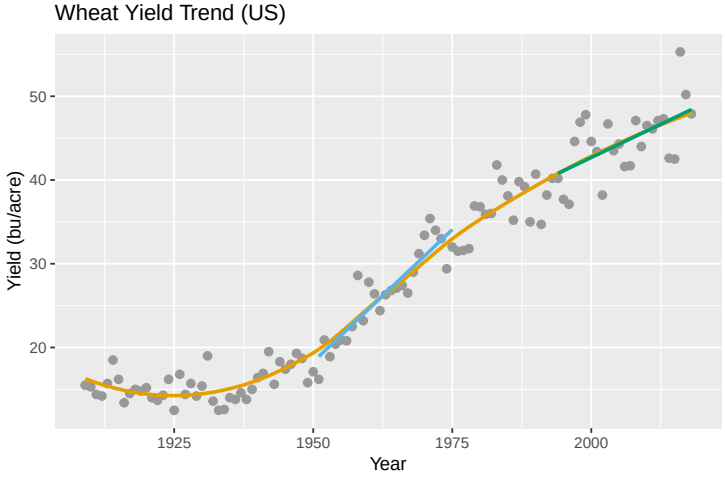
Appendix

Context

Why do we need to reduce standard deviation? Can we just work for larger treatment differences?

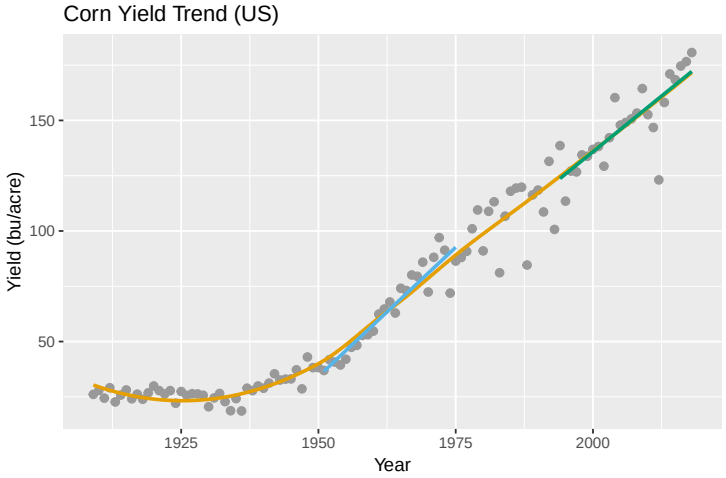
```
> trend.dat <- read.csv('wwheat.yield.csv')
> ggplot(trend.dat, aes(Year,Value)) +
+ geom_point(colour = cbPalette[1],size=2) +
```

```
+ geom_smooth(color=cbPalette[2],se = FALSE) +
+ scale_colour_manual(values=cbPalette) +
+   labs( x='Year', y='Yield (bu/acre)', title = "Wheat Yield Trend (US)") +
+   geom_smooth(data=trend.dat[trend.dat$Year %in% 1951:1975,], method = lm,color=cbPalette[
+   geom_smooth(data=trend.dat[trend.dat$Year %in% 1994:2018,], method = lm,color=cbPalette[
```



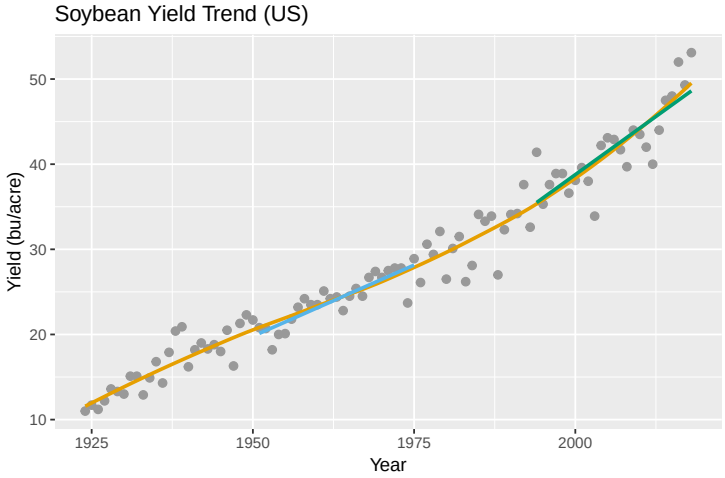

```
> lm(Value ~ Year, data=trend.dat[trend.dat$Year %in% 1951:1975,])

Call:
lm(formula = Value ~ Year, data = trend.dat[trend.dat$Year %in%
    1951:1975, ])

Coefficients:
    (Intercept)             Year
-1212.757000000      0.631307692

> lm(Value ~ Year, data=trend.dat[trend.dat$Year %in% 1994:2018,])

Call:
lm(formula = Value ~ Year, data = trend.dat[trend.dat$Year %in%
    1994:2018, ])

Coefficients:
   (Intercept)            Year
-593.157692308     0.317923077

> min.wheat <- min(trend.dat$Year)

> trend.dat <- read.csv('corn.yield.csv')
> trend.dat <- trend.dat[trend.dat$Year>=min.wheat,]
> ggplot(trend.dat, aes(Year,Value)) +
```

```
+ geom_point(colour = cbPalette[1],size=2) +
+ geom_smooth(color=cbPalette[2],se = FALSE) +
+ scale_colour_manual(values=cbPalette) +
+   labs( x='Year', y='Yield (bu/acre)', title = "Corn Yield Trend (US)") +
+   geom_smooth(data=trend.dat[trend.dat$Year %in% 1951:1975,], method = lm,color=cbPalette[
+   geom_smooth(data=trend.dat[trend.dat$Year %in% 1994:2018,], method = lm,color=cbPalette[
```



