DSA 8020 R Session 10: Random and Mixed Effects Models and Computer Experiments

Whitney

Contents

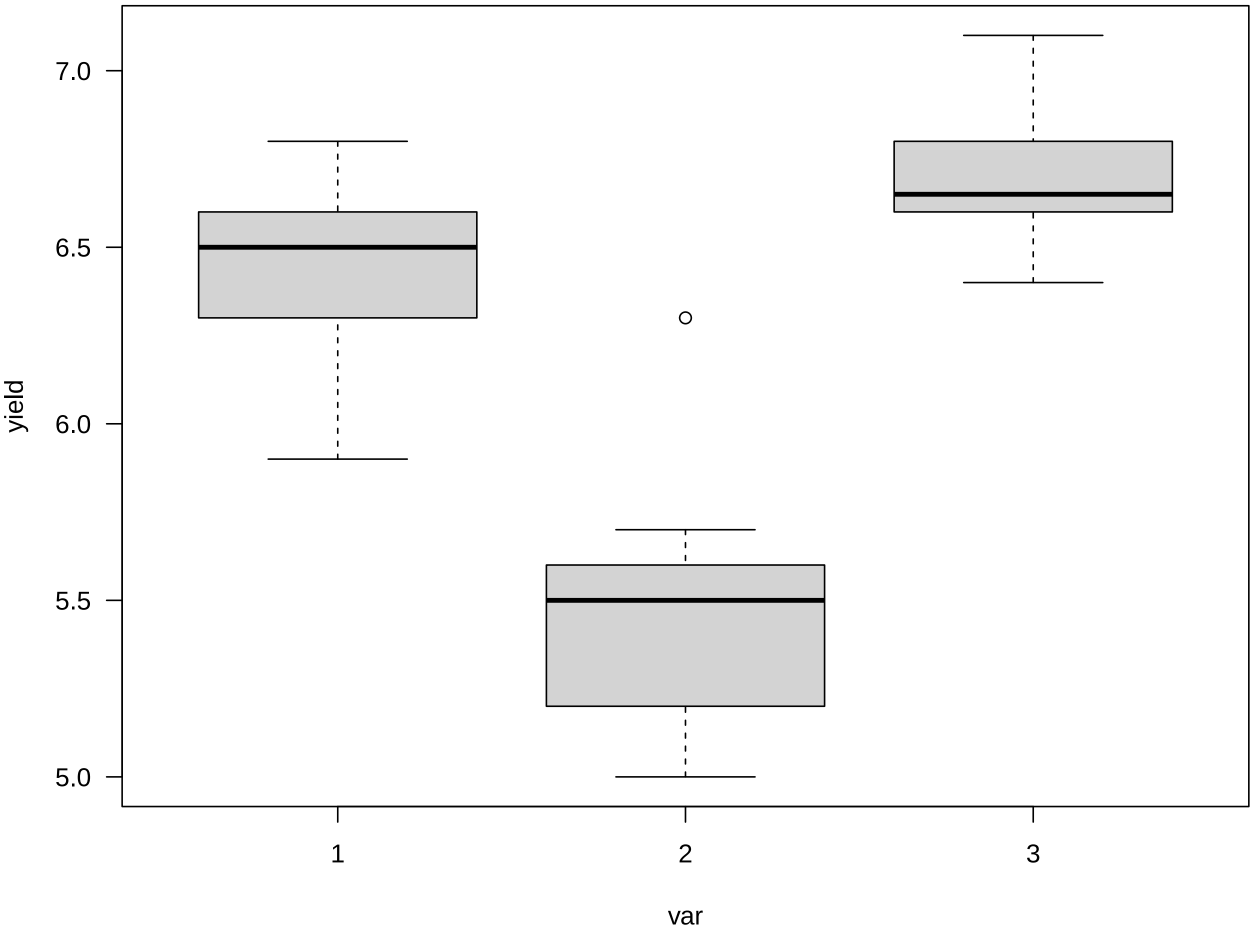
Random Effects Example

Suppose that an agronomist is studying a large number of varieties of soybeans for yield. The agronomist randomly selects three varieties, and then randomly assigns each of those varieties to 10 of 30 available plots.

