DAY 1: RSTAN, STATISTICAL MODELS

STAN WORKSHOP

POISSON DATA

POISSON.DATA

- R:
  `source(“poisson.data.R”)`

- Looks like:

```
N <- 1000
x <- c(4,7,2,1,0,1,5,0,2,0,5,6,7,9, …
```

STEP 1: LOOK AT DATA; BUILD POISSON MODEL

- Hint:

```
data {
  int N;
  int<lower=0> x[N];
}
```

$$x_n \sim \mathrm{poisson}(\lambda)$$

or equivalently

$$p(\lambda, x) = \prod_{n=1}^{N} \frac{1}{x_n!} \lambda^{x_n} \exp(-\lambda)$$

$$= \exp\left(\sum_{n=1}^{N} \mathrm{poisson\_log}(x_n, \lambda) \right)$$

- Also write a posterior predictive check:

```
generated quantities {
   int x_ppc;
   x_ppc <- poisson_rng(lambda);
}
```

- Try to do this using **increment_log_prob()** directly (you'll see why later)

STEP 1: LOOK AT DATA; BUILD POISSON MODEL

```
data {
  int N;
  real x[N];
}
parameters {
  real<lower = 0> lambda;
}
model {
  lambda ~ cauchy(0, 1);
  x ~ poisson(lambda);
}
generated quantities {
  real x_ppc;
  x_ppc <- poisson_rng(lambda);
}
```

STEP 1: LOOK AT DATA; BUILD POISSON MODEL

```
data {
  int N;
  real x[N];
}
parameters {
  real<lower = 0> lambda;
}
model {
  increment_log_prob(cauchy_log(lambda, 0, 1));
  for (n in 1:N)
     increment_log_prob(poisson_log(x[n], lambda));
}
generated quantities {
  real x_ppc;
  x_ppc <- poisson_rng(lambda);
}
```

STEP 1: LOOK AT DATA; BUILD POISSON MODEL

- Does the fit look right?
- Compare the x_ppc to x
- What needs to be done?

STEP 2: ZERO-INFLATION

- Add zero inflation.
  If x[n] == 0, then it either came from the Poisson likelihood or it was 0 with probability theta.
- This is a mixture model; also example of marginalizing discrete parameter

$$p(x_n = 0 \,|\, \lambda, \theta) = \theta \times 1 + (1 - \theta) \times \text{poisson}(x_n | \lambda)$$

$$p(x_n \neq 0 \,|\, \lambda, \theta) = \,?$$

- Make sure to do the posterior predictive check
- Hint: use the log_mix function.

STEP 2: ZERO-INFLATION

- Add zero inflation.
  If x[n] == 0, then it either came from the Poisson likelihood or it was 0 with probability theta.
- This is a mixture model; also example of marginalizing discrete parameter

$$p(x_n = 0 \,|\, \lambda, \theta) = \theta \times 1 + (1 - \theta) \times \mathrm{poisson}(x_n | \lambda)$$

$$p(x_n \neq 0 \,|\, \lambda, \theta) = \theta \times 0 + (1 - \theta) \times \mathrm{poisson}(x_n , \lambda)$$

- Make sure to do the posterior predictive check
- Hint: use the log_mix function.

STEP 2: ZERO-INFLATION

```
...
parameters {
  real<lower = 0> lambda;
  real<lower = 0, upper = 1> theta;
}
model {
  increment_log_prob(cauchy_log(lambda, 0, 1));
  for (n in 1:N)
    if (x[n] == 0)
      increment_log_prob(log_mix(theta, 0, poisson_log(x[n], lambda);
    else
      increment_log_prob(log1m(theta)
                         + poisson_log(x[n], lambda));
}
generated quantities {
  real x_ppc;
  if (bernoulli_rng(theta) == 1)
    x_ppc <- 0;
  else
    x_ppc <- poisson_rng(lambda);
}
```

STEP 2: ZERO-INFLATION

- Still wrong. What's going on?

