Começando na estimação bayesiana

Igor Costa

18/05/2022

```
require(ggplot2)
require(dplyr)
require(tidyr)
require(lme4)
```

Este material foi feito para os encontros do Laboratório de Psicolinguística e Aquisição da Linguagem - LAPAL/PUC-Rio. e pretendem ser uma introdução compreensiva sobre a lógica da inferência bayesiana, partindo na noção de função de verossimilhança, priors e a relação destas com a posterior.

Estimativas de máxima verossimilhança (*Maximum Likelihood Estimation* - MLE).

Começando com um exemplo... Partindo da distribuição binomial:

$P(x|n,p) = \binom{n}{k}p^k(1-p)^{n-k}$

Vamos usar essa para obter a probabilidade associada à ocorrência de um evento, digamos, ao lançar uma moeda não viciada 5 vezes, qual seria a probabilidade de obter 3 caras? Assumindo que $1 = cara$ e $0 = coroa$, então, há dez combinações possíveis, no lançamento de 5 moedas, que nos dão o resultado desejado, ou seja, 3 caras e 2 coroas:

$A = \{(1,1,1,0,0),(1,0,1,1,0),(1,0,0,1,1),(1,1,0,1,0),(1,1,0,0,1),(1,0,1,0,1),(0,1,1,1,0),$

$(0,0,1,1,1),(0,1,0,1,1),(0,1,1,0,1)\}$

No R, a função `choose()`nos dá o total de combinações possíveis:

```
choose(5, 3)
```

```
## [1] 10
```

Matematicamente, essa relação é grafada: $\binom{5}{3}$ e pode ser expressa genericamente pela equação:

$\binom{n}{k} = \frac{n!}{k!(n-k)!}$

Para os dados acima, poderíamos simplesmente inseri-los na fórmula:

$\binom{5}{3} = \frac{5!}{3!(5-3)!} = \frac{5!}{3!2!} = \frac{5\times4\times3\times2}{3\times2\times2} = \frac{120}{12} = 10$

Como dissemos que a moeda não é viciada, então há exatamente $\frac{1}{2}$ (50%) de chance de obtermos uma cara (ou uma coroa) em cada lançamento. Ora, se um lançamento não influencia nos demais, ou seja, se são lançamentos independentes, então podemos afirmar que a probabilidade de qualquer uma das possibilidades acima é dada pela multiplicação das probabilidades de cada lançamento individual. Assim, digamos, para o padrão de resultados $P(1,0,0,1,1)$, temos:

$P(1,0,0,1,1) = \frac{1}{2}\times\frac{1}{2}\times\frac{1}{2}\times\frac{1}{2}\times\frac{1}{2} = \frac{1}{32}$

Ora, para cada um dos outros padrões essa mesma probabilidade se mantém. Logo, a probabilidade de A é simplemente a soma dessas 10 probabilidades individuais:

$P(A) = \frac{1}{32} + \frac{1}{32} + \frac{1}{32} + \frac{1}{32} + \frac{1}{32} + \frac{1}{32} + \frac{1}{32} + \frac{1}{32} + \frac{1}{32} + \frac{1}{32} = 10 \times \frac{1}{32} = \frac{10}{32} = 0.3125$

Se você quiser, pode calcular no R com a função `dbinom`:

```
dbinom(3, 5, 1/2)
```

```
## [1] 0.3125
```

A fim de generalizar um pouco mais, imagine que denotemos por p a probabilidade de sucesso (cara) e por q, ou seja, $1 - p$, a probabilidade de fracasso (coroa). Com isso, para o padrão de resultados já mostrado acima, teríamos:

$P(1, 0, 0, 1, 1) = p \times q \times q \times p \times p = p^3q^2$

Com o uso dessa notação, poderíamos recalcular a $P(A)$, que seria:

$P(A) = 10p^3q^2 = 10 \times (\frac{1}{2})^3 \times (\frac{1}{2})^2 = 0.3125$

Usamos o número 10 simplesmente porque há 10 combinações possíveis de resultados, como visto acima. Com isso, podemos generalizar para n ensaios com k sucessos (logo, $n - k$ fracassos). Assim sendo, a probabilidade de obter um único sucesso é dada por:

$P(X = k) = p^k q^{n-k}$

É dessa relação que tiramos a fórmula da distribuição binomial dada acima e repetida abaixo[1]:

$P(x|n, p) = \binom{n}{k} p^k (1 - p)^{n-k}$

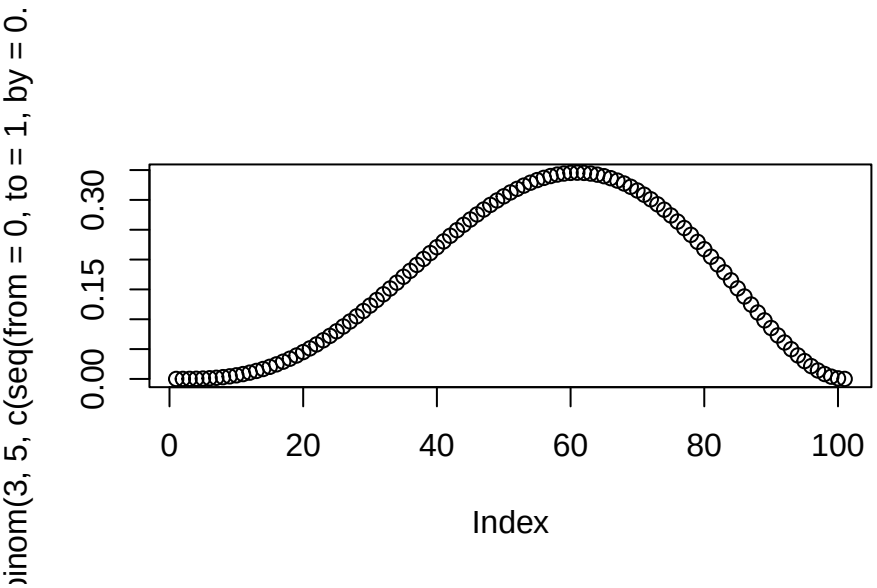
Nesse caso dado acima, temos que a probabilidade associada ao evento "lançar uma moeda não viciada" é conhecido, ou seja, 50%. Agora imaginemos que essa probabilidade não seja conhecida, ou seja, esse parâmetro (que pode assumir qualquer valor entre 0 e 1) não seja informado. Esse, em geral, é o caso com o qual estamos lidando na realidade da pesquisa, pois em geral não sabemos qual a probabilidade de, digamos, os falantes produzirem o verbo A com marca de plural em vez de singular. Se queremos saber isso, fazemos um experimento: submetemos 5 pessoas a um contexto de produção de frases com o verbo A e registramos quantas vezes elas produzem plural e quantas singular. Suponhamos que 3 instâncias sejam de plural e 2 de singular.

Na fórmula acima, n e k estão definidos, ou seja, 5 e 3 e queremos saber os possíveis valores de p.

A questão é que não sabemos qual a probabilidade do parâmetro populacional, ou seja, qual a probabilidade de os falantes produzirem verbos no plural. Dado os resultados do meu experimento, vou buscar o valor da probabilidade mais comum, mais verossímel. Logo, em vez de colocar na minha função uma probabilidade, como fizemos para o caso da moeda não viciada, vamos colocar todos os valores possíveis enter 0 e 1 (não todos, porque são infinitos, mas muitos deles).

```
plot(dbinom(3, 5, c(seq(from = 0, to = 1, by = 0.01))))
```

[1] O núcleo dessa fórmula é dado por $p^k(1 - p)^{n-k}$. O restante é apenas uma constante. Guarde essa informação, pois será útil daqui a pouco.

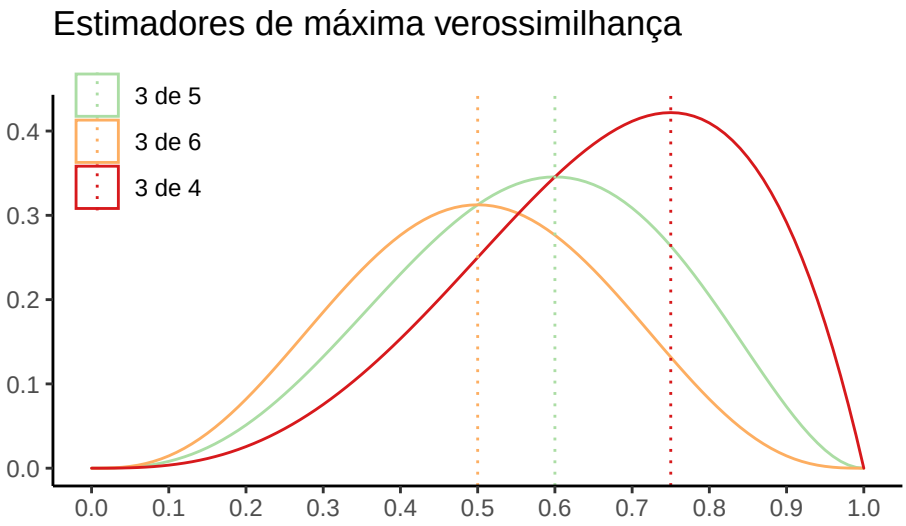


Quando fazemos isso, a função binomial que tínhamos antes (chamada de *Probability Mass Function* - PMF) passa a receber outro nome, *Função de Verossimilhança* ou *Likelihood Function*. No R, as funções cujo nome começa com `d` (`dnorm()`, `dbinom()`, `dpois()`, etc), quando recebem um vetor de probabilidades, nos dão justamente a *densidade de probabilidades* ou (*Probability Density Function* - PDF) associada ao fenômeno que está sendo modelado.