Model: $y_{ij} = \mu + \alpha_i + \epsilon_{ij}$, $\alpha_i s \overset{i.i.d.}{\sim} \mathrm{N}(0, \sigma_\alpha^2)$, $\epsilon_{ij} s \overset{i.i.d.}{\sim} \mathrm{N}(0, \sigma^2)$. $\alpha_i s$ and $\epsilon_{ij} s$ are independent to each other

Read the data into R

```r
v1 <- c(6.6, 6.4, 5.9, 6.6, 6.2, 6.7, 6.3, 6.5, 6.5, 6.8)
v2 <- c(5.6, 5.2, 5.3, 5.1, 5.7, 5.6, 5.6, 6.3, 5.0, 5.4)
v3 <- c(6.9, 7.1, 6.4, 6.7, 6.5, 6.6, 6.6, 6.6, 6.8, 6.8)
yield <- c(v1, v2, v3)
var <- factor(c(rep(1, 10), rep(2, 10), rep(3, 10)))
plot(yield ~ var, las = 1)
```



Fitting a fixed effects model

```
fixef <- lm(yield ~ var)
anova(fixef)
```

```
## Analysis of Variance Table
##
## Response: yield
##           Df Sum Sq Mean Sq F value    Pr(>F)
## var        2  8.306  4.1530  49.593 9.114e-10 ***
## Residuals 27  2.261  0.0837
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
coefficients(fixef)
```

```
## (Intercept)        var2        var3
##        6.45       -0.97        0.25
```

Fitting a random effects model

```
library(lme4)
```

```
## Loading required package: Matrix
```

```
randef <- lmer(yield ~ 1 + (1|var), REML = TRUE)
summary(randef)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: yield ~ 1 + (1 | var)
##
## REML criterion at convergence: 21.6
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -1.8839 -0.6181  0.1118  0.4962  2.7828
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  var      (Intercept) 0.40693  0.6379
##  Residual             0.08374  0.2894
## Number of obs: 30, groups:  var, 3
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept)   6.2100     0.3721   16.69
```

Let's construct CIs for σ_α^2, σ^2, and μ

```
## Compute the confidence intervals (CIs) using profile likelihood
CIs <- confint(randef, oldNames = FALSE)
```

```
## Computing profile confidence intervals ...
```

```
CIs
```

```
##                         2.5 %    97.5 %
## sd_(Intercept)|var 0.2637525 1.5512218
## sigma              0.2265053 0.3877781
## (Intercept)        5.3618584 7.0581407
```

RCBD: Fixed vs. Random Block

Load R libraries

```
library(lsmeans)
library(lmerTest)
```

Read the data

```
### Create the data set
x <- c(52, 47, 44, 51, 42, 60, 55, 49, 52, 43, 56, 48, 45, 44, 38)
trt <- rep(c("A", "B", "C"), each = 5)
blk <- rep(1:5, 3)
dat <- data.frame(x = x, trt = trt, blk = as.factor(blk))
```

Fixed block

```
fixef <- lm(x ~ trt + blk, data = dat)
anova(fixef)
```

```
## Analysis of Variance Table
##
## Response: x
##           Df Sum Sq Mean Sq F value    Pr(>F)
## trt        2   89.2   44.60  7.6239 0.0140226 *
## blk        4  363.6   90.90 15.5385 0.0007684 ***
## Residuals  8   46.8    5.85
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
```

Random block

```
randef <- lmer(x ~ trt + (1|blk), REML = TRUE, data = dat)
summary(randef)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite’s method [
## lmerModLmerTest]
## Formula: x ~ trt + (1 | blk)
##    Data: dat
##
## REML criterion at convergence: 71.1
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -1.1417 -0.6147 -0.1494  0.5772  1.3390
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  blk      (Intercept) 28.35    5.324
##  Residual              5.85    2.419
## Number of obs: 15, groups:  blk, 5
##
## Fixed effects:
##             Estimate Std. Error     df t value Pr(>|t|)
## (Intercept)   47.200      2.615  5.054  18.047 8.76e-06 ***
## trtB            4.600      1.530  8.000   3.007   0.0169 *
## trtC           -1.000      1.530  8.000  -0.654   0.5316
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##      (Intr) trtB
## trtB -0.292
## trtC -0.292  0.500
```

```
lsmeans(randef, list(pairwise ~ trt), adjust = "none")
```

```
## $`lsmeans of trt`
##  trt lsmean   SE   df lower.CL upper.CL
##  A     47.2 2.62 5.05     40.5     53.9
##  B     51.8 2.62 5.05     45.1     58.5
##  C     46.2 2.62 5.05     39.5     52.9
##
## Degrees-of-freedom method: kenward-roger
## Confidence level used: 0.95
##
## $`pairwise differences of trt`
##  1     estimate   SE df t.ratio p.value
##  A - B     -4.6 1.53  8  -3.007  0.0169
##  A - C      1.0 1.53  8   0.654  0.5316
##  B - C      5.6 1.53  8   3.661  0.0064
##
## Degrees-of-freedom method: kenward-roger
```

