INTRODUCTION TO BAYESIAN STATISTICS

Arnaud Legrand
Univ. Grenoble Alpes, CNRS, Inria, Grenoble INP
POLARIS days, May 2019
Given a some observations, how to generate “similar” values?

- GANs? No, not today
- How to leverage your knowledge about the system?
- How to check whether it is reasonable or not?

This talk:

1. Brief introduction to Bayesian statistics.
2. Brief introduction to Bayesian sampling
3. Brief presentation of STAN
Bayesian Statistics
Model: we assume $y \sim M(\theta, x)$

- $\theta$: Model parameters
- $y$: Dependent data (response)
- $x$: Independent data (covariates/predictors/constants)

Examples:

- $y \sim N(\mu, \sigma)$
- $y \sim x^2 + U(\alpha, \beta)$
- $y \sim N(\alpha x + \beta, \sigma)$
- $y \sim N(\alpha \log(x) + \beta, \alpha' x + \beta')$

Everyone: Model data as random
Notation

- \( p(A) \) = probability that \( A \) occurs
- \( p(A, B) \) = probability that \( A \) and \( B \) occurs
- \( p(A|B) \) = probability that \( A \) occurs, given that \( B \) occurs
Notation

- $p(A) = \text{probability that } A \text{ occurs}$
- $p(A, B) = \text{probability that } A \text{ and } B \text{ occurs}$
- $p(A|B) = \text{probability that } A \text{ occurs, given that } B \text{ occurs}$

Conjunction rule

- $p(A, B) = p(A|B)p(B)$
- $p(B, A) = p(B|A)p(A)$
Notation

- \( p(A) \) = probability that \( A \) occurs
- \( p(A, B) \) = probability that \( A \) and \( B \) occurs
- \( p(A|B) \) = probability that \( A \) occurs, given that \( B \) occurs

Conjunction rule

- \( p(A, B) = p(A|B)p(B) \)
- \( p(B, A) = p(B|A)p(A) \)

Bayes rule  Equate and divide by \( p(B) \)

\[
p(A|B) = \frac{p(B|A)p(A)}{p(B)}
\]
Bayesians: Data is fixed (observed), model parameters as random

\[ p(\theta, y, x) = p(y, \theta, x) \]
\[ p(\theta | y, x)p(y, x) = p(y | \theta, x)p(\theta, x) \]

Hence \( p(\theta | y, x) = \frac{p(y | \theta, x)p(\theta, x)}{p(y, x)} = \frac{p(y | \theta, x)p(\theta)p(x)}{p(y, x)} \)
\[ \propto p(y | \theta, x)p(\theta) \quad (y, \text{ and } x \text{ are fixed for a given data set}) \]
Bayesian Statistics

Bayesians: Data is fixed (observed), model parameters as random

\[ p(\theta, y, x) = p(y, \theta, x) \]
\[ p(\theta|y, x)p(y, x) = p(y|\theta, x)p(\theta, x) \]

Hence

\[ p(\theta|y, x) = \frac{p(y|\theta, x)p(\theta, x)}{p(y, x)} = \frac{p(y|\theta, x)p(\theta)p(x)}{p(y, x)} \]
\[ \propto p(y|\theta, x)p(\theta) \] (y, and x are fixed for a given data set)

Bayes rule

\[ p(\theta|y, x) \propto p(y|\theta, x)p(\theta, x) \] assuming \( y \sim M(\theta, x) \)

- **Posterior**: The answer, probability distributions of parameters
- **Likelihood**: A (model specific) computable function of the parameters
- **Prior**: "Initial guess", existing knowledge of the system

The key to building Bayesian models is specifying the likelihood function, same as frequentist.
A € problem
When spun on edge 250 times, a Belgian 1€ coin came up heads 140 times and tails 110. It looks very suspicious to me. If the coin were unbiased, the chance of getting a result as extreme as that would be less than 7%.

– From "The Guardian" quoted by MacKay in Information Theory, Inference and Learning Algorithms

• Model: \( Y \sim B(\pi) \)
• Data: \( y = 1, 0, 1, 1, 0, 0, 1, 1, \ldots \)
• \( p(y|\pi = 1/2) \)
  \[
  = \frac{(140+110)!}{110!140!} \cdot \left(\frac{1}{2}\right)^{110} \cdot \left(\frac{1}{2}\right)^{140}
  \approx 0.00835
  \]
When spun on edge 250 times, a Belgian 1€ coin came up heads 140 times and tails 110. It looks very suspicious to me. If the coin were unbiased, the chance of getting a result as extreme as that would be less than 7%.

