

贝叶斯推断和Stan应用

王敏杰

<https://github.com/perlatex/Stan-in-Action>

2022-10-12

我们今天讲一个数据故事

校长交办了一个任务

估算全校同学的平均身高

- 全校普查似乎不现实
- 随机选取200名同学，然后根据这200名同学的身高，推算全校总体的情况

id	height
1	173.72
2	170.89
3	182.11
4	176.21
5	167.08
6	183.12
7	169.74
8	153.99
9	160.58

不费吹灰之力

很快计算出200个同学的平均身高

mean_height

164.89

马上兴高采烈地报告校长，全校同学的身高均值是164.89。

如果您是校长， 您对结果满意？

应该不满意



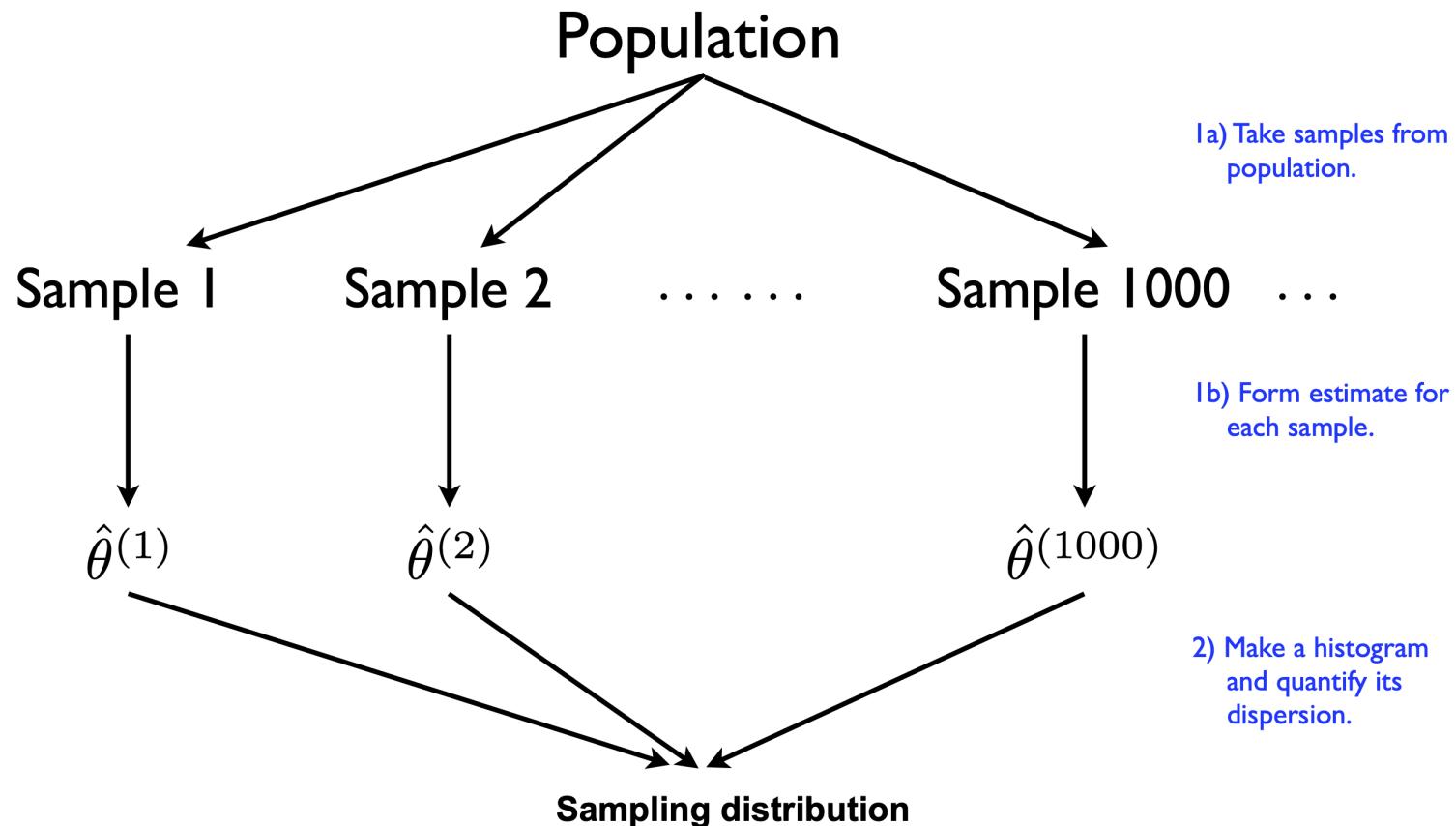
- 因为这200个学生，相对全校而言，是一个很小的样本，难免以偏概全。
- 这个164.89可靠性有多高，不确定性是多少？

我努力改进

你看到校长脸色有些不好看，弱弱地说：“要不我重新再找200个学生，再做一次，或者再找第三组的200个学生，然后第四组，这样重复很多次”。

我努力改进

你看到校长脸色有些不好看，弱弱地说：“要不我重新再找200个学生，再做一次，或者再找第三组的200个学生，然后第四组，这样重复很多次”。



给我一个范围，今天就要

还没等解释完，校长就打断了你的话，“时间来不及了，再说，这又不是做核酸，那有那么多的人力物力，你就用200个同学的身高值，给我估算一个吧，给个范围也行”

给我一个范围，今天就要

还没等解释完，校长就打断了你的话，“时间来不及了，再说，这又不是做核酸，那有那么多的人力物力，你就用200个同学的身高值，给我估算一个吧，给个范围也行”

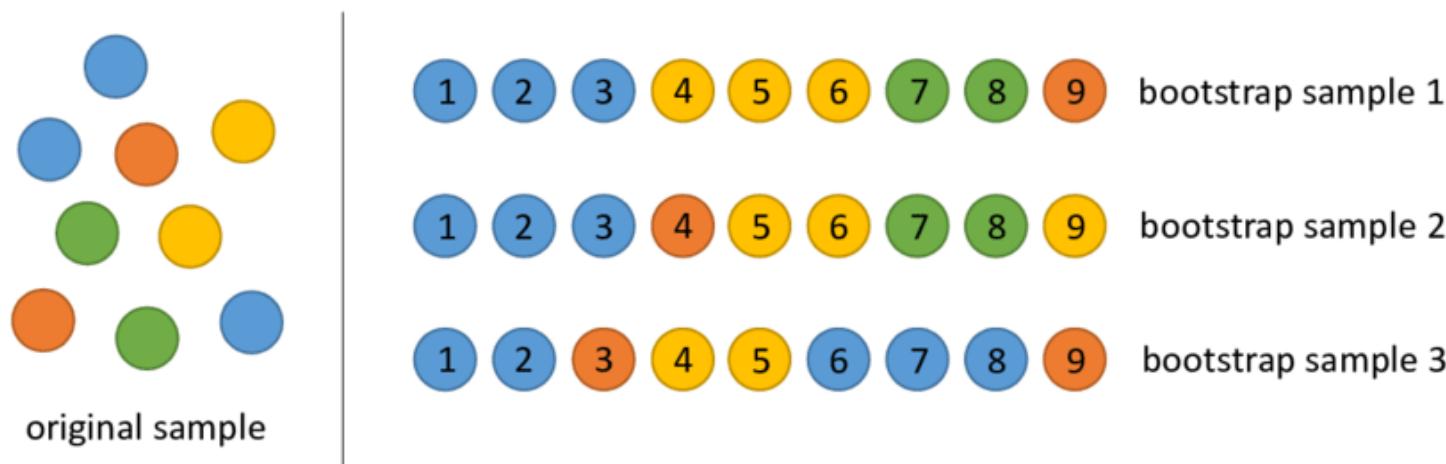


这下有点头疼了，只有一个样本，还要给出范围，那怎么办呢？

一、bootstrap resampling

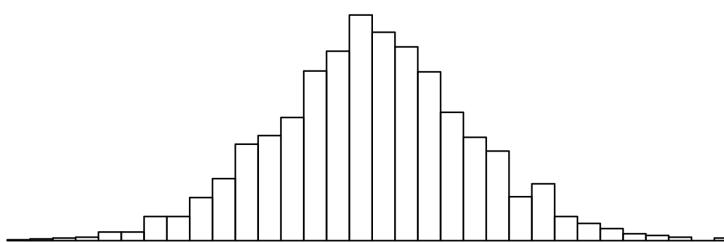
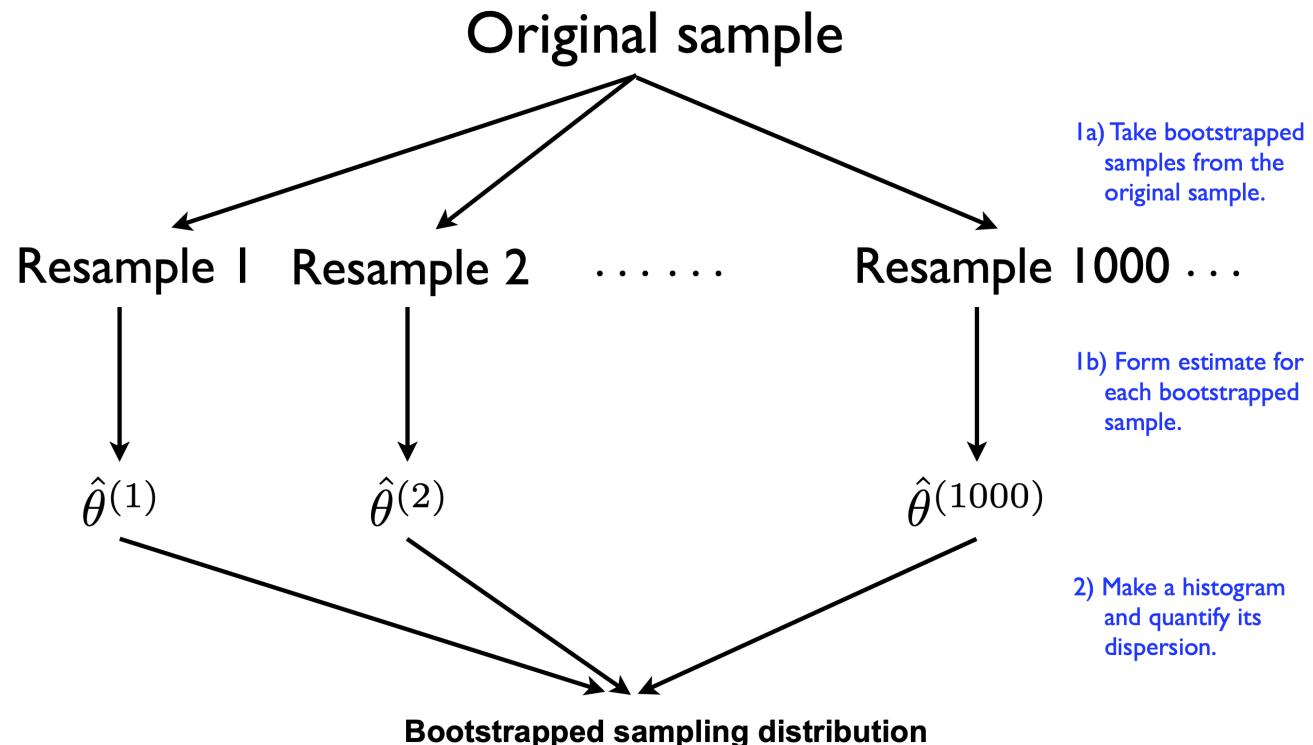
bootstrap resampling