Computer Experiments

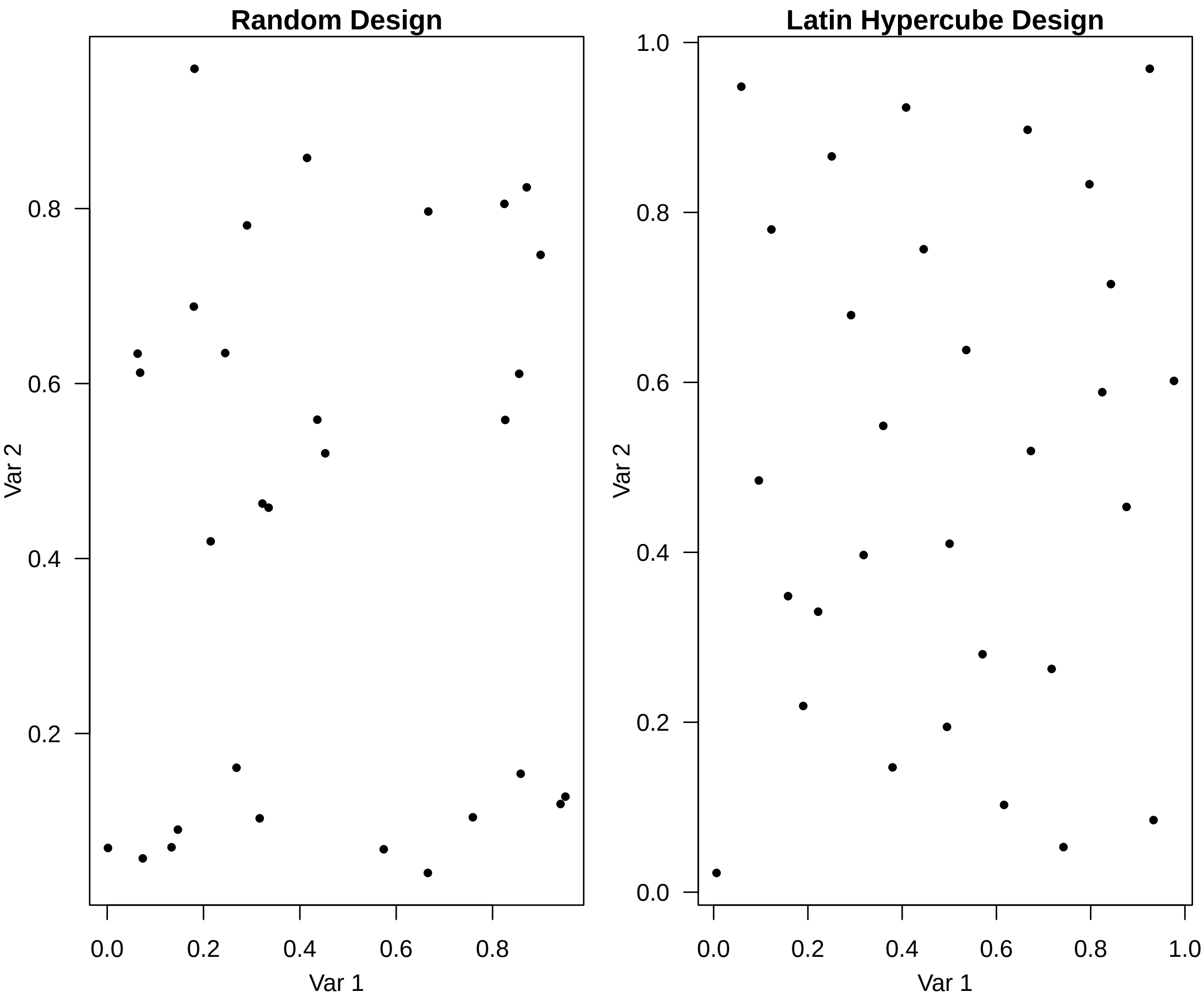
Design: Latin hypercube

```
# install.packages("lhs") # Latin Hypercube Sample #Package
library(lhs)
# Generate a good n x k LHD
LHD = maximinLHS(n = 30, k = 2, dup = 5)
# "dup" is an integer tuning parameter that determines the number of
# candidate points considered. Larger values should improve results
# but require more computational resources.

# Display the LHD
LHD
```

```
##              [,1]       [,2]
##  [1,] 0.221755087 0.33007830
##  [2,] 0.379639503 0.14686624
##  [3,] 0.797359005 0.83314876
##  [4,] 0.570596837 0.27999838
##  [5,] 0.157882978 0.34836282
##  [6,] 0.445643589 0.75669330
##  [7,] 0.824562514 0.58838950
##  [8,] 0.250526791 0.86589616
##  [9,] 0.359914122 0.54877913
## [10,] 0.616250646 0.10268276
## [11,] 0.976751848 0.60169145
## [12,] 0.673116737 0.51920994
## [13,] 0.122561276 0.77993792
## [14,] 0.318234138 0.39674549
## [15,] 0.742275827 0.05303083
## [16,] 0.666185295 0.89719952
## [17,] 0.500703842 0.41009612
## [18,] 0.717129275 0.26273225
## [19,] 0.291518917 0.67910203
## [20,] 0.842907554 0.71561482
## [21,] 0.190088808 0.21913620
## [22,] 0.408408053 0.92350893
## [23,] 0.876065864 0.45339935
## [24,] 0.495073745 0.19453092
## [25,] 0.095906340 0.48448903
## [26,] 0.925350411 0.96908045
## [27,] 0.058706750 0.94801022
## [28,] 0.933369762 0.08481102
## [29,] 0.006159548 0.02262674
## [30,] 0.536073998 0.63803179
```

```r
par(las = 1, mar = c(3.5, 3.5, 1.2, 0.5), mgp = c(2.2, 1, 0), mfrow = c(1, 2))
plot(cbind(runif(n = 30), runif(n = 30)), cex = 0.8, pch = 16,
     xlab = "Var 1", ylab = "Var 2", main = "Random Design")
plot(LHD, cex = 0.8, pch = 16, xlab = "Var 1", ylab = "Var 2",
     main = "Latin Hypercube Design")
```



