Introduction à la modélisation statistique bayésienne

Ladislas Nalborczyk

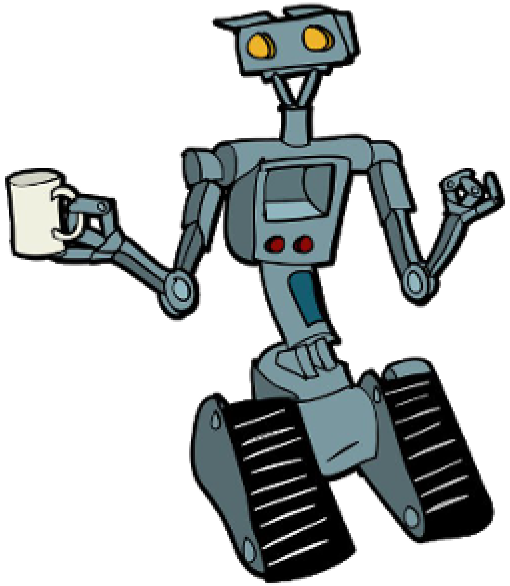
LPC, LNC, CNRS, Aix-Marseille Univ.

Planning

Modèles multi-niveaux

Le but est de construire un modèle qui puisse **apprendre à plusieurs niveaux**, qui puisse produire des estimations qui seront informées par les différents groupes présents dans les données. Nous allons suivre l'exemple suivant tout au long de ce cours.

Imaginons que nous ayons construit un robot visiteur de cafés, et que celui-ci s'amuse à mesurer le temps d'attente après avoir commandé. Ce robot visite 20 cafés différents, 5 fois le matin et 5 fois l'après-midi.



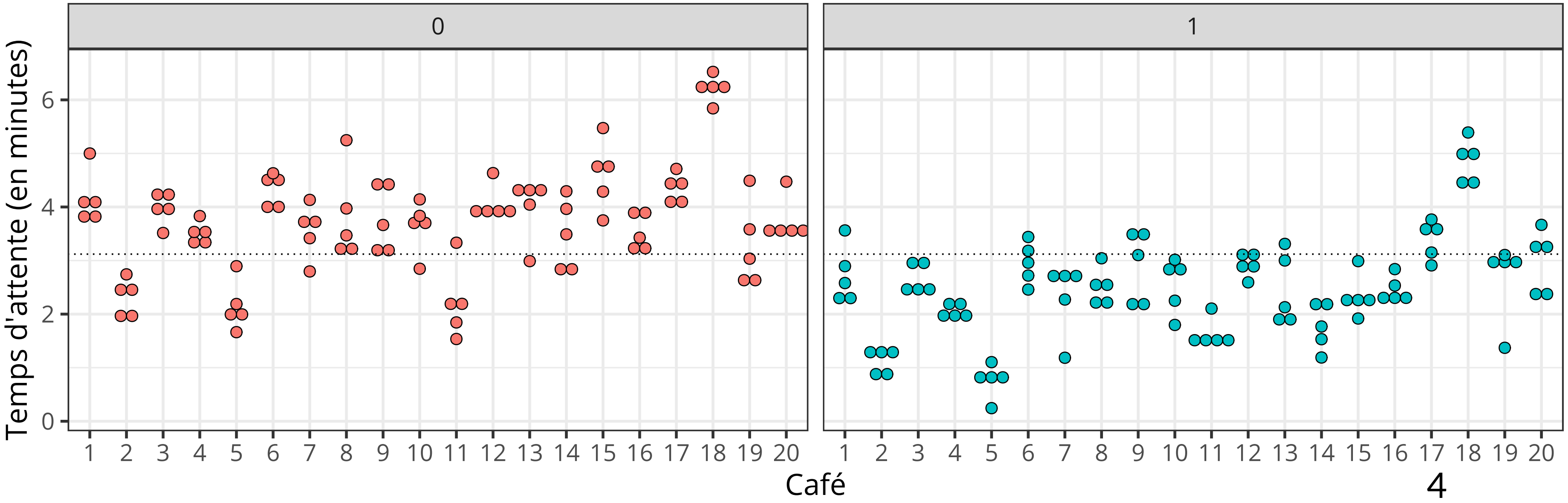
Robot et café

```
library(tidyverse)

df <- read.csv("data/robot.csv")
head(df, 15)
```

```
   cafe afternoon      wait
1     1         0 4.9989926
2     1         1 2.2133944
3     1         0 4.1866730
4     1         1 3.5624399
5     1         0 3.9956779
6     1         1 2.8957176
7     1         0 3.7804582
8     1         1 2.3844837
9     1         0 3.8617982
10    1         1 2.5800004
11    2         0 2.7421223
12    2         1 1.3525907
13    2         0 2.5215095
14    2         1 0.9628102
15    2         0 1.9543977
```

Robot et café

```
df %>%
  ggplot(aes(x = factor(cafe), y = wait, fill = factor(afternoon) ) ) +
  geom_dotplot(
    stackdir = "center", binaxis = "y",
    dotsize = 1, show.legend = FALSE
    ) +
  geom_hline(yintercept = mean(df$wait), linetype = 3) +
  facet_wrap(~afternoon, ncol = 2) +
  labs(x = "Café", y = "Temps d'attente (en minutes)")
```



Robot et café, premier modèle

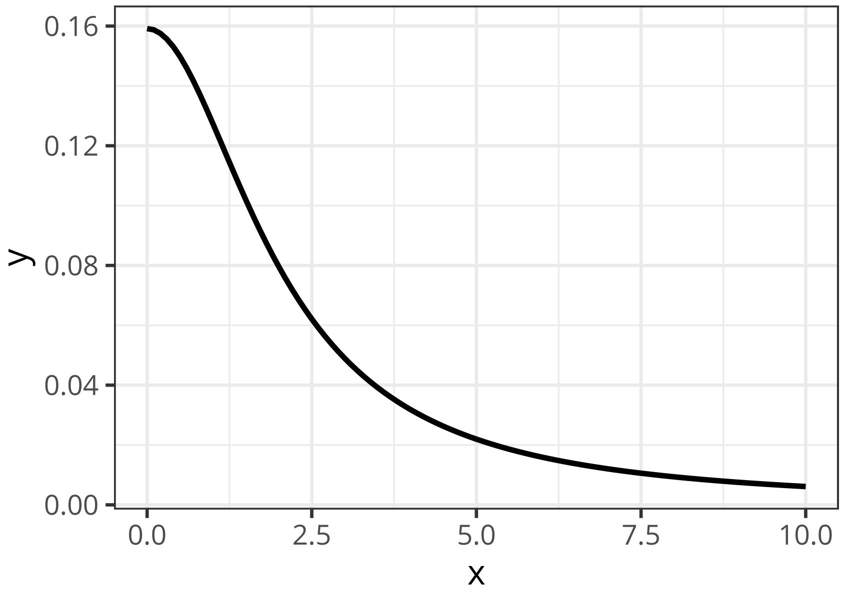
On peut construire un premier modèle, qui estime le temps moyen (sur tous les bistrots confondus) pour être servi.

$$
\begin{aligned}
w_i &\sim \text{Normal}(\mu_i, \sigma) \\
\mu_i &= \alpha \\
\alpha &\sim \text{Normal}(5, 10) \\
\sigma &\sim \text{HalfCauchy}(0, 2)
\end{aligned}
$$

Half-Cauchy

$$p(x \mid x_0, \gamma) = \left(\pi\gamma\left[1 + \left(\frac{x - x_0}{\gamma}\right)^2\right]\right)^{-1}$$

```r
ggplot(data = data.frame(x = c(0, 10) ), aes(x = x) ) +
    stat_function(
        fun = dcauchy,
        args = list(location = 0, scale = 2), size = 1.5
        )
```



Robot et café, premier modèle

Robot et café, premier modèle

```r
library(rethinking)
library(brms)

mod1 <- brm(
  wait ~ 1,
  prior = c(
    set_prior("normal(5, 10)", class = "Intercept"),
    set_prior("cauchy(0, 2)", class = "sigma")
    ),
  data = df,
  cores = parallel::detectCores()
  )
```

Robot et café, premier modèle

```r
library(rethinking)
library(brms)

mod1 <- brm(
  wait ~ 1,
  prior = c(
    set_prior("normal(5, 10)", class = "Intercept"),
    set_prior("cauchy(0, 2)", class = "sigma")
    ),
  data = df,
  cores = parallel::detectCores()
  )
```

```r
posterior_summary(mod1, probs = c(0.025, 0.975) )
```

Robot et café, premier modèle

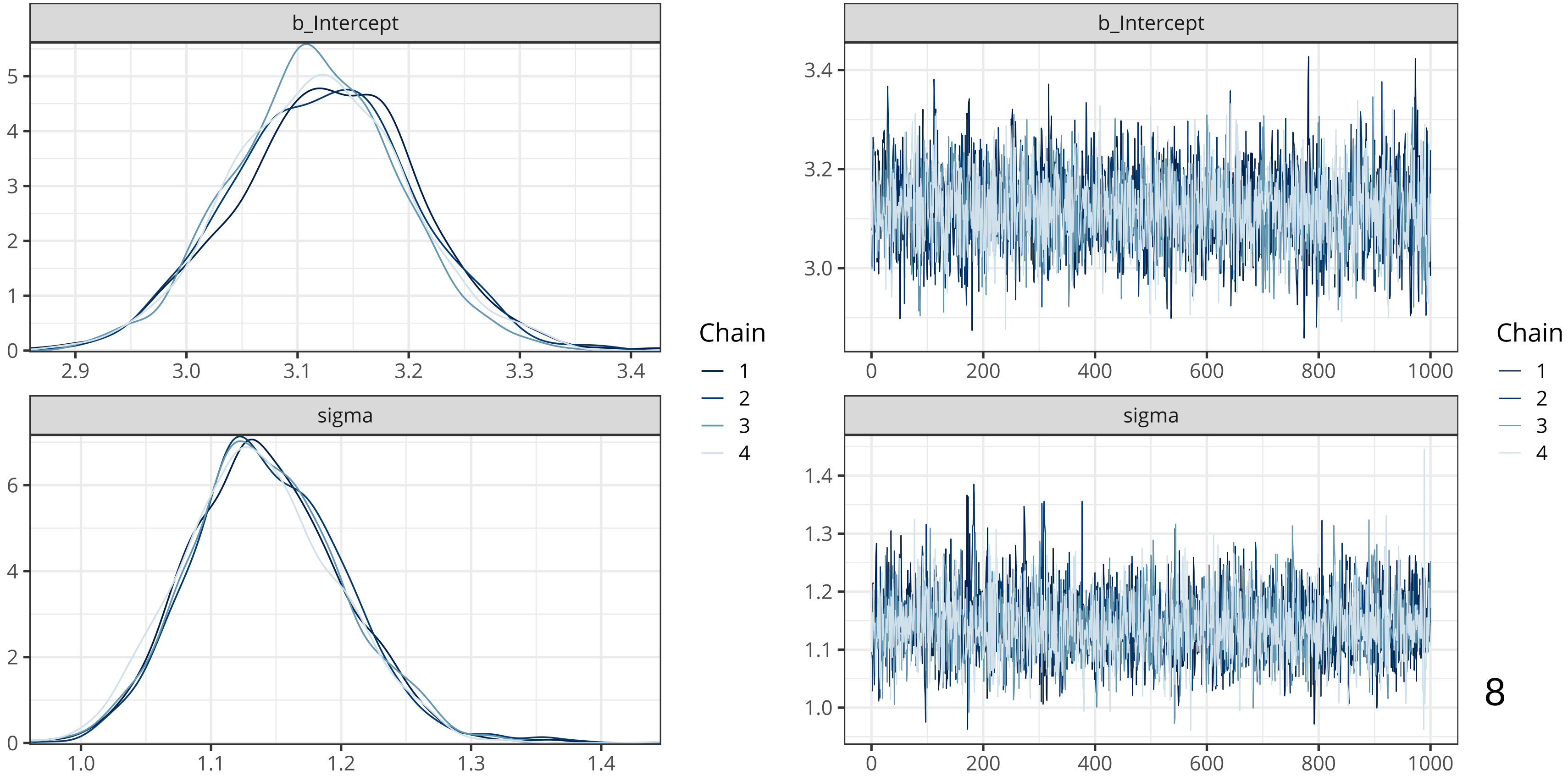
```r
library(rethinking)
library(brms)

mod1 <- brm(
  wait ~ 1,
  prior = c(
    set_prior("normal(5, 10)", class = "Intercept"),
    set_prior("cauchy(0, 2)", class = "sigma")
    ),
  data = df,
  cores = parallel::detectCores()
  )
```

```r
posterior_summary(mod1, probs = c(0.025, 0.975) )
```

```
               Estimate  Est.Error       Q2.5      Q97.5
b_Intercept    3.120812 0.07842501    2.966944    3.276102
sigma          1.141664 0.05816792    1.035442    1.261113
lp__        -314.603239 1.00223908 -317.127929 -313.645961
```

Diagnostic plot

```
plot(
  mod1, combo = c("dens_overlay", "trace"),
  theme = theme_bw(base_size = 16, base_family = "Open Sans")
  )
```



Un intercept par café

Deuxième modèle qui estime un intercept par café. Équivalent à construire 20 *dummy variables*.