Mas como interpretar isso no caso do experimento fictício que estamos simulando: dado os resultados do meu experimento (3 verbos no plural em 5), o mais verossímil é assumir que a probabilidade de produção de plural com esse verbo é de cerca de 60% (olhe o gráfico!), pois esse é o valor que maximiza a função (*Estimador de Máxima Verossimilhança* ou *Maximum Value Estimator*).

```
dat <- data.frame(x = c(seq(from = 0, to = 1, by = 0.01)),
                  y1 = dbinom(3, 5, c(seq(from = 0, to = 1, by = 0.01))),
                  y2 = dbinom(3, 6, c(seq(from = 0, to = 1, by = 0.01))),
                  y3 = dbinom(3, 4, c(seq(from = 0, to = 1, by = 0.01))))

data <- dat %>%
  pivot_longer(names_to = "dist", values_to = "Valores", cols = c("y1", "y2", "y3"))

max_v <- data %>%
  group_by(dist) %>%
  filter(Valores == max(Valores))

legend_pal <- c("#ABDDA4", "#FDAE61", "#D7191C")

data %>%
  ggplot(aes(x = x, y = Valores, color = dist)) +
  geom_density(stat = "identity", alpha = 0.5) +
  geom_vline(data = max_v, aes(xintercept = x, color = dist),
             linetype = "dotted") +
  scale_color_manual(labels = c("3 de 5", "3 de 6", "3 de 4"),
                     breaks = c("y1", "y2", "y3"),
                     values = legend_pal, name = NULL) +
  scale_x_continuous(breaks = seq(from = 0, to =1, by = 0.1)) +
  labs(x = "", y = "") +
  theme_classic() +
```

```
  theme(legend.position = c(0.1, .9)) +
  ggtitle("Estimadores de máxima verossimilhança\n")
```

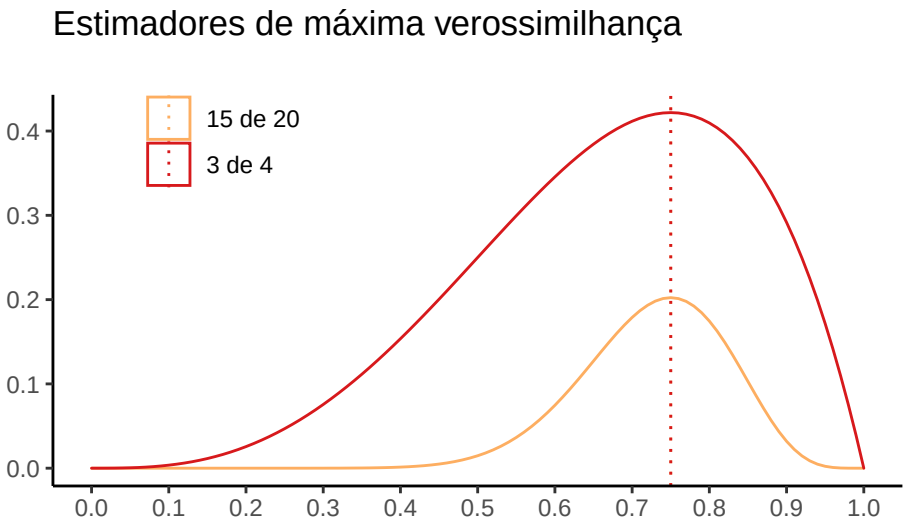


Vamos pensar que em vez de 5 tenhamos feito 6 *trials* e que as respostas com verbo plural tenham sido ainda 3 (Olhe o gráfico acima!). Nesse caso, o valor que maximiza a função é justamente 50%. Então, a melhor aposta que podemos fazer, dado o nosso experimento, é que a probabilidade de plural com verbos do tipo A seja esse valor. Se pararmos para pensar, esse é o justamente o sentido de um experimento: a partir de uma tarefa, estimar a probabilidade de determinado fenômeno ocorrer. Se nosso experimento tivesse nos dados 3 respostas em 4, nossa melhor estimativa, a que maximiza a função de verossimilhança, é 75.

Outra informação importante que buscamos ao fazer um experimento diz respeito ao grau de segurança da nossa estimativa. Uma coisa é levar 5 pessoas ao laboratório e obter 3 verbos no plural, outra é levar 20 pessoas e obter 15 verbos no plural e 5 no singular. Veja o gráfico:

```
dat <- data.frame(x = c(seq(from = 0, to = 1, by = 0.01)),
                  y2 = dbinom(15, 20, c(seq(from = 0, to = 1, by = 0.01))),
                  y3 = dbinom(3, 4, c(seq(from = 0, to = 1, by = 0.01))))

data <- dat %>%
  pivot_longer(names_to = "dist", values_to = "Valores", cols = c("y2", "y3"))

max_v <- data %>%
  group_by(dist) %>%
  filter(Valores == max(Valores))

legend_pal <- c("#FDAE61", "#D7191C")

data %>%
  ggplot(aes(x = x, y = Valores, color = dist)) +
  geom_density(stat = "identity", alpha = 0.5) +
  geom_vline(data = max_v, aes(xintercept = x, color = dist),
             linetype = "dotted") +
  scale_color_manual(labels = c("15 de 20", "3 de 4"),
                     breaks = c("y2", "y3"),
                     values = legend_pal, name = NULL) +
  scale_x_continuous(breaks = seq(from = 0, to =1, by = 0.1)) +
  labs(x = "", y = "") +
  theme_classic() +
```

```
  theme(legend.position = c(0.2, .9)) +
  ggtitle("Estimadores de máxima verossimilhança\n")
```



Perceba que o estimador de máxima verossimilhança continua o mesmo (75%), mas, aumentando minha amostra, temos agora muito mais confiança no nosso estimador: com esses dados, eu posso não saber exatamente qual é a probabilidade de produzir um verbo do tipo A no plural, mas, "chutando" a partir do gráfico, me parece bem provável que este valor seja algo entre 50% e 90%. Com apenas 4 participantes na tarefa, minha segurança estava, quando muito, entre 20% ou 30% e 100%.

