

# Kevert modellek - Ismetelt mereses elemzesek

*Zoltan Kekecs*

*17 November 2020*

## Contents

<b>1</b>	<b>Absztrakt</b>	<b>2</b>
<b>2</b>	<b>Adatmenedzsment es leiro statisztikak</b>	<b>2</b>
2.1	Package-ek betoltese . . . . .	2
2.2	Sajat funkcio . . . . .	2
2.3	Sebgyogyulas adat betoltese . . . . .	2
2.4	Adatellenorzes . . . . .	3
<b>3</b>	<b>Ismetelt mereses eredmények vizgalata kevert linearis modellekkel</b>	<b>4</b>
3.1	Klaszteres szerkezet keresese az adatokban . . . . .	4
3.2	Az adattabla atformazasa szelesbol hosszu formatumba . . . . .	5
3.3	Kevert lineáris modell kialakítása . . . . .	6
3.4	Az eltérő modellek összehasonlítása . . . . .	6
3.5	A modell kiegészítése a napokból származó négyzetes járulékkal . . . . .	9

# 1 Absztrakt

Ez a gyakorlat az ismertetett meréses elemzésekkel foglalkozik. Amikor az elemzésünkben ugyan attól a vizsgálati személytől több adat is szerepel ugyan abból a változóból, ismertetett meréses elemzést végzünk (pl. a személy kezelése előtti és utáni depresszió szintje).

## 2 Adatmenedzsment és leíró statisztikák

### 2.1 Package-ek betöltése

A következő package-ekre lesz szükség a gyakorláthoz:

```
library(psych) # for describe
library(tidyverse) # for tidy code and ggplot\t

## Warning: package 'tidyverse' was built under R version 3.6.3

library(lme4) # for lmer() mixed models
library(lmerTest) # for significance test on lmer() mixed models
library(cAIC4) # for cAIC
library(r2glmm) # for r2beta
library(MuMIn) # for r.squaredGLMM
```

### 2.2 Saját funkció

Ezzel a funkcióval kinyerhetjük a standardizált Beta együtthatót a kevert modellekből. Ez a funkció innen lett átveve: <https://stackoverflow.com/questions/25142901/standardized-coefficients-for-lmer-model>

```
stdCoef.lmerMod <- function(object) {
  sdy <- sd(getME(object, "y"))
  sdx <- apply(getME(object, "X"), 2, sd)
  sc <- fixef(object) * sdx/sdy
  se.fixef <- coef(summary(object))[, "Std. Error"]
  se <- se.fixef * sdx/sdy
  return(data.frame(stdcoef = sc, stdse = se))
}
```

### 2.3 Sebgyógyulás adat betöltése

A gyakorlat során a sebgyógyulás adatbázissal fogunk dolgozni. Ez egy szimulált adatbázis, ami a műtet során ejtett bemetszések gyógyulását vizsgáljuk annak függvényében, hogy a betegek ágya milyen közel van az ablakhoz, és hogy mennyi napfény éri őket a felépülés időszak alatt. Ez a kutatás azt az elméletet teszteli, hogy a kórházi betegeknek szükségük van a külvilággal való kapcsolatra ahhoz, hogy gyorsan felépüljenek. Egy ablak, ami a szabadba nyílik, megteremtheti ezt a kapcsolatot a külvilággal, ezért a kutatásunk azt vizsgálja, hogy befolyásolja-e a sebgyógyulás mértékét az, hogy a személynek milyen közel van az ágya a legközelebbi ablakhoz. Az elmélet egy változata azt állítja, hogy az ablak nem csak a külvilággal való szorosabb kapcsolat megteremtésén keresztül vezet gyorsabb gyógyuláshoz, hanem azon keresztül is, hogy több napfényt enged a szobába, és az elmélet szerint a napfény is jótékony hatással van a gyógyulásra.

Változók az adatbázisban:

- ID: azonosító kód
- day\_1, day\_2, ..., day\_7: A műtet utáni 1-7. napon egy orvos megvizsgálta a beteg bemetszési sebeit, és értékelte azokat egy standardizált seb-allapot értékkel. Minél nagyobb ez az érték, annál nagyobb vagy rosszabb állapotú a seb (pl. gyulladt). Minden személynek mind a 7 naphoz külön seb-allapot érték tartozik.

- distance\_window: A személy agyához legközelebbi ablak távolsága az agytól meterben.
- location: A korhási szarny, ahol a páciens agya van. Ket állasu faktor változo: szintjei “north wing” és “south wing” (a “south wing”-ben több napfény éri a pácienseket, ez a változo azért fontos).

```
data_wound = read_csv("https://raw.githubusercontent.com/kekecsz/PSYP13_Data_analysis_class-2018/master/
# assign ID and location as factors
data_wound = data_wound %>% mutate(ID = factor(ID), location = factor(location))
```

