Cochran 1947

Peter Claussen

10/3/2017

```
## Warning: package 'ggplot2' was built under R version 3.3.2
```

Spatial analysis of Cochran, 1947, in ARM table format:

```
arm.dat <- data.frame(
plot=c(101, 804, 103, 503, 303, 801, 302, 803, 202, 701, 403, 504, 404, 802, 102, 602, 104, 501, 304, 6
treatment=as.factor(c(1, 1, 2, 2, 3, 3, 4, 4, 5, 5, 6, 6, 7, 7, 8, 8, 9, 9, 10, 10, 11, 11, 12, 12, 13,
replicate=as.factor(c(1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1,
block=as.factor(c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
column=as.factor(c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
y=c(1, 4, 1, 1, 3, 4, 3, 4, 2, 3, 4, 1, 4, 4, 1, 2, 1, 1, 3, 2, 4, 3, 2, 3, 3, 2, 4, 2, 2, 3, 2, 1),
x=c(1, 8, 3, 7, 3, 5, 2, 7, 2, 5, 3, 8, 4, 6, 2, 6, 4, 5, 4, 8, 1, 7, 4, 6, 1, 5, 2, 7, 3, 8, 1, 6),
lat=c(3, 24, 3, 3, 17, 24, 17, 24, 10, 17, 24, 3, 24, 24, 3, 10, 3, 3, 17, 10, 24, 17, 10, 17, 17, 10,
lon=c(2, 33.5, 11, 29, 11, 20, 6.5, 29, 6.5, 20, 11, 33.5, 15.5, 24.5, 6.5, 24.5, 15.5, 20, 15.5, 33.5,
assessment1=c(1240, 640, 1770, 1410, 1460, 630, 1370, 560, 2180, 1680, 700, 1300, 810, 970, 1660, 1950,
)
arm.dat$block <- as.factor(arm.dat$y)
arm.dat$column<- as.factor(arm.dat$x)
```

This function duplicates most of the R code used by ARM to select a best spatial model for a single data column.

```
select.spatial.model <- function(arm.dat) {
  #Calculate base (treatment only) model
  crd.lm <- lm(assessment1 ~ treatment, data=arm.dat,x=TRUE)

  #Calculate neighbor residuals
  plots <- dim(arm.dat)[1]
  cols <- max(arm.dat$x)
  rows <- max(arm.dat$y)
  mat.dim <- rows*cols
  # using a weight matrix conceptually simplifies computing neighbors.
  #In the SAS validation, I use a loop; converting to a loop might be more efficient.
  W.row <- matrix(rep(0, mat.dim*mat.dim),nrow = mat.dim)
  W.col <- matrix(rep(0, mat.dim*mat.dim),nrow = mat.dim)
  arm.dat$plot <- 1:plots

  # create a trial map
  trial.map <- matrix(rep(0,mat.dim),nrow=rows)
  for(p in 1:plots) {
    trial.map[arm.dat$y[p],arm.dat$x[p]] <- arm.dat$plot[p]
  }
  arm.dat$x.m <- arm.dat$x-1
  arm.dat$x.p <- arm.dat$x+1
  arm.dat$y.m <- arm.dat$y-1
  arm.dat$y.p <- arm.dat$y+1
  arm.dat$x.m[arm.dat$x.m<min(arm.dat$x)] <- NA
  arm.dat$x.p[arm.dat$x.p>cols] <- NA
  arm.dat$y.m[arm.dat$y.m<min(arm.dat$y)] <- NA
```

```r
arm.dat$y.p[arm.dat$y.p>rows] <- NA

#adjust neighbor covariates according distance between plot centers.
#This might affect the Papadakis model.
plot.width <- 4
plot.height <- 6
plot.buffer <- 0.5
row.buffer <- 1
col.space <- plot.width+plot.buffer
row.space <- plot.height+row.buffer
row.neighbors <- NA

#determine neighbors from trial map.
for(p in 1:plots) {
  col.neighbors <- trial.map[c(arm.dat$y.m[p], arm.dat$y.p[p]), arm.dat$x[p]]
  row.neighbors <- trial.map[arm.dat$y[p], c(arm.dat$x.m[p],arm.dat$x.p[p])]
  W.row[p,row.neighbors] <- 1/col.space
  W.col[p,col.neighbors] <- 1/row.space
}
W <- W.col + W.row
W <- W[1:plots,1:plots]
row.sums <- rowSums(W)
row.sums[row.sums==0] <- 1
W <- W/row.sums
W.row <- W.row[1:plots,1:plots]
row.sums <- rowSums(W.row)
row.sums[row.sums==0] <- 1
W.row <- W.row/row.sums
W.col <- W.col[1:plots,1:plots]
row.sums <- rowSums(W.col)
row.sums[row.sums==0] <- 1
W.col <- W.col/row.sums
arm.dat$res <- NA

#compute residuals
crd.res <- residuals(crd.lm)
arm.dat$res[as.numeric(names(crd.res))] <- crd.res
arm.dat$res[is.na(arm.dat$res)] <- 0
arm.dat$X.row = W.row %*% arm.dat$res
arm.dat$X.col = W.col %*% arm.dat$res
arm.dat$X = W %*% arm.dat$res

#Compute design model
arm.lm <- update(crd.lm, . ~ . + replicate)

#Calculate and compare trend and neighbor models
trend1.lm <- update(crd.lm, . ~ lat + lon + . )
trend2.lm <- update(crd.lm, . ~ lat + lon + I(lat^2) + I(lon^2) + I(lat*lon) + . )
trend3.lm <- update(crd.lm, . ~ lat + lon + I(lat^2) + I(lon^2) + I(lat*lon) + I(lat^3) + I(lon^3) +
nnx.lm <- update(crd.lm, . ~ X.col + . , data=arm.dat)
nny.lm <- update(crd.lm, . ~ X.row + . , data=arm.dat)
nnxy.lm <- update(crd.lm, . ~ X + . , data=arm.dat)
nns.lm <- update(crd.lm, . ~ X.row + X.col + . , data=arm.dat)
```

```
  model.list <- list(crd.lm, arm.lm, nnx.lm, nny.lm, nnxy.lm, nns.lm, trend1.lm, trend2.lm, trend3.lm)
  model.names <- c("CRD","Design","NNCol","NNRow","NNPap","NNRowCol","Trend1","Trend2","Trend3")
  anova.tbl <- do.call(anova, model.list)

  #Generate a model comparison table
  anova.tbl$RMS <- anova.tbl$RSS/anova.tbl$Res.Df
  anova.tbl$AIC <- unlist(lapply(model.list, AIC))
  anova.tbl$BIC <- unlist(lapply(model.list, BIC))
  anova.tbl$logLik <- unlist(lapply(model.list, logLik))
  best.tbl <- anova.tbl[, c('Res.Df','RMS', 'AIC', 'BIC', 'logLik')]
  # add to the table the row numbers corresponding to minimum error measures
  best.aic <- which.min(anova.tbl$AIC)
  best.bic <- which.min(anova.tbl$BIC)
  best.rms <- which.min(anova.tbl$RMS)
  best.tbl <- rbind(c(0,best.rms,best.aic,best.bic,0),best.tbl)
  row.names(best.tbl) <- c("Best Index",model.names)
  best.idx <- best.aic
  names(model.list) <- model.names

  #select a model for further analysis
  arm.lm <- model.list[[best.idx]]
  aov.tbl<-anova(arm.lm)
  res.row <- dim(aov.tbl)[1]


  #--------------------------------------
  # skipping the code for computing least square means and variance-covariance matrices
  # for mean comparison error terms.
  #--------------------------------------
  crd.tbl<-anova(crd.lm)
  return(list(arm.dat=arm.dat,
    model.list=model.list,
    aov.tbl=aov.tbl,
    best.tbl=best.tbl
  ))
}
```

