Reproducibility and Stan

Aki Vehtari, Aalto University

and

Stan development team

with special thanks to

Andrew Gelman, Ben Goodrich, Bob Carpenter, Breck Baldwin, Daniel Lee, Daniel Simpson, Eric Novik, Jonah Gabry, Lauren Kennedy, Michael Betancourt, Rob Trangucci, Sean Talts

Funding acknowledgement: DARPA agreement D17AC00001, Academy of Finland grant 313122

mc-stan.org

Stan - probabilistic programming framework

- Language, inference engine, user interfaces, documentation, case studies, diagnostics, packages, ...
- More than ten thousand users in social, biological, and physical sciences, medicine, engineering, and business
- Several full time developers, 34 in dev team, more than 100 contributors
- R, Python, Julia, Scala, Stata, Matlab, command line interfaces
- More than 50 R packages using Stan
- Hamiltonian Monte Carlo / No-U-Turn-Sampling, ADVI, optimization

Reproducibility and Stan

1) Reproducibility of StanCon contributed talks
2) Reproducibility of Stan
3) Validation of Bayesian inference code

StanCon 2018
Helsinki, 29-31 Aug
2018

StanCon 2018
Helsinki, 29-31 Aug 2018

- StanCon conference contributed oral submissions:
  - Notebook submission
  - On web page; notebook, link to a code repo, slides
  - DOI via Zenodo

StanCon 2018
Helsinki, 29-31 Aug 2018

- StanCon conference contributed oral submissions:
  - Notebook submission
  - On web page; notebook, link to a code repo, slides
  - DOI via Zenodo
- 50% of invited speakers have also provided the code

Materials from StanCon

StanCon's version of conference proceedings is a collection of contributed talks based on interactive notebooks. Every submission is peer reviewed by at least two reviewers. The reviewers are members of the Stan Conference Organizing Committee and the Stan Developmemt Team. This repository contains all of the accepted notebooks as well as any supplementary materials required for building the notebooks. The slides presented at the conference are also included.

Contents:

StanCon 2017 | January 21, Columbia University, New York

2017 Peer reviewed contributed talks

Twelve Cities: Does lowering speed limits save pedestrian lives?

- Authors: Jonathan Auerbach, Rob Trangucci (Columbia University)

We investigate whether American cities can expect to achieve a meaningful reduction in pedestrian deaths by lowering the

Differential Equation Based Models in Stan

- Authors: Charles Margossian, Bill Gillespie (Metrum Research Group)

Differential equations can help us model sophisticated processes in biology, physics, and many other fields. Over the past year, the Stan team has developed many tools to tackle models based on differential equations.

DOI 10.5281/zenodo.1284264

Links:

- Video
- Notebook and materials
- Slides
- http://metrumrg.com/

Differential Equations Based Models in Stan

Charles Margossian and Bill Gillespie

January 14, 2017

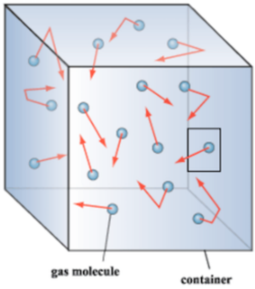
1. Introduction

Differential Equations can help us model sophisticated processes in biology, physics, and many other fields. Over the past year, the Stan team has developed many tools to tackle models based on differential equations.

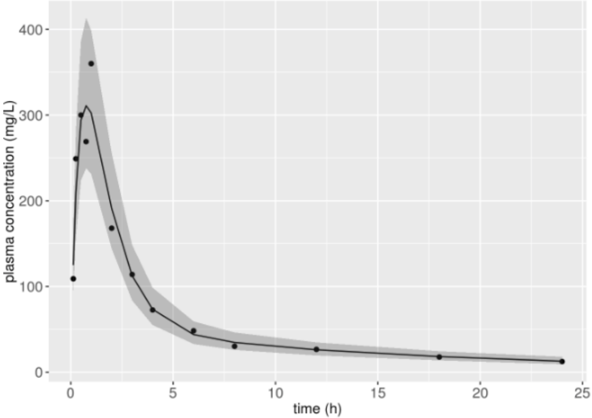
1.1 Why Use Ordinary Differential Equations (ODEs)?