## 2.4 Adatellenorzes

Vizsgáljuk meg az adattáblát a View(), describe(), és table() funkciók segítségével. Fontos, hogy az adattáblában jelenleg minden adat amit egy adott személytől gyűjtöttek **egy sorban található**.

```
View(data_wound)

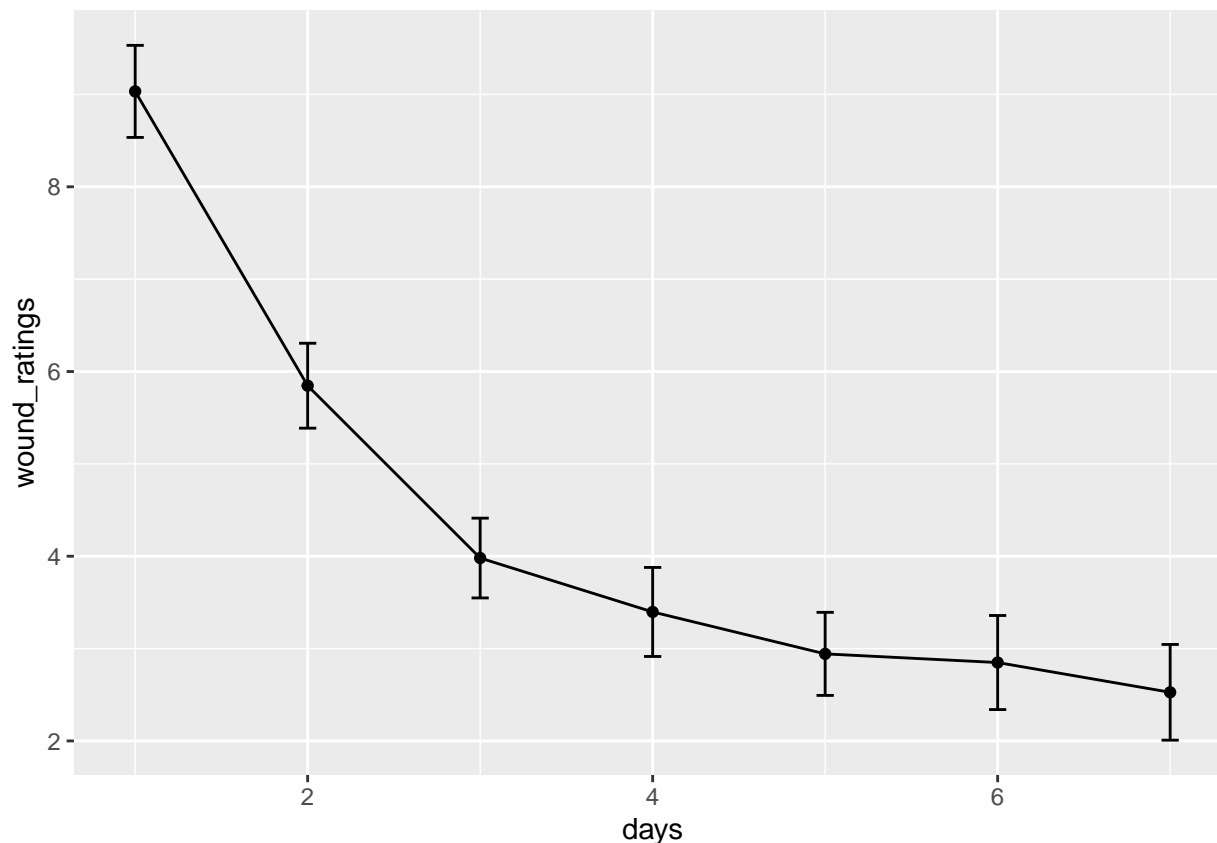
# descriptives
describe(data_wound)
table(data_wound$location)
```

Vizualizálhatjuk is az adatokat. (Az alábbi ábra elkészítéséhez először a day1-day7 változókban külön-külön kiszámoljuk az átlagokat és a standard hibát, majd a standard hibát megszorozva 1.96-al megkapjuk a konfidencia intervallumot. Végül mindezt egy új adat objektumba tesszük, és a geom\_errorbar, geom\_point, és geom\_line segítségével vizualizáljuk. Látható hogy a seb állapot értékek egyre csökken ahogyan telnek a napok.)

```
# designate which are the repeated variables
repeated_variables = c("day_1", "day_2", "day_3", "day_4", "day_5",
  "day_6", "day_7")

# explore change over time
wound_ratings = describe(data_wound[, repeated_variables])$mean
repeated_CIs = describe(data_wound[, repeated_variables])$se *
  1.96
days = as.numeric(as.factor(repeated_variables))
days = as.data.frame(days)
data_for_plot = cbind(days, wound_ratings, repeated_CIs)

data_for_plot %>% ggplot() + aes(x = days, y = wound_ratings) +
  geom_errorbar(aes(ymin = wound_ratings - repeated_CIs, ymax = wound_ratings +
    repeated_CIs), width = 0.1) + geom_point() + geom_line()
```



### 3 Ismetelt merések eredményének vizsgálata kevert lineáris modellekkel

#### 3.1 Klaszteres szerkezet keresése az adatokban

Vizsgáljuk meg a továbbiakban a sebgyógyulásra vonatkozó ismetelt merések eredményeit! Ehhez először mentsük el az adatokhoz tartozó **változónéveket egy repeated\_variables** elnevezésű objektumba, hogy később könnyen hivatkozhatunk rájuk az altalunk írt függvényekben!