搞统计的人发明了一个很不错的方法Bootstrapping，**有放回的重复抽样**，举个通俗的例子：



- 假定这里有一个口袋，里面装着200个球，你摸一个出来，记录下这个球的重量，然后放回去，搅拌一下，再摸一个出来，称下重量，再放回去，如此往复，记录到200个值后，就停下来，这200个值称之为第一个重抽样样本，然后计算下**均值**。ok，第一个样本的工作完成。
- 然后第二个样本，
- 第三个样本...
- 直到1000个重抽样样本，也就得到了1000个均值
- 最后看看这1000个均值的分布

思维导图



道理明白了，马上开干

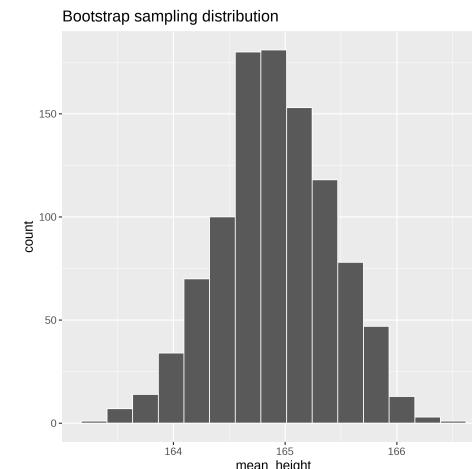
```
bootstrap_once <- function(df) {  
  boot_idx <- sample(1:nrow(df), replace = TRUE)  
  df <- df %>% slice(boot_idx)  
  return(df)  
}  
  
bootstrap_repeat <- function(df, reps = 30){  
  df_out <-  
    purrr::map_dfr(.x = 1:reps, .f = ~ bootstrap_once(df)) %>%  
    dplyr::mutate(replicate = rep(1:reps, each = nrow(df))) %>%  
    dplyr::group_by(replicate)  
  return(df_out)  
}
```

bootstrap resampling

很快得到了1000个均值

replicate	mean_height
1	165.7328
2	165.2511
3	165.0294
4	164.5848
5	165.4332
6	164.9922
7	165.1002
8	165.4287
9	164.0434
10	165.2456

画出了直方图

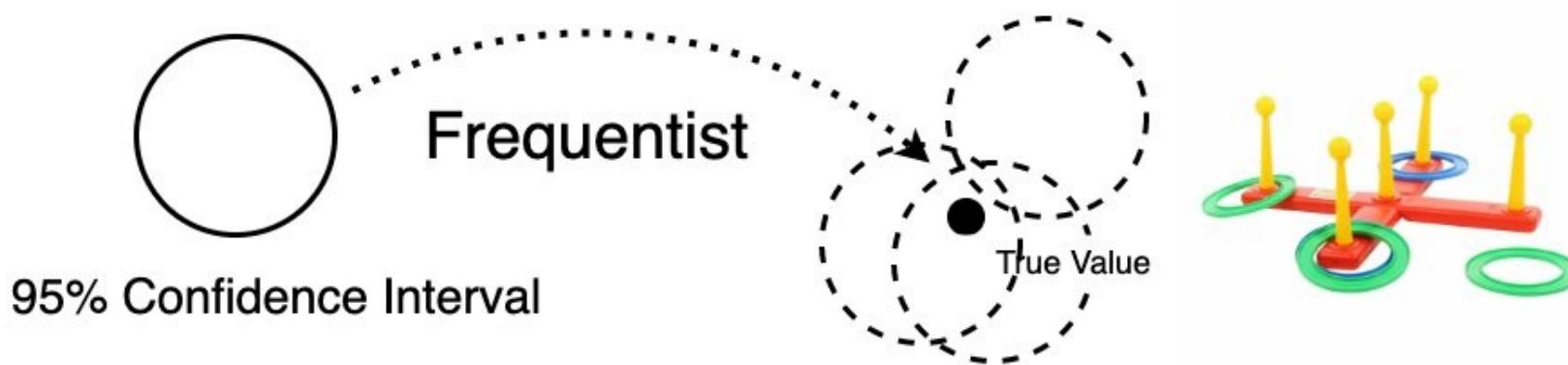


给出置信区间

mean_height	.lower	.upper	.width	.point	.interval
164.91	163.9	165.84	0.95	mean	qi

95%的置信区间

图中黑色圆点，它代表着全校同学的平均身高，它是客观存在的，并且有一个确定的值（只是我们不知道）。



我们的任务就是，捕获这个黑色圆点，这个过程有点像小朋友玩的**套圈游戏**。

置信区间就是圈圈的大小，95%的意思就是，我们扔套圈100次，95次成功套住黑点，换句话说，扔一次套圈，我有95%的概率能捕获到黑色圆点。

回到身高问题，把套圈设置在[163.9, 165.84]这个范围，我就有95%的自信说，它能捕获全校同学的平均身高。

二、来点高级的？

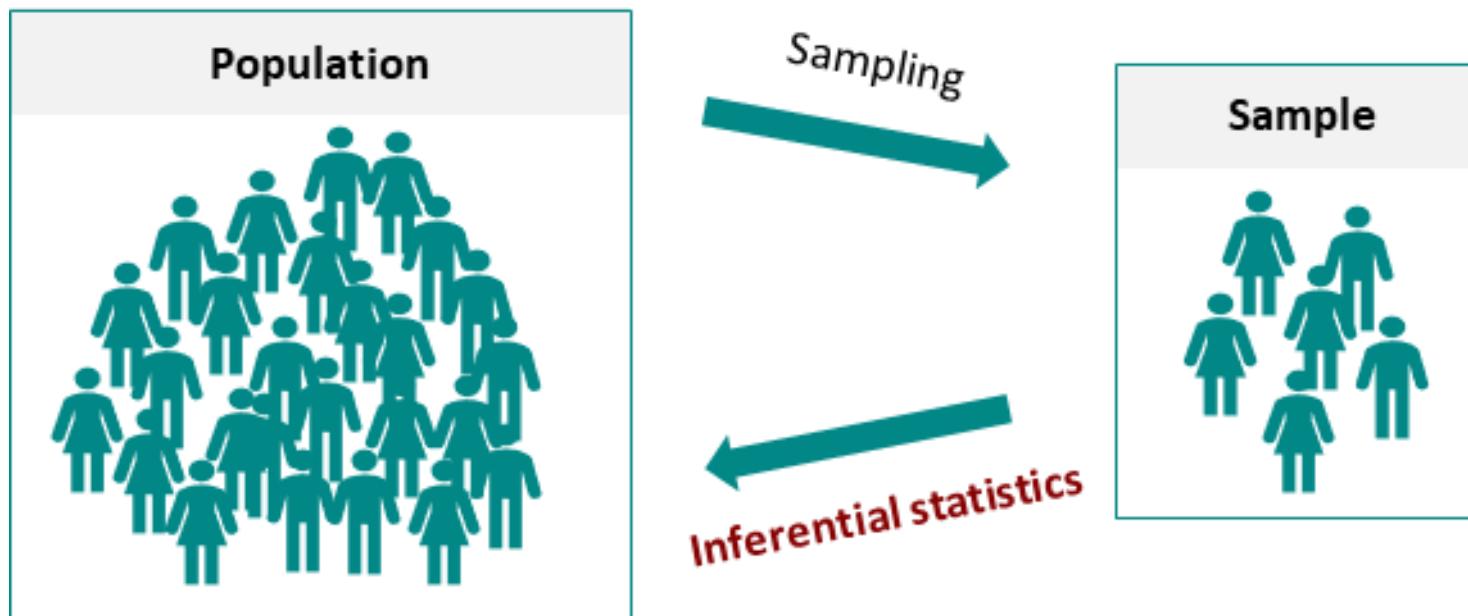
会不会一见钟情？



对的，贝叶斯。我们已经久仰大名。

回到故事的开始

观察到样本数据后，如何推断总体分布的参数 θ ?



