Thoughts about Power and Sample Size for On-Farm Strip Trials

November 5, 2018

Abstract

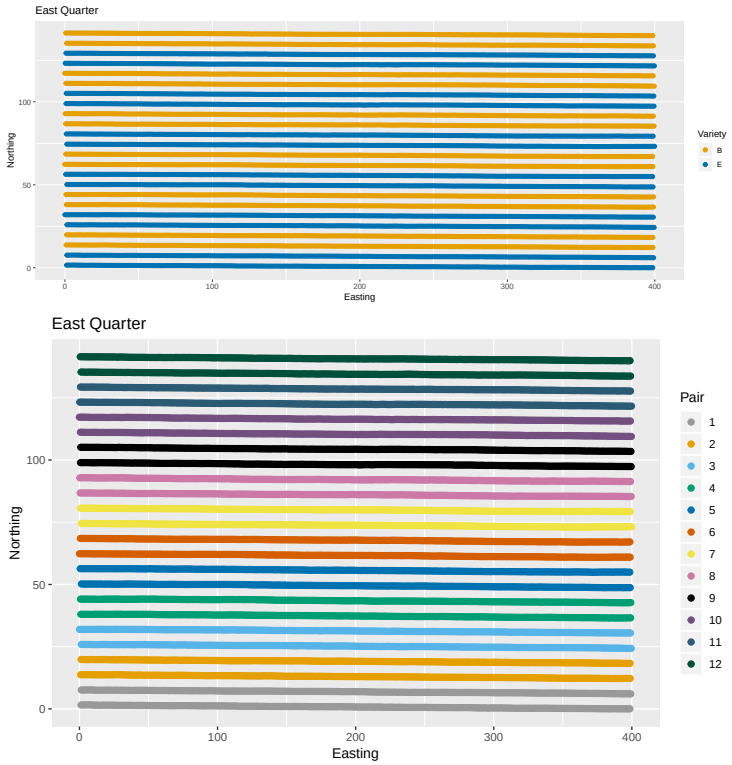
Experimentalists are frequently introduced to the design of experiments with a discussion of statistical power and sample size calculations. This discussion starts with a simple formula based on scalar (point) estimates of means and standard deviation and leads to planning replication of treatments over experimental units.

How do we extend power and sample size calculations to strip trials? When the experimental design includes strips in a field, we no longer have point estimates of the effect of treatments on experimental units; we instead can model response as a one-dimensional function of position in the strip, or a two-dimensional function of a point in space. We no longer model standard deviation as the expected variance of scalar random variable from the point estimate of the mean; instead, we can consider variation as dependent on location over both time and space.

Using examples of analysis of past on-farm strip trials, we discuss how understanding analysis methods beyond point estimation (i.e. time-series, geospatial statistics or functional data analysis) can guide planning for future on-farm trials.

1 Introduction

Consider on an-farm variety trial, with two corn varieties planted in strips. The central portion of the field was planted with a split-planter, with a planter twice the width of the cobmine used for harvest. Several varieties were planted **??**; there are 24 harvested strips in the middle of the field that are suitable for a one-way analysis of variance. We will consider each pair of harvested strips as a single expermental unit. For simplicity, we will ignore pairs of planted strips as blocks, and instead start our analysis of this data as a Completely Random Design with two treatments and 6 experiment units (strips) per treatment.



2 Simple Univariate Analysis

As a starting point, first we consider each strip as a single experimental unit. We use the model

$$y_{ij} = \mu_i + e_{ij} \tag{1}$$

for a number of treatments $i = 1, \ldots, I$ and number of strips $j = 1, \ldots, N_j$. We calculate mean and standard deviation

$$\widehat{\mu}_i = \frac{\sum_{j=1}^{N_i} y_{ij}}{N_i} \tag{2}$$

$$\widehat{\sigma}_i^2 = \frac{\sum_{j=1}^{N_i} (y_{ij} - \widehat{\mu}_i)^2}{N_i} \tag{3}$$

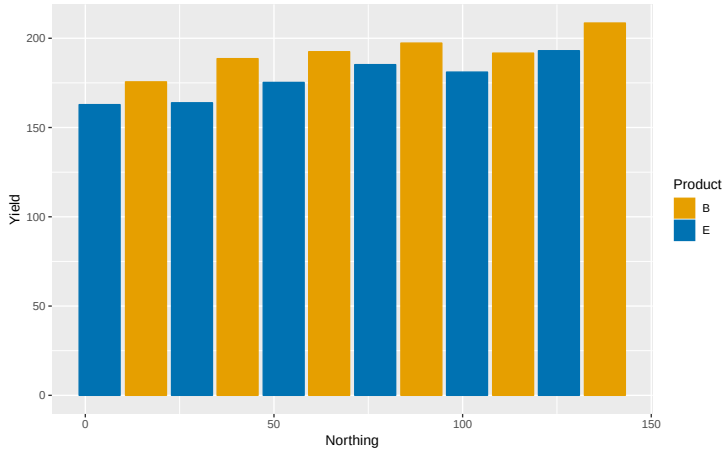
It suffices at this stage to calculate an average yield for each strip as a measure of y_{ij} - that is, we consider each individual yield estimate y_{ijk} from the yield map as a sample from y_{ij} and let

$$y_{ij} = \bar{y}_{ij.} = \frac{\sum_{k=1}^{K} y_{ijk}}{K} \tag{4}$$

```
> means.dat <- with(EastQuarter.dat, data.frame(
+   Pass = aggregate(Yield ~ Pair,EastQuarter.dat,mean,na.rm=TRUE)[,1],
+   Yield = aggregate(Yield ~ Pair,EastQuarter.dat,mean,na.rm=TRUE)[,2],
+   Product = aggregate(as.numeric(Product) ~ Pair,EastQuarter.dat,mean,na.rm=TRUE)[,2],
+   Northing = aggregate(Northing ~ Pair,EastQuarter.dat,mean,na.rm=TRUE)[,2]
+ ))
> means.dat
```

	Pass	Yield	Product	Northing
1	1	162.8279	2	3.787845
2	2	175.5724	1	16.082465
3	3	163.8603	2	28.274649
4	4	188.5818	1	40.354902
5	5	175.2685	2	52.571117
6	6	192.4624	1	64.626749
7	7	185.2256	2	76.832520
8	8	197.2943	1	89.062328
9	9	181.0466	2	101.170738
10	10	191.6723	1	113.383573
11	11	193.0240	2	125.521797
12	12	208.5366	1	137.705798

```
> means.dat$Product <- levels(EastQuarter.dat$Product)[means.dat$Product]
```



We might then proceed to a simple paired t-test, assuming independent populations, and a pooled standard deviation and a equal number of strips for each treatment. We let δ denote the difference between means $\widehat{\mu}_1 - \widehat{\mu}_2$, and we use a t-test of

$$t = \frac{\delta}{\sqrt{2\sigma^2/n}} \tag{5}$$

```
> print(product.means <- tapply(means.dat$Yield,list(means.dat$Product),mean,na.rm=TRUE))
```

```
       B        E
192.3533 176.8755

> print(product.sd <- tapply(means.dat$Yield,list(means.dat$Product),sd,na.rm=TRUE))

       B        E
10.79454 11.97669

> print(pooled.sd <- mean(product.sd))

[1] 11.38561

> print(delta <- abs(product.means[1]-product.means[2]))

       B
15.47783

> print(pooled.t <- delta / (sqrt((2*pooled.sd^2)/6)))

       B
2.354584
```

Assuming a two-sided test, we can compute a critical t and a $Pr(> t)$ by

```
> qt(1-0.05/2,10)

[1] 2.228139

> 2*(1-pt(pooled.t,10))

         B
0.04032411
```

3 Functional Analysis

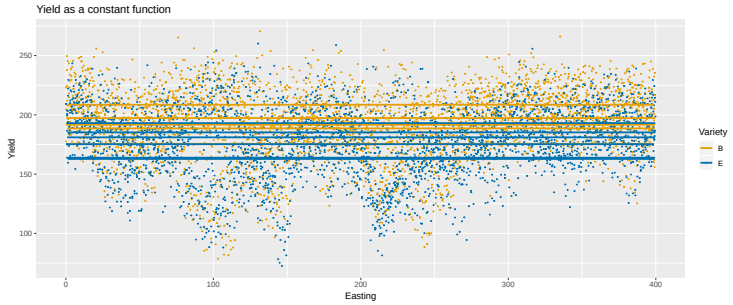
In this case, each strip represents roughly 500 observations over a range of surface conditions, and we might be interested in how different varieties respond to variations in soil or topography. Thus, a simple univariate analysis is unsatisfying. We can extend the simple case to include spatial information found along the length of the strip by modeling each strip as a fuction over position.