A változók közötti korrelációt a `cor()` függvénnyel tudjuk megvizsgálni. Figyeljük meg, hogy az ismetelt merések adatpontjai között **eros korreláció fedezhető fel**, azaz az egyes seb-allapot értékekre vonatkozó megfigyelések **nem függetlenek egymástól**. Ez várható is, hiszen a seb-allapot érték, az eredeti bemetszés mérete és a seb gyógyulásának uteme mind függenek a vizsgált betegtől. Adataink tehát csoportokba tomorulnak (klaszterekbe), hasonlóan a korábbi példánkhoz. Azonban míg ott osztály szerinti klaszterek fordultak elő, itt most a klaszterek maguk a résztvevők.

```
# correlation of repeated variables
```

```
data_wound %>% select(repeated_variables) %>% cor()
```

```
##           day_1    day_2    day_3    day_4    day_5    day_6    day_7
## day_1  1.0000000  0.8505812  0.6565436  0.5469131  0.4647985  0.3849832  0.2708845
## day_2  0.8505812  1.0000000  0.8360082  0.7686539  0.5932054  0.4679627  0.3461029
## day_3  0.6565436  0.8360082  1.0000000  0.8618242  0.7075322  0.6016984  0.4406317
## day_4  0.5469131  0.7686539  0.8618242  1.0000000  0.8542087  0.7178904  0.6015019
## day_5  0.4647985  0.5932054  0.7075322  0.8542087  1.0000000  0.8898712  0.7978323
```

```
## day_6 0.3849832 0.4679627 0.6016984 0.7178904 0.8898712 1.0000000 0.9043339
## day_7 0.2708845 0.3461029 0.4406317 0.6015019 0.7978323 0.9043339 1.0000000
```

### 3.2 Az adattábla atformazása szélesből hosszú formátumba

A klaszteres szerkezetből kifolyólag hasonlóan kevert modellekkel elemezhetjük adatainkat mint ahogy azt a bántalmazással kapcsolatos adatsornál tettük. Ehhez azonban először **át kell rendeznünk az adatainkat**, hogy használhassuk a lineáris kevert modell (`lmer()`) függvényt.

Jelenleg a dataframe-ünk minden sora egy adott pácienshez tartozó seb-állapot értékre vonatkozó 7 megfigyelésből áll (vagyis az adatgyűjtés alatt napi egy). Ezt az elrendezést **wide format**-nak nevezik (széles formátum).

Az `lmer()` függvény megfelelő működéséhez az adattábla minden sorához csak egyetlen megfigyelés tartozhat. Jelen esetben ez azt jelentené, hogy az egyes résztvevőkhöz tartozó sorok száma 1 helyett 7 kell legyen. Így az ID, `distance_window`, és `location` változók az adott pácienshez tartozó sorokban megegyeznek majd, és csak az egyes seb-állapot értékek különböznek majd, melyekhez minden sorban mindössze egyetlen oszlop tartozna így. Ezt az elrendezést általában **long format**-nak hívjuk (hosszú formátum).

A fenti átalakítás elvégzésének egy egyszerű módja, ha a `gather()` függvényt alkalmazzuk a `tidyr` csomagból.

1. meghatározzunk egy változónevet, amiben az **ismételt megfigyelések indexét** tároljuk majd az új formátumú adatbázisban. Ez nálunk az alábbi példában “days”-nek neveztük el (**key = days**).
2. meghatározzunk egy változónevet, amiben az **ismételten megfigyelt adatok** kerülnek majd. Mivel nekünk a megfigyelt adatunk a seb állapota, ezért ezt “wound\_rating”-nek neveztük el (**value = wound\_rating**).
3. meghatározzuk, hogy a jelenleg használt széles formátumban **mely oszlopok tartalmazzák az ismételten megfigyelt adatot**. Ez a széles adatbázisunkban a `day_1`, `day_2` ... `day_7` oszlopok. Ezt a `tidyverse`-ben könnyen lerövidíthetjük, a **day\_1:day\_7** kifejezés a `day_1` és `day_7` közötti oszlopok neveit jelöli.

Az `arrange()` függvény használatával az adatok rendezhetőek a hozzájuk tartozó azonosító (“ID”) alapján. Bár az adott feladat elvégzéséhez nem szükséges rendezni az adatainkat, de mégis segít a hosszú formátum átláthatóbbá tételében.

Az eredeti adatokat változtatlanul hagyva most is **új objektumot** hozunk létre adatainknak, a már megszokott módon. Az új objektum neve `data_wound_long` lesz.

```
data_wound_long = data_wound %>% gather(key = days, value = wound_rating,
  day_1:day_7) %>% arrange(ID)
```

```
data_wound_long
```

```
## # A tibble: 210 x 5
##   ID      distance_window location  days  wound_rating
##   <fct>          <dbl> <fct>    <chr>    <dbl>
## 1 ID_01            6.18 north_wing day_1      10.3
## 2 ID_01            6.18 north_wing day_2       7.44
## 3 ID_01            6.18 north_wing day_3       5.03
## 4 ID_01            6.18 north_wing day_4       5.37
## 5 ID_01            6.18 north_wing day_5       5.37
## 6 ID_01            6.18 north_wing day_6       6.3
## 7 ID_01            6.18 north_wing day_7       6.52
## 8 ID_02            7.21 north_wing day_1       8.88
## 9 ID_02            7.21 north_wing day_2       5.24
## 10 ID_02           7.21 north_wing day_3       3.96
## # ... with 200 more rows
```