– From “The Gardian” quoted by MacKay in Information Theory, Inference and Learning Algorithms

- Model: \( Y \sim B(\pi) \)
- Data: \( y = 1, 0, 1, 1, 0, 0, 1, 1, \ldots \)
- \( p(y|\pi \leq 1/2) \)
  \[ p(y|\pi \leq 1/2) = \sum_{k \leq 110} \frac{250!}{k!(250-k)!} \cdot \frac{1}{2^{250}} \approx 0.033 \]
When spun on edge 250 times, a Belgian 1€ coin came up heads 140 times and tails 110. It looks very suspicious to me. If the coin were unbiased, the chance of getting a result as extreme as that would be less than 7%.

– From ”The Guardian” quoted by MacKay in Information Theory, Inference and Learning Algorithms

- Model: $Y \sim \mathcal{B}(\pi)$
- Data: $y = 1, 0, 1, 1, 0, 0, 1, 1, 1, \ldots$
- Prior: $\pi \sim \mathcal{U}(0, 1)$

$$p(\pi|y) = \frac{p(y|\pi) \cdot p(\pi)}{p(y)} = \frac{(1 - \pi)^{n_0} \pi^{n_1} \cdot 1}{n_0!n_1!/(n_0 + n_1 + 1)!}$$
When spun on edge 250 times, a Belgian 1€ coin came up heads 140 times and tails 110. It looks very suspicious to me. If the coin were unbiased, the chance of getting a result as extreme as that would be less than 7%.

– From "The Guardian" quoted by MacKay in Information Theory, Inference and Learning Algorithms

- Model: $Y \sim \mathcal{B}(\pi)$
- Data: $y = 1, 0, 1, 1, 0, 0, 1, 1, 1, \ldots$
- Prior: $\pi \sim \mathcal{U}(0, 1)$

$$p(\pi|y) = \frac{p(y|\pi) \cdot p(\pi)}{p(y)} = \frac{(1 - \pi)^{n_0} \pi^{n_1} \cdot 1}{n_0!n_1!/(n_0 + n_1 + 1)!}$$
When spun on edge 250 times, a Belgian 1€ coin came up heads 140 times and tails 110. It looks very suspicious to me. If the coin were unbiased, the chance of getting a result as extreme as that would be less than 7%.

– From "The Guardian” quoted by MacKay in Information Theory, Inference and Learning Algorithms

• Model: \( Y \sim B(\pi) \)
• Data: \( y = 1, 0, 1, 1, 0, 0, 1, 1, 1, \ldots \)
• Prior: \( \pi \sim \mathcal{U}(0, 1) \)

\[
p(\pi|y) = \frac{p(y|\pi) \cdot p(\pi)}{p(y)} = \frac{(1 - \pi)^{n_0} \pi^{n_1} \cdot 1}{n_0! n_1! / (n_0 + n_1 + 1)!}
\]
When spun on edge 250 times, a Belgian 1€ coin came up heads 140 times and tails 110. It looks very suspicious to me. If the coin were unbiased, the chance of getting a result as extreme as that would be less than 7%.

– From "The Guardian" quoted by MacKay in Information Theory, Inference and Learning Algorithms

- Model: $Y \sim \mathcal{B}(\pi)$
- Data: $y = 1, 0, 1, 1, 0, 0, 1, 1, 1, \ldots$
- Prior: $\pi \sim \mathcal{U}(0, 1)$

$$p(\pi|y) = \frac{p(y|\pi) \cdot p(\pi)}{p(y)} = \frac{(1 - \pi)^{n_0}\pi^{n_1} \cdot 1}{n_0!n_1!/(n_0 + n_1 + 1)!}$$
When spun on edge 250 times, a Belgian 1€ coin came up heads 140 times and tails 110. It looks very suspicious to me. If the coin were unbiased, the chance of getting a result as extreme as that would be less than 7%.

– From "The Gardian" quoted by MacKay in Information Theory, Inference and Learning Algorithms

- Model: \( Y \sim \mathcal{B}(\pi) \)
- Data: \( y = 1, 0, 1, 1, 0, 0, 1, 1, 1, \ldots \)
- Prior: \( \pi \sim \mathcal{U}(0, 1) \)

\[
p(\pi|y) = \frac{p(y|\pi) \cdot p(\pi)}{p(y)} = \frac{(1 - \pi)^{n_0} \pi^{n_1} \cdot 1}{n_0! n_1!/(n_0 + n_1 + 1)!}
\]
When spun on edge 250 times, a Belgian 1€ coin came up heads 140 times and tails 110. It looks very suspicious to me. If the coin were unbiased, the chance of getting a result as extreme as that would be less than 7%.

– From "The Gardian" quoted by MacKay in Information Theory, Inference and Learning Algorithms

- Model: $Y \sim B(\pi)$
- Data: $y = 1, 0, 1, 1, 0, 0, 1, 1, 1, \ldots$
- Prior: $\pi \sim U(0, 1)$

\[
p(\pi|y) = \frac{p(y|\pi) \cdot p(\pi)}{p(y)} = \frac{(1 - \pi)^{n_0} \pi^{n_1} \cdot 1}{n_0!n_1!/(n_0 + n_1 + 1)!}
\]
When spun on edge 250 times, a Belgian 1€ coin came up heads 140 times and tails 110. It looks very suspicious to me. If the coin were unbiased, the chance of getting a result as extreme as that would be less than 7%.