贝叶斯公式

贝爷告诉我们，可以用贝叶斯后验概率 $p(\theta|Y)$ 来回答

$$P(A|B) = \frac{P(B|A) P(A)}{P(B)}$$



Thomas Bayes
1702 - 1761

贝叶斯公式

三百年前的光辉思想，仍然影响着现在。我们要好好膜拜下这个贝叶斯公式

$$p(\theta|Y) = \frac{p(Y|\theta)p(\theta)}{p(Y)}$$

贝叶斯公式告诉我们，要得到等式的左边，可以用等式的右边来计算。先认识下贝叶斯公式的每个部分。

- 左边 $p(\theta|Y)$ 称之为后验概率，也就是我们的目标
- $p(Y|\theta)$ 是似然函数，在给定参数后，数据出现的概率
- $p(\theta)$ 参数的先验概率，在看到数据前，参数各种可能性的分布
- $p(Y)$ 边际似然，可以忽略

贝叶斯公式

既然分母可以先忽略，就认为它为 1，于是等式可以变成

$$p(\theta|Y) \propto p(Y|\theta)p(\theta).$$

然后，我们把总体的似然函数，写成每个数据点的似然函数连乘的形式：

$$p(\theta|Y) \propto p(\theta) \prod_{n=1}^N p(y_n|\theta)$$

接着，我们两边取对数，连乘变成了连加。也就是说，我们计算的是**对数概率(log probabilities)**

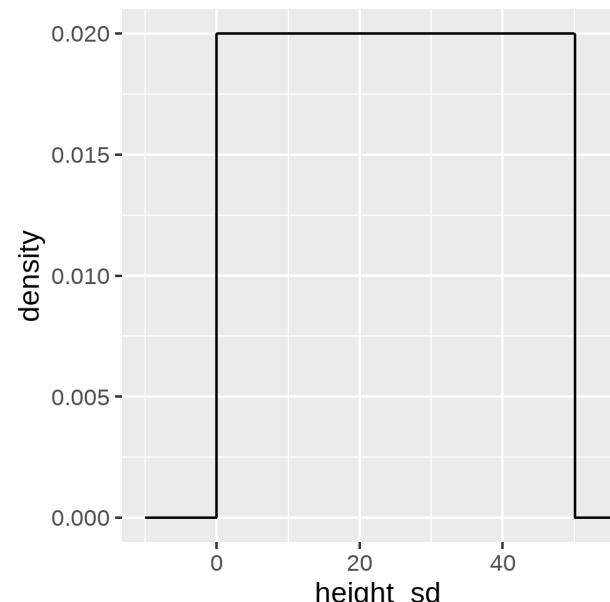
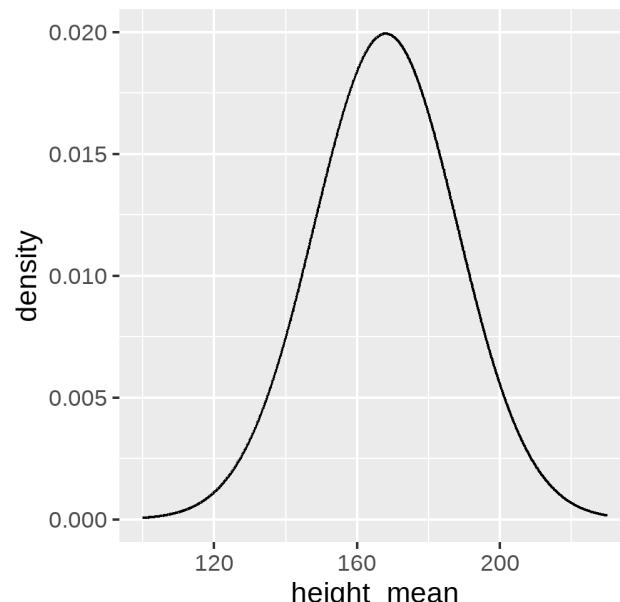
$$\log p(\theta|Y) \propto \log p(\theta) + \sum_{n=1}^N \log p(y_n|\theta)$$

感觉技术上有可操作性了，没错，它就是贝叶斯数据分析的**灵魂**，我们做贝叶斯计算都是仰仗这个等式。

需要一点前戏

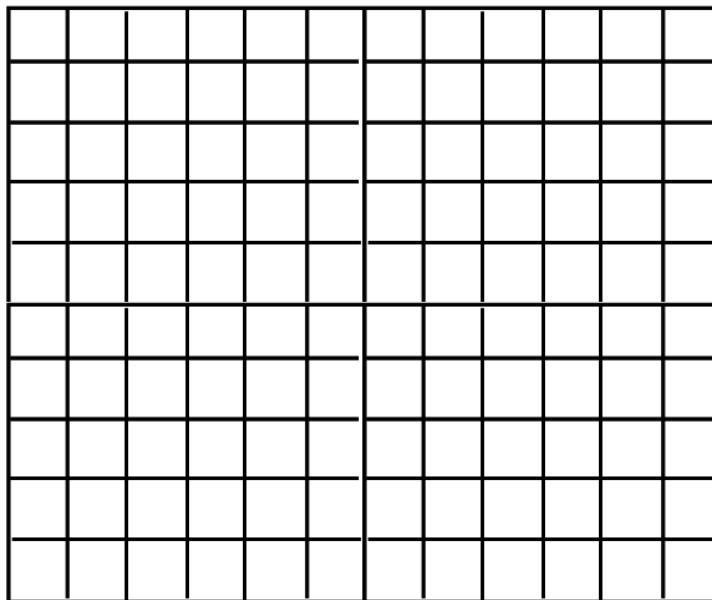
还是校长给出的**身高问题**。通过前面的身高的统计量，我们可以合理的猜测：

- 全校同学的身高均值可能是160, 162, 170, 172, ..., 或者说这个均值在一个范围之内，在这个范围内，有些值的可能性大，有些值可能性较低。比如，认为这值游离在[150,180]范围，其中168左右的可能最大，两端的可能性最低。如果寻求数学语言来描述，它比较符合正态分布的特征，那就这么定了。
- 方差在[0, 50]范围内都有可能，那就假定每个值的可能性都相等吧。



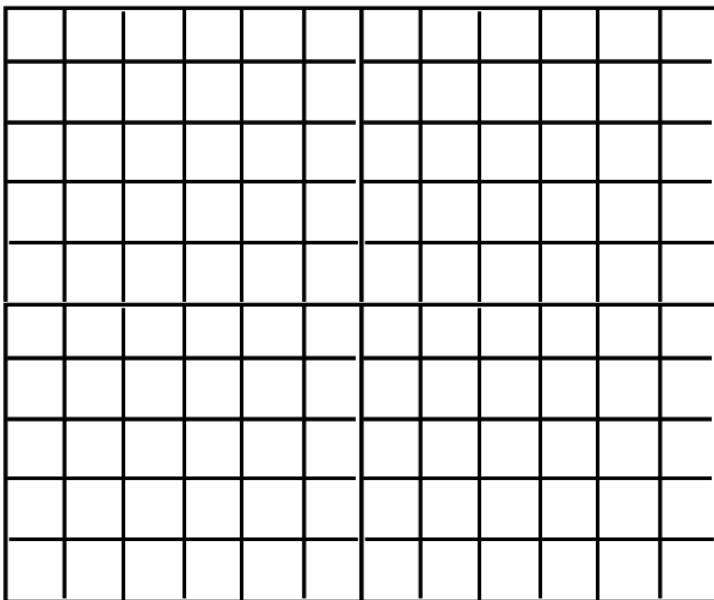
参数空间

第二步，我们需要构建一个**参数空间**，类似网格一样的东西。具体做法是，先指定参数的范围，大一点没关系，然后把这个范围内的所有**可能参数组合**都罗列出来，类似九九乘法表，比如这里构建 1000×1000 个(μ, σ)参数空间



参数空间

第二步，我们需要构建一个**参数空间**，类似网格一样的东西。具体做法是，先指定参数的范围，大一点没关系，然后把这个范围内的所有**可能参数组合**都罗列出来，类似九九乘法表，比如这里构建 1000×1000 个(μ, σ)参数空间

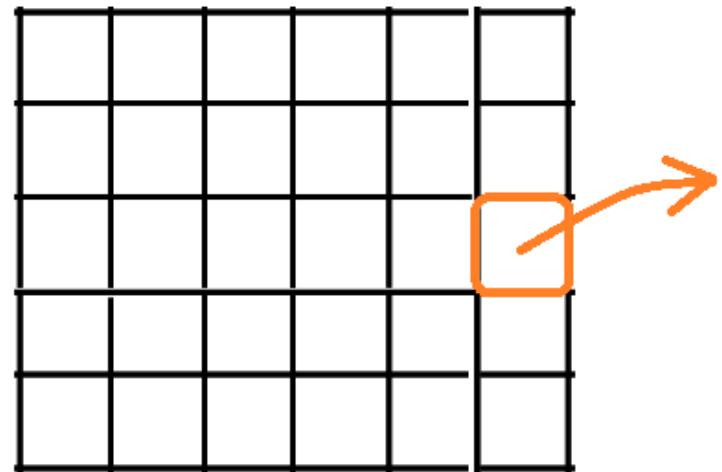


mu	`σ` = 2	`σ` = 3	`σ` = 4	`σ` = 5
$\mu = 172$	N(172,2)	N(172,3)	N(172,4)	N(172,5)
$\mu = 173$	N(173,2)	N(173,3)	N(173,4)	N(173,5)
$\mu = 174$	N(174,2)	N(174,3)	N(174,4)	N(174,5)
$\mu = 175$	N(175,2)	N(175,3)	N(175,4)	N(175,5)
$\mu = 176$	N(176,2)	N(176,3)	N(176,4)	N(176,5)

