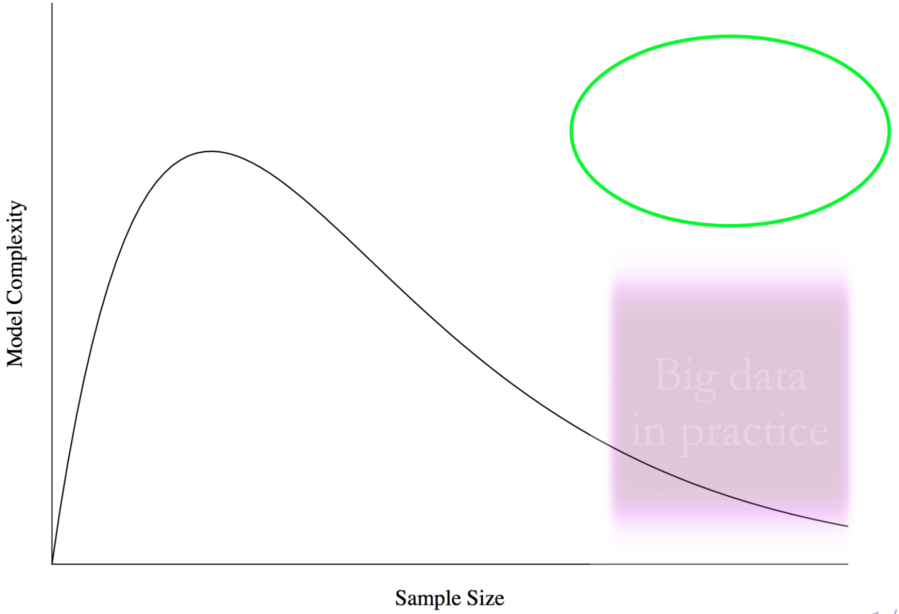
# Big Data need Big Model



# Stan: A platform for Bayesian inference

Andrew Gelman, Bob Carpenter, Matt Hoffman, Daniel Lee,
Ben Goodrich, Michael Betancourt, Marcus Brubaker,
Jiqiang Guo, Peter Li, Allen Riddell,...

Department of Statistics, Columbia University, New York
(and other places)

10 Nov 2014

ogle stan

Web Images Videos Maps News More Search tools

About 91,800,000 results (0.40 seconds)


[Eminem - **Stan** (Short Version) ft. Dido - YouTube](www.youtube.com/watch?v=aSLZFdqwh7E)

www.youtube.com/watch?v=aSLZFdqwh7E

**Artists:** Eminem, Dido

**Album:** No Angel


Eminem - **Stan** (Short Version) ft. Dido - YouTube
www.youtube.com/watch?v=aSLZFdqwh7E

**Artists:** Eminem, Dido
**Album:** No Angel
**Released:** 1999

Feedback

**Stan** (song) - Wikipedia, the free encyclopedia
en.wikipedia.org/wiki/**Stan**_(song)
"**Stan**" is the third single from the The Marshall Mathers LP, recorded in 1999 by American rapper Eminem and featuring British singer Dido. It peaked at number ...
Thank You - Rock City - Robert Browning - Murder ballad

**Stan**: Project Home Page
mc-**stan**.org/
**Stan** modeling language and C++ library for Bayesian inference. NUTS adaptive HMC (MCMC) sampling, automatic differentiation, R, shell interfaces. Gelman.

Urban Dictionary: **stan**
www.urbandictionary.com/define.php?term=**stan**
Based on the central character in the Eminem song of the same name, a "**stan**" is an overzealous maniacal fan for any celebrity or athlete.

# Stan

Stan is a probabilistic programming language implementing full Bayesian statistical inference with

- MCMC sampling (NUTS, HMC)

and penalized maximum likelihood estimation with

- Optimization (BFGS)

Stan is coded in C++ and runs on all major platforms (Linux, Mac, Windows).

Stan is freedom-respecting, open-source software (new BSD core, GPLv3 interfaces).

## Interfaces

Download and getting started instructions, organized by interface:

- RStan v2.5.0 (R)
- PyStan v2.5.0 (Python)
- CmdStan v2.5.0 (shell, command-line terminal)
- MatlabStan (MATLAB)
- Stan.jl (Julia)

Home
RStan
PyStan
CmdStan
MatlabStan
Stan.jl
Manual
Examples
Groups
Issues
Contribute
Source

# Ordered probit

```
data {
  int<lower=2> K;
  int<lower=0> N;
  int<lower=1> D;
  int<lower=1,upper=K> y[N];
  row_vector[D] x[N]; }
parameters {
  vector[D] beta;
  ordered[K-1] c; }
model {
  vector[K] theta;
  for (n in 1:N) {
    real eta;
    eta <- x[n] * beta;
    theta[1] <- 1 - Phi(eta - c[1]);
    for (k in 2:(K-1))
      theta[k] <- Phi(eta - c[k-1]) - Phi(eta - c[k]);
    theta[K] <- Phi(eta - c[K-1]);
    y[n] ~ categorical(theta);
  } }
```

# Measurement error model

```
data {
  ...
  real x_meas[N];      // measurement of x
  real<lower=0> tau;   // measurement noise
}
parameters {
  real x[N];           // unknown true value
  real mu_x;           // prior location
  real sigma_x;        // prior scale
  ...
}
model {
  x ~ normal(mu_x, sigma_x);  // prior
  x_meas ~ normal(x, tau);    // measurement model
  y ~ normal(alpha + beta * x, sigma);
  ...
}
```

# Stan overview

# Stan overview

- Fit open-ended Bayesian models

# Stan overview

- Fit open-ended Bayesian models
- Specify log posterior density in C++

# Stan overview

- Fit open-ended Bayesian models
- Specify log posterior density in C++
- Code a distribution once, then use it everywhere

# Stan overview

- Fit open-ended Bayesian models
- Specify log posterior density in C++
- Code a distribution once, then use it everywhere
- Hamiltonian No-U-Turn sampler

# Stan overview

- Fit open-ended Bayesian models
- Specify log posterior density in C++
- Code a distribution once, then use it everywhere
- Hamiltonian No-U-Turn sampler
- Autodiff

# Stan overview

- Fit open-ended Bayesian models
- Specify log posterior density in C++
- Code a distribution once, then use it everywhere
- Hamiltonian No-U-Turn sampler
- Autodiff
- Runs from R, Python, Matlab, Julia; postprocessing

# People

# People

- Stan core (15)

# People

- Stan core (15)
- Research collaborators (30)
- Developers (100)

# People

- Stan core (15)
- Research collaborators (30)
- Developers (100)
- User community (1000)

# People

- Stan core (15)
- Research collaborators (30)
- Developers (100)
- User community (1000)
- Users (10000)

# Funding

- National Science Foundation
- Institute for Education Sciences
- Department of Energy
- Novartis
- YouGov

# Roles of Stan

# Roles of Stan

- Bayesian inference for unsophisticated users (alternative to Stata, Bugs, etc.)