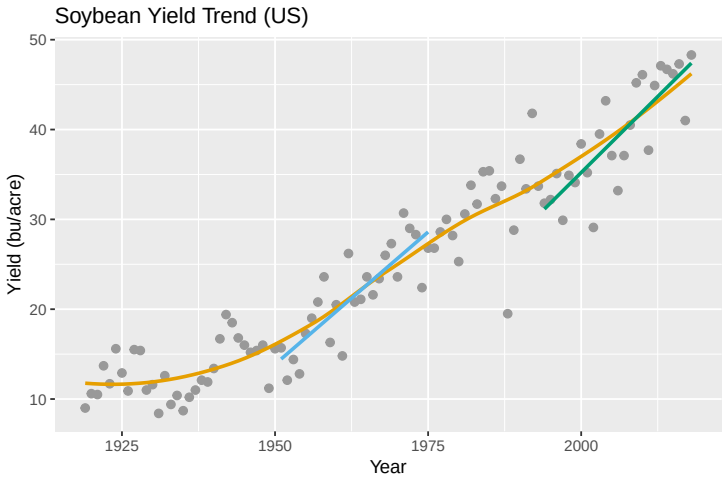

```
> lm(Value ~ Year, data=trend.dat[trend.dat$Year %in% 1951:1975,])

Call:
lm(formula = Value ~ Year, data = trend.dat[trend.dat$Year %in%
    1951:1975, ])

Coefficients:
   (Intercept)            Year
-4521.58000000      2.33630769

> lm(Value ~ Year, data=trend.dat[trend.dat$Year %in% 1994:2018,])

Call:
lm(formula = Value ~ Year, data = trend.dat[trend.dat$Year %in%
    1994:2018, ])

Coefficients:
   (Intercept)            Year
-3896.93553846      2.01638462

> trend.dat <- read.csv('soybean.yield.csv')
> trend.dat <- trend.dat[trend.dat$Year>=min.wheat,]
> ggplot(trend.dat, aes(Year,Value)) +
+ geom_point(colour = cbPalette[1],size=2) +
```

```
+ geom_smooth(color=cbPalette[2],se = FALSE) +
+ scale_colour_manual(values=cbPalette) +
+   labs( x='Year', y='Yield (bu/acre)', title = "Soybean Yield Trend (US)") +
+   geom_smooth(data=trend.dat[trend.dat$Year %in% 1951:1975,], method = lm,color=cbPalette[
+   geom_smooth(data=trend.dat[trend.dat$Year %in% 1994:2018,], method = lm,color=cbPalette[
```




```
> lm(Value ~ Year, data=trend.dat[trend.dat$Year %in% 1951:1975,])

Call:
lm(formula = Value ~ Year, data = trend.dat[trend.dat$Year %in%
    1951:1975, ])

Coefficients:
   (Intercept)            Year
-632.261000000     0.334384615

> lm(Value ~ Year, data=trend.dat[trend.dat$Year %in% 1994:2018,])

Call:
lm(formula = Value ~ Year, data = trend.dat[trend.dat$Year %in%
    1994:2018, ])

Coefficients:
    (Intercept)             Year
-1053.841230769      0.546307692

> trend.dat <- read.csv('swheat.yield.csv')
> trend.dat <- trend.dat[trend.dat$Year>=min.wheat,]
> ggplot(trend.dat, aes(Year,Value)) +
+ geom_point(colour = cbPalette[1],size=2) +
+ geom_smooth(color=cbPalette[2],se = FALSE) +
```

```
+ scale_colour_manual(values=cbPalette) +
+   labs( x='Year', y='Yield (bu/acre)', title = "Soybean Yield Trend (US)") +
+   geom_smooth(data=trend.dat[trend.dat$Year %in% 1951:1975,], method = lm,color=cbPalette[
+   geom_smooth(data=trend.dat[trend.dat$Year %in% 1994:2018,], method = lm,color=cbPalette[
```




```
> lm(Value ~ Year, data=trend.dat[trend.dat$Year %in% 1951:1975,])

Call:
lm(formula = Value ~ Year, data = trend.dat[trend.dat$Year %in%
    1951:1975, ])

Coefficients:
    (Intercept)             Year
-1134.985000000      0.589153846

> lm(Value ~ Year, data=trend.dat[trend.dat$Year %in% 1994:2018,])

Call:
lm(formula = Value ~ Year, data = trend.dat[trend.dat$Year %in%
    1994:2018, ])

Coefficients:
(Intercept)         Year
  -1318.790        0.677
```

Bibliography

[CC57] William G Cochran and Gertrude M Cox. *Experimental Design*, volume 2. John Wiley and Sons, Inc., May 1957.

[GG84] Kwanchai A Gomez and Arturo A Gomez. Statistical Procedures for Agricultural Research, 1984.

[Kue00] R O Kuehl. *Design of experiments: statistical principles of research design and analysis* . Brooks/Cole, 2 edition, 2000.

[MJ02] George A Milliken and Dallas E Johnson. *Analysis of Messy Data*, volume Volume III of *Analysis of Covariance*. Chapman and Hall/CRC, 2002.

[ST60] Robert G D Steel and James H Torrie. *Principles and Procedures of Statistics, A Biometrical Approach*. McGraw-Hill, second edition, 1960.