先验概率的对数

第三步，在参数空间里，计算每个参数在先验分布下的概率密度对数，即下面等式红色部分

$$\log p(\theta|Y) \propto \log p(\theta) + \sum_{n=1}^N \log p(y_n|\theta)$$



```
mu = 175  
dnorm(mu, mean = 168, sd = 20, log = TRUE)  
#> [1] -3.975921
```

```
sigma = 10  
dunif(sigma, min = 0, max = 50, log = T)  
#> [1] -3.912023
```

对数似然

第四步，在参数空间里，每一个参数组合所对应的分布下，计算观察到的200个身高值的**对数似然之和**，即下面等式红色部分

$$\log p(\theta|Y) \propto \log p(\theta) + \sum_{n=1}^N \log p(y_n|\theta)$$

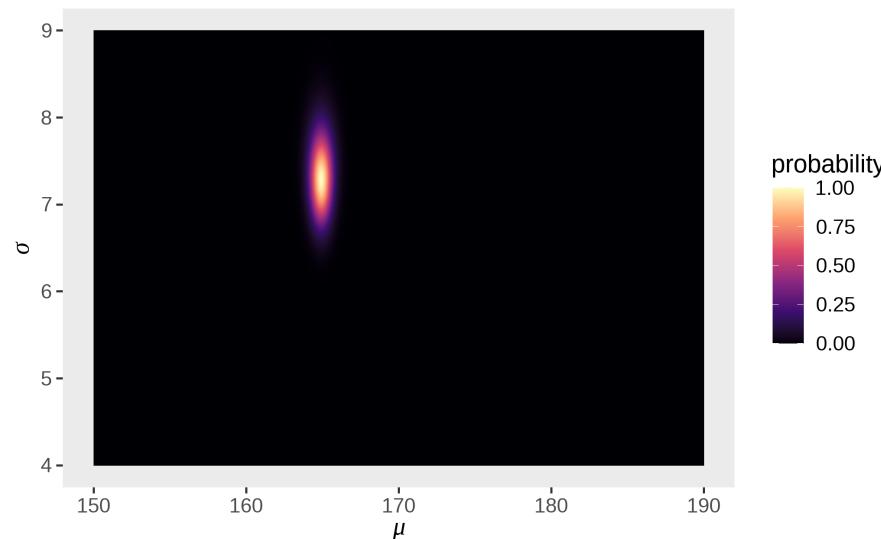
这里有 1000×1000 个(μ, σ)组合，所以会产生 1000×1000 个值

后验概率的对数

第五步，把**先验概率的对数**和**似然概率的对数**加起来，得到后验概率对数(log probabilities)，求指数后就是**后验概率**

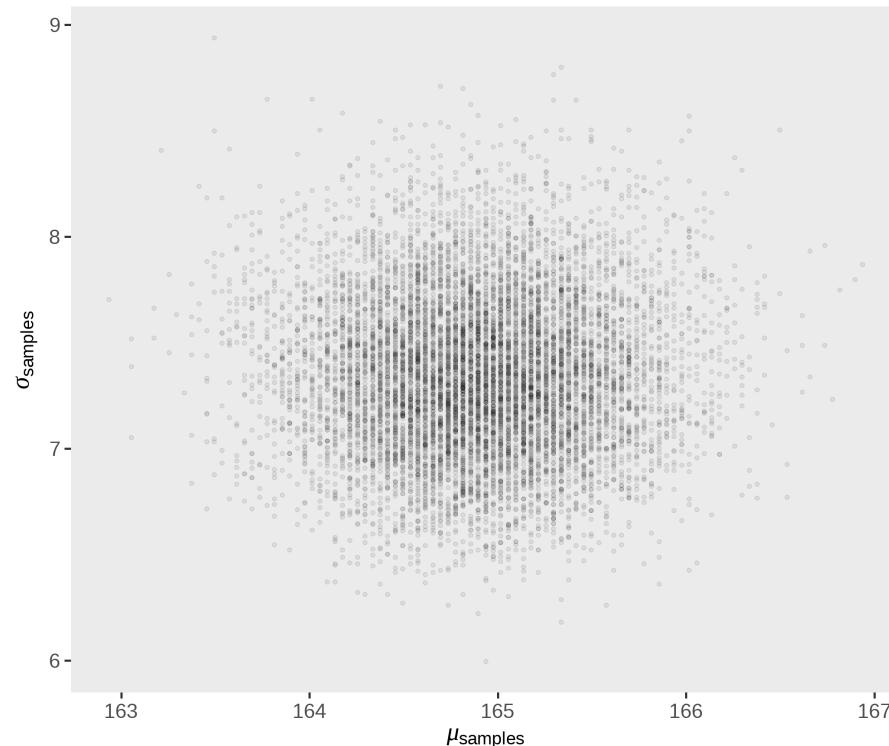
$$\log p(\theta|Y) \propto \log p(\theta) + \sum_{n=1}^N \log p(y_n|\theta)$$

此时，可以想象成一共有 1000×1000 个坑，每个坑装着一个后验概率，有高有低，看上去就像若干个小山峰。



抽样

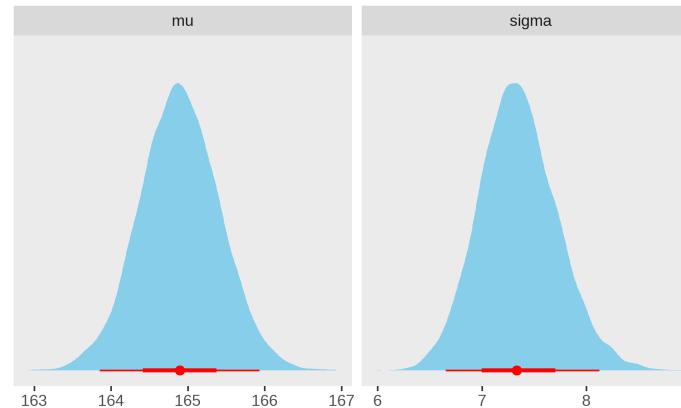
第六步，按照后验概率值的大小抽取样本，得到后验分布。



为什么要抽样呢？因为目前得到的只是概率对数(求指数后是概率)，即每个坑出现的概率，而我们要得到是参数的具体值，身高的均值，所以按照概率大小抽取样本。

后验分布

有了样本，就可以得到均值和标准差的分布

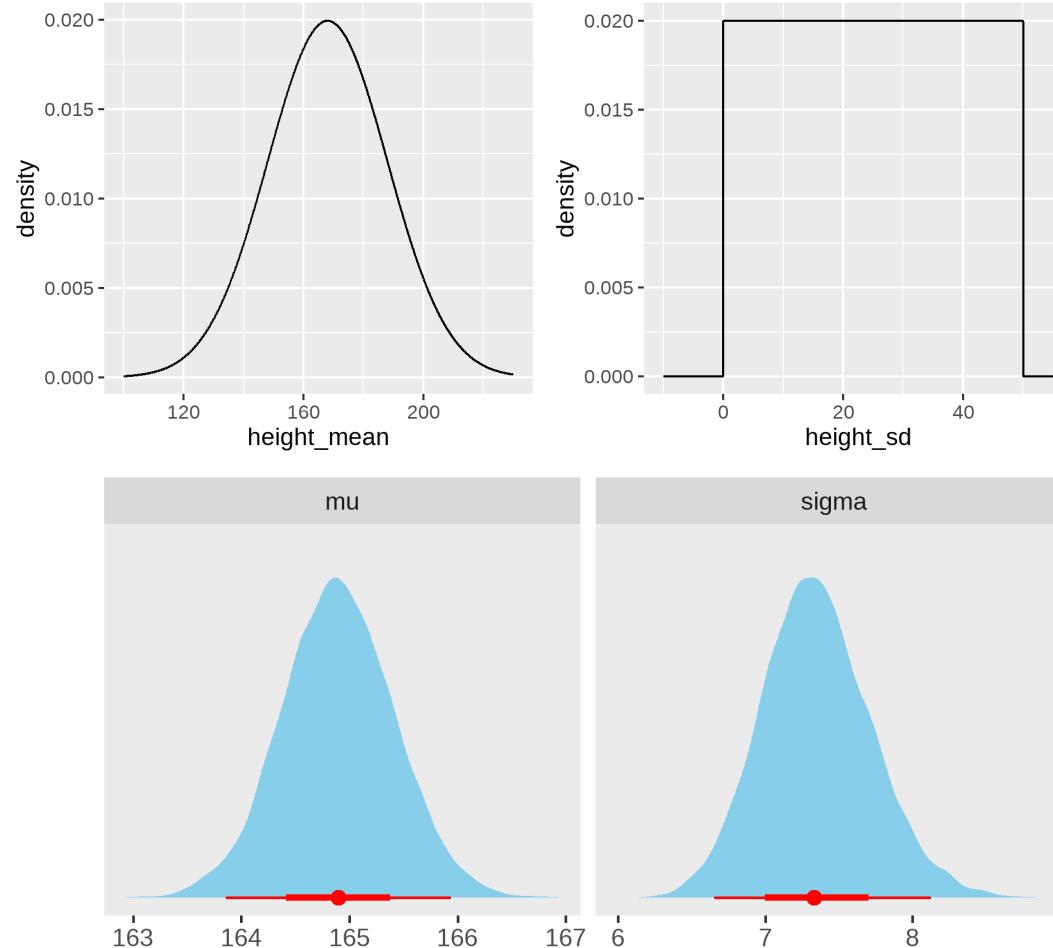


以及后验概率的**最高密度区间**

name	value	.lower	.upper	.width	.point	.interval
mu	164.868332	164.054054	165.695696	0.89	mode	hdi
sigma	7.321609	6.727728	7.913914	0.89	mode	hdi

