Introducción a los modelos de efectos mixtos

Santiago Gualchi

Miércoles 12 de junio de 2019

Breve repaso

En las últimas reuniones vimos que las **regresiones lineales** nos sirven para modelar el comportamiento de nuestras variables dependientes numéricas como una función polinómica de grado 1 (o sea, lineal). Podemos representar esto formalmente mediante la ecuación:

$$y_i = a + b_1 x_{1i} + b_2 x_{2i} + \cdots + b_k x_{ki} + \varepsilon_i$$

donde x_{1i}, x_{2i} ... x_{ki} son los valores que toman nuestros predictores para la observación i; y_i es el valor de la variable dependiente para la observación i; a es la intersección de la función (el valor de base para cualquier observación); b_1, b_2 ... b_k son los coeficientes de cada predictor; y ε_i es el error o la corrección que se aplica al modelo para que se adecúe a lo efectivamente observado:

Variable	Significado
x_{ki}	Valor del predictor x_k para la observación i.
y_i	Valor que toma la variable dependiente para la observación i.
a	Intersección.
b_k	Coeficiente del predictor n.
ε_i	Diferencia entre $\hat{y}_i$ y y_i

También, vimos que podemos modelar nuestras variables dependientes categóricas mediante una **regresión logística**:

$$\text{logit}(p_i) = a + b_1 x_{1i} + b_2 x_{2i} + \cdots + b_k x_{ki} + \varepsilon_i$$

donde $\text{logit}(p_i)$ es el logaritmo natural de las **chances del evento** p_i. Existe una diferencia, entonces, entre los resultados de una regresión lineal y de una regresión logística, y es que el resultado de una regresión logística no es el valor que toma la variable dependiente (y), sino el logaritmo natural de las chances que tiene la variable dependiente de tomar cierto valor dados ciertos valores de los predictores ($\text{logit}(p_i)$).

Efectos fijos y efectos aleatorios

Cuando intentamos modelar el comportamiento de una variable a través de una regresión lineal o una regresión logística, vamos a tomar predictores que sean **repetibles**. Para que un factor sea repetible, el conjunto de los niveles de ese factor debe ser fijo, de ahí que a estos predictores se los llame **efectos fijos**[1]. Por ejemplo, si queremos evaluar el tiempo de reacción en una tarea de decisión léxica en función de la frecuencia del estímulo, podemos

[1] Por supuesto que los conjuntos de niveles posibles van a depender de cómo son operacionalizadas las variables.

repetir una misma frecuencia para distintos estímulos experimentales. O, si queremos predecir el orden de verbo-objeto en función del orden de adposición-nombre, podemos considerar grupos de observaciones (lenguas) que repitan el nivel preposición-nombre y otras que repitan el nivel nombre-posposición.

No obstante, a la hora de tomar un experimento o de recoger datos, muy probablemente acabemos observando casos para los que la variable dependiente no puede predecirse en su totalidad a partir de los predictores considerados. Las regresiones lineales y logísticas consiguen recoger esta variación a partir de la inclusión del error ε_i. Ahora bien, las posibles fuentes de ruido pueden incluir variables moderadoras, *confounders*, predictores que no se tuvieron en cuenta, pero también existen otras fuentes de ruido que vienen de una selección aleatoria que se realiza sobre la población de interés. Con esto me refiero a, por ejemplo, los sujetos que participan de la tarea y los ítems léxicos empleados en el caso dele experimento de decisión léxica, y a las lenguas seleccionadas en el ejemplo sobre orden de constituyentes.

Estos factores que suman ruido se conocen como **efectos aleatorios** porque esperamos tener un poco de variación aleatoria para cada uno de ellos (sujetos, ítems, lenguas, o el que fuera). Esta variación puede deberse a distintos factores, pero lo cierto es que no podemos controlar todas las posibles variables en nuestro experimento. Con los efectos aleatorios el conjunto de los posibles niveles no es fijo (o por lo menos no claramente fijo). Si para un experimento elegimos 60 sujetos, no hay ninguna razón a priori que nos impida elegir otros 60 sujetos (siempre que podamos repetir los niveles para nuestros efectos fijos), y lo mismo para el caso de los ítems y las lenguas. De hecho, a la hora de replicar un experimento el interés está puesto en repetir los efectos fijos, pero no hay ningún interés en repetir los efectos aleatorios (hacerlo podría ser incluso contraproducente).

Latencias en una tarea de decisión léxica para neologismos del neerlandés

De Vaan *et al.* (2007) realizaron un experimento en el que presentaban neologismos terminados en *-heid* '-idad' a un grupo de 26 sujetos en dos condiciones. En la primera condición, se les presentaba a los sujetos la base (eg. *lobbig* 'esponjoso') y 40 estímulos después el neologismo (eg. *lobbigheid* 'esponjosidad'). En la segunda condición se les presentaba el neologismo y 40 estímulos después se repetía. Para cada estímulo, un sujeto era expuesto a una sola de las condiciones. Se esperaba que los tiempos de latencia fueran menores cuando el sujeto era expuesto a la segunda condición.