LogLikelihood

Lembre-se de quando calculamos a probabilidade do evento $P(1, 0, 0, 1, 1)$:

$P(1, 0, 0, 1, 1) = \frac{1}{2} \times \frac{1}{2} \times \frac{1}{2} \times \frac{1}{2} \times \frac{1}{2} = \frac{1}{32}$

Nesse caso, a probabilidade conjunta de cada um dos lançamentos da moeda era a multiplicação de cada probabilidade individual. A tarefa de fazer cada uma dessas multiplicações é muito mais complexa computacionalmente do que fazer uma soma, por exemplo. Logo, a solução encontrada é calcular o logaritmo da probabilidade de cada um desses eventos e fazer a soma deles. Nesse caso, em vez da Verossimilhança (*Likelihood*), temos o Logaritmo da Verossimilhança (*LogLikelihood*). Se você quiser retranformar para Verossimilhança novamente, basta usar a função `exp()` (a função exponencial $f(x) = a^x$), como mostrado abaixo:

```
# Multiplicação das probabilidades
(1/2 * 1/2 * 1/2 * 1/2 * 1/2)
```

```
## [1] 0.03125
```

```
# Soma dos logaritmos das probabilidades
(log_a <- log(1/2) + log(1/2) + log(1/2) + log(1/2) + log(1/2))
```

```
## [1] -3.465736
```

```
# Retornando à probabilidade conjunta
exp(log_a)
```

```
## [1] 0.03125
```

Como você pôde notar, a soma dos logaritmos das probabilidades nos leva ao mesmo valor da multiplicação das probabilidades.

Mas por que isso é importante para a discussão que estamos fazendo aqui? Para entender, vamos imaginar um experimento fictício, em que comparamos 20 participantes, mensurando seus tempos de reação (RT) entre verbos do tipo A e do tipo B. Como estamos brincando de Deus aqui, definamos que a média de leitura de verbo A seja 50 milisegundos mais rápida que verbos do tipo B.

```
set.seed(1245)

fake_data <- data.frame(suj = c(1:20, 1:20),
                        cond = c(rep("A", 20), rep("B", 20)),
                        RT = c(rnorm(20, 200, 50), rnorm(20, 250, 50)))

fake_data$suj <- as.factor(fake_data$suj)
fake_data$cond <- as.factor(fake_data$cond)

fake_data %>%
  group_by(cond) %>%
  summarise(Médias = mean(RT),
            sd = sd(RT))
```

```
## # A tibble: 2 x 3
##   cond  Médias    sd
##   <fct>  <dbl> <dbl>
## 1 A       178.  42.8
## 2 B       250.  47.6
```

Minha amostra, no caso em questão, não chegou extamente a esses valores porque, obviamente, é uma amostra aleatória:

```
250-178
```

```
## [1] 72
```

Agora pretendo buscar um modelo matemático que descreva esses dados adequadamente. Será que as condições experimentais ajudam a explicar melhor esses dados ou não faz diferença tê-las aí? Para saber isso, vamos ajustar dois modelos aos dados, um modelo linear cheio, em que os RTs são modelados em função das condições; e um modelo linear nulo, em que os RTs são modelados sem as condições, com base apenas na média global dos dados. Por fim, vamos comparar esses dois modelos com a função `anova()`.

Um adendo: neste caso, como nós criamos os dados, sabemos que o modelo cheio é mais adequado, mas o R não sabe disso.

```
# Modelo cheio: RT em função das condições
# Claramente o modelo adequado neste caso
mod_cheio <- lmer(RT~cond + (1|suj), data= fake_data)

# Modelo nulo: RT sem considerar as condições
# Claramente um modelo menos explicativo do processo gerativo dos dados
mod_nulo <- lmer(RT~1 + (1|suj), data= fake_data)

# Comparação entre os dois modelos
anova(mod_cheio, mod_nulo)
```

```
## Data: fake_data
## Models:
## mod_nulo: RT ~ 1 + (1 | suj)
## mod_cheio: RT ~ cond + (1 | suj)
##           npar    AIC    BIC  logLik deviance  Chisq Df Pr(>Chisq)
## mod_nulo     3 442.95 448.02 -218.48   436.95
```

```
## mod_cheio    4 422.26 429.01 -207.13   414.26 22.696  1  1.898e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Como você deve ter notado, existe uma coluna ali denominada `logLik`: isso porque os modelos lineares mistos ajustados com a função `lmer()` usam uma estimativa de máxima verossimilhança para buscar o melhor ajuste aos dados. Mais especificamente, usam, por padrão, um tipo específico chamado REML (*Restricted Maximum Likelihood*):