此时，给出的不在点估计，而是区间估计，给出了各种可能值，以及各可能值的概率。

后验和先验对比



回顾下，贝叶斯的视角

- 我们先赋予参数主观的先验信息，也就意味着参数是变化的值
- 但数据是固定的
- 数据更新了先验信息，得到了后验信息

回顾下，贝叶斯的视角

- 我们先赋予参数主观的先验信息，也就意味着参数是变化的值
- 但数据是固定的
- 数据更新了先验信息，得到了后验信息



图中黑色圆点，是我们的目标。按照贝叶斯的观点，它不是一个固定的或者确定的值，而是各种可能的值，贝叶斯给出的是，**最有可能的是哪些值，以及这些可能值的概率是多少。**

网格近似的方法优劣

以上是通过**网格近似**的方法得到身高分布的后验概率

- 这种方法理解起来并不难
- 但做起来比较麻烦，需要构建参数网格，对于较复杂的模型，计算量会陡增，内存占用大、比较费时，因此在实际的数据中，一般不采用这种方法。

网格近似的方法优劣

以上是通过**网格近似**的方法得到身高分布的后验概率

- 这种方法理解起来并不难
- 但做起来比较麻烦，需要构建参数网格，对于较复杂的模型，计算量会陡增，内存占用大、比较费时，因此在实际的数据中，一般不采用这种方法。

网格近似的方法可以帮助我们很好地理解贝叶斯数据分析。

三、轮到今天的主角们

概率编程工具有很多

1. BUGS (Bayesian inference Using Gibbs Sampling)
2. JAGS (Just Another Gibbs Sampler)
3. PyMC (Python)
4. Turing.Jl (Julia)
5. Stan

什么是Stan



Stan 是当前主流的概率编程语言，主要用于贝叶斯推断。

- Stan广泛应用于社会学、生物、医学、物理、工程和商业等领域
- 贝叶斯不是新东西，但Stan是新东西。

Stan的历史

名字的由来

- 波兰犹太裔核物理学家 Stanislaw Ulam，二战期间研究原子弹时，发明了蒙特卡罗方法
- 蒙特卡罗方法是什么呢？以概率统计理论为指导的数值计算方法
- 贝叶斯界用这种蒙特卡罗方法开发一套程序，并用它创始人的名字Stan命名

开发团队

- 这套程序是由纽约哥伦比亚大学 Andrew Gelman 于2012年发起，由核心开发团队共同开发和维护

Stan如何工作

这里面太多数学和计算机的内容了，核心科技，我真不太懂，求放过。

求放过我吧



如何使用Stan

- Stan首先会把Stan代码翻译成C++，然后在本地编译

如何使用Stan

- Stan首先会把Stan代码翻译成C++，然后在本地编译
- Stan 使用先进的采样技术(Hamiltonian Monte Carlo技术的 No-U-turn 采样器)，允许复杂的贝叶斯模型快速收敛

如何使用Stan

- Stan首先会把Stan代码翻译成C++，然后在本地编译
- Stan 使用先进的采样技术(Hamiltonian Monte Carlo技术的 No-U-turn 采样器)，允许复杂的贝叶斯模型快速收敛
- Stan提供了与 (R, Python, shell, MATLAB, Julia, Stata) 流行语言的接口
 - 在R语言里用rstan, [CmdStanR](#) 包
 - 在Python用PyStan包

如何使用Stan

- Stan首先会把Stan代码翻译成C++，然后在本地编译
- Stan 使用先进的采样技术(Hamiltonian Monte Carlo技术的 No-U-turn 采样器)，允许复杂的贝叶斯模型快速收敛
- Stan提供了与 (R, Python, shell, MATLAB, Julia, Stata) 流行语言的接口
 - 在R语言里用rstan, [CmdStanR](#) 包
 - 在Python用PyStan包
- 把Stan当作R/Python的一个宏包

如何使用Stan

- Stan首先会把Stan代码翻译成C++，然后在本地编译
- Stan 使用先进的采样技术(Hamiltonian Monte Carlo技术的 No-U-turn 采样器)，允许复杂的贝叶斯模型快速收敛
- Stan提供了与 (R, Python, shell, MATLAB, Julia, Stata) 流行语言的接口
 - 在R语言里用rstan, [CmdStanR](#) 包
 - 在Python用PyStan包
- 把Stan当作R/Python的一个宏包
- 在R语言里，还有bayesplot, tidybayes, loo等辅助宏包，完成Stan模型可视化、规整和分析

Stan的优势

相比于传统的方法来说，Stan模型

Stan的优势

相比于传统的方法来说， Stan模型

- 更好的**可操作性**
 - 从模型表达式到代码，更符合人的直觉
 - 模型灵活性。修改几行代码，就转化成一个新的模型

Stan的优势

相比于传统的方法来说， Stan模型

- 更好的可操作性
 - 从模型表达式到代码，更符合人的直觉
 - 模型灵活性。修改几行代码，就转化成一个新的模型
- 更好的透明性
 - 模型的假设
 - 模型的参数

Stan的优势

相比于传统的方法来说， Stan模型

- 更好的可操作性
 - 从模型表达式到代码，更符合人的直觉
 - 模型灵活性。修改几行代码，就转化成一个新的模型
- 更好的透明性
 - 模型的假设
 - 模型的参数
- 更好的可解释性
 - 从贝叶斯公式出发，解释起来更符合常识

Stan代码框架

Stan语法非常严谨，数据结构接近R语言，声明语句类似C++语言，具体可以参考[官方手册](#)。

```
data {  
    // 导入数据  
}  
parameters {  
    // 定义模型要估计的参数  
}  
model {  
    // 后验概率函数  
}
```

从模型到Stan代码

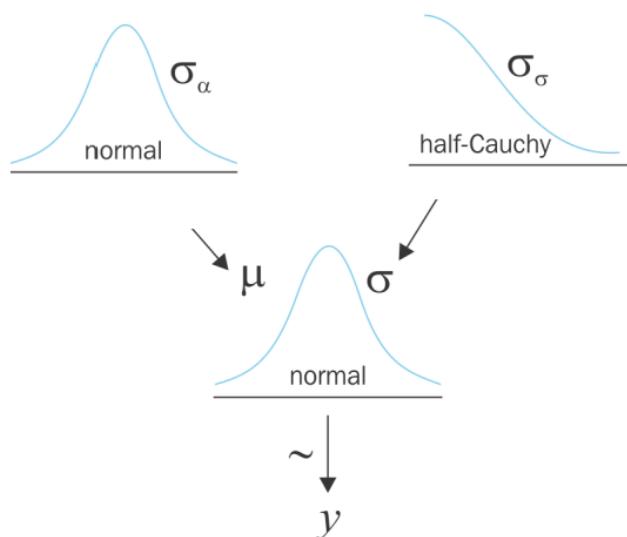
模型

$$y_i \sim \text{normal}(\mu, \sigma)$$

$$\mu \sim \text{normal}(168, 20)$$

$$\sigma \sim \text{half-Cauchy}(0, 1)$$

图示



Stan代码

```
data {  
    int N;  
    vector[N] y;  
}  
parameters {  
    real mu;  
    real<lower=0> sigma;  
}  
model {  
    y ~ normal(mu, sigma);  
  
    mu ~ normal(168, 20);  
    sigma ~ cauchy(0, 1);  
}
```

编译

```
stan_program1 <- write_stan_file(" stanfile.stan"
data {
  int<lower=0> N;
  vector[N] y;
}
parameters {
  real mu;
  real<lower=0> sigma;
}
model {
  mu ~ normal(168, 20);
  sigma ~ cauchy(0, 1);

  y ~ normal(mu, sigma);
}
")
)

stan_data1 <- list(
  N = length(d$height),
  y = d$height
)

modell <- cmdstan_model(stan_file = stan_program1)
fit1 <- modell$sample(data = stan_data1)
```

最高興的事

啊哈，得到了样本，是很高兴的事情

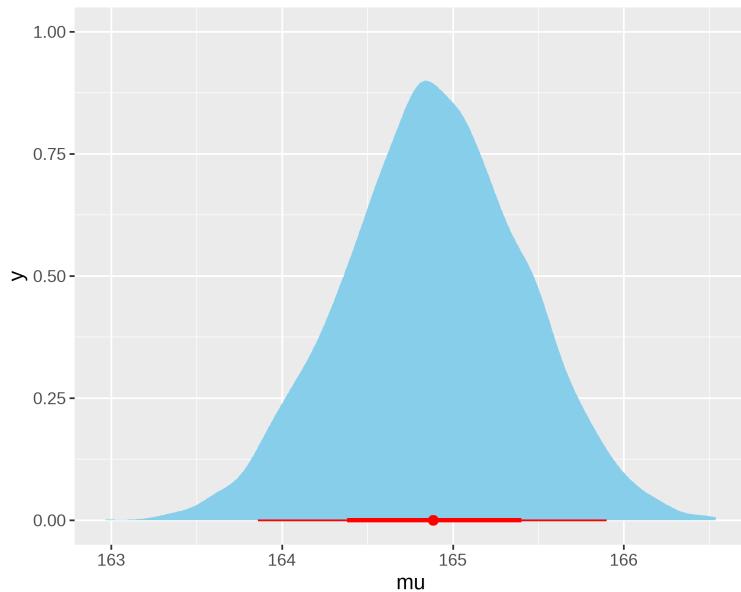


样本

lp_	mu	sigma	.chain	.iteration	.draw	
-500.833	165.748	7.56119	1	1	1	
-500.129	164.983	7.77253	1	2	2	
-500.441	164.446	6.84750	1	3	3	
-500.132	164.694	7.75621	1	4	4	
-500.224	165.133	6.82375	1	5	5	
-499.686	165.067	7.59675	1	6	6	
-499.365	164.701	7.37024	1	7	7	
-499.365	164.701	7.37024	1	8	8	
-499.742	164.849	7.64036	1	9	9	
-499.742	164.849	7.64036	1	10	10	