– From ”The Gardian” quoted by MacKay in Information Theory, Inference and Learning Algorithms

• Model: \( Y \sim B(\pi) \)
• Data: \( y = 1, 0, 1, 1, 0, 0, 1, 1, 1, \ldots \)
• Prior: \( \pi \sim U(0, 1) \)

\[
p(\pi|y) = \frac{p(y|\pi) \cdot p(\pi)}{p(y)} = \frac{(1 - \pi)^{n_0} \pi^{n_1} \cdot 1}{n_0!n_1!/(n_0 + n_1 + 1)!}
\]
When spun on edge 250 times, a Belgian 1€ coin came up heads 140 times and tails 110. It looks very suspicious to me. If the coin were unbiased, the chance of getting a result as extreme as that would be less than 7%.

– From “The Guardian” quoted by MacKay in Information Theory, Inference and Learning Algorithms

- Model: $Y \sim \mathcal{B}(\pi)$
- Data: $y = 1, 0, 1, 1, 0, 0, 1, 1, 1, \ldots$
- Prior: $\pi \sim \mathcal{U}(0, 1)$

\[
p(\pi|y) = \frac{p(y|\pi) \cdot p(\pi)}{p(y)} = \frac{(1 - \pi)^{n_0} \pi^{n_1} \cdot 1}{n_0!n_1!/(n_0 + n_1 + 1)!}
\]
When spun on edge 250 times, a Belgian 1€ coin came up heads 140 times and tails 110. It looks very suspicious to me. If the coin were unbiased, the chance of getting a result as extreme as that would be less than 7%.

– From ”The Gardian” quoted by MacKay in Information Theory, Inference and Learning Algorithms

• Model: \( Y \sim B(\pi) \)
• Data: \( y = 1, 0, 1, 1, 0, 0, 1, 1, 1, \ldots \)
• Prior: \( \pi \sim U(0, 1) \)

\[
p(\pi|y) = \frac{p(y|\pi) \cdot p(\pi)}{p(y)} = \frac{(1 - \pi)^{n_0} \pi^{n_1} \cdot 1}{n_0! n_1! / (n_0 + n_1 + 1)!}
\]
When spun on edge 250 times, a Belgian 1€ coin came up heads 140 times and tails 110. It looks very suspicious to me. If the coin were unbiased, the chance of getting a result as extreme as that would be less than 7%.

– From "The Gardian" quoted by MacKay in Information Theory, Inference and Learning Algorithms

- Model: $Y \sim B(\pi)$
- Data: $y = 1, 0, 1, 1, 0, 0, 1, 1, 1, \ldots$
- Prior: $\pi \sim U(0, 1)$

\[
p(\pi|y) = \frac{p(y|\pi) \cdot p(\pi)}{p(y)} = \frac{(1 - \pi)^{n_0} \pi^{n_1} \cdot 1}{n_0! n_1! / (n_0 + n_1 + 1)!}
\]
When spun on edge 250 times, a Belgian 1€ coin came up heads 140 times and tails 110. It looks very suspicious to me. If the coin were unbiased, the chance of getting a result as extreme as that would be less than 7%.

– From "The Guardian" quoted by MacKay in Information Theory, Inference and Learning Algorithms

- Model: \( Y \sim \mathcal{B}(\pi) \)
- Data: \( y = 1, 0, 1, 1, 0, 0, 1, 1, 1, \ldots \)
- Prior: \( \pi \sim \mathcal{U}(0, 1) \)

\[
p(\pi|y) = \frac{p(y|\pi) \cdot p(\pi)}{p(y)} = \frac{(1 - \pi)^{n_0} \pi^{n_1} \cdot 1}{n_0! n_1! / (n_0 + n_1 + 1)!}
\]
When spun on edge 250 times, a Belgian 1€ coin came up heads 140 times and tails 110. It looks very suspicious to me. If the coin were unbiased, the chance of getting a result as extreme as that would be less than 7%.

– From "The Gardian" quoted by MacKay in *Information Theory, Inference and Learning Algorithms*

- **Model:** \( Y \sim B(\pi) \)
- **Data:** \( y = 1, 0, 1, 1, 0, 0, 1, 1, 1, \ldots \)
- **Prior:** \( \pi \sim U(0, 1) \)

\[
p(\pi | y) = \frac{p(y | \pi) \cdot p(\pi)}{p(y)} = \frac{(1 - \pi)^{n_0} \pi^{n_1} \cdot 1}{n_0! n_1! / (n_0 + n_1 + 1)!}
\]
When spun on edge 250 times, a Belgian 1€ coin came up heads 140 times and tails 110. It looks very suspicious to me. If the coin were unbiased, the chance of getting a result as extreme as that would be less than 7%.

