Stan:
Probabilistic Modeling & Bayesian Inference

Development Team

Andrew Gelman, **Bob Carpenter**, Daniel Lee, Ben Goodrich,
Michael Betancourt, Marcus Brubaker, Jiqiang Guo, Allen Riddell,
Marco Inacio, Jeffrey Arnold, **Mitzi Morris**, Rob Trangucci,
Rob Goedman, Brian Lau, Jonah Sol Gabry, Robert L. Grant,
Krzysztof Sakrejda, Aki Vehtari, Rayleigh Lei, Sebastian Weber,
Charles Margossian, Vincent Picaud, Imad Ali, **Sean Talts**,
Ben Bales, Ari Hartikainen, Matthijs Vàkàr, Andrew Johnson,
Dan Simpson

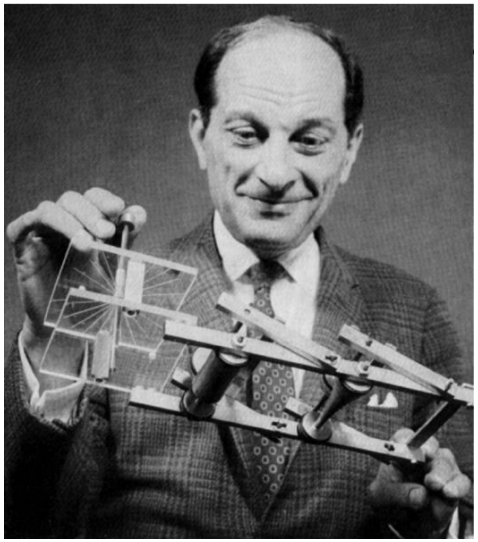
Stan 2.17 (November 2017) http://mc-stan.org

Stan

Who, What, and Why?

Who is Stan?

- Stan is named in honor of **Stanislaw Ulam** (1909–1984)
- Co-inventor of the **Monte Carlo method**



Ulam holding the Fermiac, Enrico Fermi's physical Monte Carlo simulator for random neutron diffusion;

image from G. C. Geisler (2000) Los Alamos report LA-UR-2532

What is Stan?

- Stan is an **imperative probabilistic programming language**
  - cf., BUGS: declarative; Church: functional; Figaro: OO
- Stan **program**: defines a probability model
  - declares data and (constrained) parameter variables
  - defines log posterior (or penalized likelihood)
- Stan **inference**: fits model to data & makes predictions
  - MCMC for full Bayesian inference
  - VB for approximate Bayesian inference
  - MLE for penalized maximum likelihood estimation

Why Choose Stan?

- **Expressive**
  - Stan is a full imperative programming language
  - continuously differentiable log densities
- **Robust**
  - usually works; signals when it doesn't
- **Efficient**
  - effective sample size / time (i.e., information)
  - multi-core and GPU code complete on branches
- Ongoing **open source** development
- **Community** support!

What's Next for Stan?

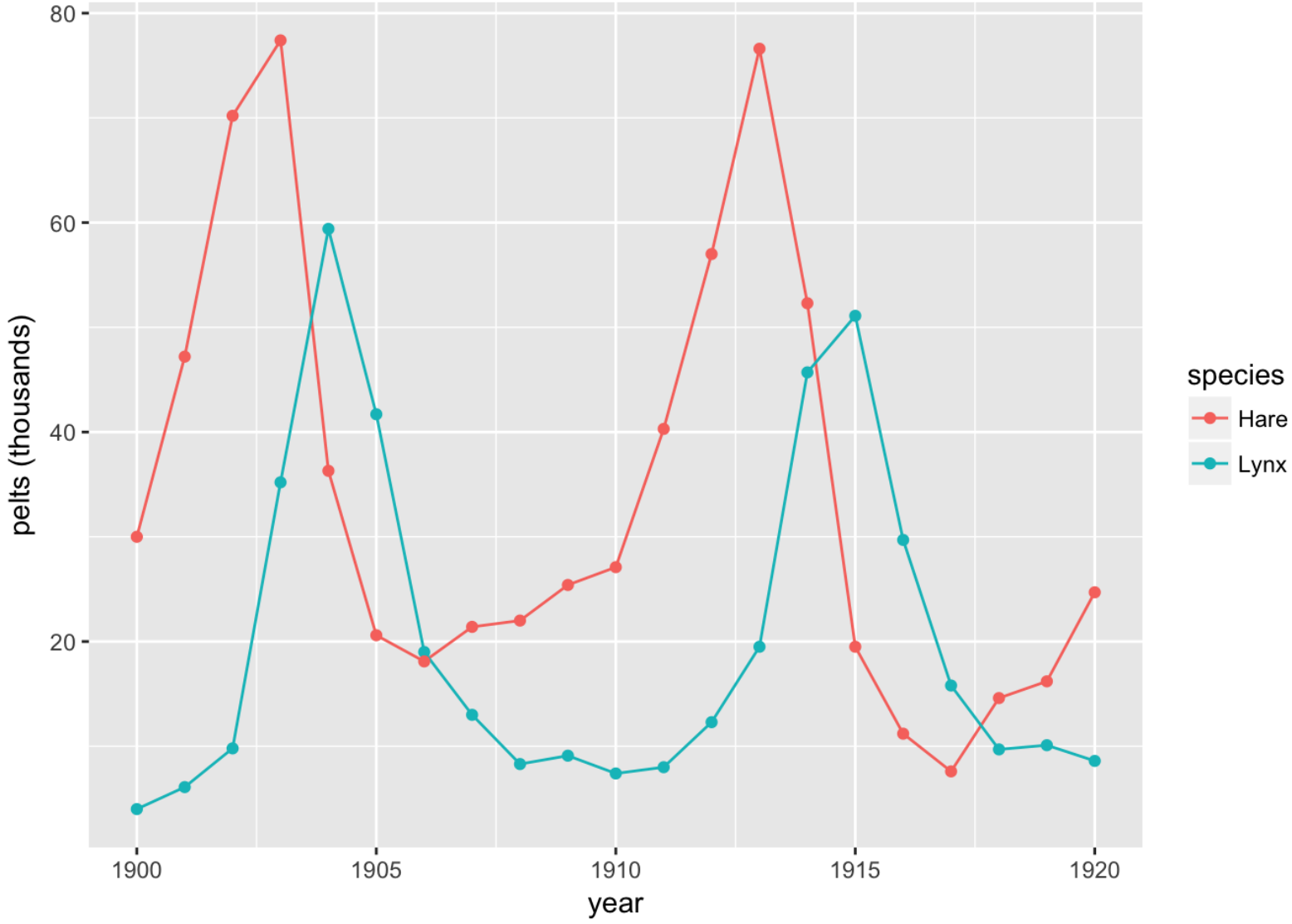
- Distributed likelihoods: **multi-CPU** (MPI)
- Big matrix operations: **GPU** (OpenCL)
- Sparse matrix operations
- Distributed data: asynchronous **expectation propagation**
- Approximations: parallel **max marginal mode**
- **Coursera** specialization
  - Gelman: Bayesian data analysis, Multilevel regression
  - Carpenter: Monte Carlo methods, Stan
  - Fall 2018