Un intercept par café

Deuxième modèle qui estime un intercept par café. Équivalent à construire 20 *dummy variables*.

$$
\begin{aligned}
w_i &\sim \mathrm{Normal}(\mu_i, \sigma) \\
\mu_i &= \alpha_{\text{café}[i]} \\
\alpha_{\text{café}[i]} &\sim \mathrm{Normal}(5, 10) \\
\sigma &\sim \mathrm{HalfCauchy}(0, 2)
\end{aligned}
$$

Un intercept par café

Deuxième modèle qui estime un intercept par café. Équivalent à construire 20 *dummy variables*.

$$
\begin{aligned}
w_i &\sim \text{Normal}(\mu_i, \sigma) \\
\mu_i &= \alpha_{\text{café}[i]} \\
\alpha_{\text{café}[i]} &\sim \text{Normal}(5, 10) \\
\sigma &\sim \text{HalfCauchy}(0, 2)
\end{aligned}
$$

```r
mod2 <- brm(
  wait ~ 0 + factor(cafe),
  prior = c(
    set_prior("normal(5, 10)", class = "b"),
    set_prior("cauchy(0, 2)", class = "sigma")
    ),
  data = df,
  cores = parallel::detectCores()
  )
```

Un intercept par café

```r
posterior_summary(mod2)
```

```
                 Estimate  Est.Error         Q2.5        Q97.5
b_factorcafe1    3.447826 0.26320601    2.9340206    3.9592694
b_factorcafe2    1.731522 0.26318882    1.2125090    2.2399696
b_factorcafe3    3.320775 0.25982381    2.8159365    3.8334424
b_factorcafe4    2.789301 0.26028359    2.2797502    3.2959125
b_factorcafe5    1.461260 0.25790026    0.9573526    1.9682942
b_factorcafe6    3.639357 0.26030385    3.1223523    4.1561719
b_factorcafe7    2.945275 0.26454629    2.4211992    3.4610923
b_factorcafe8    3.169239 0.26355173    2.6475875    3.6875978
b_factorcafe9    3.338017 0.25706741    2.8450755    3.8408056
b_factorcafe10   3.096117 0.26262895    2.5877355    3.5994435
b_factorcafe11   1.916892 0.26155356    1.3982956    2.4272224
b_factorcafe12   3.490926 0.25875445    2.9862146    4.0131991
b_factorcafe13   3.220629 0.25639054    2.7190859    3.7160198
b_factorcafe14   2.629674 0.25935726    2.1267711    3.1327653
b_factorcafe15   3.474186 0.26216959    2.9560301    3.9932096
b_factorcafe16   2.997613 0.26246522    2.4991750    3.5111527
b_factorcafe17   3.877741 0.25453380    3.3778956    4.3793450
b_factorcafe18   5.532561 0.25145915    5.0490608    6.0256970
b_factorcafe19   2.971695 0.26994984    2.4515471    3.5055459
b_factorcafe20   3.363134 0.26888873    2.8260119    3.9030927
sigma            0.821503 0.04425388    0.7413788    0.9163758
lp__          -310.296703 3.48432717 -317.9993798 -304.5513934
```

Modèle multi-niveaux

Est-ce qu'on ne pourrait pas faire en sorte que le temps mesuré au café 1 *informe* la mesure réalisée au café 2, et au café 3 ? Ainsi que le temps moyen pour être servi ? Nous allons apprendre le prior à partir des données…

Modèle multi-niveaux

Est-ce qu'on ne pourrait pas faire en sorte que le temps mesuré au café 1 *informe* la mesure réalisée au café 2, et au café 3 ? Ainsi que le temps moyen pour être servi ? Nous allons apprendre le prior à partir des données…

$$
\begin{aligned}
\text{Niveau 1 : } w_i &\sim \text{Normal}(\mu_i, \sigma) \\
\mu_i &= \alpha_{\text{café}[i]} \\
\text{Niveau 2 : } \alpha_{\text{café}} &\sim \text{Normal}(\alpha, \sigma_{\text{café}}) \\
\alpha &\sim \text{Normal}(5, 10) \\
\sigma_{\text{café}} &\sim \text{HalfCauchy}(0, 2) \\
\sigma &\sim \text{HalfCauchy}(0, 2)
\end{aligned}
$$

Modèle multi-niveaux

Est-ce qu'on ne pourrait pas faire en sorte que le temps mesuré au café 1 *informe* la mesure réalisée au café 2, et au café 3 ? Ainsi que le temps moyen pour être servi ? Nous allons apprendre le prior à partir des données…

$$
\begin{aligned}
\text{Niveau 1 : } w_i &\sim \text{Normal}(\mu_i, \sigma) \\
\mu_i &= \alpha_{\text{café}[i]} \\
\text{Niveau 2 : } \alpha_{\text{café}} &\sim \text{Normal}(\alpha, \sigma_{\text{café}}) \\
\alpha &\sim \text{Normal}(5, 10) \\
\sigma_{\text{café}} &\sim \text{HalfCauchy}(0, 2) \\
\sigma &\sim \text{HalfCauchy}(0, 2)
\end{aligned}
$$

Le prior de l'intercept pour chaque café ($\alpha_{\text{café}}$) est maintenant fonction de deux paramètres (α et $\sigma_{\text{café}}$). α et $\sigma_{\text{café}}$ sont appelés des **hyper-paramètres**, ce sont des paramètres pour des paramètres, et leurs priors sont appelés des **hyperpriors**. Il y a deux niveaux dans le modèle…

Équivalences (encore)

$$
\begin{aligned}
w_i &\sim \text{Normal}(\mu_i, \sigma) \\
\mu_i &= \alpha_{\text{café}[i]} \\
\alpha_{\text{café}} &\sim \text{Normal}(\alpha, \sigma_{\text{café}})
\end{aligned}
$$

Équivalences (encore)

$$
\begin{aligned}
w_i &\sim \text{Normal}(\mu_i, \sigma) \\
\mu_i &= \alpha_{\text{café}[i]} \\
\alpha_{\text{café}} &\sim \text{Normal}(\alpha, \sigma_{\text{café}})
\end{aligned}
$$

NB : α est ici défini dans le prior de $\alpha_{\text{café}}$ mais on pourrait, de la même manière, le définir dans le modèle linéaire :

Équivalences (encore)

$$
\begin{aligned}
w_i &\sim \text{Normal}(\mu_i, \sigma) \\
\mu_i &= \alpha_{\text{café}[i]} \\
\alpha_{\text{café}} &\sim \text{Normal}(\alpha, \sigma_{\text{café}})
\end{aligned}
$$

NB : α est ici défini dans le prior de $\alpha_{\text{café}}$ mais on pourrait, de la même manière, le définir dans le modèle linéaire :

$$
\begin{aligned}
w_i &\sim \text{Normal}(\mu_i, \sigma) \\
\mu_i &= \alpha + \alpha_{\text{café}[i]} \\
\alpha_{\text{café}} &\sim \text{Normal}(0, \sigma_{\text{café}}) \\
\alpha &\sim \text{Normal}(5, 10)
\end{aligned}
$$

Équivalences (encore)

$$
\begin{aligned}
w_i &\sim \mathrm{Normal}(\mu_i, \sigma) \\
\mu_i &= \alpha_{\text{café}[i]} \\
\alpha_{\text{café}} &\sim \mathrm{Normal}(\alpha, \sigma_{\text{café}})
\end{aligned}
$$

NB : α est ici défini dans le prior de $\alpha_{\text{café}}$ mais on pourrait, de la même manière, le définir dans le modèle linéaire :

$$
\begin{aligned}
w_i &\sim \mathrm{Normal}(\mu_i, \sigma) \\
\mu_i &= \alpha + \alpha_{\text{café}[i]} \\
\alpha_{\text{café}} &\sim \mathrm{Normal}(0, \sigma_{\text{café}}) \\
\alpha &\sim \mathrm{Normal}(5, 10)
\end{aligned}
$$

On peut toujours “enlever” la moyenne d'une distribution gaussienne et la considérer comme une constante plus une gaussienne centrée sur zéro.

Équivalences (encore)

$$
\begin{aligned}
w_i &\sim \text{Normal}(\mu_i, \sigma) \\
\mu_i &= \alpha_{\text{café}[i]} \\
\alpha_{\text{café}} &\sim \text{Normal}(\alpha, \sigma_{\text{café}})
\end{aligned}
$$

NB : α est ici défini dans le prior de $\alpha_{\text{café}}$ mais on pourrait, de la même manière, le définir dans le modèle linéaire :

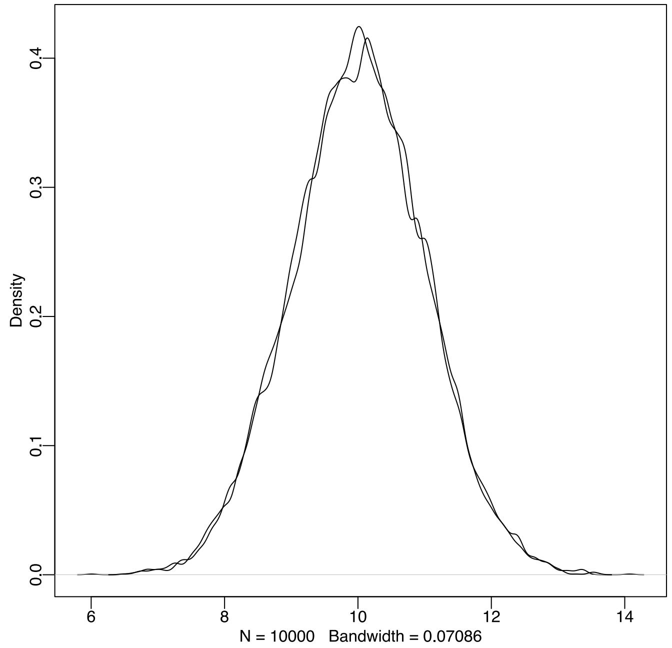
$$
\begin{aligned}
w_i &\sim \text{Normal}(\mu_i, \sigma) \\
\mu_i &= \alpha + \alpha_{\text{café}[i]} \\
\alpha_{\text{café}} &\sim \text{Normal}(0, \sigma_{\text{café}}) \\
\alpha &\sim \text{Normal}(5, 10)
\end{aligned}
$$

On peut toujours “enlever” la moyenne d'une distribution gaussienne et la considérer comme une constante plus une gaussienne centrée sur zéro.

NB : quand α est défini dans le modèle linéaire, les $\alpha_{\text{café}}$ représentent des déviations de l'intercept moyen. Il faut donc ajouter α et $\alpha_{\text{café}}$ pour obtenir le temps d'attente moyen par café…

Équivalences (encore)

```r
y1 <- rnorm(1e4, 10, 1)
y2 <- 10 + rnorm(1e4, 0, 1)

dens(y1)
dens(y2, add = TRUE)
```