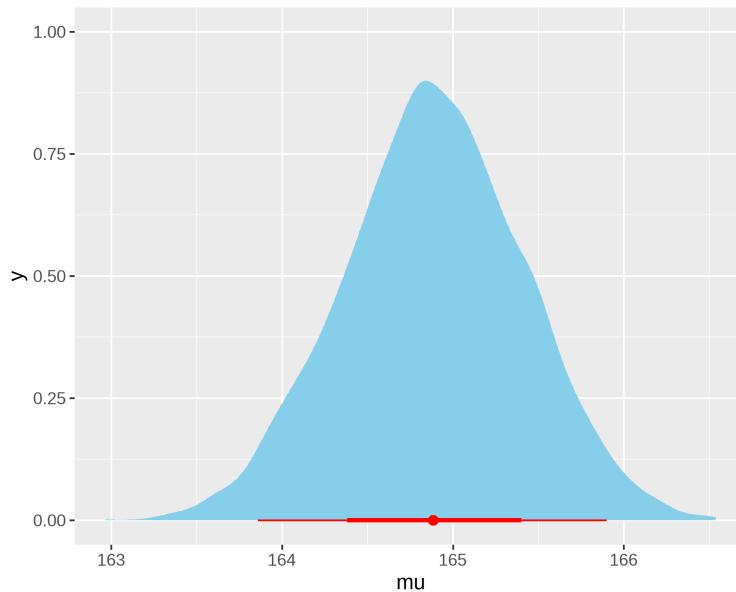
样本

赶快画个图



样本

赶快画个图



接着统计下

.variable	.value	.lower	.upper	.width	.point	.interval
mu	164.839427	164.0150	165.70800	0.89	mode	hdi
sigma	7.274654	6.7294	7.91079	0.89	mode	hdi

四、线性模型

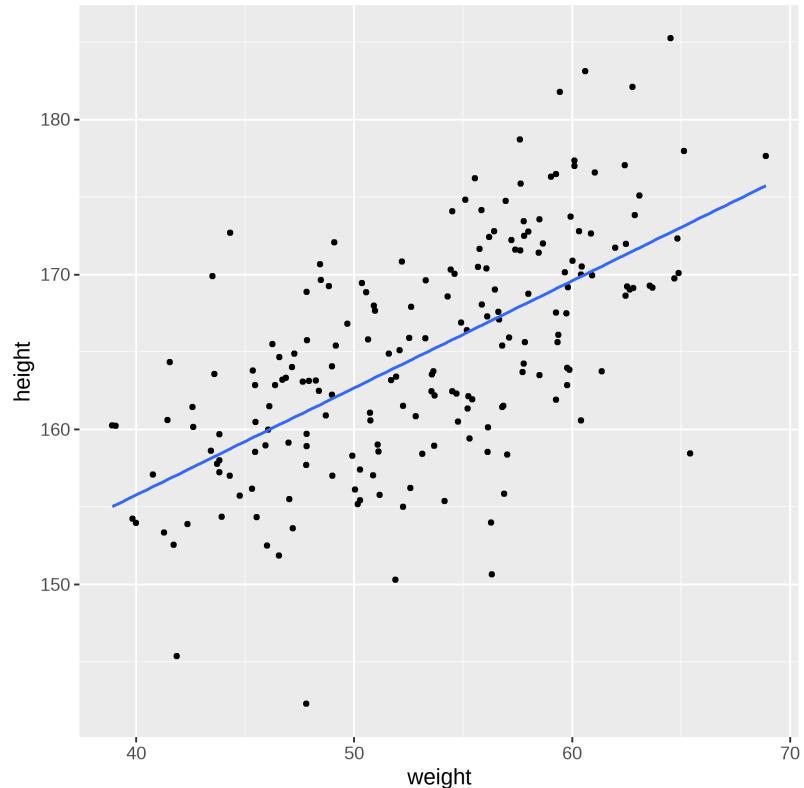
线性模型

我在测量身高的时候，偷偷也测量了体重

id	height	weight
1	173.72	59.93
2	170.89	60.03
3	182.11	62.77
4	176.21	55.54
5	167.08	56.65
6	183.12	60.61
7	169.74	64.70
8	153.99	56.28
9	160.58	60.41
10	165.42	56.80
11	181.77	59.44

线性模型

那我们可以探索下身高和体重的关联



假定数学表达式如下

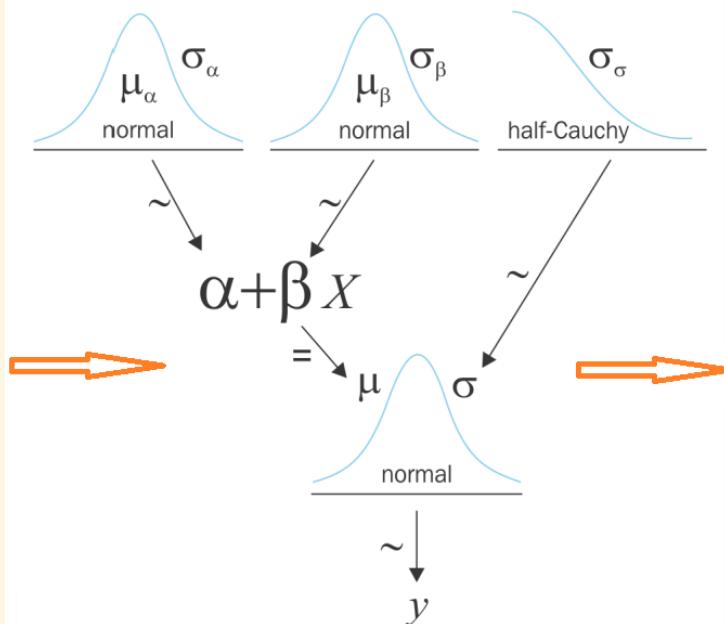
$$\begin{aligned} \text{height}_i &\sim \text{normal}(\mu_i, \sigma) \\ \mu_i &= \alpha + \beta \text{weight}_i \\ \alpha &\sim \text{normal}(0, 4) \\ \beta &\sim \text{normal}(0, 4) \\ \sigma &\sim \text{half-Cauchy}(1) \end{aligned}$$

从模型到Stan代码

模型

$y_n \sim \text{normal}(\mu_n, \sigma)$
 $\mu_n = \alpha + \beta x_n$
 $\alpha \sim \text{normal}(0, 4)$
 $\beta \sim \text{normal}(0, 4)$
 $\sigma \sim \text{half-Cauchy}(1)$

图示



Stan代码

```
data {  
    int<lower=0> N;  
    vector[N] y;  
    vector[N] x;  
}  
parameters {  
    real alpha;  
    real beta;  
    real<lower=0> sigma;  
}  
model {  
    y ~ normal(alpha + beta * x, sigma);  
  
    alpha ~ normal(0, 4);  
    beta ~ normal(0, 4);  
    sigma ~ cauchy(0, 1);  
}
```

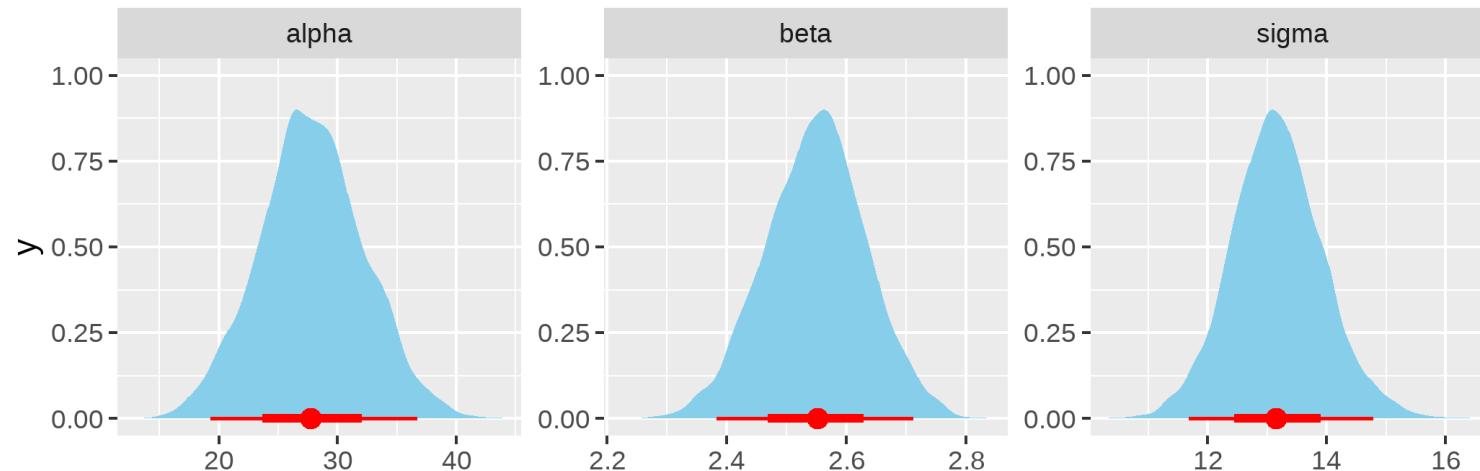
编译运行

```
stan_program2 <- write_stan_file(" stan_data2 <- list( N = nrow(d), x = d$weight, y = d$height )")
```

```
data { int<lower=0> N; vector[N] y; vector[N] x; } parameters { real alpha; real beta; real<lower=0> sigma; } model { y ~ normal(alpha + beta * x, sigma); alpha ~ normal(0, 4); beta ~ normal(0, 4); sigma ~ exponential(1); }
```