Let d denote distance along the East-West axis (*Easting*). We then let y_{ij} be a function of d, and

$$\widehat{\mu}_i(d) = \frac{\sum_{j=1}^{N_i} y_{ij}(d)}{N_i} \tag{6}$$

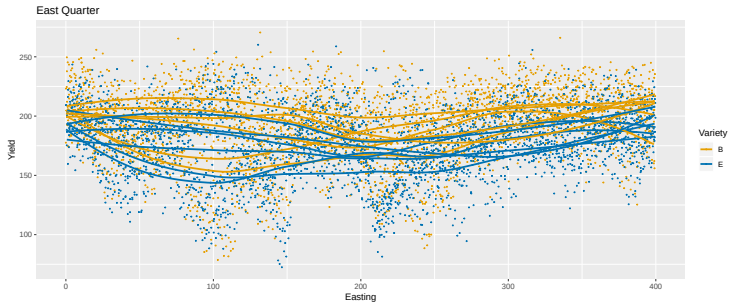
$$\widehat{\sigma}_i^2(d) = \frac{\sum_{j=1}^{N_i} (y_{ij(d)} - \widehat{\mu}_i(d))^2}{N_i} \tag{7}$$

When we let $y_{ij}(d) = \widehat{\mu}_i$, then $\widehat{\mu}_i(d)$ is a constant function returning the strip mean at every d along the length of the field:



3.1 Yield Function

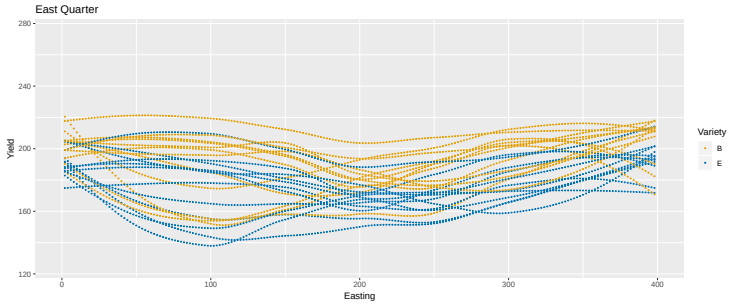
Assume that most error is sampling error, so we produce LOESS smoothed curves.



Create a data frame of passes, one observation for every 2 meters, excluding ends

```
> LOESS.basis <- rep(seq(2,398,by=2))
> total.passes <- max(EastQuarter.dat$PassNo)
> Smoothed.dat <- data.frame(
+   Yield = rep(0,total.passes*length(LOESS.basis)),
+   Easting = rep(LOESS.basis,total.passes),
+   PassNo = rep(1:total.passes,each=length(LOESS.basis)),
+   Pair = rep(0,total.passes*length(LOESS.basis)),
+   Product = rep(levels(EastQuarter.dat$Product),total.passes*length(LOESS.basis)/2),
+   Northing = rep(0,total.passes*length(LOESS.basis))
+ )
> #for(i in 1:max(EastQuarter.dat$PassNo)) {
> for(i in 1:total.passes) {
+   current.pass <- EastQuarter.dat[EastQuarter.dat$PassNo==i,]
+   current.loess <- loess(Yield ~ Easting,data=current.pass)
+   current.smoothed <- predict(current.loess, data.frame(Easting = LOESS.basis))
+   Smoothed.dat$Yield[Smoothed.dat$PassNo==i] <- current.smoothed
+   Smoothed.dat$Pair[Smoothed.dat$PassNo==i] <- current.pass$Pair[1]
```

```
+   Smoothed.dat$Product[Smoothed.dat$PassNo==i] <- current.pass$Product[1]
+   Smoothed.dat$Northing[Smoothed.dat$PassNo==i] <- mean(current.pass$Northing)
+ }
> Smoothed.dat$Pass <- as.factor(Smoothed.dat$PassNo)
```



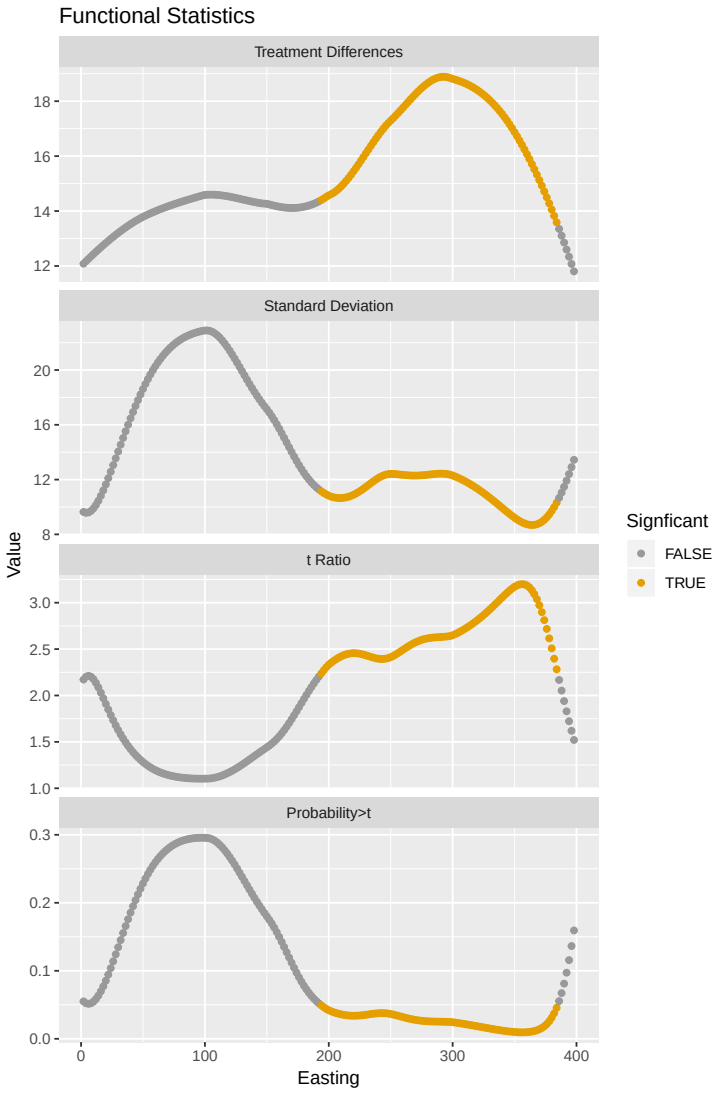
3.2 Functional t tests

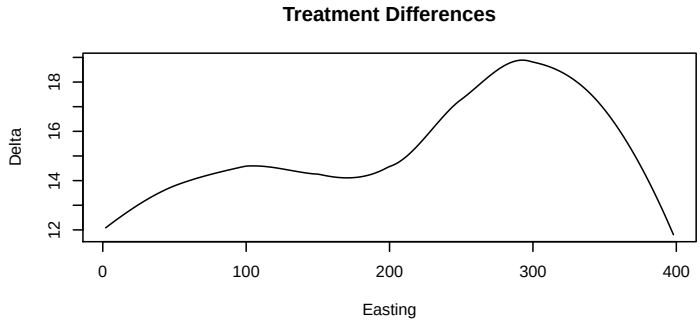
Given that μ and σ are functions, we can write a functional t-test as

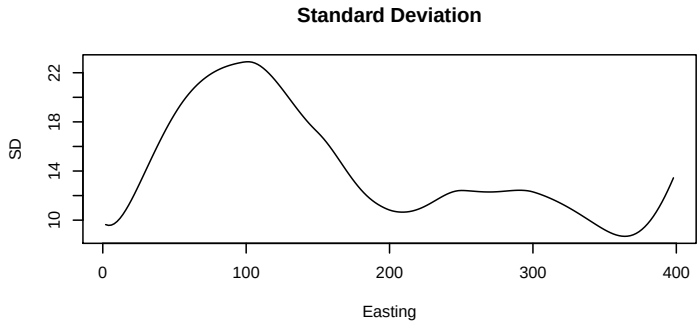
$$t(d) = \frac{\delta(d)}{\sqrt{2\sigma(d)^2/n}} \tag{8}$$

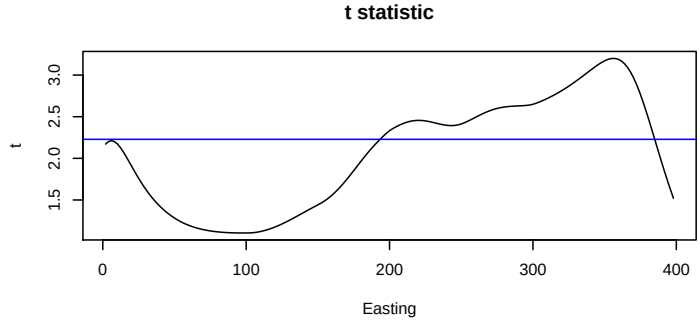
where $\delta(t) = abs(\mu_1(d) - \mu_2(d))$.
We compute this using LOESS basis points as d.