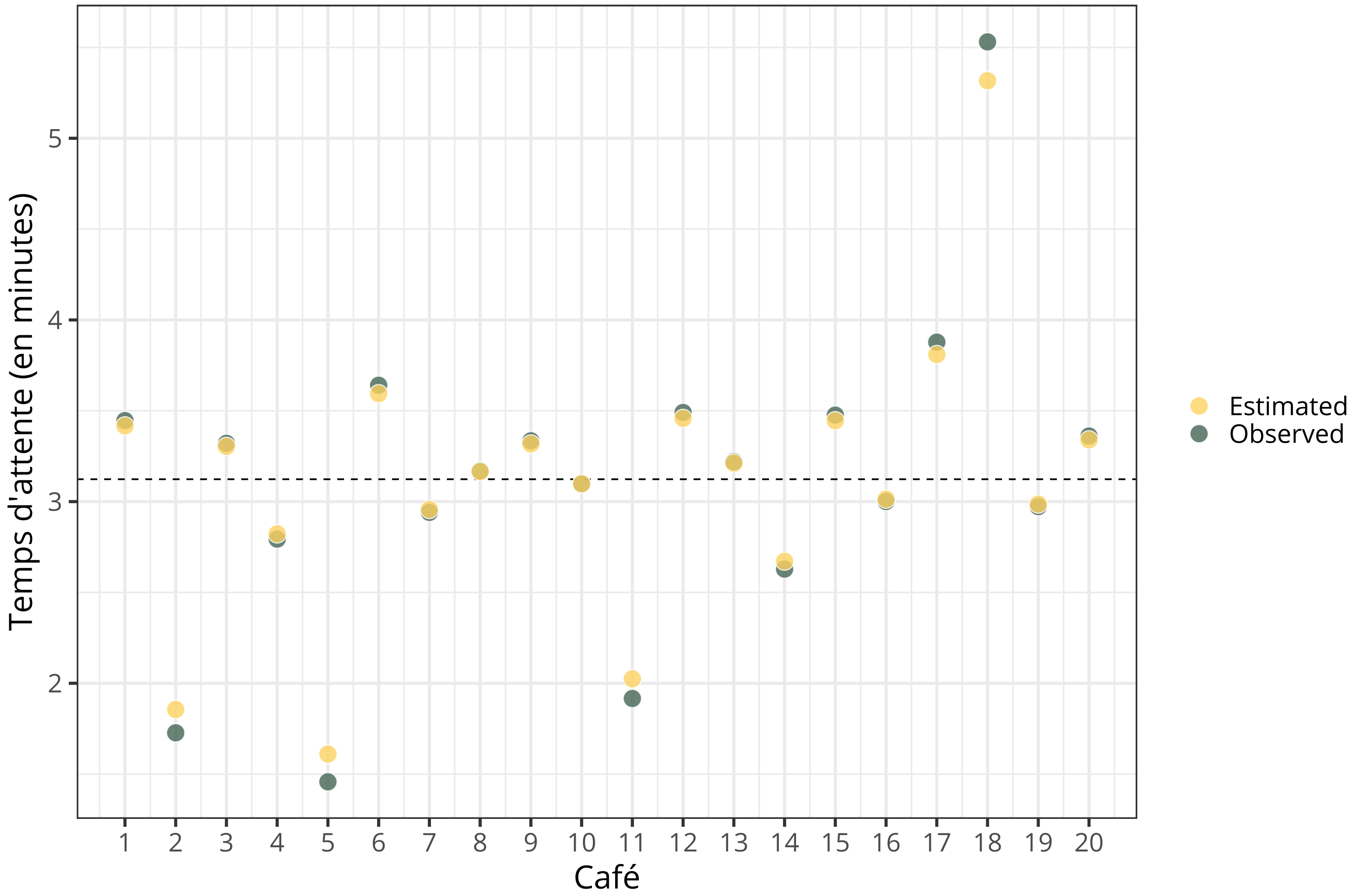


Modèle multi-niveaux

```r
mod3 <- brm(
  wait ~ 1 + (1 | cafe),
  prior = c(
    set_prior("normal(5, 10)", class = "Intercept"),
    set_prior("cauchy(0, 2)", class = "sigma"),
    set_prior("cauchy(0, 2)", class = "sd")
    ),
  data = df,
  warmup = 1000, iter = 5000,
  cores = parallel::detectCores()
  )
```

Ce modèle a 23 paramètres, l'intercept général α, la variabilité résiduelle σ, la variabilité entre les cafés $\sigma_{\text{café}}$, et un intercept par café.

Shrinkage



Temps d'attente (en minutes)
5
4
3
2
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
Café
Estimated
Observed

Shrinkage magic (Efron & Morris, 1977)

Stein's Paradox in Statistics

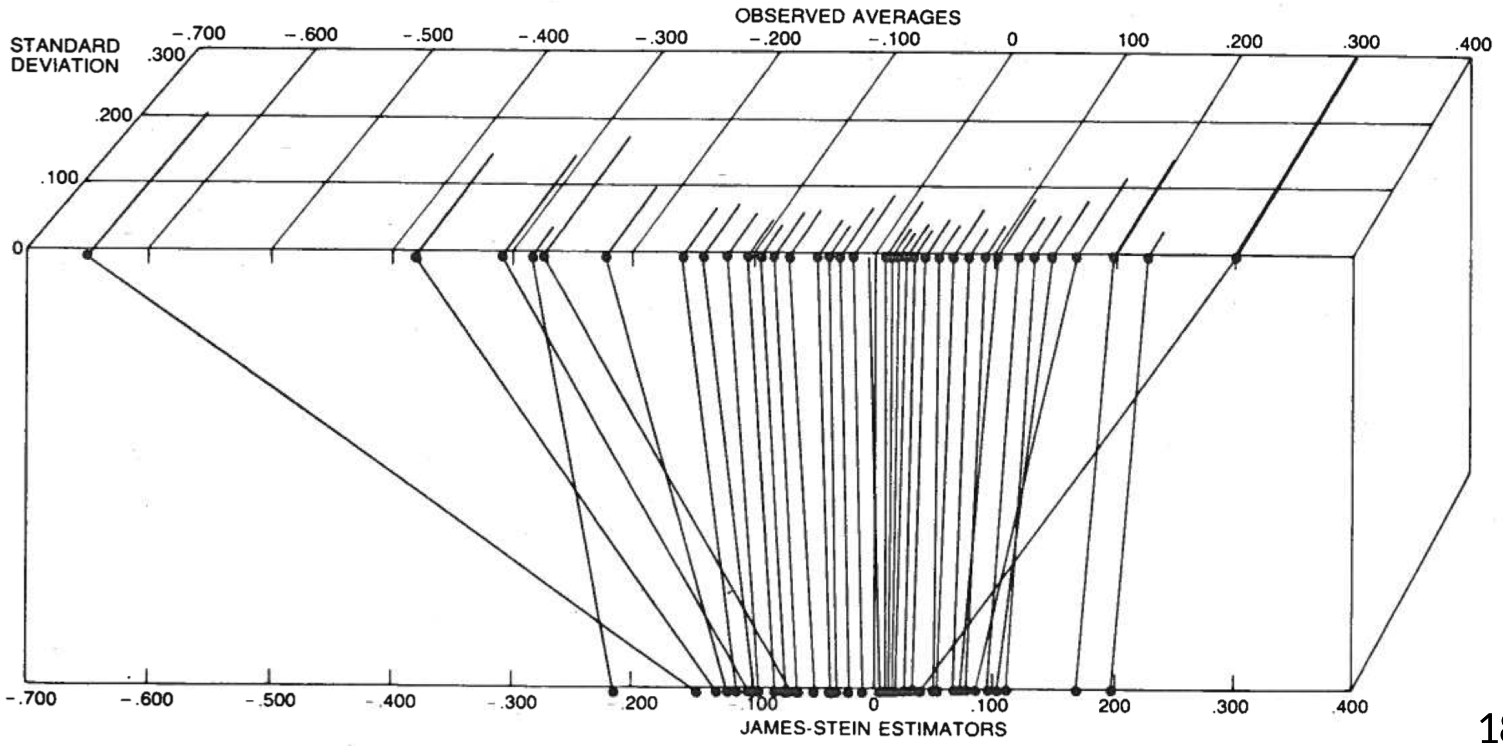
The best guess about the future is usually obtained by computing the average of past events. Stein's paradox defines circumstances in which there are estimators better than the arithmetic average

by Bradley Efron and Carl Morris

The James-Stein estimator is defined as $z = \bar{y} + c(y - \bar{y})$, where $\bar{y}$ is the grand average, y an individual estimation and c a constant, the "shrinking factor" (Efron & Morris, 1977).

Shrinkage magic (Efron & Morris, 1977)

The James-Stein estimator is determined both by the variability in the measure (e.g., its standard deviation, influencing the shrinking factor c) and by its distance to the average estimation (i.e., $y - \bar{y}$). In other words, extreme observations are trusted less. Shrinkage therefore acts as **a safeguard against overfitting** in multilevel models.



Pooling

Le *shrinkage* observé slide précédente est dû à des phénomènes de partage (*pooling*) de l'information entre les cafés. L'estimation de l'intercept pour chaque café informe l'estimation de l'intercept des autres cafés, ainsi que l'estimation de l'intercept général (i.e., la moyenne générale).

Pooling

Le *shrinkage* observé slide précédente est dû à des phénomènes de partage (*pooling*) de l'information entre les cafés. L'estimation de l'intercept pour chaque café informe l'estimation de l'intercept des autres cafés, ainsi que l'estimation de l'intercept général (i.e., la moyenne générale).

On distingue en général trois perspectives (ou stratégies) :

Pooling

Le *shrinkage* observé slide précédente est dû à des phénomènes de partage (*pooling*) de l'information entre les cafés. L'estimation de l'intercept pour chaque café informe l'estimation de l'intercept des autres cafés, ainsi que l'estimation de l'intercept général (i.e., la moyenne générale).

On distingue en général trois perspectives (ou stratégies) :

Pooling

Le *shrinkage* observé slide précédente est dû à des phénomènes de partage (*pooling*) de l'information entre les cafés. L'estimation de l'intercept pour chaque café informe l'estimation de l'intercept des autres cafés, ainsi que l'estimation de l'intercept général (i.e., la moyenne générale).

On distingue en général trois perspectives (ou stratégies) :

- **Complete pooling** : on suppose que le temps d'attente est invariant, on estime un intercept commun (`mod1`)

Pooling

Le *shrinkage* observé slide précédente est dû à des phénomènes de partage (*pooling*) de l'information entre les cafés. L'estimation de l'intercept pour chaque café informe l'estimation de l'intercept des autres cafés, ainsi que l'estimation de l'intercept général (i.e., la moyenne générale).

On distingue en général trois perspectives (ou stratégies) :

- **Complete pooling** : on suppose que le temps d'attente est invariant, on estime un intercept commun (`mod1`)

Pooling

Le *shrinkage* observé slide précédente est dû à des phénomènes de partage (*pooling*) de l'information entre les cafés. L'estimation de l'intercept pour chaque café informe l'estimation de l'intercept des autres cafés, ainsi que l'estimation de l'intercept général (i.e., la moyenne générale).

On distingue en général trois perspectives (ou stratégies) :

- **Complete pooling** : on suppose que le temps d'attente est invariant, on estime un intercept commun (`mod1`)
- **No pooling** : on suppose que les temps d'attente de chaque café sont uniques et indépendants: on estime un intercept par café, mais sans informer le *niveau* supérieur (`mod2`)

Pooling

Le *shrinkage* observé slide précédente est dû à des phénomènes de partage (*pooling*) de l'information entre les cafés. L'estimation de l'intercept pour chaque café informe l'estimation de l'intercept des autres cafés, ainsi que l'estimation de l'intercept général (i.e., la moyenne générale).

On distingue en général trois perspectives (ou stratégies) :

- **Complete pooling** : on suppose que le temps d'attente est invariant, on estime un intercept commun (`mod1`)
- **No pooling** : on suppose que les temps d'attente de chaque café sont uniques et indépendants: on estime un intercept par café, mais sans informer le *niveau* supérieur (`mod2`)

Pooling

Le *shrinkage* observé slide précédente est dû à des phénomènes de partage (*pooling*) de l'information entre les cafés. L'estimation de l'intercept pour chaque café informe l'estimation de l'intercept des autres cafés, ainsi que l'estimation de l'intercept général (i.e., la moyenne générale).

On distingue en général trois perspectives (ou stratégies) :

- **Complete pooling** : on suppose que le temps d'attente est invariant, on estime un intercept commun (`mod1`)
- **No pooling** : on suppose que les temps d'attente de chaque café sont uniques et indépendants: on estime un intercept par café, mais sans informer le *niveau* supérieur (`mod2`)
- **Partial pooling** : on utilise un prior adaptatif, comme dans l'exemple précédent (`mod3`)

Pooling

Le *shrinkage* observé slide précédente est dû à des phénomènes de partage (*pooling*) de l'information entre les cafés. L'estimation de l'intercept pour chaque café informe l'estimation de l'intercept des autres cafés, ainsi que l'estimation de l'intercept général (i.e., la moyenne générale).

On distingue en général trois perspectives (ou stratégies) :

- **Complete pooling** : on suppose que le temps d'attente est invariant, on estime un intercept commun (`mod1`)
- **No pooling** : on suppose que les temps d'attente de chaque café sont uniques et indépendants: on estime un intercept par café, mais sans informer le *niveau* supérieur (`mod2`)
- **Partial pooling** : on utilise un prior adaptatif, comme dans l'exemple précédent (`mod3`)

La stratégie *complete pooling* en général *underfit* les données (faibles capacités de prédiction) tandis que le stratégie *no pooling* revient à *overfitter* les données (faibles capacités de prédiction ici aussi). La stratégie *partial pooling* (celle des modèles multi-niveaux) permet de balancer *underfitting* et *overfitting*.

Comparaison de modèles

On peut comparer ces trois modèles en utilisant le WAIC (discuté au Cours n°07).