Para acceder a estos datos vamos a necesitar importar el paquete `languageR` (Baayen, 2007). Además, vamos a cargar `ggplot2`, `lme4`, que es el paquete que nos va a permitir ajustar modelos mixtos, y `lmerTest`, que nos va a permitir calcular los valores p de los coeficientes:

```
library(languageR)
library(ggplot2)
library(lme4)
library(lmerTest)
```

El dataset de la investigación llevada a cabo por De Vaan *et al.* (2007) está disponible bajo el nombre de `primingHeid`.

```r
# Sacamos los outliers para simplificar el análisis.
primingHeid <- primingHeid[primingHeid$RT < 7.1, ]

# Seleccionamos las columnas de interés.
primingHeid <-
  primingHeid[, c("Subject", "Word", "RT", "Condition", "ResponseToPrime",
                  "RTtoPrime")]
```

El contenido de estas columnas queda recogido en la siguiente tabla:

Variable	Significado
Subject	Factor con los niveles de sujeto.
Word	Factor con los niveles de palabra.
RT	Vector numérico con el logaritmo de las latencias para la decisión léxica.
Condition	Factor con los niveles para el tratamiento del priming. Si es `baseheid` el prime es la base, si es `heid` el prime es el neologismo.
ResponseToPrime	Factor con niveles `correct` e `incorrect` para la respuesta al prime.
RTtoPrime	Vector numérico con el logaritmo de las latencias para la decisión léxica.

A continuación exploramos un poco el data set.

```r
str(primingHeid)
```

```
## 'data.frame':    787 obs. of  6 variables:
##  $ Subject        : Factor w/ 26 levels "pp1","pp10","pp11",..: 1 1 1 1 1 1 1 1 1
##  $ Word           : Factor w/ 40 levels "aftandsheid",..: 4 24 28 10 33 38 25 16 3
##  $ RT             : num  6.69 6.81 6.51 6.58 6.86 ...
##  $ Condition      : Factor w/ 2 levels "baseheid","heid": 2 1 1 2 2 1 1 1 2 2 ...
##  $ ResponseToPrime: Factor w/ 2 levels "correct","incorrect": 1 1 1 1 1 2 1 1 1 1
##  $ RTtoPrime      : num  6.88 6.56 6.65 6.96 6.83 ...
```

```r
head(primingHeid)
```