Analysis: Gaussian Process

```
# Load the data
neuron <- read.table("http://deanvossdraguljic.ietsandbox.net/DeanVossDraguljic/R-data/neuron.txt", head
head(neuron, 10)
```

```
##        gNaFsc    gKdrsc fr
## 1  0.38593729 0.2120652 33
## 2  0.04666927 0.4594742  0
## 3  1.00000000 0.4473344 46
## 4  0.95467637 0.3351407 44
## 5  0.53334929 0.7981310 41
## 6  0.59166751 0.6042714 41
## 7  0.18570301 0.3799469 31
## 8  0.49927784 0.2444170 36
## 9  0.74609113 0.3949591 42
## 10 0.07269414 1.0000000  0
```

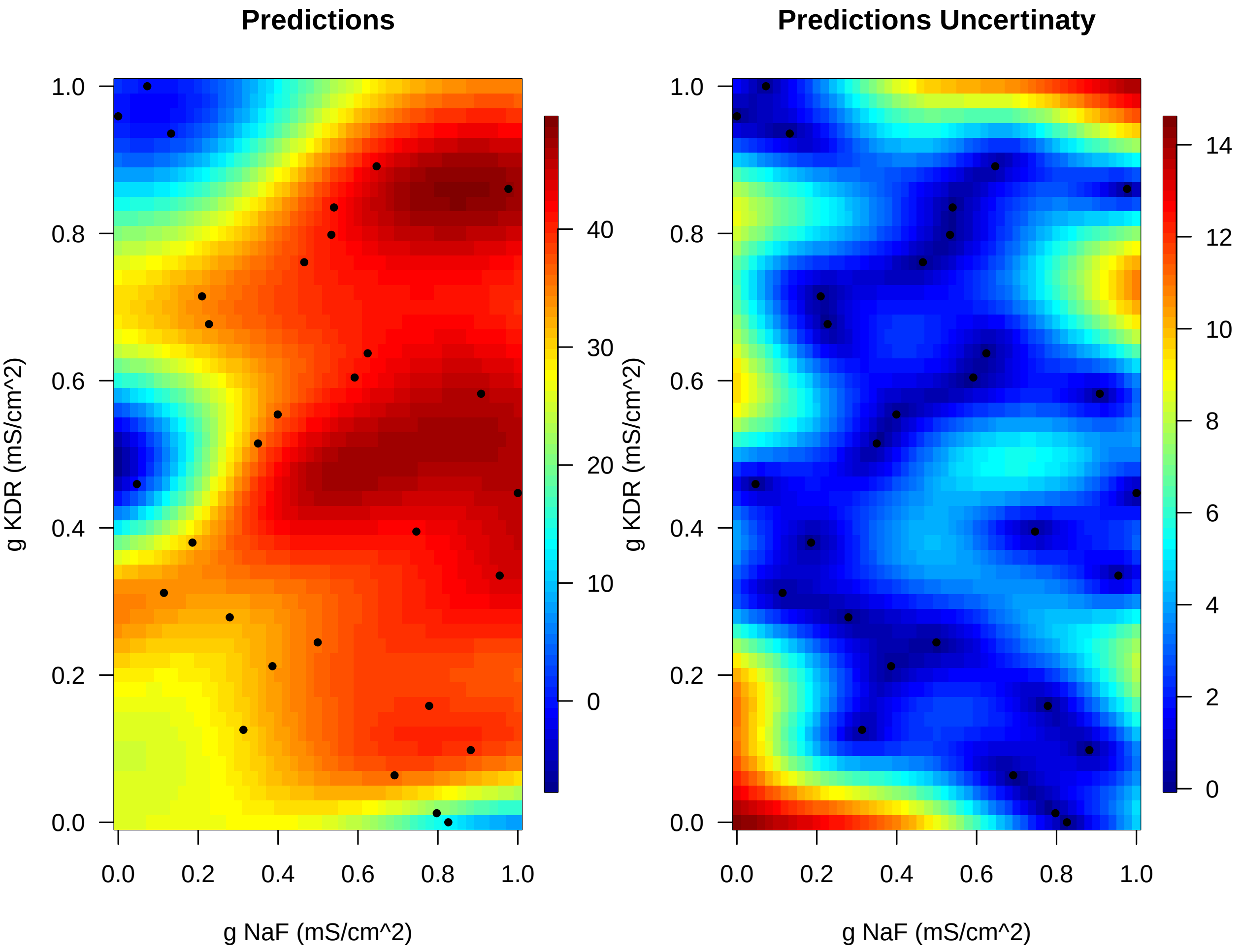
```r
# Fit a GP
library(mlegp)
GPFit <- mlegp(neuron[, 1:2], neuron[, 3])
```

```
## no reps detected - nugget will not be estimated
##
## ========== FITTING GP # 1 ==============================
## running simplex # 1...
## ...done
## ...simplex #1 complete, loglike = -104.446501 (convergence)
## running simplex # 2...
## ...done
## ...simplex #2 complete, loglike = -104.446501 (convergence)
## running simplex # 3...
## ...done
## ...simplex #3 complete, loglike = -104.446502 (convergence)
## running simplex # 4...
## ...done
## ...simplex #4 complete, loglike = -104.446501 (convergence)
## running simplex # 5...
## ...done
## ...simplex #5 complete, loglike = -104.446501 (convergence)
##
## using L-BFGS method from simplex #1...
