Quantitative Methods in Linguistics – Lecture 9

Adrian Brasoveanu*

May 18, 2014

Contents

*These notes have been generated with the 'knitr' package (Xie 2013) and are based on many sources, including but not limited to: Abelson (1995), Miles and Shevlin (2001), Faraway (2004), De Veaux et al. (2005), Braun and Murdoch (2007), Gelman and Hill (2007), Baayen (2008), Johnson (2008), Wright and London (2009), Gries (2009), Kruschke (2011), Diez et al. (2013), Gries (2013).

We review here linear models, go over certain details in more depth, and simulate data for a variety of linear models, including t-tests and 1-way and 2-ways anovas. We'll also briefly introduce mixed-effects models. We'll see that they really involve just a more structured way of generating the same kind of data we have been dealing with up until now. This set of lecture notes is primarily based on Kéry (2010).

1 Essentials of linear models

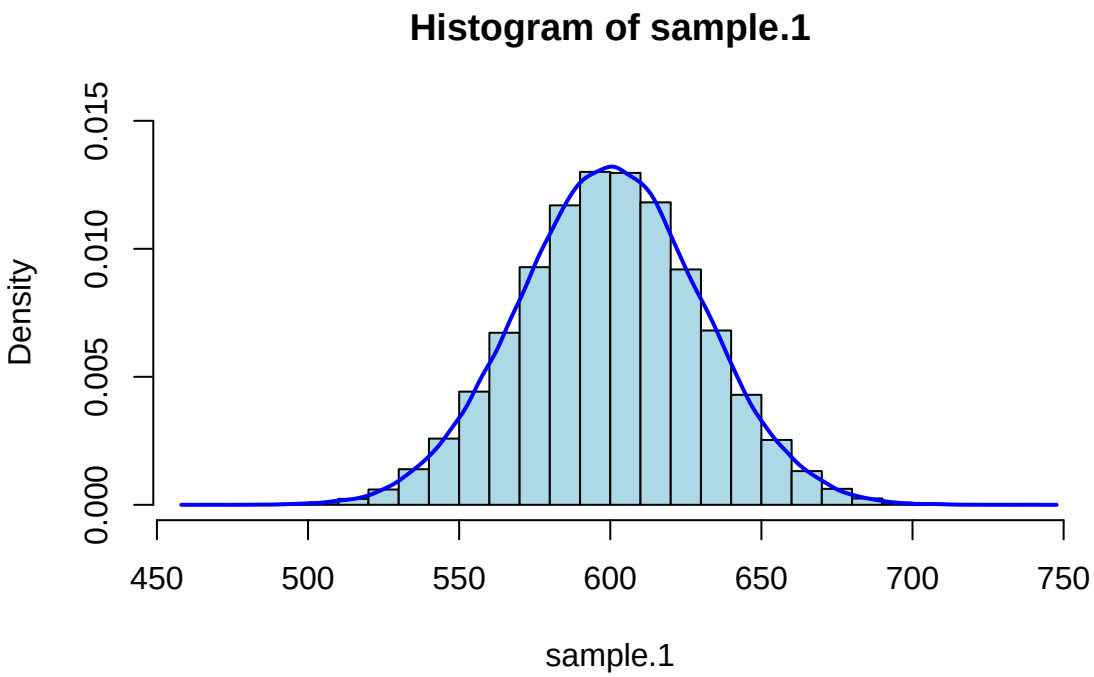
Linear models have a stochastic / random component and a deterministic component.

For example, for OLS (ordinary least squares) regression, the deterministic part is the equation $Y = X \cdot \beta$ and the stochastic part is the error distribution, assumed to be $Normal(0, \sigma^2)$. That is, an OLS regression model is parametrized by the coefficients β together with the error variance σ^2.

1.1 The stochastic component of linear models: probability distributions

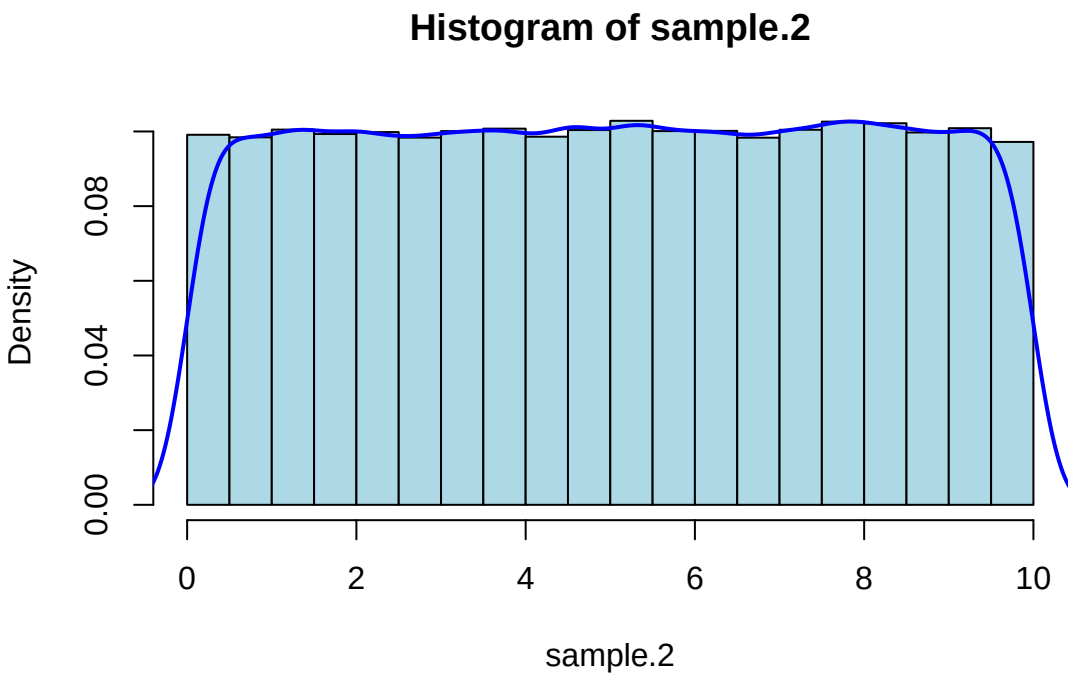
1.1.1 Normal distributions

```
> n.obs <- 1e+05
> mu <- 600
> st.dev <- 30
> sample.1 <- rnorm(n = n.obs, mean = mu, sd = st.dev)
> head(sample.1)

[1] 593.5 572.7 597.8 539.7 590.5 589.1

> hist(sample.1, col = "lightblue", freq = FALSE, ylim = c(0, 0.015),
+     breaks = 30)
> lines(density(sample.1), col = "blue", lwd = 2)
```



1.1.2 Continuous uniform distributions

```
> n.obs <- 1e+05
> a <- lower.limit <- 0
> b <- upper.limit <- 10
> sample.2 <- runif(n = n.obs, min = a, max = b)
> head(sample.2)

[1] 3.729 1.454 7.670 5.788 8.712 3.020

> hist(sample.2, col = "lightblue", freq = FALSE, breaks = 30)
> lines(density(sample.2), col = "blue", lwd = 2)
```



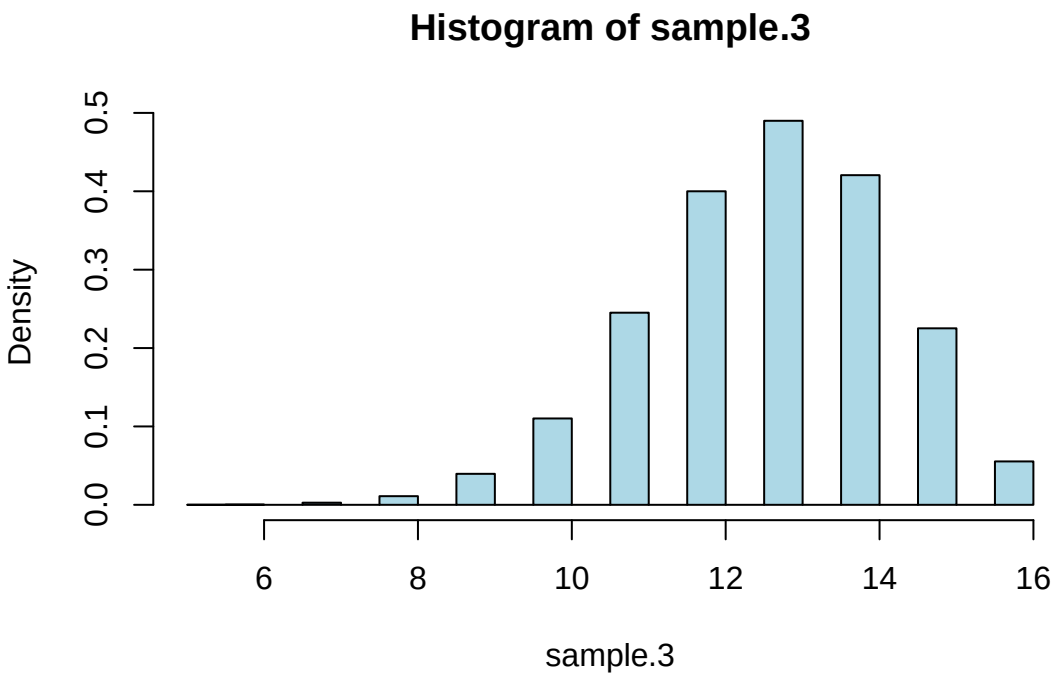
1.1.3 Binomial distributions

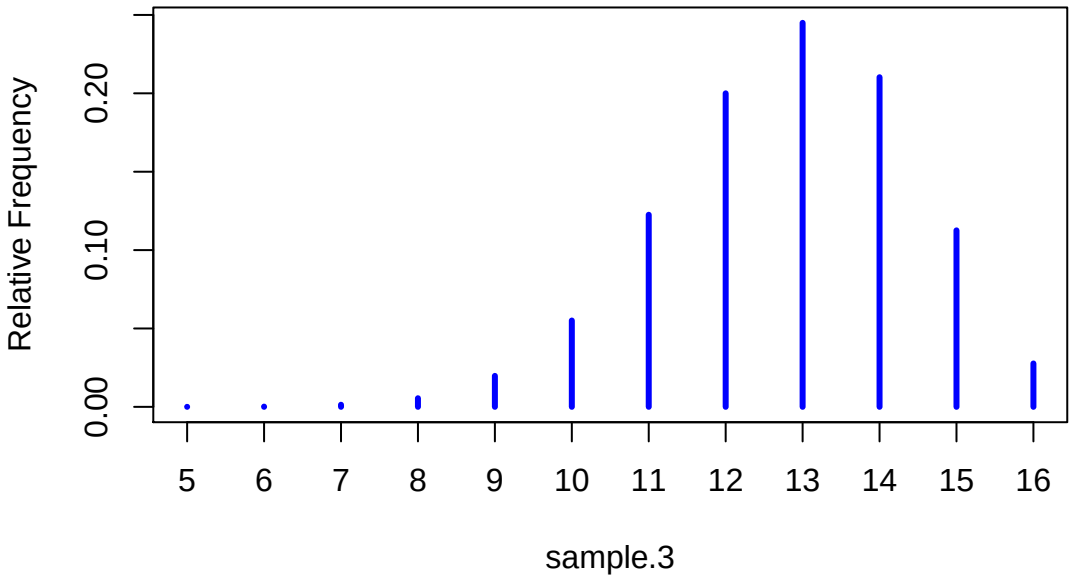
The 'set of coin-flips' distribution. We get the Bernoulli distribution when the set of coin-flips is a singleton.

```
> n.obs <- 1e+05
```

N is the number of participants that flip the coin / decide whether the indefinite has wide scope relative to the universal; *p* is the probability of getting heads / scoping out the indefinite.

```
> N <- 16
> p <- 0.8
> sample.3 <- rbinom(n = n.obs, size = N, prob = p)
> head(sample.3)

[1] 11 12 14 14 13 16

> par(mfrow = c(2, 1))
> hist(sample.3, col = "lightblue", freq = FALSE)
> plot(prop.table(table(sample.3)), lwd = 3, ylab = "Relative Frequency",
+     col = "blue")
```






```
> par(mfrow = c(1, 1))
> table(sample.3)

sample.3
   5    6    7    8    9   10   11   12   13   14   15   16
```

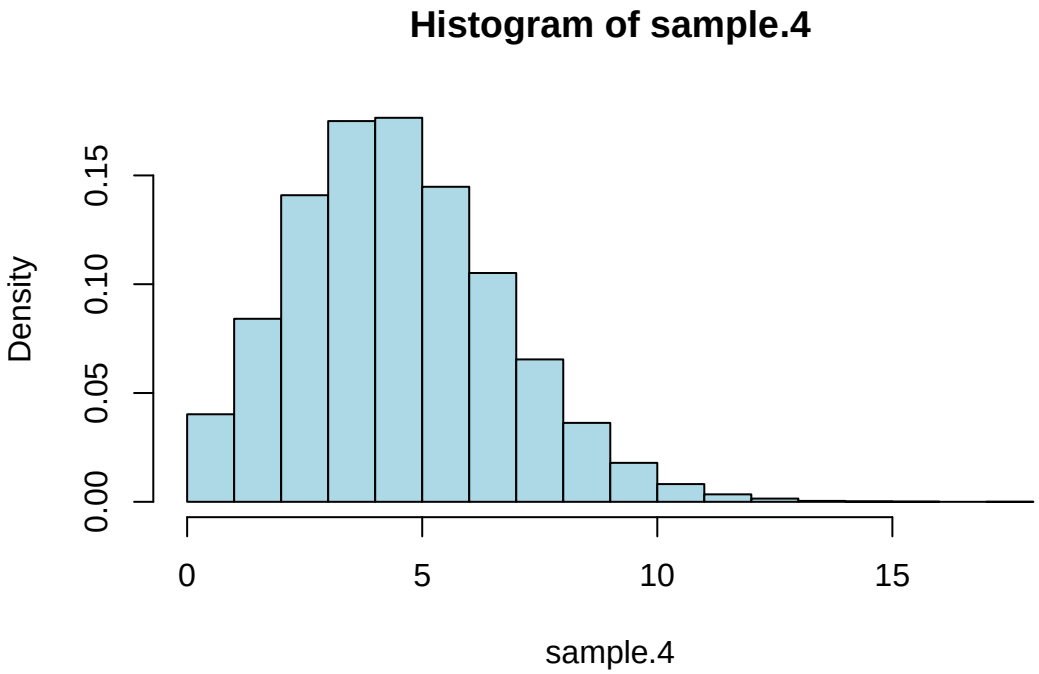
```
    3    17   136   551  1976  5509 12256 20000 24497 21029 11258  2768

> prop.table(table(sample.3))

sample.3
      5       6       7       8       9      10      11      12      13
0.00003 0.00017 0.00136 0.00551 0.01976 0.05509 0.12256 0.20000 0.24497
     14      15      16
0.21029 0.11258 0.02768

> sum(prop.table(table(sample.3)))

[1] 1
```

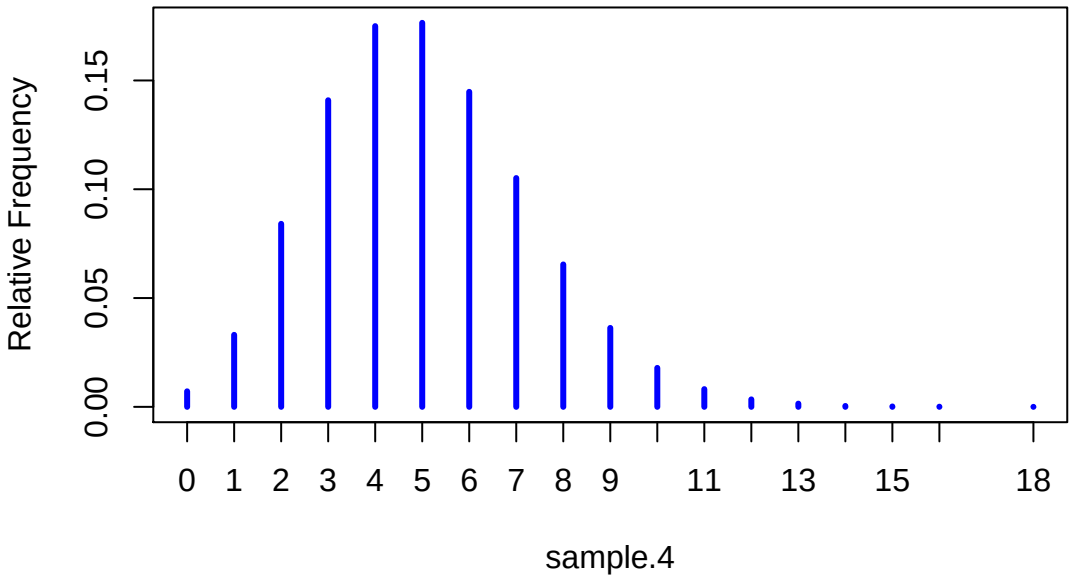
1.1.4 Poisson distributions

```
> n.obs <- 1e+05
```

Poisson distributions are parametrized by their mean rate λ. For example, l below could be the average number of quantifiers (including modals, adverbs etc.) per sentence, i.e., the mean rate of quantifier 'arrivals / events' per unit of 'time / space'.

```
> l <- 5
> sample.4 <- rpois(n = n.obs, lambda = l)
> head(sample.4)

[1] 4 6 5 2 6 3

> par(mfrow = c(2, 1))
> hist(sample.4, col = "lightblue", freq = FALSE)
> plot(prop.table(table(sample.4)), lwd = 3, ylab = "Relative Frequency",
+     col = "blue")
```






```
> par(mfrow = c(1, 1))
```

1.2 The deterministic component of linear models: linear predictors and design matrices

Since we are now familiar with a range of linear models, it is time to learn in more detail exactly what the model descriptions are and how they can be adapted to match our hypotheses. In particular, we introduce the concept of *design matrix* for linear / generalized linear models (GLMs). We will see how the same linear model can be described in different ways; these different descriptions are formalized by means of different design matrices and are called *parameterizations* of a model.

We will learn about design matrices with a toy data set consisting of 8 observations.

```
> y <- response <- c(6, 8, 5, 7, 9, 11, 5.5, 10)
> f1 <- categorical.predictor.1 <- factor(c("level1", "level1", "level1",
+     "level2", "level2", "level2", "level3", "level3"))
> f2 <- categorical.predictor.2 <- factor(c("level1", "level1", "level1",
+     "level2", "level2", "level2", "level2", "level2"))
> f3 <- categorical.predictor.3 <- factor(c("level1", "level2", "level3",
+     "level1", "level2", "level3", "level1", "level2"))
> x <- continuous.covariate <- c(40, 45, 39, 50, 52, 57, 38, 44)
> (toy.data.set <- data.frame(y, f1, f2, f3, x))

     y     f1     f2     f3  x
1  6.0 level1 level1 level1 40
2  8.0 level1 level1 level2 45
3  5.0 level1 level1 level3 39
4  7.0 level2 level2 level1 50
5  9.0 level2 level2 level2 52
6 11.0 level2 level2 level3 57
7  5.5 level3 level2 level1 38
8 10.0 level3 level2 level2 44

> str(toy.data.set)

'data.frame': 8 obs. of  5 variables:
 $ y : num  6 8 5 7 9 11 5.5 10
 $ f1: Factor w/ 3 levels "level1","level2",..: 1 1 1 2 2 2 3 3
 $ f2: Factor w/ 2 levels "level1","level2": 1 1 1 2 2 2 2 2
 $ f3: Factor w/ 3 levels "level1","level2",..: 1 2 3 1 2 3 1 2
 $ x : num  40 45 39 50 52 57 38 44
```

The design matrix has:

- as many columns as the fitted model has parameters
- as many rows as there are observations, i.e., as many rows as there are responses in the response vector

When the design matrix is matrix-multiplied with the parameter vector β, it yields the linear predictor $X \cdot \beta$, another vector:

- the linear predictor contains the expected value of the response given the values of all the explanatory / predictor variables in the model
- expected value means the response that would be observed when all random variation is averaged out; to get actual responses, we add the random variation, a.k.a. the error

We examine a progression of typical linear models: t-tests, simple linear regressions, 1-way and 2-way ANOVAs etc. Note: ANOVAs in this context refer to actual linear models that we use to model data, they

are not used for model comparison. The basic intuition of breaking down the variance (the mean squared error) into components is the same.

This progression will partially overlap with the material we just covered. Our goal is to take a closer look at these models to find out how R represents them internally to estimate their parameters. We will see different ways of writing what is essentially the same model, i.e., different parameterizations of a model, and how these affect the interpretation of the model parameters.

1.2.1 The mean / intercept-only model, i.e., the null model

```
> y

[1]  6.0  8.0  5.0  7.0  9.0 11.0  5.5 10.0

> lm(y ~ 1)


Call:
lm(formula = y ~ 1)

Coefficients:
(Intercept)
       7.69

> round(mean(y), 4)

[1] 7.688

> model.matrix(lm(y ~ 1))

  (Intercept)
1           1
2           1
3           1
4           1
5           1
6           1
7           1
8           1
attr(,"assign")
[1] 0
```

Expected response:

```
> (designMatrix <- model.matrix(lm(y ~ 1)))

  (Intercept)
1           1
2           1
3           1
4           1
5           1
6           1
7           1
8           1
attr(,"assign")
```

```
[1] 0

> (betaVector <- coef(lm(y ~ 1)))

(Intercept)
      7.687

> (expectedResponse <- designMatrix %*% betaVector)

   [,1]
1 7.687
2 7.687
3 7.687
4 7.687
5 7.687
6 7.687
7 7.687
8 7.687

> data.frame(actualResponse = y, predictedResponse = expectedResponse)

  actualResponse predictedResponse
1            6.0             7.687
2            8.0             7.687
3            5.0             7.687
4            7.0             7.687
5            9.0             7.687
6           11.0             7.687
7            5.5             7.687
8           10.0             7.687
```

1.2.2 T-tests

```
> cbind(y, f2)

        y f2
[1,]  6.0  1
[2,]  8.0  1
[3,]  5.0  1
[4,]  7.0  2
[5,]  9.0  2
[6,] 11.0  2
[7,]  5.5  2
[8,] 10.0  2

> lm(y ~ f2)


Call:
lm(formula = y ~ f2)

Coefficients:
(Intercept)     f2level2
       6.33         2.17
```

```
> lm(y ~ 1 + f2)


Call:
lm(formula = y ~ 1 + f2)

Coefficients:
(Intercept)     f2level2
       6.33         2.17

> model.matrix(lm(y ~ f2))

  (Intercept) f2level2
1           1        0
2           1        0
3           1        0
4           1        1
5           1        1
6           1        1
7           1        1
8           1        1
attr(,"assign")
[1] 0 1
attr(,"contrasts")
attr(,"contrasts")$f2
[1] "contr.treatment"
```

Expected response:

```
> (designMatrix <- model.matrix(lm(y ~ f2)))

  (Intercept) f2level2
1           1        0
2           1        0
3           1        0
4           1        1
5           1        1
6           1        1
7           1        1
8           1        1
attr(,"assign")
[1] 0 1
attr(,"contrasts")
attr(,"contrasts")$f2
[1] "contr.treatment"

> (betaVector <- coef(lm(y ~ f2)))

(Intercept)    f2level2
      6.333       2.167

> (expectedResponse <- designMatrix %*% betaVector)

   [,1]
1 6.333
```

```
2 6.333
3 6.333
4 8.500
5 8.500
6 8.500
7 8.500
8 8.500
```

```
> data.frame(actualResponse = y, predictedResponse = expectedResponse)

  actualResponse predictedResponse
1            6.0             6.333
2            8.0             6.333
3            5.0             6.333
4            7.0             8.500
5            9.0             8.500
6           11.0             8.500
7            5.5             8.500
8           10.0             8.500
```

Alternative parametrization

```
> lm(y ~ -1 + f2)


Call:
lm(formula = y ~ -1 + f2)

Coefficients:
f2level1  f2level2
    6.33      8.50

> model.matrix(lm(y ~ -1 + f2))

  f2level1 f2level2
1        1        0
2        1        0
3        1        0
4        0        1
5        0        1
6        0        1
7        0        1
8        0        1
attr(,"assign")
[1] 1 1
attr(,"contrasts")
attr(,"contrasts")$f2
[1] "contr.treatment"
```

Expected response for alternative parametrization:

```
> (designMatrix <- model.matrix(lm(y ~ -1 + f2)))

  f2level1 f2level2
1        1        0
```

```
2        1        0
3        1        0
4        0        1
5        0        1
6        0        1
7        0        1
8        0        1
attr(,"assign")
[1] 1 1
attr(,"contrasts")
attr(,"contrasts")$f2
[1] "contr.treatment"

> (betaVector <- coef(lm(y ~ -1 + f2)))

f2level1 f2level2
   6.333    8.500

> (expectedResponse <- designMatrix %*% betaVector)

   [,1]
1 6.333
2 6.333
3 6.333
4 8.500
5 8.500
6 8.500
7 8.500
8 8.500

> data.frame(actualResponse = y, predictedResponse = expectedResponse)

  actualResponse predictedResponse
1            6.0             6.333
2            8.0             6.333
3            5.0             6.333
4            7.0             8.500
5            9.0             8.500
6           11.0             8.500
7            5.5             8.500
8           10.0             8.500
```

1.2.3 Simple linear regression

```
> cbind(y, x)

        y  x
[1,]  6.0 40
[2,]  8.0 45
[3,]  5.0 39
[4,]  7.0 50
[5,]  9.0 52
[6,] 11.0 57
```

```
[7,]  5.5 38
[8,] 10.0 44

> lm(y ~ x)


Call:
lm(formula = y ~ x)

Coefficients:
(Intercept)            x
     -3.942        0.255

> model.matrix(lm(y ~ x))

  (Intercept)  x
1           1 40
2           1 45
3           1 39
4           1 50
5           1 52
6           1 57
7           1 38
8           1 44
attr(,"assign")
[1] 0 1
```

Expected response:

```
> (designMatrix <- model.matrix(lm(y ~ x)))

  (Intercept)  x
1           1 40
2           1 45
3           1 39
4           1 50
5           1 52
6           1 57
7           1 38
8           1 44
attr(,"assign")
[1] 0 1

> (betaVector <- coef(lm(y ~ x)))

(Intercept)           x
    -3.9419      0.2549

> (expectedResponse <- designMatrix %*% betaVector)

    [,1]
1  6.254
2  7.528
3  5.999
4  8.803
```

```
5  9.312
6 10.587
7  5.744
8  7.273

> data.frame(actualResponse = y, predictedResponse = round(expectedResponse,
+     2))

  actualResponse predictedResponse
1            6.0              6.25
2            8.0              7.53
3            5.0              6.00
4            7.0              8.80
5            9.0              9.31
6           11.0             10.59
7            5.5              5.74
8           10.0              7.27
```

1.2.4 One-way ANOVA

As a linear model, one-way ANOVA generalizes the t-test: the predictor is a categorical variable (factor) with more than 2 outcomes (levels).

Effects parameterization (R default):

```
> cbind(y, f1)

        y f1
[1,]  6.0  1
[2,]  8.0  1
[3,]  5.0  1
[4,]  7.0  2
[5,]  9.0  2
[6,] 11.0  2
[7,]  5.5  3
[8,] 10.0  3

> lm(y ~ f1)


Call:
lm(formula = y ~ f1)

Coefficients:
(Intercept)     f1level2     f1level3
       6.33         2.67         1.42

> model.matrix(lm(y ~ f1))

  (Intercept) f1level2 f1level3
1           1        0        0
2           1        0        0
3           1        0        0
4           1        1        0
5           1        1        0
```

```
6           1        1        0
7           1        0        1
8           1        0        1
attr(,"assign")
[1] 0 1 1
attr(,"contrasts")
attr(,"contrasts")$f1
[1] "contr.treatment"
```

Expected response for effects parametrization:

```
> (designMatrix <- model.matrix(lm(y ~ f1)))

  (Intercept) f1level2 f1level3
1           1        0        0
2           1        0        0
3           1        0        0
4           1        1        0
5           1        1        0
6           1        1        0
7           1        0        1
8           1        0        1
attr(,"assign")
[1] 0 1 1
attr(,"contrasts")
attr(,"contrasts")$f1
[1] "contr.treatment"

> (betaVector <- coef(lm(y ~ f1)))

(Intercept)    f1level2    f1level3
      6.333       2.667       1.417

> (expectedResponse <- designMatrix %*% betaVector)

   [,1]
1 6.333
2 6.333
3 6.333
4 9.000
5 9.000
6 9.000
7 7.750
8 7.750

> data.frame(actualResponse = y, predictedResponse = expectedResponse)

  actualResponse predictedResponse
1            6.0             6.333
2            8.0             6.333
3            5.0             6.333
4            7.0             9.000
5            9.0             9.000
6           11.0             9.000
7            5.5             7.750
8           10.0             7.750
```

Means parameterization:

```
> lm(y ~ -1 + f1)


Call:
lm(formula = y ~ -1 + f1)

Coefficients:
f1level1  f1level2  f1level3
    6.33      9.00      7.75

> model.matrix(lm(y ~ -1 + f1))

  f1level1 f1level2 f1level3
1        1        0        0
2        1        0        0
3        1        0        0
4        0        1        0
5        0        1        0
6        0        1        0
7        0        0        1
8        0        0        1
attr(,"assign")
[1] 1 1 1
attr(,"contrasts")
attr(,"contrasts")$f1
[1] "contr.treatment"
```

Expected response for means parametrization:

```
> (designMatrix <- model.matrix(lm(y ~ -1 + f1)))

  f1level1 f1level2 f1level3
1        1        0        0
2        1        0        0
3        1        0        0
4        0        1        0
5        0        1        0
6        0        1        0
7        0        0        1
8        0        0        1
attr(,"assign")
[1] 1 1 1
attr(,"contrasts")
attr(,"contrasts")$f1
[1] "contr.treatment"

> (betaVector <- coef(lm(y ~ -1 + f1)))

f1level1 f1level2 f1level3
   6.333    9.000    7.750

> (expectedResponse <- designMatrix %*% betaVector)
```

```
   [,1]
1 6.333
2 6.333
3 6.333
4 9.000
5 9.000
6 9.000
7 7.750
8 7.750

> data.frame(actualResponse = y, predictedResponse = expectedResponse)

  actualResponse predictedResponse
1            6.0             6.333
2            8.0             6.333
3            5.0             6.333
4            7.0             9.000
5            9.0             9.000
6           11.0             9.000
7            5.5             7.750
8           10.0             7.750
```

1.2.5 Two-way ANOVA

Two-way ANOVAs are a further generalizations of t-tests / one-way ANOVAs: we allow for multiple categorical predictors.

```
> cbind(y, f2, f3)

        y f2 f3
[1,]  6.0  1  1
[2,]  8.0  1  2
[3,]  5.0  1  3
[4,]  7.0  2  1
[5,]  9.0  2  2
[6,] 11.0  2  3
[7,]  5.5  2  1
[8,] 10.0  2  2
```

Effects parametrization, no interactions:

```
> lm(y ~ f2 + f3)


Call:
lm(formula = y ~ f2 + f3)

Coefficients:
(Intercept)     f2level2     f3level2     f3level3
       4.65         2.27         2.83         2.21

> model.matrix(lm(y ~ f2 + f3))

  (Intercept) f2level2 f3level2 f3level3
```

```
1           1        0        0        0
2           1        0        1        0
3           1        0        0        1
4           1        1        0        0
5           1        1        1        0
6           1        1        0        1
7           1        1        0        0
8           1        1        1        0
attr(,"assign")
[1] 0 1 2 2
attr(,"contrasts")
attr(,"contrasts")$f2
[1] "contr.treatment"

attr(,"contrasts")$f3
[1] "contr.treatment"
```

What is the interpretation of the coefficients? How is the intercept coefficient different from all the other coefficients, a.k.a. the main effects?

Expected response for effects parametrization, no interactions:

```
> (designMatrix <- model.matrix(lm(y ~ f2 + f3)))

  (Intercept) f2level2 f3level2 f3level3
1           1        0        0        0
2           1        0        1        0
3           1        0        0        1
4           1        1        0        0
5           1        1        1        0
6           1        1        0        1
7           1        1        0        0
8           1        1        1        0
attr(,"assign")
[1] 0 1 2 2
attr(,"contrasts")
attr(,"contrasts")$f2
[1] "contr.treatment"

attr(,"contrasts")$f3
[1] "contr.treatment"

> (betaVector <- coef(lm(y ~ f2 + f3)))

(Intercept)    f2level2    f3level2    f3level3
      4.652       2.273       2.833       2.212

> (expectedResponse <- designMatrix %*% betaVector)

   [,1]
1 4.652
2 7.485
3 6.864
4 6.924
5 9.758
6 9.136
```

```
7 6.924
8 9.758

> data.frame(actualResponse = y, predictedResponse = expectedResponse)

  actualResponse predictedResponse
1            6.0             4.652
2            8.0             7.485
3            5.0             6.864
4            7.0             6.924
5            9.0             9.758
6           11.0             9.136
7            5.5             6.924
8           10.0             9.758
```

Effects parametrization, with interactions:

```
> lm(y ~ f2 * f3)


Call:
lm(formula = y ~ f2 * f3)

Coefficients:
      (Intercept)           f2level2           f3level2
             6.00               0.25               2.00
        f3level3  f2level2:f3level2  f2level2:f3level3
            -1.00               1.25               5.75

> model.matrix(lm(y ~ f2 * f3))

  (Intercept) f2level2 f3level2 f3level3 f2level2:f3level2
1           1        0        0        0                 0
2           1        0        1        0                 0
3           1        0        0        1                 0
4           1        1        0        0                 0
5           1        1        1        0                 1
6           1        1        0        1                 0
7           1        1        0        0                 0
8           1        1        1        0                 1
  f2level2:f3level3
1                 0
2                 0
3                 0
4                 0
5                 0
6                 1
7                 0
8                 0
attr(,"assign")
[1] 0 1 2 2 3 3
attr(,"contrasts")
attr(,"contrasts")$f2
[1] "contr.treatment"
```

```
attr(,"contrasts")$f3
[1] "contr.treatment"
```

What is the interpretation of the coefficients now? How is the intercept coefficient different from all the other coefficients, a.k.a. the main effects and the interaction effects? How are the main effects different from the interaction effects?