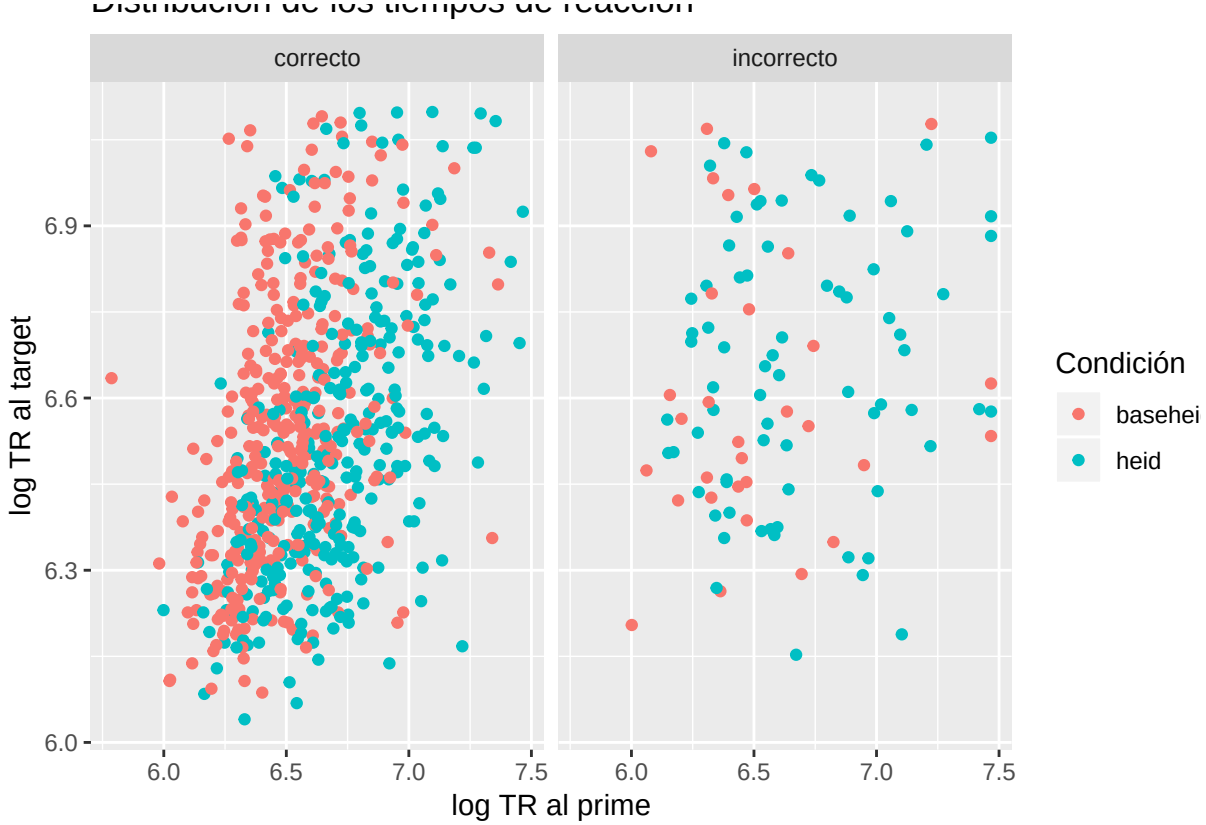
```
##   Subject         Word       RT Condition ResponseToPrime RTtoPrime
## 1     pp1   basaalheid 6.693324      heid         correct  6.884487
## 2     pp1  markantheid 6.809039  baseheid         correct  6.558198
## 3     pp1 ontroerdheid 6.508769  baseheid         correct  6.651572
## 4     pp1  contentheid 6.576470      heid         correct  6.960348
## 5     pp1    riantheid 6.857514      heid         correct  6.826545
## 6     pp1  tembaarheid 6.349139  baseheid       incorrect  6.824374
```

```r
summary(primingHeid)
```

```
##     Subject              Word           RT           Condition
##  pp16   : 40   royaalheid  : 25   Min.   :6.040   baseheid:397
##  pp22   : 40   geurloosheid: 24   1st Qu.:6.356   heid    :390
```

```
##  pp28    : 40   ontroerdheid: 24   Median :6.516
##  pp14    : 39   riantheid   : 24   Mean   :6.543
##  pp19    : 39   tilbaarheid : 24   3rd Qu.:6.707
##  pp2     : 39   beschutheid : 23   Max.   :7.098
##  (Other):550   (Other)     :643
##   ResponseToPrime   RTtoPrime
##  correct  :683    Min.   :5.787
##  incorrect:104    1st Qu.:6.385
##                   Median :6.560
##                   Mean   :6.594
##                   3rd Qu.:6.753
##                   Max.   :7.467
##
```

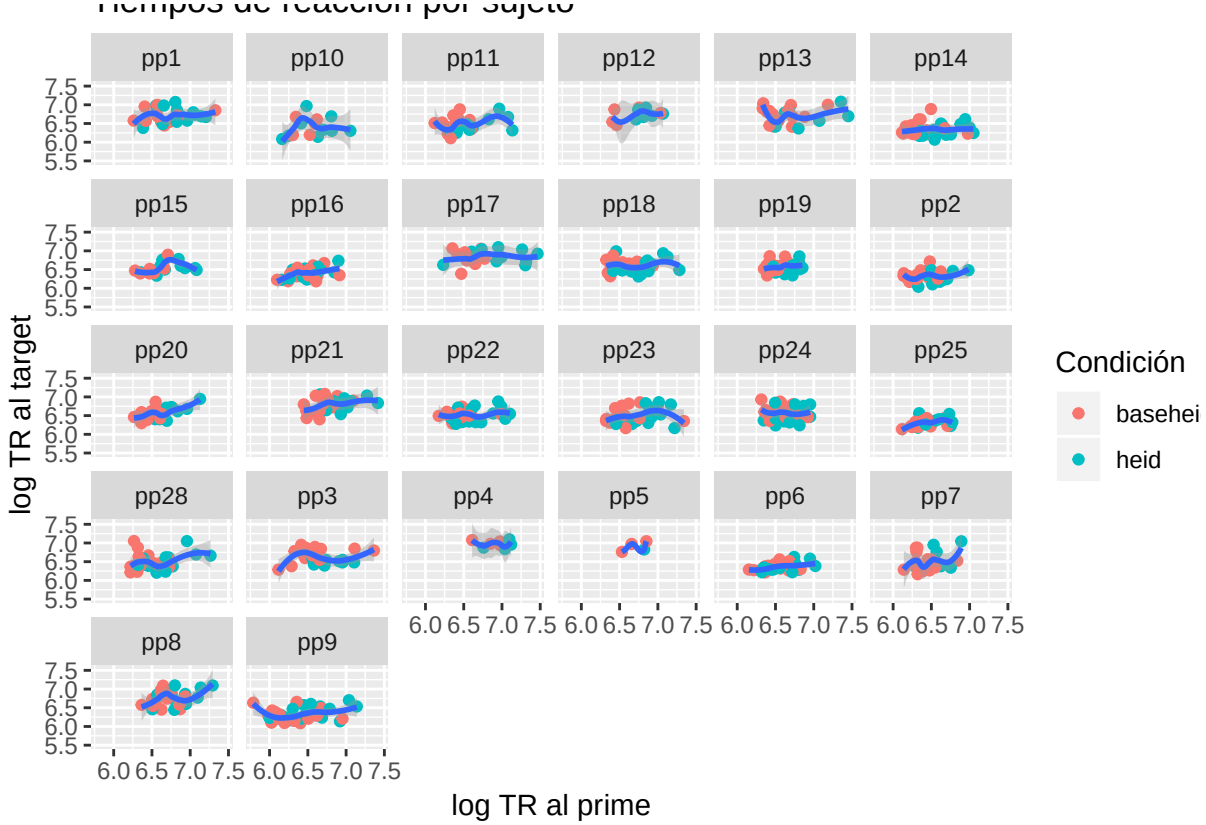
```r
ggplot(primingHeid, aes(x = RTtoPrime, y = RT)) +
  geom_point(aes(color = Condition)) +
  facet_grid(. ~ ResponseToPrime) +
  labs(
    title = "Distribución de los tiempos de reacción",
    x = "log TR al prime",
    y = "log TR al target",
    color = "Condición"
  )
```