Basic Spatial analysis of Cochran 1947

```
res <- select.spatial.model(arm.dat)
res$aov.tbl
```

```
## Analysis of Variance Table
##
## Response: assessment1
##                  Df  Sum Sq Mean Sq  F value    Pr(>F)
## lat               1 2811651 2811651 481.8858 1.028e-07 ***
## lon               1   17507   17507   3.0006  0.126841
## I(lat^2)          1 4096953 4096953 702.1724 2.791e-08 ***
## I(lon^2)          1  168150  168150  28.8191  0.001043 **
## I(lat * lon)      1   22786   22786   3.9053  0.088678 .
## I(lat^3)          1  145806  145806  24.9895  0.001567 **
## I(lon^3)         1   27511   27511   4.7152  0.066475 .
## I(lat * lon^2)   1   34138   34138   5.8509  0.046169 *
## I(lat^2 * lon)   1   14208   14208   2.4352  0.162607
## treatment       15  667068   44471   7.6219  0.005718 **
## Residuals        7   40843    5835
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
arm.dat <- res$arm.dat
rcb.lm <- res$model.list[[2]]
```

We will be creating an imaginary uniformity trial by calculating spatial effects and estimate plot yield as if only a single treatment where planted in each plot. We'll use the treatment with an arithmetic mean closest to the median of all treatments as our check for an imaginary uniformity trial.

```
print(treatment.means <- tapply(arm.dat$assessment1,list(arm.dat$treatment),mean))
```

```
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15
##  940 1590 1045  965 1930 1000  890 1805 1430 1750 1365 1930 1695 1275 1995
##   16
## 1620
```

```
print(treatment.median <- median(treatment.means))
```

```
## [1] 1510
```

```
print(check.trt <-which.min(abs(treatment.means-treatment.median)))
```

```
## 2
## 2
```

```
check.trt = 1
```

This suggests why there's a need for spatial analysis - the median is an untreated treatment.

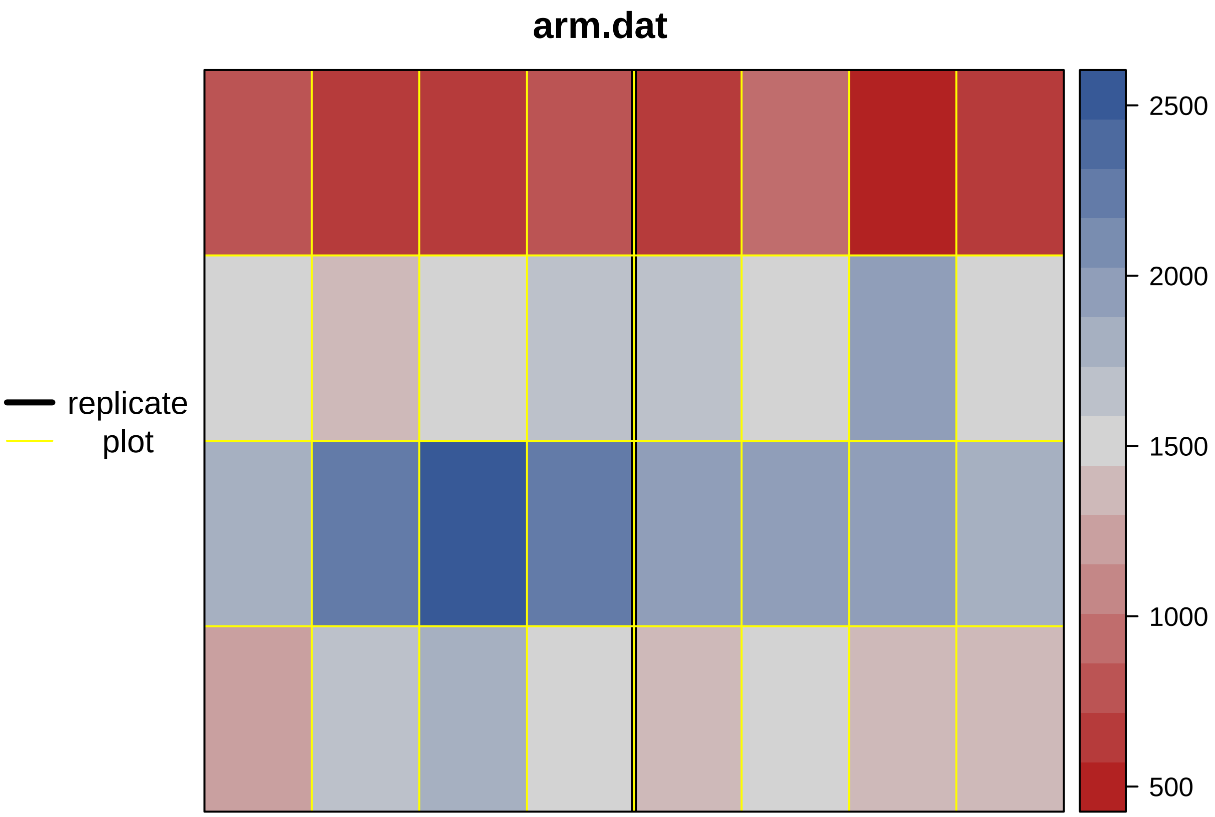
Now create a data frame for prediction

```
pred.dat <- arm.dat[,c("plot","replicate", "y", "x", "lat", "lon","X.col","X.row","X")]
pred.dat$treatment <- levels(arm.dat$treatment)[which(levels(arm.dat$treatment)==check.trt)]
pred.dat$RCB <- predict(rcb.lm,pred.dat)

pred.dat$block <- as.factor(pred.dat$y)
pred.dat$column<- as.factor(pred.dat$x)
```