Expected response for effects parametrization, with interactions:

```
> (designMatrix <- model.matrix(lm(y ~ f2 * f3)))

  (Intercept) f2level2 f3level2 f3level3 f2level2:f3level2
1           1        0        0        0                 0
2           1        0        1        0                 0
3           1        0        0        1                 0
4           1        1        0        0                 0
5           1        1        1        0                 1
6           1        1        0        1                 0
7           1        1        0        0                 0
8           1        1        1        0                 1
  f2level2:f3level3
1                 0
2                 0
3                 0
4                 0
5                 0
6                 1
7                 0
8                 0
attr(,"assign")
[1] 0 1 2 2 3 3
attr(,"contrasts")
attr(,"contrasts")$f2
[1] "contr.treatment"

attr(,"contrasts")$f3
[1] "contr.treatment"

> (betaVector <- coef(lm(y ~ f2 * f3)))

      (Intercept)          f2level2          f3level2          f3level3
             6.00              0.25              2.00             -1.00
f2level2:f3level2 f2level2:f3level3
             1.25              5.75

> (expectedResponse <- designMatrix %*% betaVector)

   [,1]
1  6.00
2  8.00
3  5.00
4  6.25
5  9.50
6 11.00
7  6.25
8  9.50
```

```
> data.frame(actualResponse = y, predictedResponse = expectedResponse)

  actualResponse predictedResponse
1            6.0             6.00
2            8.0             8.00
3            5.0             5.00
4            7.0             6.25
5            9.0             9.50
6           11.0            11.00
7            5.5             6.25
8           10.0             9.50
```

Means parametrization, with interactions:

```
> lm(y ~ -1 + f2 * f3 - f2 - f3)


Call:
lm(formula = y ~ -1 + f2 * f3 - f2 - f3)

Coefficients:
f2level1:f3level1  f2level2:f3level1  f2level1:f3level2
             6.00               6.25               8.00
f2level2:f3level2  f2level1:f3level3  f2level2:f3level3
             9.50               5.00              11.00

> model.matrix(lm(y ~ -1 + f2 * f3 - f2 - f3))

  f2level1:f3level1 f2level2:f3level1 f2level1:f3level2 f2level2:f3level2
1                 1                 0                 0                 0
2                 0                 0                 1                 0
3                 0                 0                 0                 0
4                 0                 1                 0                 0
5                 0                 0                 0                 1
6                 0                 0                 0                 0
7                 0                 1                 0                 0
8                 0                 0                 0                 1
  f2level1:f3level3 f2level2:f3level3
1                 0                 0
2                 0                 0
3                 1                 0
4                 0                 0
5                 0                 0
6                 0                 1
7                 0                 0
8                 0                 0
attr(,"assign")
[1] 1 1 1 1 1 1
attr(,"contrasts")
attr(,"contrasts")$f2
[1] "contr.treatment"

attr(,"contrasts")$f3
[1] "contr.treatment"
```

Expected response for means parametrization, with interactions:

```
> (designMatrix <- model.matrix(lm(y ~ -1 + f2 * f3 - f2 - f3)))

  f2level1:f3level1 f2level2:f3level1 f2level1:f3level2 f2level2:f3level2
1                 1                 0                 0                 0
2                 0                 0                 1                 0
3                 0                 0                 0                 0
4                 0                 1                 0                 0
5                 0                 0                 0                 1
6                 0                 0                 0                 0
7                 0                 1                 0                 0
8                 0                 0                 0                 1
  f2level1:f3level3 f2level2:f3level3
1                 0                 0
2                 0                 0
3                 1                 0
4                 0                 0
5                 0                 0
6                 0                 1
7                 0                 0
8                 0                 0
attr(,"assign")
[1] 1 1 1 1 1 1
attr(,"contrasts")
attr(,"contrasts")$f2
[1] "contr.treatment"

attr(,"contrasts")$f3
[1] "contr.treatment"

> (betaVector <- coef(lm(y ~ -1 + f2 * f3 - f2 - f3)))

f2level1:f3level1 f2level2:f3level1 f2level1:f3level2 f2level2:f3level2
             6.00              6.25              8.00              9.50
f2level1:f3level3 f2level2:f3level3
             5.00             11.00

> (expectedResponse <- designMatrix %*% betaVector)

   [,1]
1  6.00
2  8.00
3  5.00
4  6.25
5  9.50
6 11.00
7  6.25
8  9.50

> data.frame(actualResponse = y, predictedResponse = expectedResponse)

  actualResponse predictedResponse
1            6.0             6.00
2            8.0             8.00
```

```
3           5.0             5.00
4           7.0             6.25
5           9.0             9.50
6          11.0            11.00
7           5.5             6.25
8          10.0             9.50
```

1.3 ANCOVA

ANCOVA stands for analysis of co-variance. It is a linear model that has both categorical and continuous predictors. Because it has categorical predictors, it's like an ANOVA. The continuous predictors are called *covariates*, hence the name of ANCOVA. Our last 4 sets of lecture notes or so discussed an ANCOVA model for weight (our old y) that had height (x_1) as a covariate / continous predictor and gender (x_2) as a categorical predictor.

Main effects model:

```
> lm(y ~ f1 + x)


Call:
lm(formula = y ~ f1 + x)

Coefficients:
(Intercept)     f1level2     f1level3            x
    -17.423       -4.039        1.608        0.575
```

Main effects + interactions:

```
> lm(y ~ f1 * x)


Call:
lm(formula = y ~ f1 * x)

Coefficients:
(Intercept)     f1level2     f1level3            x   f1level2:x
   -13.0000      -6.5385     -10.0000       0.4677       0.0707
 f1level3:x
     0.2823
```

Same, R's way of specifying the interaction term separately:

```
> lm(y ~ f1 + x + f1:x)


Call:
lm(formula = y ~ f1 + x + f1:x)

Coefficients:
(Intercept)     f1level2     f1level3            x   f1level2:x
   -13.0000      -6.5385     -10.0000       0.4677       0.0707
 f1level3:x
     0.2823
```

Effects parametrization:

```
> model.matrix(lm(y ~ f1 + x))

  (Intercept) f1level2 f1level3  x
1           1        0        0 40
2           1        0        0 45
3           1        0        0 39
4           1        1        0 50
5           1        1        0 52
6           1        1        0 57
7           1        0        1 38
8           1        0        1 44
attr(,"assign")
[1] 0 1 1 2
attr(,"contrasts")
attr(,"contrasts")$f1
[1] "contr.treatment"

> model.matrix(lm(y ~ f1 * x))

  (Intercept) f1level2 f1level3  x f1level2:x f1level3:x
1           1        0        0 40          0          0
2           1        0        0 45          0          0
3           1        0        0 39          0          0
4           1        1        0 50         50          0
5           1        1        0 52         52          0
6           1        1        0 57         57          0
7           1        0        1 38          0         38
8           1        0        1 44          0         44
attr(,"assign")
[1] 0 1 1 2 3 3
attr(,"contrasts")
attr(,"contrasts")$f1
[1] "contr.treatment"
```

Means parametrization:

```
> model.matrix(lm(y ~ f1 + x - 1))

  f1level1 f1level2 f1level3  x
1        1        0        0 40
2        1        0        0 45
3        1        0        0 39
4        0        1        0 50
5        0        1        0 52
6        0        1        0 57
7        0        0        1 38
8        0        0        1 44
attr(,"assign")
[1] 1 1 1 2
attr(,"contrasts")
attr(,"contrasts")$f1
[1] "contr.treatment"

> model.matrix(lm(y ~ (f1 * x - 1 - x)))
```

```
  f1level1 f1level2 f1level3 f1level1:x f1level2:x f1level3:x
1        1        0        0         40          0          0
2        1        0        0         45          0          0
3        1        0        0         39          0          0
4        0        1        0          0         50          0
5        0        1        0          0         52          0
6        0        1        0          0         57          0
7        0        0        1          0          0         38
8        0        0        1          0          0         44
attr(,"assign")
[1] 1 1 1 2 2 2
attr(,"contrasts")
attr(,"contrasts")$f1
[1] "contr.treatment"
```

2 T-tests: more realistic examples

This and all the remaining sections in this set of lecture notes will discuss more realistic examples for all the linear models mentioned above. In each case, we will simulate the data, run the analysis in R, and compare the true parameter values (the ones we used to simulate the data) and the parameter estimates obtained from the sample of simulated data.

The combination of simulating the data, examining the design matrices of the R models, and finally comparing the true and estimated values of the parameters will provide a hands-on, in-depth understanding of all these fundamental models. To make sure you fully graps these models, you should simulate new data sets with different kinds of predictors and combinations of predictors, and estimate R models for these new data sets. For example, you could think of an experiment you want to run, and then figure out at least one way to simulate data that would be very similar to what you will get when you actually run the experiment.

2.1 T-test with equal variances

We start with the data generation.

Observations in group / treatment 1:

```
> n1 <- 60
```

Observations in group / treatment 2:

```
> n2 <- 40
```

Mean for group 1:

```
> mu1 <- 105
```

Mean for group 2:

```
> mu2 <- 77.5
```

Average population sd for both groups / treatments:

```
> sigma <- 2.75
```

Total sample size:

```
> (n <- n1 + n2)

[1] 100
```

Indicator / dummy variable for group 2:

```
> (x <- rep(c(0, 1), c(n1, n2)))

  [1] 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 0 0 0 0 0 0 0 0 0
 [36] 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 1 1 1 1 1 1 1 1 1 1 1 1
 [71] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

Mean for group 1 serves as the intercept β_0:

```
> (beta.0 <- mu1)

[1] 105
```

β_1 is the difference $group_2 - group_1$:

```
> (beta.1 <- mu2 - mu1)

[1] -27.5
```

Expectation (deterministic part of the model):

```
> (E.y <- beta.0 + beta.1 * x)

  [1] 105.0 105.0 105.0 105.0 105.0 105.0 105.0 105.0 105.0 105.0 105.0
 [12] 105.0 105.0 105.0 105.0 105.0 105.0 105.0 105.0 105.0 105.0 105.0
 [23] 105.0 105.0 105.0 105.0 105.0 105.0 105.0 105.0 105.0 105.0 105.0
 [34] 105.0 105.0 105.0 105.0 105.0 105.0 105.0 105.0 105.0 105.0 105.0
 [45] 105.0 105.0 105.0 105.0 105.0 105.0 105.0 105.0 105.0 105.0 105.0
 [56] 105.0 105.0 105.0 105.0 105.0  77.5  77.5  77.5  77.5  77.5  77.5
 [67]  77.5  77.5  77.5  77.5  77.5  77.5  77.5  77.5  77.5  77.5  77.5
 [78]  77.5  77.5  77.5  77.5  77.5  77.5  77.5  77.5  77.5  77.5  77.5
 [89]  77.5  77.5  77.5  77.5  77.5  77.5  77.5  77.5  77.5  77.5  77.5
[100]  77.5
```

Add random variation (stochastic part of the model):

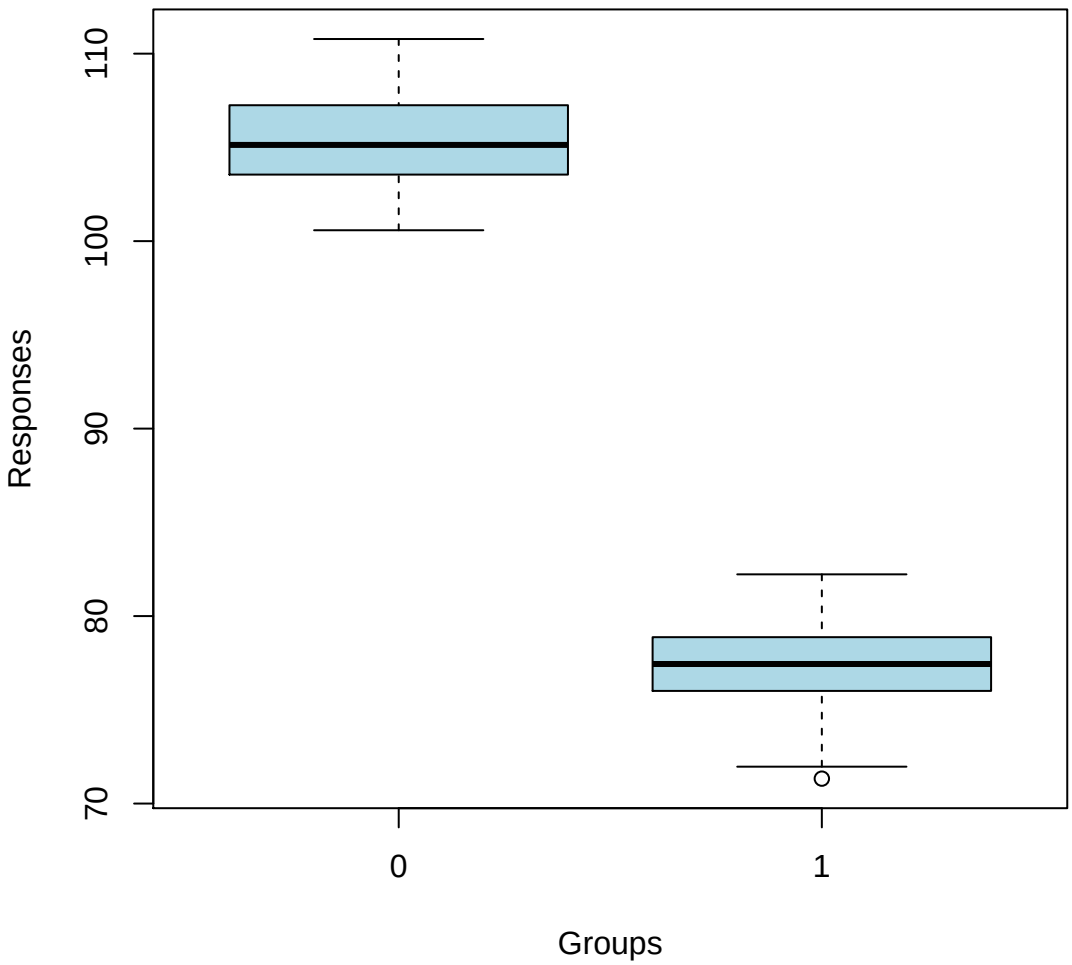
```
> y.obs <- rnorm(n = n, mean = E.y, sd = sigma)
```

We take a look at the final data set:

```
> cbind(round(y.obs, 2), E.y, x)

                E.y x
  [1,] 110.78 105.0 0
  [2,] 108.85 105.0 0
  [3,] 104.38 105.0 0
  [4,] 104.24 105.0 0
  [5,] 106.67 105.0 0
  [6,] 102.37 105.0 0
  [7,] 110.30 105.0 0
```

```
 [8,] 107.59 105.0 0
 [9,] 105.93 105.0 0
[10,] 108.18 105.0 0
[11,] 104.57 105.0 0
[12,] 104.02 105.0 0
[13,] 103.87 105.0 0
[14,] 105.00 105.0 0
[15,] 102.54 105.0 0
[16,] 105.40 105.0 0
[17,] 101.53 105.0 0
[18,] 102.93 105.0 0
[19,] 103.66 105.0 0
[20,] 104.37 105.0 0
[21,] 104.66 105.0 0
[22,] 102.12 105.0 0
[23,] 104.84 105.0 0
[24,] 106.23 105.0 0
[25,] 105.47 105.0 0
[26,] 105.18 105.0 0
[27,] 103.44 105.0 0
[28,] 107.22 105.0 0
[29,] 100.58 105.0 0
[30,] 107.27 105.0 0
[31,] 102.46 105.0 0
[32,] 108.91 105.0 0
[33,] 107.23 105.0 0
[34,] 109.02 105.0 0
[35,] 102.10 105.0 0
[36,] 103.11 105.0 0
[37,] 108.43 105.0 0
[38,] 108.13 105.0 0
[39,] 105.09 105.0 0
[40,] 106.90 105.0 0
[41,] 107.11 105.0 0
[42,] 104.46 105.0 0
[43,] 107.59 105.0 0
[44,] 105.80 105.0 0
[45,] 102.42 105.0 0
[46,] 105.79 105.0 0
[47,] 107.11 105.0 0
[48,] 109.07 105.0 0
[49,] 106.66 105.0 0
[50,] 105.06 105.0 0
[51,] 110.21 105.0 0
[52,] 106.65 105.0 0
[53,] 100.65 105.0 0
[54,] 102.14 105.0 0
[55,] 102.51 105.0 0
[56,] 102.02 105.0 0
[57,] 104.85 105.0 0
[58,] 107.89 105.0 0
[59,] 104.67 105.0 0
[60,] 108.78 105.0 0
```

```
 [61,]  77.85  77.5 1
 [62,]  76.88  77.5 1
 [63,]  76.48  77.5 1
 [64,]  78.87  77.5 1
 [65,]  72.64  77.5 1
 [66,]  78.88  77.5 1
 [67,]  76.78  77.5 1
 [68,]  80.24  77.5 1
 [69,]  76.67  77.5 1
 [70,]  77.46  77.5 1
 [71,]  78.50  77.5 1
 [72,]  73.26  77.5 1
 [73,]  81.53  77.5 1
 [74,]  82.23  77.5 1
 [75,]  75.00  77.5 1
 [76,]  76.48  77.5 1
 [77,]  71.97  77.5 1
 [78,]  74.94  77.5 1
 [79,]  80.09  77.5 1
 [80,]  75.87  77.5 1
 [81,]  76.13  77.5 1
 [82,]  79.35  77.5 1
 [83,]  78.61  77.5 1
 [84,]  77.91  77.5 1
 [85,]  77.42  77.5 1
 [86,]  71.33  77.5 1
 [87,]  76.87  77.5 1
 [88,]  79.15  77.5 1
 [89,]  79.51  77.5 1
 [90,]  79.28  77.5 1
 [91,]  76.58  77.5 1
 [92,]  78.54  77.5 1
 [93,]  80.43  77.5 1
 [94,]  77.95  77.5 1
 [95,]  78.33  77.5 1
 [96,]  78.49  77.5 1
 [97,]  76.10  77.5 1
 [98,]  75.92  77.5 1
 [99,]  75.18  77.5 1
[100,]  75.43  77.5 1

> plot(as.factor(x), y.obs, col = "lightblue", xlab = "Groups", ylab = "Responses")
```



Analysis:

```
> fit1 <- lm(y.obs ~ x)
> summary(fit1)


Call:
lm(formula = y.obs ~ x)

Residuals:
   Min     1Q Median     3Q    Max
-5.948 -1.465 -0.162  1.775  5.331

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  105.450      0.322   327.7   <2e-16 ***
x            -28.172      0.509   -55.4   <2e-16 ***
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.49 on 98 degrees of freedom
Multiple R-squared:  0.969,Adjusted R-squared:  0.969
F-statistic: 3.07e+03 on 1 and 98 DF,  p-value: <2e-16

> model.matrix(fit1)

    (Intercept) x
1             1 0
2             1 0
3             1 0
4             1 0
5             1 0
6             1 0
7             1 0
8             1 0
9             1 0
10            1 0
11            1 0
12            1 0
13            1 0
14            1 0
15            1 0
16            1 0
17            1 0
18            1 0
19            1 0
20            1 0
21            1 0
22            1 0
23            1 0
24            1 0
25            1 0
26            1 0
27            1 0
28            1 0
29            1 0
30            1 0
31            1 0
32            1 0
33            1 0
34            1 0
35            1 0
36            1 0
37            1 0
38            1 0
39            1 0
40            1 0
41            1 0
42            1 0
43            1 0
44            1 0
45            1 0
```

```
46              1 0
47              1 0
48              1 0
49              1 0
50              1 0
51              1 0
52              1 0
53              1 0
54              1 0
55              1 0
56              1 0
57              1 0
58              1 0
59              1 0
60              1 0
61              1 1
62              1 1
63              1 1
64              1 1
65              1 1
66              1 1
67              1 1
68              1 1
69              1 1
70              1 1
71              1 1
72              1 1
73              1 1
74              1 1
75              1 1
76              1 1
77              1 1
78              1 1
79              1 1
80              1 1
81              1 1
82              1 1
83              1 1
84              1 1
85              1 1
86              1 1
87              1 1
88              1 1
89              1 1
90              1 1
91              1 1
92              1 1
93              1 1
94              1 1
95              1 1
96              1 1
97              1 1
98              1 1
```

```
99            1 1
100           1 1
attr(,"assign")
[1] 0 1

> t.test(y.obs ~ x)


Welch Two Sample t-test

data:  y.obs by x
t = 55.9, df = 86.39, p-value < 2.2e-16
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 27.17 29.17
sample estimates:
mean in group 0 mean in group 1
         105.45           77.28

> t.test(y.obs ~ x, var.equal = T)


Two Sample t-test

data:  y.obs by x
t = 55.37, df = 98, p-value < 2.2e-16
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 27.16 29.18
sample estimates:
mean in group 0 mean in group 1
         105.45           77.28
```

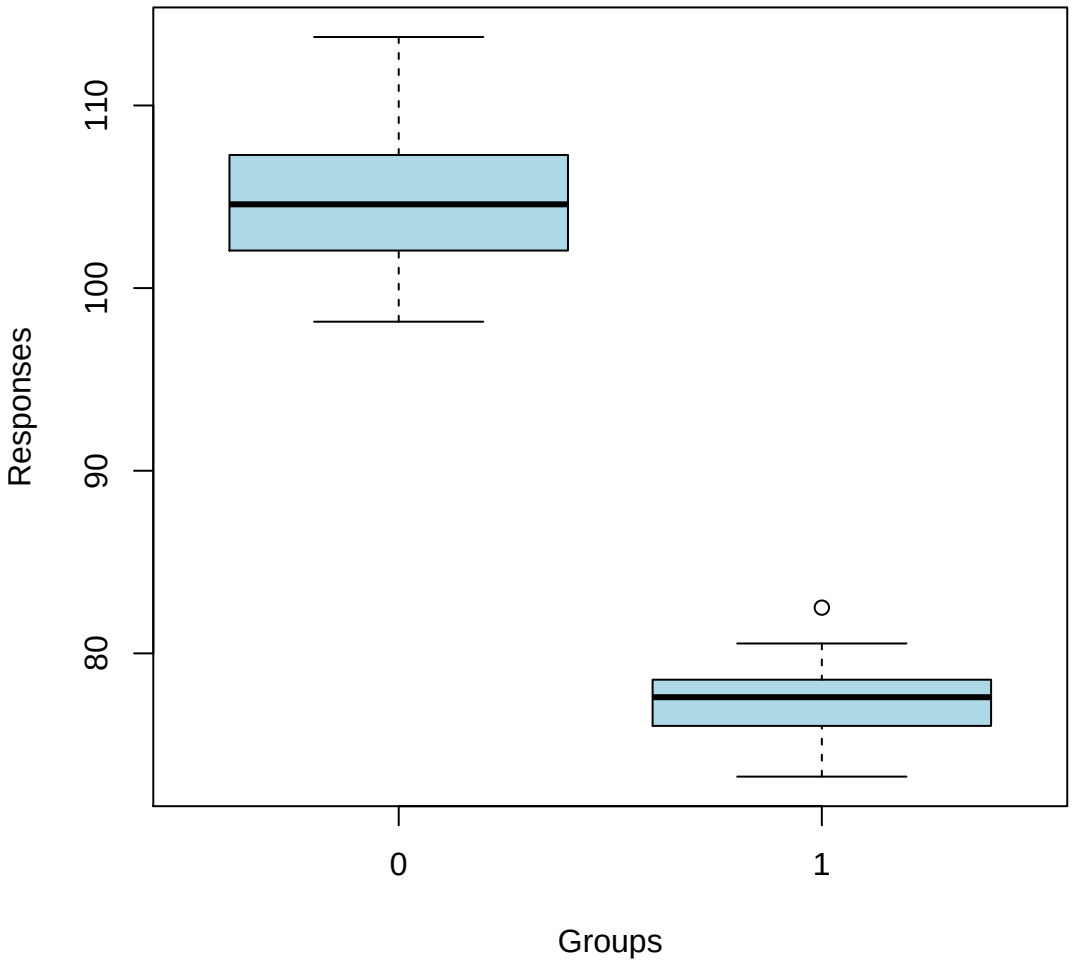
2.2 T-test with unequal variances

Data generation:

```
> n1 <- 60  # Number of observations in group 1
> n2 <- 40  # Number of observations in group 1
> mu1 <- 105  # Population mean for group 1
> mu2 <- 77.5  # Population mean for group 2
> sigma1 <- 3  # Population SD for group 1
> sigma2 <- 2.5  # Population SD for group 2
> n <- n1 + n2  # Total sample size
> y1 <- rnorm(n1, mu1, sigma1)  # Data for group 1
> y2 <- rnorm(n2, mu2, sigma2)  # Data for group 2
> y <- c(y1, y2)  # Aggregate both data sets
> x <- rep(c(0, 1), c(n1, n2))  # Indicator for group 2
> cbind(round(y, 2), x)  # The final data set

              x
  [1,] 102.07 0
  [2,] 109.56 0
  [3,] 102.05 0
```

```
 [4,] 100.49 0
 [5,] 109.15 0
 [6,] 109.63 0
 [7,]  98.16 0
 [8,] 104.22 0
 [9,] 102.29 0
[10,] 106.59 0
[11,] 103.26 0
[12,] 105.25 0
[13,] 108.44 0
[14,] 105.73 0
[15,] 106.70 0
[16,] 107.70 0
[17,] 102.66 0
[18,] 100.11 0
[19,] 100.28 0
[20,] 104.78 0
[21,] 109.54 0
[22,] 101.18 0
[23,] 105.31 0
[24,] 107.76 0
[25,] 108.67 0
[26,] 100.14 0
[27,] 108.75 0
[28,] 103.82 0
[29,] 100.36 0
[30,] 105.00 0
[31,] 100.42 0
[32,] 101.04 0
[33,] 102.17 0
[34,]  99.04 0
[35,] 110.12 0
[36,] 104.14 0
[37,] 104.98 0
[38,] 105.83 0
[39,] 107.12 0
[40,] 113.75 0
[41,] 102.69 0
[42,] 102.07 0
[43,] 101.28 0
[44,]  98.69 0
[45,] 102.10 0
[46,] 100.53 0
[47,] 105.81 0
[48,] 107.77 0
[49,] 103.77 0
[50,] 104.17 0
[51,] 108.42 0
[52,] 103.37 0
[53,] 105.09 0
[54,] 108.45 0
[55,] 107.13 0
[56,] 104.40 0
```

```
 [57,] 100.57 0
 [58,] 105.81 0
 [59,] 107.20 0
 [60,] 107.38 0
 [61,]  79.70 1
 [62,]  76.97 1
 [63,]  76.99 1
 [64,]  75.49 1
 [65,]  75.68 1
 [66,]  80.07 1
 [67,]  77.57 1
 [68,]  78.86 1
 [69,]  77.09 1
 [70,]  74.61 1
 [71,]  78.31 1
 [72,]  78.62 1
 [73,]  82.51 1
 [74,]  80.30 1
 [75,]  75.76 1
 [76,]  74.30 1
 [77,]  76.62 1
 [78,]  77.03 1
 [79,]  74.73 1
 [80,]  77.81 1
 [81,]  78.14 1
 [82,]  78.01 1
 [83,]  77.25 1
 [84,]  78.42 1
 [85,]  77.05 1
 [86,]  79.47 1
 [87,]  76.30 1
 [88,]  80.16 1
 [89,]  78.49 1
 [90,]  78.34 1
 [91,]  77.63 1
 [92,]  77.57 1
 [93,]  74.53 1
 [94,]  78.13 1
 [95,]  74.28 1
 [96,]  80.54 1
 [97,]  73.25 1
 [98,]  78.21 1
 [99,]  73.34 1
[100,]  79.10 1

> plot(as.factor(x), y, col = "lightblue", xlab = "Groups", ylab = "Responses")
```



Analysis:

```
> t.test(y ~ x)


Welch Two Sample t-test

data:  y by x
t = 49.16, df = 97.29, p-value < 2.2e-16
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 26.05 28.25
sample estimates:
mean in group 0 mean in group 1
         104.58           77.43
```

3 Simple linear regression

Parameters and predictor for data generation:

```
> n <- 16  # Number of observations
> a <- 40  # Intercept
> b <- -1.5  # Slope
> sigma2 <- 25  # Residual variance
> x <- 1:16  # Values of covariate
```

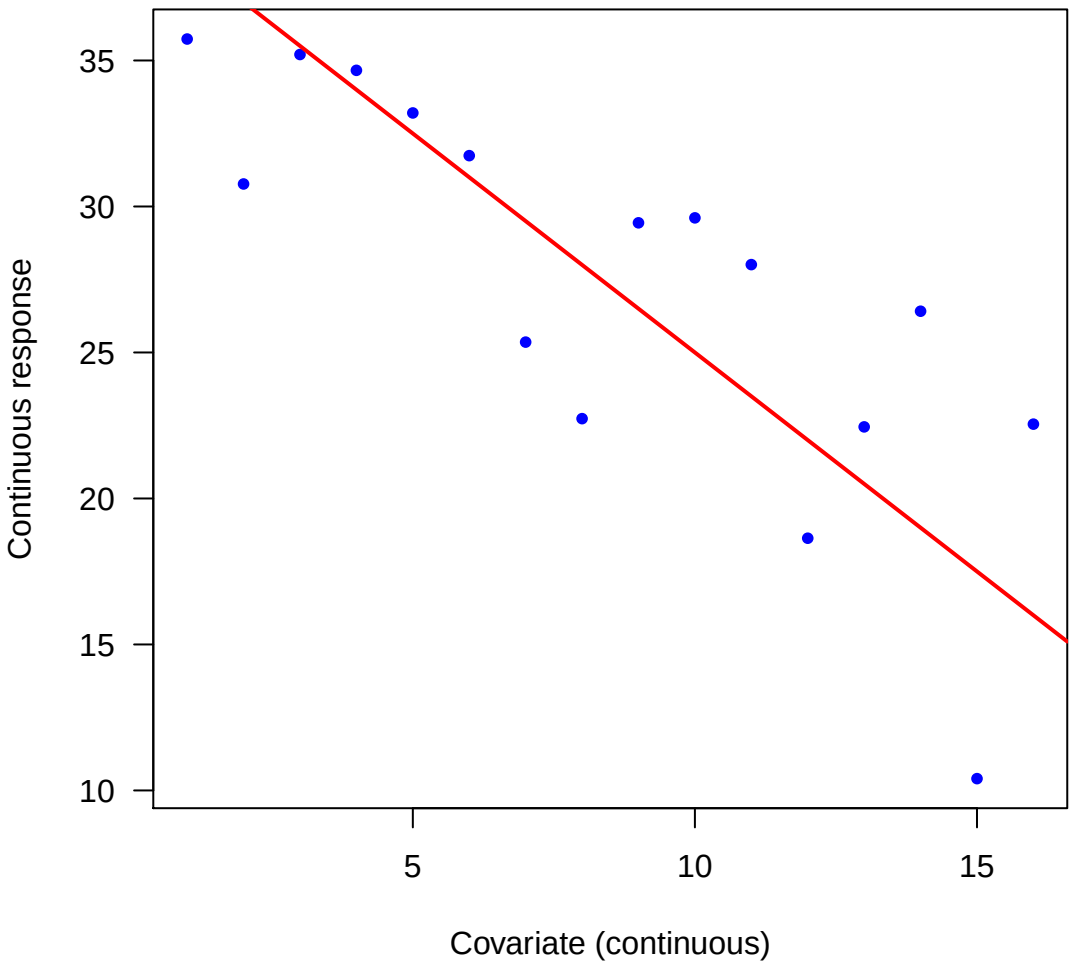
We assemble data set:

```
> normal.error <- rnorm(n, mean = 0, sd = sqrt(sigma2))
> (y <- a + b * x + normal.error)

 [1] 26.426 27.536 36.189 33.977 26.435 29.406 25.117 27.879 23.332 21.615
[11] 25.080 22.518 19.787 22.801 10.887  9.415
```

Alternatively:

```
> (y <- rnorm(n, a + b * x, sd = sqrt(sigma2)))

 [1] 35.74 30.77 35.20 34.66 33.20 31.74 25.36 22.73 29.44 29.61 28.01
[12] 18.64 22.45 26.41 10.41 22.55
```

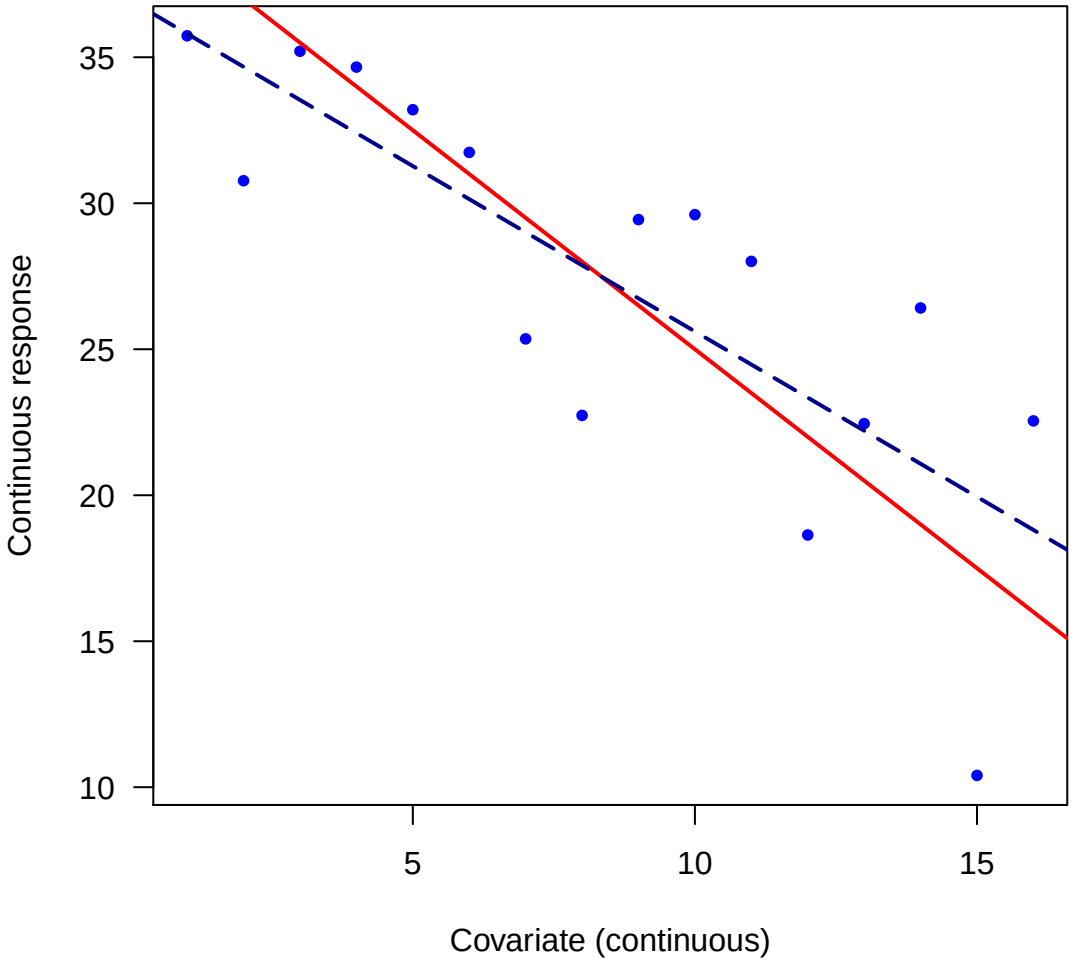
```
> plot(x, y, xlab = "Covariate (continuous)", las = 1, ylab = "Continuous response",
+     cex = 1, pch = 20, col = "blue")
> abline(c(a, b), col = "red", lwd = 2)
```



Analysis:

