Stan for the people

Two day introductory workshop
on Bayesian modeling

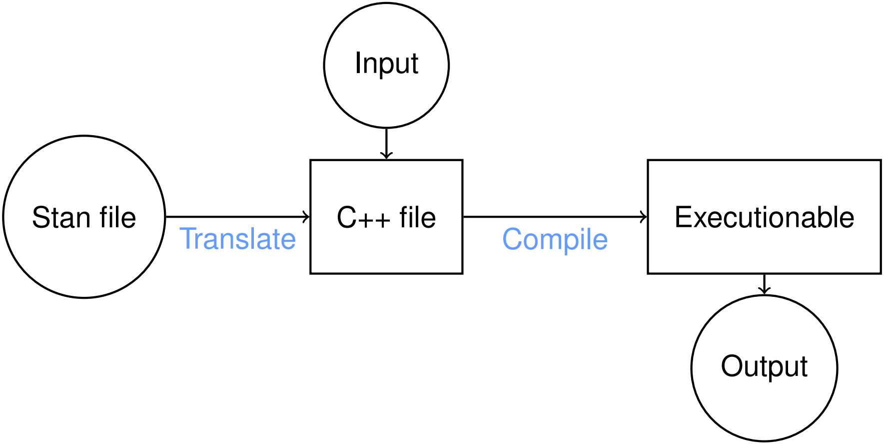
McGill University
January 25th 2019

mc-stan.org

Part III
Stan

- Stan is an expressive language for joint distributions.
- It automatically computes derivatives.
- It automatically performs inference algorithms.

How Stan works



How Stan works

- The Stan file specifies the joint distribution $p(\theta, z) = p(z|\theta)p(\theta)$
- The input includes:
  - the data, z
  - tuning parameters for the algorithm
- The output can include:
  - an approximate sample from the posterior distribution
  - summaries of the run which can help us diagnose problems.

Inference algorithms in Stan

- Hamiltonian Monte Carlo (HMC)
- No-U Turn Sampler (NUTS)
- Automatic differentiation variational inference (ADVI)
- Lasso
- ...

We can manage the Stan file, the input, and the output using a scripting language, such as:

- R
- Python
- Julia
- The command line
- ...

Example 1: linear regression

The data generating process is:

$$Y \sim \mathrm{Normal}\left(X\beta, \sigma^2\right)$$

Our goal is to estimate $\theta = (\beta, \sigma^2)$, based on the observation $Z = (X, Y)$ and prior knowledge we have of β and σ.

- `data/linear.data.r`

Example1: linear regression

As a prior, we use:

- $\beta \sim \mathrm{Normal}(2.0, 1.0)$
- $\sigma^2 \sim \mathrm{Gamma}(1.0, 1.0)$

which encode information from previously observed data.

Writing the Stan file

We need a statement that specifies the log joint distribution. Recall:

$$p(\theta, z) = p(z|\theta)p(\theta)$$

Then:

$$\log p(\theta, z) = \log p(z|\theta) + \log p(\theta)$$

Writing the Stan file

Stan retains certain C++ features:

- Variables need to be declared.
- Each statement must end with a semi-colon.

For example:

```
real x;
```

Writing the Stan file

A Stan program is divided into coding blocks:

- data
- parameter
- model

Writing the Stan file

```
data {
 Declare the data that will be given as an input.
}

parameters {
 Declare the parameters we want to sample.
}

model {
 Compute the log joint distribution.
}
```

Writing the Stan file

```
model {
  target += normal_lpdf(y | beta * x, sigma);

  // or equivalently

  y ~ normal(beta * x, sigma);
}
```

Writing the Stan file

Live demo.

Diagnose

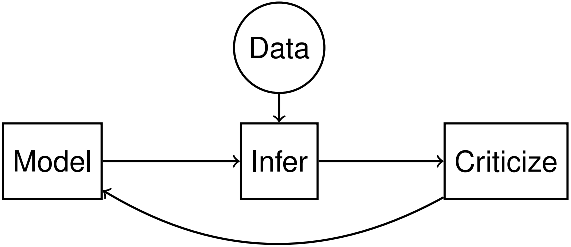
Are all (4) Markov chains exploring the target posterior?
Look at:

- the trace plots and the density plots
- the statistic $\hat{R}$.

The pairs plot can help us diagnose more difficult problems.

Posterior predictive checks

- Recall Box's loop.
- Does our model accurately describe the data?



Posterior predictive checks (ppc)

Given our posterior distribution for θ, what kind of data, y_{pred}, do we generate?

Proposition:
Each time we draw a sample, $\theta^{(i)} = (\beta^{(i)}, \sigma^{(i)})$*, we will also simulate data, according to*:

$$y_{\text{pred}} \sim \text{Normal}\left(x\beta^{(i)}, \sigma^{(i)}\right)$$

Posterior predictive checks

To do this, we will use the `generated quantities` block.

Live demo.

Improving the model

- The ppc suggest our model can improve with an intercept parameter.
- *Exercise: repeat the above procedure, but this time add a parameter* β_0*.*

Remarks on the prior distribution

The prior distribution can encode:

- an existing posterior distribution
- theoretical information
- a regularization devise (see Lasso, Ridge, etc.)
- any quantitative assumption.

The prior should be thought of as *part of the model*.