A fontos megjegyezni, hogy a 'days' változó jelenleg a széles formátumból származó változó neveket tartalmazza ('day\_1', 'day\_2' stb.). Az egyszerűbb kezelhetőség érdekében ezeket egyszerűen az egyes napokat jelölő számokra (1-7) cseréljük. Ezt legkönnyebben a `mutate()` és `recode()` függvényekkel valósíthatjuk meg.

```
# change the days variable to a numerical vector
data_wound_long = data_wound_long %>% mutate(days = recode(days,
  day_1 = 1, day_2 = 2, day_3 = 3, day_4 = 4, day_5 = 5, day_6 = 6,
  day_7 = 7))
```

Tekintsük most meg, hogyan néz ki az új dataframe-ünk!

```
View(data_wound_long)
```

### 3.3 Kevert lineáris modell kialakítása

Most, hogy megfelelő alakba hoztuk adatainkat, előállíthatjuk az előrejelzésekhez szükséges modellt. Ezzel a modellel a műtét utáni nap (days), az ablaktól való távolság ('distance\_window') és északi vagy déli elhelyezés ('location') alapján megbecsülhető lesz a seb-állapot érték ('wound\_rating').

Mivel az előrejelzésünk kimenete a résztvevők szerinti klaszteres szerkezetet mutat, ezért a **random effect prediktor** a résztvevő **azonosítója ('ID')** lesz. Az korábbi gyakorlathoz hasonlóan, most is két modellt fogunk illeszteni, a random intercept és a random slope modelleket.

Említést érdemel, hogy a **random intercept model** esetében azt feltételezzük, hogy minden résztvevő eltér a teljes vagy baseline seb-állapot értékeit tekintve, de a fix hatás prediktorok ('days', 'distance\_window', és 'location') azonosak az egyes résztvevők esetében. Ezzel szemben, a **random slope model** esetében nem csak a baseline seb-állapot érték, de a fix hatás prediktorok is résztvevőnként változóak.

Mivel 3 különböző fix hatás prediktor is rendelkezésünkre áll, ezért alkalmazhatjuk a random slope modellt, ami a prediktorok mellett a résztvevőkből származó random hatástól is függeni fog. **A random effect-et a "+ (days|ID)" alakban tehetjük a modellhez.** Ez a kifejezés azt jelenti hogy a modellünkben megengedjük hogy idő múlásának hatása más és más legyen résztvevőnként a seb gyógyulására. Másnéven **az emberek különbözhetnek abban, egy nap alatt mennyit gyógyul a sebük.**

További lehetőségként felmerül, hogyha a másik két prediktor szerinti véletlen meredekséget is szeretnénk bevezetni a modellbe, akkor azt a + (days|ID) + (distance\_window|ID) + (location|ID) kifejezéssel érhetjük el, ha azt szeretnénk hogy ne legyen köztük korreláció, és a + (days + distance\_window + location|ID) kifejezéssel, ha azt szeretnénk hogy korreláljanak. Most maradjunk egyelőre a korábban leírt, egyszerűbb + (days|ID) modellnél.

```
mod_rep_int = lmer(wound_rating ~ days + distance_window + location +
  (1 | ID), data = data_wound_long)
mod_rep_slope = lmer(wound_rating ~ days + distance_window +
  location + (days | ID), data = data_wound_long)
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, :
## Model failed to converge with max|grad| = 0.00253673 (tol = 0.002, component 1)
```

### 3.4 Az eltérő modellek összehasonlítása

Hasonlítsuk most össze a különböző modellek alapján alkotott előrejelzéseket!