We deal with an ODE when we want to determine a function $y(t)$ at a specific time but only know the derivative of that function, $\frac{dy}{dt}$. In other words, we know the rate at which a quantity of interest changes but not the quantity itself. In many scenarios, the rate depends on the quantity itself.

To get a basic intuition, let us consider an example. Imagine a gas container with a hole in it. We can think of the gas as being made of molecules that move randomly in the container. Each molecule has a small chance of leaking through the hole. Thus the more molecules there are inside the container, the higher the number of escaping molecules per unit time.



```
p1 <- ggplot(pred, aes(x = time, y = cObs))
p1 <- p1 + geom_point() +
  labs(x = "time (h)", y = "plasma concentration (mg/L)") +
  theme(text = element_text(size = 12), axis.text = element_text(size = 12),
        legend.position = "none", strip.text = element_text(size = 8))
p1 + geom_line(aes(x = time, y = median)) +
  geom_ribbon(aes(ymin = lb, ymax = ub), alpha = 0.25)
```



We're not done yet! We also have data for the drug response. So let's see if the model's predictions agree with this data too:

```
## predictions for future observations for drug effect
pred <- as.data.frame(fit, pars = "respObsPred") %>%
```

stan-dev / **stancon_talks**

Watch 26 | Star 111 | Fork 44

<> Code | Issues 0 | Pull requests 0 | Projects 0 | Wiki | Insights

Branch: **master** | **stancon_talks** / 2017 / Contributed-Talks / **05_margossian** /

Create new file | Upload files | Find file | History

jgabry Remove space from directory name — Latest commit `e018641` on 23 Jan 2017

..		
figures	Remove space from directory name	2 years ago
models	Remove space from directory name	2 years ago
tools	Remove space from directory name	2 years ago
ODEStan.Rmd	Remove space from directory name	2 years ago
ODEStan.html	Remove space from directory name	2 years ago
pkgSetup.R	Remove space from directory name	2 years ago

```
transformed data {
  vector[nObs] logCObs;
  int nCmt;

  logCObs = log(cObs);
  nCmt = 3; ## Fixed. The code specifically handles models with 3 compartments.
}

parameters {
  real<lower = 0> CL;
  real<lower = 0> Q;
  real<lower = 0> V1;
  real<lower = 0> V2;
  real<lower = 0> ka;
  real<lower = 0> sigma;
}

transformed parameters {
  vector<lower = 0>[nt] cHat;
  vector<lower = 0>[nObs] cHatObs;
  matrix<lower = 0>[nt, nCmt] x;

  x = twoCptModel(time, amt, cmt, evid,
                           CL, Q, V1, V2, ka);

  cHat = col(x, 2) ./ V1;

  cHatObs = cHat[iObs]; ## predictions for observed data records
}

model {
  CL ~ lognormal(log(10), 0.25);
  Q ~ lognormal(log(15), 0.5);
  V1 ~ lognormal(log(35), 0.25);
  V2 ~ lognormal(log(105), 0.5);
  sigma ~ cauchy(0, 1);

  logCObs ~ normal(log(cHatObs), sigma);
}

generated quantities{
    real cObsPred[nObs];
```

Reproducibility of notebooks

5 Original Computing Environment

```
## Warning in readLines(makevars_file): incomplete final line found on 'C:
## \Users\Bebop/.R/Makevars'
```

```
## CXXFLAGS=-O3 -Wno-unused-variable -Wno-unused-function
```

```
## Session info ------------------------------------------------------------
```

```
##  setting  value
##  version  R version 3.4.1 (2017-06-30)
##  system   x86_64, mingw32
##  ui       RTerm
##  language (EN)
##  collate  Spanish_Argentina.1252
##  tz       America/Buenos_Aires
##  date     2018-01-31
```

```
## Packages ----------------------------------------------------------------
```

```
##  package      * version   date       source
##  BH             1.65.0-1  2017-08-24 CRAN (R 3.4.1)
##  colorspace     1.3-2     2016-12-14 CRAN (R 3.4.1)
##  dichromat      2.0-0     2013-01-24 CRAN (R 3.4.1)
##  digest         0.6.12    2017-01-27 CRAN (R 3.4.1)
##  ggplot2      * 2.2.1     2016-12-30 CRAN (R 3.4.1)
```

