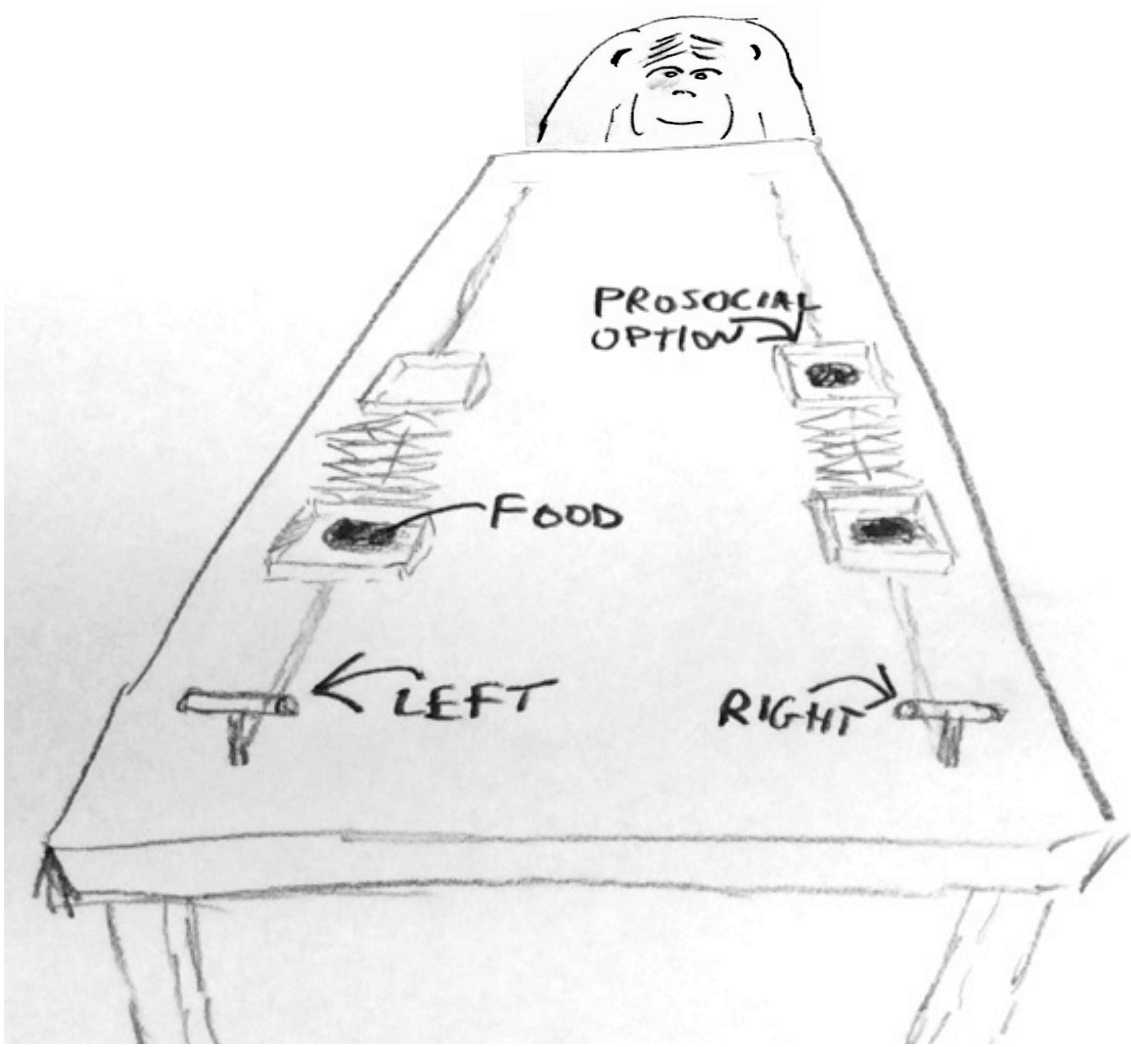
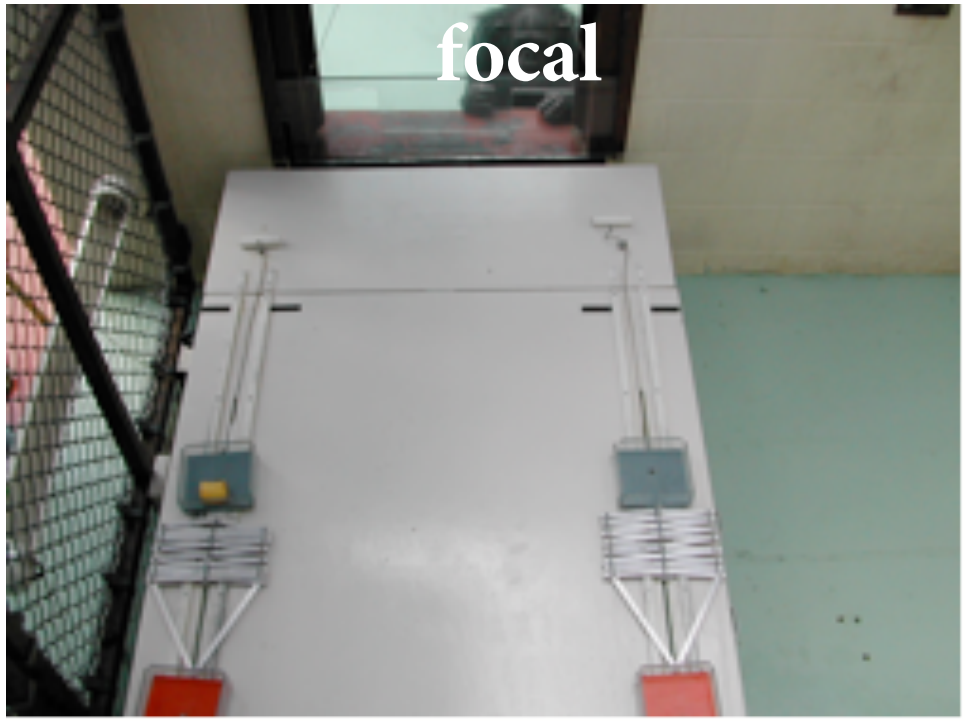
Statistical Rethinking
Winter 2019