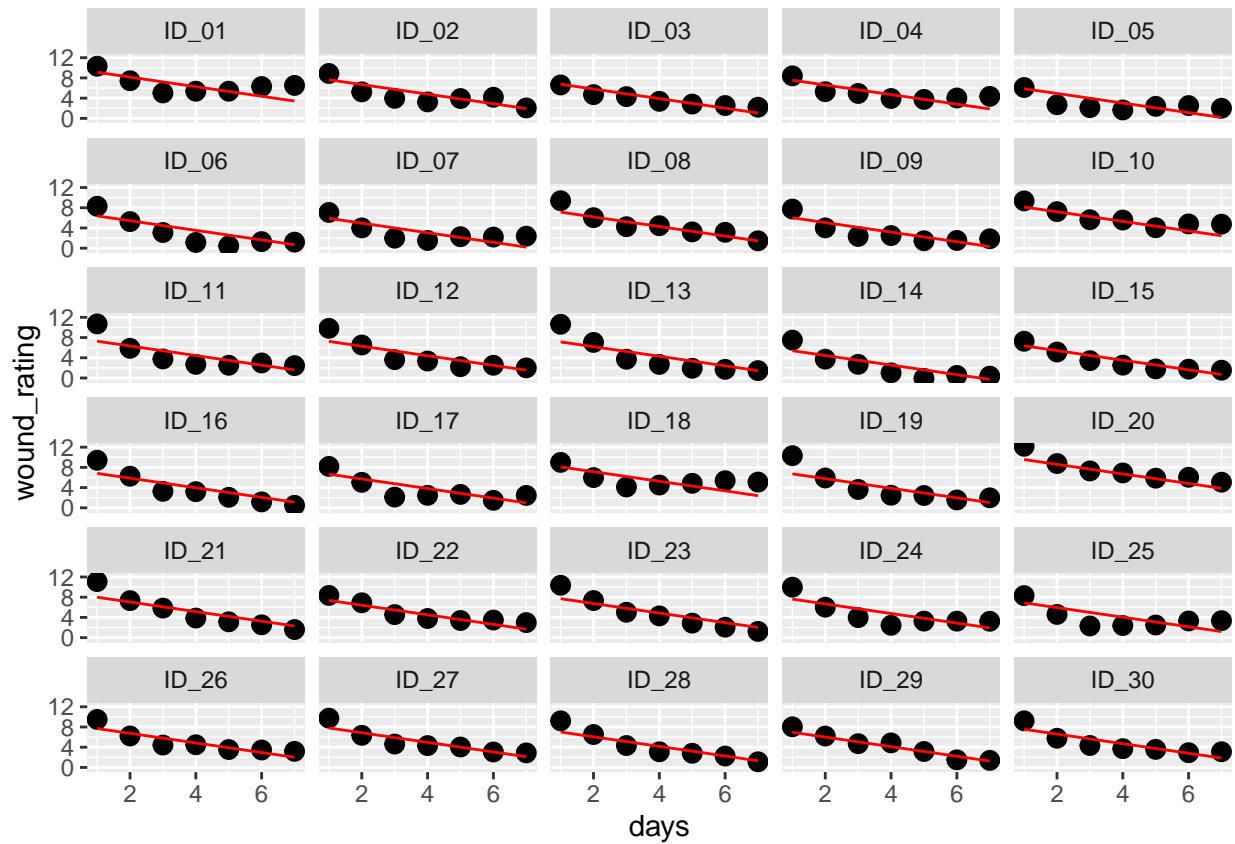
A könnyebb összehasonlíthatóság kedvéért, vizualizáljuk adatainkat! Ehhez először tároljuk el predikciónk eredményeit egy új változóban, majd ábrázolhatjuk az egyes előrejelzett értékeket a valódi értékek függvényében, az egyes (random intercept és random slope) modellekre vonatkozó külön-külön ábrákon.

(Az alábbiakban létrehoztunk egy másolatot az adatokat tartalmazó objektumról, hogy az eredeti adatok változatlanul megmaradhassanak.)

```
data_wound_long_withpreds = data_wound_long
data_wound_long_withpreds$pred_int = predict(mod_rep_int)
data_wound_long_withpreds$pred_slope = predict(mod_rep_slope)
```

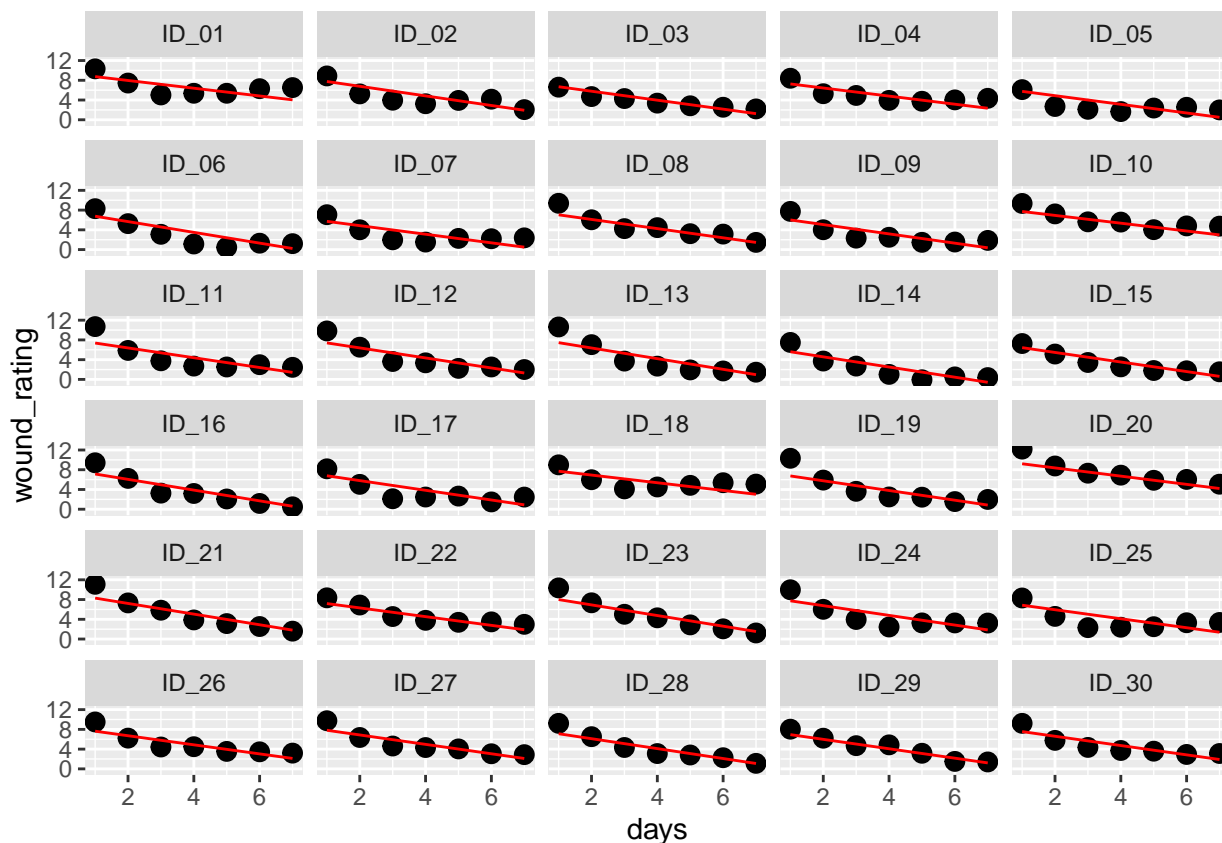
```
# random intercept model
```

```
ggplot(data_wound_long_withpreds, aes(y = wound_rating, x = days,
  group = ID)) + geom_point(size = 3) + geom_line(color = "red",
  aes(y = pred_int, x = days)) + facet_wrap(~ID, ncol = 5)
```



```
# random slope and intercept model
```

```
ggplot(data_wound_long_withpreds, aes(y = wound_rating, x = days,
  group = ID)) + geom_point(size = 3) + geom_line(color = "red",
  aes(y = pred_slope, x = days)) + facet_wrap(~ID, ncol = 5)
```