En esta figura sobre el eje x está representado el tiempo de reacción al prime en escala logarítmica y sobre el eje y el tiempo de reacción al target en escala logarítmica. Los puntos rosados representan casos en los que el prime era la base, y los puntos celestes

representan los casos en los que el prime era el neologismo. Sobre el gráfico de la izquierda están ploteados los casos en los que se respondió correctamente si el prime se trataba o no de una palabra, y sobre el gráfico de la derecha están ploteados los casos en los que se contestó de forma incorrecta.

A simple vista, se observa que hubo muchas más respuestas correctas que incorrectas. También se decidió más rápidamente cuando el prime era una base, que cuando era un neologismo, y que en estos últimos casos hubo más errores. No es tan claro, cuál fue el comportamiento en relación al tiempo de reacción en la decisión del target.



En esta figura se observa que el rendimiento entre los sujetos fue más bien dispar. Por ejemplo, el sujeto pp6 y el sujeto pp14, tuvieron rendimientos más bien constanes en cuanto al tiempo de reacción del target. Otros sujetos, como pp11 y pp24 tardaron, más en identficar el target cuando tardaron más en identificar el prime.

Un primer modelo

Si intentamos predecir el TR del target en función de la condición del prime y de la interacción entre el TR del prime y la respuesta del prime. Podríamos usar una regresión lineal:

$$TR_i = a + b_1 \cdot condicion_i + b_2 \cdot TR_{prime,i} + b_3 \cdot resp_{prime,i} + b_4 \cdot TR_{prime,i} \cdot resp_{prime,i} + \varepsilon_i$$

```
ph_lm <- lm(RT ~ Condition + RTtoPrime * ResponseToPrime, data = primingHeid)
summary(ph_lm)
```

```
##
```

```
## Call:
## lm(formula = RT ~ Condition + RTtoPrime * ResponseToPrime, data = primingHeid)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.5970 -0.1500 -0.0258  0.1303  0.6280
##
## Coefficients:
##                                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)                         3.59591    0.21405  16.800  < 2e-16
## Conditionheid                      -0.08945    0.01656  -5.403 8.72e-08
## RTtoPrime                           0.45137    0.03289  13.725  < 2e-16
## ResponseToPrimeincorrect            2.37079    0.42734   5.548 3.96e-08
## RTtoPrime:ResponseToPrimeincorrect -0.34015    0.06434  -5.286 1.62e-07
##
## (Intercept)                        ***
## Conditionheid                      ***
## RTtoPrime                          ***
## ResponseToPrimeincorrect           ***
## RTtoPrime:ResponseToPrimeincorrect ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2119 on 782 degrees of freedom
## Multiple R-squared:  0.2172, Adjusted R-squared:  0.2132
## F-statistic: 54.26 on 4 and 782 DF,  p-value: < 2.2e-16
```

Este modelo tiene coeficientes altamente significativos para todas las variables de interés. Sin embargo, no da cuenta de los efectos aleatorios de las variables de sujeto y de palabra. Y, al no dar cuenta de estos efectos, tampoco respeta el supuesto de independencia.

¿Podemos modelar los efectos aleatorios como efectos fijos?