Predator-Prey Dynamics

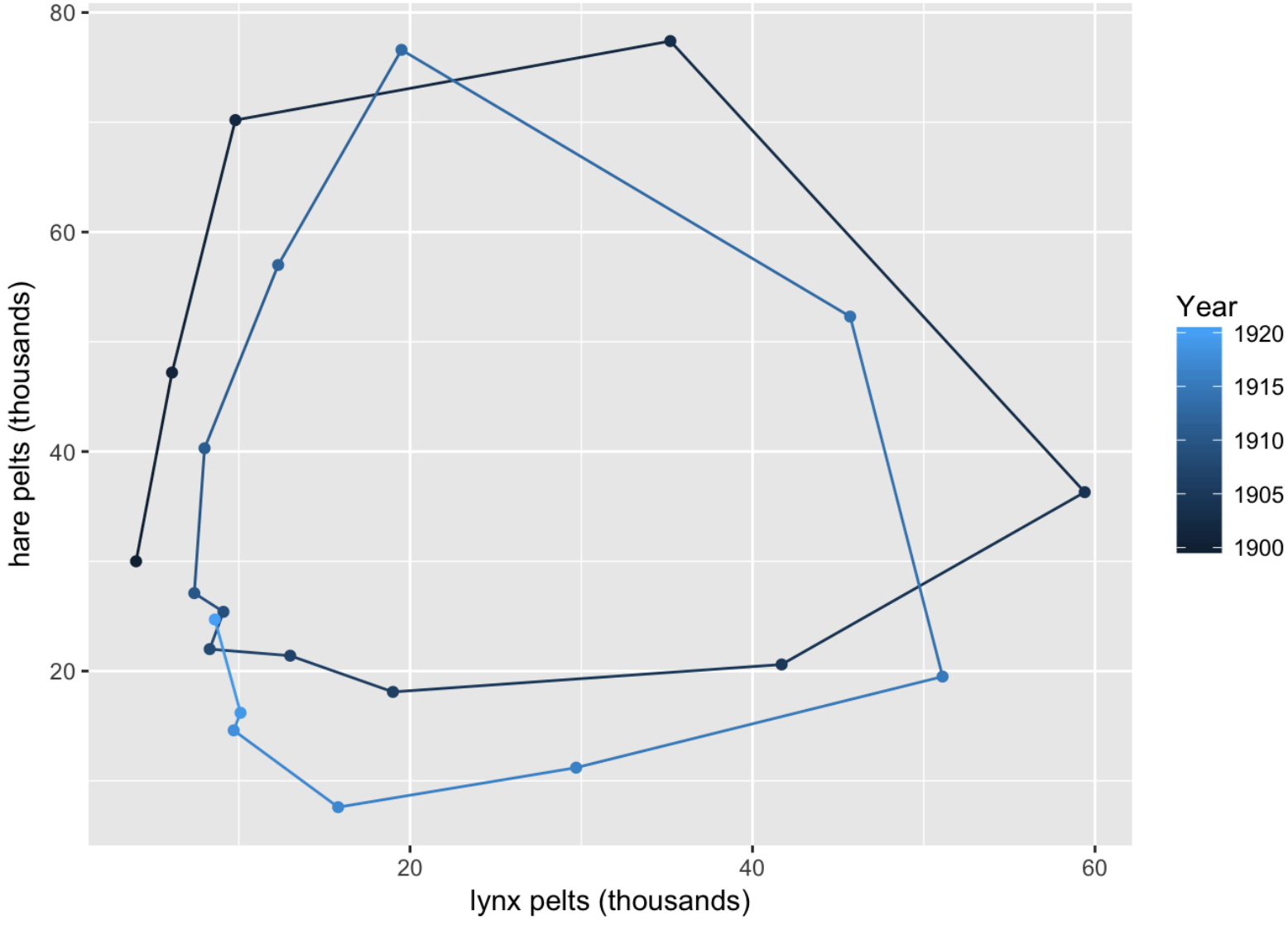
Lynxes and Hares

- **Snowshoe hare** (prey): herbivorous cousin of rabbits
- **Canadian lynx** (predator): feline eating primarily hares

Hudson Bay Co. Pelts, 1900–20



Pelts, Phase Space



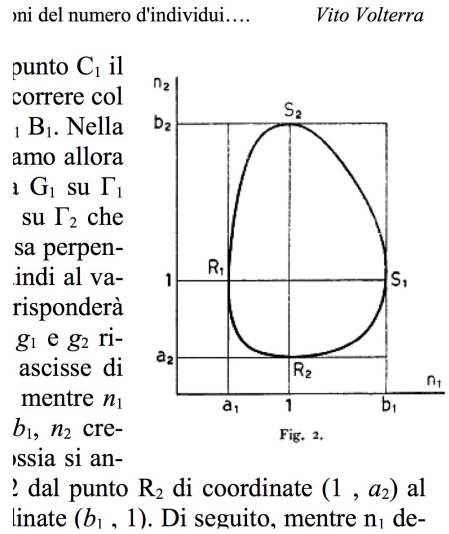
Volterra's (1927) Model

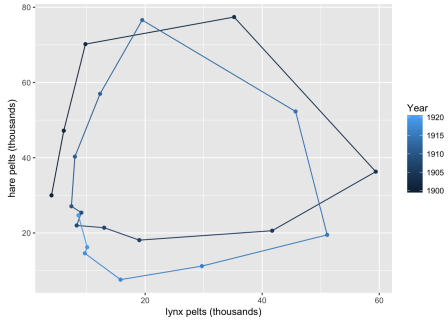
- population: $u(t)$ **prey**, $v(t)$ **predator**

$$\frac{\mathrm{d}}{\mathrm{d}t}u = (\alpha - \beta v)u$$

$$\frac{\mathrm{d}}{\mathrm{d}t}v = (-\gamma + \delta u)v$$

  - α: prey growth, intrinsic
  - β: prey shrinkage due to predation
  - γ: predator shrinkage, intrinsic
  - δ: predator growth from predation
- dynamics lead to **oscillation** as observed





Volterra, V., 1927. *Variazioni e fluttuazioni del numero d'individui in specie animali conviventi*. C. Ferrari.

Lotka-Volterra in Stan (dynamics)

```
real[] dz_dt(data real t,       // time (unused)
             real[] z,          // system state
             real[] theta,      // parameters
             data real[] x_r,   // real data (unused)
             data int[] x_i) {  // integer data (unused)
  real u = z[1];                // extract state
  real v = z[2];

  real alpha = theta[1];
  real beta = theta[2];
  real gamma = theta[3];
  real delta = theta[4];

  real du_dt = (alpha - beta * v) * u;
  real dv_dt = (-gamma + delta * u) * v;
  return { du_dt, dv_dt };
}
```

Data-Generating Model

- **Known** variables are observed
  - $y_{n,k}$: pelts for species k at times t_n for $n \in 0:N$
- **Unknown** variables must be inferred (**inverse problem**)
  - initial state: z_k^{init}: initial population for k
  - subsequent states $z_{n,k}$: population k at time t_n
  - parameters $\alpha, \beta, \gamma, \delta, \sigma > 0$
- **Likelihood** assumes errors are proportional (not additive)

$$y_{n,k} \sim \text{LogNormal}(\hat{z}_{n,k}, \sigma_k),$$

where $\hat{z}_n$ is **solution** at t_n to L-V diff eqs for initial z^{init}

equivalently: $\log y_{n,k} = \log \hat{z}_{n,k} + \epsilon_{n,k}$, with $\epsilon_{n,k} \sim \text{Normal}(0, \sigma_k)$

L-V in Stan (solution to ODE)

- Define variables for populations predicted by ode, given
  - system function (`dz_dt`), initial populations (`z0`)
  - initial time (`0.0`), solution times (`ts`)
  - parameters (`theta`), data arrays (unused: `rep_array(...)`)
  - tolerances (`1e-6`, `1-e4`), max iterations (`1e3`)

```
transformed parameters {
  real z[N, 2]
    = integrate_ode_rk45(dz_dt, z0, 0.0, ts, theta,
                         rep_array(0.0, 0), rep_array(0, 0),
                         1e-6, 1e-4, 1e3);
}
```

L-V in Stan (data, parameters)