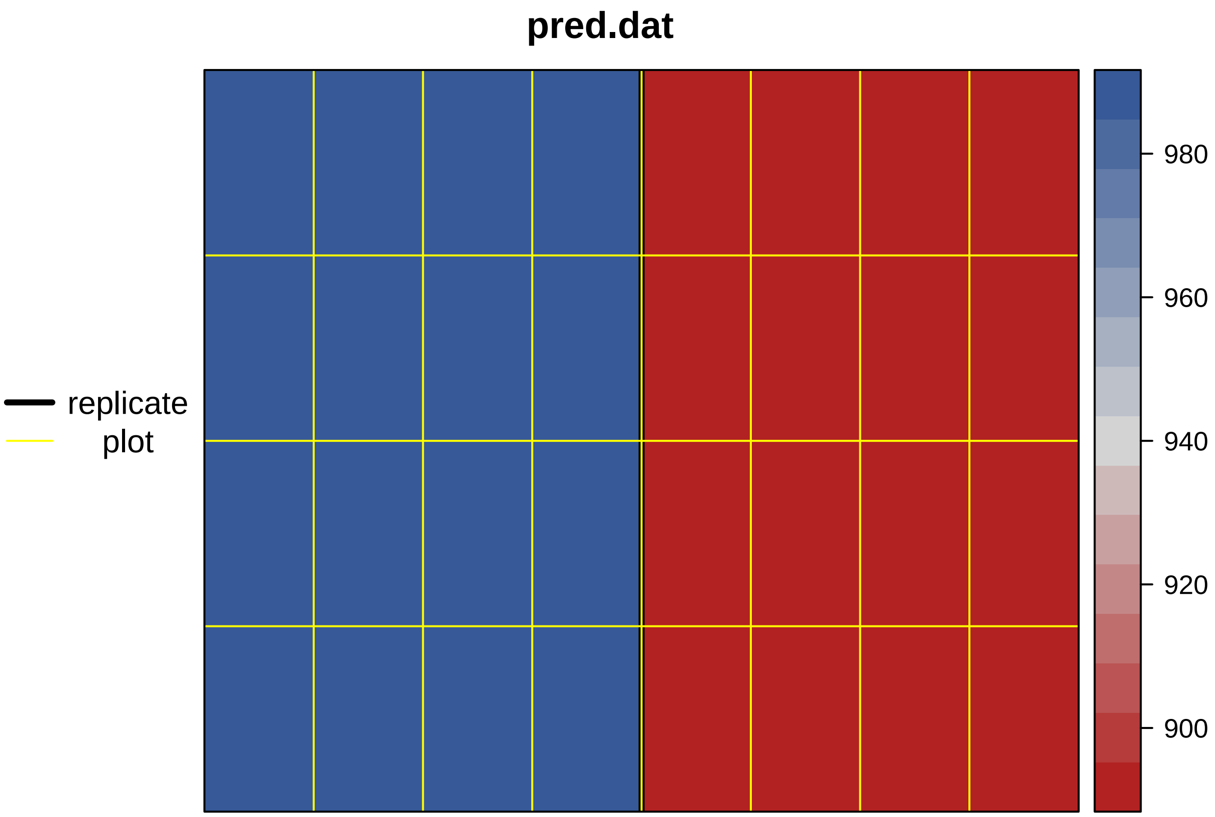
Slide 6 (Actual Yield)

```
library(agridat)
desplot(assessment1 ~ x+y, arm.dat, out1=replicate, out2=plot)
```



Predicted Yield, RCB model

```
desplot(RCB ~ x+y, pred.dat, out1=replicate, out2=plot)
```