# Roles of Stan

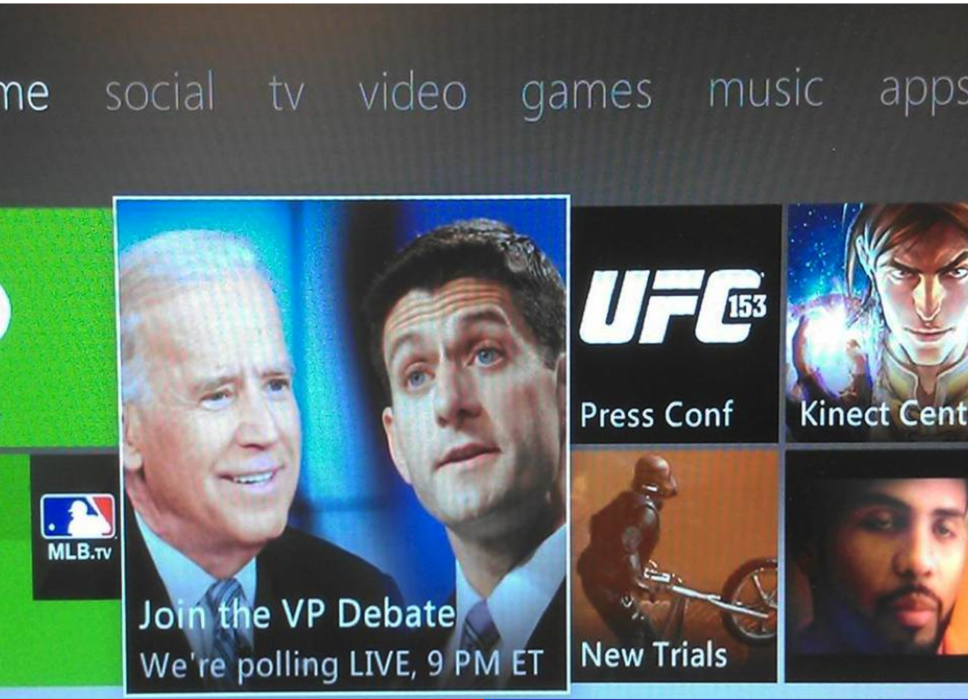
- Bayesian inference for unsophisticated users (alternative to Stata, Bugs, etc.)
- Bayesian inference for sophisticated users (alternative to programming it yourself)

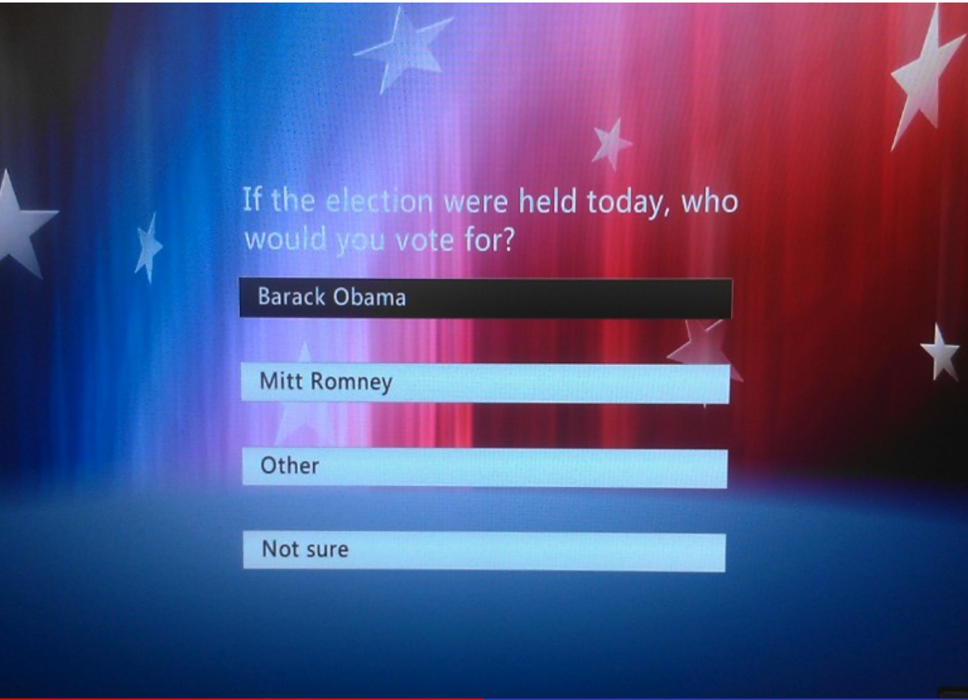
# Roles of Stan

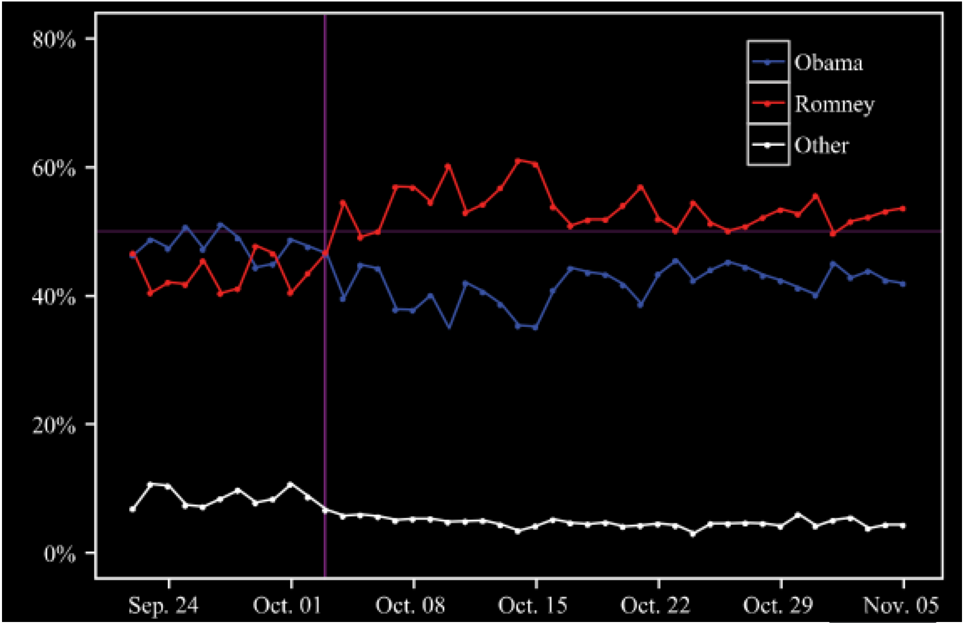
- Bayesian inference for unsophisticated users (alternative to Stata, Bugs, etc.)
- Bayesian inference for sophisticated users (alternative to programming it yourself)
- Fast and scalable gradient computation

# Roles of Stan

- Bayesian inference for unsophisticated users (alternative to Stata, Bugs, etc.)
- Bayesian inference for sophisticated users (alternative to programming it yourself)
- Fast and scalable gradient computation
- Environment for developing new algorithms


me
social
tv
video
games
music
apps
MLB.TV
Join the VP Debate
We're polling LIVE, 9 PM ET
UFC 153
Press Conf
Kinect Cent
New Trials


If the election were held today, who would you vote for?
Barack Obama
Mitt Romney
Other
Not sure


80%
60%
40%
20%
0%
Obama
Romney
Other
Sep. 24
Oct. 01
Oct. 08
Oct. 15
Oct. 22
Oct. 29
Nov. 05

"This week, the New York Times and CBS News published a story using, in part, information from a non-probability, opt-in survey sparking concern among many in the polling community. In general, these methods have little grounding in theory and the results can vary widely based on the particular method used."
— Michael Link,
President, American Association for Public Opinion Research

**Michael W. Link** is Chief Methodologist for Research Methods at The Nielsen Company
base of experience in survey research, having worked in academia (University of South C
1999), not-for-profit research (RTI International, 1999-2004), government (Centers for Di
Prevention, 2004-2007), and the private sector (Nielsen, 2007-present). He received his
Science from the University of South Carolina. Michael's research centers around devel
methodologies for confronting some of the most pressing issues facing survey research,
techniques for improving survey participation and data quality (use of address-based san
call screening technologies), methodological issues involving use of multiple modes in da
mail, CATI, field, mobile, meters), and obtaining participation from hard-to-survey populat
isolated, racial and ethnic groups). His numerous research articles have appeared in *Pu*
*Quarterly* and other leading scientific journals.