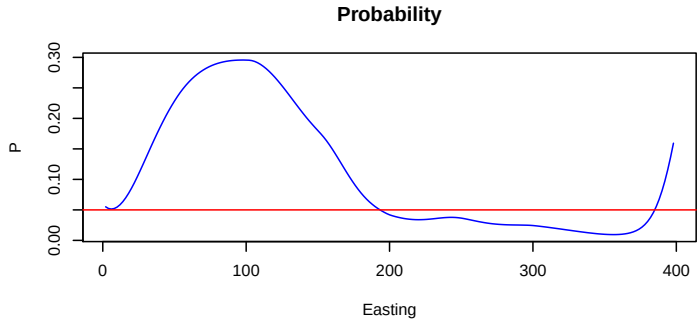
```
> TTests <- data.frame(
+   Easting = LOESS.basis
+ )
> for(i in 1:length(LOESS.basis)) {
+   l <- LOESS.basis[i]
+   obs <- Smoothed.dat[Smoothed.dat$Easting==l,]
+   product.means <- tapply(obs$Yield,list(obs$Product),mean,na.rm=TRUE)
+   product.sd <- tapply(obs$Yield,list(obs$Product),sd,na.rm=TRUE)
+   TTests$Delta[i] <- product.means[1]-product.means[2]
+   TTests$SD[i] <- sqrt(sum(product.sd^2)/2)
+ }
> TTests$t <- TTests$Delta/(sqrt((2*TTests$SD^2)/6))
> TTests$P <- 2*(1-pt(TTests$t,10))
```


Functional Statistics
Treatment Differences
Standard Deviation
t Ratio
Probability>t
Value
Easting
Signficant
FALSE
TRUE


Treatment Differences
Delta
12
14
16
18
0
100
200
300
400
Easting


Standard Deviation
SD
10
14
18
22
0
100
200
300
400
Easting


t statistic
t
1.5
2.0
2.5
3.0
0
100
200
300
400
Easting


Probability
P
0.00
0.10
0.20
0.30
0
100
200
300
400
Easting

4 Planning

Suppose we have a field where we might test strips, and that we have yield monitor data. Remember our t statistic,

$$t_{\alpha/2,\nu} = \frac{\delta}{\sqrt{2\sigma^2/n}} \tag{9}$$

Skpping some of the math, to detect and declare significant a difference of δ, we require

$$n \geq \frac{\sigma^2}{\delta, \nu}^2 (t_{\alpha/2} - t_{\beta}, \nu)^2$$

where $t_{\beta,\nu}$ is the critical value to avoid a Type II error.

Since t are dependent on degrees of freedom, $\nu = 2*(n-1)$, when estimating n it is sometimes more convienient to use the z score instead, and normalize σ and δ by dividing by the mean μ.

Similarly, if we have a specific number of replicates, what difference can we hope to detect, given a constant σ and n

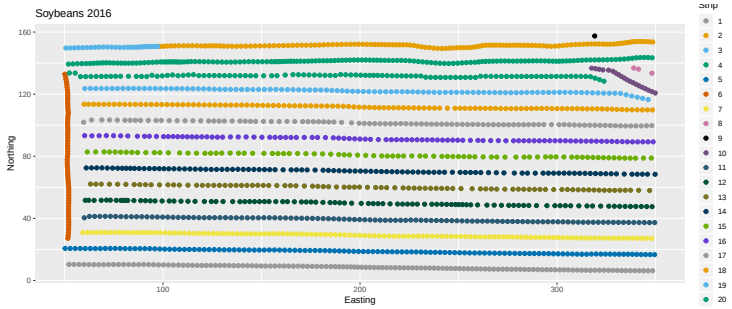
$$\delta = \sigma^2(t_{\alpha/2} - t_{\beta})/\sqrt{n}$$

To illustrate, we start with field of soybeans. We trim the east and west ends from the field - these contain the endrows that are not analyzable

```
> load(file="../../2017/Workshop/Case Study 1/Pooled.Rda")
> uniformity.dat <- subset(Soybean2016.dat,Soybean2016.dat$Easting>50)
> uniformity.dat <- subset(uniformity.dat,uniformity.dat$Easting<350)
```

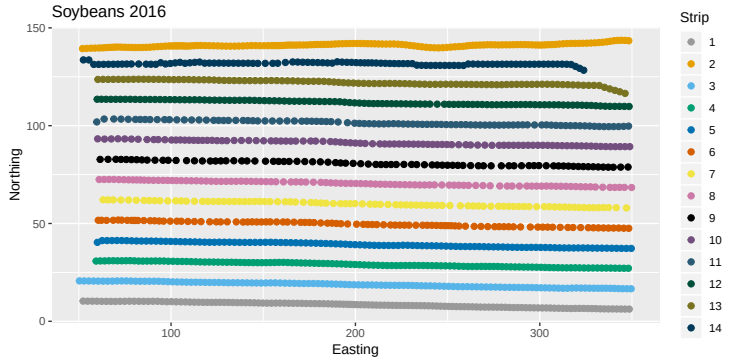
The harvest data does not explicitly identify strips in the field, so we need tag these manually.

```
> uniformity.dat$Gap <- c(0,uniformity.dat$Seconds[2:dim(uniformity.dat)[1]] -uniformity.dat
> hist(uniformity.dat$Gap)
> uniformity.dat$PassNo <- 0
> Pass <- 1
> for(i in 1:dim(uniformity.dat)[1]) {
+   if(uniformity.dat$Gap[i]>5) {
+     Pass <- Pass+1
+   }
+   uniformity.dat$PassNo[i] <- Pass
+ }
```



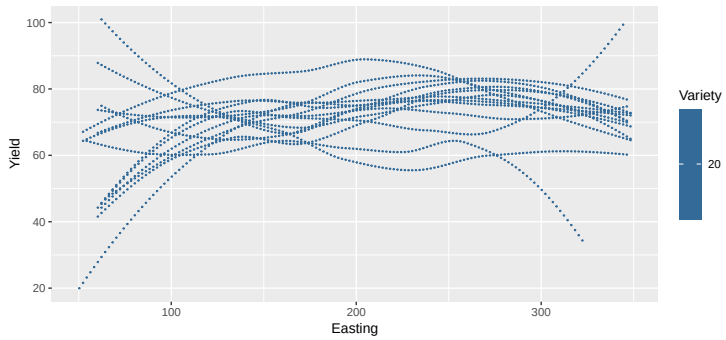
There are some short or incomplete harvest passes, we'll remove those as well.

```
> uniformity.dat <- uniformity.dat[!(uniformity.dat$PassNo %in% c(2,3,6,8,9,10)),]
```



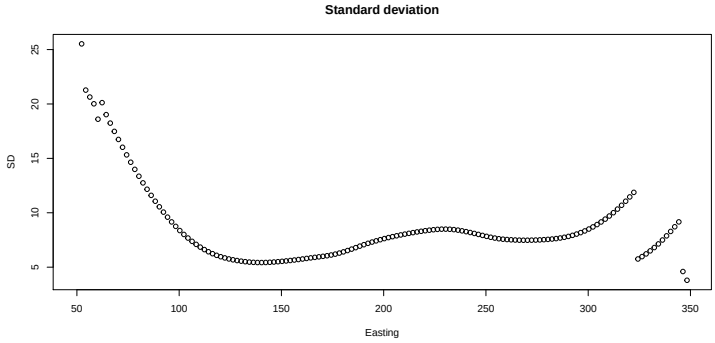
Repeat the interpolation for yield functions as from earlier.