Comparaison de modèles

On peut comparer ces trois modèles en utilisant le WAIC (discuté au Cours n°07).

```r
# calcul du WAIC et ajout du WAIC à chaque modèle
mod1 <- add_criterion(mod1, "waic")
mod2 <- add_criterion(mod2, "waic")
mod3 <- add_criterion(mod3, "waic")

# comparaison des WAIC de chaque modèle
w <- loo_compare(mod1, mod2, mod3, criterion = "waic")
print(w, simplify = FALSE)
```

Comparaison de modèles

On peut comparer ces trois modèles en utilisant le WAIC (discuté au Cours n°07).

```r
# calcul du WAIC et ajout du WAIC à chaque modèle
mod1 <- add_criterion(mod1, "waic")
mod2 <- add_criterion(mod2, "waic")
mod3 <- add_criterion(mod3, "waic")

# comparaison des WAIC de chaque modèle
w <- loo_compare(mod1, mod2, mod3, criterion = "waic")
print(w, simplify = FALSE)
```

```
     elpd_diff se_diff elpd_waic se_elpd_waic p_waic se_p_waic waic   se_waic
mod3    0.0       0.0  -253.9       8.3         18.2    1.5     507.7   16.6
mod2   -0.9       1.3  -254.7       8.4         19.7    1.6     509.4   16.9
mod1  -57.3      10.6  -311.1      10.6          2.1    0.4     622.2   21.1
```

Comparaison de modèles

On peut comparer ces trois modèles en utilisant le WAIC (discuté au Cours n°07).

```r
# calcul du WAIC et ajout du WAIC à chaque modèle
mod1 <- add_criterion(mod1, "waic")
mod2 <- add_criterion(mod2, "waic")
mod3 <- add_criterion(mod3, "waic")

# comparaison des WAIC de chaque modèle
w <- loo_compare(mod1, mod2, mod3, criterion = "waic")
print(w, simplify = FALSE)
```

```
     elpd_diff se_diff elpd_waic se_elpd_waic p_waic se_p_waic waic   se_waic
mod3    0.0       0.0  -253.9       8.3         18.2    1.5     507.7   16.6
mod2   -0.9       1.3  -254.7       8.4         19.7    1.6     509.4   16.9
mod1  -57.3      10.6  -311.1      10.6          2.1    0.4     622.2   21.1
```

On remarque que le modèle 3 a seulement 18 *effective parameters* (*pWAIC*), et moins de paramètres que le modèle 2, alors qu'il en a *en réalité* 2 de plus... `posterior_summary(mod3)[3, 1]` nous donne le sigma du prior adaptatif des $\alpha_{\text{café}}$ ($\sigma_{\text{café}} = 0.82$). On remarque que ce sigma est très faible et correspond à assigner un prior très contraignant, ou *régularisateur*.

Comparaison de modèles

On compare le premier modèle (*complete pooling model*) et le troisième modèle (*partial pooling model*).

Comparaison de modèles

On compare le premier modèle (*complete pooling model*) et le troisième modèle (*partial pooling model*).

```
posterior_summary(mod1, pars = c("^b", "sigma") )
```

Comparaison de modèles

On compare le premier modèle (*complete pooling model*) et le troisième modèle (*partial pooling model*).

```
posterior_summary(mod1, pars = c("^b", "sigma") )
```

```
            Estimate  Est.Error     Q2.5    Q97.5
b_Intercept 3.120812 0.07842501 2.966944 3.276102
sigma       1.141664 0.05816792 1.035442 1.261113
```

Comparaison de modèles

On compare le premier modèle (*complete pooling model*) et le troisième modèle (*partial pooling model*).

```r
posterior_summary(mod1, pars = c("^b", "sigma") )
```

```
            Estimate  Est.Error     Q2.5    Q97.5
b_Intercept 3.120812 0.07842501 2.966944 3.276102
sigma       1.141664 0.05816792 1.035442 1.261113
```

```r
posterior_summary(mod3, pars = c("^b", "sigma") )
```

Comparaison de modèles

On compare le premier modèle (*complete pooling model*) et le troisième modèle (*partial pooling model*).

```
posterior_summary(mod1, pars = c("^b", "sigma") )
```

```
            Estimate  Est.Error     Q2.5    Q97.5
b_Intercept 3.120812 0.07842501 2.966944 3.276102
sigma       1.141664 0.05816792 1.035442 1.261113
```

```
posterior_summary(mod3, pars = c("^b", "sigma") )
```

```
             Estimate  Est.Error      Q2.5     Q97.5
b_Intercept 3.1232197 0.20418306 2.7166225 3.5240819
sigma       0.8219111 0.04373907 0.7418185 0.9134082
```

Comparaison de modèles

On compare le premier modèle (*complete pooling model*) et le troisième modèle (*partial pooling model*).

```
posterior_summary(mod1, pars = c("^b", "sigma") )
```

```
             Estimate  Est.Error     Q2.5    Q97.5
b_Intercept 3.120812 0.07842501 2.966944 3.276102
sigma       1.141664 0.05816792 1.035442 1.261113
```

```
posterior_summary(mod3, pars = c("^b", "sigma") )
```

```
             Estimate  Est.Error      Q2.5     Q97.5
b_Intercept 3.1232197 0.20418306 2.7166225 3.5240819
sigma       0.8219111 0.04373907 0.7418185 0.9134082
```

Les deux modèles font la même prédiction (en moyenne) pour α, mais le modèle 3 est plus incertain de sa prédiction que le modèle 1…

Comparaison de modèles

On compare le premier modèle (*complete pooling model*) et le troisième modèle (*partial pooling model*).

```r
posterior_summary(mod1, pars = c("^b", "sigma") )
```

```
            Estimate  Est.Error     Q2.5    Q97.5
b_Intercept 3.120812 0.07842501 2.966944 3.276102
sigma       1.141664 0.05816792 1.035442 1.261113
```

```r
posterior_summary(mod3, pars = c("^b", "sigma") )
```

```
             Estimate  Est.Error      Q2.5     Q97.5
b_Intercept 3.1232197 0.20418306 2.7166225 3.5240819
sigma       0.8219111 0.04373907 0.7418185 0.9134082
```

Les deux modèles font la même prédiction (en moyenne) pour α, mais le modèle 3 est plus incertain de sa prédiction que le modèle 1…

L'estimation de σ du modèle 3 est plus petite que celle du modèle 1 car le modèle 3 **décompose** la variabilité non expliquée en deux sources : la variabilité du temps d'attente entre les cafés $\sigma_{\text{café}}$ et la variabilité résiduelle σ.

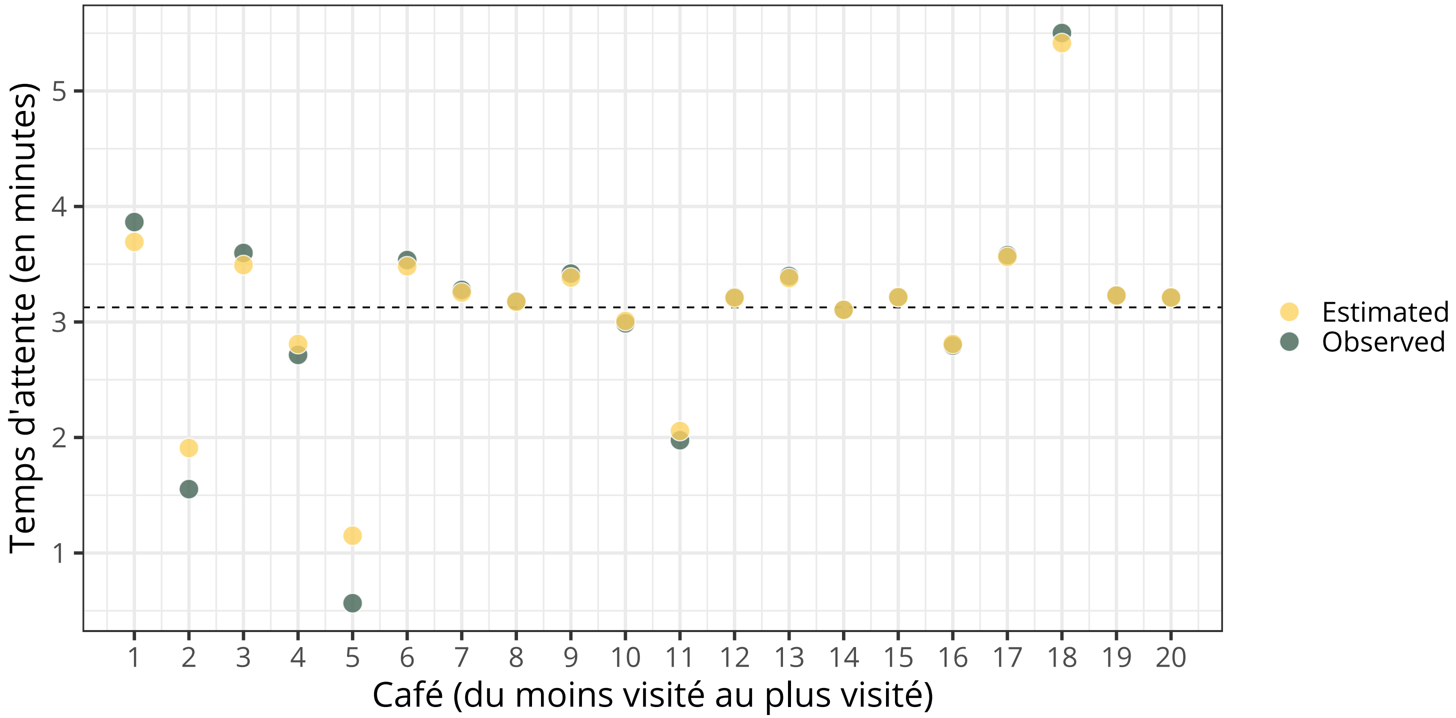
Robot et café

Imaginons que notre robot ne visite pas tous les cafés le même nombre de fois (comme dans le cas précédent) mais qu'il visite plus souvent les cafés proches de chez lui…

```
df2 <- read.csv("data/robot_inequal.csv")

mod4 <- brm(
  wait ~ 1 + (1 | cafe),
  prior = c(
    set_prior("normal(5, 10)", class = "Intercept"),
    set_prior("cauchy(0, 2)", class = "sigma"),
    set_prior("cauchy(0, 2)", class = "sd")
    ),
  data = df2,
  warmup = 1000, iter = 5000,
  cores = parallel::detectCores()
  )
```

Shrinkage

On observe que les cafés qui sont souvent visités (à droite) subissent moins l'effet du *shrinkage*. Leur estimation est moins "tirée" vers la moyenne générale que les estimations des cafés les moins souvent visités (à gauche).



Aparté : effets fixes et effets aléatoires

Cinq définitions (contradictoires) relevées par Gelman (2005).

Aparté : effets fixes et effets aléatoires

Cinq définitions (contradictoires) relevées par Gelman (2005).

- Fixed effects are constant across individuals, and random effects vary.

Aparté : effets fixes et effets aléatoires

Cinq définitions (contradictoires) relevées par Gelman (2005).

- Fixed effects are constant across individuals, and random effects vary.
- Effects are fixed if they are interesting in themselves or random if there is interest in the underlying population.

Aparté : effets fixes et effets aléatoires

Cinq définitions (contradictoires) relevées par Gelman (2005).

- Fixed effects are constant across individuals, and random effects vary.
- Effects are fixed if they are interesting in themselves or random if there is interest in the underlying population.
- When a sample exhausts the population, the corresponding variable is fixed; when the sample is a small (i.e., negligible) part of the population the corresponding variable is random.

Aparté : effets fixes et effets aléatoires

Cinq définitions (contradictoires) relevées par Gelman (2005).

- Fixed effects are constant across individuals, and random effects vary.
- Effects are fixed if they are interesting in themselves or random if there is interest in the underlying population.
- When a sample exhausts the population, the corresponding variable is fixed; when the sample is a small (i.e., negligible) part of the population the corresponding variable is random.
- If an effect is assumed to be a realized value of a random variable, it is called a random effect.

Aparté : effets fixes et effets aléatoires

Cinq définitions (contradictoires) relevées par Gelman (2005).

- Fixed effects are constant across individuals, and random effects vary.
- Effects are fixed if they are interesting in themselves or random if there is interest in the underlying population.
- When a sample exhausts the population, the corresponding variable is fixed; when the sample is a small (i.e., negligible) part of the population the corresponding variable is random.
- If an effect is assumed to be a realized value of a random variable, it is called a random effect.
- Fixed effects are estimated using least squares (or, more generally, maximum likelihood) and random effects are estimated with shrinkage.

Aparté : effets fixes et effets aléatoires

Cinq définitions (contradictoires) relevées par Gelman (2005).

- Fixed effects are constant across individuals, and random effects vary.
- Effects are fixed if they are interesting in themselves or random if there is interest in the underlying population.
- When a sample exhausts the population, the corresponding variable is fixed; when the sample is a small (i.e., negligible) part of the population the corresponding variable is random.
- If an effect is assumed to be a realized value of a random variable, it is called a random effect.
- Fixed effects are estimated using least squares (or, more generally, maximum likelihood) and random effects are estimated with shrinkage.

Gelman & Hill (2007) suggèrent plutôt l'utilisation des termes de *constant effects* et *varying effects*, et de toujours utiliser la modèlisation multi-niveaux, en considérant que ce qu'on appelle *effet fixe* peut simplement être considéré comme un *effet aléatoire* dont la variance serait égale à 0.