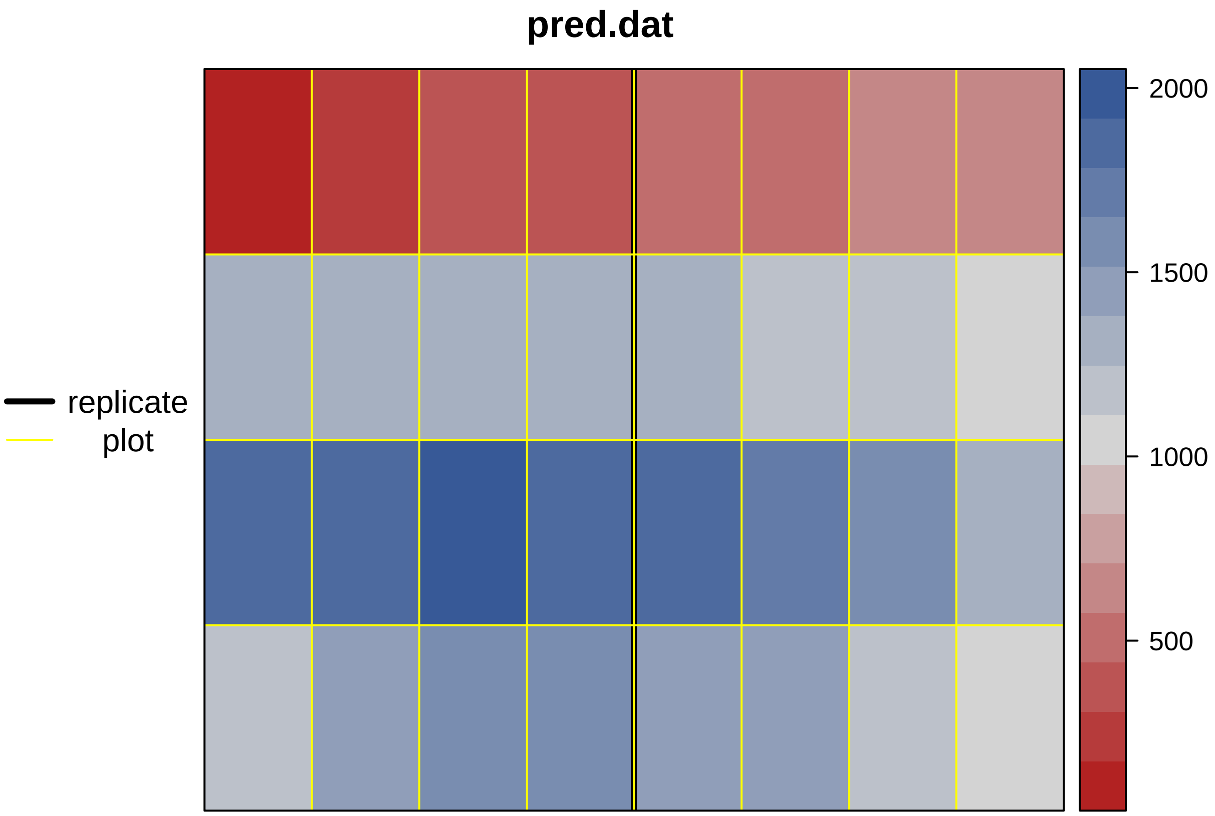
Computed plot level predict values from each model. For the presentation, I copy these to ARM and use ARM's assessment map.

```
pred.dat$CRD <- predict(res$model.list$CRD,newdata=pred.dat)
pred.dat$Col <- predict(res$model.list$NNCol,newdata=pred.dat)
pred.dat$Row <- predict(res$model.list$NNRow,newdata=pred.dat)
pred.dat$RowCol <- predict(res$model.list$NNRowCol,newdata=pred.dat)
pred.dat$Pap <- predict(res$model.list$NNPap,newdata=pred.dat)
pred.dat$Trend1 <- predict(res$model.list$Trend1,newdata=pred.dat)
pred.dat$Trend2 <- predict(res$model.list$Trend2,newdata=pred.dat)
pred.dat$Trend3 <- predict(res$model.list$Trend3,newdata=pred.dat)
pred.dat[,c("RCB","Col","Row","RowCol","Pap","Trend1","Trend2","Trend3")]
```

```
##          RCB      Col       Row    RowCol       Pap    Trend1    Trend2
## 1  985.3125 957.5225 1108.6508 1181.7953 1394.5503 1476.9381 1432.2743
## 2  894.6875 922.4775  771.3492  698.2047  485.4497  403.0619  447.7257
## 3  985.3125 974.0942 1267.5718 1409.8748 1660.7653 1459.3378 1514.6960
## 4  894.6875 981.7009 1381.0867 1556.3141 1882.7434 1424.1371 1216.2220
## 5  985.3125 953.0400 1533.5211 1601.5607 1764.3989  784.4877 1469.9597
## 6  894.6875 934.7025 1296.7613 1286.5023 1308.0536  429.4624  607.3276
## 7  985.3125 938.3700 1426.4927 1434.2884 1419.3353  793.2879 1395.1653
## 8  894.6875 976.8109 1154.0568 1303.6397 1541.1006  411.8621  539.5361
## 9  985.3125 955.3492 1715.1451 1797.4580 1989.8693 1130.7129 1752.9482
## 10 894.6875 942.8525 1043.7851 1057.9115 1093.2814  766.8874 1503.7190
## 11 985.3125 970.8342  845.9447  963.2939 1083.9814  447.0627  520.6800
## 12 894.6875 950.4592 1063.2448 1107.5611 1234.7719 1415.3369 1045.0791
## 13 985.3125 946.1125  832.9716  853.6651  822.2148  438.2626  583.3087
## 14 894.6875 929.2691  764.8626  717.9797  567.3289  420.6622  592.7368
## 15 985.3125 961.8692 1608.1167 1712.7101 1979.3727 1468.1379 1492.7900
## 16 894.6875 934.1592 1835.1466 1838.4239 1776.6710 1095.5123 1714.6901
## 17 985.3125 967.3026 1348.6539 1466.8641 1698.1606 1450.5376 1497.9921
## 18 894.6875 959.9675 1251.3554 1338.1707 1499.8163 1441.7374 1442.6785
## 19 985.3125 955.6209 1403.7897 1478.1087 1670.7938  775.6876 1506.1442
## 20 894.6875 943.5317 2094.6094 2141.9312 1929.3659 1077.9119 1463.9023
## 21 985.3125 936.6042  498.9133  472.8646  427.6984  464.6631  279.5932
## 22 894.6875 953.9909  761.6193  810.9206  977.7789  749.2871 1383.0391
## 23 985.3125 949.9159 1744.3347 1806.3361 1933.5619 1113.1126 1811.0387
## 24 894.6875 954.3984 1475.1420 1546.7743 1725.4168  758.0872 1462.6839
## 25 985.3125 938.6417 1822.1735 1842.5307 1613.8998  802.0881 1281.7612
## 26 894.6875 938.7775 1617.8465 1632.7924 1624.5928 1104.3124 1782.1693
## 27 985.3125 970.2909  761.6193  874.4009  969.7016  455.8629  419.4415
## 28 894.6875 957.6584 1416.7629 1499.3940 1716.7541 1086.7121 1608.6011
## 29 985.3125 964.4501 1685.9555 1802.8630 2104.6499 1121.9128 1801.2983
## 30 894.6875 941.2225 1977.8512 2012.7854 1789.2740  740.4869 1264.7846
## 31 985.3125 950.5950 1621.0898 1682.1532 1737.7341 1139.5131 1665.9883
## 32 894.6875 956.1642 1115.1374 1183.1804 1286.2148 1432.9373 1348.7552
##       Trend3
## 1  1239.5802
## 2   640.4198
## 3  1539.6248
## 4  1213.7110
## 5  1378.4414
## 6   486.0940
## 7  1347.9324
## 8   578.6953
## 9  1892.3683
## 10 1317.3442
## 11  372.2292
## 12  993.9097
## 13  436.4990
## 14  530.3730
## 15 1435.9851
## 16 1690.5953
## 17 1559.8582
## 18 1506.0443
## 19 1365.1256
## 20 1339.0424
## 21  162.2289
## 22 1155.8206
## 23 1899.9947
## 24 1244.4562
## 25 1264.2398
## 26 1817.7437
## 27  283.9255
## 28 1527.9085
## 29 1927.9893
## 30 1060.7964
## 31 1783.7729
## 32 1387.5423
```

Slide 40 (Predicted Yield, Cubic Trend Model)

```
desplot(Trend3 ~ x+y, pred.dat, out1=replicate, out2=plot)
```



Residuals plot.