– From "The Gardian" quoted by MacKay in Information Theory, Inference and Learning Algorithms

- Model: $Y \sim \mathcal{B}(\pi)$
- Data: $y = 1, 0, 1, 1, 0, 0, 1, 1, 1, \ldots$
- Prior: $\pi \sim \mathcal{T}(0, 1)$

$$p(\pi | y) = \frac{p(y | \pi) \cdot p(\pi)}{p(y)} = \frac{(1 - \pi)^{n_0} \pi^{n_1} \cdot (2 - 4|\pi - 0.5|)}{\text{some normalization}}$$
When spun on edge 250 times, a Belgian 1€ coin came up heads 140 times and tails 110. It looks very suspicious to me. If the coin were unbiased, the chance of getting a result as extreme as that would be less than 7%.

– From "The Gardian" quoted by MacKay in Information Theory, Inference and Learning Algorithms

- Model: $Y \sim \mathcal{B}(\pi)$
- Data: $y = 1, 0, 1, 1, 0, 0, 1, 1, 1, \ldots$
- Prior: $\pi \sim \mathcal{T}(0, 1)$

$$p(\pi|y) = \frac{p(y|\pi) \cdot p(\pi)}{p(y)} = \frac{(1 - \pi)^{n_0} \pi^{n_1} \cdot (2 - 4|\pi - 0.5|)}{\text{some normalization}}$$
When spun on edge 250 times, a Belgian 1€ coin came up heads 140 times and tails 110. It looks very suspicious to me. If the coin were unbiased, the chance of getting a result as extreme as that would be less than 7%.

– From "The Guardian" quoted by MacKay in Information Theory, Inference and Learning Algorithms

- Model: $Y \sim \mathcal{B}(\pi)$
- Data: $y = 1, 0, 1, 1, 0, 0, 1, 1, 1, \ldots$
- Prior: $\pi \sim \mathcal{T}(0, 1)$

\[
p(\pi|y) = \frac{p(y|\pi) \cdot p(\pi)}{p(y)} = \frac{(1 - \pi)^{n_0} \pi^{n_1} \cdot (2 - 4|\pi - 0.5|)}{\text{some normalization}}
\]
When spun on edge 250 times, a Belgian 1€ coin came up heads 140 times and tails 110. It looks very suspicious to me. If the coin were unbiased, the chance of getting a result as extreme as that would be less than 7%.

– From "The Gardian" quoted by MacKay in Information Theory, Inference and Learning Algorithms

• Model: $Y \sim B(\pi)$
• Data: $y = 1, 0, 1, 1, 0, 0, 1, 1, 1, \ldots$
• Prior: $\pi \sim T(0, 1)$

$$p(\pi|y) = \frac{p(y|\pi) \cdot p(\pi)}{p(y)} = \frac{(1 - \pi)^{n_0} \pi^{n_1} \cdot (2 - 4|\pi - 0.5|)}{\text{some normalization}}$$
When spun on edge 250 times, a Belgian 1€ coin came up heads 140 times and tails 110. It looks very suspicious to me. If the coin were unbiased, the chance of getting a result as extreme as that would be less than 7%.

– From "The Gardian" quoted by MacKay in Information Theory, Inference and Learning Algorithms

- Model: \( Y \sim \mathcal{B}(\pi) \)
- Data: \( y = 1, 0, 1, 1, 0, 0, 1, 1, 1, \ldots \)
- Prior: \( \pi \sim \mathcal{T}(0, 1) \)

\[
p(\pi|y) = \frac{p(y|\pi) \cdot p(\pi)}{p(y)} = \frac{(1 - \pi)^{n_0} \pi^{n_1} \cdot (2 - 4|\pi - 0.5|)}{\text{some normalization}}
\]
When spun on edge 250 times, a Belgian 1€ coin came up heads 140 times and tails 110. It looks very suspicious to me. If the coin were unbiased, the chance of getting a result as extreme as that would be less than 7%.

– From "The Guardian" quoted by MacKay in *Information Theory, Inference and Learning Algorithms*

- Model: $Y \sim \mathcal{B}(\pi)$
- Data: $y = 1, 0, 1, 1, 0, 0, 1, 1, 1, \ldots$
- Prior: $\pi \sim \mathcal{T}(0, 1)$

$$p(\pi|y) = \frac{p(y|\pi) \cdot p(\pi)}{p(y)} = \frac{(1 - \pi)^{n_0} \pi^{n_1} \cdot (2 - 4|\pi - 0.5|)}{\text{some normalization}}$$
When spun on edge 250 times, a Belgian 1€ coin came up heads 140 times and tails 110. It looks very suspicious to me. If the coin were unbiased, the chance of getting a result as extreme as that would be less than 7%.