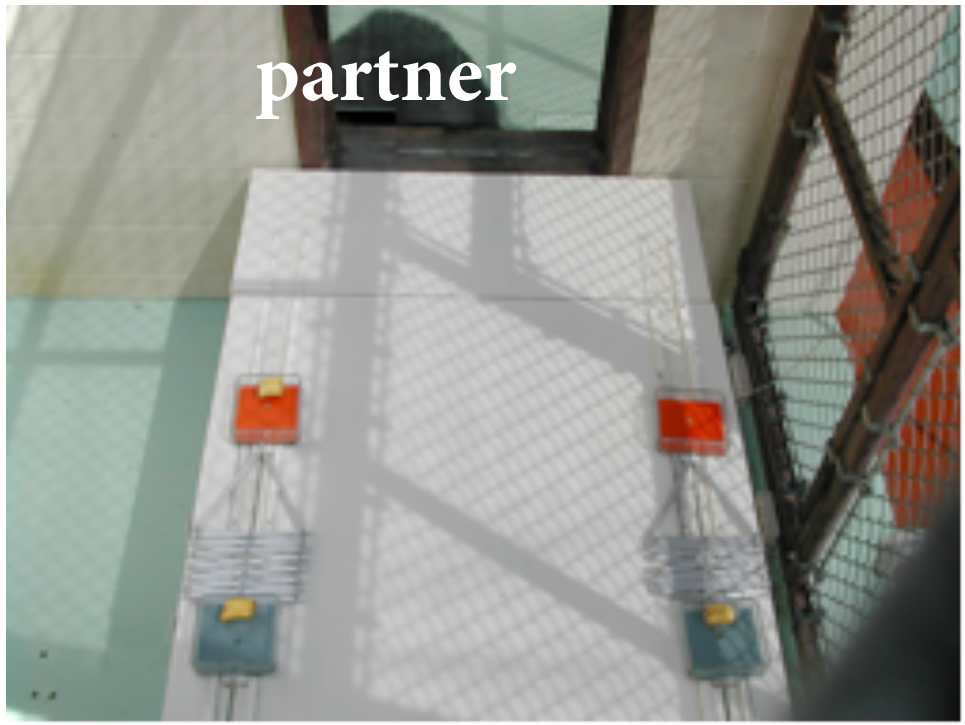
Lecture 16 / Week 8

Multilevel Models

Prosocial chimpanzees

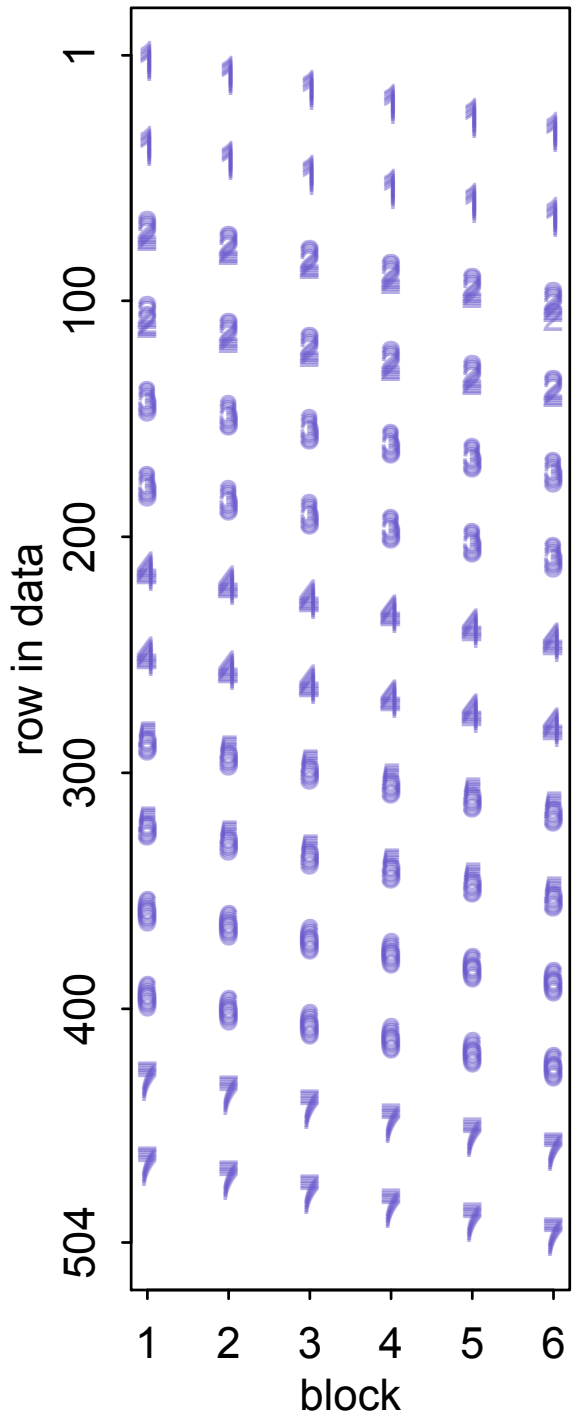






Cross-classification

- Can use more than one cluster type
- Chimpanzee experiment data
  - Pulls in chimpanzees
  - Pulls in blocks
  - Each chimp in each block
  - Not nested, but *cross-classified*



Multilevel chimpanzees

$$
\begin{aligned}
L_i &\sim \text{Binomial}(1, p_i) \\
\text{logit}(p_i) &= \alpha_{\text{ACTOR}[i]} + \gamma_{\text{BLOCK}[i]} + \beta_{\text{TREATMENT}[i]} \\
\beta_j &\sim \text{Normal}(0, 0.5) \quad \text{, for } j = 1..4 \\
\alpha_j &\sim \text{Normal}(\bar{\alpha}, \sigma_\alpha) \quad \text{, for } j = 1..7 \\
\gamma_j &\sim \text{Normal}(0, \sigma_\gamma) \quad \text{, for } j = 1..6 \\
\bar{\alpha} &\sim \text{Normal}(0, 1.5) \\
\sigma_\alpha &\sim \text{Exponential}(1) \\