Siguiendo nuestro ejemplo, digamos que queremos modelar el tiempo de reacción del target en función de la condición del prime. Tendríamos un modelo como el que sigue:

$$TR_i = a + b \cdot condicion_i + \varepsilon_i$$

Ahora supongamos que tenemos 5 sujetos y queremos tenerlos en cuenta como efectos fijos que influyen en el tiempo de reacción (al fin y al cabo quién es el sujeto no deja de ser una variable). Usando variables *dummy* tendríamos que agregar 4 predictores a nuestro modelo[2]:

$$TR_i = a + b_1 condicion_i + b_2 s_{1i} + b_3 s_{2i} + b_4 s_{3i} + b_5 s_{4i} + \varepsilon_i.$$

¡¿Qué hacemos entonces con los 26 sujetos de los datos del neerlandés?! ¡Necesitamos 25 predictores extra! Y 26 sujetos no son tantos. Además, tenemos otros problemas:

[2]Para incluir predictores categóricos en un modelo lineal se codifican por contraste (*dummy encoding*). Al hacer esto se generan tantas variables como el número de niveles − 1.

- Este modelo no tiene en cuenta que la condición del prime puede influir más o menos en los tiempos de reacción de los distintos sujetos, o sea, no tiene en cuenta que la pendiente puede ser distinta según el sujeto. Para controlar eso tendríamos que incluir una interacción entre cada predictor y cada sujeto!
- Este modelo no tiene en cuenta el efecto aleatorio de palabra. Para hacerlo tendríamos que incluir un predictor por cada ítem ($-$ 1). Pero también deberíamos incluir las interacciones entre ítem y condición.
- Este modelo no permite predecir el comportamiento de nuevos sujetos.

¿Podemos modelar los efectos aleatorios mediante ANOVAs?

Correr un ANOVA nos permite incluir un **término de error** como podría ser sujeto o ítem. No obstante, **solo** nos permite incluir **un** término de error. Si queremos tener en cuenta el efecto aleatorio de sujeto y el efecto aleatorio de ítem tenemos que correr dos ANOVAs. El problema de correr dos análisis es que vamos a necesitar resultados significativos para apoyar nuestra hipótesis en los dos ANOVAs. Dicho de otra forma, no es un método tan poderoso. O, al menos, podríamos encontrar otros que lo sean más. Además, los ANOVAs tienen otros problemas, como que no nos permiten modelar variables dependientes nominales.

Modelos de efectos mixtos

La solución al problema es usar **modelos de efectos mixtos** (o modelos mixtos). El nombre viene justamente de que combinan efectos fijos y efectos aleatorios. ¿Cómo se consigue esto? Lo que vamos a hacer es tomar nuestra función para la regresión lineal[3], y vamos a agregarle un término de grado 0 que va a ser específico, por ejemplo, a cada sujeto:

$$TR_i = a + a_s + b \cdot condicion_i + \varepsilon_{is}$$

El término a_s representa la diferencia entre la intersección general y la intersección de un sujeto específico. Pensándolo en términos de nuestro ejemplo, este término estaría explicando diferencias en el tiempo de reacción de los sujetos que no están relacionadas con la variable lexicalidad.

Para definir este modelo en R, usamos la función `lmer()` del paquete `lme4`. La sintaxis es muy parecida a la de `lm()` salvo que para agregar la intersección aleatoria de sujeto tenemos que agregar a la fórmula `+ (1 | Subject)` donde se indica que `Subject` es un efecto aleatorio sobre `1` (la intersección).

```
ph_lmer1 <- lmer(RT ~ Condition + (1 | Subject), data = primingHeid)
summary(ph_lmer1)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: RT ~ Condition + (1 | Subject)
##    Data: primingHeid
##
## REML criterion at convergence: -334
```

[3]Funciona en forma análoga para las regresiones logísticas, remplazando la variable independiente por $\text{logit}(p_i)$. En estos casos, el test se llama **modelo de efectos mixtos generalizado**.

```
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.3365 -0.7180 -0.1059  0.6746  3.0872
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  Subject  (Intercept) 0.02905  0.1704
##  Residual             0.03402  0.1844
## Number of obs: 787, groups:  Subject, 26
##
## Fixed effects:
##                Estimate Std. Error        df t value Pr(>|t|)
## (Intercept)     6.57515    0.03485  25.89043 188.686   <2e-16 ***
## Conditionheid   0.01314    0.01320 759.61537   0.995     0.32
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr)
## Conditionhd -0.189
```

A su vez, podemos agregar fácilmente una intersección aleatoria de palabra.

```
ph_lmer2 <- lmer(RT ~ Condition + (1|Subject) + (1|Word), data = primingHeid)
summary(ph_lmer2)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: RT ~ Condition + (1 | Subject) + (1 | Word)
##    Data: primingHeid
##
## REML criterion at convergence: -361.1
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.3946 -0.7139 -0.1280  0.6578  3.0913
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  Word     (Intercept) 0.002923 0.05407
##  Subject  (Intercept) 0.029302 0.17118
##  Residual             0.031126 0.17643
## Number of obs: 787, groups:  Word, 40; Subject, 26
##
## Fixed effects:
##                Estimate Std. Error        df t value Pr(>|t|)
## (Intercept)   6.580e+00  3.593e-02 2.888e+01 183.147   <2e-16 ***
## Conditionheid 9.115e-03  1.269e-02 7.289e+02   0.718    0.473
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr)
## Conditionhd -0.177
```