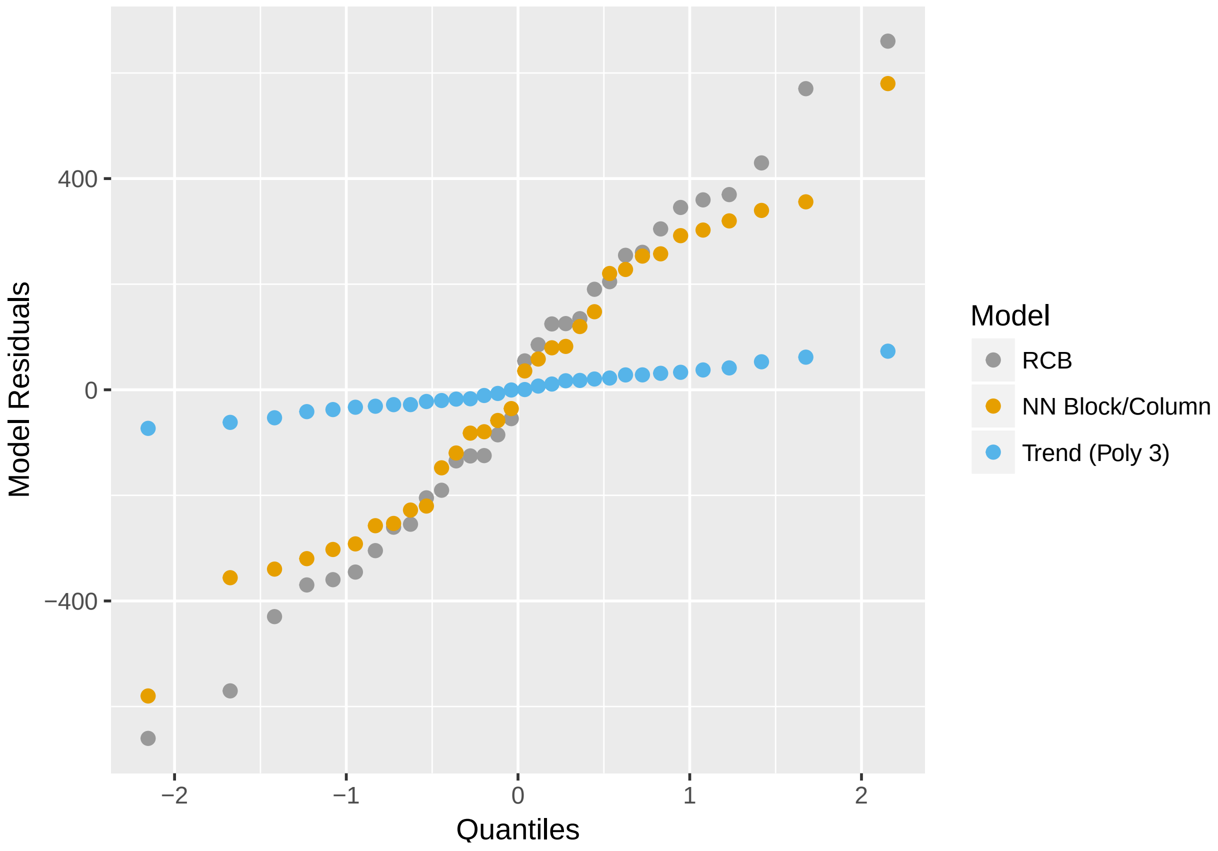
I didn't include this in the presentation, but this plot demonstrates how the spatial model improves error estimates.

```
pred.dat$CRDresid <- resid(res$model.list$CRD)
pred.dat$RCBresid <- resid(res$model.list$Design)
pred.dat$Trend3resid <- resid(res$model.list$Trend3)
pred.dat$RowColresid <- resid(res$model.list$NNRowCol)
```

And a data frame for plotting residuals

```
residuals.dat <- data.frame(
  Residual = c(pred.dat$RCBresid,
               pred.dat$RowColresid,
               pred.dat$Trend3resid),
  Model = c(rep("RCB",length(pred.dat$RCBresid)),
            rep("RowCol",length(pred.dat$RowColresid)),
            rep("Trend3",length(pred.dat$Trend3resid)))
)
```

```
ggplot(residuals.dat, aes(sample=Residual,color=Model,linetype=Model)) +
  stat_qq(aes(color=Model),size = 2) +
  scale_colour_manual(values=cbPalette,labels = c("RCB", "NN Block/Column","Trend (Poly 3)")) +
  labs(colour = "Model", x="Quantiles",y="Model Residuals")
```



Slide 22

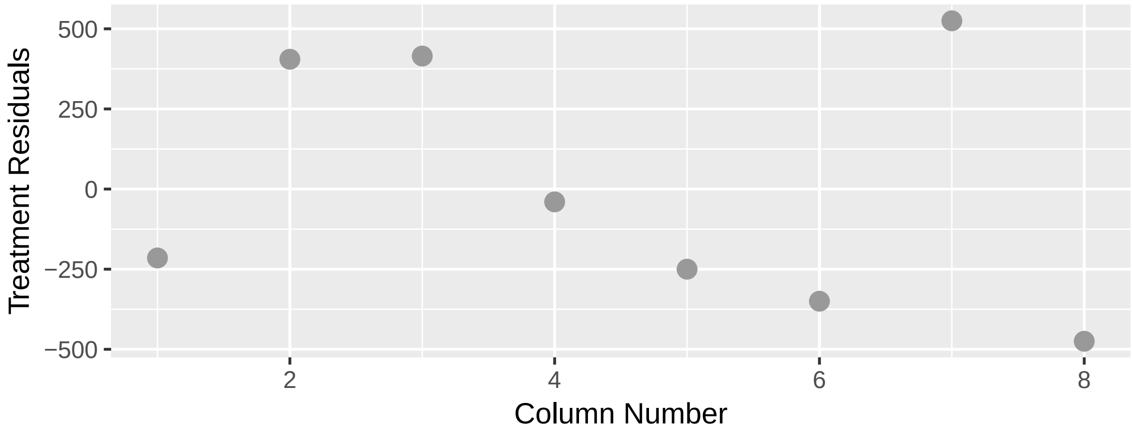
To illustrate nearest neighbor calculations, we compute row neighbor residual means for row 3