Reproducibility of R session

checkpoint: Install Packages from Snapshots on the Checkpoint Server for Reproducibility

The goal of checkpoint is to solve the problem of package reproducibility in R. Specifically, checkpoint allows you to install packages as they existed on CRAN on a specific snapshot date as if you had a CRAN time machine. To achieve reproducibility, the checkpoint() function installs the packages required or called by your project and scripts to a local library exactly as they existed at the specified point in time. Only those packages are available to your project, thereby avoiding any package updates that came later and may have altered your results. In this way, anyone using checkpoint's checkpoint() can ensure the reproducibility of your scripts or projects at any time. To create the snapshot archives, once a day (at midnight UTC) Microsoft refreshes the Austria CRAN mirror on the "Microsoft R Archived Network" server (<https://mran.microsoft.com/>). Immediately after completion of the rsync mirror process, the process takes a snapshot, thus creating the archive. Snapshot archives exist starting from 2014-09-17.

Version:	0.4.3
Depends:	R (≥ 3.0.0)
Imports:	utils
Suggests:	knitr, rmarkdown, testthat (≥ 0.9), MASS, darts, mockery, cli
Published:	2017-12-19
Author:	Microsoft Corporation
Maintainer:	Rich Calaway <richcala at microsoft.com>

Reproducibility of R session

- Checkpoint handles CRAN, but not git repos
- Checkpoint doesn't handle external libraries and compiler environment

Reproducibility of Stan

- There is more code for tests than algorithms
- Continuous integration tests
- Regression tests

Reproducibility of Stan

Stan is designed to allow full reproducibility. However, this is only possible up to the external constraints imposed by floating point arithmetic.

Stan results will only be exactly reproducible if *all* of the following components are *identical*:

- Stan version
- Stan interface (RStan, PyStan, CmdStan) and version, plus version of interface language (R, Python, shell)
- versions of included libraries (Boost and Eigen)
- operating system version
- computer hardware including CPU, motherboard and memory
- C++ compiler, including version, compiler flags, and linked libraries
- same configuration of call to Stan, including random seed, chain ID, initialization and data

Time and machine independent reproducibility of stochastic computation

- In case of MCMC, even small differences in floating point arithmetic can lead to very different Markov chains
- To freeze everything containers could be used...

Reproducibility of inference algorithms

- If you make changes to your existing code
  or if you implement an algorithm from some paper
  - how do you know the code is correct?

Reproducibility of inference algorithms

- If you make changes to your existing code
  or if you implement an algorithm from some paper
  - how do you know the code is correct?
- In case of stochastic algorithms and differences in floating point arithmetic make bitwise comparison impossible

Is your code producing draws from the posterior distribution?

- In case of simple models with analytic marginal posterior distributions testing easier

Is your code producing draws from the posterior distribution?

- In case of simple models with analytic marginal posterior distributions testing easier
- Picking some true parameter values, generating a single data set from the model, and comparing whether the true parameter values are inside some interval is not enough!

Validating Bayesian Inference Algorithms with Simulation-Based Calibration

Talts, Betancourt, Simpson, Vehtari, Gelman

- Sample a ground truth from the prior, $\tilde{\theta} \sim \pi(\theta)$

Validating Bayesian Inference Algorithms with Simulation-Based Calibration

Talts, Betancourt, Simpson, Vehtari, Gelman

- Sample a ground truth from the prior, $\tilde{\theta} \sim \pi(\theta)$
- Sample data from the corresponding data generating process, $\tilde{y} \sim \pi(y|\tilde{\theta})$

Validating Bayesian Inference Algorithms with Simulation-Based Calibration

Talts, Betancourt, Simpson, Vehtari, Gelman

- Sample a ground truth from the prior, $\tilde{\theta} \sim \pi(\theta)$
- Sample data from the corresponding data generating process, $\tilde{y} \sim \pi(y|\tilde{\theta})$
- Sample from the posterior $\pi(\theta \mid \tilde{y})$ using your algorithm