An AAPOR member since 1993, Michael served as AAPOR Conference Chair in back-to
& 2010), a member of both the Cell Phone and Online task forces, an instructor for an A
numerous short-courses, a reviewer for the student paper competition on several occasi
regular reviewer for *Public Opinion Quarterly*. He is a member of SAPOR, serving from 2
President, Conference Chair, and Student Paper Competition Organizer and also a mem

In 2011 he, along with several research colleagues, received AAPOR's Warren J. Mitofsk
Award for their work on address based sampling designs. His current research focuses o
technologies, such as mobile and social platforms, as vehicles for measuring and unders
attitudes and behaviors. He will be teaching a short course on "The Role of New Techno
Augmenting, or Replacing Traditional Surveys" at the 2012 AAPOR conference.

# Nielsen feels the heat of competition as it flubs its ratings of news broadcasts, putting ABC ahead of NBC

BY DON KAPLAN

**In spite of the goof, its global president took time to slam rival Rentrak, which collects different kind of data from viewers**

NEW YORK DAILY NEWS / Sunday, October 19, 2014, 2:00 AM

A A A

**MEDIA**

# *TV Ratings by Nielsen Had Errors for Months*

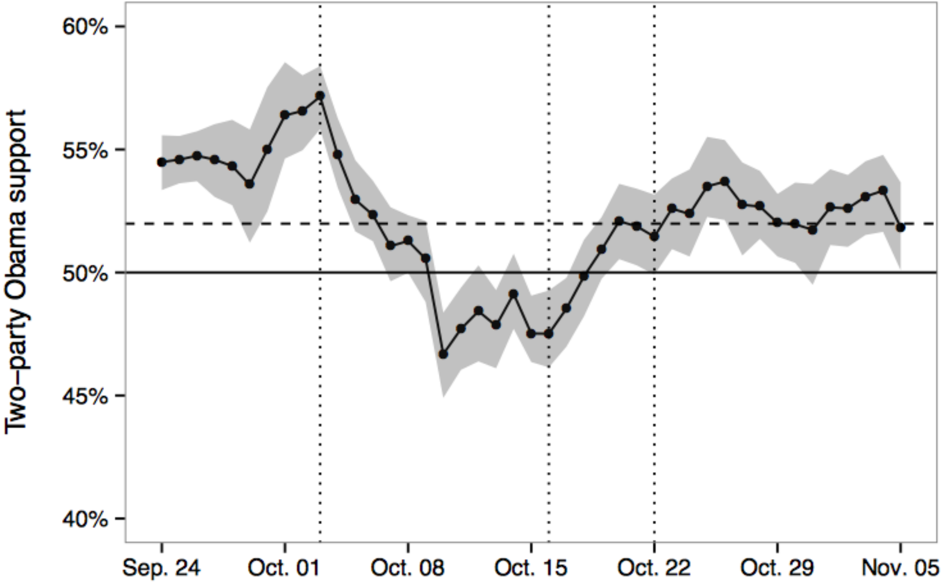
By **BILL CARTER** and **EMILY STEEL** OCT. 10, 2014

Email

Share

Nielsen, the television research firm, acknowledged on Friday that it had been reporting inaccurate ratings for the broadcast networks for the last seven months, a mistake that raises questions about the company's increasingly criticized system for measuring TV audiences.

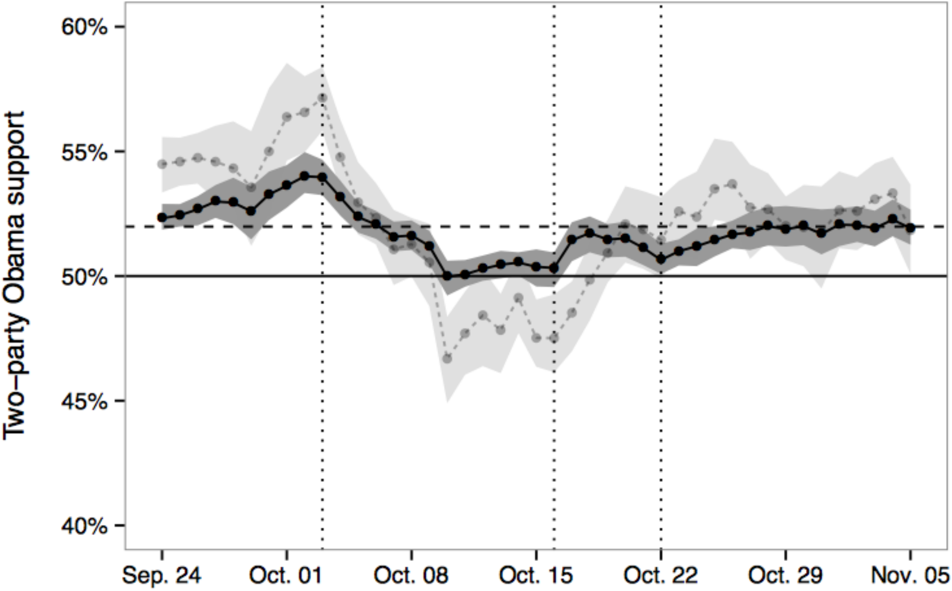
# Xbox estimates, adjusting for demographics