STEP 3: TRUNCATION

- Add truncation to the model.
  For this exercise, let's fix the upper bound to 9.
- Now, the likelihood looks like:

$$p(x_n = 0 \,|\, \lambda, \theta) = \theta \times 1 + (1 - \theta) \times \frac{\text{poisson}(x_n|\lambda)}{\text{poisson\_cdf}(U|\lambda)}$$

$$p(x_n \neq 0 \,|\, \lambda, \theta) = (1 - \theta) \times \frac{\text{poisson}(x_n|\lambda)}{\text{poisson\_cdf}(U \,|\, \lambda)}$$

- Hint: use poisson_cdf_log is evaluated on the log scale

STEP 3: TRUNCATION

```
...
parameters {
  real<lower = 0> lambda;
  real<lower = 0, upper = 1> theta;
}
model {
  increment_log_prob(cauchy_log(lambda, 0, 1));
  for (n in 1:N) {
    real log_trunc_poisson;
    log_trunc_poisson <- poisson_log(x[n], lambda) - poisson_cdf_log(9, lambda);
    if (x[n] == 0)
      increment_log_prob(log_mix(theta, 0, log_trunc_poisson));
    else
      increment_log_prob(log1m(theta) + log_trunc_poisson);
  }
}
generated quantities {
  real x_ppc;
  if (bernoulli_rng(theta) == 1)
    x_ppc <- 0;
  else {
    x_ppc <- poisson_rng(lambda);
    while (x_ppc > 9)
      x_ppc <- poisson_rng(lambda);
  }
}
```

RECAP

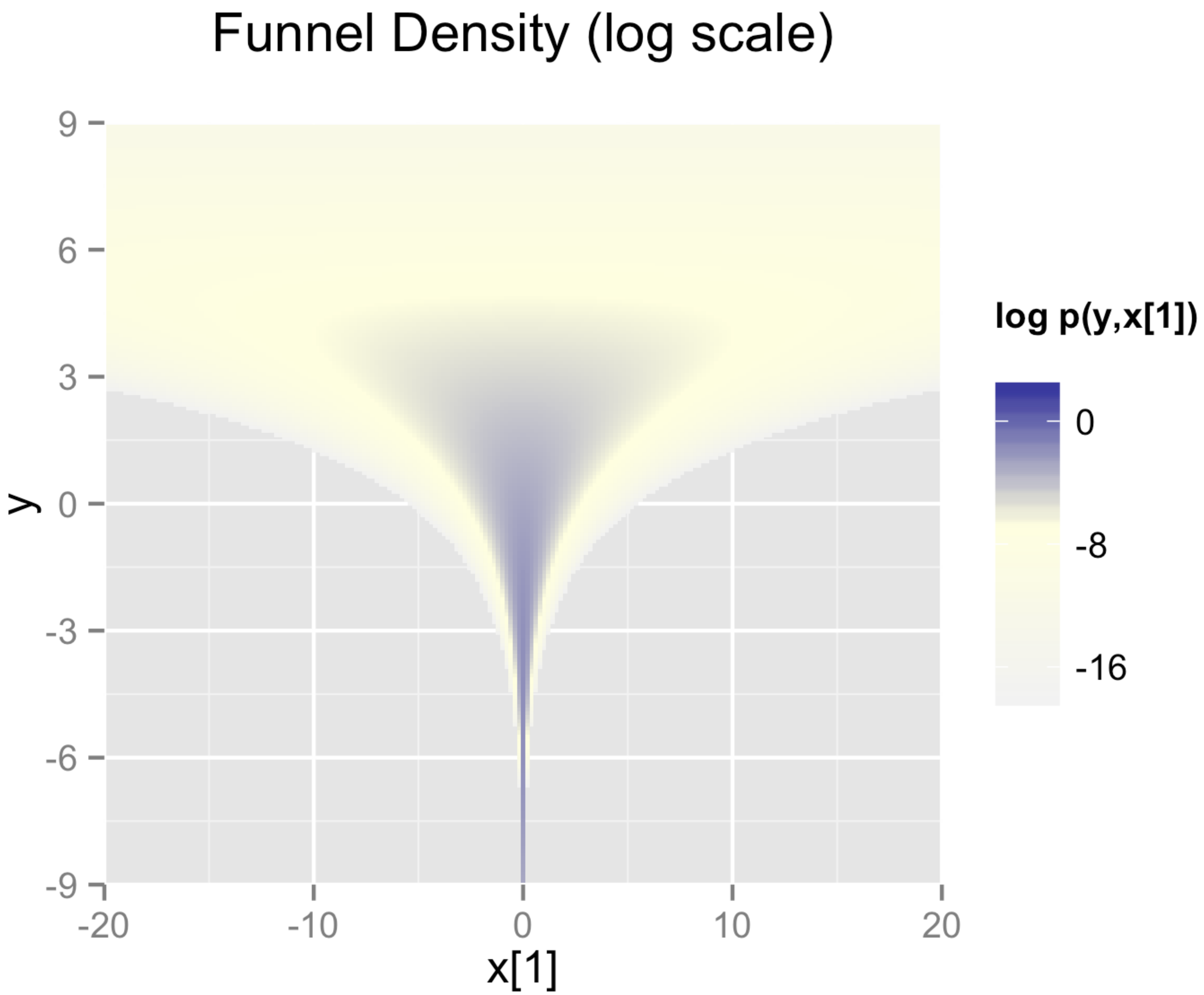
- Wrote model without analytic solution
- Zero-inflation is example of marginalizing out a discrete parameter
- Iterated over 3 different models
- increment_log_prob() can be used to write any log joint distribution function