– From "The Guardian" quoted by MacKay in Information Theory, Inference and Learning Algorithms

- Model: $Y \sim \mathcal{B}(\pi)$
- Data: $y = 1, 0, 1, 1, 0, 0, 1, 1, 1, \ldots$
- Prior: $\pi \sim \mathcal{T}(0, 1)$

$$p(\pi|y) = \frac{p(y|\pi) \cdot p(\pi)}{p(y)} = \frac{(1 - \pi)^{n_0} \pi^{n_1} \cdot (2 - 4|\pi - 0.5|)}{\text{some normalization}}$$
When spun on edge 250 times, a Belgian 1€ coin came up heads 140 times and tails 110. It looks very suspicious to me. If the coin were unbiased, the chance of getting a result as extreme as that would be less than 7%.

– From "The Guardian" quoted by MacKay in Information Theory, Inference and Learning Algorithms

- Model: $Y \sim \mathcal{B}(\pi)$
- Data: $y = 1, 0, 1, 1, 0, 0, 1, 1, 1, \ldots$
- Prior: $\pi \sim \mathcal{T}(0, 1)$

\[
p(\pi|y) = \frac{p(y|\pi) \cdot p(\pi)}{p(y)} = \frac{(1 - \pi)^{n_0} \pi^{n_1} \cdot (2 - 4|\pi - 0.5|)}{\text{some normalization}}
\]
When spun on edge 250 times, a Belgian 1€ coin came up heads 140 times and tails 110. It looks very suspicious to me. If the coin were unbiased, the chance of getting a result as extreme as that would be less than 7%.

– From "The Guardian" quoted by MacKay in Information Theory, Inference and Learning Algorithms

- Model: $Y \sim \mathcal{B}(\pi)$
- Data: $y = 1, 0, 1, 1, 0, 0, 1, 1, 1, \ldots$
- Prior: $\pi \sim \mathcal{T}(0, 1)$

$$p(\pi|y) = \frac{p(y|\pi) \cdot p(\pi)}{p(y)} = \frac{(1 - \pi)^{n_0} \pi^{n_1} \cdot (2 - 4|\pi - 0.5|)}{\text{some normalization}}$$
When spun on edge 250 times, a Belgian 1€ coin came up heads 140 times and tails 110. It looks very suspicious to me. If the coin were unbiased, the chance of getting a result as extreme as that would be less than 7%.

– From "The Guardian" quoted by MacKay in Information Theory, Inference and Learning Algorithms

- Model: \( Y \sim \mathcal{B}(\pi) \)
- Data: \( y = 1, 0, 1, 1, 0, 0, 1, 1, 1, \ldots \)
- Prior: \( \pi \sim \mathcal{T}(0, 1) \)

\[
p(\pi|y) = \frac{p(y|\pi) \cdot p(\pi)}{p(y)} = \frac{(1 - \pi)^{n_0} \pi^{n_1} \cdot (2 - 4|\pi - 0.5|)}{\text{some normalization}}
\]
When spun on edge 250 times, a Belgian 1€ coin came up heads 140 times and tails 110. It looks very suspicious to me. If the coin were unbiased, the chance of getting a result as extreme as that would be less than 7%.

– From “The Guardian” quoted by MacKay in Information Theory, Inference and Learning Algorithms

Model: $Y \sim B(\pi)$

Data: $y = 1, 0, 1, 1, 0, 0, 1, 1, 1, \ldots$

Prior: $\pi \sim \mathcal{T}(0, 1)$

$$p(\pi | y) = \frac{p(y | \pi) \cdot p(\pi)}{p(y)} = \frac{(1 - \pi)^{n_0} \pi^{n_1} \cdot (2 - 4|\pi - 0.5|)}{\text{some normalization}}$$
A Simple Gaussian Model
Initial Belief and First Observations

- Model: $Y \sim \mathcal{N}(\mu, \sigma)$
- Prior: $\mu \sim \mathcal{U}(0, 20)$ and $\sigma \sim \mathcal{U}(0, 5)$

```r
set.seed(162);
n = 20; mu = 12.5; sigma = 1.6;
Y = rnorm(n, mean = mu, sd = sigma);
```
• Model: $Y \sim \mathcal{N}(\mu, \sigma)$
• Prior: $\mu \sim \mathcal{U}(0, 20)$ and $\sigma \sim \mathcal{U}(0, 5)$

```r
set.seed(162);
n=20; mu=12.5; sigma=1.6;
Y=rnorm(n, mean=mu, sd=sigma);
Y
```

Model: $Y \sim \mathcal{N}(\mu, \sigma)$, hence $p(y|\mu, \sigma) = \frac{1}{\sigma \sqrt{2\pi}} \exp \left( -\frac{1}{2} \left( \frac{y-\mu}{2} \right)^2 \right)$