De esta forma estamos agregando coeficientes para dar cuenta de los efectos aleatorios de sujeto y de palabra. No obstante, si se fijan en los resúmenes los modelos mixtos que entrenamos, no hay ningún valor estimado para estos coeficientes. Para encontrar esos valores usamos la función `ranef()`.

```
ph_ranef <- ranef(ph_lmer2)
head(ph_ranef$Subject)
```

```
##      (Intercept)
## pp1   0.09252332
## pp10 -0.10442693
## pp11 -0.07034247
## pp12  0.17520742
## pp13  0.14830282
## pp14 -0.21850291
```

```
head(ph_ranef$Word)
```

```
##             (Intercept)
## aftandsheid  0.06886867
## antiekheid  -0.01065041
## banaalheid   0.02708795
## basaalheid   0.01086944
## bebrildheid  0.05337035
## beschutheid -0.03587828
```

También, la función `coef()` nos permite explorar los coeficientes estimados por sujeto y por palabra teniendo en cuenta los efectos fijos y los efectos aleatorios.

```
ph_coef <- coef(ph_lmer2)
head(ph_coef$Subject)
```

```
##      (Intercept) Conditionheid
## pp1     6.672902   0.009114762
## pp10    6.475952   0.009114762
## pp11    6.510037   0.009114762
## pp12    6.755586   0.009114762
## pp13    6.728682   0.009114762
## pp14    6.361876   0.009114762
```

```
head(ph_coef$Word)
```

```
##             (Intercept) Conditionheid
## aftandsheid    6.649248   0.009114762
## antiekheid     6.569729   0.009114762
```

```
## banaalheid     6.607467   0.009114762
## basaalheid     6.591248   0.009114762
## bebrildheid    6.633749   0.009114762
## beschutheid    6.544501   0.009114762
```

Algo a tener en cuenta es que el total de valores para un efecto aleatorio tiene como media el coeficiente sobre el que se aplica (en este caso, la intersección).

```
mean(ph_coef$Subject[, 1])
```

```
## [1] 6.580379
```

```
mean(ph_coef$Word[, 1])
```

```
## [1] 6.580379
```

Volviendo al modelo que habíamos propuesto al principio, podemos agregar la interacción entre el tiempo de reacción del prime y la respuesta del prime:

```
ph_lmer3 <- lmer(
    RT ~ Condition + RTtoPrime * ResponseToPrime + (1|Subject) + (1|Word),
    data = primingHeid
  )
summary(ph_lmer3)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: RT ~ Condition + RTtoPrime * ResponseToPrime + (1 | Subject) +
##     (1 | Word)
##    Data: primingHeid
##
## REML criterion at convergence: -411.2
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.4572 -0.6834 -0.1277  0.6066  3.4185
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  Word     (Intercept) 0.001252 0.03538
##  Subject  (Intercept) 0.022214 0.14904
##  Residual             0.029576 0.17198
## Number of obs: 787, groups:  Word, 40; Subject, 26
##
## Fixed effects:
##                                  Estimate Std. Error        df t value
## (Intercept)                       5.27073    0.20767 732.93529  25.380
## Conditionheid                    -0.03876    0.01393 752.41535  -2.782
## RTtoPrime                         0.19871    0.03145 732.94047   6.318
## ResponseToPrimeincorrect          1.63316    0.36722 745.14582   4.447
## RTtoPrime:ResponseToPrimeincorrect -0.22877    0.05511 749.66759  -4.151
```

```
##                                    Pr(>|t|)
## (Intercept)                         < 2e-16 ***
## Conditionheid                       0.00553 **
## RTtoPrime                          4.59e-10 ***
## ResponseToPrimeincorrect           1.00e-05 ***
## RTtoPrime:ResponseToPrimeincorrect 3.69e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) Cndtnh RTtPrm RspnTP
## Conditionhd  0.401
## RTtoPrime   -0.989 -0.431
## RspnsTPrmnc -0.489 -0.169  0.493
## RTtPrm:RsTP  0.489  0.160 -0.493 -0.999
```

Entonces, ¿cómo se interpreta esto? Para cada observación, le se va a sumar a la intersección fija las intersecciones aleatorias correspondientes al sujeto y al ítem en cuestión. El resto se interpreta como en la regresión lineal. Tenemos otros 3 efectos fijos principales: condición, TR de prime y respuesta al prime; y una interacción entre TR de prime y respuesta al prime. Cuando el prime es el neologismo, el TR se reduce (aunque en un grado bajo). Además, la velocidad de respuesta del prime está también asociada a la velocidad de respuesta del target, y una respuesta incorrecta del prime está acompañada de un aun mayor TR del target. La interacción entre TR del prime y respuesta al prime marca que, cuando la respuesta es incorrecta, prácticamente se anula el efecto de TR del prime.