Látható, hogy az eltérő modellek alapján kapott eredmények között nincs számottevő eltérés.

A `cAIC()` és `anova()` függvények segítségével további megállapításokat tehetünk az egyes modellek illeszkedéséről, ami egy újabb lehetséges szempont lehet a modellek összehasonlításánál.

```
cAIC(mod_rep_int)$caic
```

```
## [1] 757.0182
```

```
cAIC(mod_rep_slope)$caic
```

```
## [1] 757.3564
```

```
anova(mod_rep_int, mod_rep_slope)
```

```
## refitting model(s) with ML (instead of REML)
```

```
## Data: data_wound_long
```

```
## Models:
```

```
## mod_rep_int: wound_rating ~ days + distance_window + location + (1 | ID)
```

```
## mod_rep_slope: wound_rating ~ days + distance_window + location + (days | ID)
```

```
##           Df    AIC    BIC  logLik deviance Chisq Chi Df Pr(>Chisq)
```

```
## mod_rep_int     6 773.40 793.49 -380.70   761.40
```

```
## mod_rep_slope   8 774.82 801.60 -379.41   758.82 2.583     2    0.2749
```

A fenti módszerek egyikével se találunk jelentős eltérést a két modell használata között, így a jelenlegi minta esetén semmiféle előnnyel sem jár a random slope módszer. Ez persze magában még nem elegendő ahhoz, hogy feltételezhessük, hogy más mintánál is hasonló lenne a helyzet. Látható tehát, hogy az adatelemzés során fontos tisztában lennünk a korábbi kutatások eredményeivel, és a vizsgált kérdéskörre vonatkozó elméletekkel.



Jelenleg,-híjján bármiféle korábbi ismeretnek,- folytassuk a random intercept modell használatával.

### 3.5 A modell kiegészítése a napokból származó négyzetes járulékkal

Az egyes ábrákat vizsgálva megfigyelhetjük, hogy a napok és a seb-állapot értékek közötti összefüggés nem lineáris. A sebek látszólag gyorsabban gyógyulnak az első néhány napban, mint később.

A nem lineáris viselkedés figyelembevétele érdekében, adjuk hozzá a napokból származó négyzetes járulékot a modellünkhöz!

```
mod_rep_int_quad = lmer(wound_rating ~ days + I(days^2) + distance_window +  
  location + (1 | ID), data = data_wound_long)
```

Mentsük előrejelzéseinket egy új dataframe-be, amely a korábbi előrejelzéseket is tartalmazza!