- Variables for known constants, observed data

```
data {
  int<lower = 0> N;          // num measurements
  real ts[N];                // measurement times > 0
  real y0[2];                // initial pelts
  real<lower = 0> y[N, 2];   // subsequent pelts
}
```

- Variables for unknown parameters

```
parameters {
  real<lower = 0> theta[4];  // alpha, beta, gamma, delta
  real<lower = 0> z0[2];     // initial population
  real<lower = 0> sigma[2];  // scale of prediction error
}
```

L-V in Stan (priors, likelihood)

- Sampling statements for priors and likelihood

```
model {
  // priors
  sigma ~ lognormal(0, 0.5);
  theta[{1, 3}] ~ normal(1, 0.5);
  theta[{2, 4}] ~ normal(0.05, 0.05);

  z0[1] ~ lognormal(log(30), 5);
  z0[2] ~ lognormal(log(5), 5);

  // likelihood (lognormal)
  for (k in 1:2) {
    y0[k] ~ lognormal(log(z0[k]), sigma[k]);
    y[ , k] ~ lognormal(log(z[, k]), sigma[k]);
  }
}
```

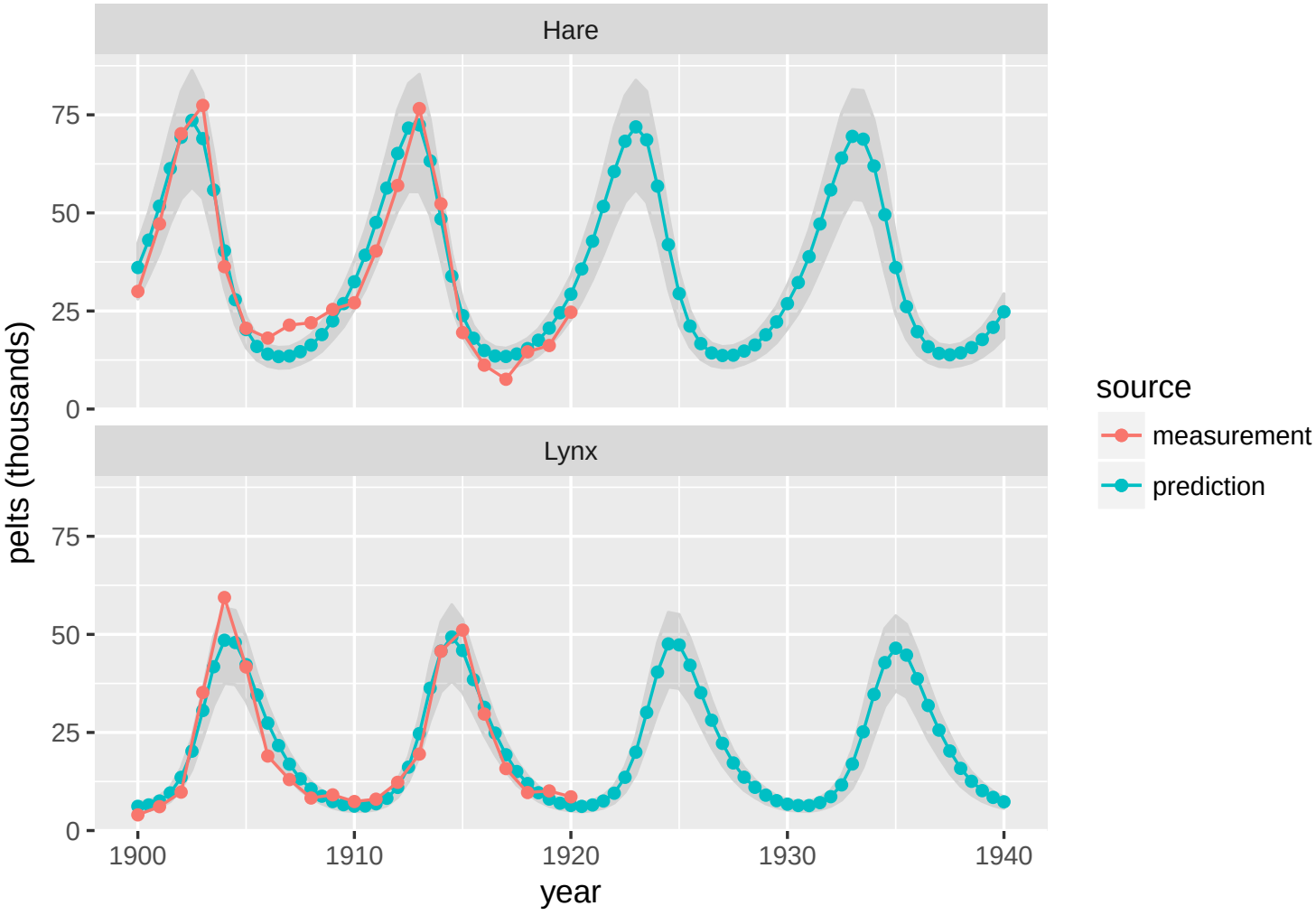
Lotka-Volterra Parameter Estimates

```
> print(fit, c("theta", "sigma"), probs=c(0.1, 0.5, 0.9))

         mean se_mean    sd   10%   50%   90%  n_eff Rhat
theta[1] 0.55       0  0.07  0.46  0.54  0.64   1168    1
theta[2] 0.03       0  0.00  0.02  0.03  0.03   1305    1
theta[3] 0.80       0  0.10  0.68  0.80  0.94   1117    1
theta[4] 0.02       0  0.00  0.02  0.02  0.03   1230    1
sigma[1] 0.29       0  0.05  0.23  0.28  0.36   2673    1
sigma[2] 0.29       0  0.06  0.23  0.29  0.37   2821    1
```

- Rhat near 1 signals convergence; n_eff is effective sample size
- 10%, ... posterior quantiles; e.g., Pr[$\alpha \in (0.46, 0.64) \mid y$] = 0.8
- posterior mean is Bayesian point estimate: $\hat{\alpha} = \theta_1 = 0.55$
- standard error in posterior mean estimate is 0 (with rounding)
- posterior standard deviation of α estimated as 0.07

Lotka-Volterra Posterior Predictions



- training data (1900-1920); future predictions (1921+)

Bayesian Methodology

Probability is Epistemic

- **John Stuart Mill** (*Logic* 1882, Part III, Ch. 2):
  - ... the probability of an event is not a quality of the event itself, but a mere name for the degree of ground which we, or some one else, have for expecting it.
  - Every event is **in itself certain**, not probable; if we knew all, we should either know positively that it will happen, or positively that it will not.
  - ... its probability to us means the **degree of expectation of its occurrence**, which we are warranted in entertaining by our present evidence.
- Probabilities **quantify uncertainty**
- Statistical reasoning is **counterfactual**

Random Variables

- Random variables are the currency of probability theory
- Random variables typically take numbers as values
- Imagine a bin filled with balls representing the way the world might be
- A ball records the value of every random variable
- Examples
  - the sum of the three best among a roll of four dice (d6)
  - time before the next traffic accident on a given highway
  - prevalence of a disease in a population

Events

- Event is a set of outcomes
- Usually subset of random variable values
  - prevalence of disease $\lambda > 0.02$
  - probability that one player is better than another, $\theta_1 > \theta_2$
  - probability that team A wins a game against B
  - probability that team A betas team B by more than 5 points
- Probability is that one of the outcomes occurs

Conditional Probability

- What is probability a man is taller than 6'?
  - What if I tell you he's Dutch?
  - What if I tell you he's a professional athlete?
  - What if I tell you he's a jockey? or basketball player?
  - What if I tell you his mother is taller than 6'?