```
> summary(lm(y ~ x))


Call:
lm(formula = y ~ x)

Residuals:
   Min     1Q Median     3Q    Max
 -9.54  -3.71   1.63   2.91   5.34

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   36.937      2.262   16.33  1.6e-10 ***
x             -1.133      0.234   -4.84  0.00026 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 4.31 on 14 degrees of freedom
Multiple R-squared:  0.626,Adjusted R-squared:  0.599
F-statistic: 23.4 on 1 and 14 DF,  p-value: 0.000261

> cbind(estimates = round(coef(lm(y ~ x)), 2), true.values = c(a, b))

            estimates true.values
(Intercept)     36.94        40.0
x               -1.13        -1.5

> plot(x, y, xlab = "Covariate (continuous)", las = 1, ylab = "Continuous response",
+      cex = 1, pch = 20, col = "blue")
> abline(c(a, b), col = "red", lwd = 2)
> abline(lm(y ~ x), col = "darkblue", lwd = 2, lty = 5)
```



3.1 Interpretation of confidence intervals

Consider the frequentist inference about the slope parameter:

```
> print(summary(lm(y ~ x))$coef, dig = 3)

            Estimate Std. Error t value Pr(>|t|)
(Intercept)    36.94      2.262   16.33 1.65e-10
x              -1.13      0.234   -4.84 2.61e-04
```

A quick and dirty frequentist 95% CI:

```
> coefs <- summary(lm(y ~ x))$coef
> print(c(coefs[2, 1] - 2 * coefs[2, 2], coefs[2, 1] + 2 * coefs[2,
+     2]), dig = 3)

[1] -1.601 -0.665
```

This means that if we took 100 samples of 16 observations each from the same population and estimated a 95% CI for the slope of the linear regression, then we would expect 95 of the 100 CIs to contain the true value of the population slope.

Remember: in frequentist statistics, parameters are unknown, but **fixed**. Only the data is random and varies – and only this variation can be quantified in terms of probabilities.

We cannot make any direct probability statement about the population parameters themselves. The true value of a parameter is either in or out of any given CI, there is no probability associated with this. The parameter is just an unknown *constant*.

In particular: it is wrong to say that the population slope lies in [insert whatever interval you just got] with a probability of 95%. We gave this as an intuitive paraphrase of the 95% CI, but this is not correct. The probability statement in the 95% CI refers to the reliability of the tool, i.e., the computation of the confidence interval, and not to the parameter for which a CI is constructed.

4 One-way ANOVA

A.k.a. fixed-effects ANOVA. We'll introduce random-effects ANOVA – which is our first example of a mixed-effects model – in the next section. Generating data sets for fixed-effects vs. random-effects (a.k.a. repeated-measures) ANOVAs will clearly show how closely related these models are, and where the exact difference between them lies.

Parameters and predictor for data generation:

```
> ngroups <- 5  # Number of groups / treatments
> nsample <- 10  # Number of observations in each group
> (n <- ngroups * nsample)  # Total number of data points

[1] 50

> pop.means <- c(50, 40, 45, 55, 60)  # Population means for each of the groups
> sigma <- 3  # Residual sd (note: assumption of homoscedasticity)
> normal.error <- rnorm(n, 0, sigma)  # Residuals
> sort(round(normal.error, 2))

 [1] -6.06 -5.59 -5.54 -5.36 -5.25 -5.00 -3.97 -3.81 -3.80 -3.49 -2.77
[12] -2.68 -2.36 -1.48 -1.35 -1.34 -1.25 -1.15 -0.92 -0.91 -0.82 -0.55
[23] -0.36 -0.07  0.05  0.29  0.45  0.72  0.92  1.17  1.17  1.22  1.36
[34]  1.58  2.04  2.59  2.71  2.73  2.80  3.06  3.73  4.19  4.27  4.30
[45]  4.60  4.71  4.88  5.38  5.45  7.56
```

```
> (x <- rep(1:5, rep(nsample, ngroups)))  # Indicator for population

 [1] 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4
[36] 4 4 4 4 4 5 5 5 5 5 5 5 5 5 5

> rep(nsample, ngroups)

[1] 10 10 10 10 10

> (means <- rep(pop.means, rep(nsample, ngroups)))

 [1] 50 50 50 50 50 50 50 50 50 50 40 40 40 40 40 40 40 40 40 40 45 45 45
[24] 45 45 45 45 45 45 45 55 55 55 55 55 55 55 55 55 55 60 60 60 60 60 60
[47] 60 60 60 60

> (X <- as.matrix(model.matrix(~as.factor(x) - 1)))  # Create design matrix

   as.factor(x)1 as.factor(x)2 as.factor(x)3 as.factor(x)4 as.factor(x)5
1              1             0             0             0             0
2              1             0             0             0             0
3              1             0             0             0             0
4              1             0             0             0             0
5              1             0             0             0             0
6              1             0             0             0             0
7              1             0             0             0             0
8              1             0             0             0             0
9              1             0             0             0             0
10             1             0             0             0             0
11             0             1             0             0             0
12             0             1             0             0             0
13             0             1             0             0             0
14             0             1             0             0             0
15             0             1             0             0             0
16             0             1             0             0             0
17             0             1             0             0             0
18             0             1             0             0             0
19             0             1             0             0             0
20             0             1             0             0             0
21             0             0             1             0             0
22             0             0             1             0             0
23             0             0             1             0             0
24             0             0             1             0             0
25             0             0             1             0             0
26             0             0             1             0             0
27             0             0             1             0             0
28             0             0             1             0             0
29             0             0             1             0             0
30             0             0             1             0             0
31             0             0             0             1             0
32             0             0             0             1             0
33             0             0             0             1             0
34             0             0             0             1             0
35             0             0             0             1             0
36             0             0             0             1             0
37             0             0             0             1             0
```

```
38             0             0             0             1             0
39             0             0             0             1             0
40             0             0             0             1             0
41             0             0             0             0             1
42             0             0             0             0             1
43             0             0             0             0             1
44             0             0             0             0             1
45             0             0             0             0             1
46             0             0             0             0             1
47             0             0             0             0             1
48             0             0             0             0             1
49             0             0             0             0             1
50             0             0             0             0             1
attr(,"assign")
[1] 1 1 1 1 1
attr(,"contrasts")
attr(,"contrasts")$`as.factor(x)`
[1] "contr.treatment"
```

We assemble the responses. The deterministic part:

```
> X %*% as.matrix(pop.means)

   [,1]
1    50
2    50
3    50
4    50
5    50
6    50
7    50
8    50
9    50
10   50
11   40
12   40
13   40
14   40
15   40
16   40
17   40
18   40
19   40
20   40
21   45
22   45
23   45
24   45
25   45
26   45
27   45
28   45
29   45
30   45
```

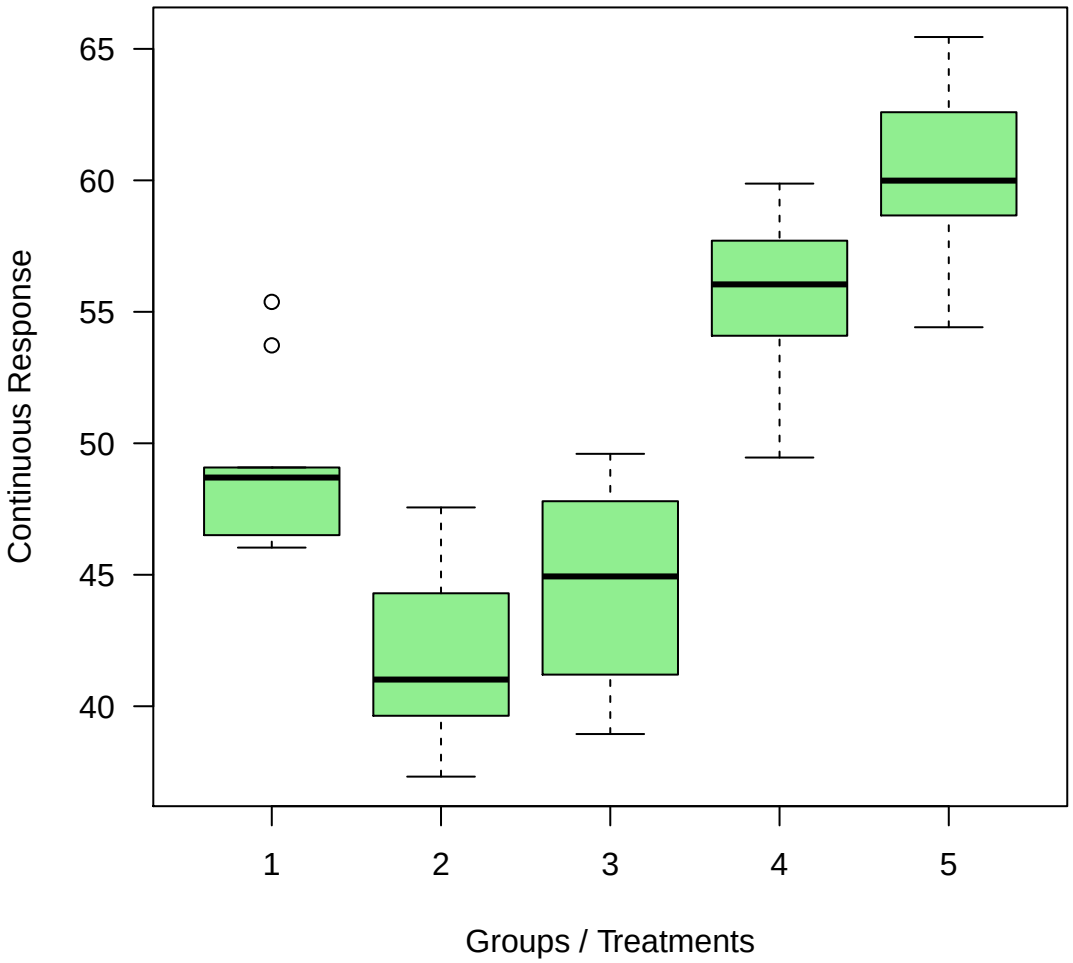
```
31   55
32   55
33   55
34   55
35   55
36   55
37   55
38   55
39   55
40   55
41   60
42   60
43   60
44   60
45   60
46   60
47   60
48   60
49   60
50   60
```

The deterministic & stochastic components together:

```
> y <- X %*% as.matrix(pop.means) + normal.error
> round(y, 2)

    [,1]
1  55.38
2  47.64
3  49.08
4  46.51
5  48.65
6  48.75
7  46.03
8  46.19
9  48.85
10 53.73
11 39.64
12 41.58
13 39.18
14 40.29
15 37.32
16 44.71
17 44.30
18 42.04
19 40.45
20 47.56
21 38.94
22 43.52
23 42.23
24 49.60
25 41.20
26 46.36
27 49.19
```

```
28 47.80
29 40.00
30 47.73
31 55.92
32 56.17
33 56.22
34 55.72
35 49.75
36 59.88
37 49.46
38 54.09
39 57.71
40 58.06
41 62.59
42 58.66
43 61.17
44 60.05
45 65.45
46 54.64
47 64.27
48 59.45
49 54.41
50 59.93
```

```
> boxplot(y ~ x, col = "lightgreen", xlab = "Groups / Treatments", ylab = "Continuous Response",
+     main = "", las = 1)
```



Analysis:

```
> print(summary(lm(y ~ as.factor(x))), dig = 3)


Call:
lm(formula = y ~ as.factor(x))

Residuals:
   Min     1Q Median     3Q    Max
 -5.84  -2.33  -0.13   2.57   6.30

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)      49.08       1.09   45.11  < 2e-16 ***
as.factor(x)2    -7.37       1.54   -4.79  1.8e-05 ***
as.factor(x)3    -4.42       1.54   -2.87  0.00615 **
as.factor(x)4     6.22       1.54    4.04  0.00021 ***
```

```
as.factor(x)5    10.98       1.54    7.14  6.4e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.44 on 45 degrees of freedom
Multiple R-squared:  0.81,Adjusted R-squared:  0.793
F-statistic:   48 on 4 and 45 DF,  p-value: 1.12e-15

> print(anova(lm(y ~ as.factor(x))), dig = 3)

Analysis of Variance Table

Response: y
             Df Sum Sq Mean Sq F value  Pr(>F)
as.factor(x)  4   2273     568      48 1.1e-15 ***
Residuals    45    533      12
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

> print(summary(lm(y ~ as.factor(x)))$coef, dig = 3)

              Estimate Std. Error t value Pr(>|t|)
(Intercept)      49.08       1.09   45.11 4.17e-39
as.factor(x)2    -7.37       1.54   -4.79 1.83e-05
as.factor(x)3    -4.42       1.54   -2.87 6.15e-03
as.factor(x)4     6.22       1.54    4.04 2.06e-04
as.factor(x)5    10.98       1.54    7.14 6.40e-09

> print(summary(lm(y ~ as.factor(x)))$sigma, dig = 3)

[1] 3.44
```

5 Random-effects ANOVA

Data generation:

```
> npop <- 10  # Number of groups / treatments: now 10 rather than 5
> nsample <- 12  # Number of observations in each group
> (n <- npop * nsample)  # Total number of data points

[1] 120
```

Random effects: the means for the different groups are correlated and they come from a probability distribution. In contrast to fixed-effects ANOVA, where they are taken to be independent and fixed / not variable.

```
> pop.grand.mean <- 50  # Grand mean
> pop.sd <- 5  # sd of population effects about mean
> pop.means <- rnorm(n = npop, mean = pop.grand.mean, sd = pop.sd)
> round(pop.means, 2)

 [1] 41.05 52.69 54.16 52.27 42.90 49.50 47.39 60.92 35.71 54.48

> sigma <- 3  # Residual sd
```

```
> normal.error <- rnorm(n, 0, sigma)  # Draw residuals
> sort(round(normal.error, 2))

  [1] -12.05  -5.56  -5.22  -5.22  -4.94  -4.61  -4.23  -4.22  -4.21  -4.12
 [11]  -4.05  -4.03  -3.86  -3.85  -3.64  -3.59  -3.50  -3.46  -3.20  -2.96
 [21]  -2.91  -2.84  -2.78  -2.67  -2.65  -2.48  -2.38  -2.35  -2.21  -2.07
 [31]  -2.05  -2.00  -1.99  -1.97  -1.88  -1.85  -1.81  -1.80  -1.49  -1.40
 [41]  -1.31  -1.29  -1.25  -1.23  -1.22  -1.21  -1.12  -1.09  -1.07  -0.99
 [51]  -0.94  -0.82  -0.58  -0.55  -0.39  -0.33  -0.26  -0.18  -0.15  -0.08
 [61]  -0.04   0.00   0.01   0.02   0.04   0.04   0.20   0.40   0.45   0.47
 [71]   0.50   0.57   0.71   0.75   0.77   0.80   0.83   1.06   1.07   1.21
 [81]   1.27   1.31   1.36   1.36   1.45   1.58   1.70   1.87   2.04   2.16
 [91]   2.19   2.28   2.31   2.44   2.50   2.52   2.61   2.83   2.85   3.14
[101]   3.27   3.27   3.44   3.46   3.57   3.62   3.67   3.81   3.85   4.36
[111]   4.54   4.91   5.10   5.27   5.38   5.63   5.84   6.60   6.69   6.91
```

Note that the stochastic part of the model consists of **two** stochastic processes now:

- the usual one involving the probability distribution for individual observations (`normal.error` above)
- the one involving the probability distribution for the group / population means (the random effects)

We generate the predictor:

```
> (x <- rep(1:npop, rep(nsample, npop)))

  [1]  1  1  1  1  1  1  1  1  1  1  1  1  2  2  2  2  2  2  2  2  2  2  2
 [24]  2  3  3  3  3  3  3  3  3  3  3  3  3  3  4  4  4  4  4  4  4  4  4  4
 [47]  4  4  5  5  5  5  5  5  5  5  5  5  5  5  5  6  6  6  6  6  6  6  6  6
 [70]  6  6  6  7  7  7  7  7  7  7  7  7  7  7  7  7  8  8  8  8  8  8  8  8
 [93]  8  8  8  8  9  9  9  9  9  9  9  9  9  9  9  9  9 10 10 10 10 10 10 10
[116] 10 10 10 10 10

> rep(nsample, npop)

 [1] 12 12 12 12 12 12 12 12 12 12

> (X <- as.matrix(model.matrix(~as.factor(x) - 1)))

    as.factor(x)1 as.factor(x)2 as.factor(x)3 as.factor(x)4 as.factor(x)5
1               1             0             0             0             0
2               1             0             0             0             0
3               1             0             0             0             0
4               1             0             0             0             0
5               1             0             0             0             0
6               1             0             0             0             0
7               1             0             0             0             0
8               1             0             0             0             0
9               1             0             0             0             0
10              1             0             0             0             0
11              1             0             0             0             0
12              1             0             0             0             0
13              0             1             0             0             0
14              0             1             0             0             0
15              0             1             0             0             0
16              0             1             0             0             0
```

17	0	1	0	0	0
18	0	1	0	0	0
19	0	1	0	0	0
20	0	1	0	0	0
21	0	1	0	0	0
22	0	1	0	0	0
23	0	1	0	0	0
24	0	1	0	0	0
25	0	0	1	0	0
26	0	0	1	0	0
27	0	0	1	0	0
28	0	0	1	0	0
29	0	0	1	0	0
30	0	0	1	0	0
31	0	0	1	0	0
32	0	0	1	0	0
33	0	0	1	0	0
34	0	0	1	0	0
35	0	0	1	0	0
36	0	0	1	0	0
37	0	0	0	1	0
38	0	0	0	1	0
39	0	0	0	1	0
40	0	0	0	1	0
41	0	0	0	1	0
42	0	0	0	1	0
43	0	0	0	1	0
44	0	0	0	1	0
45	0	0	0	1	0
46	0	0	0	1	0
47	0	0	0	1	0
48	0	0	0	1	0
49	0	0	0	0	1
50	0	0	0	0	1
51	0	0	0	0	1
52	0	0	0	0	1
53	0	0	0	0	1
54	0	0	0	0	1
55	0	0	0	0	1
56	0	0	0	0	1
57	0	0	0	0	1
58	0	0	0	0	1
59	0	0	0	0	1
60	0	0	0	0	1
61	0	0	0	0	0
62	0	0	0	0	0
63	0	0	0	0	0
64	0	0	0	0	0
65	0	0	0	0	0
66	0	0	0	0	0
67	0	0	0	0	0
68	0	0	0	0	0
69	0	0	0	0	0

```
70                0                0                0                0                0
71                0                0                0                0                0
72                0                0                0                0                0
73                0                0                0                0                0
74                0                0                0                0                0
75                0                0                0                0                0
76                0                0                0                0                0
77                0                0                0                0                0
78                0                0                0                0                0
79                0                0                0                0                0
80                0                0                0                0                0
81                0                0                0                0                0
82                0                0                0                0                0
83                0                0                0                0                0
84                0                0                0                0                0
85                0                0                0                0                0
86                0                0                0                0                0
87                0                0                0                0                0
88                0                0                0                0                0
89                0                0                0                0                0
90                0                0                0                0                0
91                0                0                0                0                0
92                0                0                0                0                0
93                0                0                0                0                0
94                0                0                0                0                0
95                0                0                0                0                0
96                0                0                0                0                0
97                0                0                0                0                0
98                0                0                0                0                0
99                0                0                0                0                0
100               0                0                0                0                0
101               0                0                0                0                0
102               0                0                0                0                0
103               0                0                0                0                0
104               0                0                0                0                0
105               0                0                0                0                0
106               0                0                0                0                0
107               0                0                0                0                0
108               0                0                0                0                0
109               0                0                0                0                0
110               0                0                0                0                0
111               0                0                0                0                0
112               0                0                0                0                0
113               0                0                0                0                0
114               0                0                0                0                0
115               0                0                0                0                0
116               0                0                0                0                0
117               0                0                0                0                0
118               0                0                0                0                0
119               0                0                0                0                0
120               0                0                0                0                0
    as.factor(x)6 as.factor(x)7 as.factor(x)8 as.factor(x)9 as.factor(x)10
1               0             0             0             0              0
```

2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0
5	0	0	0	0	0
6	0	0	0	0	0
7	0	0	0	0	0
8	0	0	0	0	0
9	0	0	0	0	0
10	0	0	0	0	0
11	0	0	0	0	0
12	0	0	0	0	0
13	0	0	0	0	0
14	0	0	0	0	0
15	0	0	0	0	0
16	0	0	0	0	0
17	0	0	0	0	0
18	0	0	0	0	0
19	0	0	0	0	0
20	0	0	0	0	0
21	0	0	0	0	0
22	0	0	0	0	0
23	0	0	0	0	0
24	0	0	0	0	0
25	0	0	0	0	0
26	0	0	0	0	0
27	0	0	0	0	0
28	0	0	0	0	0
29	0	0	0	0	0
30	0	0	0	0	0
31	0	0	0	0	0
32	0	0	0	0	0
33	0	0	0	0	0
34	0	0	0	0	0
35	0	0	0	0	0
36	0	0	0	0	0
37	0	0	0	0	0
38	0	0	0	0	0
39	0	0	0	0	0
40	0	0	0	0	0
41	0	0	0	0	0
42	0	0	0	0	0
43	0	0	0	0	0
44	0	0	0	0	0
45	0	0	0	0	0
46	0	0	0	0	0
47	0	0	0	0	0
48	0	0	0	0	0
49	0	0	0	0	0
50	0	0	0	0	0
51	0	0	0	0	0
52	0	0	0	0	0
53	0	0	0	0	0
54	0	0	0	0	0

55	0	0	0	0	0
56	0	0	0	0	0
57	0	0	0	0	0
58	0	0	0	0	0
59	0	0	0	0	0
60	0	0	0	0	0
61	1	0	0	0	0
62	1	0	0	0	0
63	1	0	0	0	0
64	1	0	0	0	0
65	1	0	0	0	0
66	1	0	0	0	0
67	1	0	0	0	0
68	1	0	0	0	0
69	1	0	0	0	0
70	1	0	0	0	0
71	1	0	0	0	0
72	1	0	0	0	0
73	0	1	0	0	0
74	0	1	0	0	0
75	0	1	0	0	0
76	0	1	0	0	0
77	0	1	0	0	0
78	0	1	0	0	0
79	0	1	0	0	0
80	0	1	0	0	0
81	0	1	0	0	0
82	0	1	0	0	0
83	0	1	0	0	0
84	0	1	0	0	0
85	0	0	1	0	0
86	0	0	1	0	0
87	0	0	1	0	0
88	0	0	1	0	0
89	0	0	1	0	0
90	0	0	1	0	0
91	0	0	1	0	0
92	0	0	1	0	0
93	0	0	1	0	0
94	0	0	1	0	0
95	0	0	1	0	0
96	0	0	1	0	0
97	0	0	0	1	0
98	0	0	0	1	0
99	0	0	0	1	0
100	0	0	0	1	0
101	0	0	0	1	0
102	0	0	0	1	0
103	0	0	0	1	0
104	0	0	0	1	0
105	0	0	0	1	0
106	0	0	0	1	0
107	0	0	0	1	0

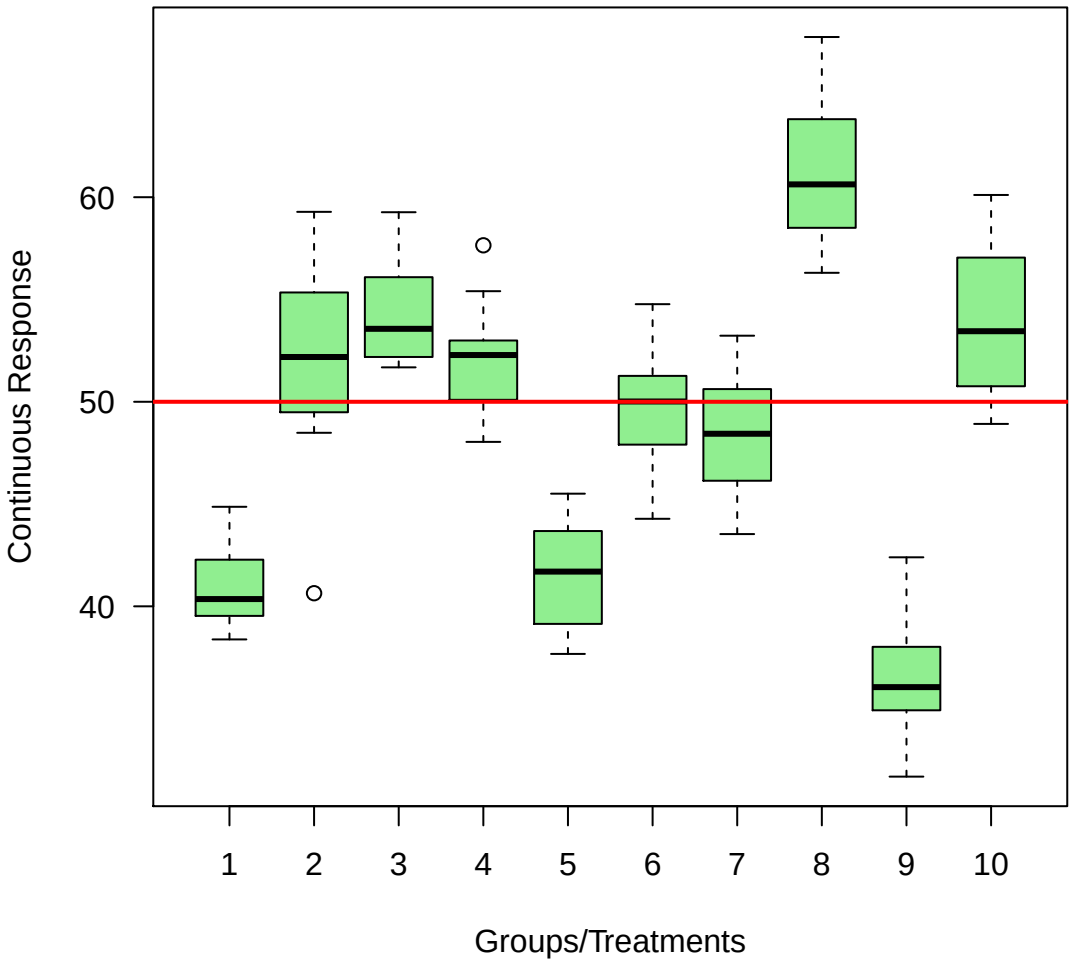
```
108             0             0             0             1             0
109             0             0             0             0             1
110             0             0             0             0             1
111             0             0             0             0             1
112             0             0             0             0             1
113             0             0             0             0             1
114             0             0             0             0             1
115             0             0             0             0             1
116             0             0             0             0             1
117             0             0             0             0             1
118             0             0             0             0             1
119             0             0             0             0             1
120             0             0             0             0             1
attr(,"assign")
 [1] 1 1 1 1 1 1 1 1 1 1
attr(,"contrasts")
attr(,"contrasts")$`as.factor(x)`
[1] "contr.treatment"
```

We generate the response. First, we assemble the deterministic component and the random-effects stochastic component:

```
> pop.means

 [1] 41.05 52.69 54.16 52.27 42.90 49.50 47.39 60.92 35.71 54.48

> as.numeric(X %*% as.matrix(pop.means))

  [1] 41.05 41.05 41.05 41.05 41.05 41.05 41.05 41.05 41.05 41.05 41.05
 [12] 41.05 52.69 52.69 52.69 52.69 52.69 52.69 52.69 52.69 52.69 52.69
 [23] 52.69 52.69 54.16 54.16 54.16 54.16 54.16 54.16 54.16 54.16 54.16
 [34] 54.16 54.16 54.16 52.27 52.27 52.27 52.27 52.27 52.27 52.27 52.27
 [45] 52.27 52.27 52.27 52.27 42.90 42.90 42.90 42.90 42.90 42.90 42.90
 [56] 42.90 42.90 42.90 42.90 42.90 49.50 49.50 49.50 49.50 49.50 49.50
 [67] 49.50 49.50 49.50 49.50 49.50 49.50 47.39 47.39 47.39 47.39 47.39
 [78] 47.39 47.39 47.39 47.39 47.39 47.39 47.39 60.92 60.92 60.92 60.92
 [89] 60.92 60.92 60.92 60.92 60.92 60.92 60.92 60.92 35.71 35.71 35.71
[100] 35.71 35.71 35.71 35.71 35.71 35.71 35.71 35.71 35.71 54.48 54.48
[111] 54.48 54.48 54.48 54.48 54.48 54.48 54.48 54.48 54.48 54.48
```

Then, we add the individual-level stochastic component:

```
> y <- as.numeric(X %*% as.matrix(pop.means) + normal.error)  # recall that as.numeric is essential
> boxplot(y ~ x, col = "lightgreen", xlab = "Groups/Treatments", ylab = "Continuous Response",
+     main = "", las = 1)
> abline(h = pop.grand.mean, col = "red", lwd = 2)
```



Restricted maximum likelihood (REML) analysis; this is the default in R because it yields unbiased estimates for small samples. For model comparison with `anova`, make sure you switch to the regular maximum likelihood (ML) analysis:

```
> library("lme4")
> pop <- as.factor(x)  # Define x as a factor and call it pop
> lme.fit <- lmer(y ~ 1 + 1 | pop)
```

Inspect results:

```
> print(lme.fit, cor = F, dig = 3)

Linear mixed model fit by REML ['lmerMod']
Formula: y ~ 1 + 1 | pop
REML criterion at convergence: 654.7
Random effects:
 Groups   Name        Std.Dev.
 pop      (Intercept) 7.40
```

```
 Residual             3.17
Number of obs: 120, groups: pop, 10
Fixed Effects:
(Intercept)
       49.1

> print(summary(lme.fit), cor = F, dig = 3)

Linear mixed model fit by REML ['lmerMod']
Formula: y ~ 1 + 1 | pop

REML criterion at convergence: 654.7

Scaled residuals:
   Min     1Q Median     3Q    Max
-3.567 -0.678 -0.055  0.706  2.318

Random effects:
 Groups   Name        Variance Std.Dev.
 pop      (Intercept) 54.8     7.40
 Residual             10.0     3.17
Number of obs: 120, groups: pop, 10

Fixed effects:
            Estimate Std. Error t value
(Intercept)    49.06       2.36    20.8
```

Compare with the true values:

```
> pop.sd

[1] 5

> sigma

[1] 3

> pop.grand.mean

[1] 50
```

Estimated random effects:

```
> ranef(lme.fit)

$pop
   (Intercept)
1      -7.9398
2       2.8802
3       5.1989
4       2.8709
5      -7.5007
6       0.7678
7      -0.7110
8      11.9132
```

```
9     -12.4606
10      4.9812
```

Compare with the true values:

```
> round(data.frame(true.values = pop.means - pop.grand.mean, RMLEs = ranef(lme.fit)$pop[,
+     1]), 2)

   true.values  RMLEs
1        -8.95  -7.94
2         2.69   2.88
3         4.16   5.20
4         2.27   2.87
5        -7.10  -7.50
6        -0.50   0.77
7        -2.61  -0.71
8        10.92  11.91
9       -14.29 -12.46
10        4.48   4.98
```

ML analysis (as opposed to REML):

```
> lme.fit2 <- lmer(y ~ 1 + 1 | pop, REML = FALSE)
> print(lme.fit2, cor = F, dig = 3)

Linear mixed model fit by maximum likelihood  ['lmerMod']
Formula: y ~ 1 + 1 | pop
     AIC      BIC   logLik deviance df.resid
   664.2    672.6   -329.1    658.2      117
Random effects:
 Groups   Name        Std.Dev.
 pop      (Intercept) 7.02
 Residual             3.17
Number of obs: 120, groups: pop, 10
Fixed Effects:
(Intercept)
       49.1

> print(summary(lme.fit2), cor = F, dig = 3)

Linear mixed model fit by maximum likelihood  ['lmerMod']
Formula: y ~ 1 + 1 | pop

     AIC      BIC   logLik deviance df.resid
   664.2    672.6   -329.1    658.2      117

Scaled residuals:
   Min     1Q Median     3Q    Max
-3.565 -0.681 -0.058  0.707  2.319

Random effects:
 Groups   Name        Variance Std.Dev.
 pop      (Intercept) 49.3     7.02
 Residual             10.0     3.17
Number of obs: 120, groups: pop, 10
```

```
Fixed effects:
            Estimate Std. Error t value
(Intercept)    49.06       2.24    21.9