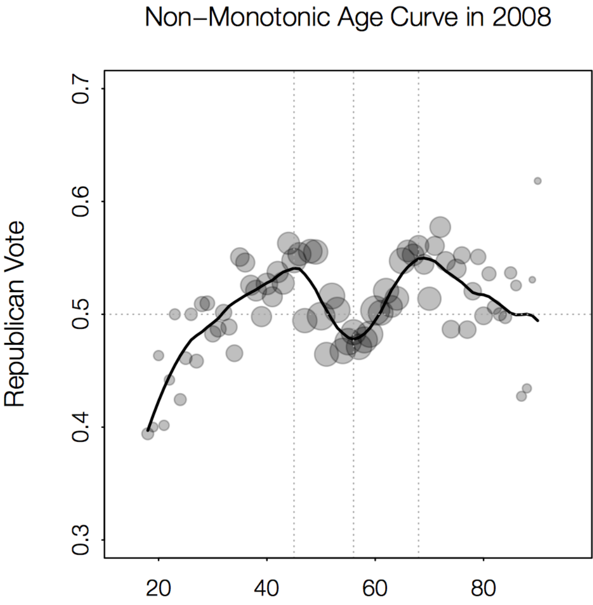

The Presidential
2012 DEBATES
#nbcpolitics

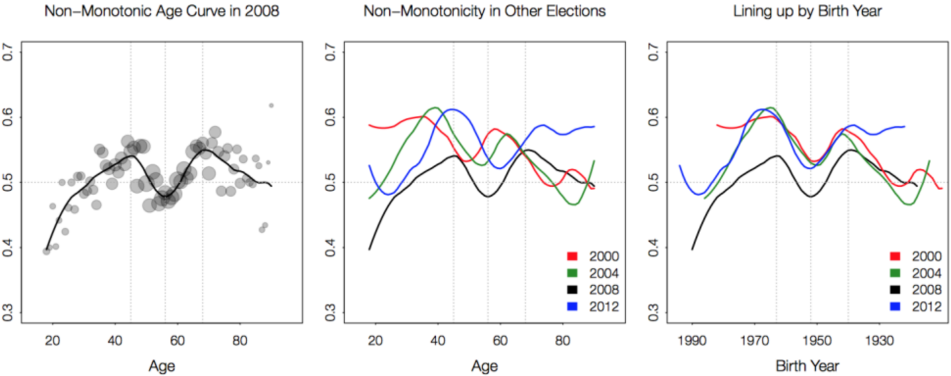

- Karl Rove, *Wall Street Journal*, 7 Oct: "Mr. Romney's bounce is significant."
- Nate Silver, *New York Times*, 6 Oct: "Mr. Romney has not only improved his own standing but also taken voters away from Mr. Obama's column."

# Xbox estimates, adjusting for demographics and partisanship

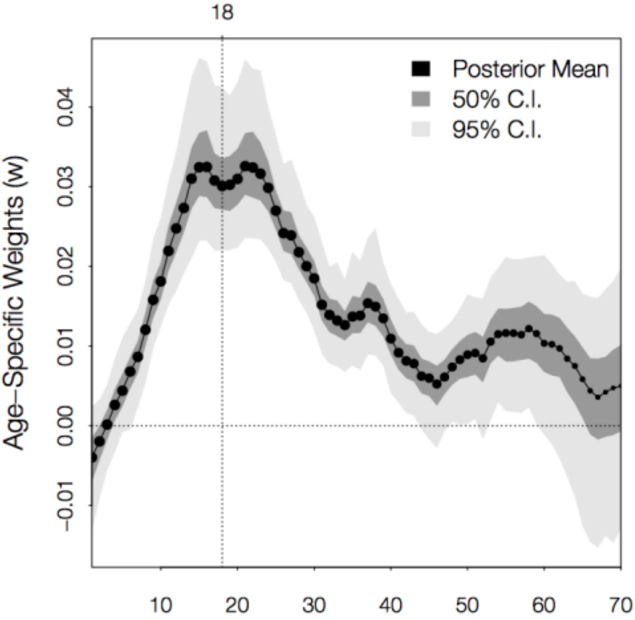


# Jimmy Carter Republicans and George W. Bush Democrats

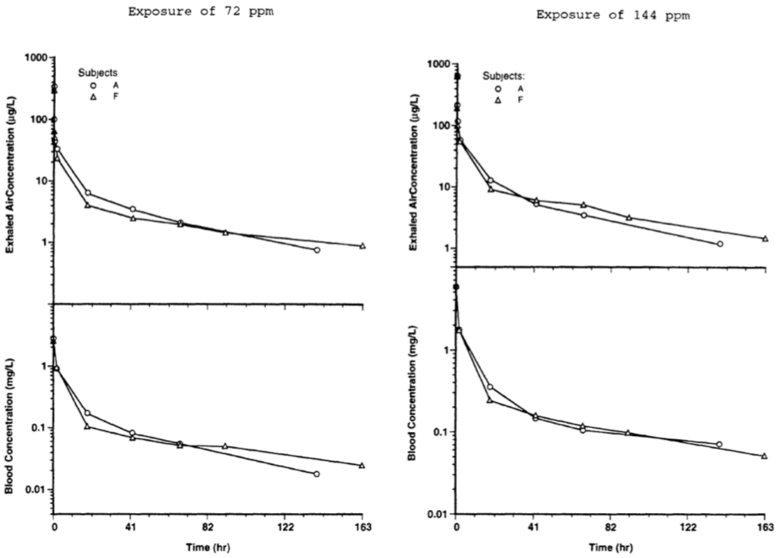



Non−Monotonic Age Curve in 2008
Non−Monotonicity in Other Elections
Lining up by Birth Year
0.3
0.4
0.5
0.6
0.7
20
40
60
80
Age
2000
2004
2008
2012
1990
1970
1950
1930
Birth Year

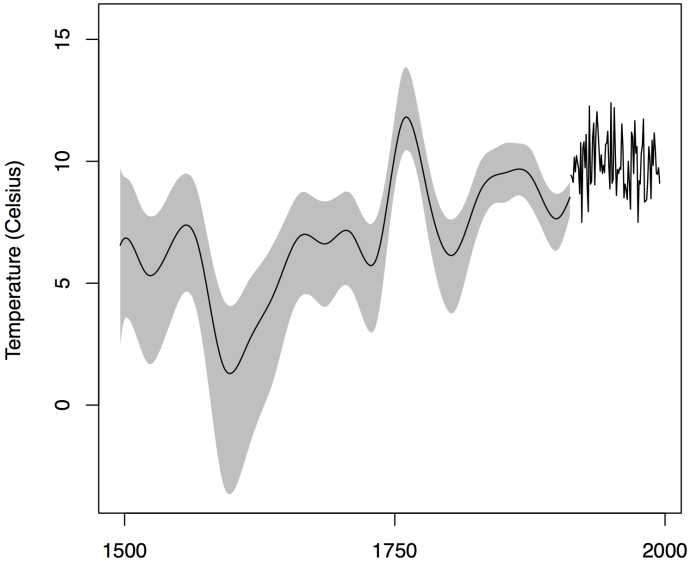
# The Formative Years



# Toxicology


Exposure of 72 ppm
Exposure of 144 ppm
Subjects
O A
Δ F
Subjects:
O A
Δ F
Exhaled AirConcentration (µg/L)
Blood Concentration (mg/L)
1000
100
10
1
0.1
0.01
0
41
82
122
163
Time (hr)

# Earth science



# Lots of other applications

Astronomy, ecology, linguistics, epidemiology, soil science, . . .