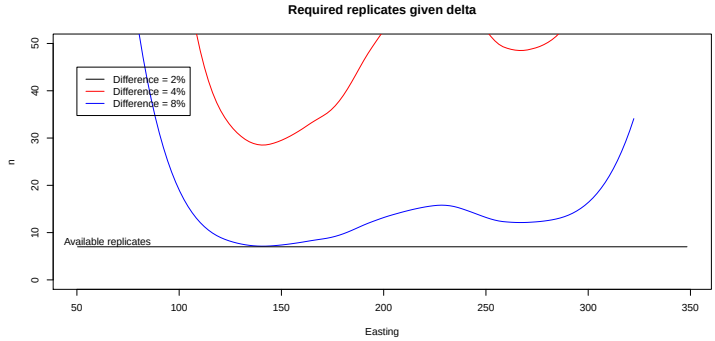
```
> LOESS.basis <- rep(seq(min(uniformity.dat$Easting),max(uniformity.dat$Easting),by=2))
> total.passes <- length(levels(uniformity.dat$Pass))
> SmoothedU.dat <- data.frame(
+   Yield = rep(0,total.passes*length(LOESS.basis)),
+   Easting = rep(LOESS.basis,total.passes),
+   PassNo = rep(levels(uniformity.dat$Pass),each=length(LOESS.basis)),
+   Northing = rep(0,total.passes*length(LOESS.basis))
+ )
> for(i in levels(uniformity.dat$Pass)) {
+   current.pass <- uniformity.dat[uniformity.dat$Pass==i,]
+   current.loess <- loess(Yield ~ Easting,data=current.pass)
+   current.smoothed <- predict(current.loess, data.frame(Easting = LOESS.basis))
+   SmoothedU.dat$Yield[SmoothedU.dat$PassNo==i] <- current.smoothed
+   SmoothedU.dat$Northing[SmoothedU.dat$PassNo==i] <- mean(current.pass$Northing)
+ }
```

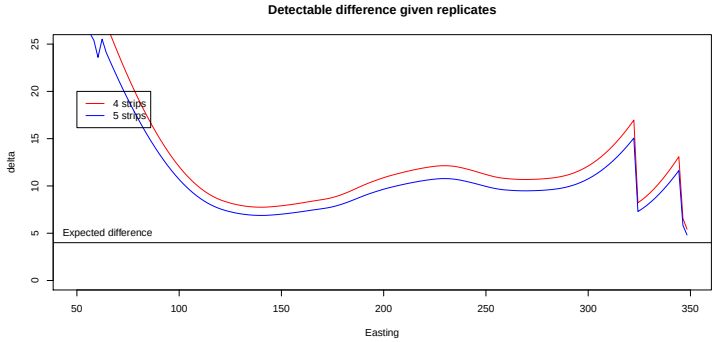


Now calculate n and δ, given the measure σ across the field.

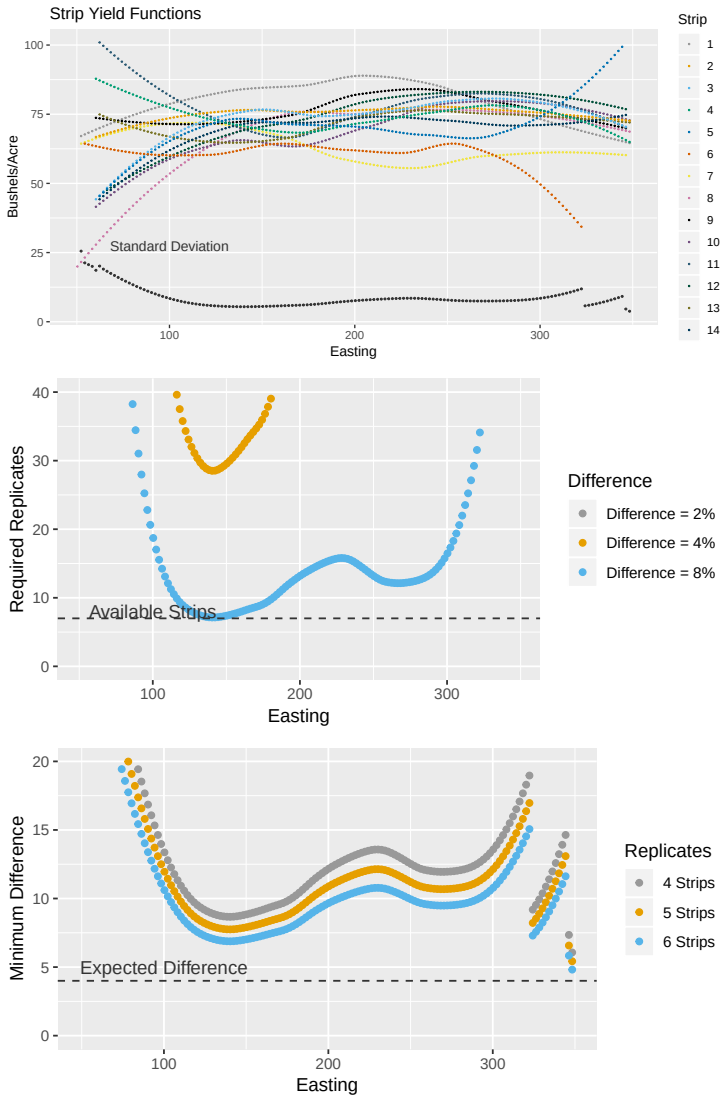
```
> Power <- data.frame(
+   Easting = LOESS.basis
+ )
> for(i in 1:length(LOESS.basis)) {
+   l <- LOESS.basis[i]
+   obs <- SmoothedU.dat[SmoothedU.dat$Easting==l,]
+   Power$SD[i] <- sd(obs$Yield,na.rm=TRUE)
+   Power$CV[i] <- 100*Power$SD[i]/mean(obs$Yield)
+   Power$n[i] <- length(!is.na(obs$Yield))/2
+ }
> Power$D1 <- (Power$CV/2)^2 * (qnorm(1-0.05/2)-qnorm(0.2))^2
> Power$D4 <- (Power$CV/4)^2 * (qnorm(1-0.05/2)-qnorm(0.2))^2
> Power$D8 <- (Power$CV/8)^2 * (qnorm(1-0.05/2)-qnorm(0.2))^2
> Power$R4 =  Power$SD*(qt(1-0.05/2,((5-1)*2))-qt(0.2,((5-1)*2)))/sqrt(4)
> Power$R5 =  Power$SD*(qt(1-0.05/2,((5-1)*2))-qt(0.2,((5-1)*2)))/sqrt(5)
> Power$R6 =  Power$SD*(qt(1-0.05/2,((6-1)*2))-qt(0.2,((6-1)*2)))/sqrt(6)
```