结果出来了

参数的后验概率分布



参数的最高密度区间

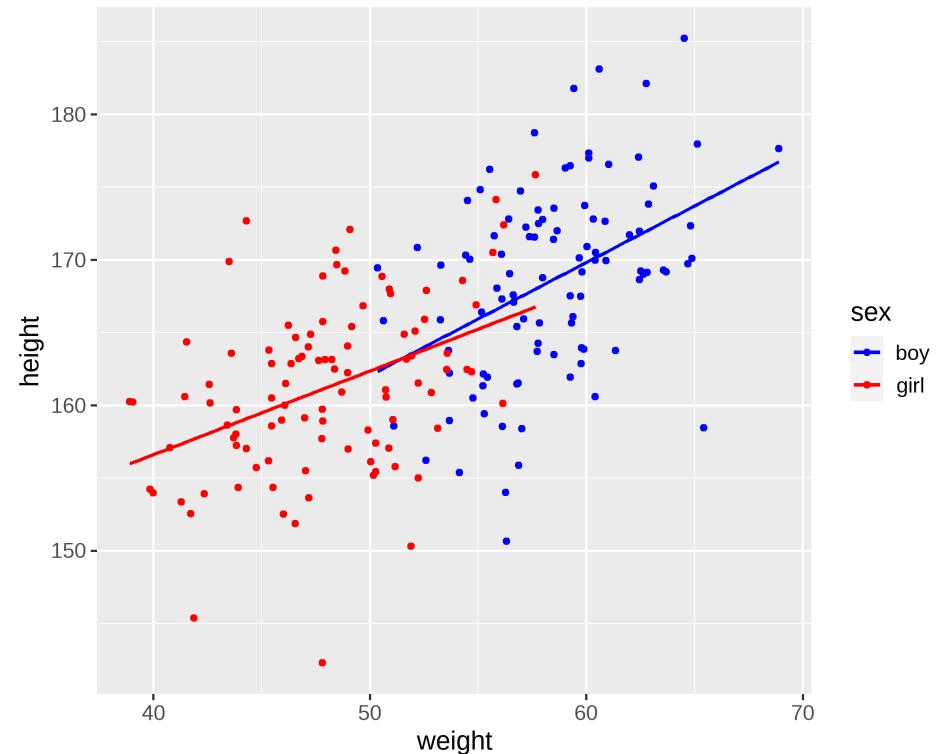
.variable	.value	.lower	.upper	.width	.point	.interval
alpha	26.548093	20.71120	34.91270	0.89	mode	hdi
beta	2.562755	2.40571	2.67581	0.89	mode	hdi
sigma	13.089618	11.95580	14.41740	0.89	mode	hdi

五、多层次模型

多层模型

我们再进一步，不同性别身高和体重的关系，应该是不一样的，我们也探索下呢

	id	sex	height	weight
1	boy	173.72	59.93	
2	boy	170.89	60.03	
3	boy	182.11	62.77	
4	boy	176.21	55.54	
5	boy	167.08	56.65	
6	boy	183.12	60.61	
7	boy	169.74	64.70	
8	boy	153.99	56.28	
9	boy	160.58	60.41	
10	boy	165.42	56.80	



多层模型

这里不是单纯的两个独立的回归分析，而是分成男孩和女孩两组，模型中我们既要考虑组内的变化，又要考虑组与组之间的变化。因此，多层模型写成如下形

$$\begin{aligned}\text{height}_i &\sim \text{Normal}(\mu_i, \sigma) \\ \mu_i &= \alpha_{j[i]} + \beta_{j[i]} \text{weight}_i \\ \begin{pmatrix} \alpha_j \\ \beta_j \end{pmatrix} &\sim N \left(\begin{pmatrix} \mu_\alpha \\ \mu_\beta \end{pmatrix}, \begin{pmatrix} \sigma_\alpha^2 & \rho\sigma_\alpha\sigma_\beta \\ \rho\sigma_\alpha\sigma_\beta & \sigma_\beta^2 \end{pmatrix} \right)\end{aligned}$$

然后加上先验

$$\begin{aligned}\mu_\alpha &\sim \text{Normal}(0, 2) \\ \mu_\beta &\sim \text{Normal}(0, 2) \\ \sigma &\sim \text{Exponential}(1) \\ \sigma_\alpha &\sim \text{Exponential}(1) \\ \sigma_\beta &\sim \text{Exponential}(1) \\ \rho &\sim \text{LKJcorr}(2)\end{aligned}$$

编译运行

```
stan_program3 <- write_stan_file("stan.stan"
data {
  int N;                                // number of obs
  int K;                                // number of predictors
  matrix[N, K] X;                      // model_matrix
  vector[N] y;                          // y
  int J;                                // number of grouping
  int<lower=1, upper=J> g[N]; // index for grouping
}
parameters {
  array[J] vector[K] beta;
  vector[K] MU;
  real<lower=0> sigma;

  vector<lower=0>[K] tau;
  corr_matrix[K] Rho;
}
model {
  vector[N] mu;
  sigma ~ exponential(1);
  tau ~ exponential(1);
  Rho ~ lkj_corr(2);

  for(i in 1:N) {
    mu[i] = X[i] * beta[g[i]];
  }
}
```

结果

可以得到男孩和女孩的不同的截距和斜率

variable	mean	median	sd	mad	q5	q95	rhat	ess_bulk	ess_tail
beta[1,1]	128.580	127.821	5.695	6.096	122.292	138.783	1.183	16.180	460.145
beta[2,1]	128.696	127.951	5.631	6.228	122.507	138.678	1.210	14.324	468.976
beta[1,2]	0.685	0.699	0.098	0.103	0.511	0.794	1.182	16.359	444.736
beta[2,2]	0.679	0.693	0.118	0.133	0.470	0.809	1.235	12.962	294.658

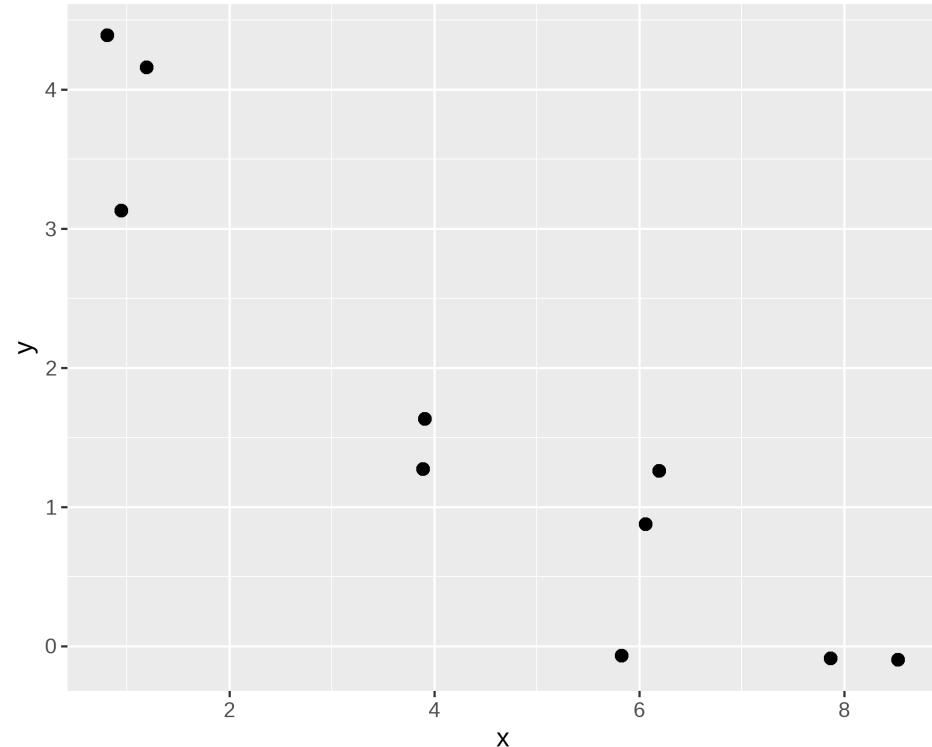
相关系数

variable	mean	median	sd	mad	q5	q95	rhat	ess_bulk	ess_tail
Rho[1,1]	1.000	1.000	0.000	0.000	1.000	1.00	NA	NA	NA
Rho[2,1]	-0.097	-0.195	0.401	0.375	-0.695	0.65	1.191	117.676	1085.084
Rho[1,2]	-0.097	-0.195	0.401	0.375	-0.695	0.65	1.191	117.676	1085.084
Rho[2,2]	1.000	1.000	0.000	0.000	1.000	1.00	NA	NA	NA