Bayesian Data Analysis

- "By Bayesian data analysis, we mean practical methods for making inferences from data using probability models for quantities we observe and about which we wish to learn."
- "The essential characteristic of Bayesian methods is their **explicit use of probability for quantifying uncertainty** in inferences based on statistical analysis."

Gelman et al., *Bayesian Data Analysis*, 3rd edition, 2013

Bayesian Methodology

- Set up **full probability model**
  - for **all** observable & unobservable quantities
  - consistent w. problem knowledge & data collection
- **Condition** on observed data (where Stan comes in!)
  - to calculate posterior probability of unobserved quantities (e.g., parameters, predictions, missing data)
- **Evaluate**
  - model fit and implications of posterior
- **Repeat** as necessary

Ibid.

Where do Models Come from?

- Sometimes model comes first, based on substantive considerations
  - toxicology, economics, ecology, physics, . . .
- Sometimes model chosen based on data collection
  - traditional statistics of surveys and experiments
- Other times the data comes first
  - observational studies, meta-analysis, . . .
- Usually its a mix

Model Checking

- Do the inferences make sense?
  - are parameter values consistent with model's prior?
  - does simulating from parameter values produce reasonable fake data?
  - are marginal predictions consistent with the data?
- Do predictions and event probabilities for new data make sense?
- ***Not***: Is the model true?
- ***Not***: What is Pr[model is true]?
- ***Not***: Can we "reject" the model?

Model Improvement

- Expanding the model
  - hierarchical and multilevel structure . . .
  - more flexible distributions (overdispersion, covariance)
  - more structure (geospatial, time series)
  - more modeling of measurement methods and errors
  - . . .
- Including more data
  - breadth (more predictors or kinds of observations)
  - depth (more observations)

Properties of Bayesian Inference

- Explores full range of parameters consistent with prior info and data*
  - * if such agreement is possible
  - Stan automates this procedure with diagnostics
- Inferences can be plugged in directly for
  - parameter estimates minimizing expected error
  - predictions for future outcomes with uncertainty
  - event probability updates conditioned on data
  - risk assessment / decision analysis conditioned on uncertainty

"All Models are Wrong, but some are useful"

"Now it would be very remarkable if any system existing in the real world could be exactly represented by any simple model. However, cunningly chosen parsimonious models often do provide remarkably useful approximations."

— George Box (1979)

- Slide title was section title in Box's paper

Box, G. E. P. 1979. Robustness in the strategy of scientific model building. In *Robustness in Statistics*, Academic Press.

Model (Mis)Specification

- A model is **well specified** if it matches the data-generating process of the data.
- All of our models will be **misspecified** to some extent
  - mildly misspecified: Newtonian physics (most speeds)
  - wildly misspecified: social science regression
  - wildly mispsecified: Newtonian physics (near speed of light)
- Models assumptions and predictions must be **tested**
  - ideally with **cross-validation** on quantities of interest

The Folk Theorem

"When you have computational problems, often there's a problem with your model."

— Andrew Gelman, blog

- The usual culprits are
  - bugs in: samplers, data munging, model coding, etc.
  - model misspecification

http://andrewgelman.com/2008/05/13/the_folk_theore/

Model Calibration

- Consider 100 days for which a meteorologist predicted a 70% chance of rain
  - about 70 of them should have had rain
  - not fewer, not more!
  - technically, expect Binomial$(100, 0.7)$ rainy day from a calibrated model
- Use posterior predictive checks to test calibration on
  - training data—can it fit?
  - held out data—can it predict?
  - cross-validation—approximates held out with trainin data
- Also applies to interval coverage of parameter values

Model Sharpness

- Ideal forecasts are deterministic
  - predict 100% chance of rain or 0% chance of rain
  - always right
- A forecast of 90% chance of rain reduces uncertainty more than a 50% prediction
- A model is **sharp** if it has narrow posterior intervals
  - Prediction Pr$[\alpha \in (1.2, 1.9)] = 0.9$
  - is sharper than Pr$[\alpha \in (1, 2)] = 0.9$
- I.e., sharper models are more certain in its predictions
- Given calibration, we want our predictions to be **sharp**

Cross-Validation

- Uses single data set to model held-out performance
- Assumes stationarity (as most models do)
- Partition data evenly into disjoint subsets (called **folds**)
  - 10 is a common choice
  - **leave-one-out** (LOO) uses a the number of training data points
- For each fold
  - estimate model on all data but that fold
  - test on that fold
- Usual comparison statistic is held out log likelihood

Bayesian Inference

Notation for Basic Quantities

- **Basic Quantities**
  - y: observed data
  - θ: parameters (and other unobserved quantities)
  - x: constants, predictors for conditional (aka "discriminative") models
- **Basic Predictive Quantities**
  - $\tilde{y}$: unknown, potentially observable quantities
  - $\tilde{x}$: constants, predictors for unknown quantities

Naming Conventions

- **Joint**: $p(y, \theta)$
- **Sampling / Likelihood**: $p(y|\theta)$
  - Sampling is function of y with θ fixed (prob function)
  - Likelihood is function of θ with y fixed (*not* prob function)
- **Prior**: $p(\theta)$
- **Posterior**: $p(\theta|y)$
- **Data Marginal (Evidence)**: $p(y)$
- **Posterior Predictive**: $p(\tilde{y}|y)$

Bayes's Rule for Posterior

$$
\begin{aligned}
p(\theta|y) &= \frac{p(y,\theta)}{p(y)} && \text{[def of conditional]} \\
&= \frac{p(y|\theta)\,p(\theta)}{p(y)} && \text{[chain rule]} \\
&= \frac{p(y|\theta)\,p(\theta)}{\int_\Theta p(y,\theta')\;d\theta'} && \text{[law of total prob]} \\
&= \frac{p(y|\theta)\,p(\theta)}{\int_\Theta p(y|\theta')\,p(\theta')\;d\theta'} && \text{[chain rule]}
\end{aligned}
$$

- *Inversion:* Final result depends only on sampling distribution (likelihood) $p(y|\theta)$ and prior $p(\theta)$

Bayes's Rule up to Proportion

- If data y is fixed, then

$$\begin{aligned} p(\theta|y) &= \frac{p(y|\theta)\,p(\theta)}{p(y)} \\ &\propto p(y|\theta)\,p(\theta) \\ &= p(y,\theta) \end{aligned}$$

- Posterior proportional to likelihood times prior
- Equivalently, posterior proportional to joint
- The nasty integral for data marginal $p(y)$ goes away

What Stan Computes

Draws from Posterior

- Stan performs (Markov chain) Monte Carlo sampling
- Produces sequence of draws

$$\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(M)}$$

- wher each draw $\theta^{(m)}$ is marginally distributed according to the posterior $p(\theta|y)$
- Draws characterize posterior
- Plot with histograms, kernel density estimates, etc.