> round(data.frame(true.values = pop.means - pop.grand.mean, RMLEs = ranef(lme.fit)$pop[,
+     1], MLEs = ranef(lme.fit2)$pop[, 1]), 3)

   true.values   RMLEs    MLEs
1       -8.947  -7.940  -7.926
2        2.689   2.880   2.875
3        4.162   5.199   5.190
4        2.267   2.871   2.866
5       -7.102  -7.501  -7.488
6       -0.496   0.768   0.767
7       -2.614  -0.711  -0.710
8       10.916  11.913  11.893
9      -14.294 -12.461 -12.439
10       4.481   4.981   4.973
```

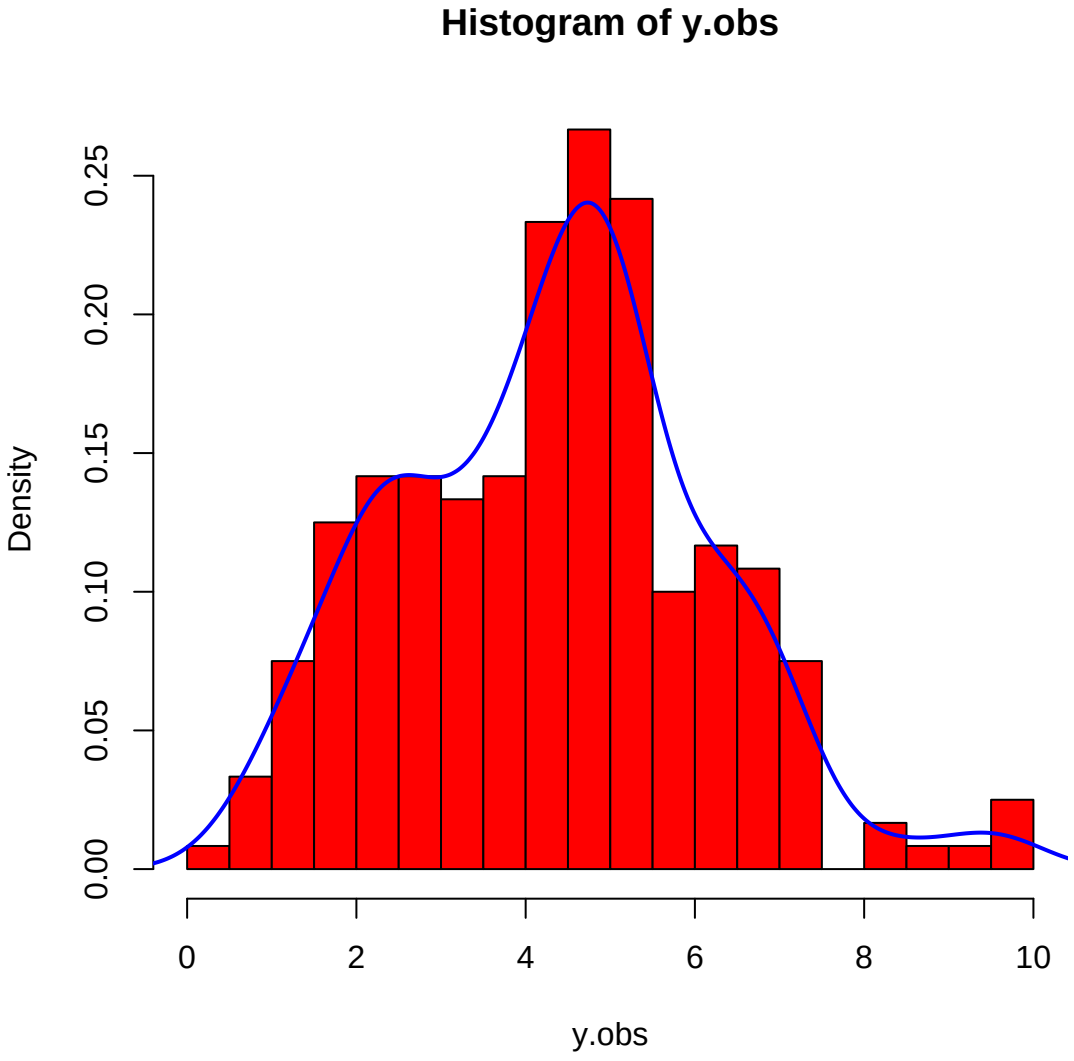
Other kinds of hierarchical examples besides population-subpopulations-individuals:

- the mean GPA of undergraduate students in the Humanities division (the population), the mean GPA of undergraduate students in the departments in the Humanities division (subpopulations), the GPA of individual students in each of these departments
- the mean time for deciding whether *wug* is an Eng. word in the entire population of native Eng. speakers (the population), the mean reaction time for deciding whether *wug* is an Eng. word for particular speakers X, Y, Z etc. (subpopulations), the individual measurements / times taken to decide whether *wug* is an Eng. word by a particular speaker (assume the experiment was long enough and the same speaker was presented with *wug* multiple times)
- this last example of measurements nested within subjects, which are nested within an experimental manipulation (i.e., a fixed effect) is what motivated the particular syntax of `lmer()` formulas

Another random-effects ANOVA example:

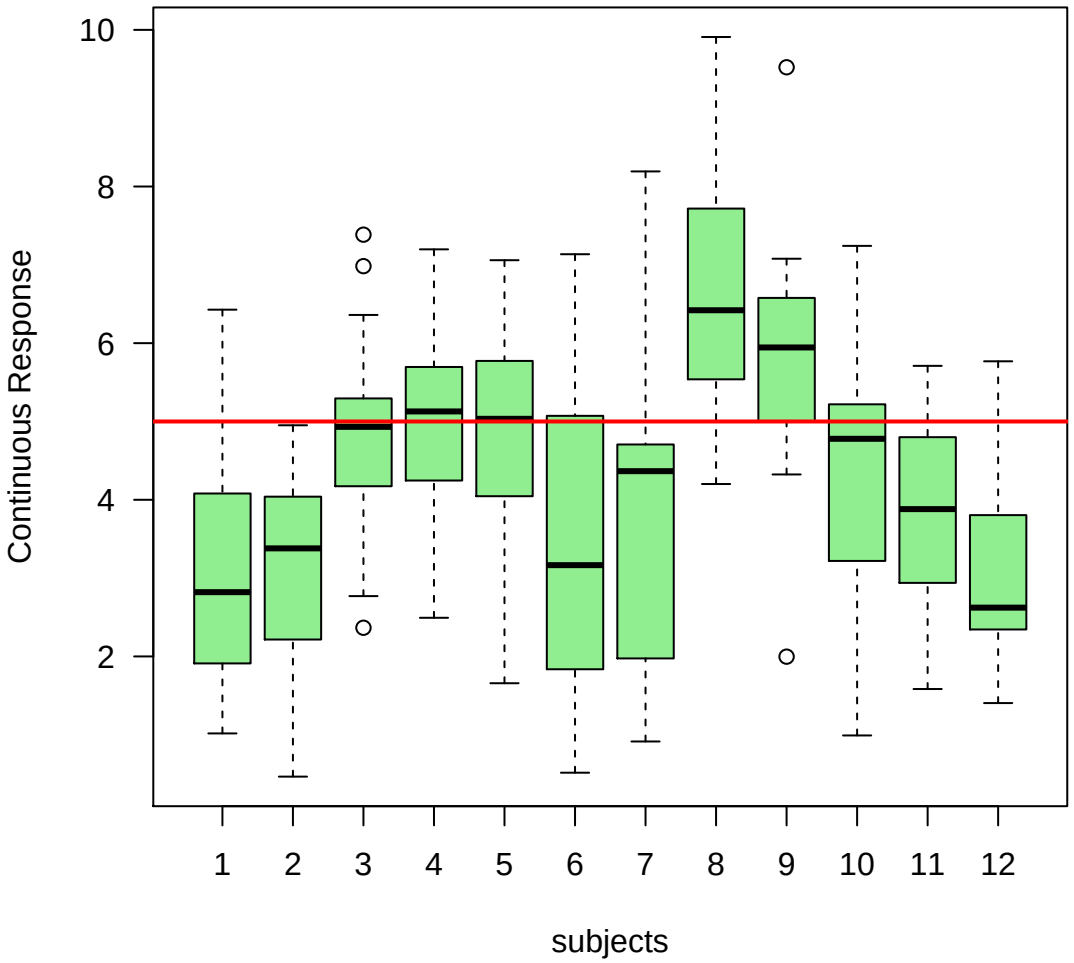
```
> mu.pop <- 5
> sigma.subj <- 1.3
> sigma.obs <- 1.5
> subj.means <- rnorm(12, mu.pop, sigma.subj)
> subj.means

 [1] 3.193 3.329 4.030 4.741 4.534 3.757 4.042 5.652 5.955 4.059 3.685
[12] 3.671

> y.obs <- vector(length = 12 * 20)
> y.obs[1:20] <- rnorm(20, subj.means[1], sigma.obs)
> y.obs[21:40] <- rnorm(20, subj.means[2], sigma.obs)
> y.obs[21:40] <- rnorm(20, subj.means[2], sigma.obs)
> y.obs[41:60] <- rnorm(20, subj.means[3], sigma.obs)
> y.obs[61:80] <- rnorm(20, subj.means[4], sigma.obs)
> y.obs[81:100] <- rnorm(20, subj.means[5], sigma.obs)
> y.obs[101:120] <- rnorm(20, subj.means[6], sigma.obs)
> y.obs[121:140] <- rnorm(20, subj.means[7], sigma.obs)
> y.obs[141:160] <- rnorm(20, subj.means[8], sigma.obs)
```

```
> y.obs[161:180] <- rnorm(20, subj.means[9], sigma.obs)
> y.obs[181:200] <- rnorm(20, subj.means[10], sigma.obs)
> y.obs[201:220] <- rnorm(20, subj.means[11], sigma.obs)
> y.obs[221:240] <- rnorm(20, subj.means[12], sigma.obs)
> summary(y.obs)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  0.466   2.960   4.400   4.330   5.280   9.910

> hist(y.obs, breaks = 25, col = "red", freq = FALSE)
> lines(density(y.obs), col = "blue", lwd = 2)
```




```
> x <- rep(1:12, each = 20)
> x

  [1]  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  2  2  2
 [24]  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  3  3  3  3  3  3
```

```
 [47]  3  3  3  3  3  3  3  3  3  3  3  3  3  3  4  4  4  4  4  4  4  4  4
 [70]  4  4  4  4  4  4  4  4  4  4  4  4  5  5  5  5  5  5  5  5  5  5  5
 [93]  5  5  5  5  5  5  5  5  6  6  6  6  6  6  6  6  6  6  6  6  6  6  6
[116]  6  6  6  6  6  7  7  7  7  7  7  7  7  7  7  7  7  7  7  7  7  7  7
[139]  7  7  8  8  8  8  8  8  8  8  8  8  8  8  8  8  8  8  8  8  8  8  9
[162]  9  9  9  9  9  9  9  9  9  9  9  9  9  9  9  9  9  9 10 10 10 10
[185] 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 11 11 11 11 11 11 11
[208] 11 11 11 11 11 11 11 11 11 11 11 11 11 12 12 12 12 12 12 12 12 12 12
[231] 12 12 12 12 12 12 12 12 12 12

> str(x)

 int [1:240] 1 1 1 1 1 1 1 1 1 1 ...

> x <- as.factor(x)
> str(x)

 Factor w/ 12 levels "1","2","3","4",..: 1 1 1 1 1 1 1 1 1 1 ...

> summary(x)

 1  2  3  4  5  6  7  8  9 10 11 12
20 20 20 20 20 20 20 20 20 20 20 20

> boxplot(y.obs ~ x, col = "lightgreen", xlab = "subjects", ylab = "Continuous Response",
+     main = "", las = 1)
> abline(h = mu.pop, col = "red", lwd = 2)
```




```
> library("lme4")
> m1 <- lmer(y.obs ~ 1 + 1 | x)
> print(summary(m1), cor = F, dig = 3)

Linear mixed model fit by REML ['lmerMod']
Formula: y.obs ~ 1 + 1 | x

REML criterion at convergence: 897.7

Scaled residuals:
    Min      1Q  Median      3Q     Max
-2.4876 -0.6176  0.0932  0.6125  2.9520

Random effects:
 Groups   Name        Variance Std.Dev.
 x        (Intercept) 1.21     1.10
 Residual             2.18     1.48
```

```
Number of obs: 240, groups: x, 12

Fixed effects:
            Estimate Std. Error t value
(Intercept)    4.326      0.331    13.1

> data.frame(intercept = mu.pop, sigma.x = sigma.subj, sigma.resid = sigma.obs)

  intercept sigma.x sigma.resid
1         5     1.3         1.5
```

6 Two-way ANOVA

We first set the parameters and the predictors.

Sample size:

```
> n.groups.1 <- 5
> n.groups.2 <- 3
> nsample <- 12
> (n <- n.groups.1 * nsample)

[1] 60
```

We generate the two factors (look at the documentation / help file for the `gl` function):

```
> (groups.1 <- gl(n = n.groups.1, k = nsample, length = n, labels = paste("level.",
+     c(1:n.groups.1), sep = "")))

 [1] level.1 level.1 level.1 level.1 level.1 level.1 level.1 level.1
 [9] level.1 level.1 level.1 level.1 level.2 level.2 level.2 level.2
[17] level.2 level.2 level.2 level.2 level.2 level.2 level.2 level.2
[25] level.3 level.3 level.3 level.3 level.3 level.3 level.3 level.3
[33] level.3 level.3 level.3 level.3 level.4 level.4 level.4 level.4
[41] level.4 level.4 level.4 level.4 level.4 level.4 level.4 level.4
[49] level.5 level.5 level.5 level.5 level.5 level.5 level.5 level.5
[57] level.5 level.5 level.5 level.5
Levels: level.1 level.2 level.3 level.4 level.5
```

`groups.2` is replicated for each level of `groups.1`:

```
> (groups.2 <- gl(n = n.groups.2, k = nsample/n.groups.2, length = n,
+     labels = paste("level.", c(1:n.groups.2), sep = "")))

 [1] level.1 level.1 level.1 level.1 level.2 level.2 level.2 level.2
 [9] level.3 level.3 level.3 level.3 level.1 level.1 level.1 level.1
[17] level.2 level.2 level.2 level.2 level.3 level.3 level.3 level.3
[25] level.1 level.1 level.1 level.1 level.2 level.2 level.2 level.2
[33] level.3 level.3 level.3 level.3 level.1 level.1 level.1 level.1
[41] level.2 level.2 level.2 level.2 level.3 level.3 level.3 level.3
[49] level.1 level.1 level.1 level.1 level.2 level.2 level.2 level.2
[57] level.3 level.3 level.3 level.3
Levels: level.1 level.2 level.3

> summary(groups.1)
```

```
level.1 level.2 level.3 level.4 level.5
     12      12      12      12      12

> summary(groups.2)

level.1 level.2 level.3
     20      20      20

> contrasts(groups.1)

        level.2 level.3 level.4 level.5
level.1       0       0       0       0
level.2       1       0       0       0
level.3       0       1       0       0
level.4       0       0       1       0
level.5       0       0       0       1

> contrasts(groups.2)

        level.2 level.3
level.1       0       0
level.2       1       0
level.3       0       1

> data.frame(groups.1, groups.2)

   groups.1 groups.2
1   level.1  level.1
2   level.1  level.1
3   level.1  level.1
4   level.1  level.1
5   level.1  level.2
6   level.1  level.2
7   level.1  level.2
8   level.1  level.2
9   level.1  level.3
10  level.1  level.3
11  level.1  level.3
12  level.1  level.3
13  level.2  level.1
14  level.2  level.1
15  level.2  level.1
16  level.2  level.1
17  level.2  level.2
18  level.2  level.2
19  level.2  level.2
20  level.2  level.2
21  level.2  level.3
22  level.2  level.3
23  level.2  level.3
24  level.2  level.3
25  level.3  level.1
26  level.3  level.1
27  level.3  level.1
28  level.3  level.1
```

```
29  level.3  level.2
30  level.3  level.2
31  level.3  level.2
32  level.3  level.2
33  level.3  level.3
34  level.3  level.3
35  level.3  level.3
36  level.3  level.3
37  level.4  level.1
38  level.4  level.1
39  level.4  level.1
40  level.4  level.1
41  level.4  level.2
42  level.4  level.2
43  level.4  level.2
44  level.4  level.2
45  level.4  level.3
46  level.4  level.3
47  level.4  level.3
48  level.4  level.3
49  level.5  level.1
50  level.5  level.1
51  level.5  level.1
52  level.5  level.1
53  level.5  level.2
54  level.5  level.2
55  level.5  level.2
56  level.5  level.2
57  level.5  level.3
58  level.5  level.3
59  level.5  level.3
60  level.5  level.3
```

We then choose the effects. The intercept:

```
> baseline <- 40
```

Then the main effects:

```
> (groups.1.effects <- c(-10, -5, 5, 10))  # groups.1 effects

[1] -10  -5   5  10

> (groups.2.effects <- c(5, 10))  # groups.2 effects

[1]  5 10
```

Finally, the interaction effects – 8 of them, i.e., $(5-1)\cdot(3-1)$ (why do we count them like this?):

```
> (interaction.effects <- c(-2, 3, 0, 4, 4, 0, 3, -2))

[1] -2  3  0  4  4  0  3 -2

> (all.effects <- c(baseline, groups.1.effects, groups.2.effects, interaction.effects))

 [1]  40 -10  -5   5  10   5  10  -2   3   0   4   4   0   3  -2
```

```
> length(all.effects)

[1] 15
```

We generate the residuals:

```
> sigma <- 3
> normal.error <- rnorm(n, 0, sigma)
> round(normal.error, 2)

 [1]  1.31  1.67  1.85  0.14 -1.94  2.06  4.44  3.01  1.17  5.86 -2.21
[12] -2.27  1.98 -2.36 -1.73  0.25 -1.93  4.69 -6.44 -2.65  1.30 -1.08
[23]  3.27 -3.62  3.58 -0.50 -0.50 -3.28  3.40  1.45 -1.07 -2.06 -1.57
[34]  2.02 -5.39 -3.70 -0.08  2.63 -0.54  0.62  5.29 -7.46  4.90  1.96
[45]  1.07  1.50 -2.31  1.06 -5.05  0.11  3.57 -5.52 -3.59 -1.24  5.39
[56]  0.91 -2.17 -5.69 -1.66  1.61
```

We create design matrix:

```
> X <- as.matrix(model.matrix(~groups.1 * groups.2))
> dim(X)

[1] 60 15

> head(X)

  (Intercept) groups.1level.2 groups.1level.3 groups.1level.4
1           1               0               0               0
2           1               0               0               0
3           1               0               0               0
4           1               0               0               0
5           1               0               0               0
6           1               0               0               0
  groups.1level.5 groups.2level.2 groups.2level.3
1               0               0               0
2               0               0               0
3               0               0               0
4               0               0               0
5               0               1               0
6               0               1               0
  groups.1level.2:groups.2level.2 groups.1level.3:groups.2level.2
1                               0                               0
2                               0                               0
3                               0                               0
4                               0                               0
5                               0                               0
6                               0                               0
  groups.1level.4:groups.2level.2 groups.1level.5:groups.2level.2
1                               0                               0
2                               0                               0
3                               0                               0
4                               0                               0
5                               0                               0
6                               0                               0
  groups.1level.2:groups.2level.3 groups.1level.3:groups.2level.3
```

```
1                                 0                                 0
2                                 0                                 0
3                                 0                                 0
4                                 0                                 0
5                                 0                                 0
6                                 0                                 0
  groups.1level.4:groups.2level.3 groups.1level.5:groups.2level.3
1                                 0                                 0
2                                 0                                 0
3                                 0                                 0
4                                 0                                 0
5                                 0                                 0
6                                 0                                 0
```

We generate the response by putting together the deterministic and stochastic parts of the model. First, look at the deterministic part:

```
> data.frame(groups.1, groups.2, E.y = as.numeric(as.matrix(X) %*% as.matrix(all.effects)),
+     error = round(normal.error, 2))

   groups.1 groups.2 E.y error
1   level.1  level.1  40  1.31
2   level.1  level.1  40  1.67
3   level.1  level.1  40  1.85
4   level.1  level.1  40  0.14
5   level.1  level.2  45 -1.94
6   level.1  level.2  45  2.06
7   level.1  level.2  45  4.44
8   level.1  level.2  45  3.01
9   level.1  level.3  50  1.17
10  level.1  level.3  50  5.86
11  level.1  level.3  50 -2.21
12  level.1  level.3  50 -2.27
13  level.2  level.1  30  1.98
14  level.2  level.1  30 -2.36
15  level.2  level.1  30 -1.73
16  level.2  level.1  30  0.25
17  level.2  level.2  33 -1.93
18  level.2  level.2  33  4.69
19  level.2  level.2  33 -6.44
20  level.2  level.2  33 -2.65
21  level.2  level.3  44  1.30
22  level.2  level.3  44 -1.08
23  level.2  level.3  44  3.27
24  level.2  level.3  44 -3.62
25  level.3  level.1  35  3.58
26  level.3  level.1  35 -0.50
27  level.3  level.1  35 -0.50
28  level.3  level.1  35 -3.28
29  level.3  level.2  43  3.40
30  level.3  level.2  43  1.45
31  level.3  level.2  43 -1.07
32  level.3  level.2  43 -2.06
33  level.3  level.3  45 -1.57
```

```
34  level.3  level.3  45  2.02
35  level.3  level.3  45 -5.39
36  level.3  level.3  45 -3.70
37  level.4  level.1  45 -0.08
38  level.4  level.1  45  2.63
39  level.4  level.1  45 -0.54
40  level.4  level.1  45  0.62
41  level.4  level.2  50  5.29
42  level.4  level.2  50 -7.46
43  level.4  level.2  50  4.90
44  level.4  level.2  50  1.96
45  level.4  level.3  58  1.07
46  level.4  level.3  58  1.50
47  level.4  level.3  58 -2.31
48  level.4  level.3  58  1.06
49  level.5  level.1  50 -5.05
50  level.5  level.1  50  0.11
51  level.5  level.1  50  3.57
52  level.5  level.1  50 -5.52
53  level.5  level.2  59 -3.59
54  level.5  level.2  59 -1.24
55  level.5  level.2  59  5.39
56  level.5  level.2  59  0.91
57  level.5  level.3  58 -2.17
58  level.5  level.3  58 -5.69
59  level.5  level.3  58 -1.66
60  level.5  level.3  58  1.61
```

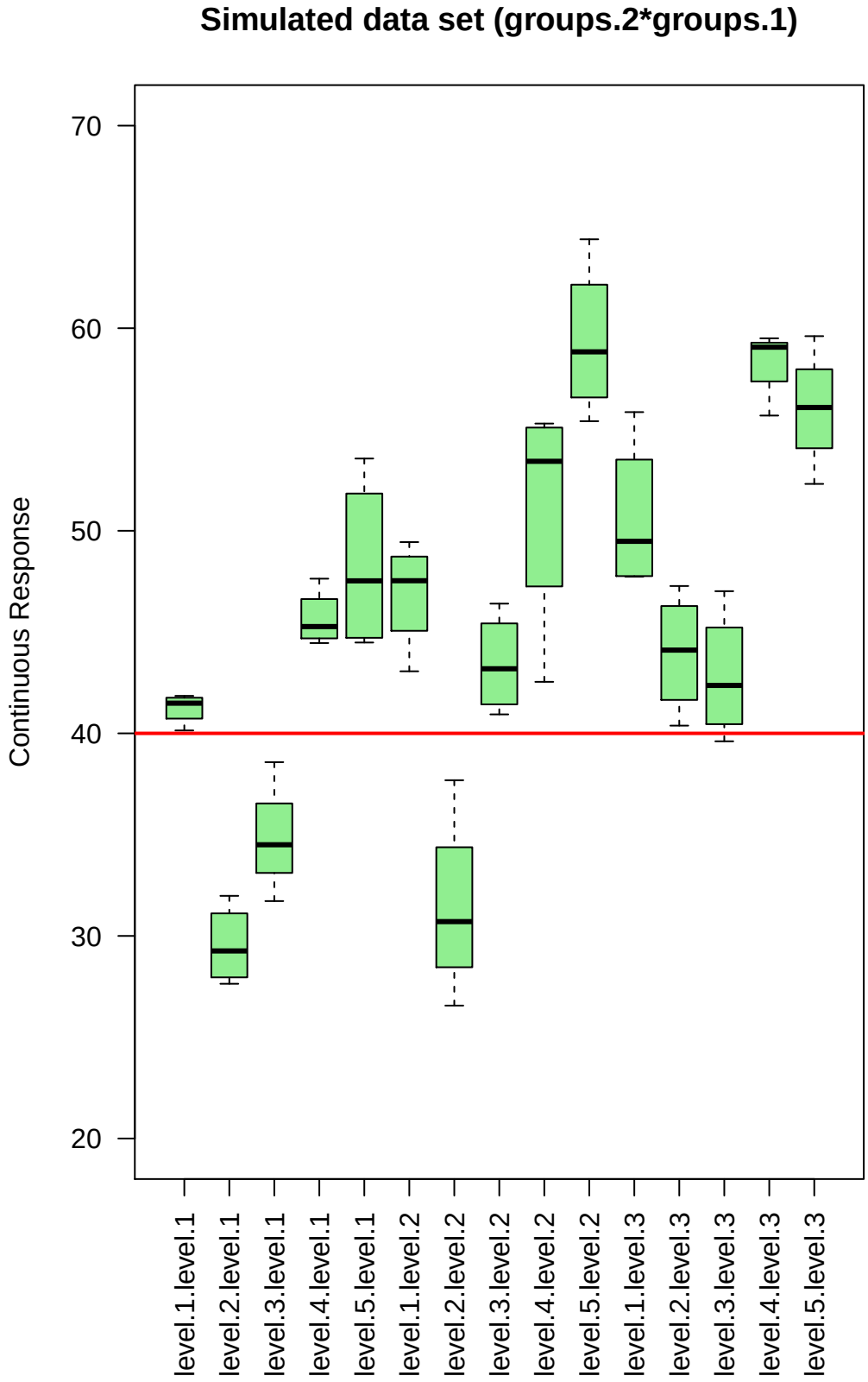
Now everything together:

```
> y <- as.numeric(as.matrix(X) %*% as.matrix(all.effects) + normal.error)  # recall that as.numeric is
> data.frame(groups.1, groups.2, E.y = as.vector(as.matrix(X) %*% as.matrix(all.effects)),
+     error = round(normal.error, 2), y = round(y, 2))

   groups.1 groups.2 E.y error     y
1   level.1  level.1  40  1.31 41.31
2   level.1  level.1  40  1.67 41.67
3   level.1  level.1  40  1.85 41.85
4   level.1  level.1  40  0.14 40.14
5   level.1  level.2  45 -1.94 43.06
6   level.1  level.2  45  2.06 47.06
7   level.1  level.2  45  4.44 49.44
8   level.1  level.2  45  3.01 48.01
9   level.1  level.3  50  1.17 51.17
10  level.1  level.3  50  5.86 55.86
11  level.1  level.3  50 -2.21 47.79
12  level.1  level.3  50 -2.27 47.73
13  level.2  level.1  30  1.98 31.98
14  level.2  level.1  30 -2.36 27.64
15  level.2  level.1  30 -1.73 28.27
16  level.2  level.1  30  0.25 30.25
17  level.2  level.2  33 -1.93 31.07
18  level.2  level.2  33  4.69 37.69
19  level.2  level.2  33 -6.44 26.56
```

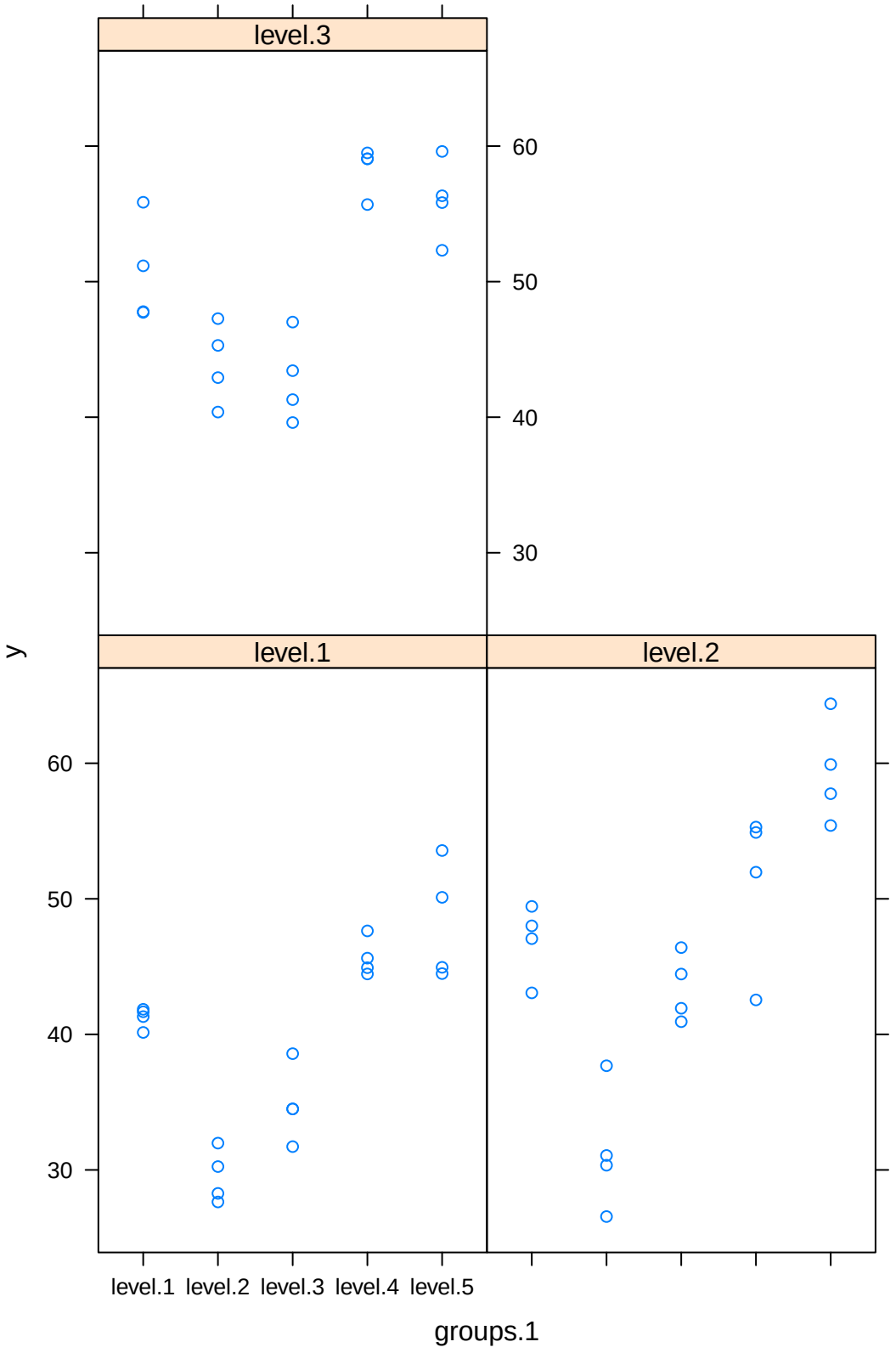
```
20  level.2  level.2  33 -2.65 30.35
21  level.2  level.3  44  1.30 45.30
22  level.2  level.3  44 -1.08 42.92
23  level.2  level.3  44  3.27 47.27
24  level.2  level.3  44 -3.62 40.38
25  level.3  level.1  35  3.58 38.58
26  level.3  level.1  35 -0.50 34.50
27  level.3  level.1  35 -0.50 34.50
28  level.3  level.1  35 -3.28 31.72
29  level.3  level.2  43  3.40 46.40
30  level.3  level.2  43  1.45 44.45
31  level.3  level.2  43 -1.07 41.93
32  level.3  level.2  43 -2.06 40.94
33  level.3  level.3  45 -1.57 43.43
34  level.3  level.3  45  2.02 47.02
35  level.3  level.3  45 -5.39 39.61
36  level.3  level.3  45 -3.70 41.30
37  level.4  level.1  45 -0.08 44.92
38  level.4  level.1  45  2.63 47.63
39  level.4  level.1  45 -0.54 44.46
40  level.4  level.1  45  0.62 45.62
41  level.4  level.2  50  5.29 55.29
42  level.4  level.2  50 -7.46 42.54
43  level.4  level.2  50  4.90 54.90
44  level.4  level.2  50  1.96 51.96
45  level.4  level.3  58  1.07 59.07
46  level.4  level.3  58  1.50 59.50
47  level.4  level.3  58 -2.31 55.69
48  level.4  level.3  58  1.06 59.06
49  level.5  level.1  50 -5.05 44.95
50  level.5  level.1  50  0.11 50.11
51  level.5  level.1  50  3.57 53.57
52  level.5  level.1  50 -5.52 44.48
53  level.5  level.2  59 -3.59 55.41
54  level.5  level.2  59 -1.24 57.76
55  level.5  level.2  59  5.39 64.39
56  level.5  level.2  59  0.91 59.91
57  level.5  level.3  58 -2.17 55.83
58  level.5  level.3  58 -5.69 52.31
59  level.5  level.3  58 -1.66 56.34
60  level.5  level.3  58  1.61 59.61
```

Plot of generated data:

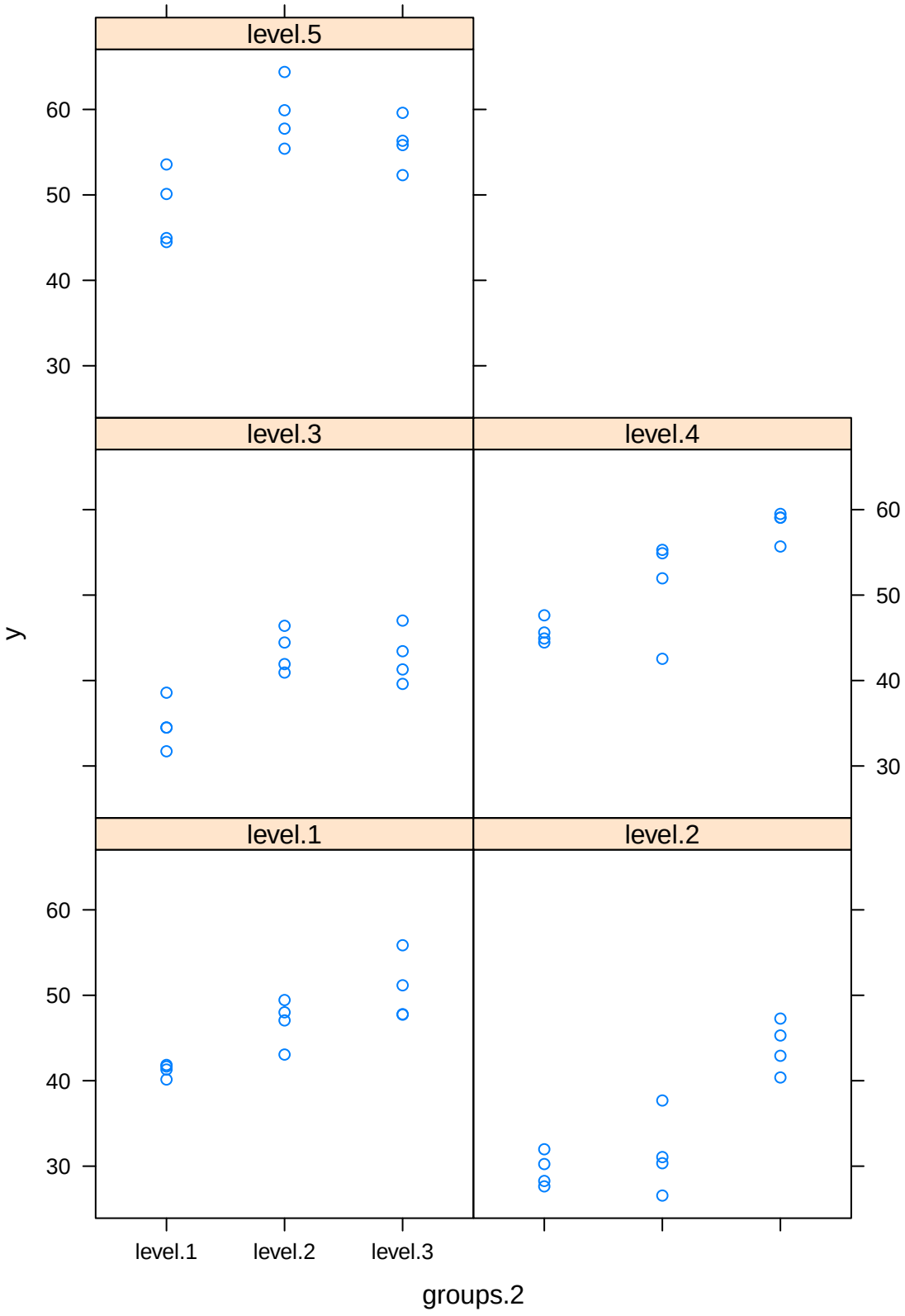
```
> par(mar = c(8, 6, 4, 2))
> boxplot(y ~ groups.1 * groups.2, col = "lightgreen", xlab = "", ylab = "Continuous Response",
+     main = "Simulated data set (groups.2*groups.1)", las = 2, ylim = c(20,
+         70))
> abline(h = 40, col = "red", lwd = 2)
```


Simulated data set (groups.2*groups.1)
Continuous Response
70
60
50
40
30
20
level.1.level.1
level.2.level.1
level.3.level.1
level.4.level.1
level.5.level.1
level.1.level.2
level.2.level.2
level.3.level.2
level.4.level.2
level.5.level.2
level.1.level.3
level.2.level.3
level.3.level.3
level.4.level.3
level.5.level.3