\sigma_\gamma &\sim \text{Exponential}(1)
\end{aligned}
$$

varying intercepts on actor ⟶

Multilevel chimpanzees

$$
\begin{aligned}
L_i &\sim \text{Binomial}(1, p_i) \\
\text{logit}(p_i) &= \alpha_{\text{ACTOR}[i]} + \gamma_{\text{BLOCK}[i]} + \beta_{\text{TREATMENT}[i]} \\
\beta_j &\sim \text{Normal}(0, 0.5) \quad \text{, for } j = 1..4 \\
\alpha_j &\sim \text{Normal}(\bar{\alpha}, \sigma_\alpha) \quad \text{, for } j = 1..7 \\
\gamma_j &\sim \text{Normal}(0, \sigma_\gamma) \quad \text{, for } j = 1..6 \\
\bar{\alpha} &\sim \text{Normal}(0, 1.5) \\
\sigma_\alpha &\sim \text{Exponential}(1) \\
\sigma_\gamma &\sim \text{Exponential}(1)
\end{aligned}
$$

varying intercepts on actor ⟶ $\alpha_j \sim \text{Normal}(\bar{\alpha}, \sigma_\alpha)$, for $j = 1..7$

varying intercepts on block ⟶ $\gamma_j \sim \text{Normal}(0, \sigma_\gamma)$, for $j = 1..6$