# Steps of Bayesian data analysis

# Steps of Bayesian data analysis

- Model building

# Steps of Bayesian data analysis

- Model building
- Inference

# Steps of Bayesian data analysis

- Model building
- Inference
- Model checking

# Steps of Bayesian data analysis

- Model building
- Inference
- Model checking
- Model understanding and improvement

# Background on Bayesian computation

# Background on Bayesian computation

- Point estimates and standard errors

# Background on Bayesian computation

- Point estimates and standard errors
- Hierarchical models

# Background on Bayesian computation

- Point estimates and standard errors
- Hierarchical models
- Posterior simulation

# Background on Bayesian computation

- Point estimates and standard errors
- Hierarchical models
- Posterior simulation
- Markov chain Monte Carlo (Gibbs sampler and Metropolis algorithm)

# Background on Bayesian computation

- Point estimates and standard errors
- Hierarchical models
- Posterior simulation
- Markov chain Monte Carlo (Gibbs sampler and Metropolis algorithm)
- Hamiltonian Monte Carlo

# Solving problems

# Solving problems

- *Problem*: Gibbs too slow, Metropolis too problem-specific

# Solving problems

- *Problem*: Gibbs too slow, Metropolis too problem-specific
- *Solution*: Hamiltonian Monte Carlo

# Solving problems

- *Problem*: Gibbs too slow, Metropolis too problem-specific
- *Solution*: Hamiltonian Monte Carlo

- *Problem*: Interpreters too slow, won't scale

# Solving problems

- *Problem*: Gibbs too slow, Metropolis too problem-specific
- *Solution*: Hamiltonian Monte Carlo

- *Problem*: Interpreters too slow, won't scale
- *Solution*: Compilation

# Solving problems

- *Problem*: Gibbs too slow, Metropolis too problem-specific
- *Solution*: Hamiltonian Monte Carlo

- *Problem*: Interpreters too slow, won't scale
- *Solution*: Compilation

- *Problem*: Need gradients of log posterior for HMC

# Solving problems

- *Problem*: Gibbs too slow, Metropolis too problem-specific
- *Solution*: Hamiltonian Monte Carlo

- *Problem*: Interpreters too slow, won't scale
- *Solution*: Compilation

- *Problem*: Need gradients of log posterior for HMC
- *Solution*: Reverse-mode algorithmic differentation

# Solving problems

- *Problem*: Gibbs too slow, Metropolis too problem-specific
- *Solution*: Hamiltonian Monte Carlo

- *Problem*: Interpreters too slow, won't scale
- *Solution*: Compilation

- *Problem*: Need gradients of log posterior for HMC
- *Solution*: Reverse-mode algorithmic differentation

- *Problem*: Existing algo-diff slow, limited, unextensible

# Solving problems

- *Problem*: Gibbs too slow, Metropolis too problem-specific
- *Solution*: Hamiltonian Monte Carlo

- *Problem*: Interpreters too slow, won't scale
- *Solution*: Compilation

- *Problem*: Need gradients of log posterior for HMC
- *Solution*: Reverse-mode algorithmic differentation

- *Problem*: Existing algo-diff slow, limited, unextensible
- *Solution*: Our own algo-diff

# Solving problems

- *Problem*: Gibbs too slow, Metropolis too problem-specific
- *Solution*: Hamiltonian Monte Carlo

- *Problem*: Interpreters too slow, won't scale
- *Solution*: Compilation

- *Problem*: Need gradients of log posterior for HMC
- *Solution*: Reverse-mode algorithmic differentation

- *Problem*: Existing algo-diff slow, limited, unextensible
- *Solution*: Our own algo-diff

- *Problem*: Algo-diff requires fully templated functions

# Solving problems

- *Problem*: Gibbs too slow, Metropolis too problem-specific
- *Solution*: Hamiltonian Monte Carlo

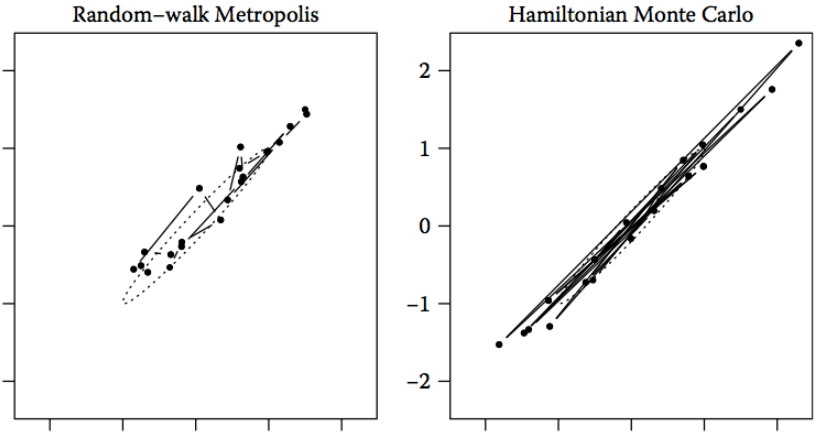
- *Problem*: Interpreters too slow, won't scale
- *Solution*: Compilation

- *Problem*: Need gradients of log posterior for HMC
- *Solution*: Reverse-mode algorithmic differentation

- *Problem*: Existing algo-diff slow, limited, unextensible
- *Solution*: Our own algo-diff

- *Problem*: Algo-diff requires fully templated functions
- *Solution*: Our own density library, Eigen linear algebra

# Radford Neal (2011) on Hamiltonian Monte Carlo

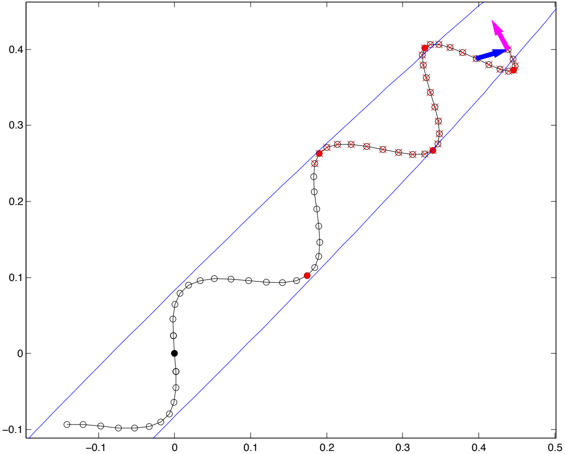



"One practical impediment to the use of Hamiltonian Monte Carlo is the need to select suitable values for the leapfrog stepsize, $\epsilon$, and the number of leapfrog steps $L$ ... Tuning HMC will usually require preliminary runs with trial values for $\epsilon$ and $L$ ... Unfortunately, preliminary runs can be misleading ..."

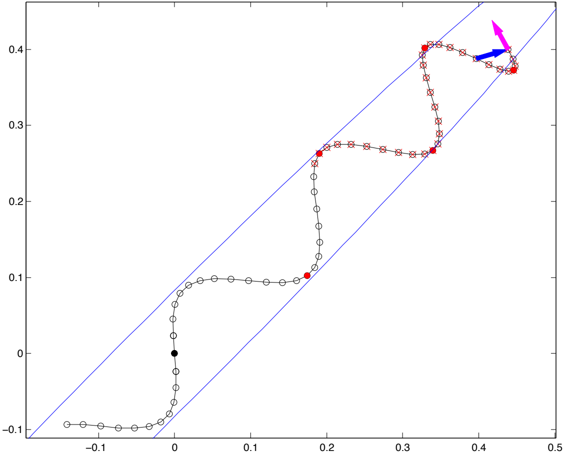
# The No U-Turn Sampler

- Created by Matt Hoffman
- Run the HMC steps until they start to turn around (bend with an angle $> 180^\circ$)
- Computationally efficient
- Requires no tuning of #steps
- Complications to preserve detailed balance