```
> par(mar = c(5, 4, 4, 2))
> library("lattice")
> xyplot(y ~ groups.1 | groups.2, main = "Relationship between y and groups.1 (by groups.2)")
```


Relationship between y and groups.1 (by groups.2)
level.3
level.1
level.2
y
60
50
40
30
level.1
level.2
level.3
level.4
level.5
groups.1

```
> xyplot(y ~ groups.2 | groups.1, main = "Relationship between y and groups.2 (by groups.1)")
```


Relationship between y and groups.2 (by groups.1)
level.5
level.3
level.4
level.1
level.2
y
60
50
40
30
level.1
level.2
level.3
groups.2

6.1 Aside: using simulation to assess the bias and precision of an estimator

```
> print(lm(y ~ groups.1 * groups.2), dig = 3)


Call:
lm(formula = y ~ groups.1 * groups.2)

Coefficients:
                    (Intercept)                   groups.1level.2
                         41.243                           -11.710
                groups.1level.3                   groups.1level.4
                         -6.418                             4.416
                groups.1level.5                   groups.2level.2
                          7.033                             5.650
                groups.2level.3  groups.1level.2:groups.2level.2
                          9.395                            -3.768
groups.1level.3:groups.2level.2  groups.1level.4:groups.2level.2
                          2.954                            -0.135
groups.1level.5:groups.2level.2  groups.1level.2:groups.2level.3
                          5.440                             5.040
groups.1level.3:groups.2level.3  groups.1level.4:groups.2level.3
                         -1.380                             3.275
groups.1level.5:groups.2level.3
                         -1.648
```

Let's compare the true values with the estimated values:

```
> data.frame(true = all.effects, estimates = round(lm(y ~ groups.1 *
+     groups.2)$coef, 2), row.names = names(lm(y ~ groups.1 * groups.2)$coef))

                                true estimates
(Intercept)                       40     41.24
groups.1level.2                  -10    -11.71
groups.1level.3                   -5     -6.42
groups.1level.4                    5      4.42
groups.1level.5                   10      7.03
groups.2level.2                    5      5.65
groups.2level.3                   10      9.39
groups.1level.2:groups.2level.2   -2     -3.77
groups.1level.3:groups.2level.2    3      2.95
groups.1level.4:groups.2level.2    0     -0.14
groups.1level.5:groups.2level.2    4      5.44
groups.1level.2:groups.2level.3    4      5.04
groups.1level.3:groups.2level.3    0     -1.38
groups.1level.4:groups.2level.3    3      3.27
groups.1level.5:groups.2level.3   -2     -1.65
```

We generate 1000 datasets to assess the estimators:

```
> n.iter <- 1000  # Desired number of iterations
> estimates <- array(dim = c(n.iter, length(all.effects)))  # Data structure to hold results
> for (i in 1:n.iter) {
+     # Run simulation n.iter times
```

```
+     if (i%%100 == 0)
+         {
+             cat(i, "iterations out of", n.iter, "done\n")
+         }  # Print every 100th iteration number so that we know how far we got
+     normal.error <- rnorm(n, 0, sigma)  # residuals
+     y <- as.numeric(as.matrix(X) %*% as.matrix(all.effects) + normal.error)  # assemble data
+     fit.model <- lm(y ~ groups.1 * groups.2)  # fit the model
+     estimates[i, ] <- fit.model$coefficients  # keep values of coefs
+ }

100 iterations out of 1000 done
200 iterations out of 1000 done
300 iterations out of 1000 done
400 iterations out of 1000 done
500 iterations out of 1000 done
600 iterations out of 1000 done
700 iterations out of 1000 done
800 iterations out of 1000 done
900 iterations out of 1000 done
1000 iterations out of 1000 done
```

Compare the true values and the mean estimates taken over the 1000 iterations:

```
> data.frame(true = all.effects, mean.estimates = round(apply(estimates,
+     2, mean), 2), row.names = names(lm(y ~ groups.1 * groups.2)$coef))

                               true mean.estimates
(Intercept)                      40          39.96
groups.1level.2                 -10          -9.89
groups.1level.3                  -5          -5.04
groups.1level.4                   5           5.00
groups.1level.5                  10           9.98
groups.2level.2                   5           5.02
groups.2level.3                  10          10.03
groups.1level.2:groups.2level.2  -2          -2.06
groups.1level.3:groups.2level.2   3           3.04
groups.1level.4:groups.2level.2   0          -0.01
groups.1level.5:groups.2level.2   4           4.11
groups.1level.2:groups.2level.3   4           3.89
groups.1level.3:groups.2level.3   0           0.02
groups.1level.4:groups.2level.3   3           2.98
groups.1level.5:groups.2level.3  -2          -1.89
```

6.2 Analysis of two-way ANOVA data

Main effects only:

```
> mainfit <- lm(y ~ groups.1 + groups.2)
> print(summary(mainfit), dig = 3)


Call:
lm(formula = y ~ groups.1 + groups.2)
```

```
Residuals:
   Min     1Q Median     3Q    Max
-9.973 -1.478  0.233  2.672  7.060

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)       38.54       1.27   30.43  < 2e-16 ***
groups.1level.2   -9.63       1.51   -6.36  4.8e-08 ***
groups.1level.3   -2.81       1.51   -1.86    0.069 .
groups.1level.4    7.74       1.51    5.11  4.4e-06 ***
groups.1level.5   10.36       1.51    6.85  8.0e-09 ***
groups.2level.2    5.68       1.17    4.84  1.2e-05 ***
groups.2level.3   12.59       1.17   10.73  6.7e-15 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.71 on 53 degrees of freedom
Multiple R-squared:  0.866,Adjusted R-squared:  0.851
F-statistic: 57.3 on 6 and 53 DF,  p-value: <2e-16
```

The means parameterization of the interaction model:

```
> intfit <- lm(y ~ groups.1 * groups.2 - 1 - groups.1 - groups.2)
> print(summary(intfit), dig = 3)


Call:
lm(formula = y ~ groups.1 * groups.2 - 1 - groups.1 - groups.2)

Residuals:
   Min     1Q Median     3Q    Max
-8.994 -2.004 -0.002  2.465  4.723

Coefficients:
                                Estimate Std. Error t value Pr(>|t|)
groups.1level.1:groups.2level.1    38.05       1.66    22.9   <2e-16 ***
groups.1level.2:groups.2level.1    31.35       1.66    18.9   <2e-16 ***
groups.1level.3:groups.2level.1    36.80       1.66    22.2   <2e-16 ***
groups.1level.4:groups.2level.1    43.87       1.66    26.4   <2e-16 ***
groups.1level.5:groups.2level.1    48.29       1.66    29.1   <2e-16 ***
groups.1level.1:groups.2level.2    43.24       1.66    26.0   <2e-16 ***
groups.1level.2:groups.2level.2    30.78       1.66    18.5   <2e-16 ***
groups.1level.3:groups.2level.2    41.68       1.66    25.1   <2e-16 ***
groups.1level.4:groups.2level.2    53.01       1.66    31.9   <2e-16 ***
groups.1level.5:groups.2level.2    58.05       1.66    35.0   <2e-16 ***
groups.1level.1:groups.2level.3    52.59       1.66    31.7   <2e-16 ***
groups.1level.2:groups.2level.3    42.87       1.66    25.8   <2e-16 ***
groups.1level.3:groups.2level.3    46.97       1.66    28.3   <2e-16 ***
groups.1level.4:groups.2level.3    60.23       1.66    36.3   <2e-16 ***
groups.1level.5:groups.2level.3    58.63       1.66    35.3   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.32 on 45 degrees of freedom
```

```
Multiple R-squared:  0.996,Adjusted R-squared:  0.995
F-statistic:  789 on 15 and 45 DF,  p-value: <2e-16
```

7 Linear mixed-effects models

Mixed-effects or mixed models contain factors, or more generally covariates, with both fixed and random effects. That is, we constrain the values for at least one set of effects (intercepts and/or slopes) to come from a normal distribution (just like we did for the simple example of random-effects ANOVA). The 'random-effects assumption' usually mean that a set of effects comes from a normal distribution; but in general, the random effects can come from any distribution.

There are at least three sets of assumptions that one may make about the random effects for the intercept and/or the slope of regression lines that are estimated from grouped data:

(*i*) only the intercepts are random; slopes are identical for all groups

(*ii*) both intercepts and slopes are random, but they are independent

(*iii*) both intercepts and slopes are random and there is a correlation between them

The additional case in which slopes are random and intercepts are fixed is not a sensible model in most circumstances.

Model (*i*) is often called a random-intercepts model. Both models (*ii*) and (*iii*) are called random-coefficients models. Model (*iii*) is the default for the `lmer()` function when fitting a random-coefficients model.

In this section, we will first generate a random-coefficients data set under model a model of type (*ii*), i.e., with both random intercepts and random slopes, but the two sets of random effects are uncorrelated. We then fit both a random-intercepts model of type (*i*) and a random-coefficients model without correlation of type (*ii*) to this dataset.

Finally, we generate a second data set that includes a correlation between random intercepts and random slopes and adopt the random-coefficients model with correlation between intercepts and slopes, i.e., a model of type (*iii*), to analyze it.

A close examination of how these data sets are assembled (i.e., simulated) will help us better understand how analogous data sets are broken down (i.e., analyzed) using mixed models.

7.1 Data generation: uncorrelated random intercepts and random slopes

The factor for groups:

```
> n.groups <- 56  # Number of groups
> n.sample <- 10  # Number of observations in each group
> (n <- n.groups * n.sample)  # Total number of data points

[1] 560

> (groups <- gl(n = n.groups, k = n.sample))  # Indicator for population

  [1] 1  1  1  1  1  1  1  1  1  1  2  2  2  2  2  2  2  2  2  2  3  3  3
 [24] 3  3  3  3  3  3  3  4  4  4  4  4  4  4  4  4  4  5  5  5  5  5  5
 [47] 5  5  5  5  6  6  6  6  6  6  6  6  6  6  7  7  7  7  7  7  7  7  7
 [70] 7  8  8  8  8  8  8  8  8  8  8  9  9  9  9  9  9  9  9  9  9  10 10
 [93] 10 10 10 10 10 10 10 10 11 11 11 11 11 11 11 11 11 11 12 12 12 12 12
[116] 12 12 12 12 12 13 13 13 13 13 13 13 13 13 13 14 14 14 14 14 14 14 14
[139] 14 14 15 15 15 15 15 15 15 15 15 15 16 16 16 16 16 16 16 16 16 16 17
[162] 17 17 17 17 17 17 17 17 17 18 18 18 18 18 18 18 18 18 18 19 19 19 19
```

```
[185] 19 19 19 19 19 19 20 20 20 20 20 20 20 20 20 21 21 21 21 21 21 21
[208] 21 21 21 22 22 22 22 22 22 22 22 22 22 23 23 23 23 23 23 23 23 23
[231] 24 24 24 24 24 24 24 24 24 24 25 25 25 25 25 25 25 25 25 25 26 26 26
[254] 26 26 26 26 26 26 26 27 27 27 27 27 27 27 27 27 27 28 28 28 28 28 28
[277] 28 28 28 28 29 29 29 29 29 29 29 29 29 29 30 30 30 30 30 30 30 30 30
[300] 30 31 31 31 31 31 31 31 31 31 31 32 32 32 32 32 32 32 32 32 32 33 33
[323] 33 33 33 33 33 33 33 34 34 34 34 34 34 34 34 34 34 35 35 35 35 35
[346] 35 35 35 35 35 36 36 36 36 36 36 36 36 36 36 37 37 37 37 37 37 37 37
[369] 37 37 38 38 38 38 38 38 38 38 38 38 39 39 39 39 39 39 39 39 39 39 40
[392] 40 40 40 40 40 40 40 40 40 41 41 41 41 41 41 41 41 41 41 42 42 42 42
[415] 42 42 42 42 42 42 43 43 43 43 43 43 43 43 43 43 44 44 44 44 44 44 44
[438] 44 44 44 45 45 45 45 45 45 45 45 45 45 46 46 46 46 46 46 46 46 46 46
[461] 47 47 47 47 47 47 47 47 47 47 48 48 48 48 48 48 48 48 48 48 49 49 49
[484] 49 49 49 49 49 49 49 50 50 50 50 50 50 50 50 50 50 51 51 51 51 51 51
[507] 51 51 51 51 52 52 52 52 52 52 52 52 52 52 53 53 53 53 53 53 53 53 53
[530] 53 54 54 54 54 54 54 54 54 54 54 55 55 55 55 55 55 55 55 55 55 56 56
[553] 56 56 56 56 56 56 56 56
56 Levels: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 ... 56
```

Continuous predictor x:

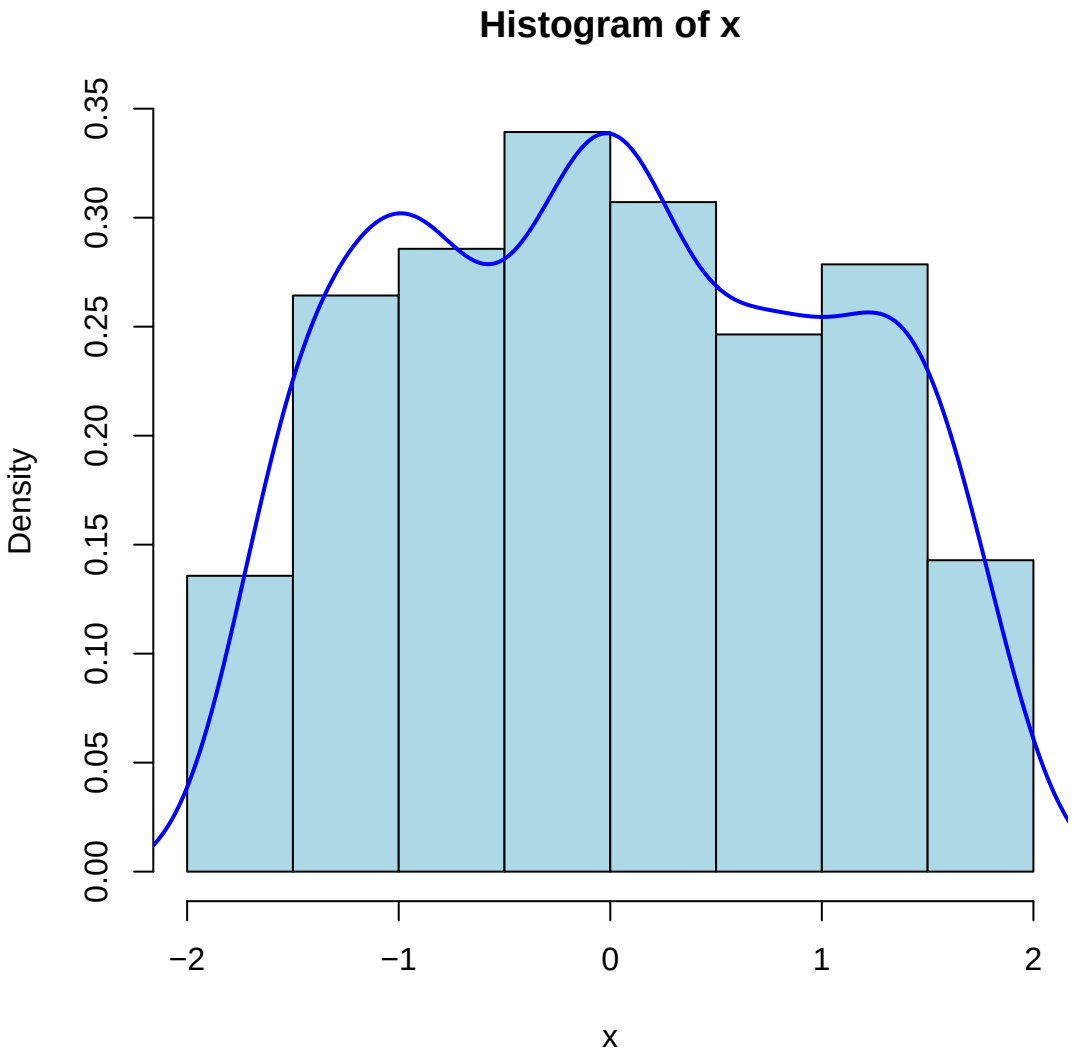
```
> original.x <- runif(n, 45, 70)
> summary(original.x)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   45.0    51.0    57.0    57.1    62.9    70.0
```

We standardize it (generally a good idea with continuous predictors):

```
> x <- scale(original.x)
> mean.orig.x <- mean(original.x)
> sd.orig.x <- sd(original.x)
> x <- (original.x - mean.orig.x)/sd.orig.x
> round(x, 2)

  [1]  0.02  1.37  1.12 -0.35 -1.45 -0.51  0.93  0.26  0.05  0.34 -0.41
 [12] -0.53  1.44  0.75  1.12 -0.88 -0.46 -0.95 -0.02  1.65  0.76  1.52
 [23] -0.44 -0.47  0.42 -1.14  0.96  1.80 -1.26 -0.20 -1.40  1.41  0.37
 [34]  0.85 -0.81  0.45  0.54 -0.20 -1.06 -0.08  0.94 -1.13  0.74 -1.01
 [45] -0.01 -0.19 -0.08 -1.06 -1.40 -1.34  1.41 -0.07 -0.12  1.03 -0.69
 [56] -0.82 -1.04  1.82 -0.59  0.41 -1.28 -0.87  1.31 -0.32  0.10 -0.55
 [67]  0.11  1.72  1.04  0.25 -0.91 -1.52  0.43 -0.46 -1.41 -0.79 -1.35
 [78] -1.64 -1.05 -0.13  0.70 -0.73 -0.08  0.09  1.38 -1.56  1.15 -0.31
 [89]  1.36 -1.32 -1.58  0.67 -1.72  0.11 -0.87  0.93 -0.14 -0.04 -1.30
[100]  0.25 -0.18 -0.30 -0.88  1.71  0.73 -0.21 -0.16 -0.70 -0.20 -0.84
[111]  1.48 -0.73  1.08  0.69  0.27  1.04  0.01  1.31 -0.21  1.53  0.30
[122] -0.55  1.36  0.16 -0.95 -0.89  0.14 -0.07 -0.43 -0.16 -1.43 -0.01
[133]  0.40  0.11  0.11  1.43  0.59 -1.39 -0.53  0.03 -0.78 -1.68 -1.52
[144] -1.11  0.23 -1.01 -1.15 -1.30 -1.12  1.18 -0.89  0.09  1.72  0.10
[155] -1.60 -0.96 -0.76 -1.18 -1.02  1.62 -0.14 -0.29 -0.02 -1.26  1.36
[166]  0.66 -1.68  0.63 -0.79  0.76 -0.72  0.83 -0.20  0.21 -0.84 -1.66
[177]  0.15  0.98 -0.07 -1.66  1.35 -0.27 -1.55  1.34 -1.01  0.61 -1.12
[188]  0.23 -0.22  0.78  0.46  1.22  1.47 -0.02 -1.55  0.33  0.02 -1.13
[199]  1.80  0.95 -1.63 -0.43 -0.80  0.36 -0.94  1.71  0.88  1.03  1.34
```

```
[210] -0.49  0.98  1.68  0.56  0.33 -1.58 -0.84 -0.68 -1.14  0.29  0.56
[221] -1.62 -1.54  0.16  1.66 -1.32 -0.99  1.22  0.88 -0.06  1.38  1.31
[232]  1.55  1.53 -0.19 -0.03  1.36 -0.56 -1.20  1.24 -1.21 -1.60 -1.31
[243]  0.41 -0.14 -0.53 -0.98 -1.43 -1.22 -0.03  0.69  0.12 -1.71  1.55
[254] -0.89  1.02 -1.09 -0.93  0.10 -0.71  0.86  0.63  1.83  0.44 -0.24
[265] -1.26  0.08 -0.43 -0.08 -1.65  0.37 -0.41 -0.99  0.58  0.62  1.36
[276]  1.00 -0.23  1.71  0.75 -0.02 -1.72  0.16  0.32  1.39  0.92  0.68
[287]  1.82 -0.09 -0.69 -1.65 -0.99 -0.65  0.08 -0.23  1.27  0.72 -0.27
[298]  1.42 -1.17 -0.22  0.93  0.61  0.05  0.29  0.21 -0.18 -0.11  1.11
[309] -1.15 -0.01  1.57 -1.55 -0.41 -1.61  0.49 -1.32  0.64 -1.14  1.45
[320]  0.10  1.29  1.05  0.07  0.45 -0.76  1.06  1.32  1.13 -0.23  0.89
[331] -1.32 -0.84  1.38  1.40 -0.68  0.63  0.72 -0.67  0.73  0.80 -0.69
[342] -0.41 -0.77  1.35  1.84 -1.37  0.48  0.14 -0.38  1.62  0.31 -0.40
[353]  0.51 -1.28 -0.86 -0.47 -0.98  0.84 -0.60  1.19 -1.68 -0.38  0.79
[364]  0.76  1.05  0.31  0.25  0.00 -1.23  0.25  1.25 -0.37 -1.59 -1.11
[375]  1.22 -0.02 -0.64 -1.40 -0.97 -1.34  1.75 -1.21 -1.03  0.43  0.85
[386] -0.30  0.40 -0.22  1.75 -1.13 -1.34  0.18 -0.87  1.58  0.82  1.40
[397] -0.68  1.09  1.13  1.51  1.75  1.45 -1.17  0.67  0.56  1.13 -0.98
[408] -0.07  0.91  0.20 -0.20  0.22 -1.19  0.09  1.83 -1.03  1.77  1.05
[419]  1.64 -0.73 -1.36 -1.27 -1.49  1.63  1.32  1.46 -1.13 -1.02 -1.73
[430] -0.43 -0.42  1.42  1.70  0.34  1.61  0.70 -1.43 -0.35  0.44  0.54
[441] -0.11 -0.91  0.72  0.90 -0.78  0.08 -1.09 -1.47 -0.78  0.76  1.33
[452]  0.29 -0.61 -0.07 -1.00 -0.43 -1.56  0.64 -0.71  1.52 -1.34 -1.40
[463] -0.40 -0.97  0.05 -0.04 -0.89 -1.38 -0.67  0.21 -0.56 -0.83  1.45
[474] -0.03 -0.38  1.10  1.39  0.74 -0.96 -1.14 -0.27 -0.18  0.79 -0.95
[485] -0.06  0.15 -0.82  1.29 -1.26  0.50 -0.12  0.99  1.27  1.32  1.34
[496] -0.36  0.79  0.58  1.04  1.40  1.02  0.27 -1.66  1.37 -0.49 -0.16
[507] -1.28  0.39 -1.54  0.46  1.23  0.95 -1.50 -0.64  0.07 -1.54  1.58
[518]  0.16  1.83 -0.39 -0.82  1.72  0.72 -1.42 -1.30  1.46 -0.99  0.24
[529] -0.87 -0.30  0.17 -0.22 -1.63 -1.52  0.05  1.77  1.31 -0.49 -0.48
[540]  1.72  0.16  0.39  0.09 -1.28 -1.61  0.02 -1.58  0.80 -1.00  1.70
[551]  0.69  1.44  0.85  1.15 -0.08 -1.60 -0.84 -0.90  1.03 -1.58

> summary(x)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-1.7300 -0.8700 -0.0216  0.0000  0.8240  1.8400

> (mn <- mean(original.x))

[1] 57.11

> (st.dev <- sd(original.x))

[1] 6.996

> x <- (original.x - mn)/st.dev
> round(x, 2)

  [1]  0.02  1.37  1.12 -0.35 -1.45 -0.51  0.93  0.26  0.05  0.34 -0.41
 [12] -0.53  1.44  0.75  1.12 -0.88 -0.46 -0.95 -0.02  1.65  0.76  1.52
 [23] -0.44 -0.47  0.42 -1.14  0.96  1.80 -1.26 -0.20 -1.40  1.41  0.37
 [34]  0.85 -0.81  0.45  0.54 -0.20 -1.06 -0.08  0.94 -1.13  0.74 -1.01
 [45] -0.01 -0.19 -0.08 -1.06 -1.40 -1.34  1.41 -0.07 -0.12  1.03 -0.69
 [56] -0.82 -1.04  1.82 -0.59  0.41 -1.28 -0.87  1.31 -0.32  0.10 -0.55
```

```
 [67]  0.11  1.72  1.04  0.25 -0.91 -1.52  0.43 -0.46 -1.41 -0.79 -1.35
 [78] -1.64 -1.05 -0.13  0.70 -0.73 -0.08  0.09  1.38 -1.56  1.15 -0.31
 [89]  1.36 -1.32 -1.58  0.67 -1.72  0.11 -0.87  0.93 -0.14 -0.04 -1.30
[100]  0.25 -0.18 -0.30 -0.88  1.71  0.73 -0.21 -0.16 -0.70 -0.20 -0.84
[111]  1.48 -0.73  1.08  0.69  0.27  1.04  0.01  1.31 -0.21  1.53  0.30
[122] -0.55  1.36  0.16 -0.95 -0.89  0.14 -0.07 -0.43 -0.16 -1.43 -0.01
[133]  0.40  0.11  0.11  1.43  0.59 -1.39 -0.53  0.03 -0.78 -1.68 -1.52
[144] -1.11  0.23 -1.01 -1.15 -1.30 -1.12  1.18 -0.89  0.09  1.72  0.10
[155] -1.60 -0.96 -0.76 -1.18 -1.02  1.62 -0.14 -0.29 -0.02 -1.26  1.36
[166]  0.66 -1.68  0.63 -0.79  0.76 -0.72  0.83 -0.20  0.21 -0.84 -1.66
[177]  0.15  0.98 -0.07 -1.66  1.35 -0.27 -1.55  1.34 -1.01  0.61 -1.12
[188]  0.23 -0.22  0.78  0.46  1.22  1.47 -0.02 -1.55  0.33  0.02 -1.13
[199]  1.80  0.95 -1.63 -0.43 -0.80  0.36 -0.94  1.71  0.88  1.03  1.34
[210] -0.49  0.98  1.68  0.56  0.33 -1.58 -0.84 -0.68 -1.14  0.29  0.56
[221] -1.62 -1.54  0.16  1.66 -1.32 -0.99  1.22  0.88 -0.06  1.38  1.31
[232]  1.55  1.53 -0.19 -0.03  1.36 -0.56 -1.20  1.24 -1.21 -1.60 -1.31
[243]  0.41 -0.14 -0.53 -0.98 -1.43 -1.22 -0.03  0.69  0.12 -1.71  1.55
[254] -0.89  1.02 -1.09 -0.93  0.10 -0.71  0.86  0.63  1.83  0.44 -0.24
[265] -1.26  0.08 -0.43 -0.08 -1.65  0.37 -0.41 -0.99  0.58  0.62  1.36
[276]  1.00 -0.23  1.71  0.75 -0.02 -1.72  0.16  0.32  1.39  0.92  0.68
[287]  1.82 -0.09 -0.69 -1.65 -0.99 -0.65  0.08 -0.23  1.27  0.72 -0.27
[298]  1.42 -1.17 -0.22  0.93  0.61  0.05  0.29  0.21 -0.18 -0.11  1.11
[309] -1.15 -0.01  1.57 -1.55 -0.41 -1.61  0.49 -1.32  0.64 -1.14  1.45
[320]  0.10  1.29  1.05  0.07  0.45 -0.76  1.06  1.32  1.13 -0.23  0.89
[331] -1.32 -0.84  1.38  1.40 -0.68  0.63  0.72 -0.67  0.73  0.80 -0.69
[342] -0.41 -0.77  1.35  1.84 -1.37  0.48  0.14 -0.38  1.62  0.31 -0.40
[353]  0.51 -1.28 -0.86 -0.47 -0.98  0.84 -0.60  1.19 -1.68 -0.38  0.79
[364]  0.76  1.05  0.31  0.25  0.00 -1.23  0.25  1.25 -0.37 -1.59 -1.11
[375]  1.22 -0.02 -0.64 -1.40 -0.97 -1.34  1.75 -1.21 -1.03  0.43  0.85
[386] -0.30  0.40 -0.22  1.75 -1.13 -1.34  0.18 -0.87  1.58  0.82  1.40
[397] -0.68  1.09  1.13  1.51  1.75  1.45 -1.17  0.67  0.56  1.13 -0.98
[408] -0.07  0.91  0.20 -0.20  0.22 -1.19  0.09  1.83 -1.03  1.77  1.05
[419]  1.64 -0.73 -1.36 -1.27 -1.49  1.63  1.32  1.46 -1.13 -1.02 -1.73
[430] -0.43 -0.42  1.42  1.70  0.34  1.61  0.70 -1.43 -0.35  0.44  0.54
[441] -0.11 -0.91  0.72  0.90 -0.78  0.08 -1.09 -1.47 -0.78  0.76  1.33
[452]  0.29 -0.61 -0.07 -1.00 -0.43 -1.56  0.64 -0.71  1.52 -1.34 -1.40
[463] -0.40 -0.97  0.05 -0.04 -0.89 -1.38 -0.67  0.21 -0.56 -0.83  1.45
[474] -0.03 -0.38  1.10  1.39  0.74 -0.96 -1.14 -0.27 -0.18  0.79 -0.95
[485] -0.06  0.15 -0.82  1.29 -1.26  0.50 -0.12  0.99  1.27  1.32  1.34
[496] -0.36  0.79  0.58  1.04  1.40  1.02  0.27 -1.66  1.37 -0.49 -0.16
[507] -1.28  0.39 -1.54  0.46  1.23  0.95 -1.50 -0.64  0.07 -1.54  1.58
[518]  0.16  1.83 -0.39 -0.82  1.72  0.72 -1.42 -1.30  1.46 -0.99  0.24
[529] -0.87 -0.30  0.17 -0.22 -1.63 -1.52  0.05  1.77  1.31 -0.49 -0.48
[540]  1.72  0.16  0.39  0.09 -1.28 -1.61  0.02 -1.58  0.80 -1.00  1.70
[551]  0.69  1.44  0.85  1.15 -0.08 -1.60 -0.84 -0.90  1.03 -1.58

> hist(x, col = "lightblue", breaks = 10, freq = FALSE)
> lines(density(x), col = "blue", lwd = 2)
```



This is the model matrix for the means parametrization of the interaction model between the groups and the continuous covariate x:

```
> Xmat <- model.matrix(~groups * x - 1 - x)
> dim(Xmat)

[1] 560 112
```

Where do these dimensions come from?