Multilevel chimpanzees

```
m13.4 <- ulam(
    alist(
        pulled_left ~ dbinom( 1 , p ) ,
        logit(p) <- a[actor] + g[block_id] + b[treatment] ,
        b[treatment] ~ dnorm( 0 , 0.5 ),

        # adaptive priors
        a[actor] ~ dnorm( a_bar , sigma_a ),
        g[block_id] ~ dnorm( 0 , sigma_g ),

        # hyper-priors
        a_bar ~ dnorm( 0 , 1.5 ),
        sigma_a ~ dexp(1),
        sigma_g ~ dexp(1)
    ) , data=dat_list , chains=4 , cores=4 , log_lik=TRUE )
```

Multilevel chimpanzees

```
m13.4 <- ulam(
    alist(
        pulled_left ~ dbinom( 1 , p ) ,
        logit(p) <- a[actor] + g[block_id] + b[treatment] ,
        b[treatment] ~ dnorm( 0 , 0.5 ),

        # adaptive priors
        a[actor] ~ dnorm( a_bar , sigma_a ),
        g[block_id] ~ dnorm( 0 , sigma_g ),

        # hyper-priors
        a_bar ~ dnorm( 0 , 1.5 ),
        sigma_a ~ dexp(1),
        sigma_g ~ dexp(1)
    ) , data=dat_list , chains=4 , cores=4 , log_lik=TRUE )
```