六、非线性的案例

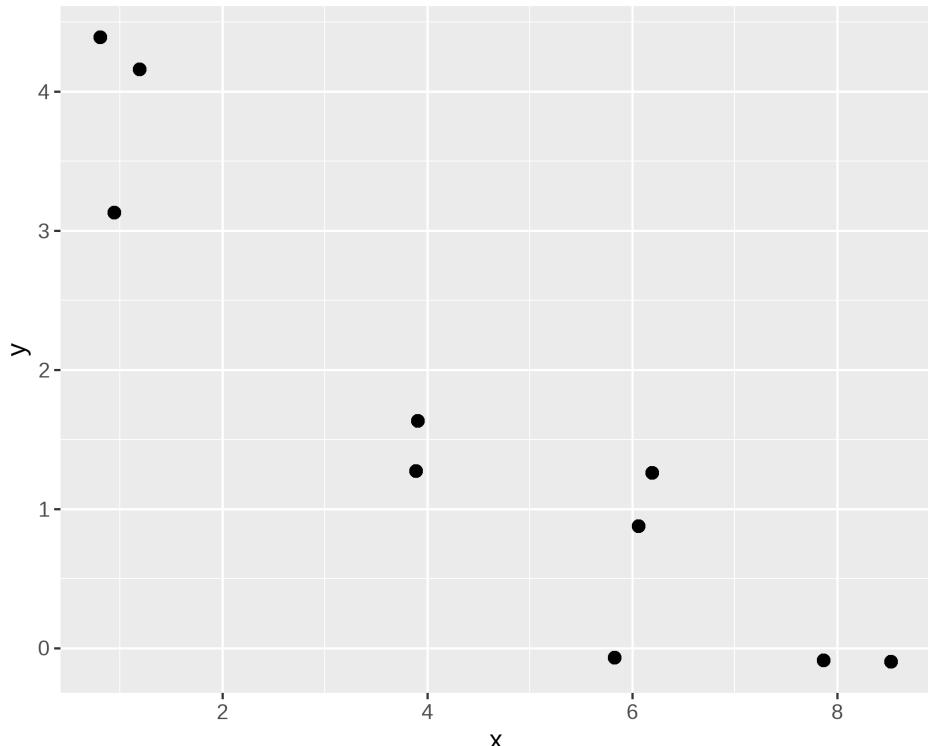
非线性的案例

图中的数据点很少，只有10个



非线性的案例

图中的数据点很少，只有10个



假定x和y满足下面等式的关系，如何估计 a 和 b

$$y_i = ae^{-bx_i} + \epsilon_i$$
$$\epsilon_i \sim \text{normal}(0, \sigma)$$

写成如下等价这种形式，更好理解

$$y_i \sim \text{normal}(\mu_i, \sigma)$$
$$\mu_i = ae^{-bx_i}$$

问题：如何估计 a 和 b ？

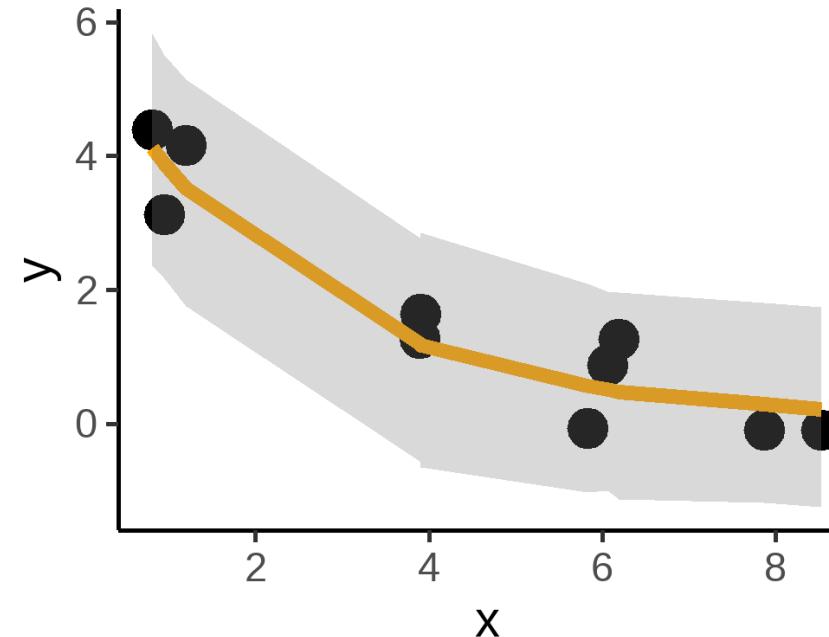
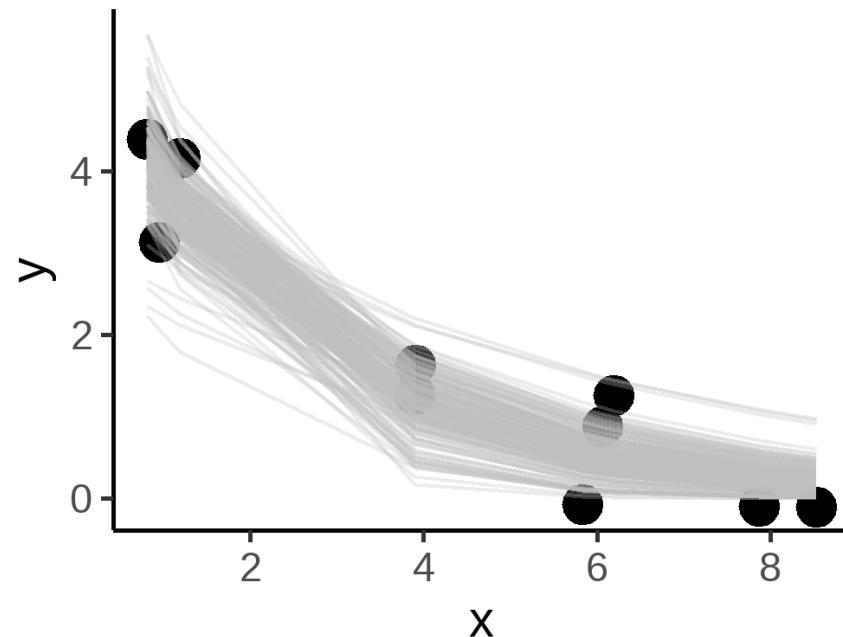
编译运行

```
stan_program4 <- write_stan_file("stan_code4.stan"
data {
  int N;
  vector[N] x;
  vector[N] y;
}
parameters {
  real a;
  real b;
  real sigma;
}
model {
  y ~ normal(a * exp(-b * x), sigma);
  a ~ normal(0, 10);
  b ~ normal(0, 10);
  sigma ~ normal(0, 3);
}

generated quantities {
  vector[N] y_rep;
  vector[N] y_fit;
  for (n in 1:N) {
    y_fit[n] = a * exp(-b * x[n]);
    y_rep[n] = normal_rng(a * exp(-b * x[n]), sigma);
```

模型的预测能力

模型推断的好不好呢？是否**捕获**到数据的特征了呢？



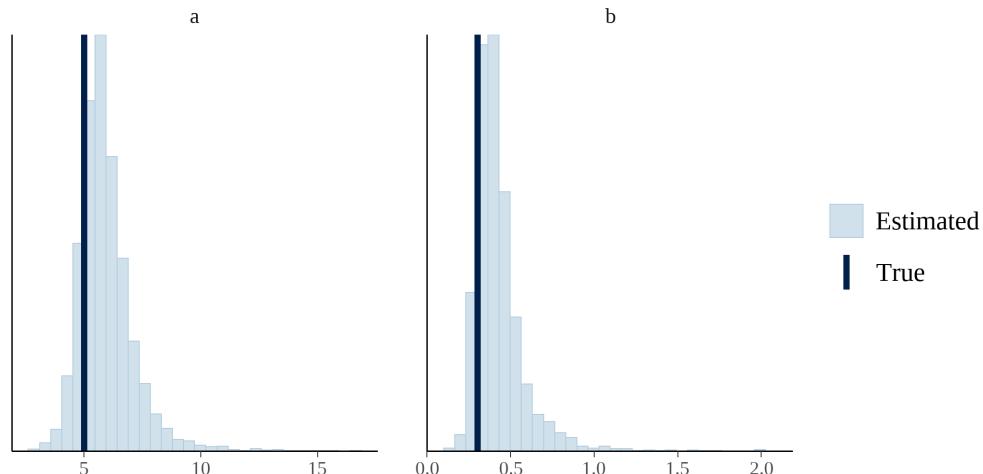
获得参数分布后，就可以从后验分布中随机抽取重复样本集。如果一个贝叶斯模型是“好”的，那么从它模拟产生的数据应该与实际观察到的数据很类似。

参数恢复

事实上，数据是我模拟的，真实值 $a = 5, b = 0.3$ 。模型给出的参数估计是

variable	mean	median	sd	mad	q5	q95	rhat	ess_bulk	ess_tail
a	5.939	5.746	1.207	0.870	4.445	7.950	1.002	1093.555	776.777
b	0.432	0.397	0.165	0.096	0.270	0.726	1.002	1013.420	740.488

模型捕获和还原了参数



七、如何开始

配置环境

- 第1步，安装 R
- 第2步，安装 Rstudio
- 第3步，安装 Rtools42到C:\rtools42，(苹果系统不需要这一步)
- 第4步，安装 CmdStanR

```
install.packages("cmdstanr", repos = c("https://mc-stan.org/r-packages/",getOption("repos")))
```

- 下载运行，<https://github.com/perlatex/Stan-in-Action/>

参考书籍

- <https://mc-stan.org/>
- <https://discourse.mc-stan.org/>
- Gelman, Andrew, John B. Carlin, Hal S. Stern, David B. Dunson, Aki Vehtari, and Donald B. Rubin. 2013. *Bayesian Data Analysis*, Third Edition. Boca Raton: Chapman; Hall/CRC.
- Kruschke, John K. 2014. *Doing Bayesian Data Analysis: A Tutorial Introduction with R*. 2nd Edition. Burlington, MA: Academic Press.
- McElreath, Richard. 2020. *Statistical Rethinking: A Bayesian Course with Examples in R and Stan*. 2nd ed. CRC Texts in Statistical Science. Boca Raton: Taylor; Francis, CRC Press.

感谢 Stan 语言之美!

本幻灯片由 R 包 **xaringan** 和 **flipbookr** 生成