```
> head(Xmat)

  groups1 groups2 groups3 groups4 groups5 groups6 groups7 groups8 groups9
1       1       0       0       0       0       0       0       0       0
2       1       0       0       0       0       0       0       0       0
3       1       0       0       0       0       0       0       0       0
4       1       0       0       0       0       0       0       0       0
5       1       0       0       0       0       0       0       0       0
```

```
6       1       0       0       0       0       0       0       0       0
  groups10 groups11 groups12 groups13 groups14 groups15 groups16 groups17
1        0        0        0        0        0        0        0        0
2        0        0        0        0        0        0        0        0
3        0        0        0        0        0        0        0        0
4        0        0        0        0        0        0        0        0
5        0        0        0        0        0        0        0        0
6        0        0        0        0        0        0        0        0
  groups18 groups19 groups20 groups21 groups22 groups23 groups24 groups25
1        0        0        0        0        0        0        0        0
2        0        0        0        0        0        0        0        0
3        0        0        0        0        0        0        0        0
4        0        0        0        0        0        0        0        0
5        0        0        0        0        0        0        0        0
6        0        0        0        0        0        0        0        0
  groups26 groups27 groups28 groups29 groups30 groups31 groups32 groups33
1        0        0        0        0        0        0        0        0
2        0        0        0        0        0        0        0        0
3        0        0        0        0        0        0        0        0
4        0        0        0        0        0        0        0        0
5        0        0        0        0        0        0        0        0
6        0        0        0        0        0        0        0        0
  groups34 groups35 groups36 groups37 groups38 groups39 groups40 groups41
1        0        0        0        0        0        0        0        0
2        0        0        0        0        0        0        0        0
3        0        0        0        0        0        0        0        0
4        0        0        0        0        0        0        0        0
5        0        0        0        0        0        0        0        0
6        0        0        0        0        0        0        0        0
  groups42 groups43 groups44 groups45 groups46 groups47 groups48 groups49
1        0        0        0        0        0        0        0        0
2        0        0        0        0        0        0        0        0
3        0        0        0        0        0        0        0        0
4        0        0        0        0        0        0        0        0
5        0        0        0        0        0        0        0        0
6        0        0        0        0        0        0        0        0
  groups50 groups51 groups52 groups53 groups54 groups55 groups56 groups1:x
1        0        0        0        0        0        0        0   0.02228
2        0        0        0        0        0        0        0   1.37446
3        0        0        0        0        0        0        0   1.12387
4        0        0        0        0        0        0        0  -0.35097
5        0        0        0        0        0        0        0  -1.45200
6        0        0        0        0        0        0        0  -0.50900
  groups2:x groups3:x groups4:x groups5:x groups6:x groups7:x groups8:x
1         0         0         0         0         0         0         0
2         0         0         0         0         0         0         0
3         0         0         0         0         0         0         0
4         0         0         0         0         0         0         0
5         0         0         0         0         0         0         0
6         0         0         0         0         0         0         0
  groups9:x groups10:x groups11:x groups12:x groups13:x groups14:x
1         0          0          0          0          0          0
2         0          0          0          0          0          0
```

```
3         0          0          0          0          0          0
4         0          0          0          0          0          0
5         0          0          0          0          0          0
6         0          0          0          0          0          0
  groups15:x groups16:x groups17:x groups18:x groups19:x groups20:x
1          0          0          0          0          0          0
2          0          0          0          0          0          0
3          0          0          0          0          0          0
4          0          0          0          0          0          0
5          0          0          0          0          0          0
6          0          0          0          0          0          0
  groups21:x groups22:x groups23:x groups24:x groups25:x groups26:x
1          0          0          0          0          0          0
2          0          0          0          0          0          0
3          0          0          0          0          0          0
4          0          0          0          0          0          0
5          0          0          0          0          0          0
6          0          0          0          0          0          0
  groups27:x groups28:x groups29:x groups30:x groups31:x groups32:x
1          0          0          0          0          0          0
2          0          0          0          0          0          0
3          0          0          0          0          0          0
4          0          0          0          0          0          0
5          0          0          0          0          0          0
6          0          0          0          0          0          0
  groups33:x groups34:x groups35:x groups36:x groups37:x groups38:x
1          0          0          0          0          0          0
2          0          0          0          0          0          0
3          0          0          0          0          0          0
4          0          0          0          0          0          0
5          0          0          0          0          0          0
6          0          0          0          0          0          0
  groups39:x groups40:x groups41:x groups42:x groups43:x groups44:x
1          0          0          0          0          0          0
2          0          0          0          0          0          0
3          0          0          0          0          0          0
4          0          0          0          0          0          0
5          0          0          0          0          0          0
6          0          0          0          0          0          0
  groups45:x groups46:x groups47:x groups48:x groups49:x groups50:x
1          0          0          0          0          0          0
2          0          0          0          0          0          0
3          0          0          0          0          0          0
4          0          0          0          0          0          0
5          0          0          0          0          0          0
6          0          0          0          0          0          0
  groups51:x groups52:x groups53:x groups54:x groups55:x groups56:x
1          0          0          0          0          0          0
2          0          0          0          0          0          0
3          0          0          0          0          0          0
4          0          0          0          0          0          0
5          0          0          0          0          0          0
6          0          0          0          0          0          0
```

There are 560 observations (rows) and 112 regression terms / variables (columns):

```
> dimnames(Xmat)[[1]]

  [1] "1"   "2"   "3"   "4"   "5"   "6"   "7"   "8"   "9"   "10"  "11"
 [12] "12"  "13"  "14"  "15"  "16"  "17"  "18"  "19"  "20"  "21"  "22"
 [23] "23"  "24"  "25"  "26"  "27"  "28"  "29"  "30"  "31"  "32"  "33"
 [34] "34"  "35"  "36"  "37"  "38"  "39"  "40"  "41"  "42"  "43"  "44"
 [45] "45"  "46"  "47"  "48"  "49"  "50"  "51"  "52"  "53"  "54"  "55"
 [56] "56"  "57"  "58"  "59"  "60"  "61"  "62"  "63"  "64"  "65"  "66"
 [67] "67"  "68"  "69"  "70"  "71"  "72"  "73"  "74"  "75"  "76"  "77"
 [78] "78"  "79"  "80"  "81"  "82"  "83"  "84"  "85"  "86"  "87"  "88"
 [89] "89"  "90"  "91"  "92"  "93"  "94"  "95"  "96"  "97"  "98"  "99"
[100] "100" "101" "102" "103" "104" "105" "106" "107" "108" "109" "110"
[111] "111" "112" "113" "114" "115" "116" "117" "118" "119" "120" "121"
[122] "122" "123" "124" "125" "126" "127" "128" "129" "130" "131" "132"
[133] "133" "134" "135" "136" "137" "138" "139" "140" "141" "142" "143"
[144] "144" "145" "146" "147" "148" "149" "150" "151" "152" "153" "154"
[155] "155" "156" "157" "158" "159" "160" "161" "162" "163" "164" "165"
[166] "166" "167" "168" "169" "170" "171" "172" "173" "174" "175" "176"
[177] "177" "178" "179" "180" "181" "182" "183" "184" "185" "186" "187"
[188] "188" "189" "190" "191" "192" "193" "194" "195" "196" "197" "198"
[199] "199" "200" "201" "202" "203" "204" "205" "206" "207" "208" "209"
[210] "210" "211" "212" "213" "214" "215" "216" "217" "218" "219" "220"
[221] "221" "222" "223" "224" "225" "226" "227" "228" "229" "230" "231"
[232] "232" "233" "234" "235" "236" "237" "238" "239" "240" "241" "242"
[243] "243" "244" "245" "246" "247" "248" "249" "250" "251" "252" "253"
[254] "254" "255" "256" "257" "258" "259" "260" "261" "262" "263" "264"
[265] "265" "266" "267" "268" "269" "270" "271" "272" "273" "274" "275"
[276] "276" "277" "278" "279" "280" "281" "282" "283" "284" "285" "286"
[287] "287" "288" "289" "290" "291" "292" "293" "294" "295" "296" "297"
[298] "298" "299" "300" "301" "302" "303" "304" "305" "306" "307" "308"
[309] "309" "310" "311" "312" "313" "314" "315" "316" "317" "318" "319"
[320] "320" "321" "322" "323" "324" "325" "326" "327" "328" "329" "330"
[331] "331" "332" "333" "334" "335" "336" "337" "338" "339" "340" "341"
[342] "342" "343" "344" "345" "346" "347" "348" "349" "350" "351" "352"
[353] "353" "354" "355" "356" "357" "358" "359" "360" "361" "362" "363"
[364] "364" "365" "366" "367" "368" "369" "370" "371" "372" "373" "374"
[375] "375" "376" "377" "378" "379" "380" "381" "382" "383" "384" "385"
[386] "386" "387" "388" "389" "390" "391" "392" "393" "394" "395" "396"
[397] "397" "398" "399" "400" "401" "402" "403" "404" "405" "406" "407"
[408] "408" "409" "410" "411" "412" "413" "414" "415" "416" "417" "418"
[419] "419" "420" "421" "422" "423" "424" "425" "426" "427" "428" "429"
[430] "430" "431" "432" "433" "434" "435" "436" "437" "438" "439" "440"
[441] "441" "442" "443" "444" "445" "446" "447" "448" "449" "450" "451"
[452] "452" "453" "454" "455" "456" "457" "458" "459" "460" "461" "462"
[463] "463" "464" "465" "466" "467" "468" "469" "470" "471" "472" "473"
[474] "474" "475" "476" "477" "478" "479" "480" "481" "482" "483" "484"
[485] "485" "486" "487" "488" "489" "490" "491" "492" "493" "494" "495"
[496] "496" "497" "498" "499" "500" "501" "502" "503" "504" "505" "506"
[507] "507" "508" "509" "510" "511" "512" "513" "514" "515" "516" "517"
[518] "518" "519" "520" "521" "522" "523" "524" "525" "526" "527" "528"
[529] "529" "530" "531" "532" "533" "534" "535" "536" "537" "538" "539"
[540] "540" "541" "542" "543" "544" "545" "546" "547" "548" "549" "550"
```

```
[551] "551" "552" "553" "554" "555" "556" "557" "558" "559" "560"

> dimnames(Xmat)[[2]]

  [1] "groups1"    "groups2"    "groups3"    "groups4"    "groups5"
  [6] "groups6"    "groups7"    "groups8"    "groups9"    "groups10"
 [11] "groups11"   "groups12"   "groups13"   "groups14"   "groups15"
 [16] "groups16"   "groups17"   "groups18"   "groups19"   "groups20"
 [21] "groups21"   "groups22"   "groups23"   "groups24"   "groups25"
 [26] "groups26"   "groups27"   "groups28"   "groups29"   "groups30"
 [31] "groups31"   "groups32"   "groups33"   "groups34"   "groups35"
 [36] "groups36"   "groups37"   "groups38"   "groups39"   "groups40"
 [41] "groups41"   "groups42"   "groups43"   "groups44"   "groups45"
 [46] "groups46"   "groups47"   "groups48"   "groups49"   "groups50"
 [51] "groups51"   "groups52"   "groups53"   "groups54"   "groups55"
 [56] "groups56"   "groups1:x"  "groups2:x"  "groups3:x"  "groups4:x"
 [61] "groups5:x"  "groups6:x"  "groups7:x"  "groups8:x"  "groups9:x"
 [66] "groups10:x" "groups11:x" "groups12:x" "groups13:x" "groups14:x"
 [71] "groups15:x" "groups16:x" "groups17:x" "groups18:x" "groups19:x"
 [76] "groups20:x" "groups21:x" "groups22:x" "groups23:x" "groups24:x"
 [81] "groups25:x" "groups26:x" "groups27:x" "groups28:x" "groups29:x"
 [86] "groups30:x" "groups31:x" "groups32:x" "groups33:x" "groups34:x"
 [91] "groups35:x" "groups36:x" "groups37:x" "groups38:x" "groups39:x"
 [96] "groups40:x" "groups41:x" "groups42:x" "groups43:x" "groups44:x"
[101] "groups45:x" "groups46:x" "groups47:x" "groups48:x" "groups49:x"
[106] "groups50:x" "groups51:x" "groups52:x" "groups53:x" "groups54:x"
[111] "groups55:x" "groups56:x"
```

There are 56 terms for groups, the coefficients of which will provide the group-specific intercepts – i.e., the intercept random effects, or intercept effects for short. There are 56 terms for the interactions between each group and the continuous covariate x, the coefficients of which will provide the group-specific slopes – i.e., the slope random effects, or slope effects for short.

```
> round(Xmat[1, ], 2)  # Print the top row for each column

   groups1    groups2    groups3    groups4    groups5    groups6
      1.00       0.00       0.00       0.00       0.00       0.00
   groups7    groups8    groups9   groups10   groups11   groups12
      0.00       0.00       0.00       0.00       0.00       0.00
  groups13   groups14   groups15   groups16   groups17   groups18
      0.00       0.00       0.00       0.00       0.00       0.00
  groups19   groups20   groups21   groups22   groups23   groups24
      0.00       0.00       0.00       0.00       0.00       0.00
  groups25   groups26   groups27   groups28   groups29   groups30
      0.00       0.00       0.00       0.00       0.00       0.00
  groups31   groups32   groups33   groups34   groups35   groups36
      0.00       0.00       0.00       0.00       0.00       0.00
  groups37   groups38   groups39   groups40   groups41   groups42
      0.00       0.00       0.00       0.00       0.00       0.00
  groups43   groups44   groups45   groups46   groups47   groups48
      0.00       0.00       0.00       0.00       0.00       0.00
  groups49   groups50   groups51   groups52   groups53   groups54
      0.00       0.00       0.00       0.00       0.00       0.00
  groups55   groups56  groups1:x  groups2:x  groups3:x  groups4:x
```

```
      0.00       0.00       0.02       0.00       0.00       0.00
 groups5:x  groups6:x  groups7:x  groups8:x  groups9:x groups10:x
      0.00       0.00       0.00       0.00       0.00       0.00
groups11:x groups12:x groups13:x groups14:x groups15:x groups16:x
      0.00       0.00       0.00       0.00       0.00       0.00
groups17:x groups18:x groups19:x groups20:x groups21:x groups22:x
      0.00       0.00       0.00       0.00       0.00       0.00
groups23:x groups24:x groups25:x groups26:x groups27:x groups28:x
      0.00       0.00       0.00       0.00       0.00       0.00
groups29:x groups30:x groups31:x groups32:x groups33:x groups34:x
      0.00       0.00       0.00       0.00       0.00       0.00
groups35:x groups36:x groups37:x groups38:x groups39:x groups40:x
      0.00       0.00       0.00       0.00       0.00       0.00
groups41:x groups42:x groups43:x groups44:x groups45:x groups46:x
      0.00       0.00       0.00       0.00       0.00       0.00
groups47:x groups48:x groups49:x groups50:x groups51:x groups52:x
      0.00       0.00       0.00       0.00       0.00       0.00
groups53:x groups54:x groups55:x groups56:x
      0.00       0.00       0.00       0.00

> Xmat[, 1]  # Print all rows for column 1 (group 1)

  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
  1   1   1   1   1   1   1   1   1   1   0   0   0   0   0   0   0   0
 19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
```

```
289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413 414
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 521 522
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558
  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
559 560
  0   0

> round(Xmat[, 57], 2)  # Print all rows for column 57 (group 1:x)

    1     2     3     4     5     6     7     8     9    10    11    12
 0.02  1.37  1.12 -0.35 -1.45 -0.51  0.93  0.26  0.05  0.34  0.00  0.00
   13    14    15    16    17    18    19    20    21    22    23    24
 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
   25    26    27    28    29    30    31    32    33    34    35    36
 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
   37    38    39    40    41    42    43    44    45    46    47    48
 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
   49    50    51    52    53    54    55    56    57    58    59    60
 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
   61    62    63    64    65    66    67    68    69    70    71    72
 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
   73    74    75    76    77    78    79    80    81    82    83    84
 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
   85    86    87    88    89    90    91    92    93    94    95    96
 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
   97    98    99   100   101   102   103   104   105   106   107   108
 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
  109   110   111   112   113   114   115   116   117   118   119   120
```

```
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
 121   122   123   124   125   126   127   128   129   130   131   132
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
 133   134   135   136   137   138   139   140   141   142   143   144
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
 145   146   147   148   149   150   151   152   153   154   155   156
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
 157   158   159   160   161   162   163   164   165   166   167   168
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
 169   170   171   172   173   174   175   176   177   178   179   180
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
 181   182   183   184   185   186   187   188   189   190   191   192
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
 193   194   195   196   197   198   199   200   201   202   203   204
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
 205   206   207   208   209   210   211   212   213   214   215   216
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
 217   218   219   220   221   222   223   224   225   226   227   228
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
 229   230   231   232   233   234   235   236   237   238   239   240
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
 241   242   243   244   245   246   247   248   249   250   251   252
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
 253   254   255   256   257   258   259   260   261   262   263   264
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
 265   266   267   268   269   270   271   272   273   274   275   276
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
 277   278   279   280   281   282   283   284   285   286   287   288
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
 289   290   291   292   293   294   295   296   297   298   299   300
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
 301   302   303   304   305   306   307   308   309   310   311   312
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
 313   314   315   316   317   318   319   320   321   322   323   324
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
 325   326   327   328   329   330   331   332   333   334   335   336
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
 337   338   339   340   341   342   343   344   345   346   347   348
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
 349   350   351   352   353   354   355   356   357   358   359   360
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
 361   362   363   364   365   366   367   368   369   370   371   372
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
 373   374   375   376   377   378   379   380   381   382   383   384
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
 385   386   387   388   389   390   391   392   393   394   395   396
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
 397   398   399   400   401   402   403   404   405   406   407   408
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
 409   410   411   412   413   414   415   416   417   418   419   420
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
 421   422   423   424   425   426   427   428   429   430   431   432
0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00
```

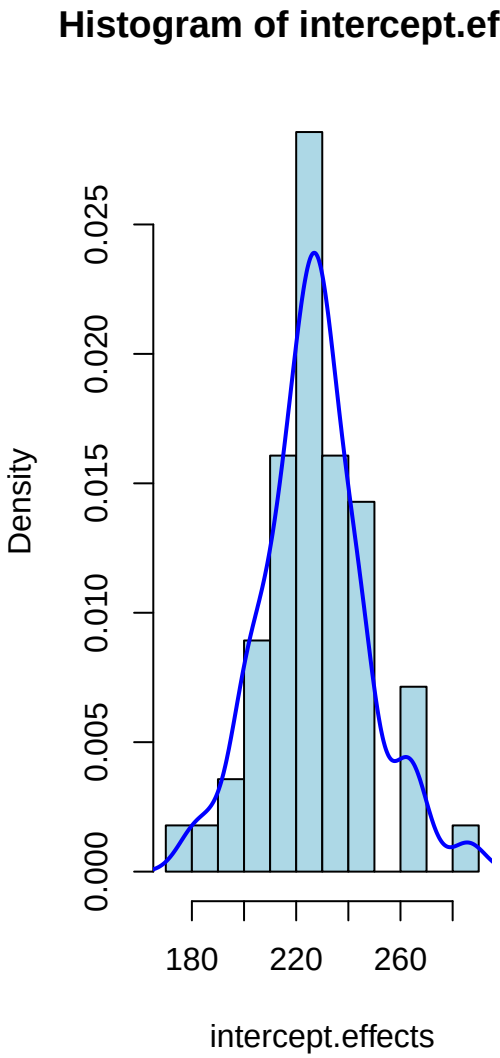
```
 433  434  435  436  437  438  439  440  441  442  443  444
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
 445  446  447  448  449  450  451  452  453  454  455  456
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
 457  458  459  460  461  462  463  464  465  466  467  468
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
 469  470  471  472  473  474  475  476  477  478  479  480
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
 481  482  483  484  485  486  487  488  489  490  491  492
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
 493  494  495  496  497  498  499  500  501  502  503  504
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
 505  506  507  508  509  510  511  512  513  514  515  516
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
 517  518  519  520  521  522  523  524  525  526  527  528
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
 529  530  531  532  533  534  535  536  537  538  539  540
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
 541  542  543  544  545  546  547  548  549  550  551  552
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
 553  554  555  556  557  558  559  560
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
```

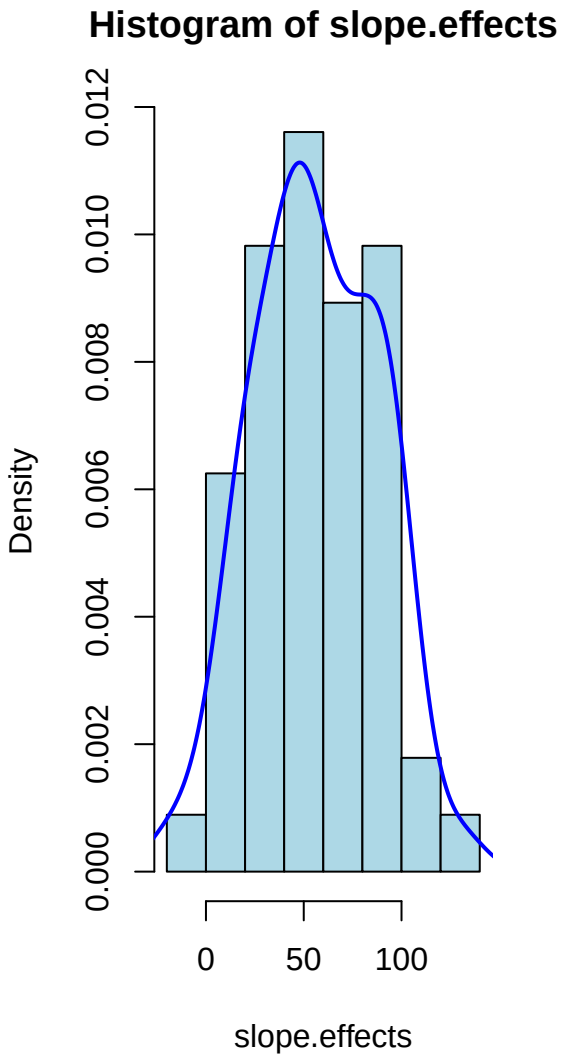
Parameters for the distributions of the random coefficients / random effects – note that the intercepts and slopes come from two independent Gaussian distributions:

```
> intercept.mean <- 230  # mu_alpha
> intercept.sd <- 20  # sigma_alpha
> slope.mean <- 60  # mu_beta
> slope.sd <- 30  # sigma_beta
```

Generate the random coefficients:

```
> intercept.effects <- rnorm(n = n.groups, mean = intercept.mean, sd = intercept.sd)
> slope.effects <- rnorm(n = n.groups, mean = slope.mean, sd = slope.sd)
> par(mfrow = c(1, 2))
> hist(intercept.effects, col = "lightblue", breaks = 10, freq = FALSE)
> lines(density(intercept.effects), col = "blue", lwd = 2)
> hist(slope.effects, col = "lightblue", breaks = 10, freq = FALSE)
> lines(density(slope.effects), col = "blue", lwd = 2)
```






```
> par(mfrow = c(1, 1))
> all.effects <- c(intercept.effects, slope.effects)  # Put them all together
> round(all.effects, 2)

  [1] 228.72 228.50 221.32 217.20 240.39 230.08 263.63 226.03 229.46 261.63
 [11] 285.97 217.17 228.39 197.27 213.72 200.51 211.56 204.72 228.20 233.51
 [21] 230.87 231.30 234.02 179.82 239.65 261.76 217.14 246.56 222.41 248.93
 [31] 245.76 236.59 233.50 222.59 224.18 235.19 199.00 205.95 224.70 216.47
 [41] 203.92 227.63 186.97 240.84 223.49 267.05 214.13 243.31 210.19 242.13
 [51] 227.51 245.65 221.79 213.76 201.54 223.87  24.51  78.16  37.19  47.12
 [61]  37.88  52.91 107.42  72.43  51.51  78.76 130.33  23.61  96.05  41.34
 [71]  54.03  86.60  99.05  61.11  47.87  87.69  18.77  92.26  18.19   2.36
 [81]  84.44  48.75  92.49  57.00  68.52  57.10  99.43 -18.17  35.87  53.42
 [91]  18.80   3.03  76.69  92.47  54.20  39.31  66.81  20.38  50.96  72.92
[101]  39.25  44.49  76.32  38.53  94.46  67.55  22.65 100.76  13.21  19.62
[111]  31.94  91.72
```

Thus, we have two stochastic components in our model *in addition to* the usual stochastic component for the individual-level responses, to which we now turn.

Generating the continuous response variable:

- the deterministic part

```
> lin.pred <- Xmat %*% all.effects  # Value of lin.predictor
> str(lin.pred)

 num [1:560, 1] 229 262 256 220 193 ...
 - attr(*, "dimnames")=List of 2
  ..$ : chr [1:560] "1" "2" "3" "4" ...
  ..$ : NULL
```

- the stochastic part

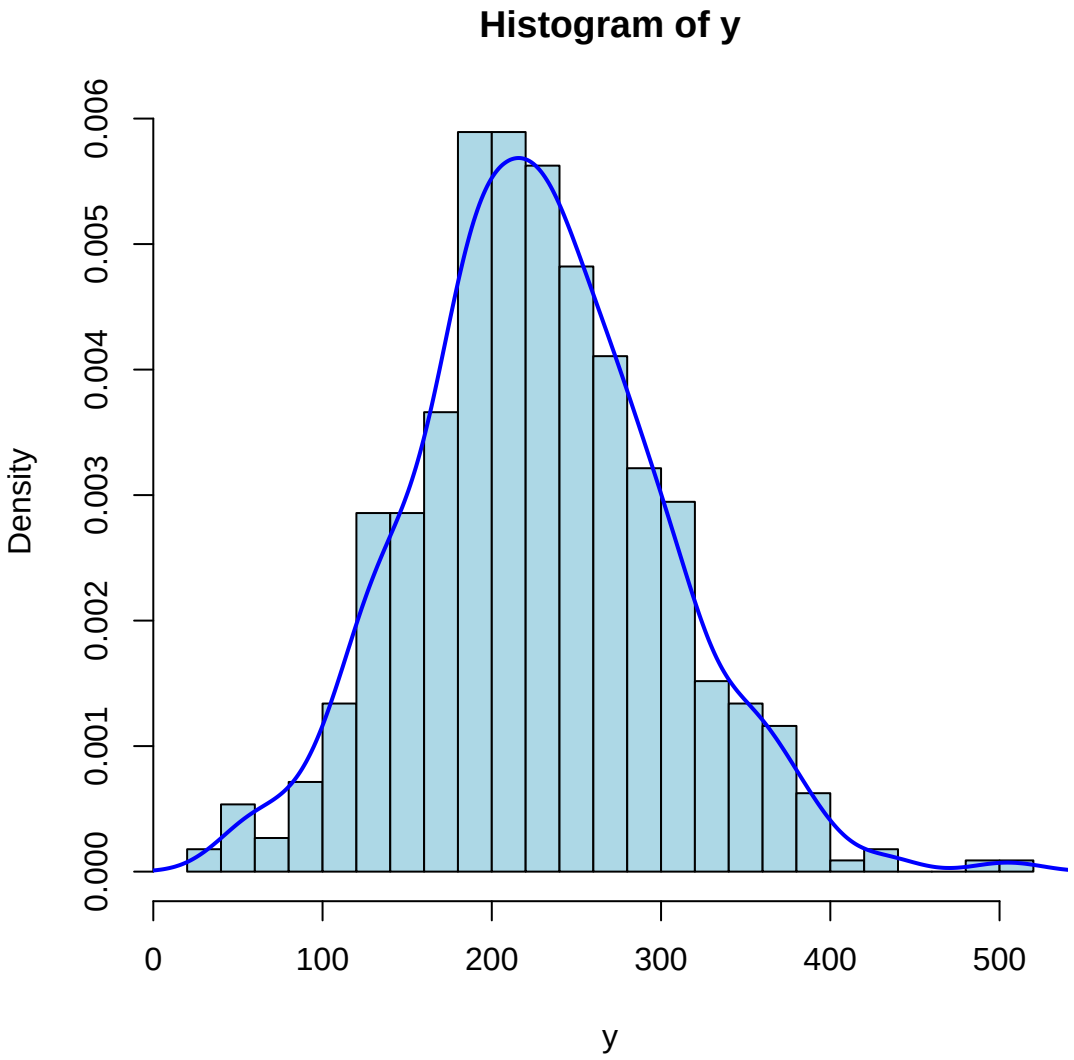
```
> sigma <- 30
> normal.error <- rnorm(n = n, mean = 0, sd = sigma)  # residuals
> str(normal.error)

 num [1:560] 40.047 -36.917 -7.598 -0.952 -19.574 ...
```

- put the two together

```
> y <- lin.pred + normal.error
> str(y)

 num [1:560, 1] 269 225 249 219 174 ...
 - attr(*, "dimnames")=List of 2
  ..$ : chr [1:560] "1" "2" "3" "4" ...
  ..$ : NULL

> # or, alternatively:
> y <- rnorm(n = n, mean = lin.pred, sd = sigma)
> str(y)

 num [1:560] 306 214 267 206 176 ...
```

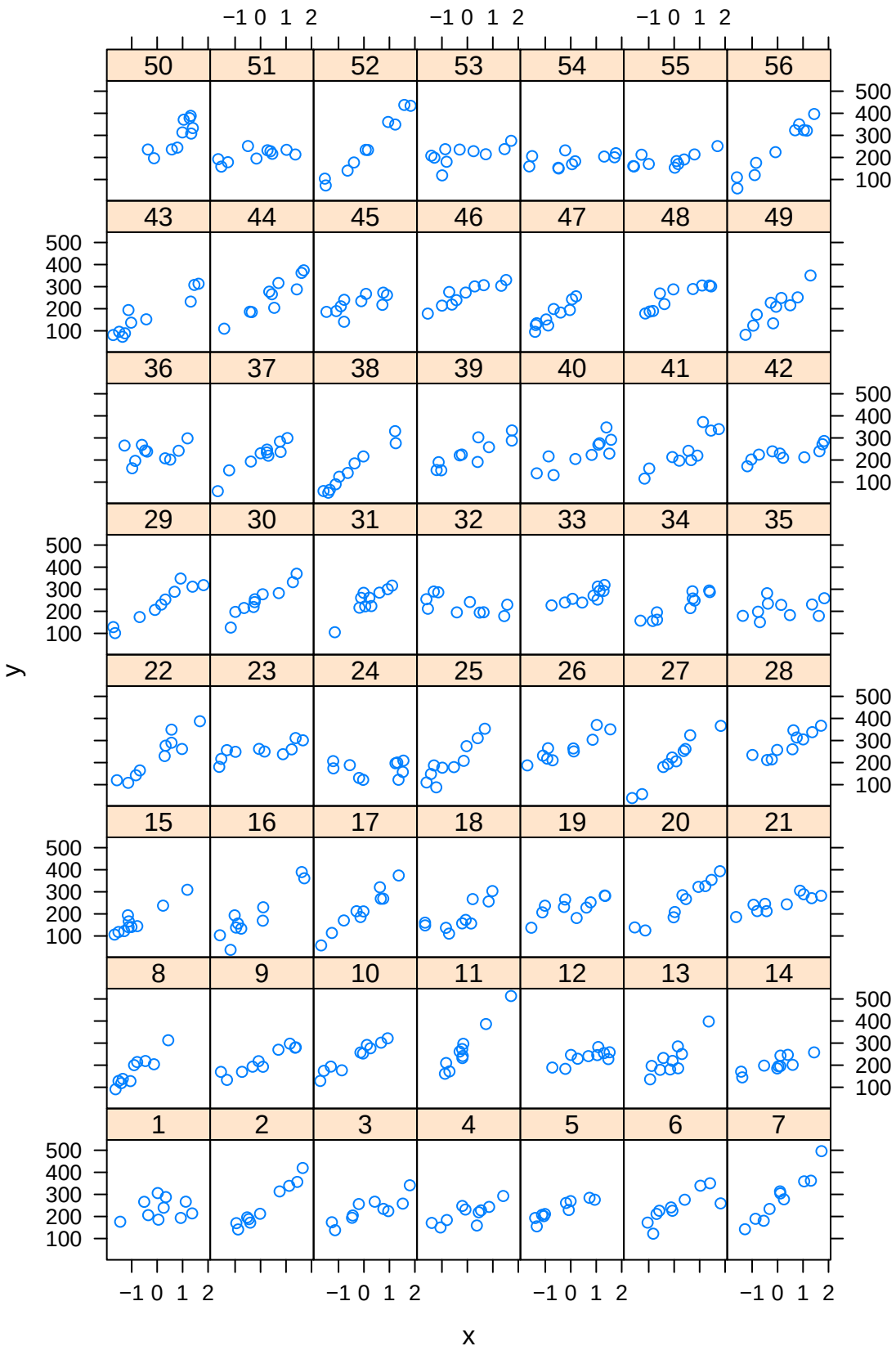
We take a look at the response variable:

```
> hist(y, col = "lightblue", breaks = 30, freq = FALSE)
> lines(density(y), col = "blue", lwd = 2)
```




```
> summary(y)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   37.1   180.0   224.0   227.0   273.0   513.0
```

```
> library("lattice")
> xyplot(y ~ x | groups)
```


50 51 52 53 54 55 56
43 44 45 46 47 48 49
36 37 38 39 40 41 42
29 30 31 32 33 34 35
22 23 24 25 26 27 28
15 16 17 18 19 20 21
8 9 10 11 12 13 14
1 2 3 4 5 6 7
−1 0 1 2
100 200 300 400 500
y
x

7.2 Analysis under a random-intercepts model

REML analysis:

```
> library("lme4")
> lme.fit1 <- lmer(y ~ x + (1 | groups))
> print(lme.fit1, cor = FALSE)

Linear mixed model fit by REML ['lmerMod']
Formula: y ~ x + (1 | groups)
REML criterion at convergence: 5878
Random effects:
 Groups   Name        Std.Dev.
 groups   (Intercept) 20.8
 Residual             43.7
Number of obs: 560, groups: groups, 56
Fixed Effects:
(Intercept)            x
      227.2         54.8

> fixef(lme.fit1)

(Intercept)           x
     227.18       54.79

> ranef(lme.fit1)

$groups
   (Intercept)
1       -1.852
2        9.855
3       -6.012
4      -10.379
5       18.438
6        5.489
7       35.056
8       -2.369
9       -7.253
10      21.207
11      37.004
12     -18.515
13       3.564
14     -13.512
15      -9.803
16     -14.029
17      -3.210
18     -16.592
19       1.785
20       9.633
21      11.194
22       3.452
23      18.535
24     -53.462
25       7.165
26      32.794
```

```
27     -10.463
28      23.393
29       1.876
30      17.225
31       7.517
32       7.286
33       6.132
34      -8.631
35     -16.931
36      10.218
37      -8.408
38     -31.800
39      -1.673
40     -14.344
41      -8.366
42     -12.009
43     -26.132
44       3.295
45       6.655
46      27.869
47     -13.263
48      15.328
49     -14.736
50      19.591
51      -5.665
52      12.292
53      -3.607
54     -30.058
55     -19.424
56       8.653

> coef(lme.fit1)

$groups
   (Intercept)     x
1        225.3 54.79
2        237.0 54.79
3        221.2 54.79
4        216.8 54.79
5        245.6 54.79
6        232.7 54.79
7        262.2 54.79
8        224.8 54.79
9        219.9 54.79
10       248.4 54.79
11       264.2 54.79
12       208.7 54.79
13       230.7 54.79
14       213.7 54.79
15       217.4 54.79
16       213.1 54.79
17       224.0 54.79
18       210.6 54.79
19       229.0 54.79
```

```
20       236.8 54.79
21       238.4 54.79
22       230.6 54.79
23       245.7 54.79
24       173.7 54.79
25       234.3 54.79
26       260.0 54.79
27       216.7 54.79
28       250.6 54.79
29       229.1 54.79
30       244.4 54.79
31       234.7 54.79
32       234.5 54.79
33       233.3 54.79
34       218.5 54.79
35       210.2 54.79
36       237.4 54.79
37       218.8 54.79
38       195.4 54.79
39       225.5 54.79
40       212.8 54.79
41       218.8 54.79
42       215.2 54.79
43       201.0 54.79
44       230.5 54.79
45       233.8 54.79
46       255.0 54.79
47       213.9 54.79
48       242.5 54.79
49       212.4 54.79
50       246.8 54.79
51       221.5 54.79
52       239.5 54.79
53       223.6 54.79
54       197.1 54.79
55       207.8 54.79
56       235.8 54.79

attr(,"class")
[1] "coef.mer"
```

Compare with the true values:

```
> data.frame(intercept.mean = intercept.mean, slope.mean = slope.mean,
+     intercept.sd = intercept.sd, slope.sd = slope.sd, sigma = sigma)

  intercept.mean slope.mean intercept.sd slope.sd sigma
1            230         60           20       30    30

> print(lme.fit1, cor = FALSE)

Linear mixed model fit by REML ['lmerMod']
Formula: y ~ x + (1 | groups)
REML criterion at convergence: 5878
```