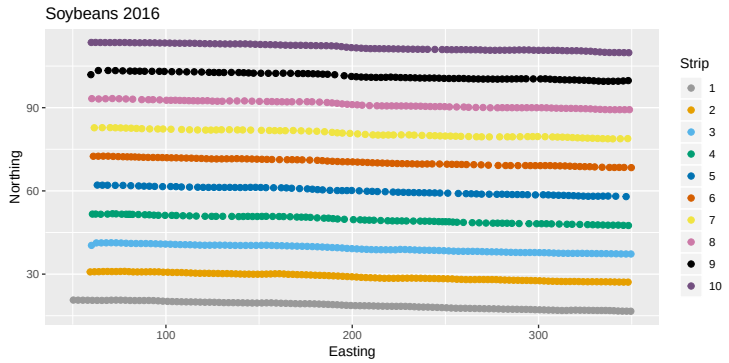



```
> annotate.color="#333333"
```

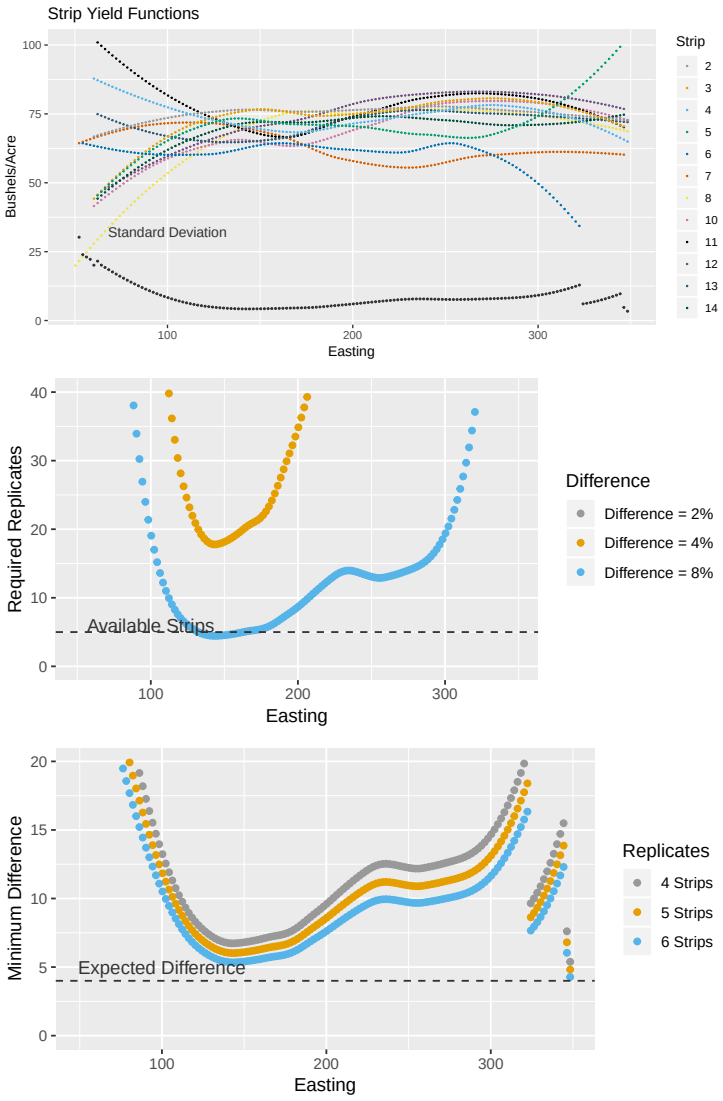


Examining the strips, the outer strips tend to be much different than the strips from the center of the field. If we restrict a strip trial to the center of the field, we can reduce variance between strips.

```
> SmoothedU.dat <- SmoothedU.dat[!(SmoothedU.dat$PassNo %in% c(1,4,19,20)),]
> uniformity.dat <- uniformity.dat[!(uniformity.dat$PassNo %in% c(1,4,19,20)),]
```



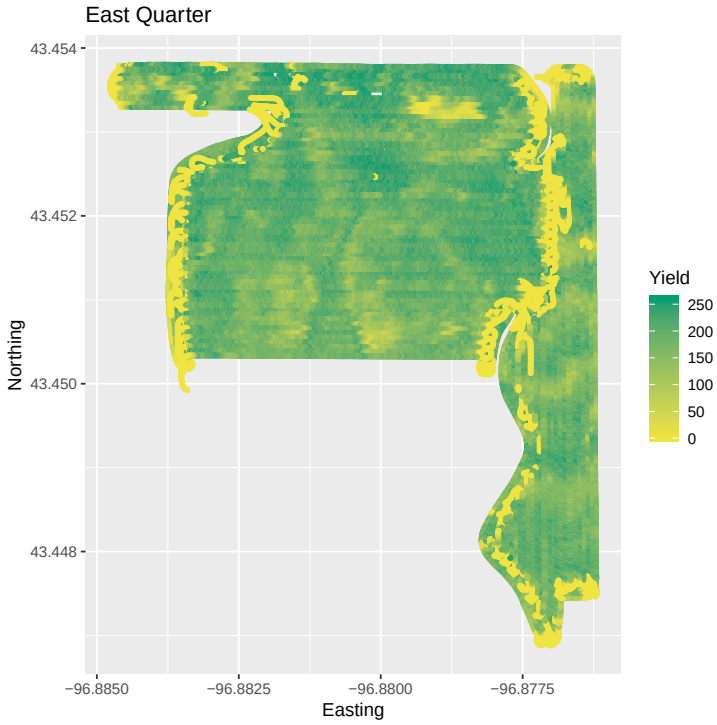

```
> #for(i in 1:length(LOESS.basis)) {
> #  l <- LOESS.basis[i]
> #  obs <- SmoothedU.dat[SmoothedU.dat$Easting==l,]
> #  Power$SD[i] <-sd(obs$Yield,na.rm=TRUE)
> #  Power$n[i] <- length(!is.na(obs$Yield))/2
> #}
> #Power$D1 <- Power$SD^2 * (qnorm(1-0.05/2)-qnorm(0.2))^2
> #Power$D4 <- (Power$SD/4)^2 * (qnorm(1-0.05/2)-qnorm(0.2))^2
> #Power$D8 <- (Power$SD/8)^2 * (qnorm(1-0.05/2)-qnorm(0.2))^2
>
> #Power$R4 =  Power$SD*(qnorm(1-0.05/2)-qnorm(0.2))/sqrt(4)
> #Power$R5 =  Power$SD*(qnorm(1-0.05/2)-qnorm(0.2))/sqrt(5)
> for(i in 1:length(LOESS.basis)) {
+   l <- LOESS.basis[i]
+   obs <- SmoothedU.dat[SmoothedU.dat$Easting==l,]
+   Power$SD[i] <- sd(obs$Yield,na.rm=TRUE)
+   Power$CV[i] <- 100*Power$SD[i]/mean(obs$Yield)
+   Power$n[i] <- length(!is.na(obs$Yield))/2
+ }
> Power$D1 <- (Power$CV/2)^2 * (qnorm(1-0.05/2)-qnorm(0.2))^2
> Power$D4 <- (Power$CV/4)^2 * (qnorm(1-0.05/2)-qnorm(0.2))^2
> Power$D8 <- (Power$CV/8)^2 * (qnorm(1-0.05/2)-qnorm(0.2))^2
> Power$R4 =  Power$SD*(qt(1-0.05/2,((5-1)*2))-qt(0.2,((5-1)*2)))/sqrt(4)
> Power$R5 =  Power$SD*(qt(1-0.05/2,((5-1)*2))-qt(0.2,((5-1)*2)))/sqrt(5)
> Power$R6 =  Power$SD*(qt(1-0.05/2,((6-1)*2))-qt(0.2,((6-1)*2)))/sqrt(6)
```

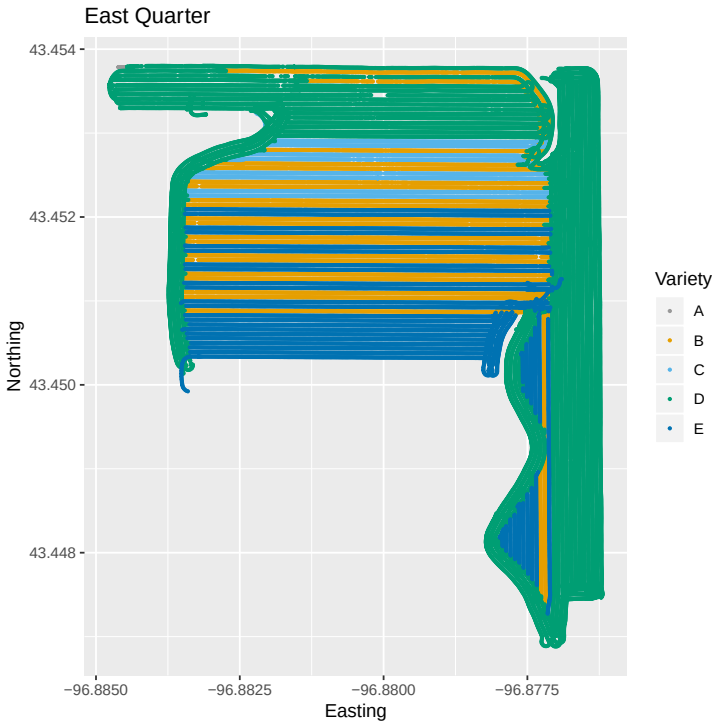

Strip Yield Functions
Bushels/Acre
Easting
Standard Deviation
Strip
2
3
4
5
6
7
8
10
11
12
13
14
Required Replicates
Available Strips
Difference
Difference = 2%
Difference = 4%
Difference = 8%
Minimum Difference
Expected Difference
Replicates
4 Strips
5 Strips
6 Strips

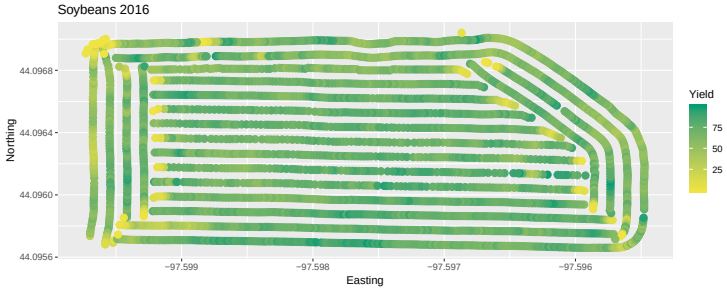
5 Appendix

5.1 Yield Trial Data

For reference, the raw trial maps with both Yield and Product.