Chequeando los supuestos

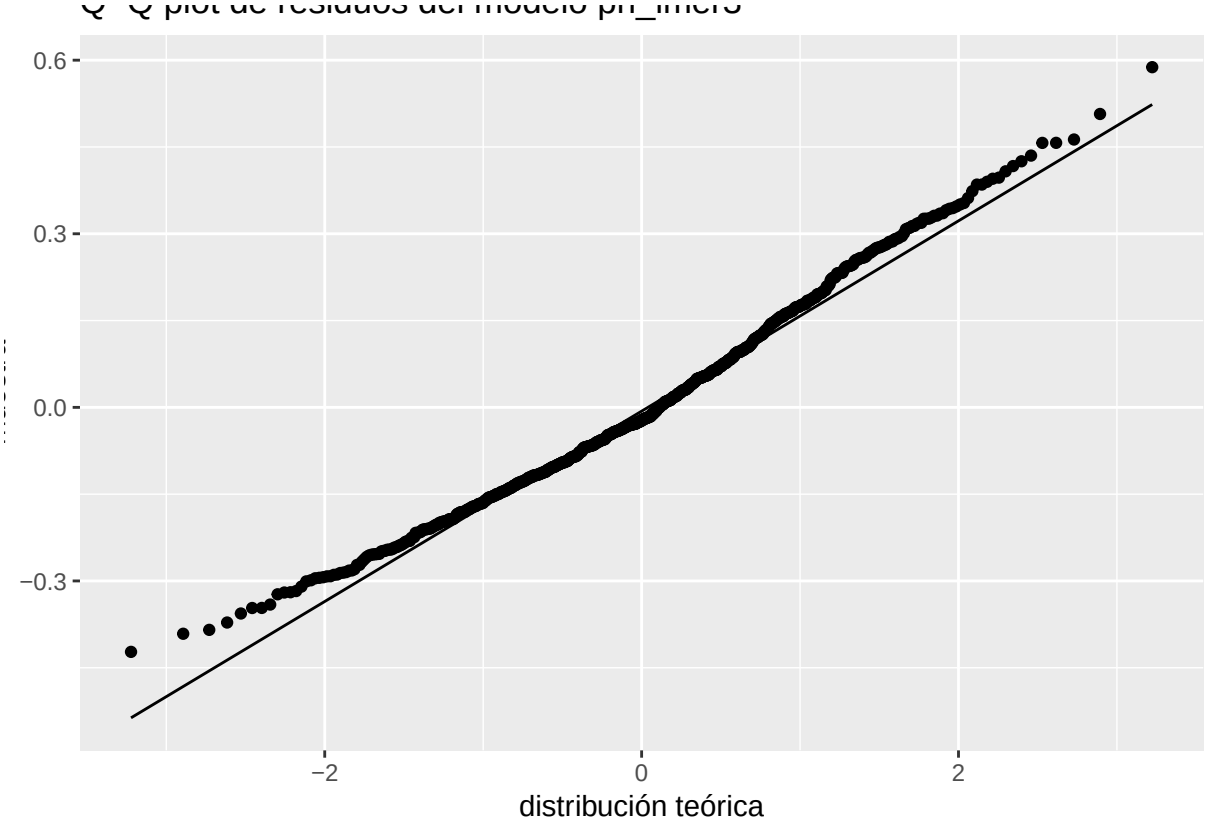
Los supuestos que tenemos que chequear son los mismos que para el modelo de regresión lineal:

- Colinealidad
- Data points influyentes
- Homoscedasticidad
- Normalidad de los residuos
- Independencia

Para verificar que los supuestos de homoscedasticidad y normalidad de los residuos se cumplan, vamos a graficar un Q-Q plot. Para eso vamos a necesitar generar un `data.frame` con los valores de los residuos (que podemos extraer usando la función `residuals()`).

```
ph_lmer3_residuals <- data.frame(residuals = residuals(ph_lmer3))
ggplot(ph_lmer3_residuals, aes(sample = residuals)) +
  stat_qq() +
  stat_qq_line() +
  labs(
    title = "Q-Q plot de residuos del modelo ph_lmer3",
    x = "distribución teórica",
    y = "muestra"
```

```
)
```



En el plot se ve que los puntos graficados se ajustan bastante bien a la recta por lo que podemos confiar en que el modelo es homoscedástico y que los residuos están distribuidos normalmente.

En cuanto a la independencia, es importante remarcar que los modelos de efectos mixtos nos permiten ajustarnos mejor a este supuesto. Al usar un modelo lineal no resolvemos el problema de que hay observaciones en nuestros datos que no son independientes. Por ejemplo, las respuestas de un mismo sujeto. Sin embargo, cuando usamos modelos mixtos también podemos incumplir este supuesto si no consideramos algún efecto aleatorio o fijo que sea relevante para este problema.