```
Random effects:
 Groups   Name        Std.Dev.
 groups   (Intercept) 20.8
 Residual             43.7
Number of obs: 560, groups: groups, 56
Fixed Effects:
(Intercept)            x
      227.2         54.8
```

7.3 Analysis under a random-coefficients model without correlation between intercept and slope

REML analysis:

```
> library("lme4")
> (lme.fit2 <- lmer(y ~ x + (1 | groups) + (0 + x | groups)))

Linear mixed model fit by REML ['lmerMod']
Formula: y ~ x + (1 | groups) + (0 + x | groups)
REML criterion at convergence: 5631
Random effects:
 Groups   Name        Std.Dev.
 groups   (Intercept) 20.7
 groups.1 x           31.1
 Residual             30.5
Number of obs: 560, groups: groups, 56
Fixed Effects:
(Intercept)            x
      228.4         57.4

> fixef(lme.fit2)

(Intercept)           x
     228.37       57.39

> ranef(lme.fit2)

$groups
   (Intercept)       x
1       2.3813 -40.502
2       5.2457  35.902
3      -6.6252 -11.731
4     -13.1711 -17.208
5      17.0972 -12.690
6       5.3677  -1.213
7      34.7253  43.820
8      14.9872  23.338
9      -9.3560  -6.259
10     26.7178   5.985
11     48.2354  61.599
12    -10.1049 -26.719
13      6.1303  30.173
14    -17.9661 -20.336
15     -3.7748  10.387
```

```
16    -10.1006  29.049
17     -2.4511  33.969
18    -20.7500  -3.129
19      1.3721 -23.243
20      2.9337  23.126
21     14.1761 -25.350
22      2.6771  27.805
23     20.3666 -33.060
24    -47.5560 -56.382
25     23.6531  29.286
26     37.2717  -6.787
27    -12.2702  40.688
28     24.1113   4.633
29      0.1679   8.840
30     19.4978  15.162
31      4.3695  22.149
32     -2.5595 -71.648
33     12.1199 -13.909
34    -10.9122  -4.211
35    -14.8037 -44.339
36      6.1554 -37.516
37    -11.0962  14.545
38    -23.9203  33.381
39     -2.4341  -7.533
40    -15.7587  -8.144
41    -17.0503  14.274
42     -6.6694 -32.785
43    -28.6472   7.291
44     -5.2658  19.269
45      2.8681 -21.007
46     31.5734 -11.759
47     -4.3225  19.440
48     17.5682  -8.934
49    -16.5780  25.395
50      2.7330  26.165
51    -12.1835 -36.083
52      6.2690  48.386
53     -8.9282 -31.583
54    -34.2215 -43.958
55    -30.0685 -34.827
56      8.7742  38.786

> coef(lme.fit2)

$groups
   (Intercept)       x
1        230.7  16.888
2        233.6  93.292
3        221.7  45.659
4        215.2  40.182
5        245.5  44.700
6        233.7  56.176
7        263.1 101.210
8        243.4  80.728
```

```
9        219.0  51.131
10       255.1  63.375
11       276.6 118.989
12       218.3  30.671
13       234.5  87.563
14       210.4  37.054
15       224.6  67.777
16       218.3  86.439
17       225.9  91.359
18       207.6  54.261
19       229.7  34.147
20       231.3  80.516
21       242.5  32.040
22       231.0  85.195
23       248.7  24.330
24       180.8   1.008
25       252.0  86.676
26       265.6  50.603
27       216.1  98.078
28       252.5  62.023
29       228.5  66.230
30       247.9  72.552
31       232.7  79.539
32       225.8 -14.258
33       240.5  43.481
34       217.5  53.179
35       213.6  13.051
36       234.5  19.874
37       217.3  71.934
38       204.4  90.771
39       225.9  49.857
40       212.6  49.246
41       211.3  71.664
42       221.7  24.605
43       199.7  64.681
44       223.1  76.659
45       231.2  36.383
46       259.9  45.631
47       224.0  76.830
48       245.9  48.456
49       211.8  82.785
50       231.1  83.555
51       216.2  21.307
52       234.6 105.776
53       219.4  25.806
54       194.1  13.432
55       198.3  22.563
56       237.1  96.176

attr(,"class")
[1] "coef.mer"
```

Compare with true values:

```
> data.frame(intercept.mean = intercept.mean, slope.mean = slope.mean,
+     intercept.sd = intercept.sd, slope.sd = slope.sd, sigma = sigma)

  intercept.mean slope.mean intercept.sd slope.sd sigma
1            230         60           20       30    30

> print(lme.fit2, cor = FALSE)

Linear mixed model fit by REML ['lmerMod']
Formula: y ~ x + (1 | groups) + (0 + x | groups)
REML criterion at convergence: 5631
Random effects:
 Groups   Name        Std.Dev.
 groups   (Intercept) 20.7
 groups.1 x           31.1
 Residual             30.5
Number of obs: 560, groups: groups, 56
Fixed Effects:
(Intercept)            x
      228.4         57.4
```

7.4 The random-coefficients model with correlation between intercepts and slopes

7.4.1 Data generation

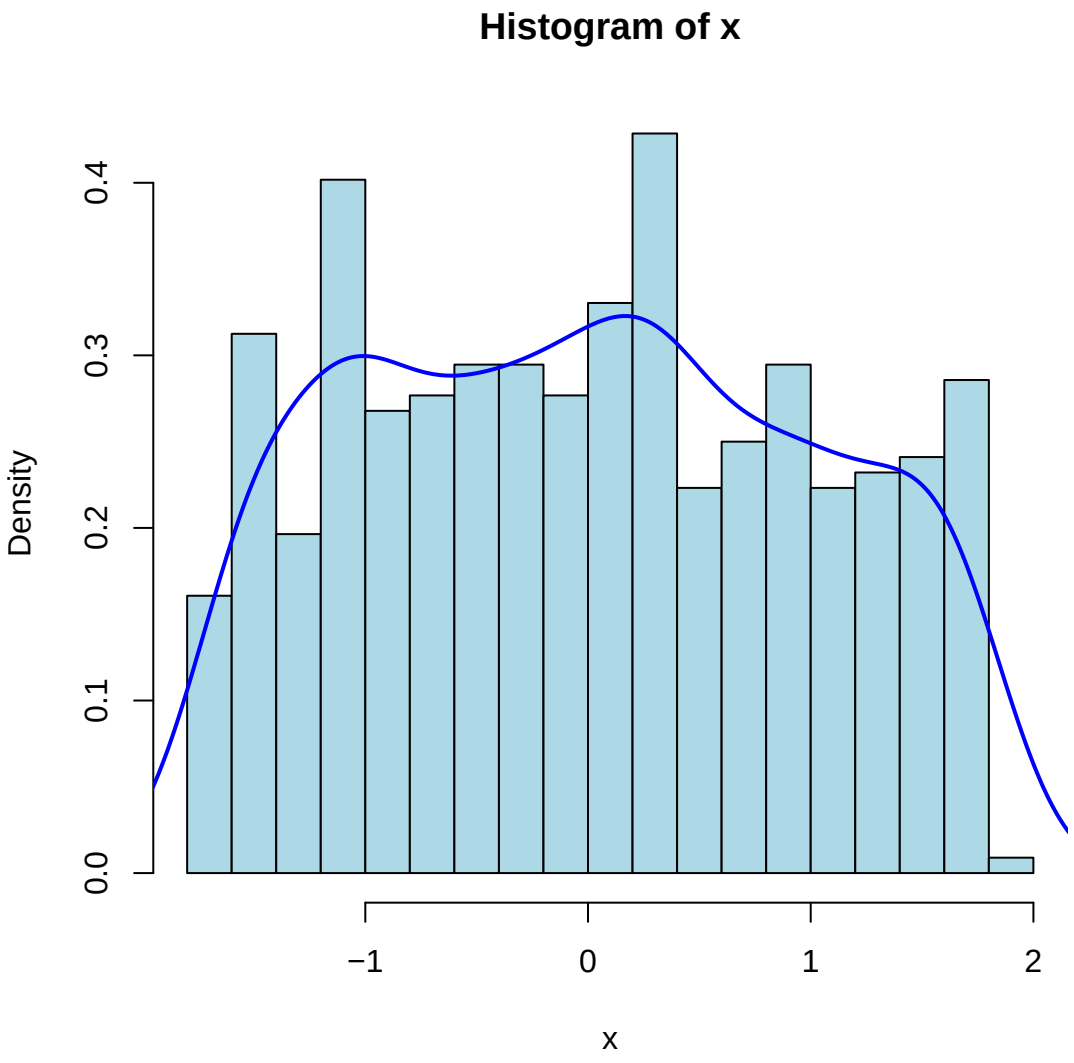
Group factor:

```
> n.groups <- 56
> n.sample <- 10
> (n <- n.groups * n.sample)

[1] 560

> (groups <- gl(n = n.groups, k = n.sample))

  [1] 1  1  1  1  1  1  1  1  1  1  2  2  2  2  2  2  2  2  2  2  3  3  3
 [24] 3  3  3  3  3  3  3  4  4  4  4  4  4  4  4  4  4  5  5  5  5  5  5
 [47] 5  5  5  5  6  6  6  6  6  6  6  6  6  6  7  7  7  7  7  7  7  7  7
 [70] 7  8  8  8  8  8  8  8  8  8  8  9  9  9  9  9  9  9  9  9  9  10 10
 [93] 10 10 10 10 10 10 10 10 11 11 11 11 11 11 11 11 11 11 12 12 12 12 12
[116] 12 12 12 12 12 13 13 13 13 13 13 13 13 13 13 14 14 14 14 14 14 14 14
[139] 14 14 15 15 15 15 15 15 15 15 15 15 16 16 16 16 16 16 16 16 16 16 17
[162] 17 17 17 17 17 17 17 17 18 18 18 18 18 18 18 18 18 18 19 19 19 19
[185] 19 19 19 19 19 20 20 20 20 20 20 20 20 20 20 21 21 21 21 21 21 21
[208] 21 21 21 22 22 22 22 22 22 22 22 22 22 23 23 23 23 23 23 23 23 23 23
[231] 24 24 24 24 24 24 24 24 24 24 25 25 25 25 25 25 25 25 25 25 26 26 26
[254] 26 26 26 26 26 26 27 27 27 27 27 27 27 27 27 27 28 28 28 28 28 28
[277] 28 28 28 28 29 29 29 29 29 29 29 29 29 29 30 30 30 30 30 30 30 30 30
[300] 30 31 31 31 31 31 31 31 31 31 31 32 32 32 32 32 32 32 32 32 32 33 33
[323] 33 33 33 33 33 33 33 33 34 34 34 34 34 34 34 34 34 34 35 35 35 35 35
[346] 35 35 35 35 35 36 36 36 36 36 36 36 36 36 36 37 37 37 37 37 37 37 37
[369] 37 37 38 38 38 38 38 38 38 38 38 38 39 39 39 39 39 39 39 39 39 39 40
[392] 40 40 40 40 40 40 40 40 40 41 41 41 41 41 41 41 41 41 41 42 42 42 42
[415] 42 42 42 42 42 42 43 43 43 43 43 43 43 43 43 43 44 44 44 44 44 44 44
```

```
[438] 44 44 44 45 45 45 45 45 45 45 45 45 45 46 46 46 46 46 46 46 46 46 46
[461] 47 47 47 47 47 47 47 47 47 47 48 48 48 48 48 48 48 48 48 48 49 49 49
[484] 49 49 49 49 49 49 49 50 50 50 50 50 50 50 50 50 50 51 51 51 51 51 51
[507] 51 51 51 51 52 52 52 52 52 52 52 52 52 52 52 53 53 53 53 53 53 53 53 53
[530] 53 54 54 54 54 54 54 54 54 54 54 55 55 55 55 55 55 55 55 55 55 56 56
[553] 56 56 56 56 56 56 56 56
56 Levels: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 ... 56
```

Standardized continuous covariate:

```
> original.x <- runif(n, 45, 70)
> x <- scale(original.x)
> hist(x, col = "lightblue", breaks = 20, freq = FALSE)
> lines(density(x), col = "blue", lwd = 2)
```



Design matrix:

```
> Xmat <- model.matrix(~groups * x - 1 - x)
```

There are 560 observations (rows) and 112 regression terms / variables (columns), just as before:

```
> dimnames(Xmat)[[1]]

  [1] "1"   "2"   "3"   "4"   "5"   "6"   "7"   "8"   "9"   "10"  "11"
 [12] "12"  "13"  "14"  "15"  "16"  "17"  "18"  "19"  "20"  "21"  "22"
 [23] "23"  "24"  "25"  "26"  "27"  "28"  "29"  "30"  "31"  "32"  "33"
 [34] "34"  "35"  "36"  "37"  "38"  "39"  "40"  "41"  "42"  "43"  "44"
 [45] "45"  "46"  "47"  "48"  "49"  "50"  "51"  "52"  "53"  "54"  "55"
 [56] "56"  "57"  "58"  "59"  "60"  "61"  "62"  "63"  "64"  "65"  "66"
 [67] "67"  "68"  "69"  "70"  "71"  "72"  "73"  "74"  "75"  "76"  "77"
 [78] "78"  "79"  "80"  "81"  "82"  "83"  "84"  "85"  "86"  "87"  "88"
 [89] "89"  "90"  "91"  "92"  "93"  "94"  "95"  "96"  "97"  "98"  "99"
[100] "100" "101" "102" "103" "104" "105" "106" "107" "108" "109" "110"
[111] "111" "112" "113" "114" "115" "116" "117" "118" "119" "120" "121"
[122] "122" "123" "124" "125" "126" "127" "128" "129" "130" "131" "132"
[133] "133" "134" "135" "136" "137" "138" "139" "140" "141" "142" "143"
[144] "144" "145" "146" "147" "148" "149" "150" "151" "152" "153" "154"
[155] "155" "156" "157" "158" "159" "160" "161" "162" "163" "164" "165"
[166] "166" "167" "168" "169" "170" "171" "172" "173" "174" "175" "176"
[177] "177" "178" "179" "180" "181" "182" "183" "184" "185" "186" "187"
[188] "188" "189" "190" "191" "192" "193" "194" "195" "196" "197" "198"
[199] "199" "200" "201" "202" "203" "204" "205" "206" "207" "208" "209"
[210] "210" "211" "212" "213" "214" "215" "216" "217" "218" "219" "220"
[221] "221" "222" "223" "224" "225" "226" "227" "228" "229" "230" "231"
[232] "232" "233" "234" "235" "236" "237" "238" "239" "240" "241" "242"
[243] "243" "244" "245" "246" "247" "248" "249" "250" "251" "252" "253"
[254] "254" "255" "256" "257" "258" "259" "260" "261" "262" "263" "264"
[265] "265" "266" "267" "268" "269" "270" "271" "272" "273" "274" "275"
[276] "276" "277" "278" "279" "280" "281" "282" "283" "284" "285" "286"
[287] "287" "288" "289" "290" "291" "292" "293" "294" "295" "296" "297"
[298] "298" "299" "300" "301" "302" "303" "304" "305" "306" "307" "308"
[309] "309" "310" "311" "312" "313" "314" "315" "316" "317" "318" "319"
[320] "320" "321" "322" "323" "324" "325" "326" "327" "328" "329" "330"
[331] "331" "332" "333" "334" "335" "336" "337" "338" "339" "340" "341"
[342] "342" "343" "344" "345" "346" "347" "348" "349" "350" "351" "352"
[353] "353" "354" "355" "356" "357" "358" "359" "360" "361" "362" "363"
[364] "364" "365" "366" "367" "368" "369" "370" "371" "372" "373" "374"
[375] "375" "376" "377" "378" "379" "380" "381" "382" "383" "384" "385"
[386] "386" "387" "388" "389" "390" "391" "392" "393" "394" "395" "396"
[397] "397" "398" "399" "400" "401" "402" "403" "404" "405" "406" "407"
[408] "408" "409" "410" "411" "412" "413" "414" "415" "416" "417" "418"
[419] "419" "420" "421" "422" "423" "424" "425" "426" "427" "428" "429"
[430] "430" "431" "432" "433" "434" "435" "436" "437" "438" "439" "440"
[441] "441" "442" "443" "444" "445" "446" "447" "448" "449" "450" "451"
[452] "452" "453" "454" "455" "456" "457" "458" "459" "460" "461" "462"
[463] "463" "464" "465" "466" "467" "468" "469" "470" "471" "472" "473"
[474] "474" "475" "476" "477" "478" "479" "480" "481" "482" "483" "484"
[485] "485" "486" "487" "488" "489" "490" "491" "492" "493" "494" "495"
[496] "496" "497" "498" "499" "500" "501" "502" "503" "504" "505" "506"
[507] "507" "508" "509" "510" "511" "512" "513" "514" "515" "516" "517"
[518] "518" "519" "520" "521" "522" "523" "524" "525" "526" "527" "528"
```

```
[529] "529" "530" "531" "532" "533" "534" "535" "536" "537" "538" "539"
[540] "540" "541" "542" "543" "544" "545" "546" "547" "548" "549" "550"
[551] "551" "552" "553" "554" "555" "556" "557" "558" "559" "560"
```

```
> dimnames(Xmat)[[2]]
```

```
  [1] "groups1"    "groups2"    "groups3"    "groups4"    "groups5"
  [6] "groups6"    "groups7"    "groups8"    "groups9"    "groups10"
 [11] "groups11"   "groups12"   "groups13"   "groups14"   "groups15"
 [16] "groups16"   "groups17"   "groups18"   "groups19"   "groups20"
 [21] "groups21"   "groups22"   "groups23"   "groups24"   "groups25"
 [26] "groups26"   "groups27"   "groups28"   "groups29"   "groups30"
 [31] "groups31"   "groups32"   "groups33"   "groups34"   "groups35"
 [36] "groups36"   "groups37"   "groups38"   "groups39"   "groups40"
 [41] "groups41"   "groups42"   "groups43"   "groups44"   "groups45"
 [46] "groups46"   "groups47"   "groups48"   "groups49"   "groups50"
 [51] "groups51"   "groups52"   "groups53"   "groups54"   "groups55"
 [56] "groups56"   "groups1:x"  "groups2:x"  "groups3:x"  "groups4:x"
 [61] "groups5:x"  "groups6:x"  "groups7:x"  "groups8:x"  "groups9:x"
 [66] "groups10:x" "groups11:x" "groups12:x" "groups13:x" "groups14:x"
 [71] "groups15:x" "groups16:x" "groups17:x" "groups18:x" "groups19:x"
 [76] "groups20:x" "groups21:x" "groups22:x" "groups23:x" "groups24:x"
 [81] "groups25:x" "groups26:x" "groups27:x" "groups28:x" "groups29:x"
 [86] "groups30:x" "groups31:x" "groups32:x" "groups33:x" "groups34:x"
 [91] "groups35:x" "groups36:x" "groups37:x" "groups38:x" "groups39:x"
 [96] "groups40:x" "groups41:x" "groups42:x" "groups43:x" "groups44:x"
[101] "groups45:x" "groups46:x" "groups47:x" "groups48:x" "groups49:x"
[106] "groups50:x" "groups51:x" "groups52:x" "groups53:x" "groups54:x"
[111] "groups55:x" "groups56:x"
```

```
> round(Xmat[1, ], 2)  # Print the top row for each column
```

```
   groups1    groups2    groups3    groups4    groups5    groups6
      1.00       0.00       0.00       0.00       0.00       0.00
   groups7    groups8    groups9   groups10   groups11   groups12
      0.00       0.00       0.00       0.00       0.00       0.00
  groups13   groups14   groups15   groups16   groups17   groups18
      0.00       0.00       0.00       0.00       0.00       0.00
  groups19   groups20   groups21   groups22   groups23   groups24
      0.00       0.00       0.00       0.00       0.00       0.00
  groups25   groups26   groups27   groups28   groups29   groups30
      0.00       0.00       0.00       0.00       0.00       0.00
  groups31   groups32   groups33   groups34   groups35   groups36
      0.00       0.00       0.00       0.00       0.00       0.00
  groups37   groups38   groups39   groups40   groups41   groups42
      0.00       0.00       0.00       0.00       0.00       0.00
  groups43   groups44   groups45   groups46   groups47   groups48
      0.00       0.00       0.00       0.00       0.00       0.00
  groups49   groups50   groups51   groups52   groups53   groups54
      0.00       0.00       0.00       0.00       0.00       0.00
  groups55   groups56  groups1:x  groups2:x  groups3:x  groups4:x
      0.00       0.00      -0.17       0.00       0.00       0.00
 groups5:x  groups6:x  groups7:x  groups8:x  groups9:x groups10:x
      0.00       0.00       0.00       0.00       0.00       0.00
groups11:x groups12:x groups13:x groups14:x groups15:x groups16:x
```

```
      0.00       0.00       0.00       0.00       0.00       0.00
groups17:x groups18:x groups19:x groups20:x groups21:x groups22:x
      0.00       0.00       0.00       0.00       0.00       0.00
groups23:x groups24:x groups25:x groups26:x groups27:x groups28:x
      0.00       0.00       0.00       0.00       0.00       0.00
groups29:x groups30:x groups31:x groups32:x groups33:x groups34:x
      0.00       0.00       0.00       0.00       0.00       0.00
groups35:x groups36:x groups37:x groups38:x groups39:x groups40:x
      0.00       0.00       0.00       0.00       0.00       0.00
groups41:x groups42:x groups43:x groups44:x groups45:x groups46:x
      0.00       0.00       0.00       0.00       0.00       0.00
groups47:x groups48:x groups49:x groups50:x groups51:x groups52:x
      0.00       0.00       0.00       0.00       0.00       0.00
groups53:x groups54:x groups55:x groups56:x
      0.00       0.00       0.00       0.00
```

We now generate the correlated random intercepts and random slopes. First, we assemble the parameters for the multivariate normal distribution:

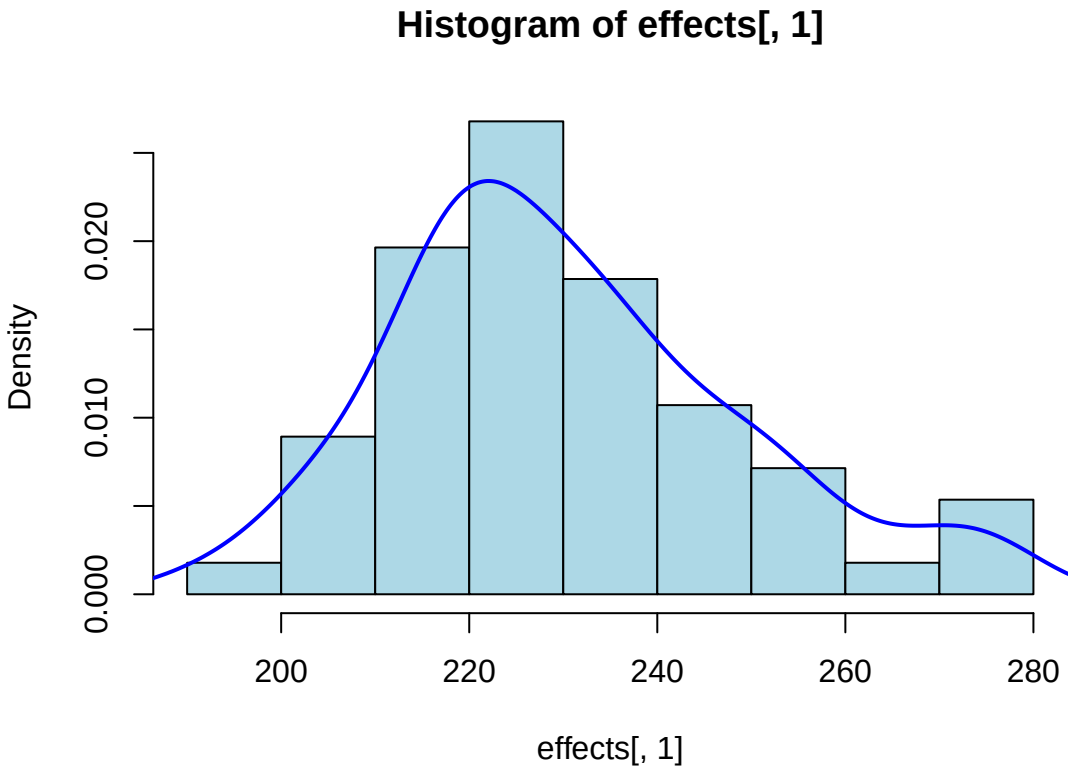
```
> intercept.mean <- 230
> intercept.sd <- 20
> slope.mean <- 60
> slope.sd <- 30
> intercept.slope.covariance <- 10
> (mu.vector <- c(intercept.mean, slope.mean))

[1] 230  60

> (var.covar.matrix <- matrix(c(intercept.sd^2, intercept.slope.covariance,
+     intercept.slope.covariance, slope.sd^2), 2, 2))

     [,1] [,2]
[1,]  400   10
[2,]   10  900
```

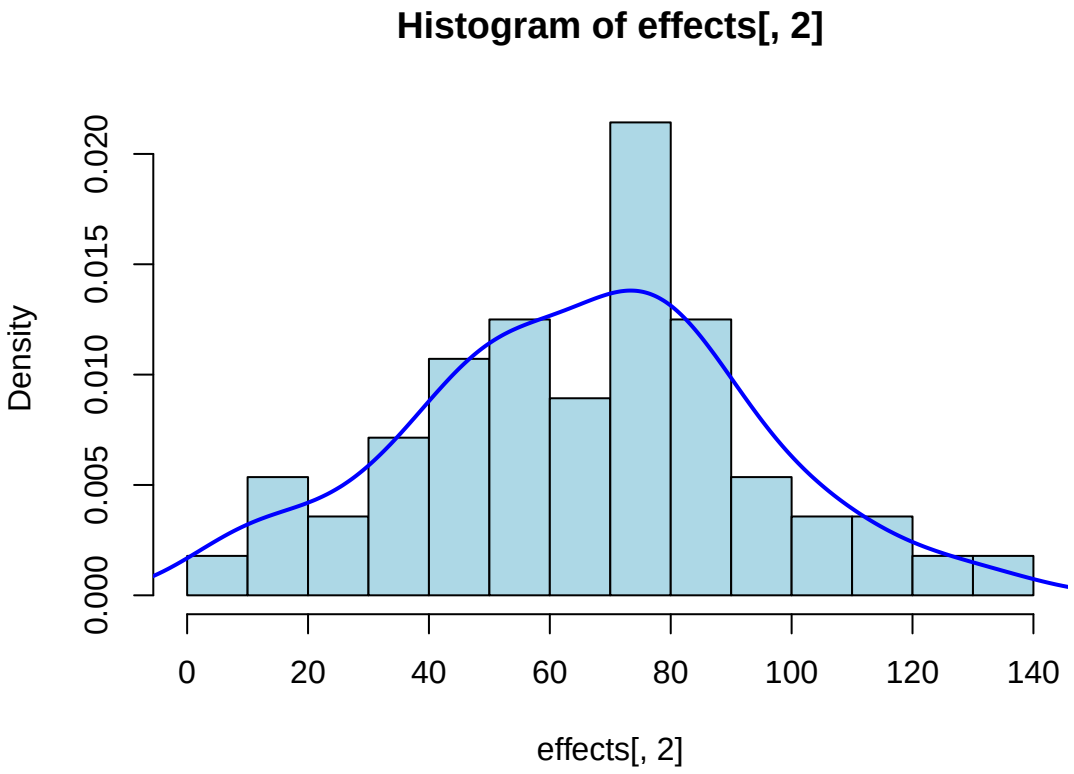
We generate the correlated random effects for intercepts and slopes:

```
> library("MASS")  # Load MASS package for mvrnorm function
> effects <- mvrnorm(n = n.groups, mu = mu.vector, Sigma = var.covar.matrix)
> round(effects, 2)

       [,1]   [,2]
 [1,] 233.1  54.90
 [2,] 228.2  74.32
 [3,] 227.0 110.10
 [4,] 244.8  87.69
 [5,] 232.6  59.21
 [6,] 201.0  88.72
 [7,] 203.2  78.57
 [8,] 243.9   5.86
 [9,] 192.3  71.19
[10,] 218.0  65.50
[11,] 218.9  60.36
[12,] 203.6  49.82
[13,] 240.8  82.92
```

```
[14,] 212.0  76.22
[15,] 227.6  10.65
[16,] 257.9  42.30
[17,] 231.9  70.99
[18,] 248.9  64.71
[19,] 253.4  34.75
[20,] 233.2  80.74
[21,] 248.3 122.52
[22,] 223.2  27.10
[23,] 236.7  39.49
[24,] 238.8  51.64
[25,] 221.9  70.09
[26,] 213.8  56.31
[27,] 253.4  91.44
[28,] 213.4  72.85
[29,] 219.2  47.97
[30,] 238.6 104.11
[31,] 206.6  70.31
[32,] 271.3  93.22
[33,] 239.7  52.36
[34,] 220.0  11.20
[35,] 221.0  15.58
[36,] 216.8  34.71
[37,] 229.5  33.38
[38,] 215.8  99.16
[39,] 225.3 132.77
[40,] 234.9  50.49
[41,] 206.6  79.11
[42,] 218.0  46.79
[43,] 221.4  76.16
[44,] 275.2  60.60
[45,] 248.0  76.70
[46,] 217.2  51.99
[47,] 222.0  64.47
[48,] 265.7  83.99
[49,] 213.6  48.55
[50,] 221.4  89.55
[51,] 255.0  41.50
[52,] 275.2  84.70
[53,] 220.7  76.25
[54,] 234.6  23.88
[55,] 228.8 100.92
[56,] 228.1 111.34
```

```
> par(mfrow = c(2, 1))
> hist(effects[, 1], col = "lightblue", breaks = 10, freq = FALSE)
> lines(density(effects[, 1]), col = "blue", lwd = 2)
> hist(effects[, 2], col = "lightblue", breaks = 10, freq = FALSE)
> lines(density(effects[, 2]), col = "blue", lwd = 2)
```

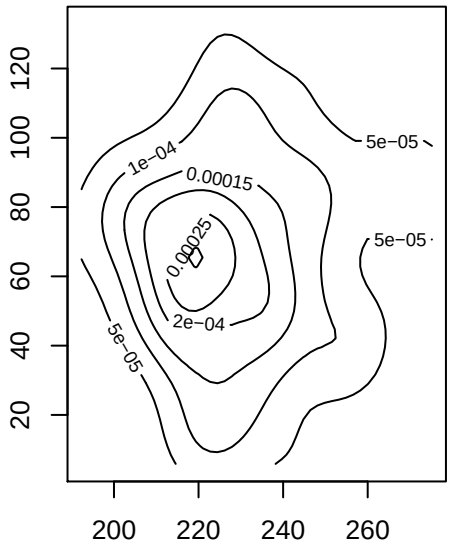

Histogram of effects[, 1]
Density
0.000
0.010
0.020
200
220
240
260
280
effects[, 1]

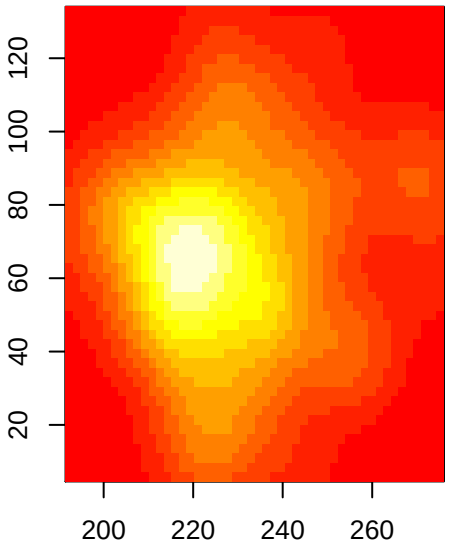

Histogram of effects[, 2]
Density
0.000
0.005
0.010
0.015
0.020
0
20
40
60
80
100
120
140
effects[, 2]

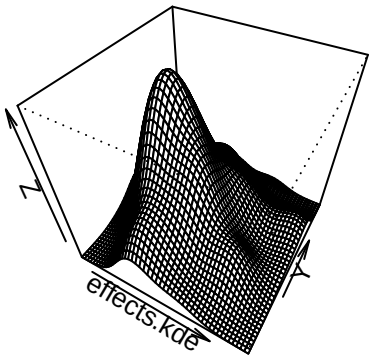
```
> par(mfrow = c(1, 1))
```

Let's plot the bivariate distribution:

```
> par(mfrow = c(2, 2))
> effects.kde <- kde2d(effects[, 1], effects[, 2], n = 50)  # kernel density estimate
> contour(effects.kde)
> image(effects.kde)
> persp(effects.kde, phi = 45, theta = 30)
> par(mfrow = c(1, 1))
```



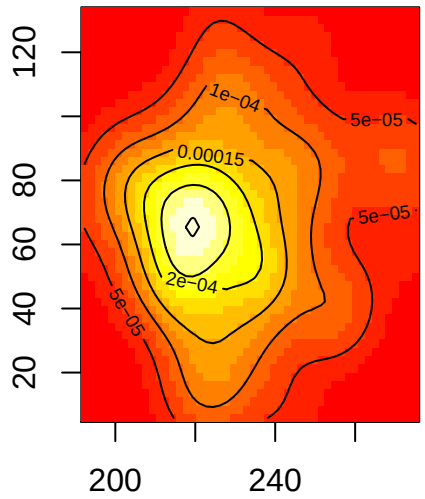


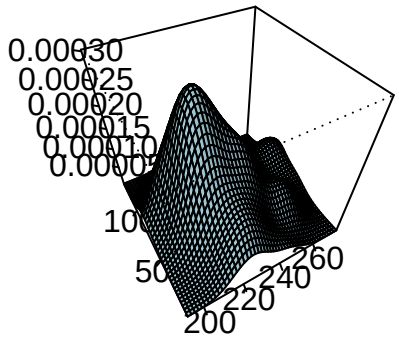


Even better:

```
> par(mfrow = c(1, 2))
> image(effects.kde)
> contour(effects.kde, add = T)
> persp(effects.kde, phi = 45, theta = -30, shade = 0.1, border = NULL,
```

```
+     col = "lightblue", ticktype = "detailed", xlab = "", ylab = "",
+     zlab = "")
```






```
> par(mfrow = c(1, 1))
```

We examine the means, sd.s, and covariance for our sample of random effects and compare them to the population parameters.