# NUTS Example Trajectory

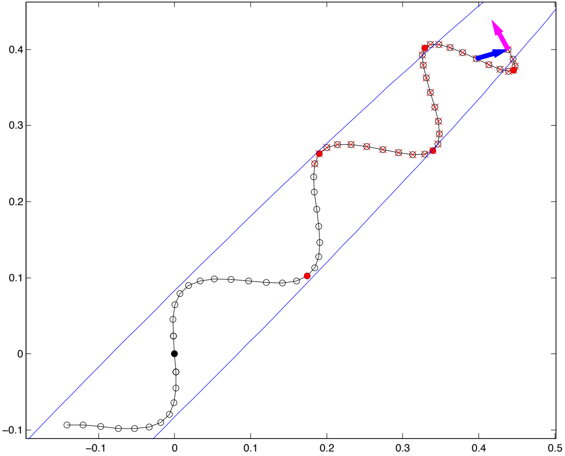


# NUTS Example Trajectory



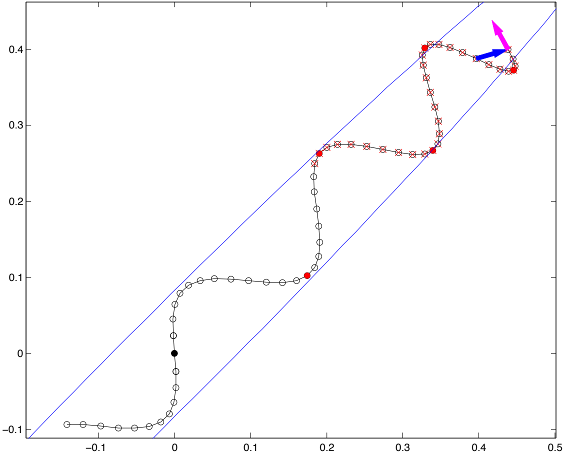

- Blue ellipse is contour of target distribution

# NUTS Example Trajectory



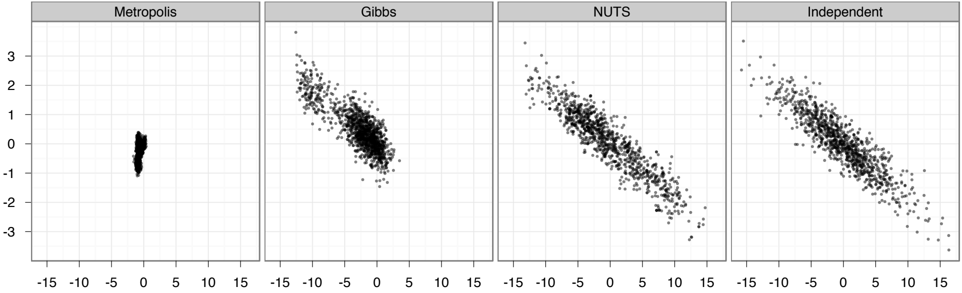

- Blue ellipse is contour of target distribution
- Initial position at black solid circle

# NUTS Example Trajectory

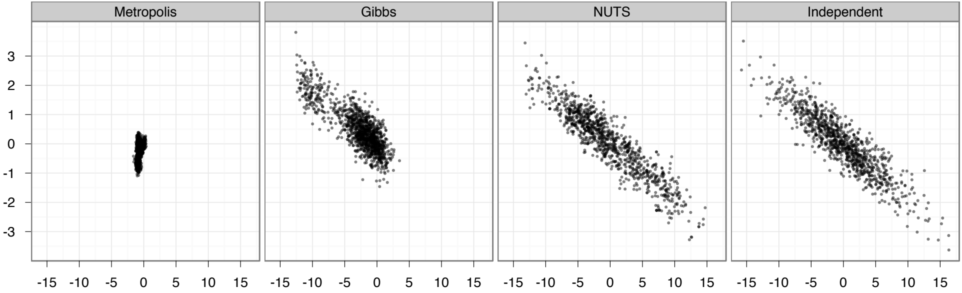



- Blue ellipse is contour of target distribution
- Initial position at black solid circle
- Arrows indicate a U-turn in momentum

# NUTS vs. Gibbs and Metropolis

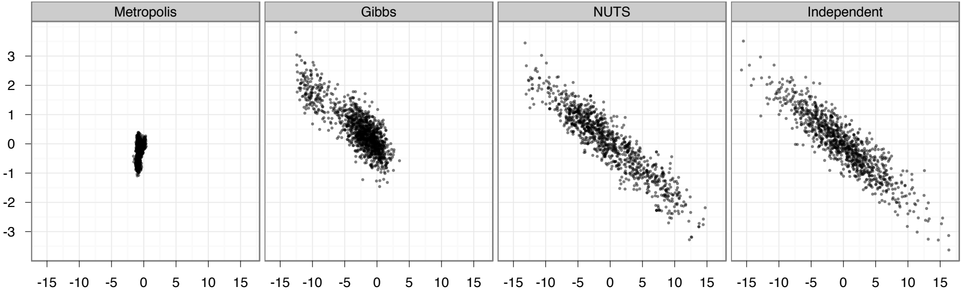


# NUTS vs. Gibbs and Metropolis



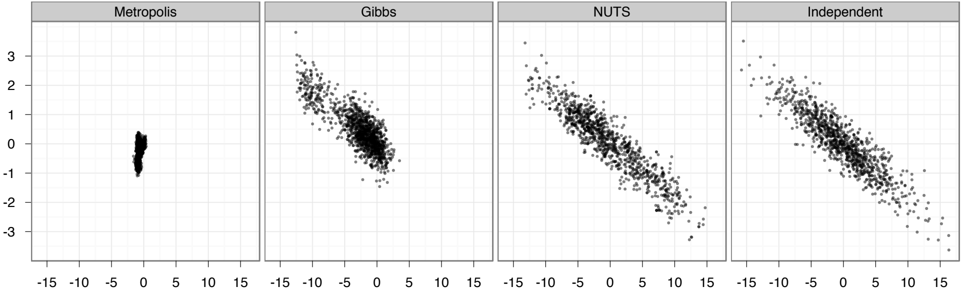

- Two dimensions of highly correlated 250-dim distribution

# NUTS vs. Gibbs and Metropolis




- Two dimensions of highly correlated 250-dim distribution
- 1M samples from Metropolis, 1M from Gibbs (thin to 1K)

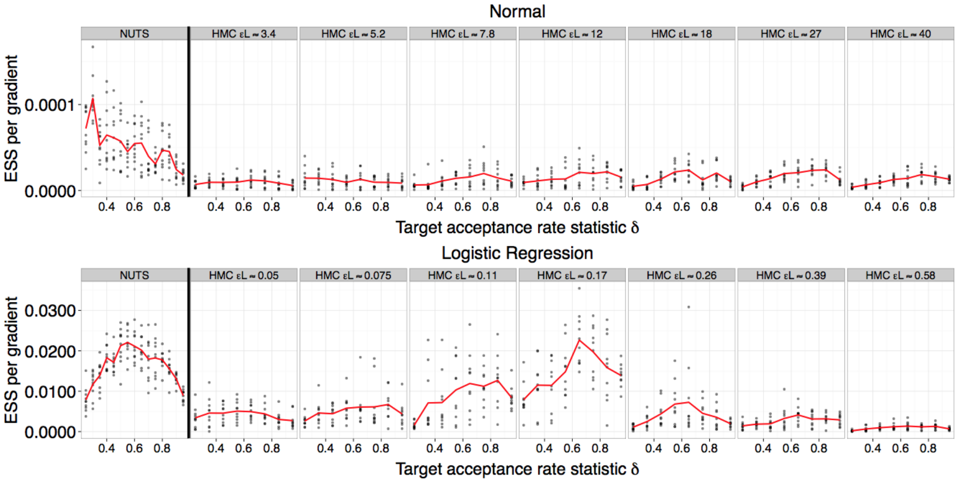
# NUTS vs. Gibbs and Metropolis




- Two dimensions of highly correlated 250-dim distribution
- 1M samples from Metropolis, 1M from Gibbs (thin to 1K)
- 1K samples from NUTS, 1K independent draws

