The Stan Modeling Language
Stan

Hamiltonian Monte Carlo

Modeling Language
Automatic Differentiation
Adaptation
A Stan model is defined by five program blocks

data

transformed data

parameters (required)

transformed parameters

model (required)

generated quantities
The “data” block reads external information

data {
    int N;
    int x[N];
    int offset;
}

The “transformed data” block allows for preprocessing of the data

```c
transformed data {
  int y[N];
  for (n in 1:N)
    y[n] <- x[n] - offset;
}
```
The “parameters” block defines the sampling space

parameters {
  real<lower=0> lambda1;
  real<lower=0> lambda2;
}

The “transformed parameters” block allows for parameter processing before the posterior is computed.

```plaintext
transformed parameters {
    real<lower=0> lambda;
    lambda <- lambda1 + lambda2;
}
```
In the “model” block we get to define our posterior

```r
model {
  y ~ poisson(lambda);
  lambda1 ~ cauchy(0, 2.5);
  lambda2 ~ cauchy(0, 2.5);
}
```
Lastly, the “generated quantities” block allows for postprocessing

generated quantities {
  int x_predict;
  x_predict <- poisson_rng(lambda) + offset;
}

data {
  int N;
  int x[N];
  int offset;
}

transformed data {
  int y[N];
  for (n in 1:N)
    y[n] <- x[n] - offset;
}

parameters {
  real<lower=0> lambda1;
  real<lower=0> lambda2;
}

transformed parameters {
  real<lower=0> lambda;
  lambda <- lambda1 + lambda2;
}

model {
  y ~ poisson(lambda);
  lambda1 ~ cauchy(0, 2.5);
  lambda2 ~ cauchy(0, 2.5);
}

generated quantities {
  int x_predict
  x_predict <- poisson_rng(lambda) + offset;
}
<table>
<thead>
<tr>
<th></th>
<th>data</th>
<th>transformed data</th>
<th>parameters</th>
<th>transformed parameters</th>
<th>model</th>
<th>generated quantities</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Execution</strong></td>
<td>Per chain</td>
<td>Per chain</td>
<td>NA</td>
<td>Per leapfrog</td>
<td>Per leapfrog</td>
<td>Per sample</td>
</tr>
<tr>
<td><strong>Variable Declarations</strong></td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Variable Scope</strong></td>
<td>Global</td>
<td>Global</td>
<td>Global</td>
<td>Global</td>
<td>Local</td>
<td>Local</td>
</tr>
<tr>
<td><strong>Variables Saved?</strong></td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Modify Posterior?</strong></td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td><strong>Random Variables</strong></td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Stan has two primitive types

int is an integer type

real is a floating point precision type
Both can be bounded

```c
int<lower=1> N;
real<upper=5> alpha;
real<lower=-1,upper=1> beta;
real gamma;
real<upper=gamma> zeta;
```
Reals extend to linear algebra types

vector[10] a; // Column vector
matrix[10, 1] b;

row_vector[10] c; // Row vector
matrix[1, 10] d;
Arrays of int, reals, vectors, and matrices are available

```c
real a[10];
vector[10] b[10];
matrix[10, 10] c[10];
```
Stan also implements a variety of constrained types

```plaintext
simplex[5] theta;       // sum(theta) = 1
positive_ordered[5] p;

corr_matrix[5] C;       // Symmetric and
cov_matrix[5] Sigma;    // positive-definite
```
All of your favorite statements are available, too

if/then/else

for (i in 1:I)

while (i < I)
There are two ways to modify the posterior

\begin{verbatim}
  y ~ normal(0, 1);
  increment_log_posterior(log_normal(y, 0, 1));
\end{verbatim}
Many sampling statements are *vectorized*

```stan
parameters {
  real mu[N];
  real<lower=0> sigma[N];
}

model {

  for (n in 1:N)
    y[n] ~ normal(mu[n], sigma[n]);
}
```
parameters {
    real mu[N];
    real<lower=0> sigma[N];
}

model {
    for (n in 1:N)
        y[n] ~ normal(mu[n], sigma[n]);

    y ~ normal(mu, sigma);
}
Because of the huge number of possible configurations, Stan uses hierarchical arguments.
Because of the huge number of possible configurations, Stan uses hierarchical arguments

\[
\begin{align*}
\text{id, data, init} \\
\text{random} \\
\text{seed} \\
\text{output} \\
\text{file, append.sample, diagnostic.file, append.diagnostic, ...} \\
\text{method} \\
\text{diagnose} \\
\text{...} \\
\text{optimize} \\
\text{...} \\
\text{sample} \\
\text{num.samples, num.warmup, save.warmup, thin} \\
\text{adapt} \\
\text{...} \\
\text{algorithm} \\
\text{hmc} \\
\text{...} \\
\text{rw.metropolis} \\
\text{...}
\end{align*}
\]
Groups

Because the Stan developers and users are distributed geographically, almost all of the discussions of Stan takes place on our group mailing lists. We prefer to communicate to a wide audience via the users group than to individuals via e-mail.

There is also an issue tracker which can be used to report code bugs or documentation typos and to request features.

Users Group

The users group is for general discussion of Stan, including modeling and installation issues:

- Stan Users Group (on Google Groups)

Everyone who joins the users group has posting privileges.

Developers Group

The developers group is for the development team to discuss Stan's code:

- Stan Developers Group (on Google Groups)

The developers group is open for everyone to read, but posting is restricted to Stan developers; see the contribution page if you want to contribute code.

Announcement Group

This group is only used to announce new releases:

- Stan Announcement Group (on Google Groups)

Buildbot Group

Stan follows a continuous integration process. To receive mail about integration test failures, sign up for:

- Stan Buildbot Group (on Google Groups)
Younger Americans are more likely to support the statewide legalization of gay marriage.
The GP models the data well

http://www.mc-stan.org/mlss14/support.data.R
And the posterior predictive checks indicate consistency
But what about the covariates we’ve ignored?

http://www.mc-stan.org/mlss14/support.data.R
Support For Gay Marriage (%)

Age

Female
Male

http://www.mc-stan.org/mlss14/support.data.R
Support For Gay Marriage ($\tilde{k}/n$)

Age

Female
Male

http://www.mc-stan.org/mlss14/support.data.R