Parameter Estimates

- An estimator maps data into a parameter estimate
- The standard Bayesian estimator is the posterior mean

$$\begin{aligned}\hat{\theta} &= \mathbb{E}[\theta \mid y] \\ &= \int_{\Theta} \theta \; p(\theta|y) \; \mathrm{d}\theta \\ &\approx \frac{1}{M} \sum_{m=1}^{M} \theta^{(m)}\end{aligned}$$

- Posterior mean minimizes expected square error

$$\hat{\theta} = \text{arg min}_{\theta} \mathbb{E}[(\theta - \hat{\theta})^2]$$

Parameter Variance Estimates

- Variances is expected squared error
- Calculated by plugging in estimated value $\hat{\theta}$

$$
\begin{aligned}
\text{var}[\theta \mid y] &= \mathbb{E}[(\theta - \mathbb{E}[\theta \mid y])^2 \mid y] \\
&= \int (\theta - \hat{\theta})^2 \, p(\theta|y) \, \mathrm{d}\theta \\
&\approx \frac{1}{M-1} \sum_{m=1}^{M} (\theta^{(m)} - \hat{\theta})^2
\end{aligned}
$$

Posterior Quantiles

- Let Y be a random variable
- If $\Pr[Y \leq q] = \alpha$, then q is the α-quantile of Y
- Median is 0.5 quantile, i.e., q such that $\Pr[Y \leq q] = 0.5$
- Central 90
- To compute quantile α for θ in the posterior $p(\theta|y)$:

$$\text{quantile}(\theta, \alpha) \;=\; \text{sort\_ascending}(\theta^{(1)}, \ldots, \theta^{(M)})\,[\,\lceil \alpha \times M \rceil\,]$$

Event Probabilities

- Events are fundamental probability bearing units which
  - are defined by sets of outcomes
  - which occur or not with some probability
- Outcomes defined by values of random variables
- Use conditions on parameters to define sets of outcomes
  - e.g., $\theta_b > \theta_g$ for more boy births than girl births
  - e.g., $z_k = 1$ for team A beating team B in game k
- A set S corresponds to an indicator function f,

$$f(s) = \begin{cases} 1 & \text{if } s \in S \\ 0 & \text{if } s \notin S \end{cases}$$

Calculating Event Probabilities

- Event probabilities are expectations of indicator functions
- Write the indicator function for event E be $\text{cond}(\theta)$

$$
\begin{aligned}
\Pr[E \mid y] &= \mathbb{E}[\text{cond}(\theta) \mid y] \\
&= \int_{\Theta} \text{I}[\text{cond}(\theta) \mid y] \, \text{d}\theta \\
&\approx \frac{1}{M} \sum_{m=1}^{M} \text{I}[\text{cond}(\theta^{(m)})]
\end{aligned}
$$

- Just proportion of posterior draws for which condition holds

Posterior Event Probabilities

- An **event** A is a collection of outcomes
- So A may be defined by an indicator f on parameters

$$f(\theta) = \begin{cases} 1 & \text{if } \theta \in A \\ 0 & \text{if } \theta \notin A \end{cases}$$

  - $f(\theta) = \mathrm{I}(\theta_1 > \theta_2)$ for $\Pr[\theta_1 > \theta_2 \mid y]$,
  - $f(\theta) = \mathrm{I}(\theta \in (0.50, 0.52)$ for $\Pr[\theta \in (0.50, 0.52) \mid y]$
- Defined by posterior expectation of indicator $f(\theta)$

$$\Pr[A \mid y] \;=\; \mathbb{E}[f(\theta) \mid y] \;=\; \int_{\Theta} f(\theta)\, p(\theta|y)\, d\theta.$$

Posterior Predictive Distribution

- Predict new data $\tilde{y}$ based on observed data y
- Marginalize parameters θ out of posterior and likelihood

$$
\begin{aligned}
p(\tilde{y} \mid y) &= \mathbb{E}[p(\tilde{y}|\theta) \mid y] \\
&= \int p(\tilde{y}|\theta)\, p(\theta|y)\, \mathrm{d}\theta. \\
&\approx \frac{1}{M} \sum_{m=1}^{M} p(\tilde{y}|\theta^{(m)})
\end{aligned}
$$

- Weights predictions $p(\tilde{y}|\theta)$ by posterior $p(\theta|y)$

Calculations in Stan

- Stan automatically computes estimates, variances, and quantiles for parameters
- To compute event probabilities, define indicator in generated quantities block

```
generated quantities {
  int<lower=0, upper=1> theta_gt_half = (theta > 0.5);
}
```

- To compute predictions, define with random number generators in the generated quantities block

```
generated quantities {
  real y_predict = normal_rng(x_predict * alpha, sigma);
}
```

Repeated Binary Trials

Repeated Binary Trial Model

- **Data**
  - $N \in \{0, 1, \ldots\}$: number of trials (constant)
  - $y_n \in \{0, 1\}$: trial n success (known, modeled data)
- **Parameter**
  - $\theta \in [0, 1]$: chance of success (unknown)
- **Prior**
  - $p(\theta) \;=\; \text{Uniform}(\theta \mid 0, 1) = 1$
- **Likelihood**
  - $p(y \mid \theta) \;=\; \prod_{n=1}^{N} \text{Bernoulli}(y_n \mid \theta) \;=\; \prod_{n=1}^{N} \theta^{y_n} \, (1 - \theta)^{1 - y_n}$
- **Posterior**
  - $p(\theta \mid y) \propto p(\theta) \, p(y \mid \theta)$

Stan Program

```
data {
  int<lower=0> N;                   // number of trials
  int<lower=0, upper=1> y[N];       // success on trial n
}
parameters {
  real<lower=0, upper=1> theta;     // chance of success
}
model {
  theta ~ uniform(0, 1);            // prior
  y ~ bernoulli(theta);             // likelihood
}
```

A Stan Program...

- defines log (posterior) density up to constant, so...
- equivalent to define log density directly:

```
model {
  target += 0;
  for (n in 1:N)
    target += log(y[n] ? theta : (1 - theta));
}
```

- equivalent to drop constant prior and vectorize likelihood:

```
model {
  y ~ bernoulli(theta);
}
```

R: Simulate Data

- Generate data

```
> theta <- 0.30;
> N <- 20;
> y <- rbinom(N, 1, 0.3);

> y

 [1] 1 1 1 1 0 0 0 0 1 1 0 0 1 0 0 0 0 0 0 1
```

- Calculate MLE as sample mean from data

```
> sum(y) / N

[1] 0.4
```

RStan: Bayesian Posterior

```
> library(rstan);

> fit <- stan("bern.stan",
              data = list(y = y, N = N));

> print(fit, probs=c(0.1, 0.9));

Inference for Stan model: bern.
4 chains, each with iter=2000; warmup=1000; thin=1;
post-warmup draws per chain=1000,
total post-warmup draws=4000.

        mean se_mean   sd    10%    90% n_eff Rhat
theta   0.41    0.00 0.10   0.28   0.55  1580    1
```

Plug in Posterior Draws

- Extracting the posterior draws

```
> theta_draws <- extract(fit)$theta;
```

- Calculating posterior mean (estimator)

```
> mean(theta_draws);

[1] 0.4128373
```

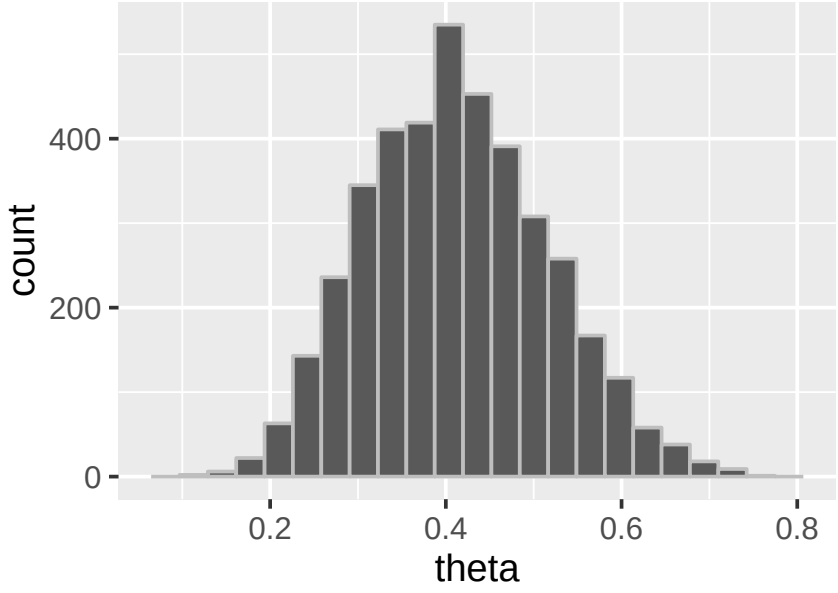
- Calculating posterior intervals

```
> quantile(theta_draws, probs=c(0.10, 0.90));

      10%       90%
0.2830349 0.5496858
```