##  iteration: 1,loglike = -104.446501
## ...L-BFGS method complete
##
## Maximum likelihood estimates found, log like =  -104.446501
## creating gp object......done
```

```r
summary(GPFit)
```

```
##
## Total observations = 30
## Dimensions = 2
##
## mu = 27.61157
## sig2:    251.8751
## nugget:  0
##
## Correlation parameters:
##
##        beta a
## 1  5.027878 2
## 2 50.228477 2
##
## Log likelihood = -104.4465
##
## CV RMSE: 7.312618
## CV RMaxSE: 1020.777
```

```
# Make prediction
predictedX = expand.grid(g_NaF = seq(0, 1, 0.02), g_KDR = seq(0, 1, 0.02))
yhats = predict(GPFit, predictedX, se.fit = T)
# Visualize predictions and their uncertainty
library(fields)
par(mfrow = c(1, 2))
image.plot(seq(0, 1, 0.02), seq(0, 1, 0.02), matrix(yhats$fit, 51, 51),
           xlab = "g NaF (mS/cm^2)", ylab = "g KDR (mS/cm^2)", las = 1,
           main = "Predictions")
points(neuron[, 1:2], pch = 16, cex = 0.75)
image.plot(seq(0, 1, 0.02), seq(0, 1, 0.02), matrix(yhats$se.fit, 51, 51),
           xlab = "g NaF (mS/cm^2)", ylab = "g KDR (mS/cm^2)", las = 1,
           main = "Predictions Uncertinaty")
points(neuron[, 1:2], pch = 16, cex = 0.75)
```


Predictions
Predictions Uncertinaty
g KDR (mS/cm^2)
g NaF (mS/cm^2)