Para chequear los data points influyentes, podemos usar la función `influence()` del paquete `influence.ME()`. Esta función lo que hace es entrenar el modelo mixto repetidas veces quitando una observación a la vez para poder explorar hasta qué punto los coeficientes se ven influenciados por puntos individuales.

Mi modelo vs el modelo nulo

Una última cosa que hay que chequear es cómo se desempeña el modelo que entrenamos en comparación con el modelo nulo, esto es el modelo que solo incluye los efectos aleatorios. Para esto, lo que hacemos es correr un ANOVA comparando ambos modelos.

```
ph_lmer_null <- lmer(RT ~ 1 + (1|Subject) + (1|Word), data = primingHeid)
anova(ph_lmer3, ph_lmer_null)
```

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

```
## Data: primingHeid
## Models:
## ph_lmer_null: RT ~ 1 + (1 | Subject) + (1 | Word)
## ph_lmer3: RT ~ Condition + RTtoPrime * ResponseToPrime + (1 | Subject) +
## ph_lmer3:     (1 | Word)
##              Df     AIC     BIC logLik deviance  Chisq Chi Df Pr(>Chisq)
## ph_lmer_null  4 -364.30 -345.63 186.15  -372.30
## ph_lmer3      8 -422.69 -385.34 219.34  -438.69 66.391      4  1.311e-13
##
## ph_lmer_null
## ph_lmer3     ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

El resultado es un valor p muy bajo, por lo que nuestro modelo difícilmente pueda ser explicada por azar.

Pendientes aleatorias

Hasta ahora los modelos que entrenamos incluyen intersecciones aleatorias, específicamente una intersección aleatoria de sujeto y una intersección aleatoria de palabra. Pero también podemos tener algo como esto:

$$RT_{is} = a + a_s + (b + b_s) \cdot condicion_{is} + \varepsilon_{is}$$

En este modelo, estamos agregando una pendiente aleatoria de sujeto para el predictor condición. Las pendientes aleatorias se suman a otras pendientes. Para este mismo ejemplo yo podría agregar una pendiente aleatoria de palabra a la variable condición:

$$RT_{is} = a + a_s + (b + b_s + b_i) \cdot condicion_{is} + \varepsilon_{is}$$

Para definir una pendiente aleatoria en R, usamos el operador `+` dentro de la definición del efecto aleatorio.

```
ph_lmer4 <- lmer(
  RT ~ Condition + (1 | Subject) + (1 + Condition | Word),
  data = primingHeid
  )
```

Entonces, ¿cuál es la diferencia? Una intersección aleatoria lo que hace es ajustar la línea de base. En nuestro ejemplo, una intersección aleatoria de sujeto representa el hecho de que ciertas personas puedan tener una mayor o menor demora en la tarea de decisión léxica. Una pendiente aleatoria, en cambio, representa el hecho de que un predictor puede afectar en distinta medida a distintas personas o con distintos ítems. En el modelo que entrenamos recién, la pendiente aleatoria de palabra para la variable condición representa el hecho de que el tipo de prime puede afectar en forma diferenciada el tiempo de procesamiento para la decisión léxica. Por ejemplo, podríamos pensar que hay bases con las que el sufijo *-heid* se presta más a afijarse sin dejar de formar un neologismo.

Conclusión

En la reunión de hoy hicimos una introducción a qué son los modelos mixtos. Por un lado, introdujimos la distinción entre efectos fijos y efectos aleatorios y la importancia de tenerlos en cuenta en nuestros modelos. También, explicamos por qué los modelos lineales y los ANOVAs no son adecuados para modelarlos. Vimos que podemos incluir efectos aleatorios en un modelo mixto como intersecciones aleatorias o como pendientes aleatorias, y cómo hacer esto en R. Por último repasamos los supuestos a tener en cuenta y cómo verificar que se cumplan.

Referencias y bibliografía consultada

Baayen, R. H. 2008. "Mixed Models." En *Analyzing Linguistic Data: A Practical Introduction to Statistics Using R*, 241–302. Cambridge: Cambridge University Press.

Baayen, R. H., D. J. Davidson, y D. M. Bates. 2018. "Mixed-Effects Modeling with Crossed Random Effects for Subjects and Items." *Journal of Memory and Language*, no. 59: 390–412.

Vaan, L. De, R. Schreuder, y R. H. Baayen. 2007. "Regular Morphologically Complex Neologisms Leave Detectable Traces in the Mental Lexicon." *The Mental Lexicon* 2: 1–23.

Winter, Bodo. 2016a. "Linear Models and Linear Mixed Effects Models in R: Tutorial 1." http://www.bodowinter.com/tutorial/bw_LME_tutorial1.pdf.

———. 2016b. "Linear Models and Linear Mixed Effects Models in R: Tutorial 2." http://www.bodowinter.com/tutorial/bw_LME_tutorial1.pdf.