Therefore $p(\mu, \sigma|y) \propto \prod_{i=1}^{n} \frac{1}{\sigma \sqrt{2\pi}} \exp \left( -\frac{1}{2} \left( \frac{y_i - \mu}{2} \right)^2 \right) \cdot \frac{1}{100}$
Mean: 12.07348806679
Standard Deviation: 1.70127707382769

Distribution of observations $Y$

Posterior distribution
Exploiting information (Normal model)

[1] "Mean: 12.07348806679"
[1] "Standard Deviation: 1.70127707382769"

Distribution of observations $Y$

Posterior distribution
EXPLOITING INFORMATION (NORMAL MODEL)

[1] "Mean: 12.07348806679"
[1] "Standard Deviation: 1.70127707382769"

Distribution of observations Y

Posterior distribution
EXPLOITING INFORMATION (NORMAL MODEL)

[1] "Mean: 12.07348806679"
[1] "Standard Deviation: 1.70127707382769"

Distribution of observations \( Y \)

Posterior distribution
Exploiting information (Normal model)

[1] "Mean: 12.07348806679"
[1] "Standard Deviation: 1.70127707382769"

Distribution of observations $Y$

Posterior distribution
Exploiting information (Normal model)

[1] "Mean: 12.07348806679"
[1] "Standard Deviation: 1.70127707382769"

Distribution of observations $Y$

Posterior distribution
Exploiting Information (Normal Model)

1. "Mean: 12.07348806679"
2. "Standard Deviation: 1.70127707382769"

Distribution of observations $Y$

Posterior distribution
EXPLOITING INFORMATION (NORMAL MODEL)

[1] "Mean: 12.07348806679"
[1] "Standard Deviation: 1.70127707382769"

Distribution of observations $Y$

Posterior distribution (Zoom)
Single point estimate (Normal model)

[1] "Mean: 12.07348806679"
[1] "Standard Deviation: 1.70127707382769"

\[ p(\mu, \sigma | y) \propto \prod_{i=1}^{n} \frac{1}{\sigma \sqrt{2\pi}} \exp \left( -\frac{1}{2} \left( \frac{y_i - \mu}{\sigma} \right)^2 \right) \cdot \frac{1}{100} \]

- **Machine Learning:** Maximum Likelihood |y
  - \( \mu_{MLE} = \frac{1}{n} \sum_{i=1}^{n} y_i \)
  - \( \sigma_{MLE} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - \mu_{MLE})^2} \)

- **Frequentist:** ensure \( \mathbb{E}[\mu_F] = \mu \) and \( \mathbb{E}[\sigma_F^2] = \sigma^2 \)
  - \( \mu_F = \frac{1}{n} \sum_{i=1}^{n} y_i \)
  - \( \sigma_F = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (y_i - \mu_F)^2} \)

- **Bayesian:** sample the posterior

9/24
• $\theta$: unknown parameter ($\mu = 12.5$, $\sigma = 1.6$)
• $y$: observation
• $\hat{\theta}$: single point estimate of $\theta$ ($\mu \approx 12.07$, $\sigma \approx 1.7$)
• $\tilde{y}$: future observations

Generating $\tilde{y}$ from $\hat{\theta}$

(does not account for the uncertainty on $\hat{\theta}$)
Generating new data

- $\theta$: unknown parameter ($\mu = 12.5$, $\sigma = 1.6$)
- $y$: observation
- $\hat{\theta}$: single point estimate of $\theta$ ($\mu \approx 12.07$, $\sigma \approx 1.7$)
- $\tilde{y}$: future observations

Generating $\tilde{y}$ from $\hat{\theta}$

(does not account for the uncertainty on $\hat{\theta}$)

Generating $\tilde{y}$ from many $\tilde{\theta} | y$

Noise on $y$ + uncertainty on $\theta$
Influence of the Prior

Take away messages:

1. With enough data, reasonable people converge.
2. If any $p(\theta) = 0$, no data will change that
   - Sometimes imposing $p(\theta) = 0$ is nice (e.g., $\theta > 0$)
3. An uninformative prior is better than a wrong highly (supposedly) informative prior.
4. With conjugate priors, calculus of the likelihood is possible
   Otherwise, the normalization is a huge pain

Computing confidence intervals, high density regions, expectation of complex functions... Samples are easier to use than distributions.