```
isREML(mod_cheio)
```

```
## [1] TRUE
```

É essa informação que vai permitir ao algoritmo do R dizer qual o melhor modelo: no caso, o que maximizar o logaritmo da verossimilhança (no caso acima, o modelo cheio maximiza esse valor, já que -207 é um número maior do que -218). Isso significa que as estimativas dos modelos lineares mistos ajustados com `lmer()` são "controladas" pela informação advinda da máxima verossimilhança[2]. Mas, e se tivermos outra informação confiável, além dos dados do nosso experimento, que possa ajudar a melhorar o ajuste desse modelo? É nesse ponto que entram os modelos bayesianos.

A noção de priors

IMPORTANTE! Muito do que está aqui eu tirei das aulas do prof. Shravan Vasishth e algumas coisas do curso do Gilberto. Vamos dar os créditos a quem de fato entende do assunto.

Voltemos ao nosso experimento com verbos do tipo A e a produção de plural. Vamos supor que eu esteja trabalhando com uma população de crianças com *Déficit Específico da Linguagem* - DEL e que haja muita informação na área indicando que essas crianças, devido a suas dificuldades com marcas de concordância, na verdade produzem tais marcas aleatoriamente, ora colocando a marca no verbo ora não colocando (imagine que elas estejam "chutando" aleatoriamente).

Se estamos seguros de que é isso o que de fato ocorre, então podemos especular sobre a distribuição de probabilidades da produção dessas crianças *antes* mesmo de realizarmos o experimento. Vamos usar a distribuição beta para isso.

Como sabemos, a distribuição beta tem dois parâmetros, a e b e o núcleo da sua fórmula é dado por:

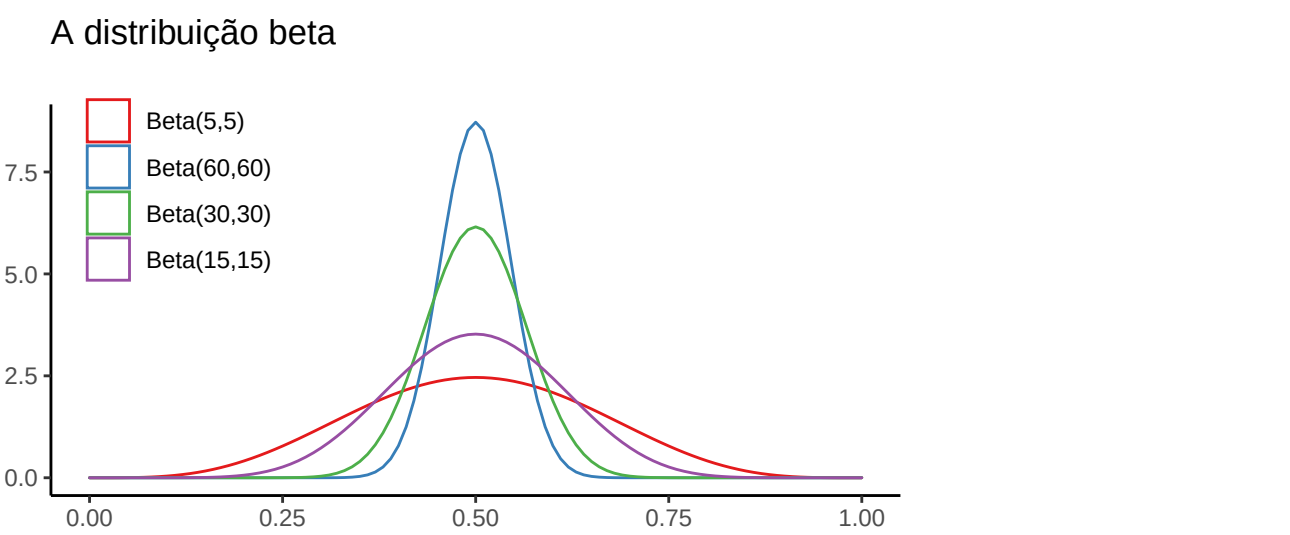
$\theta^{a-1}(1-\theta)^{b-1}$

No R, podemos calcular a densidade de probabilidades da distribuição beta para cada um dos valores entre 0 e 1 apenas acrescentando um vetor com tais valores e os parâmetros correspondentes. Vamos olhar para alguns deles:

```
beta_data <- data.frame(x = c(seq(from = 0, to = 1, by = 0.01)),
                        b_qq = dbeta(c(seq(from = 0, to = 1, by = 0.01)),
                                     shape1 = 60,
                                     shape2 = 60),
                        b_tt = dbeta(c(seq(from = 0, to = 1, by = 0.01)),
                                     shape1 = 30,
                                     shape2 = 30),
                        b_vv = dbeta(c(seq(from = 0, to = 1, by = 0.01)),
                                     shape1 = 10,
                                     shape2 = 10),
                        b_cc = dbeta(c(seq(from = 0, to = 1, by = 0.01)),
                                     shape1 = 5,
                                     shape2 = 5)) %>%
```

[2]Se você de fato quiser entender como os modelos mistos calculam esse valor, pode dar uma olhada neste paper. Mas eu confesso que não entendo nada dessa matemática altamente complexa.