Marginal Posterior Histograms

```
theta_draws_df <- data.frame(list(theta = theta_draws));
plot <-
  ggplot(theta_draws_df, aes(x = theta)) +
  geom_histogram(bins=20, color = "gray");
plot;
```



- Displays the full posterior *marginal* distribution $p(\theta \mid y)$

RStan: MAP, penalized MLE

- Stan's optimization for estimation; two views:
  - max posterior mode, aka max a posteriori (MAP)
  - max penalized likelihood (MLE)

```
> library(rstan);
> N <- 5;
> y <- c(0,1,1,0,0);
> model <- stan_model("bernoulli.stan");
> mle <- optimizing(model, data=c("N", "y"));
...
> print(mle, digits=2)
$par                $value  (log density)
theta               [1] -3.4
  0.4
```

Male Birth Ratio

Birth Rate by Sex

- **Laplace**'s data on live births in Paris from 1745–1770:

sex	*live births*
female	241 945
male	251 527

- **Question 1** (Estimation)
  What is the birth rate of boys vs. girls?
- **Question 2** (Event Probability)
  Is a boy more likely to be born than a girl?
- Bayes (1763) set up the "Bayesian" model
- Laplace (1781, 1786) solved for the posterior

Binomial Distribution

- Binomial distribution is number of successes y in N i.i.d. Bernoulli trials with chance of success θ
- If $y_1, \ldots, y_N \sim \text{Bernoulli}(\theta)$,

  then $(y_1 + \cdots + y_N) \sim \text{Binomial}(N, \theta)$
- The analytic form is

$$\text{Binomial}(y|N,\theta) \;=\; \binom{N}{y}\theta^{y}(1-\theta)^{N-y}$$

  where the binomial coefficient normalizes for permutations (i.e., which subset of n has $y_n = 1$),

$$\binom{N}{y} = \frac{N!}{y!\,(N-y)!}$$

Binomial Distribution

- Don't know order of births, only total.
- If $y_1, \ldots, y_N \sim \text{Bernoulli}(\theta)$,

  then $(y_1 + \cdots + y_N) \sim \text{Binomial}(N, \theta)$
- The analytic form is

$$\text{Binomial}(y|N,\theta) \;=\; \binom{N}{y}\theta^{y}(1-\theta)^{N-y}$$

  where the binomial coefficient normalizes for permutations (i.e., which subset of n has $y_n = 1$),

$$\binom{N}{y} = \frac{N!}{y!\,(N-y)!}$$

Mathematics vs. Simulation

- Luckily, we don't have to be as good at math as Laplace
- Nowadays, we calculate all these integrals by computer using tools like Stan

> *If you wanted to do foundational research in statistics in the mid-twentieth century, you had to be bit of a mathematician, whether you wanted to or not. . . . if you want to do statistical research at the turn of the twenty-first century, you have to be a computer programmer.*
>
> *—from Andrew's blog*

Bayes's Binomial Model

- Data
  - y: total number of male live births (251,527)
  - N : total number of live births (493,472)
- Parameter
  - $\theta \in (0, 1)$: proportion of male live births
- Likelihood

$$p(y|N,\theta) \;=\; \text{Binomial}(y|N,\theta) \;=\; \binom{N}{y}\theta^{y}(1-\theta)^{N-y}$$

- Prior

$$p(\theta) \;=\; \text{Uniform}(\theta \mid 0,1) \;=\; 1$$

Beta Distribution

- Required for analytic posterior of Bayes's model
- For parameters $\alpha, \beta > 0$ and $\theta \in (0, 1)$,

$$\text{Beta}(\theta|\alpha,\beta) \;=\; \frac{1}{\text{B}(\alpha,\beta)}\theta^{\alpha-1}\,(1-\theta)^{\beta-1}$$

- Euler's Beta function is used to normalize,

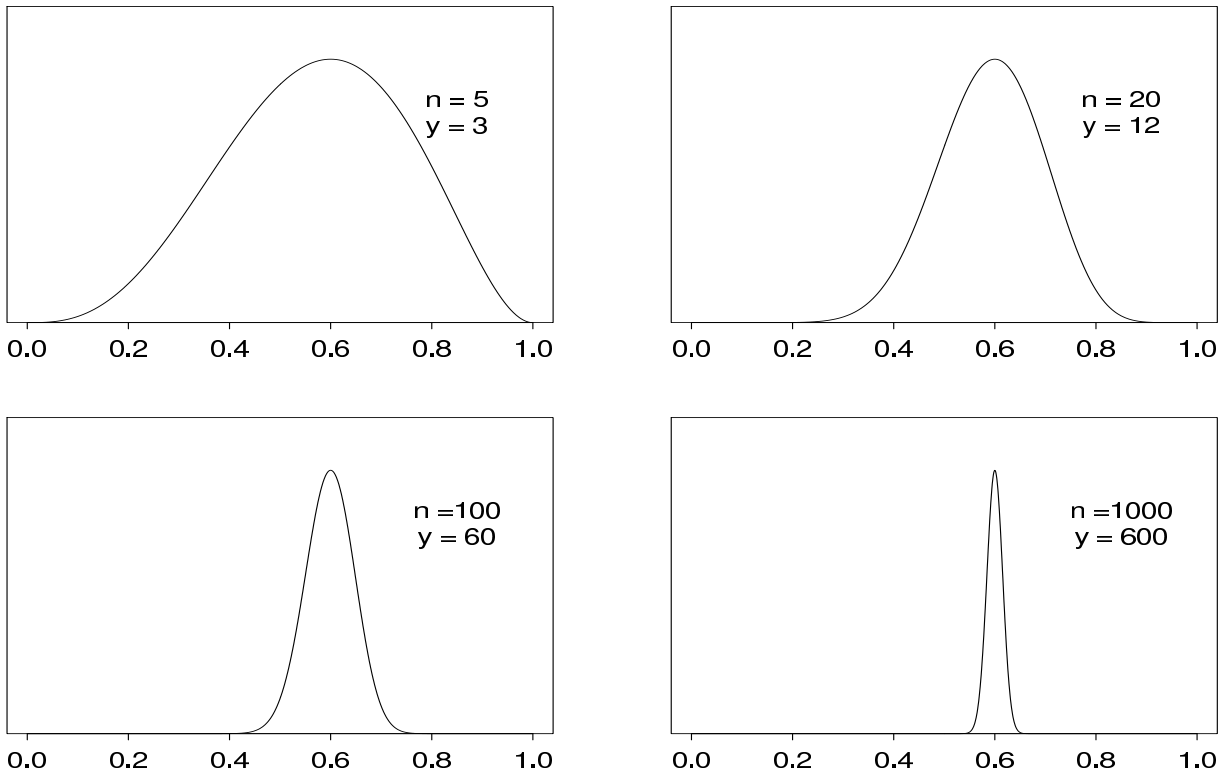
$$\text{B}(\alpha,\beta) \;=\; \int_0^1 u^{\alpha-1}(1-u)^{\beta-1}du \;=\; \frac{\Gamma(\alpha)\,\Gamma(\beta)}{\Gamma(\alpha+\beta)}$$

  where $\Gamma()$ is continuous generalization of factorial

- Note: $\text{Beta}(\theta|1,1) = \text{Uniform}(\theta|0,1)$

Beta Distribution — Examples

- Unnormalized posterior density assuming uniform prior and y successes out of n trials (all with mean 0.6).