Régularisation et terminologie

Le fait de faire varier les intercepts de chaque café est simplement une autre manière de régulariser (de manière adaptative), c'est à dire de diminuer le poids accordé aux données dans l'estimation. Le modèle devient à même d'estimer à quel point les groupes (ici les cafés) sont différents, tout en estimant les caractéristiques de chaque café…

Régularisation et terminologie

Le fait de faire varier les intercepts de chaque café est simplement une autre manière de régulariser (de manière adaptative), c'est à dire de diminuer le poids accordé aux données dans l'estimation. Le modèle devient à même d'estimer à quel point les groupes (ici les cafés) sont différents, tout en estimant les caractéristiques de chaque café…

Différence entre les **cross-classified** (ou *crossed*) multilevel models et **nested or hierarchical** multilevel models. Le premier type de modèle concerne des données structurées selon deux (ou plus) facteurs aléatoires non *nichés*. Le deuxième type de modèles concerne des données structurées de manière hiérarchique (e.g., un élève dans une classe dans une école dans une ville…). Voir ce thread pour plus de détails.

Régularisation et terminologie

Le fait de faire varier les intercepts de chaque café est simplement une autre manière de régulariser (de manière adaptative), c'est à dire de diminuer le poids accordé aux données dans l'estimation. Le modèle devient à même d'estimer à quel point les groupes (ici les cafés) sont différents, tout en estimant les caractéristiques de chaque café...

Différence entre les **cross-classified** (ou *crossed*) multilevel models et **nested or hierarchical** multilevel models. Le premier type de modèle concerne des données structurées selon deux (ou plus) facteurs aléatoires non *nichés*. Le deuxième type de modèles concerne des données structurées de manière hiérarchique (e.g., un élève dans une classe dans une école dans une ville...). Voir ce thread pour plus de détails.

Les deux types de modèles s'écrivent cependant de manière similaire, sur plusieurs *niveaux*. Le terme “multi-niveaux” (dans notre terminologie) fait donc référence à la structure du modèle, à sa spécification. À distinguer de la structure des données.

Exemple de modèle "cross-classified"

On pourrait se poser la question de savoir si la récence des cafés (leur âge) ne serait pas une source de variabilité non contrôlée ? Il suffit d'ajouter un intercept qui varie par âge, et de lui attribuer un prior adaptatif.

$$
\begin{aligned}
w_i &\sim \text{Normal}(\mu_i, \sigma) \\
\mu_i &= \alpha + \alpha_{\text{café}[i]} + \alpha_{\text{âge}[i]} \\
\alpha_{\text{café}} &\sim \text{Normal}(5, \sigma_{\text{café}}) \\
\alpha_{\text{âge}} &\sim \text{Normal}(5, \sigma_{\text{âge}}) \\
\alpha &\sim \text{Normal}(0, 10) \\
\sigma_{\text{café}} &\sim \text{HalfCauchy}(0, 2) \\
\sigma_{\text{âge}} &\sim \text{HalfCauchy}(0, 2) \\
\sigma &\sim \text{HalfCauchy}(0, 2)
\end{aligned}
$$

Robot et café : varying intercept + varying slope

On s'intéresse maintenant à l'effet du moment de la journée sur le temps d'attente. Attend-on plus le matin, ou l'après-midi ?

Robot et café : varying intercept + varying slope

On s'intéresse maintenant à l'effet du moment de la journée sur le temps d'attente. Attend-on plus le matin, ou l'après-midi ?

$$
\begin{aligned}
w_i &\sim \mathrm{Normal}(\mu_i, \sigma) \\
\mu_i &= \alpha_{\text{café}[i]} + \beta_{\text{café}[i]} A_i
\end{aligned}
$$

Robot et café : varying intercept + varying slope

On s'intéresse maintenant à l'effet du moment de la journée sur le temps d'attente. Attend-on plus le matin, ou l'après-midi ?

$$
\begin{aligned}
w_i &\sim \mathrm{Normal}(\mu_i, \sigma) \\
\mu_i &= \alpha_{\text{café}[i]} + \beta_{\text{café}[i]} A_i
\end{aligned}
$$

Où A_i est une *dummy variable* codée 0/1 pour le matin et l'après-midi et $\beta_{\text{café}}$ est un paramètre de différence entre le matin et l'après-midi.

Robot et café : varying intercept + varying slope

On s'intéresse maintenant à l'effet du moment de la journée sur le temps d'attente. Attend-on plus le matin, ou l'après-midi ?

$$
\begin{aligned}
w_i &\sim \mathrm{Normal}(\mu_i, \sigma) \\
\mu_i &= \alpha_{\text{café}[i]} + \beta_{\text{café}[i]} A_i
\end{aligned}
$$

Où A_i est une *dummy variable* codée 0/1 pour le matin et l'après-midi et $\beta_{\text{café}}$ est un paramètre de différence entre le matin et l'après-midi.

Remarque : on sait que les cafés ont des intercepts et des pentes qui covarient… Les cafés populaires seront surchargés le matin et beaucoup moins l'après-midi, résultant en une pente importante. Ces cafés auront aussi un temps d'attente moyen plus long (i.e., un intercept plus grand). Dans ces cafés, α est grand et β est loin de zéro. À l'inverse, dans un café peu populaire, le temps d'attente sera faible, ainsi que la différence entre matin et après-midi.

Robot et café : varying intercept + varying slope

On s'intéresse maintenant à l'effet du moment de la journée sur le temps d'attente. Attend-on plus le matin, ou l'après-midi ?

$$
\begin{aligned}
w_i &\sim \text{Normal}(\mu_i, \sigma) \\
\mu_i &= \alpha_{\text{café}[i]} + \beta_{\text{café}[i]} A_i
\end{aligned}
$$

Où A_i est une *dummy variable* codée 0/1 pour le matin et l'après-midi et $\beta_{\text{café}}$ est un paramètre de différence entre le matin et l'après-midi.

Remarque : on sait que les cafés ont des intercepts et des pentes qui covarient… Les cafés populaires seront surchargés le matin et beaucoup moins l'après-midi, résultant en une pente importante. Ces cafés auront aussi un temps d'attente moyen plus long (i.e., un intercept plus grand). Dans ces cafés, α est grand et β est loin de zéro. À l'inverse, dans un café peu populaire, le temps d'attente sera faible, ainsi que la différence entre matin et après-midi.

On pourrait donc utiliser la co-variation entre intercept et pente pour faire de meilleures inférences. Autrement dit, faire en sorte que l'estimation de l'intercept informe celle de la pente, et réciproquement.

Robot et café : varying intercept + varying slope

On s'intéresse maintenant à l'effet du moment de la journée sur le temps d'attente. Attend-on plus le matin, ou l'après-midi ?

Robot et café : varying intercept + varying slope

On s'intéresse maintenant à l'effet du moment de la journée sur le temps d'attente. Attend-on plus le matin, ou l'après-midi ?

$$
\begin{aligned}
w_i &\sim \mathrm{Normal}(\mu_i, \sigma) \\
\mu_i &= \alpha_{\text{café}[i]} + \beta_{\text{café}[i]} A_i \\
\begin{bmatrix} \alpha_{\text{café}} \\ \beta_{\text{café}} \end{bmatrix} &\sim \mathrm{MVNormal}\left(\begin{bmatrix} \alpha \\ \beta \end{bmatrix}, \mathbf{S} \right)
\end{aligned}
$$

Robot et café : varying intercept + varying slope

On s'intéresse maintenant à l'effet du moment de la journée sur le temps d'attente. Attend-on plus le matin, ou l'après-midi ?

$$
\begin{aligned}
w_i &\sim \mathrm{Normal}(\mu_i, \sigma) \\
\mu_i &= \alpha_{\text{café}[i]} + \beta_{\text{café}[i]} A_i \\
\begin{bmatrix} \alpha_{\text{café}} \\ \beta_{\text{café}} \end{bmatrix} &\sim \mathrm{MVNormal}\left(\begin{bmatrix} \alpha \\ \beta \end{bmatrix}, \mathbf{S} \right)
\end{aligned}
$$

La troisième ligne postule que chaque café a un intercept $\alpha_{\text{café}}$ et une pente $\beta_{\text{café}}$, définis par un prior Gaussien bivarié (i.e., à deux dimensions) ayant comme moyennes α et β et comme matrice de covariance $\mathbf{S}$.

Aparté : distribution gaussienne multivariée

$$\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

Aparté : distribution gaussienne multivariée

$$\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

Où $\boldsymbol{\mu}$ est un vecteur (à k dimensions) de moyennes, par exemple: `mu <- c(a, b)`.

Aparté : distribution gaussienne multivariée

$$\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

Où $\boldsymbol{\mu}$ est un vecteur (à k dimensions) de moyennes, par exemple: `mu <- c(a, b)`.

$\boldsymbol{\Sigma}$ est une matrice de covariance de $k \times k$ dimensions, et qui correspond à la matrice donnée par la fonction `vcov()`.

Aparté : distribution gaussienne multivariée

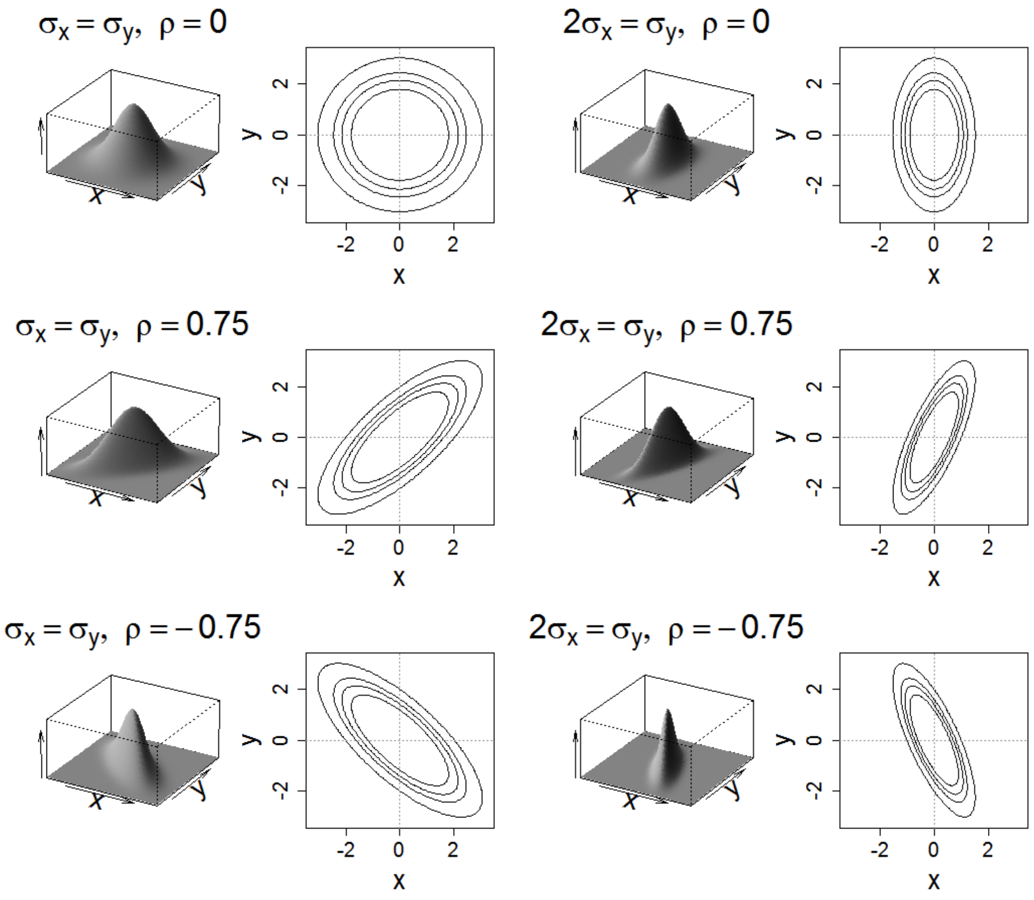
$$\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

Où $\boldsymbol{\mu}$ est un vecteur (à k dimensions) de moyennes, par exemple: `mu <- c(a, b)`.

$\boldsymbol{\Sigma}$ est une matrice de covariance de $k \times k$ dimensions, et qui correspond à la matrice donnée par la fonction `vcov()`.

$$\boldsymbol{\Sigma} = \begin{pmatrix} \sigma_{\alpha}^{2} & \sigma_{\alpha}\sigma_{\beta}\rho \\ \sigma_{\alpha}\sigma_{\beta}\rho & \sigma_{\beta}^{2} \end{pmatrix}$$

Aparté : distribution gaussienne multivariée



Aparté : distribution gaussienne multivariée, calcul matriciel

$$\boldsymbol{\Sigma} = \begin{pmatrix} \sigma_\alpha^2 & \sigma_\alpha \sigma_\beta \rho \\ \sigma_\alpha \sigma_\beta \rho & \sigma_\beta^2 \end{pmatrix}$$

Aparté : distribution gaussienne multivariée, calcul matriciel

$$\boldsymbol{\Sigma} = \begin{pmatrix} \sigma_\alpha^2 & \sigma_\alpha \sigma_\beta \rho \\ \sigma_\alpha \sigma_\beta \rho & \sigma_\beta^2 \end{pmatrix}$$

Cette matrice peut se construire de deux manières différentes, strictement équivalentes.