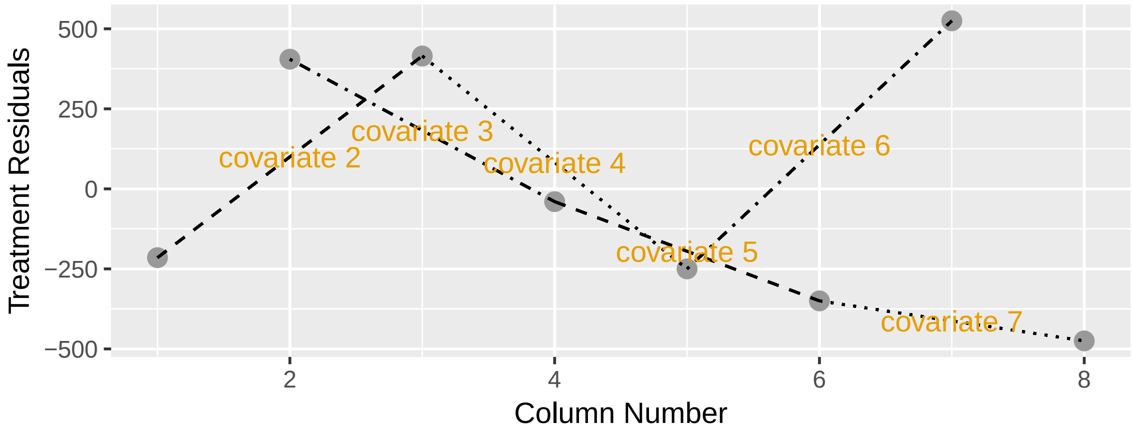
```
library(gridExtra)
```

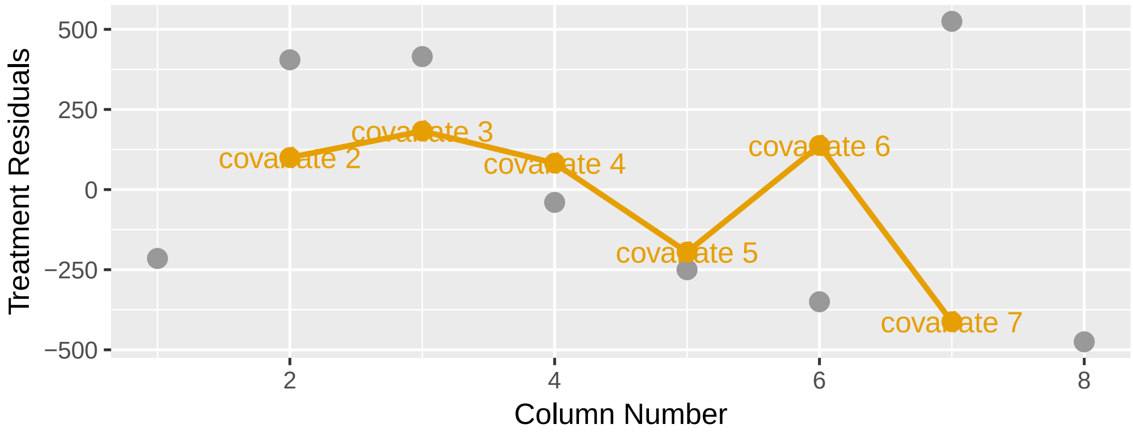
```
## Warning: package 'gridExtra' was built under R version 3.3.2
```

```
row.dat <- subset(pred.dat,pred.dat$block==3)
row.dat <- row.dat[order(row.dat$x),]
midpoints <- data.frame(
  x = row.dat$x[2:7],
  y=c((row.dat$CRDresid[1]+row.dat$CRDresid[3])/2,
      (row.dat$CRDresid[2]+row.dat$CRDresid[4])/2,
      (row.dat$CRDresid[3]+row.dat$CRDresid[5])/2,
      (row.dat$CRDresid[4]+row.dat$CRDresid[6])/2,
      (row.dat$CRDresid[5]+row.dat$CRDresid[7])/2,
      (row.dat$CRDresid[6]+row.dat$CRDresid[8])/2),
  lbl = c("covariate 2","covariate 3","covariate 4","covariate 5","covariate 6","covariate 7")
      )
```

```
grid.arrange(
  arrangeGrob(
ggplot(row.dat, aes(x, CRDresid)) + geom_point(colour = cbPalette[1],size = 3) + labs(x="Column Number"
ggplot(row.dat, aes(x, CRDresid)) + geom_point(colour = cbPalette[1],size = 3) + labs(x="Column Number"
    geom_segment(x =row.dat$x[1], y = row.dat$CRDresid[1], xend = row.dat$x[3], yend = row.dat$CRDresid
    geom_segment(x =row.dat$x[2], y = row.dat$CRDresid[2], xend = row.dat$x[4], yend = row.dat$CRDresid
    geom_segment(x =row.dat$x[3], y = row.dat$CRDresid[3], xend = row.dat$x[5], yend = row.dat$CRDresid
    geom_segment(x =row.dat$x[4], y = row.dat$CRDresid[4], xend = row.dat$x[6], yend = row.dat$CRDresid
    geom_segment(x =row.dat$x[5], y = row.dat$CRDresid[5], xend = row.dat$x[7], yend = row.dat$CRDresid
    geom_segment(x =row.dat$x[6], y = row.dat$CRDresid[6], xend = row.dat$x[8], yend = row.dat$CRDresid
    geom_text(data = midpoints, aes(x =x ,y = y, label = lbl),colour = cbPalette[2]),
    ggplot(row.dat, aes(x, CRDresid)) + geom_point(colour = cbPalette[1],size = 3) + labs(x="Column Numb
    geom_point(colour = cbPalette[1],size = 3) + labs(x="Column Number",y="Treatment Residuals") +
    geom_point(colour = cbPalette[2],size = 3,aes(x, y),data=midpoints) + geom_line(colour = cbPalette[
        geom_text(data = midpoints, aes(x =x ,y = y, label = lbl),colour = cbPalette[2])
      ))
```