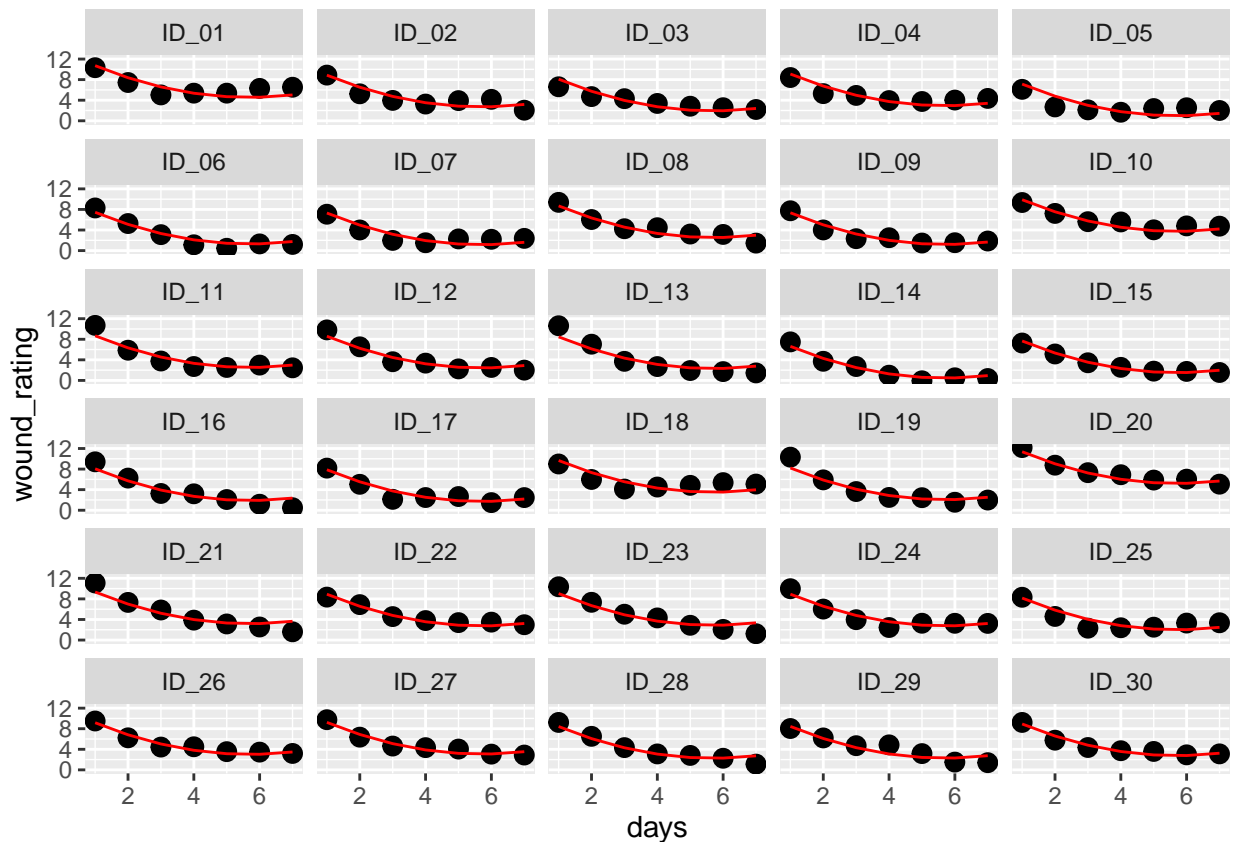
```
data_wound_long_withpreds$pred_int_quad = predict(mod_rep_int_quad)
```

Most pedig hasonlítsuk össze a négyzetes tagokkal bővített, és az eredeti modellt a modellek összehasonlításánál korábban tárgyalt módon!

```
data_wound_long_withpreds$pred_int_quad = predict(mod_rep_int_quad)
```

```
plot_quad = ggplot(data_wound_long_withpreds, aes(y = wound_rating,  
  x = days, group = ID)) + geom_point(size = 3) + geom_line(color = "red",  
  aes(y = pred_int_quad, x = days)) + facet_wrap(~ID, ncol = 5)
```

plot\_quad



```

cAIC(mod_rep_int)$caic

## [1] 757.0182

cAIC(mod_rep_int_quad)$caic

## [1] 583.2994

anova(mod_rep_int, mod_rep_int_quad)

## refitting model(s) with ML (instead of REML)
## Data: data_wound_long
## Models:
## mod_rep_int: wound_rating ~ days + distance_window + location + (1 | ID)
## mod_rep_int_quad: wound_rating ~ days + I(days^2) + distance_window + location +
## mod_rep_int_quad: (1 | ID)
##           Df    AIC    BIC  logLik deviance  Chisq Chi Df Pr(>Chisq)
## mod_rep_int      6 773.40 793.49 -380.70   761.40
## mod_rep_int_quad  7 621.08 644.51 -303.54   607.08 154.32      1 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

Az összehasonlítás alapján úgy tűnik, hogy a négyzetes tagokat is megengedő modell előrejelzései lényegesen pontosabbak mint a csak lineáris tagokat használóé.

Mivel modellünk látszólag jól illeszkedik az adatokra, nem bővítjük tovább tagokkal azt.

A négyzetes elemek felhasználásából következően várható, hogy problémák fognak jelentkezni a collinearitás tekintetében. A 'days' változó centrálásával ez a probléma a model diagnosztika c. gyakorlatban tárgyalt módon kiküszöbölhető, hiszen megszünteti a 'days' és 'days^2' közötti korrelációt.

Végezzük el a centrálást, és illesszük újra modellünket az így kapott prediktorokat használva.

```

data_wound_long = data_wound_long %>% mutate(days_centered = days -
  mean(days))

mod_rep_int_quad = lmer(wound_rating ~ days_centered + I(days_centered^2) +
  distance_window + location + (1 | ID), data = data_wound_long)

```

Az előző gyakorlathoz hasonlóan kérjük eredményeink bemutatását!

```

# Marginal R squared
r2beta(mod_rep_int_quad, method = "nsj", data = data_wound_long)

##           Effect    Rsq upper.CL lower.CL
## 1           Model 0.763    0.805    0.717
## 2    days_centered 0.700    0.752    0.643
## 3 I(days_centered^2) 0.377    0.470    0.284
## 4 distance_window 0.130    0.220    0.058
## 5 locationsouth_wing 0.103    0.189    0.039