Aparté : distribution gaussienne multivariée, calcul matriciel

$$\boldsymbol{\Sigma} = \begin{pmatrix} \sigma_{\alpha}^{2} & \sigma_{\alpha}\sigma_{\beta}\rho \\ \sigma_{\alpha}\sigma_{\beta}\rho & \sigma_{\beta}^{2} \end{pmatrix}$$

Cette matrice peut se construire de deux manières différentes, strictement équivalentes.

```
sigma_a <- 1
sigma_b <- 0.75
rho <- 0.7
cov_ab <- sigma_a * sigma_b * rho
(Sigma1 <- matrix(c(sigma_a^2, cov_ab, cov_ab, sigma_b^2), ncol = 2) )
```

Aparté : distribution gaussienne multivariée, calcul matriciel

$$\boldsymbol{\Sigma} = \begin{pmatrix} \sigma_\alpha^2 & \sigma_\alpha \sigma_\beta \rho \\ \sigma_\alpha \sigma_\beta \rho & \sigma_\beta^2 \end{pmatrix}$$

Cette matrice peut se construire de deux manières différentes, strictement équivalentes.

```
sigma_a <- 1
sigma_b <- 0.75
rho <- 0.7
cov_ab <- sigma_a * sigma_b * rho
(Sigma1 <- matrix(c(sigma_a^2, cov_ab, cov_ab, sigma_b^2), ncol = 2) )
```

```
      [,1]   [,2]
[1,] 1.000 0.5250
[2,] 0.525 0.5625
```

Aparté : distribution gaussienne multivariée, calcul matriciel

$$\mathbf{\Sigma} = \begin{pmatrix} \sigma_\alpha^2 & \sigma_\alpha \sigma_\beta \rho \\ \sigma_\alpha \sigma_\beta \rho & \sigma_\beta^2 \end{pmatrix}$$

Aparté : distribution gaussienne multivariée, calcul matriciel

$$\mathbf{\Sigma} = \begin{pmatrix} \sigma_\alpha^2 & \sigma_\alpha \sigma_\beta \rho \\ \sigma_\alpha \sigma_\beta \rho & \sigma_\beta^2 \end{pmatrix}$$

La deuxième méthode est assez utile puisqu'elle considère séparément les écart-types et les corrélations.

Aparté : distribution gaussienne multivariée, calcul matriciel

$$
\boldsymbol{\Sigma} = \begin{pmatrix} \sigma_{\alpha}^{2} & \sigma_{\alpha}\sigma_{\beta}\rho \\ \sigma_{\alpha}\sigma_{\beta}\rho & \sigma_{\beta}^{2} \end{pmatrix}
$$

La deuxième méthode est assez utile puisqu'elle considère séparément les écart-types et les corrélations.

```
(sigmas <- c(sigma_a, sigma_b) ) # standard deviations
```

Aparté : distribution gaussienne multivariée, calcul matriciel

$$\mathbf{\Sigma} = \begin{pmatrix} \sigma_\alpha^2 & \sigma_\alpha\sigma_\beta\rho \\ \sigma_\alpha\sigma_\beta\rho & \sigma_\beta^2 \end{pmatrix}$$

La deuxième méthode est assez utile puisqu'elle considère séparément les écart-types et les corrélations.

```
(sigmas <- c(sigma_a, sigma_b) ) # standard deviations
```

```
[1] 1.00 0.75
```

Aparté : distribution gaussienne multivariée, calcul matriciel

$$\mathbf{\Sigma} = \begin{pmatrix} \sigma_\alpha^2 & \sigma_\alpha \sigma_\beta \rho \\ \sigma_\alpha \sigma_\beta \rho & \sigma_\beta^2 \end{pmatrix}$$

La deuxième méthode est assez utile puisqu'elle considère séparément les écart-types et les corrélations.

```r
(sigmas <- c(sigma_a, sigma_b) ) # standard deviations
```

```
[1] 1.00 0.75
```

```r
(Rho <- matrix(c(1, rho, rho, 1), nrow = 2) ) # correlation matrix
```

Aparté : distribution gaussienne multivariée, calcul matriciel

$$\mathbf{\Sigma} = \begin{pmatrix} \sigma_\alpha^2 & \sigma_\alpha \sigma_\beta \rho \\ \sigma_\alpha \sigma_\beta \rho & \sigma_\beta^2 \end{pmatrix}$$

La deuxième méthode est assez utile puisqu'elle considère séparément les écart-types et les corrélations.

```r
(sigmas <- c(sigma_a, sigma_b) ) # standard deviations
```

```
[1] 1.00 0.75
```

```r
(Rho <- matrix(c(1, rho, rho, 1), nrow = 2) ) # correlation matrix
```

```
     [,1] [,2]
[1,]  1.0  0.7
[2,]  0.7  1.0
```

Aparté : distribution gaussienne multivariée, calcul matriciel

$$\mathbf{\Sigma} = \begin{pmatrix} \sigma_\alpha^2 & \sigma_\alpha \sigma_\beta \rho \\ \sigma_\alpha \sigma_\beta \rho & \sigma_\beta^2 \end{pmatrix}$$

La deuxième méthode est assez utile puisqu'elle considère séparément les écart-types et les corrélations.

```
(sigmas <- c(sigma_a, sigma_b) ) # standard deviations
```

```
[1] 1.00 0.75
```

```
(Rho <- matrix(c(1, rho, rho, 1), nrow = 2) ) # correlation matrix
```

```
     [,1] [,2]
[1,]  1.0  0.7
[2,]  0.7  1.0
```

```
(Sigma2 <- diag(sigmas) %*% Rho %*% diag(sigmas) )
```

Aparté : distribution gaussienne multivariée, calcul matriciel

$$\boldsymbol{\Sigma} = \begin{pmatrix} \sigma_{\alpha}^{2} & \sigma_{\alpha}\sigma_{\beta}\rho \\ \sigma_{\alpha}\sigma_{\beta}\rho & \sigma_{\beta}^{2} \end{pmatrix}$$

La deuxième méthode est assez utile puisqu'elle considère séparément les écart-types et les corrélations.

```r
(sigmas <- c(sigma_a, sigma_b) ) # standard deviations
```

```
[1] 1.00 0.75
```

```r
(Rho <- matrix(c(1, rho, rho, 1), nrow = 2) ) # correlation matrix
```

```
     [,1] [,2]
[1,]  1.0  0.7
[2,]  0.7  1.0
```

```r
(Sigma2 <- diag(sigmas) %*% Rho %*% diag(sigmas) )
```

```
      [,1]   [,2]
[1,] 1.000 0.5250
[2,] 0.525 0.5625
```

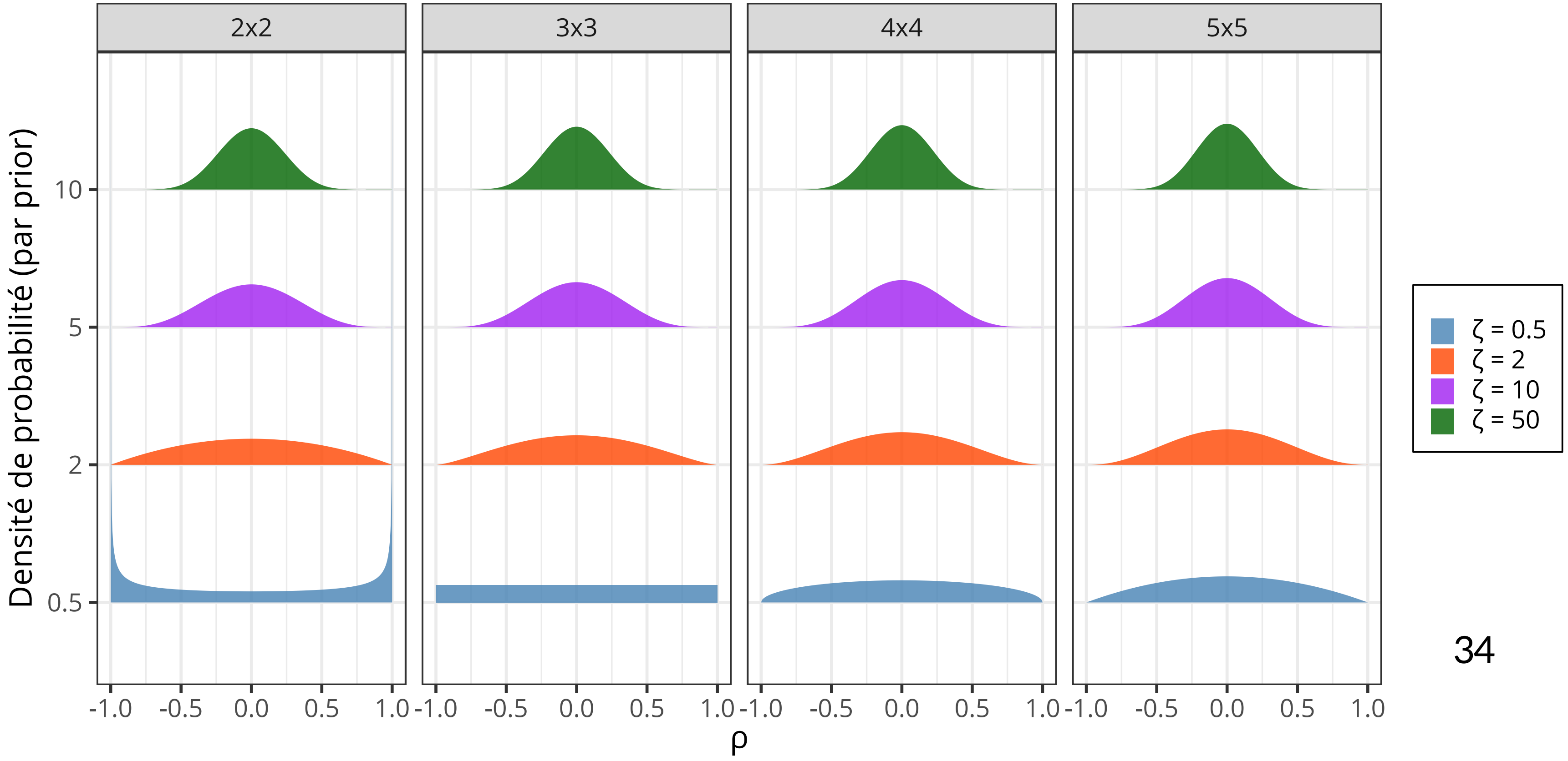
Robot et café : varying intercept + varying slope

$$
\begin{aligned}
w_i &\sim \text{Normal}(\mu_i, \sigma) \\
\mu_i &= \alpha_{\text{café}[i]} + \beta_{\text{café}[i]} A_i \\
\begin{bmatrix} \alpha_{\text{café}} \\ \beta_{\text{café}} \end{bmatrix} &\sim \text{MVNormal}\left(\begin{bmatrix} \alpha \\ \beta \end{bmatrix}, \mathbf{S} \right) \\
\mathbf{S} &= \begin{pmatrix} \sigma_\alpha & 0 \\ 0 & \sigma_\beta \end{pmatrix} \mathbf{R} \begin{pmatrix} \sigma_\alpha & 0 \\ 0 & \sigma_\beta \end{pmatrix} \\
\alpha &\sim \text{Normal}(0, 10) \\
\beta &\sim \text{Normal}(0, 10) \\
\sigma &\sim \text{HalfCauchy}(0, 1) \\
\sigma_\alpha &\sim \text{HalfCauchy}(0, 1) \\
\sigma_\beta &\sim \text{HalfCauchy}(0, 1) \\
\mathbf{R} &\sim \text{LKJ}(2)
\end{aligned}
$$

$\mathbf{S}$ est définie en factorisant $\sigma_\alpha, \sigma_\beta$, et la matrice de corrélation $\mathbf{R}$. La suite du modèle définit simplement les priors pour les effets constants. La dernière ligne spécifie le prior pour $\mathbf{R}$.

LKJ prior

D'après Lewandowski, Kurowicka, & Joe (2009). Un seul paramètre ζ spécifie la concentration de la distribution du coefficient de corrélation. Le prior $\mathrm{LKJ}(2)$ définit un prior peu informatif pour ρ qui est sceptique des corrélations extrêmes (i.e., proches de -1 ou 1).



Rappels de syntaxe

Le package `brms` utilise la même syntaxe que les fonctions de base R (comme `lm`) ou que le package `lme4`.

Rappels de syntaxe

Le package `brms` utilise la même syntaxe que les fonctions de base R (comme `lm`) ou que le package `lme4`.

```
Reaction ~ Days + (1 + Days | Subject)
```

Rappels de syntaxe

Le package `brms` utilise la même syntaxe que les fonctions de base R (comme `lm`) ou que le package `lme4`.

```
Reaction ~ Days + (1 + Days | Subject)
```

La partie gauche représente notre variable dépendante (ou *outcome*, i.e., ce qu'on essaye de prédire).

Rappels de syntaxe

Le package `brms` utilise la même syntaxe que les fonctions de base R (comme `lm`) ou que le package `lme4`.

```
Reaction ~ Days + (1 + Days | Subject)
```

La partie gauche représente notre variable dépendante (ou *outcome*, i.e., ce qu'on essaye de prédire).