Remarks on the prior distribution

Diagnostic tools:

- Just as we did posterior predictive checks, we can do *prior predictive checks*.
- Do our priors allow for all “reasonable” data configurations?
- Do they allow for “unreasonable” data configurations?

Remarks on the prior distribution

Some helpful resources:

- https://github.com/stan-dev/stan/wiki/Prior-Choice-Recommendations
- Visualization in Bayesian workflow [Gabry et al., 2017]
- Towards a principled Bayesian workflow [Betancourt, 2018]

General resources to use Stan

- The Stan user manual
- The (draft) Stan book (https://mc-stan.org/docs/2_18/stan-users-guide/index.html)
- The Stan forum (http://discourse.mc-stan.org/)

Example 2: logistic regression

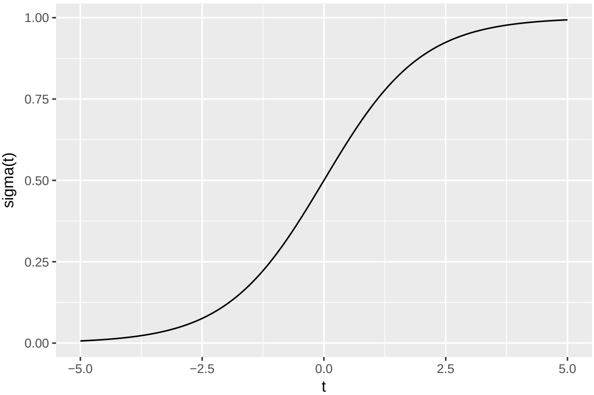
$Y \in \{0, 1\}$

$Y \sim \mathrm{Bernouilli}\left(\sigma(x\beta)\right)$

where

$$\sigma(t) = \frac{1}{1+e^{-t}} = \frac{e^t}{e^t+1}$$

Example 2: logistic regression



$$\sigma : \mathbb{R} \to (0, 1)$$

Example 2: logistic regression

Claim:

$$X\beta = \log\left(\frac{p}{1-p}\right)$$

The proof is left as an exercise.

Example 2: logistic regression

Some context for the data:

- y: outcome of a soccer game.
- x_1: goal difference between the home and the away team based on previous games *this season*.
- x_2: goal difference between the home and the away team based on previous games *last season*.

Feel free to ask me more questions about the data.

Example 2: logistic regression

Exercise: write and fit a logistic regression.

- Use `data/logistic.data.r`
- First use only one covariate, namely x_1.
- Use `y ~ bernoulli_logit(beta * x1)`.

Example 2: logistic regression

- How should we do ppc's here?

Example 2: logistic regression

- Can use misclassification rate.

```
generated quantities {
  int y_pred[N];
  int sum_err = 0;

  for (i in 1:N)
    y_pred[i] = bernoulli_logit_rng(beta * x1[i]);
  for (i in 1:N) sum_err += (y[i] != y_pred[i]);
}
```

- Can also compute the misclassification rate on a validation set.

Example 2: logistic regression

```
$summary
              mean      se_mean        sd        2.5%         25%         50%
beta     0.5393828 0.003007621 0.1044100   0.3559608   0.4661776   0.5319302
lp__   -39.3343232 0.017393778 0.6806572 -41.2310970 -39.5035019 -39.0706867
sum_err  24.6530000 0.075131633 3.9385237  17.0000000  22.0000000  24.5000000
               75%       97.5%    n_eff      Rhat
beta     0.6055403   0.7611313 1205.140 1.005007
lp__   -38.8904949 -38.8440736 1531.331 1.000010
sum_err  27.0000000  33.0000000 2748.029 1.001644
```

- The 95th quantile interval for `sum_err` is [17, 33]
- β is positive, which is what we would expect.

Example 2: logistic regression

Exercise: Augment the model by adding x_2 *as a covariate. Has the model improved?*

Example 2: logistic regression

```
$summary
              mean     se_mean        sd      2.5%        25%        50%
beta[1] -0.4567048 0.008312798 0.3356420 -1.202829 -0.6677736 -0.4243604
beta[2]  1.6653154 0.015719100 0.5613639  0.729895  1.2450485  1.6352881
lp__    -5.5497786 0.026844508 0.9695979 -8.204500 -5.9291885 -5.2451473
sum_err  2.6777500 0.024126722 1.3432632  1.000000  2.0000000  2.0000000
               75%      97.5%    n_eff      Rhat
beta[1] -0.2165342  0.1250064 1630.265 1.0008829
beta[2]  2.0328909  2.8516718 1275.362 1.0027203
lp__    -4.8622523 -4.5788527 1304.585 1.0013099
sum_err  3.0000000  6.0000000 3099.742 0.9996283
```

- The 95th quantile interval for `sum_err` is [1, 6]
- β_1 is negative, which is surprising.
- β_2 is positive, which is what we expect.

Example 2: logistic regression

Whether the model is “working” depends on our utility function.

- Do we care about predictions?
- Do we care about inference?

Office hour

Use the remaining time for questions.

References I

[Betancourt, 2018] Betancourt, M. (2018).
Towards a principled bayesian workflow.

[Gabry et al., 2017] Gabry, J., Simpson, D., Vehtari, A., Betancourt, M., and Gelman, A. (2017).
Visualization in bayesian workflow.
Royal Journal of Statistics, section A, 182:1 –14.