```
  pivot_longer(cols = c("b_qq", "b_tt", "b_vv", "b_cc"),
               names_to = "dist", values_to = "Valores")

legend_pal <- c("#ABDDA4", "#D7191C", "#FDAE61")

beta_data %>%
  ggplot(aes(x = x, y = Valores, color = dist)) +
  geom_density(stat = "identity", alpha = 0.5) +
  scale_color_brewer(palette = "Set1", name = NULL,
                     labels = c("Beta(5,5)", "Beta(60,60)", "Beta(30,30)", "Beta(15,15)")) +
  labs(x = "", y = "") +
  theme_classic() +
  theme(legend.position = c(0.15, .8)) +
  ggtitle("A distribuição beta\n")
```



Vamos olhar com carinho para essas distribuições. Todas elas nos informam que as crianças com DEL têm maior probabilidade com 50%, ou seja, que elas estão "chutando" aleatoriamente se produzirão ou não marcas de plural nos verbos do tipo A. Contudo, Beta(5,5) diz que haverá uma grande variabilidade, e que haverá também crianças que acertam quase tudo e crianças que acertam quase nada. Beta(60,60), por sua vez, nos diz que as crianças estarão produzindo consistentemente no nível da chance e que não haverá valores discrepantes (não há muita probabilidade de eu selecionar um grupo de crianças aleatoriamente e surgir 90% de marcas de plural, já que esse valor está muito distante na cauda dessa distribuição).

Sendo assim, vamos pegar Beta(30,30), que nos parece uma distribuição razoável. Suponhamos que temos essas informações, de que tais crianças produzem marcas de plural com certa consistência no nível da chance, afastando-se um pouco do meio, mas não muito, a ponto de termos um grupo qualquer com 10% ou 80% de marcas de plural.

Então concluímos o primeiro passo: definimos uma prior: Beta(30, 30), que nos dá 50% de probabilidade de crianças com DEL produzirem verbos do tipo A no plural.

O segundo passo será realizar o experimento. Imaginemos então que tenhamos realizado a tarefa descrita anteriormente e que conseguimos das 20 instâncias de verbos do tipo A, 15 plurais. Como já vimos, a estimativa de máxima verossimilhaça da distribuição binomial nos dá 75% de probabilidade.

O nosso terceiro passo, então, é multiplicar o núcleo da minha likelihood pelo núcleo da minha prior (Não se assuste com a conta abaixo! Sério! Ela é muito simples, basta você se lembrar de que, na multiplicação de bases iguais, repete-se a base e somam-se os expoentes: $a^2 \times a^5 = a^{2+5} = a^7$).

$$
\begin{aligned}
Bin(k|n,\theta) \times Beta(a,b) = \\
[\theta^k(1-\theta)^{n-k}] \times [\theta^{a-1}(1-\theta)^{b-1}] = \\
[\theta^k\theta^{(a-1)}] \times [(1-\theta)^{(n-k)}(1-\theta)^{(a-1)}] = \\
[\theta^{k+(a-1)}] \times [(1-\theta)^{(n-k)+(b-1)}]
\end{aligned}
$$

Lembre-se da fórmula da distribuição beta:

$\theta^{a-1}(1-\theta)^{b-1}$

E compare com o resultado da multiplicação feita acima, que vamos apenas rescrever abaixo:

$\theta^{[k+a]-1}(1-\theta)^{[n-k+b]-1}$

Vamos chamar os elementos entre colchetes acima de $a^* = k + a$ e $b^* = n - k + b$ e reescrever esse resultado assim:

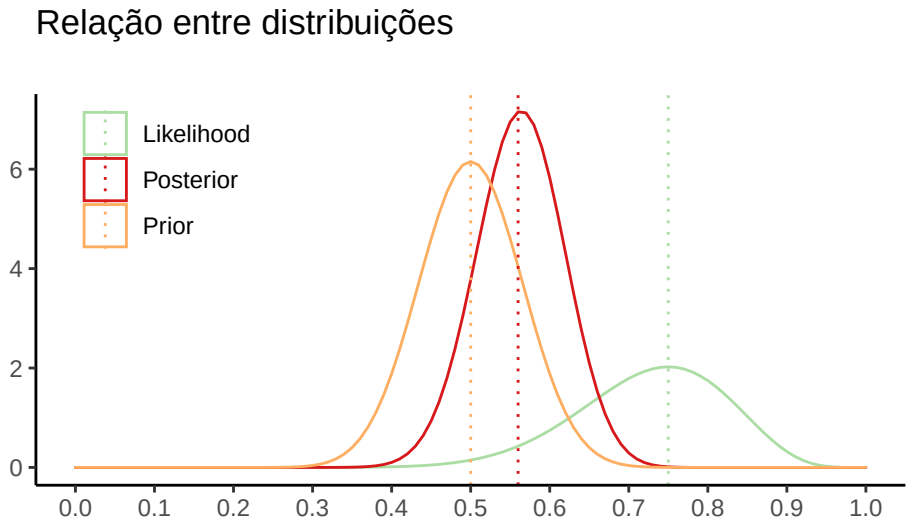
$\theta^{a^*-1}(1-\theta)^{b^*-1}$

O resultado da multiplicação da distribuição binomial (nossa *Likelihood*) pela beta (nossa *prior*) é uma nova distribuição beta com parâmetros a^* e b^*, ou seja: $Beta(k + a, n - k + b)$. Essa a nossa *posterior*, que tem o mesmo formato da nossa *prior*, mas que é modulada pelos dados, ou seja, pela *Likelihood*. Em outras palavras, a posterior é uma nova distribuição que é influenciada tanto pelos dados quanto pela informação prévia que tínhamos sobre o fenômeno em questão.

Agora que temos todos os dados, podemos realizar a análise (sem modelos matemáticos complexos, ANOVAs, nada, apenas com os cálculos que fizemos manualmente: lembre-se que o Gilberto disse que nos modelos bayesianos ficava muito mais fácil calcular).