East Quarter
43.454
43.452
43.450
43.448
Northing
−96.8850
−96.8825
−96.8800
−96.8775
Easting
Variety
A
B
C
D
E


Soybeans 2016
44.0968
44.0964
44.0960
44.0956
Northing
−97.599
−97.598
−97.597
−97.596
Easting
Yield
75
50
25

5.2 Sampling Error

Since we have an uneven number of samples per strip, we may wish to include a sampling error along with expperimental error. Let the variance of the difference between means be

$$\sigma_d^2 = 2[\frac{\sigma_s^2}{nr} + \frac{\sigma^2}{r}]$$

In the simple case, we can compute

```
> means.dat$Samples <- aggregate(Yield ~ Pair,EastQuarter.dat,length)[,2]
> means.dat$SampleSD <- aggregate(Yield ~ Pair,EastQuarter.dat,sd,na.rm=TRUE)[,2]
> means.dat$SampleSE <- sqrt(means.dat$SampleSD/means.dat$Samples)
> means.dat

   Pass    Yield Product   Northing Samples SampleSD  SampleSE
1     1 162.8279       E   3.787845     518 29.83083 0.2399760
2     2 175.5724       B  16.082465     527 39.67935 0.2743955
3     3 163.8603       E  28.274649     519 32.22227 0.2491692
4     4 188.5818       B  40.354902     526 30.38731 0.2403551
5     5 175.2685       E  52.571117     530 27.12060 0.2262099
6     6 192.4624       B  64.626749     528 24.31691 0.2146037
7     7 185.2256       E  76.832520     535 23.45168 0.2093679
8     8 197.2943       B  89.062328     530 23.57872 0.2109221
9     9 181.0466       E 101.170738     529 26.00689 0.2217259
10   10 191.6723       B 113.383573     542 24.83957 0.2140782
11   11 193.0240       E 125.521797     551 25.16112 0.2136924
12   12 208.5366       B 137.705798     534 20.67455 0.1967648

> means.lm <- lm(Yield ~ Product, data=means.dat)
> subsample.lm <- lm(Yield ~ Product + Product:Pair, data=EastQuarter.dat)
> summary(aov(subsample.lm))

              Df  Sum Sq Mean Sq F value Pr(>F)
Product        1  372714  372714   486.9 <2e-16 ***
Product:Pair  10  690476   69048    90.2 <2e-16 ***
Residuals   6357 4866275     765
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1

> library(lme4)
> subsample.lmer <- lmer(Yield ~ Product + (1 | Product:Pair), data=EastQuarter.dat)
> summary(subsample.lmer)

Linear mixed model fit by REML ['lmerMod']
Formula: Yield ~ Product + (1 | Product:Pair)
   Data: EastQuarter.dat

REML criterion at convergence: 60410.1
```

```
Scaled residuals:
    Min      1Q  Median      3Q     Max
-3.5092 -0.6371  0.0795  0.6919  3.0188

Random effects:
 Groups       Name        Variance Std.Dev.
 Product:Pair (Intercept) 128.5    11.34
 Residual                 765.5    27.67
Number of obs: 6369, groups:  Product:Pair, 12

Fixed effects:
            Estimate Std. Error t value
(Intercept)  192.354      4.654  41.327
ProductE     -15.476      6.582  -2.351

Correlation of Fixed Effects:
         (Intr)
ProductE -0.707
```

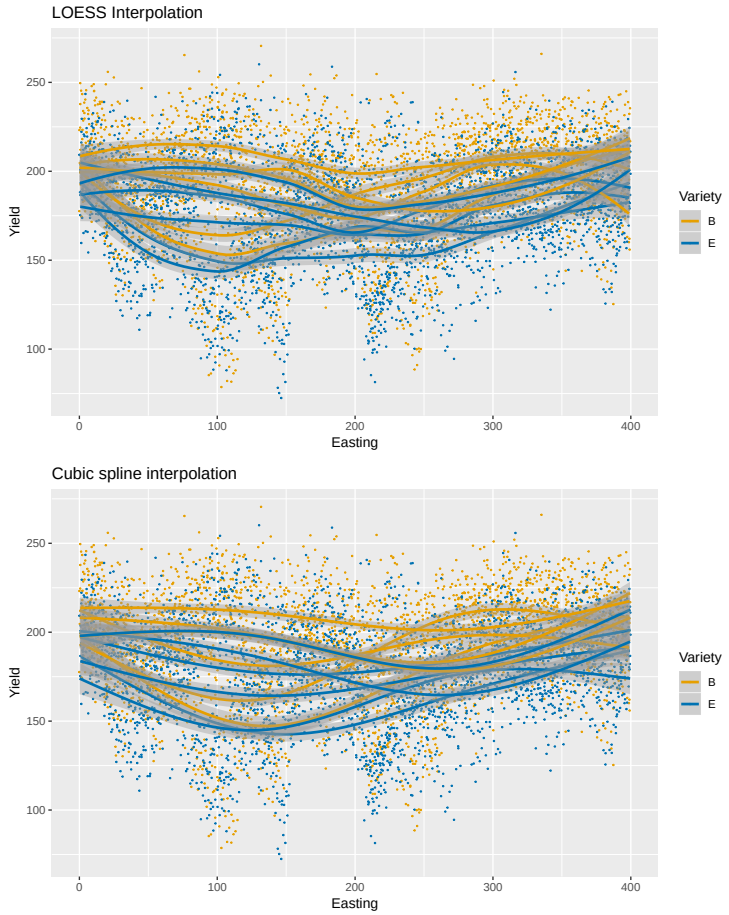
If we have n samples per experimental unit, our expected mean square for experimental error is $MSE = \sigma_s^2 + n\sigma^2$, so, solving for σ^2, then