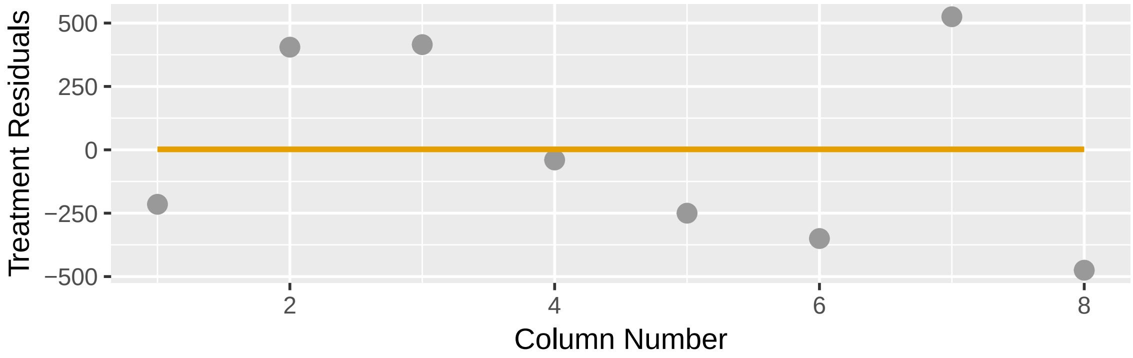




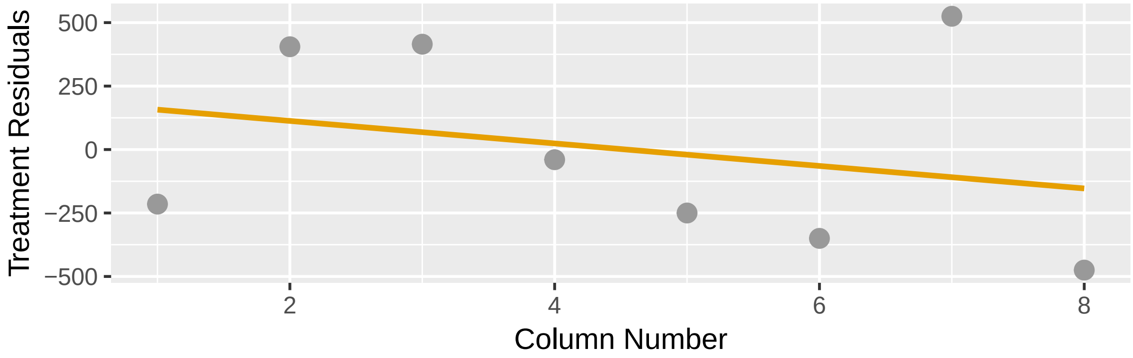


Slide 19

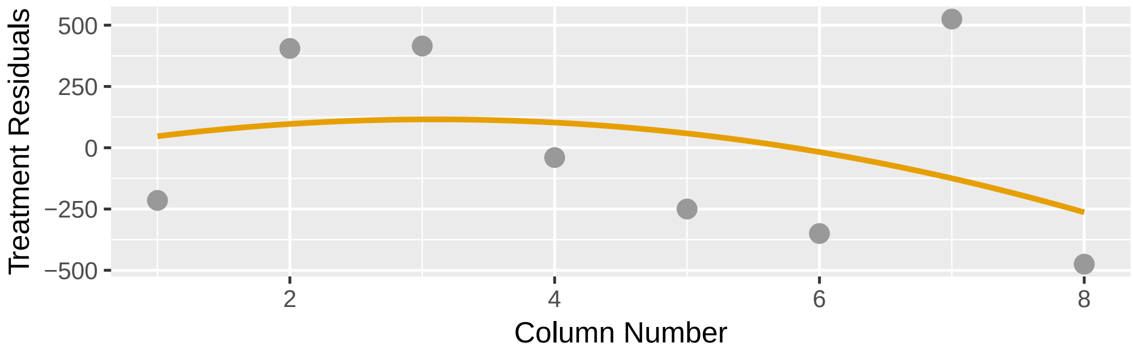
```r
grid.arrange(
  arrangeGrob(
ggplot(row.dat, aes(x, CRDresid)) + geom_point(colour = cbPalette[1],size = 3) +
   labs(x="Column Number",y="Treatment Residuals",title=expression(yield == mu)) +
   stat_smooth(method = "lm", formula = y ~ 1, se=FALSE, size = 1, color=cbPalette[2]),
ggplot(row.dat, aes(x, CRDresid)) + geom_point(colour = cbPalette[1],size = 3) +
    labs(x="Column Number",y="Treatment Residuals",title=expression(yield == mu + x)) +
    stat_smooth(method = "lm", formula = y ~ x, se=FALSE, size = 1, color=cbPalette[2]),
ggplot(row.dat, aes(x, CRDresid)) + geom_point(colour = cbPalette[1],size = 3) +
   labs(x="Column Number",y="Treatment Residuals",title=expression(yield == mu + x + x^2)) +
   stat_smooth(method = "lm", formula = y ~ x + I(x^2), se=FALSE, size = 1, color=cbPalette[2]),
ggplot(row.dat, aes(x, CRDresid)) + geom_point(colour = cbPalette[1],size = 3) +
   labs(x="Column Number",y="Treatment Residuals",title=expression(yield == mu + x + x^2 + x^3)) +
   stat_smooth(method = "lm", formula = y ~ x + I(x^2)+I(x^3), se=FALSE, size = 1, color=cbPalette[2]),
   ncol=1
      ))
```