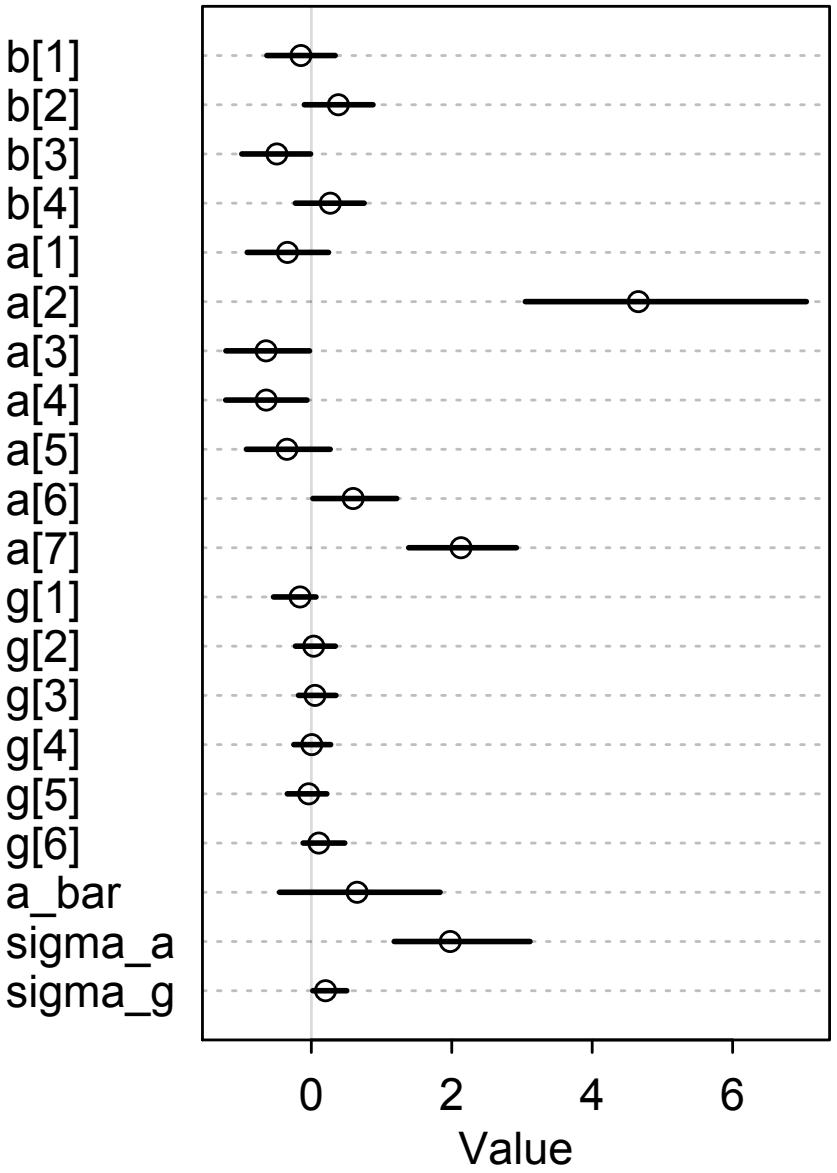
Multilevel chimpanzees

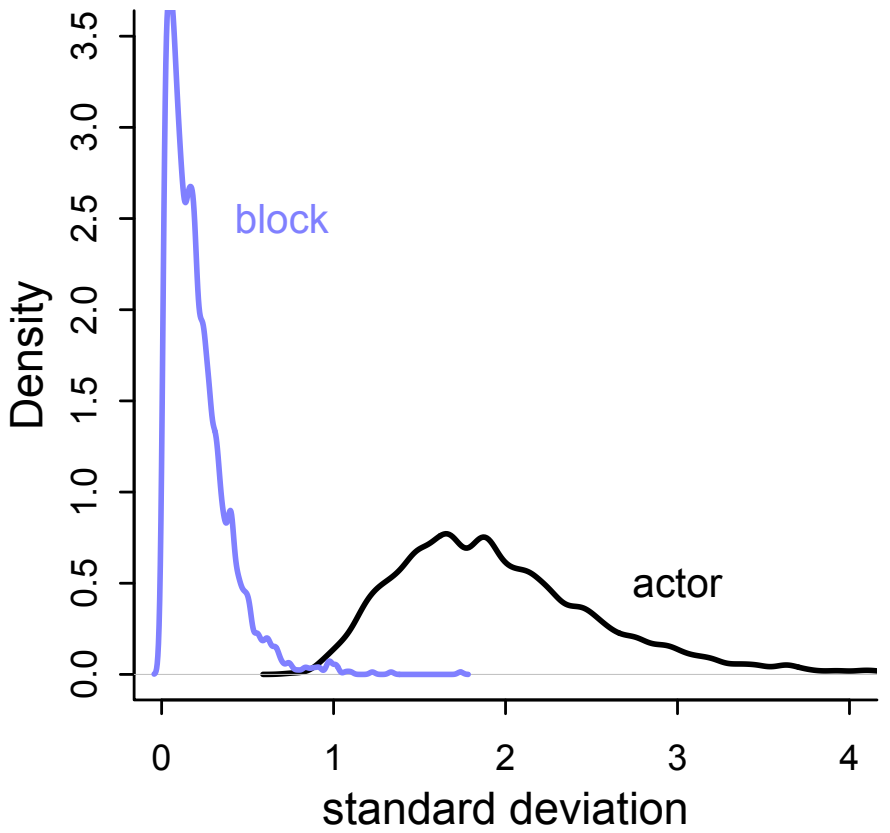
```
m13.4 <- ulam(
    alist(
        pulled_left ~ dbinom( 1 , p ) ,
        logit(p) <- a[actor] + g[block_id] + b[treatment] ,
        b[treatment] ~ dnorm( 0 , 0.5 ),

        # adaptive priors
        a[actor] ~ dnorm( a_bar , sigma_a ),
        g[block_id] ~ dnorm( 0 , sigma_g ),

        # hyper-priors
        a_bar ~ dnorm( 0 , 1.5 ),
        sigma_a ~ dexp(1),
        sigma_g ~ dexp(1)
    ) , data=dat_list , chains=4 , cores=4 , log_lik=TRUE )
```

Cross-classified chimpanzees





Cross-classified chimpanzees

- Incorporating block: no benefits; little cost

R code 13.23

```
set.seed(14)
m13.5 <- ulam(
    alist(
        pulled_left ~ dbinom( 1 , p ) ,
        logit(p) <- a[actor] + b[treatment] ,
        b[treatment] ~ dnorm( 0 , 0.5 ),
        a[actor] ~ dnorm( a_bar , sigma_a ),
        a_bar ~ dnorm( 0 , 1.5 ),
        sigma_a ~ dexp(1)
    ) , data=dat_list , chains=4 , cores=4 , log_lik=TRUE )
```

R code 13.24

```
compare( m13.4 , m13.5 )
```

```
       WAIC pWAIC dWAIC weight    SE  dSE
m13.5 531.0   8.5     0   0.73 19.23   NA
m13.4 532.9  10.9     2   0.27 19.39 1.64
```