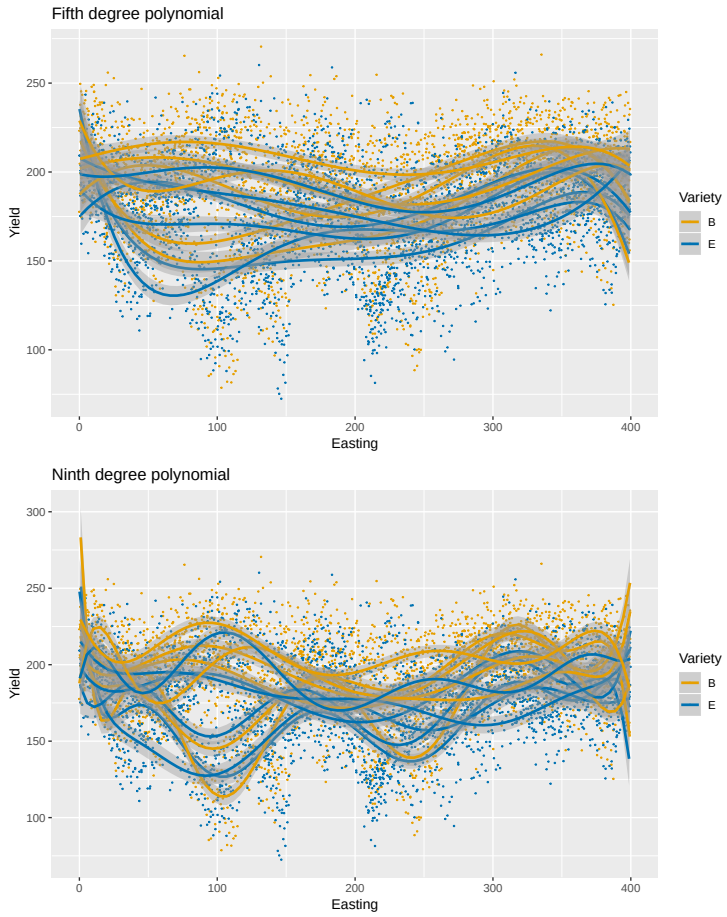
$$\sigma^2 = (MSE - MSS)/n$$

```
> (69048 - 765)/mean(means.dat$Samples)

[1] 128.6538

> library(emmeans)
> emmeans(means.lm, "Product")

 Product   emmean       SE df lower.CL upper.CL
 B       192.3533 4.654417 10 181.9826 202.7240
 E       176.8755 4.654417 10 166.5048 187.2462

Confidence level used: 0.95

> emmeans(subsample.lmer, "Product")

 Product   emmean       SE  df asymp.LCL asymp.UCL
 B       192.3537 4.654440 Inf  183.2312  201.4763
 E       176.8779 4.654489 Inf  167.7552  186.0005

Degrees-of-freedom method: asymptotic
Confidence level used: 0.95
```

5.3 Other Smoothing Functions

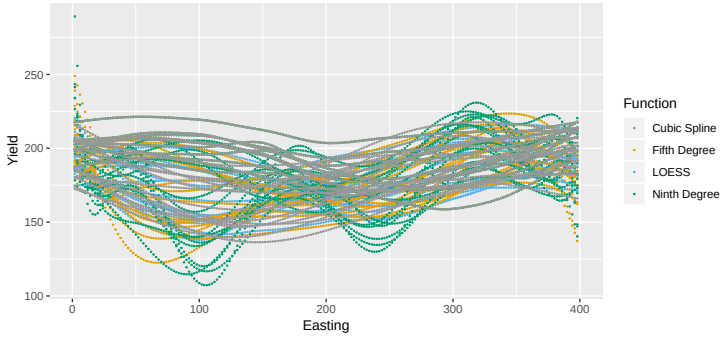
We chose LOESS as an approximating function to interpolate sampled yield to a regular grid. There are other options.

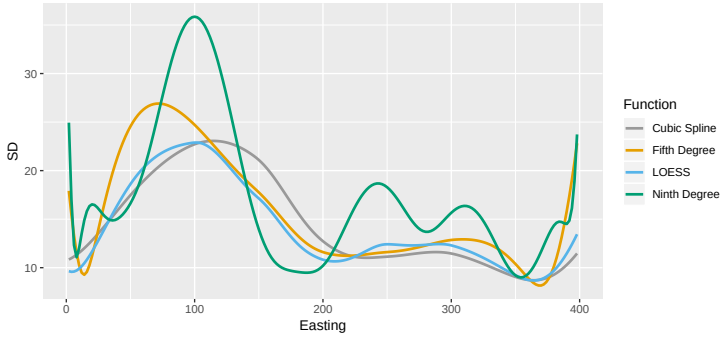


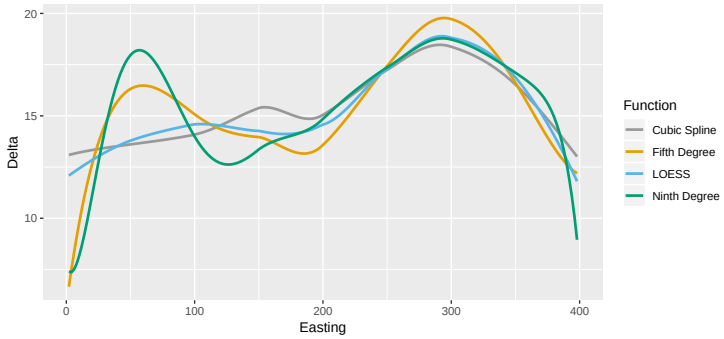


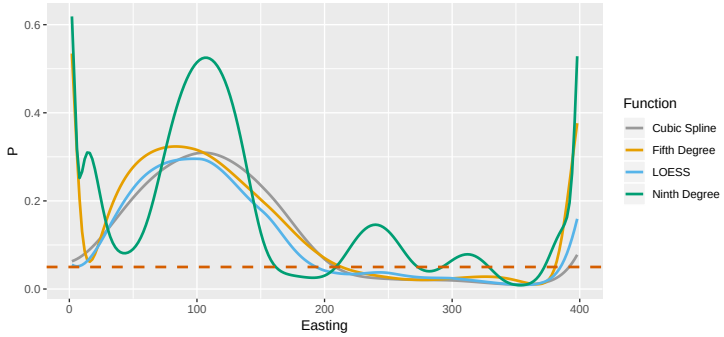

```
> LOESS.basis <- rep(seq(2,398,by=2))
> Poly5.dat <- Smoothed.dat
> Poly9.dat <- Smoothed.dat
> Spline.dat <- Smoothed.dat
> TTestsPoly5 <- TTests
> TTestsPoly9 <- TTests
> TTestsSpline <- TTests
> for(i in 1:total.passes) {
+
+   current.pass <- EastQuarter.dat[EastQuarter.dat$PassNo==i,]
+   current.poly <- lm(Yield ~ poly(Easting,5),data=current.pass)
+   current.smoothed <- predict(current.poly, data.frame(Easting = LOESS.basis))
+   Poly5.dat$Yield[Poly5.dat$PassNo==i] <- current.smoothed
+
+   current.poly <- lm(Yield ~ poly(Easting,9),data=current.pass)
+   current.smoothed <- predict(current.poly, data.frame(Easting = LOESS.basis))
```