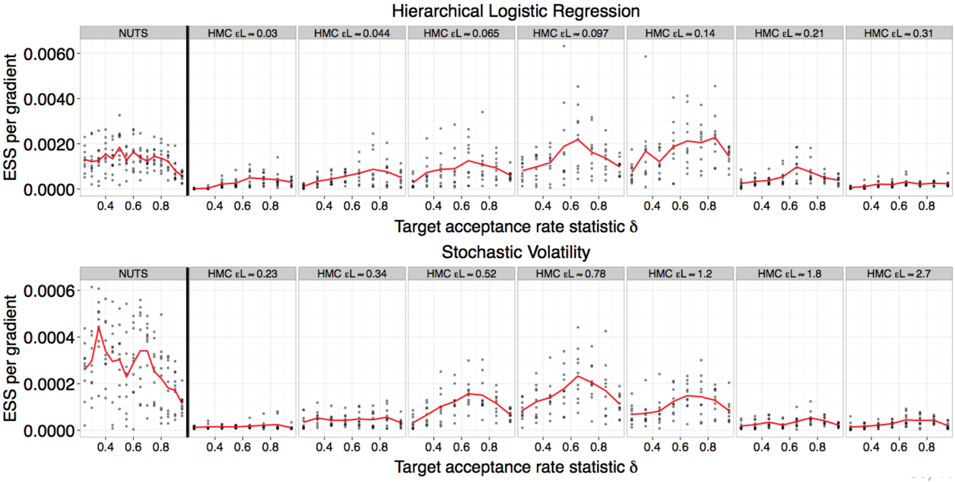
# NUTS vs. Basic HMC

- 250-D normal and logistic regression models
- Vertical axis shows effective #sims (big is good)
- (Left) NUTS; (Right) HMC with increasing $t = \epsilon L$



# NUTS vs. Basic HMC II

- Hierarchical logistic regression and stochastic volatility
- Simulation time is step size $\epsilon$ times #steps $L$
- NUTS can beat optimally tuned HMC



# Solving more problems in Stan

# Solving more problems in Stan

- *Problem*: Need ease of use of BUGS

# Solving more problems in Stan

- *Problem*: Need ease of use of BUGS
- *Solution*: Compile directed graphical model language

# Solving more problems in Stan

- *Problem*: Need ease of use of BUGS
- *Solution*: Compile directed graphical model language

- *Problem*: Need to tune parameters for HMC

# Solving more problems in Stan

- *Problem*: Need ease of use of BUGS
- *Solution*: Compile directed graphical model language

- *Problem*: Need to tune parameters for HMC
- *Solution*: Auto tuning, adaptation

# Solving more problems in Stan

- *Problem*: Need ease of use of BUGS
- *Solution*: Compile directed graphical model language

- *Problem*: Need to tune parameters for HMC
- *Solution*: Auto tuning, adaptation

- *Problem*: Efficient up-to-proportion density calcs

# Solving more problems in Stan

- *Problem*: Need ease of use of BUGS
- *Solution*: Compile directed graphical model language

- *Problem*: Need to tune parameters for HMC
- *Solution*: Auto tuning, adaptation

- *Problem*: Efficient up-to-proportion density calcs
- *Solution*: Density template metaprogramming

# Solving more problems in Stan

- *Problem*: Need ease of use of BUGS
- *Solution*: Compile directed graphical model language

- *Problem*: Need to tune parameters for HMC
- *Solution*: Auto tuning, adaptation

- *Problem*: Efficient up-to-proportion density calcs
- *Solution*: Density template metaprogramming

- *Problem*: Limited error checking, recovery

# Solving more problems in Stan

- *Problem*: Need ease of use of BUGS
- *Solution*: Compile directed graphical model language

- *Problem*: Need to tune parameters for HMC
- *Solution*: Auto tuning, adaptation

- *Problem*: Efficient up-to-proportion density calcs
- *Solution*: Density template metaprogramming

- *Problem*: Limited error checking, recovery
- *Solution*: Static model typing, informative exceptions

# Solving more problems in Stan

- *Problem*: Need ease of use of BUGS
- *Solution*: Compile directed graphical model language

- *Problem*: Need to tune parameters for HMC
- *Solution*: Auto tuning, adaptation

- *Problem*: Efficient up-to-proportion density calcs
- *Solution*: Density template metaprogramming

- *Problem*: Limited error checking, recovery
- *Solution*: Static model typing, informative exceptions

- *Problem*: Poor boundary behavior

# Solving more problems in Stan

- *Problem*: Need ease of use of BUGS
- *Solution*: Compile directed graphical model language

- *Problem*: Need to tune parameters for HMC
- *Solution*: Auto tuning, adaptation

- *Problem*: Efficient up-to-proportion density calcs
- *Solution*: Density template metaprogramming

- *Problem*: Limited error checking, recovery
- *Solution*: Static model typing, informative exceptions

- *Problem*: Poor boundary behavior
- *Solution*: Calculate limits (e.g. $\lim_{x \to 0} x \log x$)

# Solving more problems in Stan

- *Problem*: Need ease of use of BUGS
- *Solution*: Compile directed graphical model language

- *Problem*: Need to tune parameters for HMC
- *Solution*: Auto tuning, adaptation

- *Problem*: Efficient up-to-proportion density calcs
- *Solution*: Density template metaprogramming

- *Problem*: Limited error checking, recovery
- *Solution*: Static model typing, informative exceptions

- *Problem*: Poor boundary behavior
- *Solution*: Calculate limits (e.g. $\lim_{x \to 0} x \log x$)

# Solving more problems in Stan

- *Problem*: Need ease of use of BUGS
- *Solution*: Compile directed graphical model language

- *Problem*: Need to tune parameters for HMC
- *Solution*: Auto tuning, adaptation

- *Problem*: Efficient up-to-proportion density calcs
- *Solution*: Density template metaprogramming

- *Problem*: Limited error checking, recovery
- *Solution*: Static model typing, informative exceptions

- *Problem*: Poor boundary behavior
- *Solution*: Calculate limits (e.g. $\lim_{x \to 0} x \log x$)

- *Problem*: Restrictive licensing (e.g., closed, GPL, etc.)