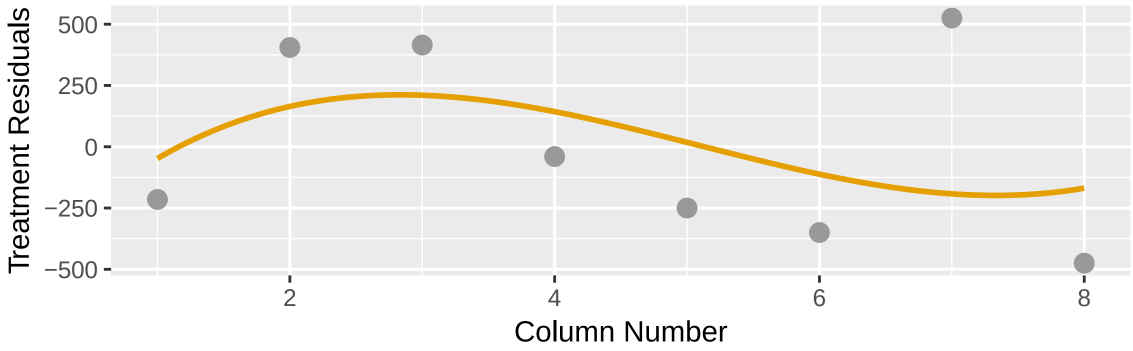
yield $= \mu$



yield $= \mu + x$



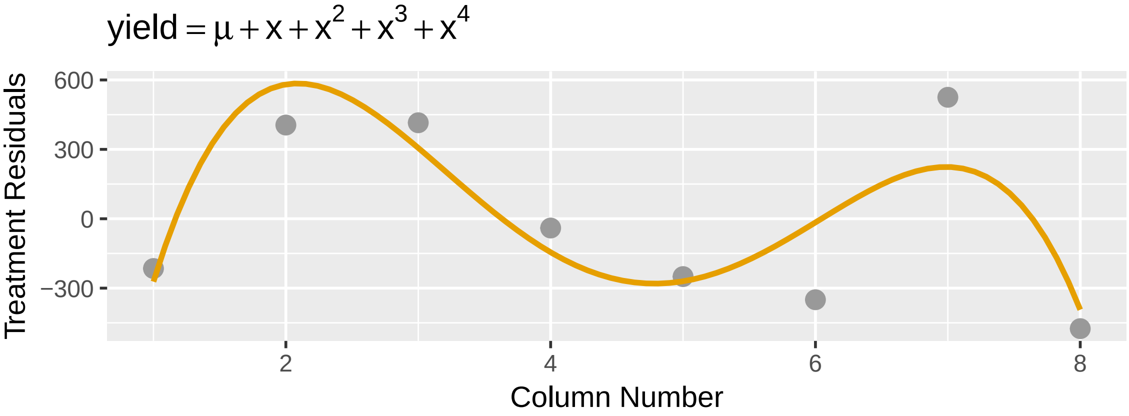
yield $= \mu + x + x^2$

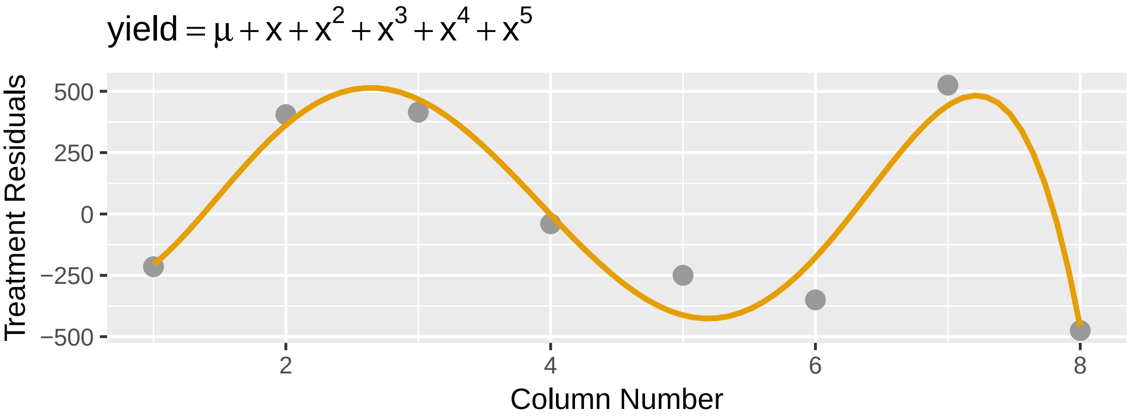


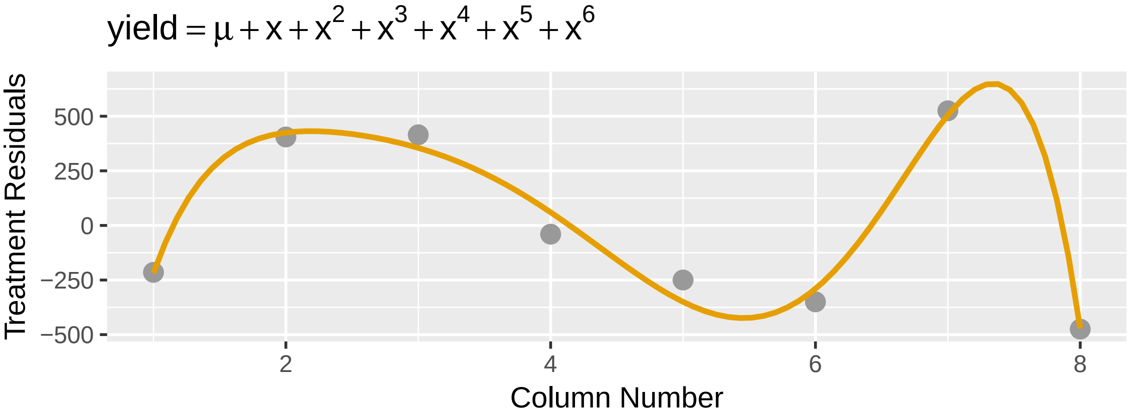
yield $= \mu + x + x^2 + x^3$

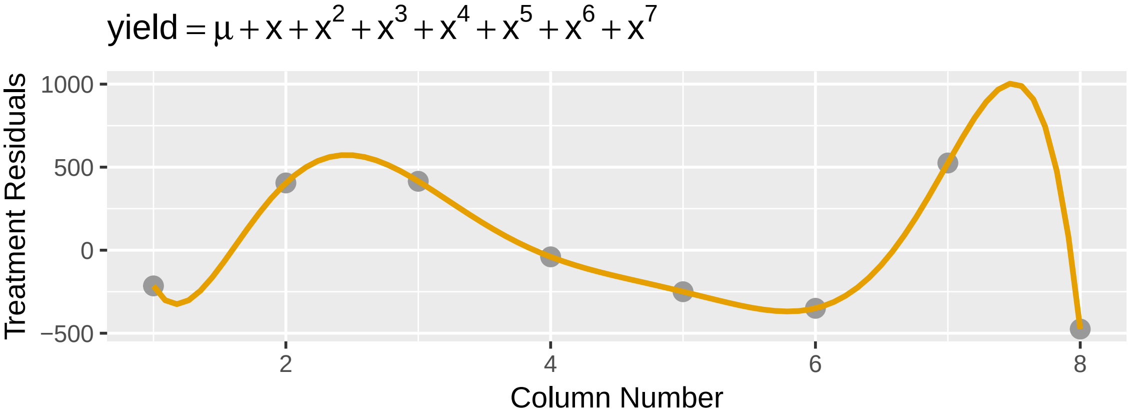
Slide 31

```
grid.arrange(
  arrangeGrob(
ggplot(row.dat, aes(x, CRDresid)) + geom_point(colour = cbPalette[1],size = 3) +
  labs(x="Column Number",y="Treatment Residuals",title=expression(yield == mu + x + x^2 + x^3 + x^4)) +
  stat_smooth(method = "lm", formula = y ~ poly(x,4), se=FALSE, size = 1, color=cbPalette[2]),
ggplot(row.dat, aes(x, CRDresid)) + geom_point(colour = cbPalette[1],size = 3) +
  labs(x="Column Number",y="Treatment Residuals",title=expression(yield == mu + x + x^2 + x^3 + x^4 + x
  stat_smooth(method = "lm", formula = y ~ poly(x,5), se=FALSE, size = 1, color=cbPalette[2]),
ggplot(row.dat, aes(x, CRDresid)) + geom_point(colour = cbPalette[1],size = 3) +
  labs(x="Column Number",y="Treatment Residuals",title=expression(yield == mu + x + x^2 + x^3 + x^4 + x
  stat_smooth(method = "lm", formula = y ~ poly(x,6), se=FALSE, size = 1, color=cbPalette[2]),
ggplot(row.dat, aes(x, CRDresid)) + geom_point(colour = cbPalette[1],size = 3) +
  labs(x="Column Number",y="Treatment Residuals",title=expression(yield == mu + x + x^2 + x^3 + x^4 + x
  stat_smooth(method = "lm", formula = y ~ poly(x,7), se=FALSE, size = 1, color=cbPalette[2]),
ncol=1
      ))
```


$\text{yield} = \mu + x + x^2 + x^3 + x^4$
Treatment Residuals
600
300
0
−300
2
4
6
8
Column Number


$\text{yield} = \mu + x + x^2 + x^3 + x^4 + x^5$
Treatment Residuals
500
250
0
−250
−500
2
4
6
8
Column Number


$\text{yield} = \mu + x + x^2 + x^3 + x^4 + x^5 + x^6$
Treatment Residuals
500
250
0
−250
−500
2
4
6
8
Column Number


$\text{yield} = \mu + x + x^2 + x^3 + x^4 + x^5 + x^6 + x^7$
Treatment Residuals
1000
500
0
−500
2
4
6
8
Column Number