Gelman et al. (2013) *Bayesian Data Analysis*, 3rd Edition.

Laplace Turns the Crank

- Given Bayes's general formula for the posterior

$$p(\theta|y,N) \;=\; \frac{\text{Binomial}(y|N,\theta)\,\text{Uniform}(\theta|0,1)}{\int_\Theta \text{Binomial}(y|N,\theta')\,p(\theta')d\theta'}$$

- Laplace used Euler's Beta function (B) to normalize the posterior, with final solution

$$p(\theta|y,N) \;=\; \text{Beta}(\theta \mid y+1,\; N-y+1)$$

Laplace turns the Crank

- What is probability that a male live birth is more probable?

$$\begin{aligned}
\Pr[\theta > 0.5] &= \int_{\Theta} \mathrm{I}[\theta > 0.5]\, p(\theta|y,N)d\theta \\
&= \int_{0.5}^{1} p(\theta|y,N)d\theta \\
&\approx 1 - 10^{-42}
\end{aligned}$$

- Laplace solved Bayes's integral by
  - determining that the posterior was a beta distribution (conjugacy!)
  - and solving the normalization (gamma functions)

Calculating Laplace's Answers

```
transformed data {
  int male = 251527;
  int female = 241945;
}
parameters {
  real<lower=0, upper=1> theta;
}
model {
  male ~ binomial(male + female, theta);
}
generated quantities {
  int<lower=0, upper=1> theta_gt_half = (theta > 0.5);
}
```

And the Answer is...

```
> fit <- stan("laplace.stan", iter=100000);
> print(fit, probs=c(0.005, 0.995), digits=3)
```

	mean	*0.5%*	*99.5%*
theta	*0.51*	*0.508*	*0.512*
theta_gt_half	*1.00*	*1.000*	*1.000*

- Q1: θ is 99% certain to lie in $(0.508, 0.512)$
- Q2: Laplace "morally certain" boys more prevalent

Estimation

- Posterior is $\text{Beta}(\theta \mid 1 + 241\,945,\ 1 + 251\,527)$
- Posterior mean:

$$\frac{1 + 251\,527}{1 + 241\,945 + 1 + 251\,527} \approx 0.50970873$$

- Maximum likelihood estimate same as posterior mode (because of uniform prior)

$$\frac{251\,527}{241\,945 + 251\,527} \approx 0.50970882$$

- As number of observations approaches ∞,
  MLE approaches posterior mean

Event Probability Inference

- What is probability that a male live birth is more likely than a female live birth?

$$
\begin{aligned}
\Pr[\theta > 0.5] &= \int_{\Theta} \mathsf{I}[\theta > 0.5]\, p(\theta|y,N)d\theta \\
&= \int_{0.5}^{1} p(\theta|y,N)d\theta \\
&= 1 - F_{\theta|y,N}(0.5) \\
&\approx 10^{-42}
\end{aligned}
$$

- $\mathsf{I}[\phi] = 1$ if condition ϕ is true and 0 otherwise.
- $F_{\theta|y,N}$ is posterior cumulative distribution function (cdf).

Fisher "Exact" Test

Bayesian "Fisher Exact Test"

- Suppose we observe the following data on handedness

	sinister	*dexter*	TOTAL
male	9 (y_1)	43	52 (N_1)
female	4 (y_2)	44	48 (N_2)

- Assume likelihoods Binomial$(y_k|N_k, \theta_k)$, uniform priors
- Are men more likely to be lefthanded?

$$\begin{aligned} \Pr[\theta_1 > \theta_2 \,|\, y, N] &= \int_\Theta \mathsf{I}[\theta_1 > \theta_2]\, p(\theta|y, N)\, d\theta \\ &\approx \frac{1}{M} \sum_{m=1}^{M} \mathsf{I}[\theta_1^{(m)} > \theta_2^{(m)}]. \end{aligned}$$

Stan Binomial Comparison

```
data {
  int y[2];
  int N[2];
}
parameters {
  vector<lower=0,upper=1> theta[2];
}
model {
  y ~ binomial(N, theta);
}
generated quantities {
  real boys_minus_girls = theta[1] - theta[2];
  int boys_gt_girls = theta[1] > theta[2];
}
```

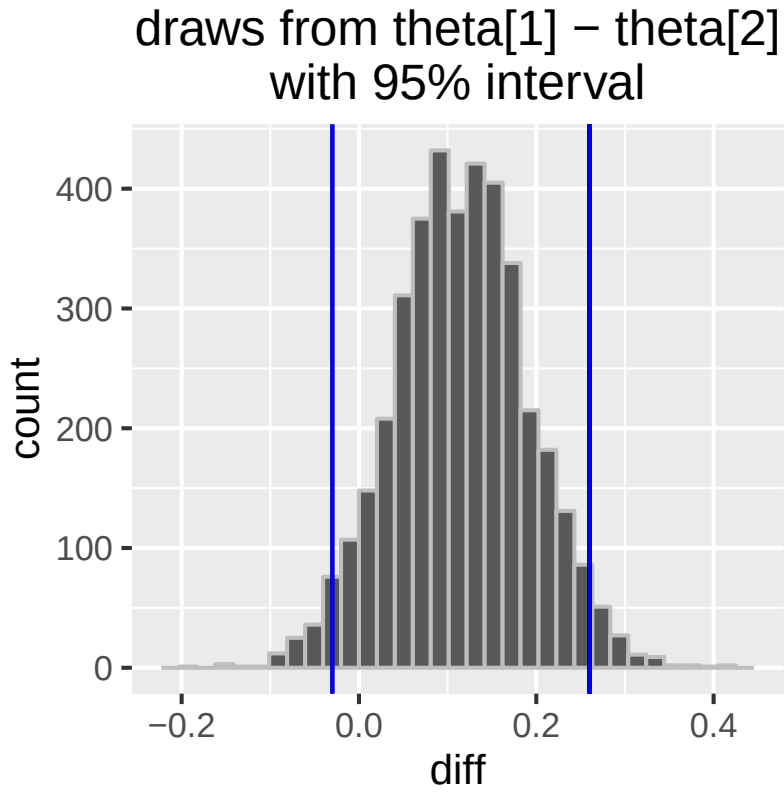
Binomial Comparison Results

	mean	2.5%	97.5%
theta[1]	0.22	0.12	0.35
theta[2]	0.11	0.04	0.21
boys_minus_girls	0.12	-0.03	0.26
boys_gt_girls	0.93	0.00	1.00

- $\Pr[\theta_1 > \theta_2 \mid y] \approx 0.93$
- $\Pr\left[(\theta_1 - \theta_2) \in (-0.03, 0.26) \mid y\right] = 95\%$

Visualizing Posterior Difference

- Plot of posterior difference, $p(\theta_1 - \theta_2 \mid y, N)$ (men - women)



- Vertical bars: central 95% posterior interval $(-0.03, 0.26)$

More Stan Models

Posterior Predictive Distribution

- Predict new data ($\tilde{y}$) given observed data (y)
- Includes two kinds of uncertainty
  - parameter estimation uncertainty: $p(\theta|y)$
  - sampling uncertainty: $p(\tilde{y}|\theta)$

$$p(\tilde{y}|y) = \int p(\tilde{y}|\theta)\ p(\theta|y)\ \mathrm{d}\theta$$

$$\approx \frac{1}{M} \sum_{m=1}^{M} p(\tilde{y}|\theta^{(m)})$$

- Can generate predictions as sample of draws $\tilde{y}^{(m)}$ based on $\theta^{(m)}$