Everything is random

- "Random effects" have many definitions
- One common & bad definition:
  - Random effects are not fixed by the experiment
- Does not matter—we want pooling for estimation benefits
- Consider e.g. treatments

R code 13.25

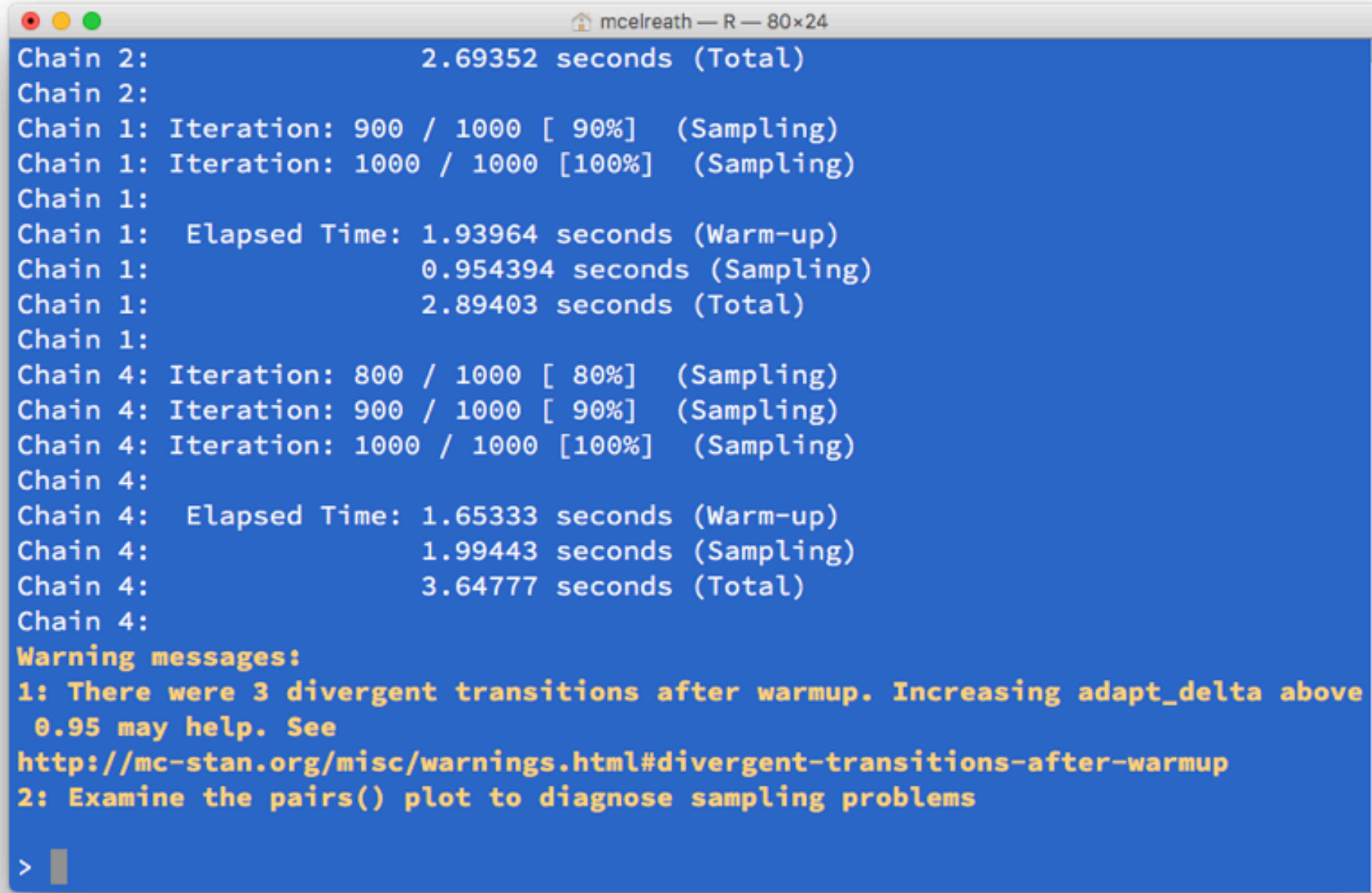
```
set.seed(15)
m13.6 <- ulam(
    alist(
        pulled_left ~ dbinom( 1 , p ) ,
        logit(p) <- a[actor] + g[block_id] + b[treatment] ,
        b[treatment] ~ dnorm( 0 , sigma_b ),
        a[actor] ~ dnorm( a_bar , sigma_a ),
        g[block_id] ~ dnorm( 0 , sigma_g ),
        a_bar ~ dnorm( 0 , 1.5 ),
        sigma_a ~ dexp(1),
        sigma_g ~ dexp(1),
        sigma_b ~ dexp(1)
```