```
> apply(effects, 2, mean)

[1] 230.21  65.41

> apply(effects, 2, sd)

[1] 18.62 28.05

> apply(effects, 2, var)

[1] 346.8 787.0

> cov(effects[, 1], effects[, 2])

[1] 26.84

> var(effects)

       [,1]   [,2]
[1,] 346.78  26.84
[2,]  26.84 787.02
```

```
> data.frame(intercept.mean = intercept.mean, slope.mean = slope.mean,
+     intercept.sd = intercept.sd, slope.sd = slope.sd, intercept.slope.covariance = intercept.slope.cov
+     sigma = sigma)

  intercept.mean slope.mean intercept.sd slope.sd
1            230         60           20       30
  intercept.slope.covariance sigma
1                         10    30
```

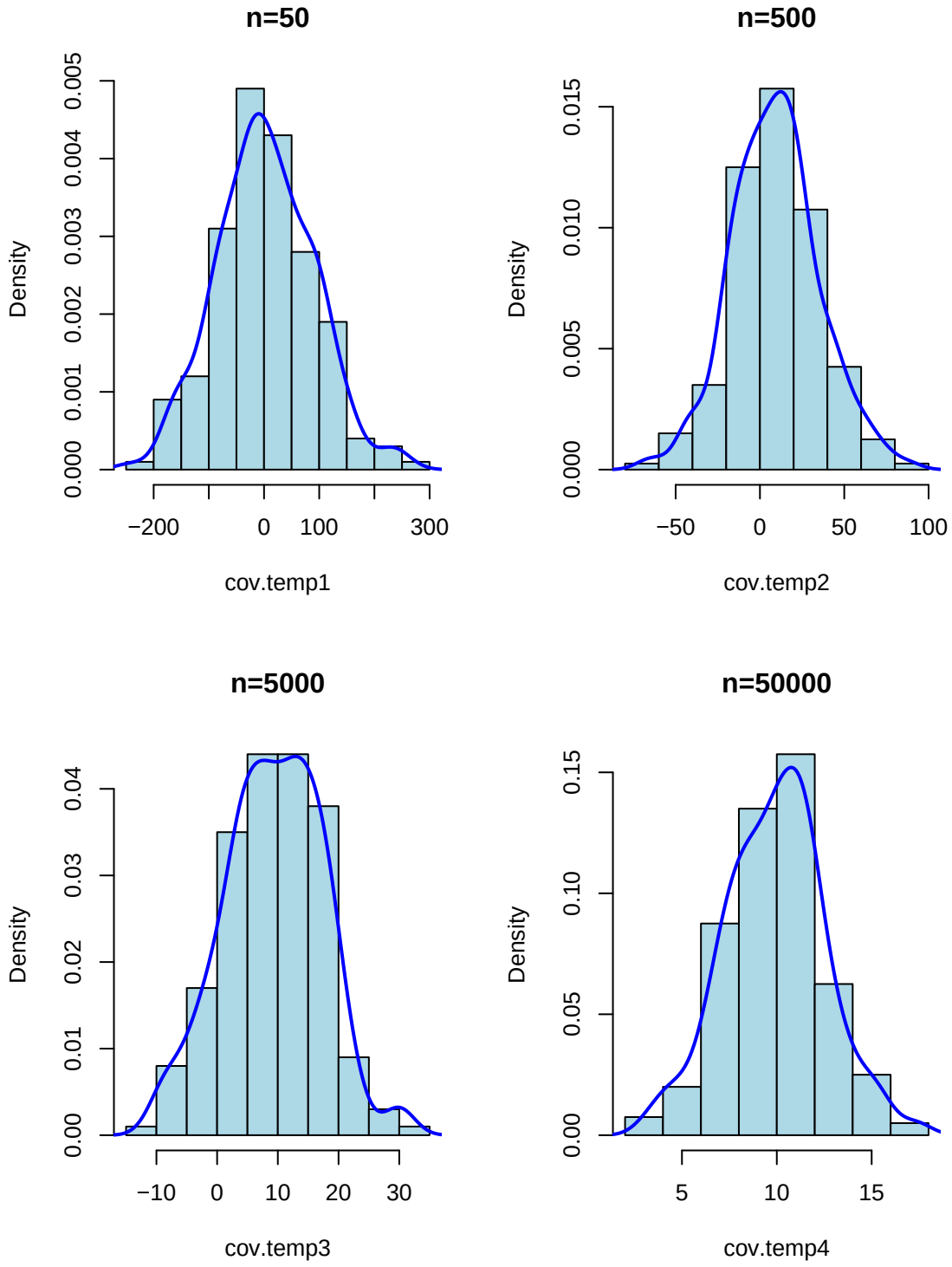
7.4.2 Aside: sampling error for intercept-slope covariance

R will not provide an SE for the covariance estimator (equivalently, for the correlation of random effects). And covariances are even harder to reliably estimate than variances, which are harder than mean estimators (it's easy to estimate measures of center / location, harder to estimate measures of dispersion and even harder to estimate measures of association).

We will see how hard covariances are to estimate by looking at 200 samples of 50, 500, 5000 and 50000 group/random effects per sample. It is only when we get to 5000, or even better, 50000 random effects (e.g., number of participants in an experiment) that the SEs for the covariance or correlation estimates become reasonably small.

We should therefore be highly skeptical of the R parameter estimates for correlations between random effects.

```
> par(mfrow = c(2, 2))
> cov.temp1 <- numeric()
> for (i in 1:200) {
+     temp1 <- mvrnorm(50, mu = mu.vector, Sigma = var.covar.matrix)
+     cov.temp1[i] <- var(temp1)[1, 2]
+ }
> hist(cov.temp1, col = "lightblue", freq = FALSE, main = "n=50")
> lines(density(cov.temp1), col = "blue", lwd = 2)
> cov.temp2 <- numeric()
> for (i in 1:200) {
+     temp2 <- mvrnorm(500, mu = mu.vector, Sigma = var.covar.matrix)
+     cov.temp2[i] <- var(temp2)[1, 2]
+ }
> hist(cov.temp2, col = "lightblue", freq = FALSE, main = "n=500")
> lines(density(cov.temp2), col = "blue", lwd = 2)
> cov.temp3 <- numeric()
> for (i in 1:200) {
+     temp3 <- mvrnorm(5000, mu = mu.vector, Sigma = var.covar.matrix)
+     cov.temp3[i] <- var(temp3)[1, 2]
+ }
> hist(cov.temp3, col = "lightblue", freq = FALSE, main = "n=5000")
> lines(density(cov.temp3), col = "blue", lwd = 2)
> cov.temp4 <- numeric()
> for (i in 1:200) {
+     temp4 <- mvrnorm(50000, mu = mu.vector, Sigma = var.covar.matrix)
+     cov.temp4[i] <- var(temp4)[1, 2]
+ }
> hist(cov.temp4, col = "lightblue", freq = FALSE, main = "n=50000")
> lines(density(cov.temp4), col = "blue", lwd = 2)
```




```
> par(mfrow = c(1, 1))
```

7.4.3 Back to data generation

```
> intercept.effects <- effects[, 1]
> round(intercept.effects, 2)

 [1] 233.1 228.2 227.0 244.8 232.6 201.0 203.2 243.9 192.3 218.0 218.9
[12] 203.6 240.8 212.0 227.6 257.9 231.9 248.9 253.4 233.2 248.3 223.2
[23] 236.7 238.8 221.9 213.8 253.4 213.4 219.2 238.6 206.6 271.3 239.7
[34] 220.0 221.0 216.8 229.5 215.8 225.3 234.9 206.6 218.0 221.4 275.2
[45] 248.0 217.2 222.0 265.7 213.6 221.4 255.0 275.2 220.7 234.6 228.8
[56] 228.1

> slope.effects <- effects[, 2]
> round(slope.effects, 2)

 [1]  54.90  74.32 110.10  87.69  59.21  88.72  78.57   5.86  71.19  65.50
[11]  60.36  49.82  82.92  76.22  10.65  42.30  70.99  64.71  34.75  80.74
[21] 122.52  27.10  39.49  51.64  70.09  56.31  91.44  72.85  47.97 104.11
[31]  70.31  93.22  52.36  11.20  15.58  34.71  33.38  99.16 132.77  50.49
[41]  79.11  46.79  76.16  60.60  76.70  51.99  64.47  83.99  48.55  89.55
[51]  41.50  84.70  76.25  23.88 100.92 111.34

> all.effects <- c(intercept.effects, slope.effects)  # Put them all together
> round(all.effects, 2)

  [1] 233.13 228.21 227.04 244.80 232.55 200.97 203.19 243.87 192.33 217.97
 [11] 218.90 203.62 240.77 212.02 227.63 257.86 231.89 248.91 253.37 233.23
 [21] 248.33 223.22 236.65 238.79 221.89 213.84 253.43 213.39 219.16 238.58
 [31] 206.59 271.29 239.69 220.00 221.03 216.76 229.46 215.75 225.27 234.92
 [41] 206.64 217.99 221.40 275.19 248.02 217.17 221.96 265.67 213.58 221.36
 [51] 254.97 275.17 220.66 234.58 228.82 228.14  54.90  74.32 110.10  87.69
 [61]  59.21  88.72  78.57   5.86  71.19  65.50  60.36  49.82  82.92  76.22
 [71]  10.65  42.30  70.99  64.71  34.75  80.74 122.52  27.10  39.49  51.64
 [81]  70.09  56.31  91.44  72.85  47.97 104.11  70.31  93.22  52.36  11.20
 [91]  15.58  34.71  33.38  99.16 132.77  50.49  79.11  46.79  76.16  60.60
[101]  76.70  51.99  64.47  83.99  48.55  89.55  41.50  84.70  76.25  23.88
[111] 100.92 111.34
```

We generate the response variable:

- the deterministic part

```
> lin.pred <- Xmat %*% all.effects
> round(as.vector(lin.pred), 2)

  [1] 223.64 205.67 288.25 311.95 141.07 306.00 202.35 147.67 299.14 311.51
 [11] 224.08 195.14 257.39 188.08 251.12 223.69 134.57 133.12 162.82 125.94
 [21] 248.78 189.98 227.55 126.95 165.48 399.07 221.62 110.06 391.00 228.72
 [31] 274.45 198.66 246.53 286.37 180.08 155.76 140.95 250.47 267.48 195.82
 [41] 219.04 211.92 278.34 175.23 167.18 165.45 217.99 339.06 181.13 217.29
 [51] 124.24 145.59  56.23 204.80 234.04 113.24 341.09 169.73 118.77  65.79
 [61] 323.52  88.21 107.56 341.21 274.21 343.56 152.88 201.13 296.30 118.34
 [71] 237.20 246.99 247.65 234.53 252.53 251.01 240.71 253.77 253.22 238.32
 [81] 161.05 212.20 303.66 241.81  84.39 169.73 192.93  94.74 247.72 213.37
 [91] 237.38 190.04 153.33 111.15 219.22 331.42 106.74 239.78 278.67 217.98
[101] 271.62 153.13 150.93 321.23 239.99 128.28 147.68 202.89 293.72 259.78
```

```
[111] 268.78 260.76 197.01 153.47 210.89 262.33 287.98 246.50 289.42 223.36
[121] 211.62 255.80 252.39 143.98 305.10 166.12 264.24 296.23 319.79 180.52
[131] 124.44 278.24  89.18 267.72  94.78 240.41 343.36 221.13 146.18 250.89
[141] 234.02 209.78 233.58 239.76 227.25 240.92 219.70 240.36 227.69 217.23
[151] 310.43 325.13 213.19 269.83 254.95 316.46 271.22 285.27 255.84 187.94
[161] 182.44 223.56 247.35 124.24 226.89 209.64 200.90 188.73 280.14 297.11
[171] 256.83 191.20 236.37 269.99 266.85 267.92 261.52 218.47 146.68 176.69
[181] 227.65 310.81 261.12 227.04 226.94 282.56 241.67 268.66 284.61 288.83
[191] 101.27 244.67  96.95 116.10 188.47 204.65 189.90 198.72 353.65 114.04
[201] 394.24  87.12  58.56 447.36  50.24 259.82 117.28 252.91 418.04 125.38
[211] 183.73 246.50 177.92 207.12 246.36 264.30 192.43 245.54 208.20 254.90
[221] 187.98 257.26 202.39 197.76 177.83 182.54 266.92 245.02 265.26 220.64
[231] 184.80 233.86 220.46 262.50 174.04 221.65 319.62 325.94 253.96 217.92
[241] 291.55 297.83 127.64 126.43 176.69 336.85 142.65 120.59 175.87 106.10
[251] 125.81 264.17 127.04 181.06 265.95 132.04 301.06 131.93 210.51 161.36
[261] 236.69 245.30 119.76 379.98 410.09 223.63 383.14 295.00 231.61 283.49
[271] 245.33 131.45 268.16 342.96 284.95 165.06 235.69 105.84  95.21 281.25
[281] 169.74 208.71 172.63 190.33 229.37 236.67 152.17 243.18 267.68 208.50
[291] 215.28 312.74 182.31 302.57 170.26  63.02 247.15 362.73 403.23 312.63
[301] 133.80 104.82 131.74 135.55 266.67 146.08 243.86 138.65 158.50 104.60
[311] 216.89 170.73 251.03 410.34 274.98 222.63 305.37 327.66 429.02 119.24
[321] 250.98 202.22 259.21 219.26 200.42 241.91 213.10 332.72 305.76 192.42
[331] 216.86 231.08 209.90 219.22 214.96 229.89 221.30 221.97 214.01 213.06
[341] 226.00 226.75 237.73 244.19 215.01 210.21 227.51 229.98 228.95 206.23
[351] 179.58 187.63 222.24 219.59 180.69 204.38 235.58 194.37 207.79 163.55
[361] 243.25 275.75 185.36 190.11 219.97 239.89 189.30 259.50 226.77 278.14
[371] 308.86 175.21 247.77 109.79  48.11 318.57 308.35 207.78 150.96 104.02
[381] 361.70 284.28 219.02 129.57 427.29 186.08 408.00 230.86 456.52 292.24
[391] 163.12 316.46 200.12 204.16 164.17 285.11 176.57 164.50 263.84 239.52
[401] 265.69 166.90 123.57  97.03 242.99 344.07 146.78 187.33 335.87  73.33
[411] 281.38 209.61 176.86 258.07 145.89 209.09 164.51 218.22 149.40 165.18
[421] 200.22 271.75 327.65 218.34 288.75 138.91 345.25 296.16 249.47 277.95
[431] 353.72 349.59 266.52 241.49 345.96 346.42 280.65 216.94 205.34 340.51
[441] 302.56 371.06 220.36 243.06 335.63 368.63 264.59 314.66 231.69 233.15
[451] 280.89 275.58 189.89 227.16 251.87 252.58 140.83 229.66 169.52 267.39
[461] 290.70 301.22 119.48 329.30 258.60 248.21 121.93 122.97 214.93 297.78
[471] 367.92 204.74 353.34 174.06 165.83 242.73 163.23 293.74 382.16 249.69
[481] 282.88 175.93 151.86 221.45 181.57 151.93 142.96 251.00 172.31 252.84
[491] 344.92 382.94 307.05 353.56 239.80 131.43 115.49 363.90 272.66 120.73
[501] 265.29 211.29 323.54 244.71 202.38 207.34 232.26 212.21 290.24 305.62
[511] 265.24 157.47 326.27 298.19 306.39 213.73 334.15 139.03 218.51 147.81
[521] 308.47 302.97 323.42 327.27 150.42 239.74 222.19 144.99 152.36 345.79
[531] 243.90 242.67 273.71 243.87 231.32 246.53 222.11 276.96 237.99 235.54
[541] 255.69 409.96 244.55  99.58 392.28 306.96 285.53 229.96 369.78 389.62
[551] 423.36 335.48 420.14 255.96  50.27 178.62 391.79 211.30 404.00 268.73
```

- the stochastic part

```
> sigma <- 30
> (normal.error <- rnorm(n = n, mean = 0, sd = sigma))  # residuals

  [1] -28.395643 -45.800557 -17.339505   5.753736  35.958756  12.159853
```

```
  [7]   9.934676 -13.877635  19.333586  14.018005 -31.938057 -45.907647
 [13]  10.326254   3.769853  14.863402 -12.273083 -29.593024 -25.131012
 [19] -45.703781  11.398331 -49.225326 -16.503834   2.627466   1.147430
 [25]  57.309777 -12.098314  21.242042  -0.479377 -50.472983  24.390947
 [31]  13.257131  20.511240   8.049786  46.419673  22.317902 -31.830881
 [37] -67.782929   5.274419  17.531696  37.883545 -41.875700 -15.280942
 [43]  33.529886 -30.456394 -32.112499  -8.463070  -3.154702 -46.456462
 [49]   0.244777  48.036408  29.090802  47.440158 -19.506744   0.851774
 [55]   0.434674 -30.480203  -4.502339 -85.873239  29.877844  -3.562819
 [61]  16.141334  -1.350901  12.491841 -20.873454  20.582895  11.739070
 [67] -42.740847  75.269062  17.734664  34.062594  23.988913   6.748172
 [73]  44.209723 -49.476894  13.082477  -7.122086   3.117265  12.088679
 [79]  31.995450   3.843505  27.796797  -6.219092  36.273774  -1.925326
 [85] -55.251512 -18.264071 -25.397204 -19.039387  27.229115  13.228631
 [91] -47.021347  11.546429  -1.874281  -8.631505  31.010959  10.574290
 [97] -33.631426  15.077195   8.467203 -13.171126 -45.009264   4.236325
[103] -36.782255  13.968938  23.392898   9.245206 -36.477090   6.589948
[109]  26.270743  -1.025589  32.035898  29.854702 -34.561178  36.940800
[115]  89.940250 -54.517411  24.112806  50.197471   1.388826  -7.601288
[121] -34.782427 -22.276860 -35.535326 -46.893872 -39.737985 -29.861657
[127]   3.857408 -24.978457 -18.490960 -28.530737 -32.916142 -30.088249
[133] -34.302143 -34.552016 -31.574713  12.051917  34.538040   2.263610
[139]   7.784002  -2.932296  28.861753   8.546293 -26.904501 -59.569073
[145]  26.149964  40.293316 -28.371597 -16.835653   1.168995 -25.418500
[151] -31.341398  12.855146  16.674112   4.261657   6.452047  -4.299720
[157] -13.236005  12.974959  13.320013 -16.784534  -1.239760 -10.962782
[163] -33.977396 -56.088843 -22.667159  25.731795  12.370770 -13.401547
[169]  58.404638  13.259009 -53.849903 -30.055214  54.327401  -0.593007
[175]   8.262326   0.008975  18.477881 -35.941371   8.884470   7.346321
[181]  37.429891  -1.969128 -57.210866   4.845552  49.525771  -4.618426
[187]  41.282963  19.794326 -10.333731  23.828496   1.716517 -32.366442
[193] -38.906307  14.614149   9.661759 -46.413239   4.526140  19.918083
[199]   8.983318  12.659761  16.391189 -15.353510  21.484873  -2.551287
[205]  11.058582 -74.226981 -61.767553 -11.115456  20.180974  33.521657
[211]  11.565617   1.992055   3.540662  22.972552  16.278772   7.560230
[217]  18.387462  40.831574 -17.319803  -3.814617  46.218650 -34.897296
[223]  22.527492   8.486905  -7.968792 -16.973666  53.461809 -43.297812
[229] -18.689309 -26.521481 -16.606422 -11.500657 -20.532877  15.224277
[235]  35.832978  29.624513   6.846830  -5.310925 -33.761099  -0.341988
[241]  53.786528 -15.036799  14.129144 -26.032227  -9.539829   6.287752
[247]   7.253720 -14.087789  21.481989 -45.244884 -40.932046  -6.559864
[253] -37.280240 -21.756007   5.296438 -28.029998  12.152798  44.154551
[259] -20.039420 -50.114216 -38.986069  35.665493 -28.944532 -61.299885
[265] -45.637497   7.858007  49.617986 -13.552482 -21.615093   8.947067
[271]  -1.675668 -41.078407 -30.942058   0.845773 -31.766777   7.173220
[277]  28.059880   1.907486  -8.052937   0.757340  -9.758492  45.365011
[283]  25.632594  24.967977  -3.074643  -5.189537 -17.778225  13.313444
[289] -31.743474  -8.585671   7.281486  28.796378 -24.586137   5.098996
[295]  40.085920  38.521674  17.114463   8.620460  10.276657  -8.523734
[301] -10.764923  -8.951699   1.638250  14.485689  -6.935057 -19.540487
[307] -20.482644   9.855934   4.745353  30.657582  30.608247  10.082725
[313]  48.651735  10.475701   6.804629 -34.169177 -23.619843  97.580794
[319] -60.506801  -3.438663 -21.927197  -6.619755  15.051484  -1.573193
```

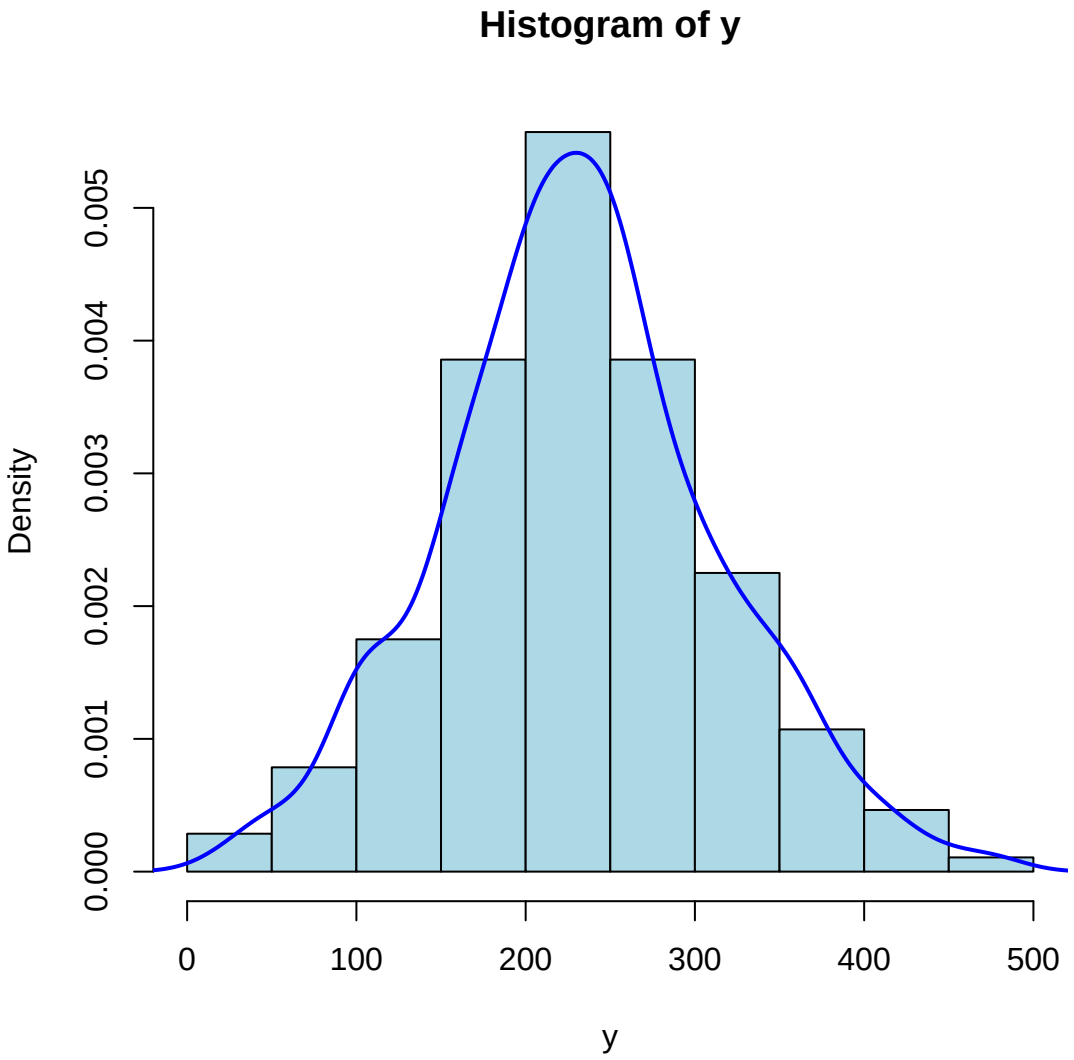
```
[325]  -4.089625  -1.949364  -8.875100 -15.308570 -17.236569  -8.221650
[331]  38.973849  -0.679098 -82.155445   3.758009 -10.346979  -5.679308
[337] 104.932244  31.058346  -0.121829 -11.328504 -31.877987 -42.935348
[343]   0.616306 -47.259203 -10.859366 -11.271620  10.009401  -7.566683
[349] -37.281826  -6.517056  54.079600 -38.528071   2.615131 -75.138314
[355]  25.547522 -15.413665  22.259465  12.273066 -34.029117 -49.523953
[361]  24.685517 -25.134617  54.699531  22.393395  19.283586   1.537982
[367] -27.052965 -47.940409  18.534105 -47.255311   3.249671  34.236106
[373]   4.829083  40.027201  64.079509  16.066711  -5.609448 -53.983951
[379] -59.429869  12.659668  23.390712 -43.671080  -7.272707   3.151474
[385] -42.184966  37.055409   0.766739 -48.957369  -1.257610  -4.731670
[391]  16.170873 -39.204135  -2.081447 -19.757817  17.226529  22.963791
[397] -40.514934 -21.357565 -40.262757  10.668177 -16.502514 -31.270827
[403] -63.968567 -25.857395  39.944531 -20.238113  54.967385  53.044691
[409] -76.262064  -2.137163 -22.883251 -28.100671 -14.316219  41.113585
[415]  -0.361117  -9.292324  15.924534  13.282379 -15.420714  41.174894
[421]   4.683607  46.482503   8.400284 -12.901927 -11.769241  -6.753774
[427]  31.761884  25.182685  17.497253 -27.046283 -35.673362  -5.032155
[433]   4.461690  -7.329417   2.272617  -5.814325  32.088231 -10.787468
[439] -51.221706 -39.768197  -6.538900  -7.385468  34.920910 -27.352243
[445]  11.896029  39.885138  25.280295  13.398571  -4.942128  27.629444
[451]   9.277938   7.514149 -15.323796  28.530465  15.602668 -43.797267
[457] -11.567458 -23.133405 -37.194604   3.802187 -18.549095  16.392896
[463]  -4.435378  14.598561  13.222946 -26.088734  32.331579   5.077230
[469]  -2.373111   9.911038  -3.386969 -15.332781 -21.760144  37.008032
[475]  20.373373 -41.022232 -11.744493 -52.976709  -2.375077  25.254187
[481]  18.865842 -58.316675  65.488838  14.917764 -29.211675  33.700082
[487]  16.013448   8.528915  22.047752  28.127671  -1.818533 -13.540402
[493] -24.204428  47.327209  39.722579  -7.929612   2.597581   8.830939
[499]   0.199828 -10.414892  -3.444581 -21.245478  -1.602401  47.443804
[505] -22.223981  34.355220  76.559548 -35.279339  17.037159 -18.747436
[511]   4.714153  14.684007  27.100530 -37.667741  20.348022  43.847031
[517]  47.132255  10.252229  -7.104332 -31.613504 -20.652870   8.807742
[523] -50.245609  -2.353328  10.426519 -25.442043  17.074112   1.683805
[529]  -1.764416  20.051883 -16.642658  -6.673471   3.193861  -2.090951
[535] -32.885735  14.734213   5.782200 -29.447776  49.033601 -19.165895
[541] -10.028897  19.879758 -11.182766  46.724448  23.347466 -36.832174
[547]   2.784089  11.104468  -2.071856   8.867018 -65.439132 -37.275554
[553]  35.093296 -10.396216  16.979319  27.497928 -31.400578  36.624666
[559]   8.205165 -23.195126
```

- add them together

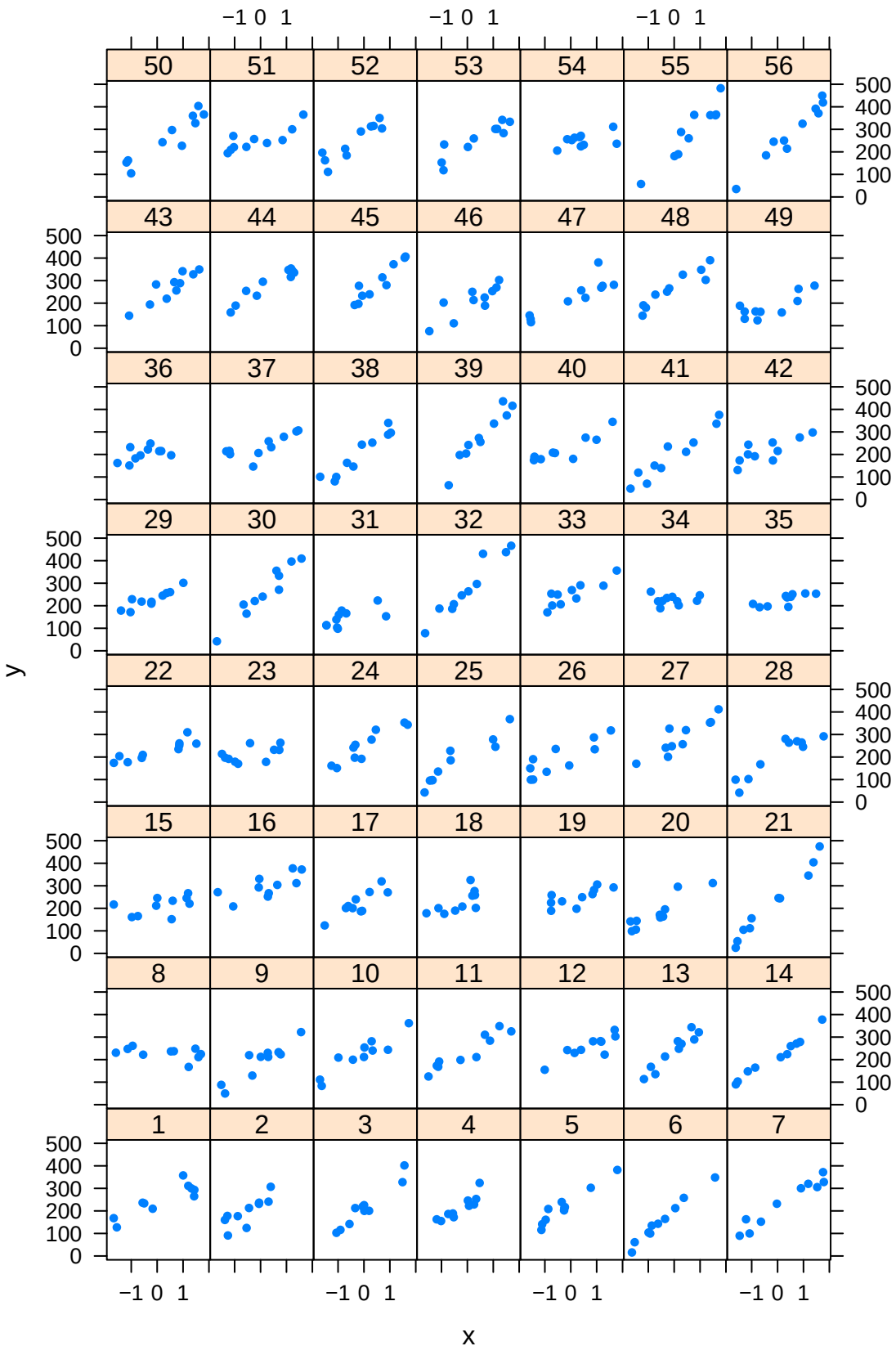
```
> y <- lin.pred + normal.error
> # or, in one go:
> y <- rnorm(n = n, mean = lin.pred, sd = sigma)
```

We take a look at the data set:

```
> hist(y, col = "lightblue", breaks = 15, freq = FALSE)
> lines(density(y), col = "blue", lwd = 2)
```




```
> library("lattice")
> xyplot(y ~ x | groups, pch = 20)
```


50 51 52 53 54 55 56
43 44 45 46 47 48 49
36 37 38 39 40 41 42
29 30 31 32 33 34 35
22 23 24 25 26 27 28
15 16 17 18 19 20 21
8 9 10 11 12 13 14
1 2 3 4 5 6 7
−1 0 1
0 100 200 300 400 500
y
x

7.4.4 REML analysis

```
> library("lme4")
> lme.fit3 <- lmer(y ~ x + (x | groups))
> print(lme.fit3, cor = FALSE)

Linear mixed model fit by REML ['lmerMod']
Formula: y ~ x + (x | groups)
REML criterion at convergence: 5633
Random effects:
 Groups   Name        Std.Dev. Corr
 groups   (Intercept) 19.6
          x           28.5     0.02
 Residual             31.0
Number of obs: 560, groups: groups, 56
Fixed Effects:
(Intercept)            x
      230.3         65.1
```

Compare with the true values:

```
> data.frame(intercept.mean = intercept.mean, slope.mean = slope.mean,
+     intercept.sd = intercept.sd, slope.sd = slope.sd, intercept.slope.correlation = intercept.slope.co
+         slope.sd), sigma = sigma)

  intercept.mean slope.mean intercept.sd slope.sd
1            230         60           20       30
  intercept.slope.correlation sigma
1                     0.01667    30
```

References

Abelson, R.P. (1995). *Statistics as Principled Argument*. L. Erlbaum Associates.

Baayen, R. Harald (2008). *Analyzing Linguistic Data: A Practical Introduction to Statistics Using R*. Cambridge University Press.

Braun, J. and D.J. Murdoch (2007). *A First Course in Statistical Programming with R*. Cambridge University Press.

De Veaux, R.D. et al. (2005). *Stats: Data and Models*. Pearson Education, Limited.

Diez, D. et al. (2013). *OpenIntro Statistics: Second Edition*. CreateSpace Independent Publishing Platform. URL: http://www.openintro.org/stat/textbook.php.

Faraway, J.J. (2004). *Linear Models With R*. Chapman & Hall Texts in Statistical Science Series. Chapman & Hall/CRC.

Gelman, A. and J. Hill (2007). *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Analytical Methods for Social Research. Cambridge University Press.

Gries, S.T. (2009). *Quantitative Corpus Linguistics with R: A Practical Introduction*. Taylor & Francis.

— (2013). *Statistics for Linguistics with R: A Practical Introduction, 2nd Edition*. Mouton De Gruyter.

Johnson, K. (2008). *Quantitative methods in linguistics*. Blackwell Pub.

Kéry, Marc (2010). *Introduction to WinBUGS for Ecologists*. Academic Press/Elsevier.

Kruschke, John K. (2011). *Doing Bayesian Data Analysis: A Tutorial with R and BUGS*. Academic Press/Elsevier.

Miles, J. and M. Shevlin (2001). *Applying Regression and Correlation: A Guide for Students and Researchers*. SAGE Publications.

Wright, D.B. and K. London (2009). *Modern regression techniques using R: A practical guide for students and researchers*. SAGE.

Xie, Yihui (2013). *Dynamic Documents with R and knitr*. Chapman and Hall/CRC.