# Solving more problems in Stan

- *Problem*: Need ease of use of BUGS
- *Solution*: Compile directed graphical model language

- *Problem*: Need to tune parameters for HMC
- *Solution*: Auto tuning, adaptation

- *Problem*: Efficient up-to-proportion density calcs
- *Solution*: Density template metaprogramming

- *Problem*: Limited error checking, recovery
- *Solution*: Static model typing, informative exceptions

- *Problem*: Poor boundary behavior
- *Solution*: Calculate limits (e.g. $\lim_{x \to 0} x \log x$)

- *Problem*: Restrictive licensing (e.g., closed, GPL, etc.)
- *Solution*: Open-source, BSD license

# New stuff: Differential equation models

Simple harmonic oscillator:

$$\begin{aligned}\frac{dz_1}{dt} &= -z_2\\ \frac{dz_2}{dt} &= -z_1 - \theta z_2\end{aligned}$$

with observations $(y_1, y_2)_t, t = 1, \ldots, T$:

$$\begin{aligned}y_{1t} &\sim \mathrm{N}(z_1(t), \sigma_1^2)\\ y_{2t} &\sim \mathrm{N}(z_2(t), \sigma_2^2)\end{aligned}$$

Given data $(y_1, y_2)_t, t = 1, \ldots, T$,
estimate initial state $(y_1, y_2)_{t=0}$ and parameter $\theta$

# Stan program: 1

```
functions {
  real[] sho(real t, real[] y, real[] theta, real[] x_r, int[] x_i) {
    real dydt[2];
    dydt[1] <- y[2];
    dydt[2] <- -y[1] - theta[1] * y[2];
    return dydt;
  }
}
data {
  int<lower=1> T;
  real y[T,2];
  real t0;
  real ts[T];
}
transformed data {
  real x_r[0];
  int x_i[0];
}
```

# Stan program: 2

```
parameters {
  real y0[2];
  vector<lower=0>[2] sigma;
  real theta[1];
}
model {
  real z[T,2];
  sigma ~ cauchy(0,2.5);
  theta ~ normal(0,1);
  y0 ~ normal(0,1);
  z <- integrate_ode(sho, y0, t0, ts, theta, x_r, x_i);
  for (t in 1:T)
    y[t] ~ normal(z[t], sigma);
}
```

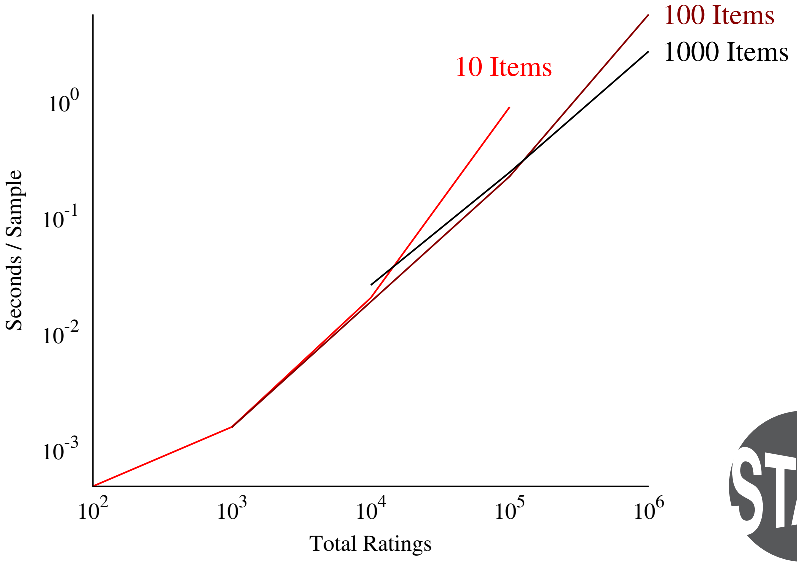
# Stan output

Run RStan with data simulated from
$\theta = 0.15$, $y_0 = (1, 0)$, and $\sigma = 0.1$:

```
Inference for Stan model: sho.
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 | 2.5% | 25% | 50% | 75% | 97.5% | n_eff | Rhat |
|---|---|---|---|---|---|---|---|---|---|---|
| y0[1] | 1.05 | 0.00 | 0.09 | 0.87 | 0.98 | 1.05 | 1.10 | 1.23 | 1172 | 1 |
| y0[2] | -0.06 | 0.00 | 0.06 | -0.18 | -0.10 | -0.06 | -0.02 | 0.06 | 1524 | 1 |
| sigma[1] | 0.14 | 0.00 | 0.04 | 0.08 | 0.11 | 0.13 | 0.16 | 0.25 | 1354 | 1 |
| sigma[2] | 0.11 | 0.00 | 0.03 | 0.06 | 0.08 | 0.10 | 0.12 | 0.18 | 1697 | 1 |
| theta[1] | 0.15 | 0.00 | 0.04 | 0.08 | 0.13 | 0.15 | 0.17 | 0.22 | 1112 | 1 |
| lp__ | 28.97 | 0.06 | 1.80 | 24.55 | 27.95 | 29.37 | 30.29 | 31.35 | 992 | 1 |

# Big Data, Big Model, Scalable Computing



# Thinking about scalability

# Thinking about scalability

- Hierarchical item response model:

| | | | | Stan | | JAGS | |
|---|---|---|---|---|---|---|---|
| # items | # raters | # groups | # data | time | memory | time | memory |
| 20 | 2,000 | 100 | 40,000 | :02m | 16MB | :03m | 220MB |
| 40 | 8,000 | 200 | 320,000 | :16m | 92MB | :40m | 1400MB |
| 80 | 32,000 | 400 | 2,560,000 | 4h:10m | 580MB | :??m | ?MB |

# Thinking about scalability

- Hierarchical item response model:

| | | | | Stan | | JAGS | |
|---|---|---|---|---|---|---|---|
| # items | # raters | # groups | # data | time | memory | time | memory |
| 20 | 2,000 | 100 | 40,000 | :02m | 16MB | :03m | 220MB |
| 40 | 8,000 | 200 | 320,000 | :16m | 92MB | :40m | 1400MB |
| 80 | 32,000 | 400 | 2,560,000 | 4h:10m | 580MB | :??m | ?MB |

- Also, Stan generated 4x effective sample size per iteration

# Future work

# Future work

- Programming

# Future work

- Programming
  - Faster gradients and higher-order derivatives

# Future work

- Programming
  - Faster gradients and higher-order derivatives
  - Functions

# Future work

- Programming
  - Faster gradients and higher-order derivatives
  - Functions
- Statistical algorithms

# Future work

- Programming
  - Faster gradients and higher-order derivatives
  - Functions
- Statistical algorithms
  - Riemannian Hamiltonian Monte Carlo

# Future work

- Programming
  - Faster gradients and higher-order derivatives
  - Functions
- Statistical algorithms
  - Riemannian Hamiltonian Monte Carlo
  - (Penalized) mle

# Future work

- Programming
  - Faster gradients and higher-order derivatives
  - Functions
- Statistical algorithms
  - Riemannian Hamiltonian Monte Carlo
  - (Penalized) mle
  - (Penalized) marginal mle

# Future work

- Programming
  - Faster gradients and higher-order derivatives
  - Functions
- Statistical algorithms
  - Riemannian Hamiltonian Monte Carlo
  - (Penalized) mle
  - (Penalized) marginal mle
  - Black-box variational Bayes

# Future work

- Programming
  - Faster gradients and higher-order derivatives
  - Functions
- Statistical algorithms
  - Riemannian Hamiltonian Monte Carlo
  - (Penalized) mle
  - (Penalized) marginal mle
  - Black-box variational Bayes
  - Data partitioning and expectation propagation