```
k = 15 # nº de verbos no plural
n = 20 # nº total de trials
a = 30 # parâmetro a da distribuição beta
b = 30 # parâmetro b da distribuição beta

x <- c(seq(from = 0, to = 1, by = 0.01))

likelihood <- dbinom(k, n, c(seq(from = 0, to = 1, by = 0.01)))

prior <- dbeta(c(seq(from = 0, to = 1, by = 0.01)),
               shape1 = a,
               shape2 = b)

posterior <- dbeta(c(seq(from = 0, to = 1, by = 0.01)),
                   shape1 = k+a, # Novos parâmetros da nossa posterior
                   shape2 = n-k+b)

analysis <- data.frame(x, likelihood, prior, posterior) %>%
  mutate(likelihood = likelihood*10) %>% # Só aumenta a visibilidade no gráfico
  pivot_longer(cols = c(likelihood, prior, posterior),
               names_to = "dist",
               values_to = "Valores")

max_v <- analysis %>%
  group_by(dist) %>%
  filter(Valores == max(Valores))

legend_pal <- c("#ABDDA4", "#D7191C", "#FDAE61")
```

```
analysis %>%
  ggplot(aes(x = x, Valores, color = dist)) +
  geom_density(stat = "identity", alpha = 0.5) +
  geom_vline(data = max_v, aes(xintercept = x, color = dist),
             linetype = "dotted") +
  scale_color_manual(labels = c("Likelihood", "Posterior", "Prior"),
                     values = legend_pal, name = NULL) +
  scale_x_continuous(breaks = seq(from = 0, to =1, by = 0.1)) +
  labs(x = "", y = "") +
  theme_classic() +
  theme(legend.position = c(0.15, .8)) +
  ggtitle("Relação entre distribuições\n")
```




```
max_v %>% select(-Valores)
```

```
## # A tibble: 3 x 2
## # Groups:   dist [3]
##       x dist
##   <dbl> <chr>
## 1  0.5  prior
## 2  0.56 posterior
## 3  0.75 likelihood
```

Os dados me davam uma indicação de que o parâmetro populacional era da ordem de 75% (*Likelihood*). Contudo, meu conhecimento da área me informava de que esse tipo de fenômeno ocorre com cerca de 50% de probabilidade (*Prior*). Eu então combinei as duas informações para chegar a um "valor mais preciso" de 56%. É por isso que se diz que a posterior é um compromisso entre a prior e a likelihood.

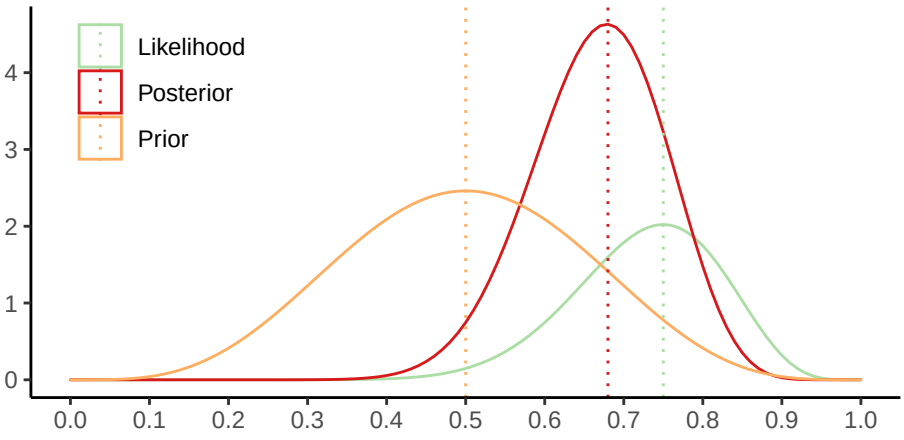
Observe as aspas neste "valor mais preciso". A prior está de fato influenciando bastante na minha análise. Isso significa que eu devo ter muita segurança da informação trazida pela prior, já que estou "contrariando" os resultados experimentais. Basicamente, se mantenho essa análise, estou confiando que os resultados do meu experimento podem não ser tão precisos (amostra muito pequena, por exemplo) e, por isso, complemento essa informação com a prior.

Priors pouco informativas

Mas imaginemos que não saibamos muito sobre o comportamento das crianças com DEL. Temos a informação de que elas ficam no nível da chance, mas que não são tão consistentes nas proximidades de 50% como

modelamos. Neste caso, podemos escolher uma prior pouco informativa, como a beta(5,5). Assim ficaria a nossa análise:

```
k = 15 # nº de verbos no plural
n = 20 # nº total de trials
a = 5 # parâmetro a da distribuição beta
b = 5 # parâmetro b da distribuição beta

x <- c(seq(from = 0, to = 1, by = 0.01))

likelihood <- dbinom(k, n, c(seq(from = 0, to = 1, by = 0.01)))

prior <- dbeta(c(seq(from = 0, to = 1, by = 0.01)),
               shape1 = a,
               shape2 = b)

posterior <- dbeta(c(seq(from = 0, to = 1, by = 0.01)),
                   shape1 = k+a, # Novos parâmetros da nossa posterior
                   shape2 = n-k+b)

analysis <- data.frame(x, likelihood, prior, posterior) %>%
  mutate(likelihood = likelihood*10) %>% # Só aumenta a visibilidade no gráfico
  pivot_longer(cols = c(likelihood, prior, posterior),
               names_to = "dist",
               values_to = "Valores")

max_v <- analysis %>%
  group_by(dist) %>%
  filter(Valores == max(Valores))

legend_pal <- c("#ABDDA4", "#D7191C", "#FDAE61")

analysis %>%
  ggplot(aes(x = x, Valores, color = dist)) +
  geom_density(stat = "identity", alpha = 0.5) +
  geom_vline(data = max_v, aes(xintercept = x, color = dist),
             linetype = "dotted") +
  scale_color_manual(labels = c("Likelihood", "Posterior", "Prior"),
                     values = legend_pal, name = NULL) +
  scale_x_continuous(breaks = seq(from = 0, to =1, by = 0.1)) +
  labs(x = "", y = "") +
  theme_classic() +
  theme(legend.position = c(0.15, .8)) +
  ggtitle("Relação entre distribuições\n")
```




```
max_v %>% select(-Valores)
```

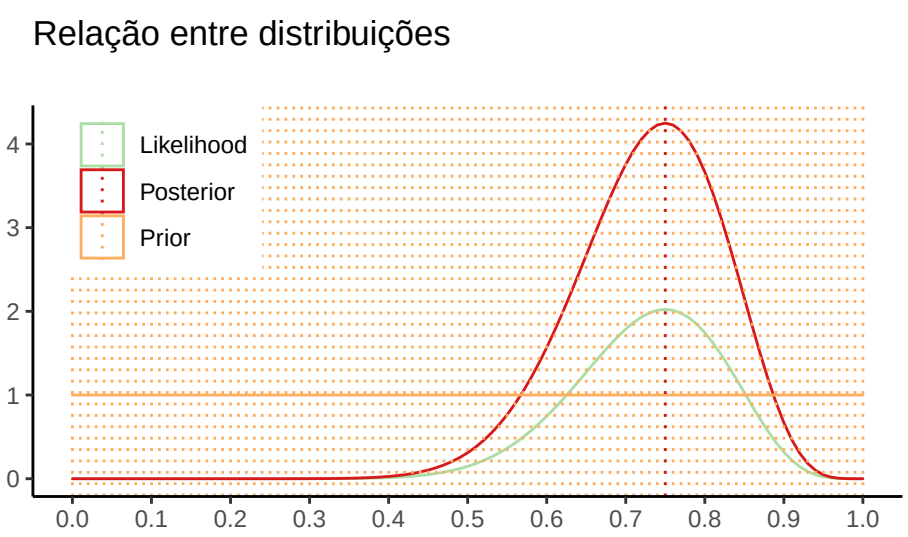
```
## # A tibble: 3 x 2
## # Groups:   dist [3]
##       x dist
##   <dbl> <chr>
## 1  0.5  prior
## 2  0.68 posterior
## 3  0.75 likelihood
```

Nesse caso, a prior não teve tanta influência e quem dominou foi a *Likelihood*: se não temos muito informação, melhor confiar mais no meu experimento mesmo.

Podemos, ainda, ter o caso de não sabermos nada quanto ao comportamento dessas crianças. Nesse caso, podemos estabelecer uma distribuição beta(1,1), em que todos os valores são igualmente possíveis (veja o gráfico).

```
k = 15 # nº de verbos no plural
n = 20 # nº total de trials
a = 1 # parâmetro a da distribuição beta
b = 1 # parâmetro b da distribuição beta

x <- c(seq(from = 0, to = 1, by = 0.01))

likelihood <- dbinom(k, n, c(seq(from = 0, to = 1, by = 0.01)))

prior <- dbeta(c(seq(from = 0, to = 1, by = 0.01)),
               shape1 = a,
               shape2 = b)

posterior <- dbeta(c(seq(from = 0, to = 1, by = 0.01)),
                   shape1 = k+a, # Novos parâmetros da nossa posterior
                   shape2 = n-k+b)

analysis <- data.frame(x, likelihood, prior, posterior) %>%
  mutate(likelihood = likelihood*10) %>% # Só aumenta a visibilidade no gráfico
  pivot_longer(cols = c(likelihood, prior, posterior),
               names_to = "dist",
               values_to = "Valores")
```

```r
max_v <- analysis %>%
  group_by(dist) %>%
  filter(Valores == max(Valores))

legend_pal <- c("#ABDDA4", "#D7191C", "#FDAE61")

analysis %>%
  ggplot(aes(x = x, Valores, color = dist)) +
  geom_density(stat = "identity", alpha = 0.5) +
  geom_vline(data = max_v, aes(xintercept = x, color = dist),
             linetype = "dotted") +
  scale_color_manual(labels = c("Likelihood", "Posterior", "Prior"),
                     values = legend_pal, name = NULL) +
  scale_x_continuous(breaks = seq(from = 0, to =1, by = 0.1)) +
  labs(x = "", y = "") +
  theme_classic() +
  theme(legend.position = c(0.15, .8)) +
  ggtitle("Relação entre distribuições\n")
```



Se isso ocorre, a *Likelihood* domina completamente e não temos qualquer influência da prior na análise dos nossos dados. Neste caso, estamos em situação idêntica ao frequentismo!