**BUGS**: Bayesian inference Using Gibbs Sampling

$$p(\theta|y, x) \propto p(y|\theta, x)p(\theta, x)$$

<table>
<thead>
<tr>
<th>Posterior</th>
<th>Likelihood</th>
<th>Prior</th>
</tr>
</thead>
</table>
Bayesian Sampling
Generating random number: direct method

- Input:
  - \( U(0, 1) \)
- A target density \( f_Y \)
- 3 Easy steps:
  1. Compute \( F_Y(t) = \int_{-\infty}^{t} f_Y(y) \, dy \)
  2. Compute the inverse \( F_Y^{-1} \)
  3. Apply \( F_Y^{-1} \) to your uniform numbers

Step 1 is generally quite complicated. The prior makes it even worse. Multi-dimensional densities: just as complicated unless the law has a very particular structure
**Rejection method**

Assume we have $M$ and $g$, s.t. $p(\theta|y) \leq Mg(\theta)$

- Draw $\theta \sim g$ and accept with probability $\frac{p(\theta|y)}{Mg(\theta)}$

Works well if $Mg$ is a good approximation of $p(.|y)$

**Issues:**

- $p$ is multiplied by the prior. Where is the max? Which $g$, which $M$?
- Is the landscape flat, hilly, spiky?
- Rejection can be quite inefficient ($\rightsquigarrow$ Importance sampling)
Monte Carlo Markov Chain simulation

Dimension by dimension (Gibbs sampler): $\theta^t_j \sim p(.|\theta_{j-1}^t, y)$
Monte Carlo Markov Chain simulation

Dimension by dimension (Gibbs sampler): \( \theta_j^t \sim p(\cdot | \theta_{-j}^{t-1}, y) \)

Metropolis-Hasting: Jumping distribution \( J \)

\[
\begin{align*}
\cdot & \quad \theta^* \sim J(\theta^{t-1}) \quad r = \frac{p(\theta^* | y)}{p(\theta^{t-1} | y)} \quad \theta^t = \begin{cases} 
\theta^* & \text{with proba. } \min(r, 1) \\
\theta^{t-1} & \text{otherwise}
\end{cases}
\end{align*}
\]

Look for high density areas

- Highly skewed (short/long-tail) or multi-modal are problematic
- Transformation, reparameterization, auxiliary variables, simulated tempering,
- Trans-dimensional Markov chains: the dimension of the parameter space can change from one iteration to the next
Try to eliminate the random walk inefficiency

- Add a momentum variable $\phi_j$ for each component $\theta_j$ and move to the right direction

Hamiltonian Monte-Carlo combines MCMC with deterministic optimization methods

- **Leapfrog**: $L$ steps of $\epsilon/2$ ($L\epsilon \approx 1$)
- No U-turn Sampler (**NUTS**): adapt step sizes locally, the trajectory continues until it turns around
Using STAN
What is Stan?

A probabilistic programming language implementing full Bayesian statistical inference with MCMC sampling (NUTS, HMC) and penalized maximum likelihood estimation with optimization (L-BFGS).

Stanislaw Ulam, namesake of Stan and co-inventor of Monte Carlo methods shown here holding the Fermiac, Enrico Fermi’s physical Monte Carlo simulator for neutron diffusion.

Bayesian Data Analysis, Gelman et al., 2013
Bayesian Course with examples in R and Stan, Richard McElreath, 2015
A SIMPLE EXAMPLE

ggplot(df, aes(x, y)) + geom_point(alpha=0.3) + theme_bw()
library(rstan)

modelString = "data { // the observations
  int<lower=1> N; // number of points
  vector[N] x;
  vector[N] y;
}
parameters { // what we want to find
  real intercept;
  real coefficient;
  real<lower=0> sigma; // indication: sigma cannot be negative
}
model {
  // We define our priors
  intercept ~ normal(0, 10); // We know that all the parameters follow a normal distribution
  coefficient ~ normal(0, 10);
  sigma ~ normal(0, 10);

  // Then, our likelihood function
  y ~ normal(coefficient*x + intercept, sigma);
  "
sm = stan_model(model_code = modelString)
Running STAN

```r
data = list(N=nrow(df), x=df$x, y=df$y)
fit = sampling(sm, data=data, iter=500, chains=8)
```

Sampling for model 'ea4b5a288cf5f1d87215860103a9026e' now (chain 1).

Chain 1: Gradient evaluation took 7.6e-05 seconds
Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.76 seconds.
Chain 1: Iteration: 1 / 500 [ 0%] (Warmup)
Chain 1: Iteration: 50 / 500 [ 10%] (Warmup)
Chain 1: Iteration: 100 / 500 [ 20%] (Warmup)
Chain 1: Iteration: 150 / 500 [ 30%] (Warmup)
Chain 1: Iteration: 200 / 500 [ 40%] (Warmup)
Chain 1: Iteration: 250 / 500 [ 50%] (Warmup)
Chain 1: Iteration: 251 / 500 [ 50%] (Sampling)
Chain 1: Iteration: 300 / 500 [ 60%] (Sampling)
Chain 1: Iteration: 350 / 500 [ 70%] (Sampling)
Chain 1: Iteration: 400 / 500 [ 80%] (Sampling)
Chain 1: Iteration: 450 / 500 [ 90%] (Sampling)
Chain 1: Iteration: 500 / 500 [100%] (Sampling)
Chain 1: Elapsed Time: 0.101632 seconds (Warm-up)
Chain 1: 0.044023 seconds (Sampling)
Chain 1: 0.145655 seconds (Total)

Sampling for model 'ea4b5a288cf5f1d87215860103a9026e' now (chain 2).

