Advanced Stan: 1

Skills recap

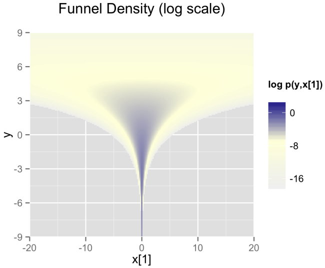
- RStan usage
- Generating fake data
- Writing models
- Debugging

Difficulties in fitting

- Adjust No-U-Turn Sampler parameters
    - target Metropolis acceptance rate
    - stepsize
- Reparameterization

Neal's Funnel

- $y \in \mathbb{R}$ and $x \in \mathbb{R}^9$
- $p(y, x) \;=\; \text{Normal}(y|0, 3) \times \prod_{n=1}^{9} \text{Normal}(x_n|0, \exp(y/2))$



Neal’s Funnel

```
// funnel_cp.stan
parameters {
  real y;
  vector[9] x;
}
model {
  y ~ normal(0, 3);
  x ~ normal(0, exp(y/2));
}
```

- Fit this model

- Why is this difficult?

Neal’s Funnel

```
funnel_cp1 <- stan("funnel_cp.stan")
funnel_cp1
## R-hat, n_eff

traceplot(funnel_cp1, "y")
## any stuck chains?
count_divergences(funnel_cp1)
## divergences after warmup!

hist_treedepth(funnel_cp1)

y_cp1 <- extract(funnel_cp1)$y
x1_cp1 <- extract(funnel_cp1)$x[,1]
plot(x1_cp1, y_cp1,
   main="Funnel Centered Parameterization")
```

HMC parameters (Ch. 57)

parameter	*description*	*constraint*	*default*
δ	target Metropolis acceptance rate	$\delta \in [0,1]$	0.80
γ	adaptation regularization scale	$\gamma > 0$	0.05
κ	adaptation relaxation exponent	$\kappa > 0$	0.75
t_0	adaptation iteration offset	$t_0 > 0$	10

Typical adjustments

- Increase `delta` (default 0.8)
- Decrease `stepsize` (default 2.0)
- Might have to increase `max_treedepth`

Adjust delta

```
funnel_cp2 <- stan(fit = funnel_cp,
                   control = list(adapt_delta = 0.99))
funnel_cp2

traceplot(funnel_cp2, "y")
## better?
count_divergences(funnel_cp2)
## compare to funnel_cp1
hist_treedepth(funnel_cp2)
## differences?

y_cp2 <- extract(funnel_cp2)$y
x1_cp2 <- extract(funnel_cp2)$x[,1]

plot(x1_cp2, y_cp2,
     main="Funnel Centered Parameterization 2")
```

Adjust stepsize

```
funnel_cp3 <- stan(fit = funnel_cp,
                   control = list(stepsize = 0.5))
funnel_cp3

traceplot(funnel_cp3, "y")
## better?
count_divergences(funnel_cp3)
## compare to funnel_cp1
hist_treedepth(funnel_cp3)
## differences between the three runs?

y_cp3 <- extract(funnel_cp3)$y
x1_cp3 <- extract(funnel_cp3)$x[,1]

plot(x1_cp3, y_cp3,
     main="Funnel Centered Parameterization 3")
```

Non-centered parameterization

```
parameters {           // funnel.stan
  real y_raw;
  vector[9] x_raw;
}
transformed parameters {
  real y;
  vector[9] x;

  y <- 3.0 * y_raw;
  x <- exp(y/2) * x_raw;
}
model {
  y_raw ~ normal(0, 1); // implies y ~ normal(0, 3)
  x_raw ~ normal(0, 1); // implies x ~ normal(0, exp(y/2))
}
```

Non-centered parameterization

```
funnel
## R-hat? n_eff?

traceplot(funnel, "y")
## How does it look?
count_divergences(funnel)

y <- extract(funnel)$y
x1 <- extract(funnel)$x[,1]

plot(x1, y, main="Funnel")
```

Non-centered parameterization

1. Add a new parameter

2. Add a sampling statement to the model for the parameter

3. Move existing parameter to transformed parameters

4. Update assignment of existing parameter

Hierarchical Voting Example

- Data: `vote.data.R`

- Centered parameterization: `vote_cp.stan`

Hierarchical Voting Example

```
data <- read_rdump("vote.data.R")
vote_cp <- stan("vote_cp.stan", data = data)

vote_cp
print(vote_cp, c("alpha_edu"))
print(vote_cp, c("sigma_sigma_alpha", "sigma_alpha"))

traceplot(vote_cp, "sigma_alpha")

count_divergences(vote_cp)
```

Change to non-centered parameterization

- One level of ncp.
- Two levels of ncp.