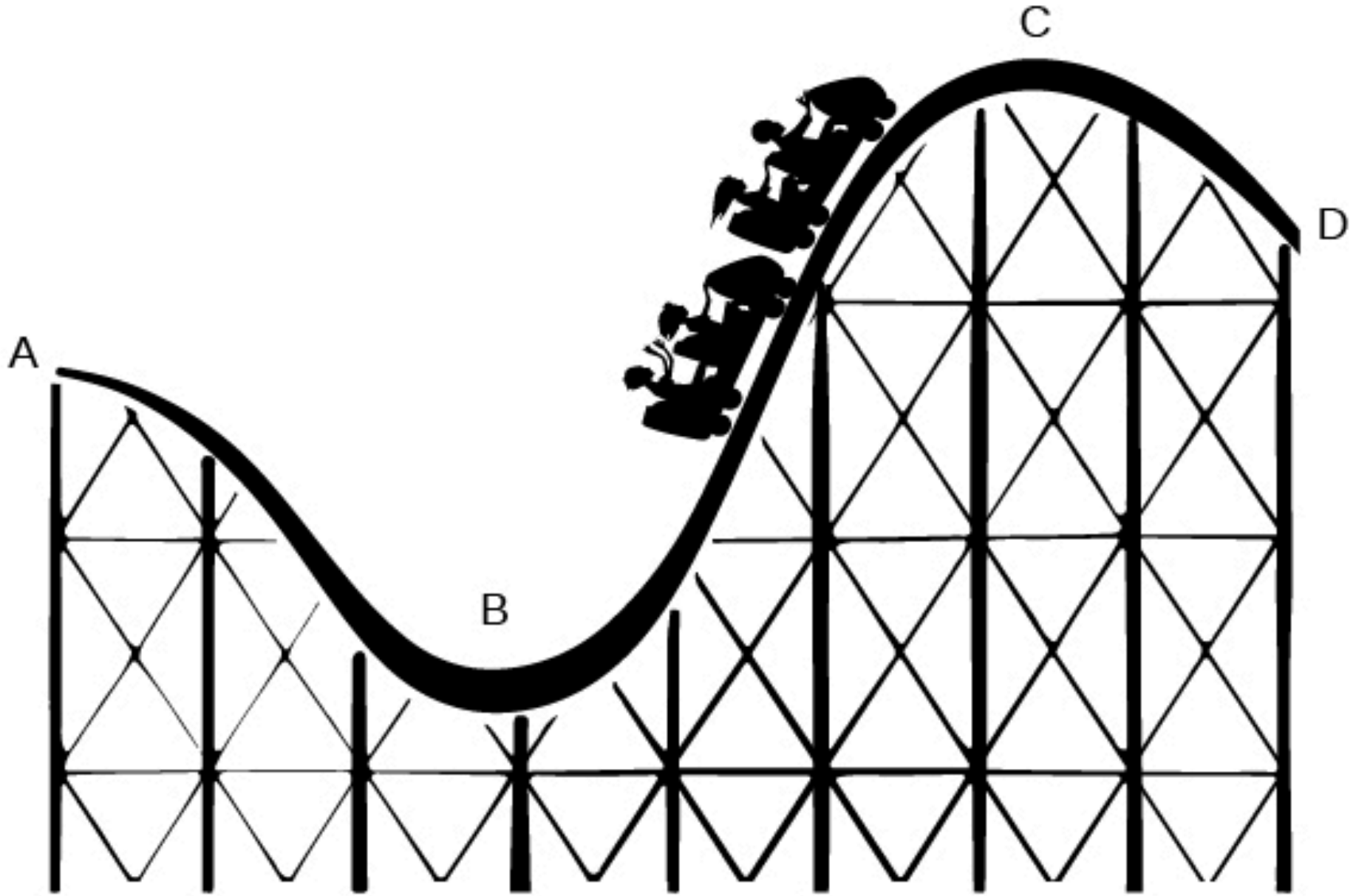
Everything is random

R code 13.25

```
set.seed(15)
m13.6 <- ulam(
    alist(
        pulled_left ~ dbinom( 1 , p ) ,
        logit(p) <- a[actor] + g[block_id] + b[treatment] ,
        b[treatment] ~ dnorm( 0 , sigma_b ),
        a[actor] ~ dnorm( a_bar , sigma_a ),
        g[block_id] ~ dnorm( 0 , sigma_g ),
        a_bar ~ dnorm( 0 , 1.5 ),
        sigma_a ~ dexp(1),
        sigma_g ~ dexp(1),
        sigma_b ~ dexp(1)
    ) , data=dat_list , chains=4 , cores=4 , log_lik=TRUE )
coeftab(m13.4,m13.6)
```