La partie droite permet de définir les prédicteurs. L'intercept est généralement implicite, de sorte que les deux écritures ci-dessous sont équivalentes.

Rappels de syntaxe

Le package `brms` utilise la même syntaxe que les fonctions de base R (comme `lm`) ou que le package `lme4`.

```
Reaction ~ Days + (1 + Days | Subject)
```

La partie gauche représente notre variable dépendante (ou *outcome*, i.e., ce qu'on essaye de prédire).

La partie droite permet de définir les prédicteurs. L'intercept est généralement implicite, de sorte que les deux écritures ci-dessous sont équivalentes.

```
c(Reaction, Memory) ~ Days + (1 + Days | Subject)
c(Reaction, Memory) ~ 1 + Days + (1 + Days | Subject)
```

Rappels de syntaxe

La première partie de la partie droite de la formule représente les effets constants (effets fixes), tandis que la seconde partie (entre parenthèses) représente les effets variables (effets aléatoires).

Rappels de syntaxe

La première partie de la partie droite de la formule représente les effets constants (effets fixes), tandis que la seconde partie (entre parenthèses) représente les effets variables (effets aléatoires).

```
c(Reaction, Memory) ~ Days + (1 | Subject)
c(Reaction, Memory) ~ Days + (Days | Subject)
```

Rappels de syntaxe

La première partie de la partie droite de la formule représente les effets constants (effets fixes), tandis que la seconde partie (entre parenthèses) représente les effets variables (effets aléatoires).

```
c(Reaction, Memory) ~ Days + (1 | Subject)
c(Reaction, Memory) ~ Days + (Days | Subject)
```

Le premier modèle ci-dessus contient seulement un intercept variable, qui varie par `Subject`. Le deuxième modèle contient également un intercept variable, mais aussi une pente variable pour l'effet de `Days`.

Rappels de syntaxe

Lorsqu'on inclut plusieurs effets variables (e.g., un intercept et une pente variables), `brms` postule qu'on souhaite aussi estimer la corrélation entre ces deux effets. Dans le cas contraire, on peut supprimer cette corrélation (i.e., la fixer à 0) en utilisant `||`.

Rappels de syntaxe

Lorsqu'on inclut plusieurs effets variables (e.g., un intercept et une pente variables), `brms` postule qu'on souhaite aussi estimer la corrélation entre ces deux effets. Dans le cas contraire, on peut supprimer cette corrélation (i.e., la fixer à 0) en utilisant `||`.

```
c(Reaction, Memory) ~ Days + (1 + Days || Subject)
```

Rappels de syntaxe

Lorsqu'on inclut plusieurs effets variables (e.g., un intercept et une pente variables), `brms` postule qu'on souhaite aussi estimer la corrélation entre ces deux effets. Dans le cas contraire, on peut supprimer cette corrélation (i.e., la fixer à 0) en utilisant `||`.

```
c(Reaction, Memory) ~ Days + (1 + Days || Subject)
```

Les modèles précédents postulaient un modèle génératif Gaussien. Ce postulat peut être changé facilement en spécifiant la fonction souhaitée via l'argument `family`.

Rappels de syntaxe

Lorsqu'on inclut plusieurs effets variables (e.g., un intercept et une pente variables), `brms` postule qu'on souhaite aussi estimer la corrélation entre ces deux effets. Dans le cas contraire, on peut supprimer cette corrélation (i.e., la fixer à 0) en utilisant `||`.

```
c(Reaction, Memory) ~ Days + (1 + Days || Subject)
```

Les modèles précédents postulaient un modèle génératif Gaussien. Ce postulat peut être changé facilement en spécifiant la fonction souhaitée via l'argument `family`.

```
brm(Reaction ~ 1 + Days + (1 + Days | Subject), family = lognormal() )
```

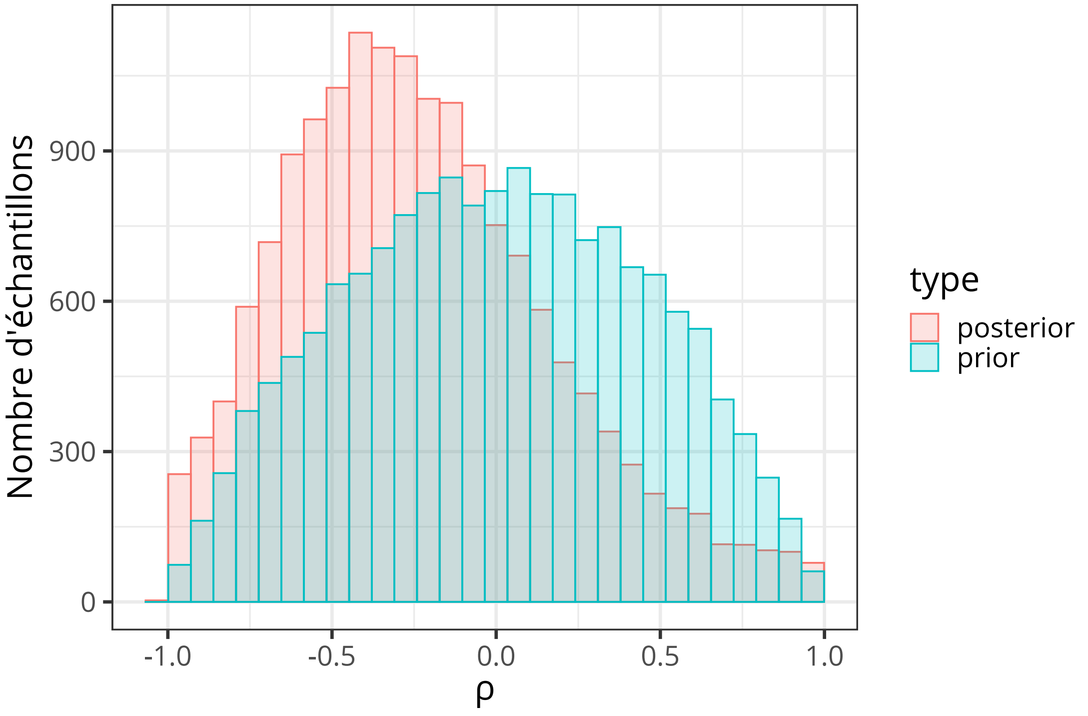
Modèle brms

On spécifie un intercept et une pente (pour l'effet d'`afternoon`) qui varient par `cafe`.

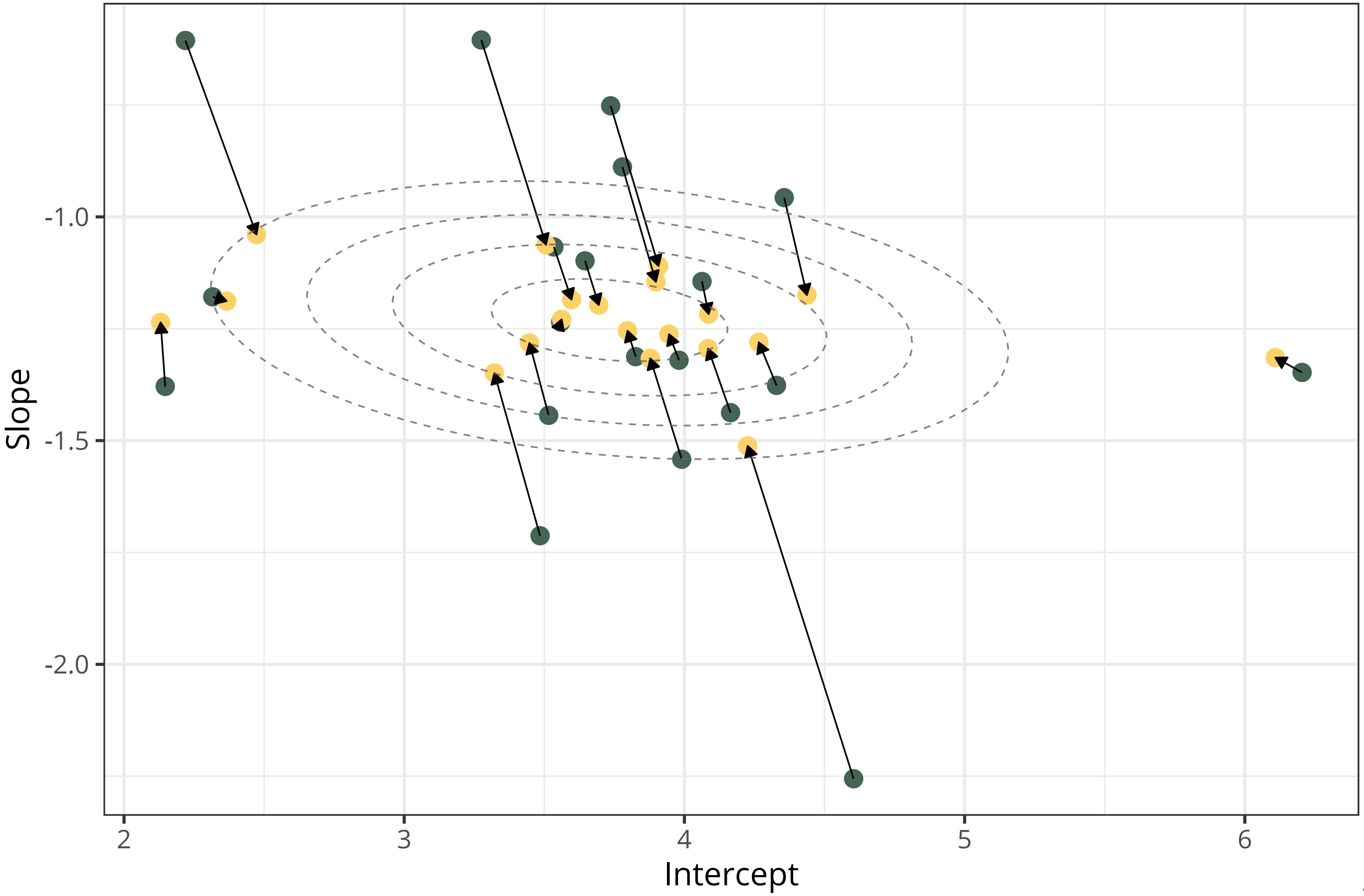
```r
mod5 <- brm(
  wait ~ 1 + afternoon + (1 + afternoon | cafe),
  prior = c(
    set_prior("normal(0, 10)", class = "Intercept"),
    set_prior("normal(0, 10)", class = "b"),
    set_prior("cauchy(0, 2)", class = "sigma"),
    set_prior("cauchy(0, 2)", class = "sd")
    ),
  data = df,
  warmup = 1000, iter = 5000,
  cores = parallel::detectCores()
  )
```

Distribution postérieure

```r
post <- posterior_samples(mod5) # extracts posterior samples
R <- rlkjcorr(16000, K = 2, eta = 2) # samples from prior

data.frame(prior = R[, 1, 2], posterior = post$cor_cafe__Intercept__afternoon) %>%
    gather(type, value, prior:posterior) %>%
    ggplot(aes(x = value, color = type, fill = type) ) +
    geom_histogram(position = "identity", alpha = 0.2) +
    labs(x = expression(rho), y = "Nombre d'échantillons")
```



Shrinkage en deux dimensions



Comparaison de modèles

On compare le premier modèle (*complete pooling model*), le troisième modèle (*partial pooling model*), et le dernier modèle.

```r
# comparaison des WAIC de chaque modèle
mod5 <- add_criterion(mod5, "waic")
w <- loo_compare(mod1, mod2, mod3, mod5, criterion = "waic")
print(w, simplify = FALSE)
```

```
     elpd_diff se_diff elpd_waic se_elpd_waic p_waic se_p_waic waic   se_waic
mod5    0.0       0.0  -155.2      10.1         26.6    2.6     310.5   20.1
mod3  -98.6       8.3  -253.9       8.3         18.2    1.5     507.7   16.6
mod2  -99.5       8.2  -254.7       8.4         19.7    1.6     509.4   16.9
mod1 -155.9      13.7  -311.1      10.6          2.1    0.4     622.2   21.1
```

```r
model_weights(mod1, mod2, mod3, mod5, weights = "waic")
```

```
        mod1         mod2         mod3         mod5 
1.987080e-68 6.215945e-44 1.470144e-43 1.000000e+00 
```

Comparaison de modèles

L'estimation du temps d'attente moyen est plus incertaine lorsqu'on prend en compte de nouvelles sources d'erreur. Cependant, l'erreur du modèle (i.e., ce qui n'est pas expliqué), la variation résiduelle σ, diminue…

```
posterior_summary(mod1, pars = c("^b", "sigma") )
```

```
            Estimate  Est.Error     Q2.5    Q97.5
b_Intercept 3.120812 0.07842501 2.966944 3.276102
sigma       1.141664 0.05816792 1.035442 1.261113
```

```
posterior_summary(mod3, pars = c("^b", "sigma") )
```

```
             Estimate  Est.Error      Q2.5     Q97.5
b_Intercept 3.1232197 0.20418306 2.7166225 3.5240819
sigma       0.8219111 0.04373907 0.7418185 0.9134082
```

```
posterior_summary(mod5, pars = c("^b", "sigma") )
```

```
              Estimate  Est.Error       Q2.5      Q97.5
b_Intercept  3.7329518 0.21447305  3.3125024  4.1630121
b_afternoon -1.2307534 0.08702525 -1.4013234 -1.0590520
sigma        0.4898756 0.02764222  0.4388531  0.5461183
```

Conclusions

Les modèles multi-niveaux (ou "modèles mixtes") sont des extensions naturelles des modèles de régression classiques, où les paramètres de ces derniers se voient eux-même attribués des "modèles", gouvernés par des hyper-paramètres.