NON-CENTERED REPARAMETERIZATION

FUNNEL

- $y \in \mathbb{R}$
- $x \in \mathbb{R}^9$

$$p(y, x) = \mathrm{Normal}(y|0, 3)$$
$$\times \prod_{n=1}^{9} \mathrm{Normal}(x_n|0, \exp(y/2))$$



WHEN DO YOU SEE THIS?

- Hierarchical models
- Variance parameters go to 0, all parameters shrink
  Variance parameters get large, all parameters spread
- Trick to handle low data situations
- Called non-centered parameterization aka the Matt trick …

STEPS

1. Add new parameter, *_raw.
2. Move original parameter to transformed parameters block.
3. Assign transformation of *_raw to original parameter.
4. Put Normal(0, 1) prior on *_raw.

CENTERED FUNNEL

- Easy to write in Stan
- Run. See any problems?

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

NON-CENTERED FUNNEL: STEP 1

- Add new parameter, *_raw.

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

NON-CENTERED FUNNEL: STEP 1

- Add new parameter, *_raw.

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

NON-CENTERED FUNNEL: STEP 2

- Move original parameter to transformed parameters block.

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

NON-CENTERED FUNNEL: STEP 2

- Move original parameter to transformed parameters block.

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

NON-CENTERED FUNNEL: STEP 3

- Assign transformation of *_raw to original parameter.

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

NON-CENTERED FUNNEL: STEP 3

- Assign transformation of *_raw to original parameter.

```
parameters {
  vector[9] x;
  real y_raw;
}
transformed parameters {
  real y;
  y <- 3 * y_raw;
}
model {
  y ~ normal(0, 3);
  x ~ normal(0, exp(y/2));
}
```

NON-CENTERED FUNNEL: STEP 4

- Put Normal(0, 1) prior on *_raw.

```
parameters {
  vector[9] x;
  real y_raw;
}
transformed parameters {
  real y;
  y <- 3 * y_raw;
}
model {
  y ~ normal(0, 3);
  x ~ normal(0, exp(y/2));
}
```

NON-CENTERED FUNNEL: STEP 4

- Put Normal(0, 1) prior on *_raw.

```
parameters {
  vector[9] x;
  real y_raw;
}
transformed parameters {
  real y;
  y <- 3 * y_raw;
}
model {
  y_raw ~ normal(0, 1);
  x ~ normal(0, exp(y/2));
}
```

NON-CENTERED FUNNEL

- Repeat for xs.
- Steps:
  1. Add new parameter, *_raw.
  2. Move original parameter to transformed parameters block.
  3. Assign transformation of *_raw to original parameter.
  4. Put Normal(0, 1) prior on *_raw.

NON-CENTERED FUNNEL

```
parameters {
  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);
  x_raw ~ normal(0, 1);
}
```

CENTERED VS NON-CENTERED

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

```
parameters {
  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);
  x_raw ~ normal(0, 1);
}
```