# marginal and conditional R squared values
r.squaredGLMM(mod_rep_int_quad)

## Warning: 'r.squaredGLMM' now calculates a revised statistic. See the help page.

##           R2m           R2c
## [1,] 0.7626648 0.8756216

```

```

# Conditional AIC
cAIC(mod_rep_int_quad)$caic

## [1] 583.2994

# Model coefficients
summary(mod_rep_int_quad)

## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: wound_rating ~ days_centered + I(days_centered^2) + distance_window +
## location + (1 | ID)
## Data: data_wound_long
##
## REML criterion at convergence: 625.7
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -2.34290 -0.61819  0.02384  0.61744  2.40352
##
## Random effects:
##  Groups   Name                Variance Std.Dev.
##  ID       (Intercept)  0.7380    0.8591
## Residual                   0.8126    0.9015
## Number of obs: 210, groups:  ID, 30
##
## Fixed effects:
##              Estimate Std. Error      df t value Pr(>|t|)
## (Intercept)    4.95676    0.50887  28.10724   9.741 1.65e-10 ***
## days_centered  -0.94826    0.03110 178.00000  -30.487 < 2e-16 ***
## I(days_centered^2) 0.27908    0.01796 178.00000   15.541 < 2e-16 ***
## distance_window -0.09016    0.03174  27.00000   -2.840  0.00846 **
## locationsouth_wing -0.84297    0.33787  27.00000   -2.495  0.01901 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##              (Intr) dys_cn I(_^2) dstnc_
## days_centrd  0.000
## I(dys_cn^2) -0.141  0.000
## distnc_wndw -0.872  0.000  0.000
## lctnsth_wng -0.288  0.000  0.000 -0.049

# Confidence intervals for the coefficients
confint(mod_rep_int_quad)

## Computing profile confidence intervals ...

##              2.5 %      97.5 %
## .sig01          0.6048206  1.10514198
## .sigma          0.8112449  0.99766032
## (Intercept)     3.9797744  5.93373628
## days_centered   -1.0092087 -0.88731515
## I(days_centered^2) 0.2438917  0.31426699
## distance_window  -0.1511234 -0.02919934
## locationsouth_wing -1.4918557 -0.19408605

```

```
# standardized Betas
stdCoef.merMod(mod_rep_int_quad)
```

```
##                stdcoef      stdse
## (Intercept)      0.0000000 0.0000000
## days_centered   -0.7486918 0.02455740
## I(days_centered^2) 0.3816481 0.02455740
## distance_window  -0.1894277 0.06669033
## locationsouth_wing -0.1663901 0.06669033
```

Mielőtt elfogadnánk eredményeinket véglegesnek, mindig futassunk modell diagnosztikát is. Ennek módjára a következő gyakorlatban fogunk kitérni.

### Gyakorlás (opcionális)

Olvassuk be a műtéti fájdalom adatsort!

Ez az adatsor a műtét utáni fájdalom mértékéről, és az ezzel feltételezhetően összefüggő néhány egyéb értékekről tartalmaz információkat.

Változóink:

- ID: résztvevő azonosítója
- pain1, pain2, pain3, pain4: A használt adatsorban a fájdalom a műtét utáni négy egymást követő napon volt mérve egy 0-tól-10-ig terjedő folytonos vizuális skálán.
- sex: a résztvevő bejelentett neme
- STAI\_trait: A résztvevő State Trait Anxiety Inventory-n elért pontszáma
- pain\_cat: fájdalom katasztrofizálása
- cortisol\_serum; cortisol\_saliva: A kortizol egy a stress hatására előállított hormon. A kortizol szintet véréből és nyálból, közvetlenül a műtét után határozták meg.
- mindfulness: A Mindfulness kérdőív alapján a résztvevőre jellemző Mindfulness érték
- weight: résztvevő tömege kg-ban.
- IQ: Résztvevő IQ-ja a műtét előtt egy héttel felvett IQ teszt alapján
- household\_income: résztvevő háztartásának bevétele USD-ben

Gyakorló feladatok:

1. Olvassuk be az adatokat (egy .csv kiterjesztésű file-ból). Az adatokat az alábbi linkről tölthetjük le: “<https://tinyurl.com/data-pain1>”.
2. Alakítsuk adatainkat hosszú formátumúvá (célszerű a gather() vagy a melt() függvények valamelyikét használni erre a célra), hogy az egyes megfigyelések külön sorba kerüljenek.
3. Állítsunk össze egy kevert lineáris modellt, hogy amivel képesek vagyunk a műtét utáni fájdalom varianciájának lehető legszélesebb körű lefedésére. (A műtét utáni fájdalom meghatározásához tetszőleges fix prediktort választhatunk, amennyiben annak feltehetően van valami köze a fájdalom mértékéhez.) Mivel adataink a résztvevők szerinti klaszteres szerkezetet mutatnak, modellünkben vegyük figyelembe a résztvevők azonosítója szerinti véletlen hatást.
4. Kísérletezzünk mind a random intercept, mind pedig a random slope modellekkel, majd hasonlítsuk össze őket a cAIC() függvény felhasználásával.
5. Alkossunk olyan random intercept és random slope modelleket, ahol az egyetlen prediktor az idő (műtét óta eltelt napok száma). Vizualizáljuk a modelljeink alapján kapott regressziós vonalakat, minden résztvevőre külön-külön, és hasonlítsuk össze hogyan illeszkednek a megfigyeléseinkre. Van bármi előnye ha az időt külön változó hatásként vizsgáljuk a random slope modellben?
6. Hasonlítsuk össze az 5. pont modelljeit a cAIC() függvény eredményei alapján is!
7. Mi a határ  $R^2$  érték a random intercept modell esetében? Pontosabb-e a konfidencia intervallum alapján a fájdalom előrejelzésében ez a modell, mint a null modell?