Chain 2: Gradient evaluation took 2e-05 seconds
Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.2 seconds.
Chain 2: Iteration: 1 / 500 [ 0%] (Warmup)
Chain 2: Iteration: 50 / 500 [ 10%] (Warmup)
Chain 2: Iteration: 100 / 500 [ 20%] (Warmup)
Chain 2: Iteration: 150 / 500 [ 30%] (Warmup)
Chain 2: Iteration: 200 / 500 [ 40%] (Warmup)
Chain 2: Iteration: 250 / 500 [ 50%] (Sampling)
Chain 2: Iteration: 251 / 500 [ 50%] (Sampling)
Chain 2: Iteration: 300 / 500 [ 60%] (Sampling)
Chain 2: Iteration: 350 / 500 [ 70%] (Sampling)
Chain 2: Iteration: 400 / 500 [ 80%] (Sampling)
Chain 2: Iteration: 450 / 500 [ 90%] (Sampling)
Chain 2: Iteration: 500 / 500 [100%] (Sampling)
Chain 2: Elapsed Time: 0.094653 seconds (Warm-up)
Chain 2: 0.044023 seconds (Sampling)
Chain 2: 0.125969 seconds (Total)
# Inspecting Results

```r
print(fit)
```

<table>
<thead>
<tr>
<th></th>
<th>mean</th>
<th>se_mean</th>
<th>sd</th>
<th>2.5%</th>
<th>25%</th>
<th>50%</th>
<th>75%</th>
<th>97.5%</th>
</tr>
</thead>
<tbody>
<tr>
<td>intercept</td>
<td>49.12</td>
<td>0.04</td>
<td>1.31</td>
<td>46.53</td>
<td>48.24</td>
<td>49.13</td>
<td>50.00</td>
<td>51.68</td>
</tr>
<tr>
<td>coefficient</td>
<td>-1.96</td>
<td>0.00</td>
<td>0.02</td>
<td>-2.01</td>
<td>-1.98</td>
<td>-1.96</td>
<td>-1.95</td>
<td>-</td>
</tr>
<tr>
<td>sigma</td>
<td>15.48</td>
<td>0.01</td>
<td>0.48</td>
<td>14.56</td>
<td>15.15</td>
<td>15.47</td>
<td>15.79</td>
<td>16.44</td>
</tr>
<tr>
<td>lp__</td>
<td>-1630.71</td>
<td>0.04</td>
<td>1.14</td>
<td>-1633.61</td>
<td>-1631.32</td>
<td>-1630.42</td>
<td>-1629.85</td>
<td>-1629.36</td>
</tr>
</tbody>
</table>

## Effective Sample Size and Potential Scale Reduction

<table>
<thead>
<tr>
<th></th>
<th>n_eff</th>
<th>Rhat</th>
</tr>
</thead>
<tbody>
<tr>
<td>intercept</td>
<td>997</td>
<td>1.00</td>
</tr>
<tr>
<td>coefficient</td>
<td>979</td>
<td>1.00</td>
</tr>
<tr>
<td>sigma</td>
<td>1057</td>
<td>1.00</td>
</tr>
<tr>
<td>lp__</td>
<td>840</td>
<td>1.01</td>
</tr>
</tbody>
</table>

Samples were drawn using NUTS(diag_e) at Wed May 22 22:30:52 2019. For each parameter, n_eff is a crude measure of effective sample size, and Rhat is the potential scale reduction factor on split chains (at convergence, Rhat=1).
Checking Convergence

\texttt{stan_trace(fit)}
**LOOKING AT SAMPLES**

```
stan_hist(fit)
```

```
ggplot(as.data.frame(rstan::extract(fit)),
      geom_point(aes(x=intercept, y=coefficient, color=sigma))
```

![Histograms of intercept, coefficient, and sigma.](image1)

![Scatter plot with color by sigma.](image2)
Wrap-up
Truth vs. Myths

Where it fails:

- Cover the space (e.g., high dimensions)
- Uninformed far away density spikes (mixtures requires informative models and priors)
- High quantiles/rare events

Informative priors and starting points may be difficult to come up with (use machine learning/simpler estimates first?)

Where it helps:

- Captures "correlations"
- Robust expectation estimation (1 simulation = very biased)
- Exploit all your knowledge about the system
- Uncertainty quantification with Monte Carlo
Disclaimer: I may be naive

**Digital twins** (haha!) of platforms in SimGrid
- As realistic as possible from observations
- Variations (what if?)

**Trace analysis:** expressive models of internal structure
- better information compression/summary
- state what is expected and detect when the system departs from it

**Anomaly detection:** performance non-regression testing (CI)
- Model the whole system (y in ”high” dimension)
- Detect novelty and continuously enrich the model and data