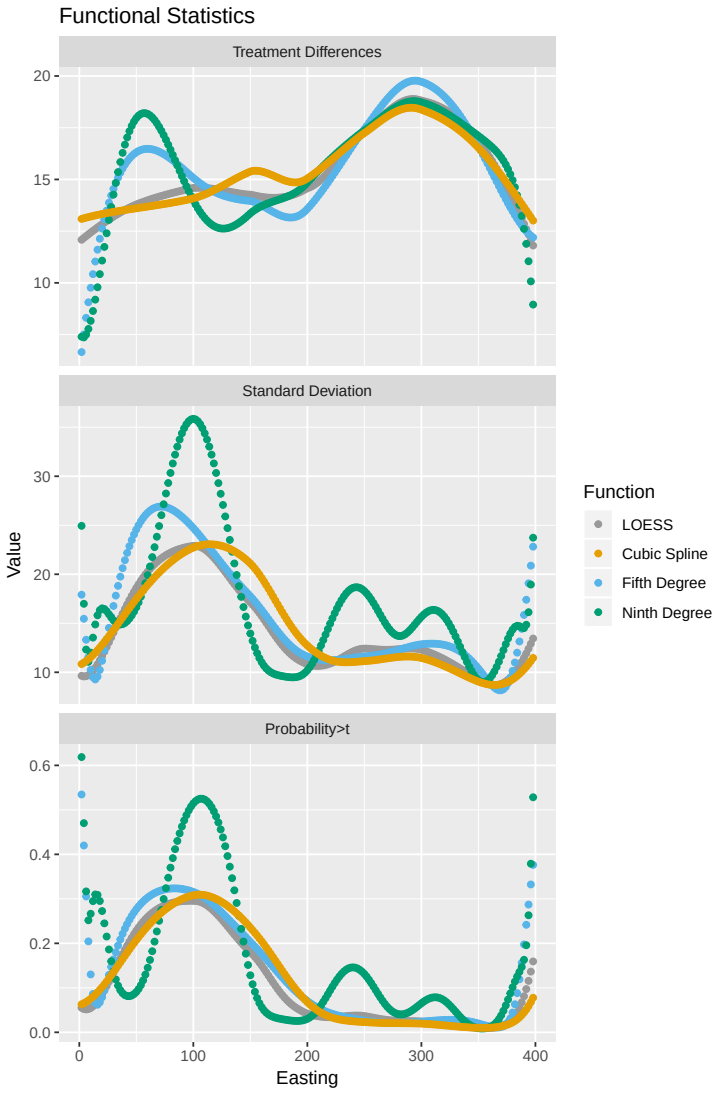
```
+   Poly9.dat$Yield[Poly9.dat$PassNo==i] <- current.smoothed
+
+   current.ns <- lm(Yield ~ ns(Easting,3),data=current.pass)
+   current.smoothed <- predict(current.ns, data.frame(Easting = LOESS.basis))
+   Spline.dat$Yield[Spline.dat$PassNo==i] <- current.smoothed
+
+ }
> for (j in 1:length(LOESS.basis)) {
+   l <- LOESS.basis[j]
+
+   obsPoly5 <- Poly5.dat[Poly5.dat$Easting==l,]
+   Poly5.means <- tapply(obsPoly5$Yield,list(obsPoly5$Product),mean,na.rm=TRUE)
+   Poly5.sd <- tapply(obsPoly5$Yield,list(obsPoly5$Product),sd,na.rm=TRUE)
+   TTestsPoly5$Delta[j] <- Poly5.means[1]-Poly5.means[2]
+   TTestsPoly5$SD[j] <- sqrt(sum(Poly5.sd^2)/2)
+
+   obs <- Poly9.dat[Poly9.dat$Easting==l,]
+   product.means <- tapply(obs$Yield,list(obs$Product),mean,na.rm=TRUE)
+   product.sd <- tapply(obs$Yield,list(obs$Product),sd,na.rm=TRUE)
+   TTestsPoly9$Delta[j] <- product.means[1]-product.means[2]
+   TTestsPoly9$SD[j] <- sqrt(sum(product.sd^2)/2)
+
+   obs <- Spline.dat[Spline.dat$Easting==l,]
+   product.means <- tapply(obs$Yield,list(obs$Product),mean,na.rm=TRUE)
+   product.sd <- tapply(obs$Yield,list(obs$Product),sd,na.rm=TRUE)
+   TTestsSpline$Delta[j] <- product.means[1]-product.means[2]
+   TTestsSpline$SD[j] <- sqrt(sum(product.sd^2)/2)
+ }

> Poly5.dat$Function <- "Fifth Degree"
> Poly9.dat$Function <- "Ninth Degree"
> Spline.dat$Function <- "Cubic Spline"
> Smoothed.dat$Function <- "LOESS"
> TTestsPoly5$Function <- "Fifth Degree"
> TTestsPoly9$Function <- "Ninth Degree"
> TTestsSpline$Function <- "Cubic Spline"
> TTests$Function <- "LOESS"
> CompSmooth.dat <- rbind(Smoothed.dat,Poly5.dat,Poly9.dat,Spline.dat)
> CompT <- rbind(TTests,TTestsPoly5,TTestsPoly9,TTestsSpline)
> CompT$t <- CompT$Delta/(sqrt((2*CompT$SD^2)/6))
> CompT$P <- 2*(1-pt(CompT$t,10))
```

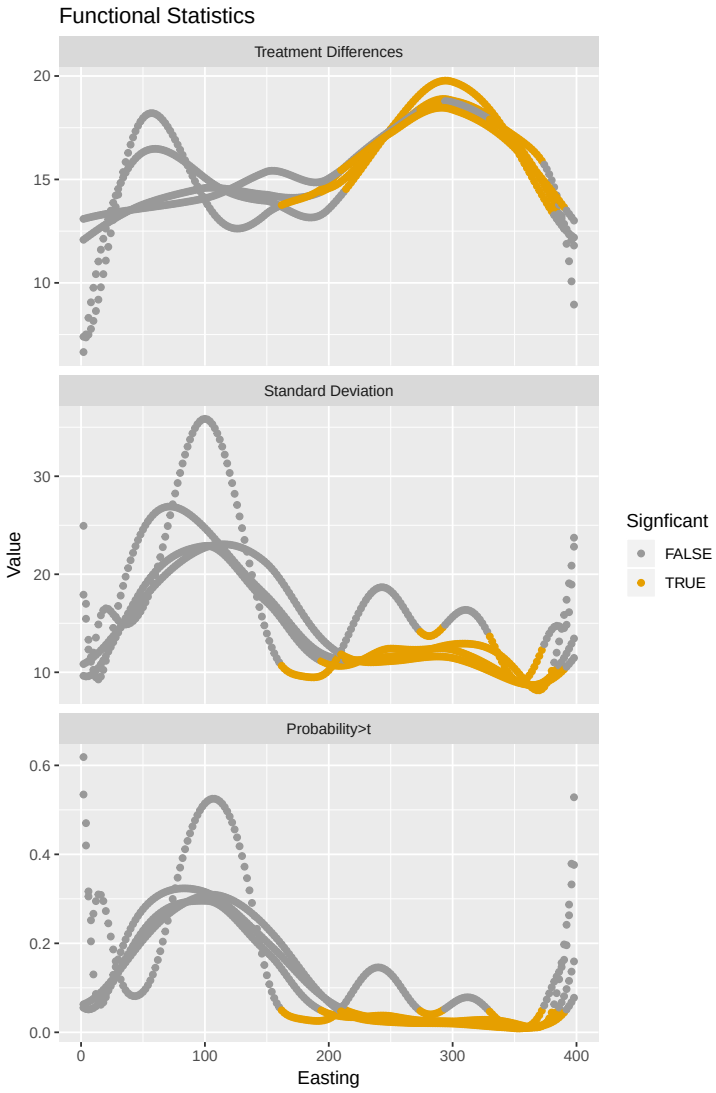

Yield
250
200
150
100
0
100
200
300
400
Easting
Function
Cubic Spline
Fifth Degree
LOESS
Ninth Degree


SD
30
20
10
0
100
200
300
400
Easting
Function
Cubic Spline
Fifth Degree
LOESS
Ninth Degree


Delta
20
15
10
0
100
200
300
400
Easting
Function
Cubic Spline
Fifth Degree
LOESS
Ninth Degree


0.6
0.4
0.2
0.0
P
0
100
200
300
400
Easting
Function
Cubic Spline
Fifth Degree
LOESS
Ninth Degree


Functional Statistics
Treatment Differences
Standard Deviation
Probability>t
Value
Easting
Function
LOESS
Cubic Spline
Fifth Degree
Ninth Degree


Functional Statistics
Treatment Differences
Standard Deviation
Probability>t
Value
Easting
Signficant
FALSE
TRUE
20
15
10
30
20
10
0.6
0.4
0.2
0.0
0
100
200
300
400

5.4 Error Terms

Our discussion to this point assumed a pooled sd; we may compute sd independently for each treatment and pooling by

$$\sigma^2 = \frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2 + \ldots (n_k - 1)s_k^2}{N - k} = \frac{\sum_i (n_i - 1)s_i^2}{N - k}$$

If we assume strips are always treated in pairs, our t statistic becomes

$$t = \frac{\delta}{\sigma/\sqrt{n}} \tag{10}$$

With two sd, we can write

$$t = \frac{\delta}{\sqrt{\sigma_1^2/n_1 + \sigma_2^2/n_2}} \tag{11}$$

```
> product.means <- tapply(means.dat$Yield,list(means.dat$Product),mean,na.rm=TRUE)
> product.means

       B        E
192.3533 176.8755

> product.sd <- tapply(means.dat$Yield,list(means.dat$Product),sd,na.rm=TRUE)
> product.sd

       B        E
10.79454 11.97669

> mean(product.sd)

[1] 11.38561

> pooled.sd <- sqrt(sum((6-1)*(product.sd^2))/(12-2))
> pooled.sd

[1] 11.40095

> delta <- product.means[1]-product.means[2]
> delta

       B
15.47783

> print(independent.t <- delta / sqrt(sum(product.sd^2/6)))

       B
2.351418

> print(pooled.t <- delta / (pooled.sd*(sqrt(2/6))))

       B
2.351418

> print(paired.t <- delta / (pooled.sd/sqrt(6)))

       B
3.325407

> 2*(1-pt(independent.t,10))

         B
0.04054249

> 2*(1-pt(pooled.t,10))

         B
0.04054249

> 2*(1-pt(paired.t,10))

          B
0.007677112
```

5.5 Permutations