Posterior Predictive Inference

- Parameters θ, observed data y, and data to predict $\tilde{y}$

$$p(\tilde{y}|y) = \int_{\Theta} p(\tilde{y}|\theta)\ p(\theta|y)\ d\theta$$

- ```
  data {
    int<lower=0> N_tilde;
    matrix[N_tilde,K] x_tilde;
    ...
  parameters {
    vector[N_tilde] y_tilde;
    ...
  model {
    y_tilde ~ normal(x_tilde * beta, sigma);
  ```
  ```

Predict w. Generated Quantities

- Replace sampling with pseudo-random number generation

```
generated quantities {
  vector[N_tilde] y_tilde;

  for (n in 1:N_tilde)
    y_tilde[n] = normal_rng(x_tilde[n] * beta, sigma);
}
```

- Must include noise for predictive uncertainty
- PRNGs only allowed in generated quantities block
  - more computationally efficient per iteration
  - more statistically efficient with i.i.d. samples (i.e., MC, not MCMC)

Linear Regression with Prediction

```
data {
  int<lower=0> N;                int<lower=0> K;
  matrix[N, K] x;                vector[N] y;
  matrix[N_tilde, K] x_tilde;
}
parameters {
  vector[K] beta;                real<lower=0> sigma;
}
model {
  y ~ normal(x * beta, sigma);
}
generated quantities {
  vector[N_tilde] y_tilde
    = normal_rng(x_tilde * beta, sigma);
}
```

Transforming Precision to Scale

```
parameters {
  real<lower=0> tau;
  ...
}
transformed parameters {
  real<lower=0> sigma = tau^(-0.5);
}
```

Transform: Non-Centered Params

```
parameters {
  vector[K] beta_std;  // non-centered
}
transformed parameters {
  vector[K] beta =  mu + sigma * beta_std;
}
model {
  // implies:  beta ~ normal(mu, sigma)
  beta_std ~ normal(0, 1);
}
```

Logistic Regression

```
data {
  int<lower=1> K;
  int<lower=0> N;
  matrix[N,K] x;
  int<lower=0,upper=1> y[N];
}
parameters {
  vector[K] beta;
}
model {
   beta ~ cauchy(0, 2.5);           // prior
   y ~ bernoulli_logit(x * beta);   // likelihood
}
```

Generalized Linear Models

- Direct parameterizations more efficient and stable
- **Logistic regression** (boolean/binary data)
  - `y ~ bernoulli(inv_logit(eta));`
  - `y ~ bernoulli_logit(eta);`
  - Probit via `Phi` (normal cdf)
  - Robit (robust) via Student-t cdf
- **Poisson regression** (count data)
  - `y ~ poisson(exp(eta));`
  - `y ~ poisson_log(eta);`
  - Overdispersion with negative binomial

GLMS, continued

- **Multi-logit regression** (categorical data)
  - `y ~ categorical(softmax(eta));`
  - `y ~ categorical_logit(eta);`
- **Ordinal logistic regression** (ordered data)
  - Add cutpoints `c`
  - `y ~ ordered_logistic(eta, c);`
- **Robust linear regression** (overdispersed noise)
  - `y ~ student_t(nu, eta, sigma);`

Time Series Autoregressive: AR(1)

```
data {
  int<lower=0> N;   vector[N] y;
}
parameters {
  real alpha;  real beta;  real sigma;
}
model {
  y[2:n] ~ normal(alpha + beta * y[1:(n-1)], sigma);
}
```

LKJ Density and Cholesky Factors

- Density on *correlation* matrices Ω
- $\text{LKJCorr}(\Omega \mid \nu) \propto \det(\Omega)^{(\nu-1)}$
  - $\nu = 1$ uniform
  - $\nu > 1$ concentrates around unit matrix
- Work with Cholesky factor L_Ω s.t. $\Omega = L_\Omega \, L_\Omega^\top$
  - Density: $\text{LKJCorrCholesky}(L_\Omega \mid \nu) \propto |J| \det(L_\Omega \, L_\Omega^\top)^{(\nu-1)}$
  - Jacobian adjustment for Cholesky factorization

Lewandowski, Kurowicka, and Joe (2009)

Covariance Random-Effects Priors

```
parameters {
  vector[2] beta[G];
  cholesky_factor_corr[2] L_Omega;
  vector<lower=0>[2] sigma;

model {
  sigma ~ cauchy(0, 2.5);
  L_Omega ~ lkj_cholesky(4);
  beta ~ multi_normal_cholesky(rep_vector(0, 2),
                               diag_pre_multiply(sigma, L_Omega));
  for (n in 1:N)
    y[n] ~ bernoulli_logit(... + x[n] * beta[gg[n]]);
```

- G groups with varying slope and intercept; gg indicates group

Example: Gaussian Process Estimation

```
data {
  int<lower=1> N;  vector[N] x; vector[N] y;
} parameters {
  real<lower=0> eta_sq, inv_rho_sq, sigma_sq;
} transformed parameters {
  real<lower=0> rho_sq; rho_sq = inv(inv_rho_sq);
} model {
  matrix[N,N] Sigma;
  for (i in 1:(N-1)) {
    for (j in (i+1):N) {
      Sigma[i,j] = eta_sq * exp(-rho_sq * square(x[i] - x[j]));
      Sigma[j,i] = Sigma[i,j];
  }}
  for (k in 1:N) Sigma[k,k] = eta_sq + sigma_sq;
  eta_sq, inv_rho_sq, sigma_sq ~ cauchy(0,5);
  y ~ multi_normal(rep_vector(0,N), Sigma);
}
```

Gaussian Process Predictions

- Add predictors `x_tilde[M]` for points to predict
- Declare predicted values `y_tilde[M]` as unconstrained parameters
- Define `Sigma[M+N,M+N]` in terms of full `append_row(x, x_tilde)`
- Model remains the same

```
append_row(y,y_tilde)
  ~ multi_normal(rep(0,N+M),Sigma);
```

Mixture of Two Normals

```
for (n in 1:N) {
  real lp1;  real lp2;

  lp1 = bernoulli_log(0, lambda)
          + normal_log(y[n], mu[1], sigma[1]);

  lp2 = bernoulli_log(1, lambda)
          + normal_log(y[n], mu[2], sigma[2]);

  target += log_sum_exp(lp1,lp2);
```

- local variables reassigned; direct increment of log posterior
- $\text{log\_sum\_exp}(\alpha, \beta) = \log(\exp(\alpha) + \exp(\beta))$
- **Much more efficient** than sampling (Rao-Blackwell Theorem)

Other Mixture Applications

- Other multimodal data
- Zero-inflated Poisson or hurdle models
- Model comparison or mixture
- Discrete change-point model
- Hidden Markov model, Kalman filter
- Almost anything with latent discrete parameters
- Other than variable choice, e.g., regression predictors
  - marginalization is exponential in number of vars

Dynamic Systems with Diff Eqs

- Simple harmonic oscillator

$$\frac{d}{dt} y_1 = -y_2 \qquad \frac{d}{dt} y_2 = -y_1 - \theta y_2$$

- Code as a function in Stan

```
functions {
  real[] sho(data real t, real[] y, real[] theta,
             data real[] x_r, data int[] x_i) {
    return { y[2],
             -y[1] - theta[1] * y[2] };
  }
}
```

Fit Noisy State Measurements

```
data {
  int<lower=1> T;       real y[T,2];
  real t0;              real ts[T];
}
parameters {
  real y0[2];                   // unknown initial state
  real theta[1];                // rates for equation
  vector<lower=0>[2] sigma;  // measurement error
}
model {
  real y_hat[T,2];
  ...priors...
  y_hat = integrate_ode(sho, y0, t0, ts, theta, x_r, x_i);
  y ~ normal(y_hat, sigma);
}
```