Conclusions

Les modèles multi-niveaux (ou "modèles mixtes") sont des extensions naturelles des modèles de régression classiques, où les paramètres de ces derniers se voient eux-même attribués des "modèles", gouvernés par des hyper-paramètres.

Cette extension permet de faire des prédictions plus précises en prenant en compte la variabilité liée aux groupes ou structures (clusters) présent(e)s dans les données. Autrement dit, en modélisant les populations d'où sont tirés les effets aléatoires (e.g., la population de participants ou de stimuli).

Conclusions

Les modèles multi-niveaux (ou “modèles mixtes”) sont des extensions naturelles des modèles de régression classiques, où les paramètres de ces derniers se voient eux-même attribués des “modèles”, gouvernés par des hyper-paramètres.

Cette extension permet de faire des prédictions plus précises en prenant en compte la variabilité liée aux groupes ou structures (clusters) présent(e)s dans les données. Autrement dit, en modélisant les populations d'où sont tirés les effets aléatoires (e.g., la population de participants ou de stimuli).

Un modèle de régression classique est équivalent à un modèle multi-niveaux où la variabilité des effets aléatoires serait fixée à 0.

Conclusions

Les modèles multi-niveaux (ou "modèles mixtes") sont des extensions naturelles des modèles de régression classiques, où les paramètres de ces derniers se voient eux-même attribués des "modèles", gouvernés par des hyper-paramètres.

Cette extension permet de faire des prédictions plus précises en prenant en compte la variabilité liée aux groupes ou structures (clusters) présent(e)s dans les données. Autrement dit, en modélisant les populations d'où sont tirés les effets aléatoires (e.g., la population de participants ou de stimuli).

Un modèle de régression classique est équivalent à un modèle multi-niveaux où la variabilité des effets aléatoires serait fixée à 0.

La cadre bayésien permet une interprétation naturelle des distributions desquelles proviennent les effets aléatoires (varying effects). En effet, ces distributions peuvent être interprétées comme des distributions a priori, dont les paramètres sont estimés à partir des données.

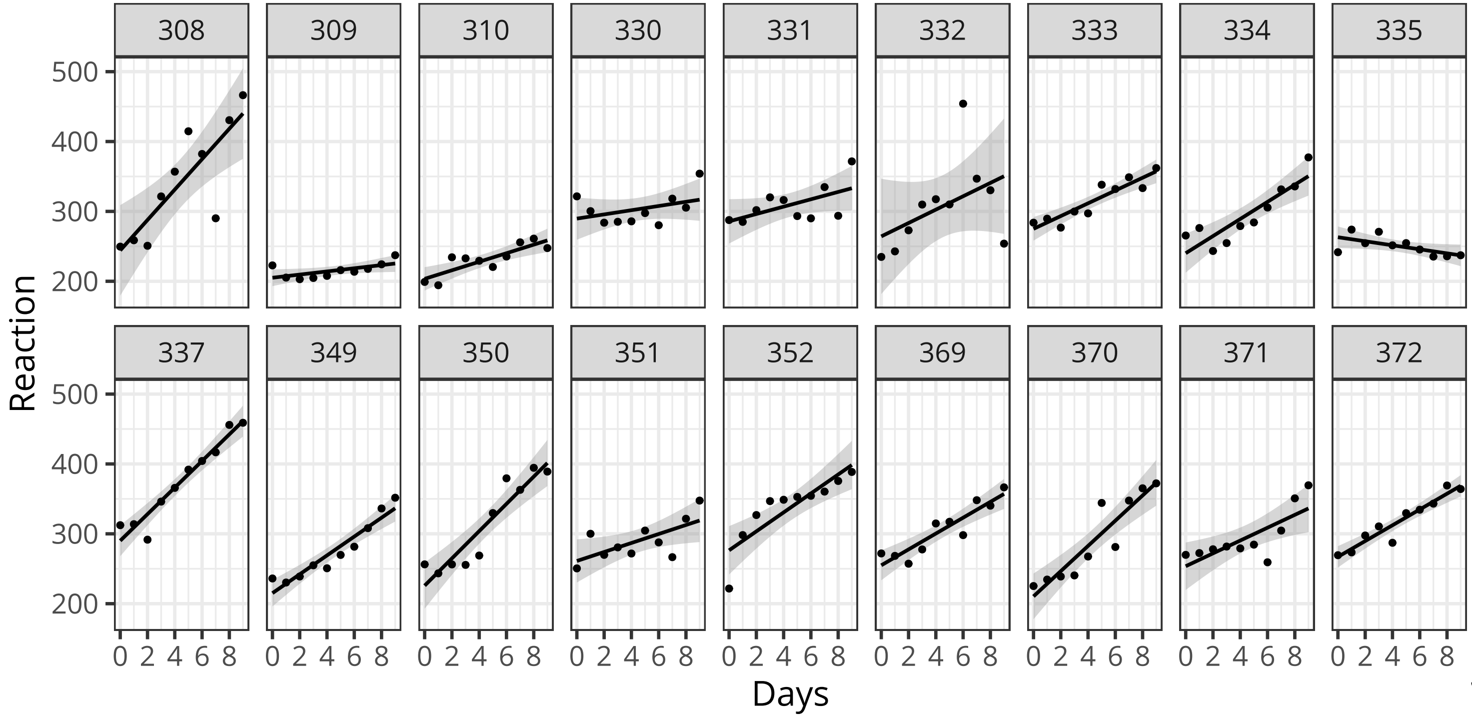
Travaux pratiques - sleepstudy

```r
library(lme4)

data(sleepstudy)
head(sleepstudy, 20)
```

```
   Reaction Days Subject
1  249.5600    0     308
2  258.7047    1     308
3  250.8006    2     308
4  321.4398    3     308
5  356.8519    4     308
6  414.6901    5     308
7  382.2038    6     308
8  290.1486    7     308
9  430.5853    8     308
10 466.3535    9     308
11 222.7339    0     309
12 205.2658    1     309
13 202.9778    2     309
14 204.7070    3     309
15 207.7161    4     309
16 215.9618    5     309
17 213.6303    6     309
18 217.7272    7     309
19 224.2957    8     309
20 237.3142    9     309
```

Travaux pratiques - sleepstudy

```
sleepstudy %>%
    ggplot(aes(x = Days, y = Reaction) ) +
    geom_smooth(method = "lm", colour = "black") +
    geom_point() +
    facet_wrap(~Subject, nrow = 2) +
    scale_x_continuous(breaks = c(0, 2, 4, 6, 8) )
```



Travaux pratiques - sleepstudy

À vous de construire les modèles mathématiques et les modèles `brms` correspondant aux modèles suivants :

- Modèle avec seulement l'effet fixe de `Days`
- Modèle avec l'effet fixe de `Days` + un effet aléatoire de `Subject` (*varying intercept*)
- Modèle avec l'effet fixe de `Days` + un effet aléatoire de `Subject` (*varying intercept*) + un effet aléatoire de `Days` (*varying slope*)

Comparez ensuite ces modèles en utilisant les outils discutés aux cours précédents (e.g., WAIC) et concluez.

Proposition de solution

```
fmod0 <- lm(Reaction ~ Days, sleepstudy)
fmod1 <- lmer(Reaction ~ Days + (1|Subject), sleepstudy)
fmod2 <- lmer(Reaction ~ Days + (1 + Days|Subject), sleepstudy)

anova(fmod1, fmod2)
```

```
Data: sleepstudy
Models:
fmod1: Reaction ~ Days + (1 | Subject)
fmod2: Reaction ~ Days + (1 + Days | Subject)
      npar    AIC    BIC  logLik deviance  Chisq Df Pr(>Chisq)
fmod1    4 1802.1 1814.8 -897.04   1794.1
fmod2    6 1763.9 1783.1 -875.97   1751.9 42.139  2  7.072e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Proposition de solution

```r
mod6 <- brm(
  Reaction ~ 1 + Days,
  prior = c(
    set_prior("normal(200, 100)", class = "Intercept"),
    set_prior("normal(0, 10)", class = "b"),
    set_prior("cauchy(0, 10)", class = "sigma")
    ),
  data = sleepstudy,
  warmup = 1000, iter = 5000,
  cores = parallel::detectCores()
  )
```

```r
posterior_summary(mod6)
```

```
              Estimate Est.Error        Q2.5      Q97.5
b_Intercept  251.99787  6.633360  238.957797  265.00193
b_Days        10.30971  1.233762    7.878538   12.75686
sigma         47.79917  2.542302   43.087029   53.05137
lp__        -963.49135  1.250759 -966.768992 -962.06677
```

Proposition de solution

```
mod7 <- brm(
  Reaction ~ 1 + Days + (1 | Subject),
  prior = c(
    set_prior("normal(200, 100)", class = "Intercept"),
    set_prior("normal(0, 10)", class = "b"),
    set_prior("cauchy(0, 10)", class = "sigma"),
    set_prior("cauchy(0, 10)", class = "sd")
    ),
  data = sleepstudy,
  warmup = 1000, iter = 5000,
  cores = parallel::detectCores()
  )
```

```
posterior_summary(mod7, pars = c("^b", "sigma") )
```

```
             Estimate Est.Error      Q2.5     Q97.5
b_Intercept 250.61951 9.9846608 230.951638 269.92308
b_Days       10.40200 0.7944101   8.838617  11.97743
sigma        31.07502 1.7155642  27.933271  34.61628
```

Proposition de solution

```
mod8 <- brm(
  Reaction ~ 1 + Days + (1 + Days | Subject),
  prior = c(
    set_prior("normal(200, 100)", class = "Intercept"),
    set_prior("normal(0, 10)", class = "b"),
    set_prior("cauchy(0, 10)", class = "sigma"),
    set_prior("cauchy(0, 10)", class = "sd")
    ),
  data = sleepstudy,
  warmup = 1000, iter = 5000,
  cores = parallel::detectCores()
  )
```

```
posterior_summary(mod8, pars = c("^b", "sigma") )
```

```
             Estimate Est.Error       Q2.5     Q97.5
b_Intercept 250.96454  7.011060 237.177684 264.93840
b_Days       10.06192  1.694789   6.628644  13.33722
sigma        25.87619  1.567558  23.045501  29.20826
```

Proposition de solution

```r
# calcul du WAIC et ajout du WAIC à chaque modèle
mod6 <- add_criterion(mod6, "waic")
mod7 <- add_criterion(mod7, "waic")
mod8 <- add_criterion(mod8, "waic")

# comparaison des WAIC de chaque modèle
w <- loo_compare(mod6, mod7, mod8, criterion = "waic")
print(w, simplify = FALSE)
```

```
     elpd_diff se_diff elpd_waic se_elpd_waic p_waic se_p_waic waic   se_waic
mod8    0.0       0.0  -860.3      22.2         32.7    8.2    1720.5   44.5
mod7  -24.5      11.5  -884.7      14.4         19.3    3.3    1769.5   28.8
mod6  -93.1      20.9  -953.3      10.6          3.2    0.5    1906.6   21.1
```

```r
# calcul du poids relatif de chaque modèle
model_weights(mod6, mod7, mod8, weights = "waic")
```

```
        mod6         mod7         mod8
3.801195e-41 2.316319e-11 1.000000e+00
```

Proposition de solution

```r
# calcul du WAIC et ajout du WAIC à chaque modèle
mod6 <- add_criterion(mod6, "waic")
mod7 <- add_criterion(mod7, "waic")
mod8 <- add_criterion(mod8, "waic")

# comparaison des WAIC de chaque modèle
w <- loo_compare(mod6, mod7, mod8, criterion = "waic")
print(w, simplify = FALSE)
```

```
     elpd_diff se_diff elpd_waic se_elpd_waic p_waic se_p_waic waic   se_waic
mod8    0.0       0.0  -860.3      22.2         32.7    8.2    1720.5   44.5
mod7  -24.5      11.5  -884.7      14.4         19.3    3.3    1769.5   28.8
mod6  -93.1      20.9  -953.3      10.6          3.2    0.5    1906.6   21.1
```

```r
# calcul du poids relatif de chaque modèle
model_weights(mod6, mod7, mod8, weights = "waic")
```

```
        mod6         mod7         mod8
3.801195e-41 2.316319e-11 1.000000e+00
```