Validating Bayesian Inference Algorithms with Simulation-Based Calibration

Talts, Betancourt, Simpson, Vehtari, Gelman

- Sample a ground truth from the prior, $\tilde{\theta} \sim \pi(\theta)$
- Sample data from the corresponding data generating process, $\tilde{y} \sim \pi(y|\tilde{\theta})$
- Sample from the posterior $\pi(\theta \mid \tilde{y})$ using your algorithm

$$\pi(\theta) = \int \pi(\theta \mid \tilde{y})\, \pi(\tilde{y} \mid \tilde{\theta})\, \pi(\tilde{\theta}) \mathrm{d}\tilde{y}\, \mathrm{d}\tilde{\theta}$$

Validating Bayesian Inference Algorithms with Simulation-Based Calibration

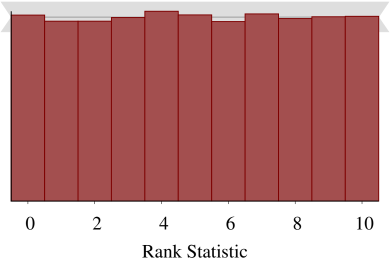
Talts, Betancourt, Simpson, Vehtari, Gelman

- Sample a ground truth from the prior, $\tilde{\theta} \sim \pi(\theta)$
- Sample data from the corresponding data generating process, $\tilde{y} \sim \pi(y|\tilde{\theta})$
- Sample from the posterior $\pi(\theta \mid \tilde{y})$ using your algorithm

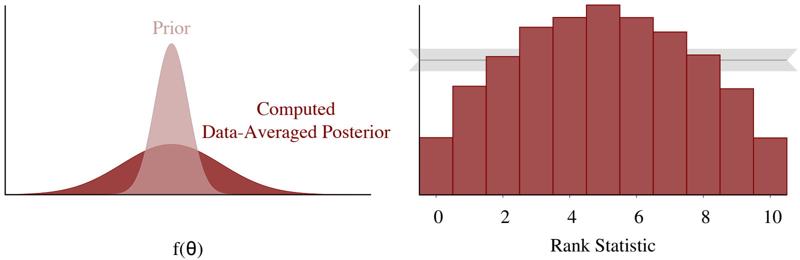
$$\pi(\theta) = \int \pi(\theta \mid \tilde{y})\, \pi(\tilde{y} \mid \tilde{\theta})\, \pi(\tilde{\theta}) \mathrm{d}\tilde{y}\, \mathrm{d}\tilde{\theta}$$

- In case of finite number of draws analyse uniformity of the rank statistic

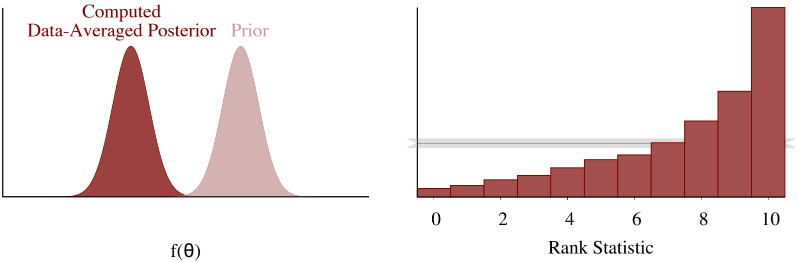
All good



Overdispersed relative to the prior



Biased relative to the prior



SBC

- Can be used to check algorithms and their implementations
- Works for MCMC and distributional approximations like INLA and VI
- Reference
  - Talts, Betancourt, Simpson, Vehtari, and Gelman (2018). Validating Bayesian Inference Algorithms with Simulation-Based Calibration. arXiv:1804.06788.

Bonus: Garden of forking paths

- Instead of picking the best result, integrate over the paths
  - you can drop only paths with practically zero contribution to the integral

Bonus: Garden of forking paths

- Instead of picking the best result, integrate over the paths
  - you can drop only paths with practically zero contribution to the integral
- Instead of selecting a model by computing model selection criterion independently for each model, condition the selection given the integral over all the models
  see, e.g.,
  - Vehtari, A., & Ojanen, J. (2012). A survey of Bayesian predictive methods for model assessment, selection and comparison. Statistics Surveys, 6, 142-228.
  - Piironen, J., & Vehtari, A. (2017). Comparison of Bayesian predictive methods for model selection. Statistics and Computing, 27(3), 711-735.