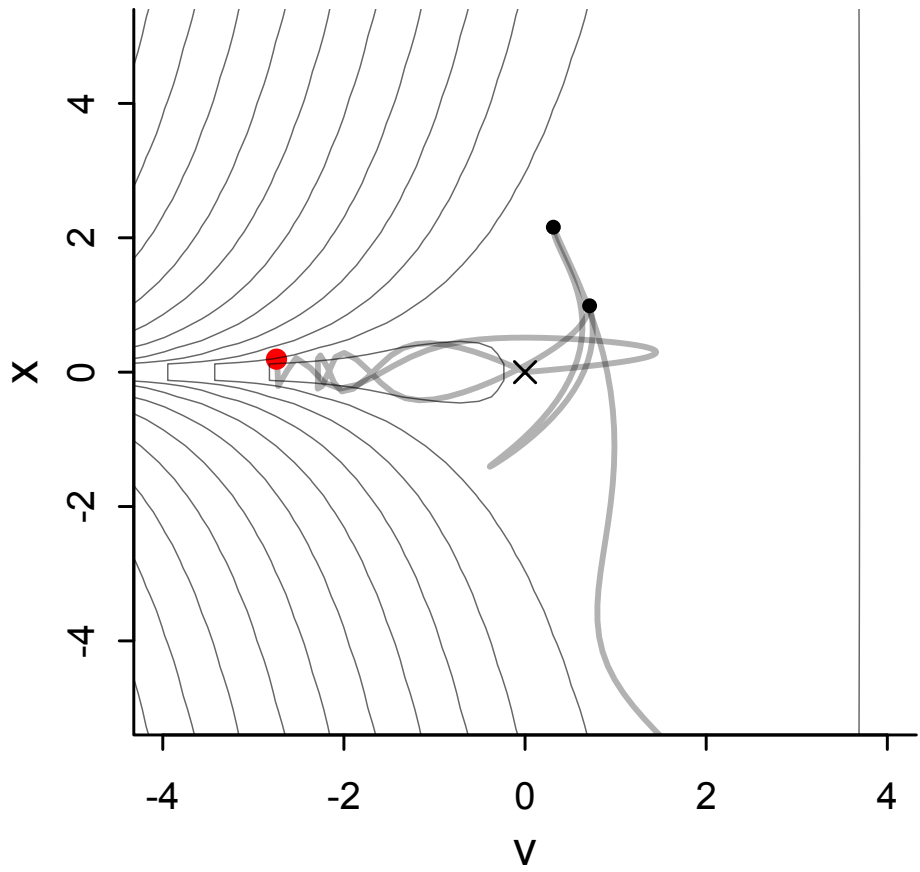
```
        m13.4   m13.6
b[1]      -0.11   -0.15
b[2]       0.41    0.34
b[3]      -0.46   -0.47
b[4]       0.31    0.25
```


mcelreath — R — 80×24
Chain 2:   2.69352 seconds (Total)
Chain 2:
Chain 1: Iteration: 900 / 1000 [90%] (Sampling)
Chain 1: Iteration: 1000 / 1000 [100%] (Sampling)
Chain 1:
Chain 1: Elapsed Time: 1.93964 seconds (Warm-up)
Chain 1:   0.954394 seconds (Sampling)
Chain 1:   2.89403 seconds (Total)
Chain 1:
Chain 4: Iteration: 800 / 1000 [80%] (Sampling)
Chain 4: Iteration: 900 / 1000 [90%] (Sampling)
Chain 4: Iteration: 1000 / 1000 [100%] (Sampling)
Chain 4:
Chain 4: Elapsed Time: 1.65333 seconds (Warm-up)
Chain 4:   1.99443 seconds (Sampling)
Chain 4:   3.64777 seconds (Total)
Chain 4:
Warning messages:
1: There were 3 divergent transitions after warmup. Increasing adapt_delta above
 0.95 may help. See
http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
2: Examine the pairs() plot to diagnose sampling problems
>


A
B
C
D

Divergent transitions

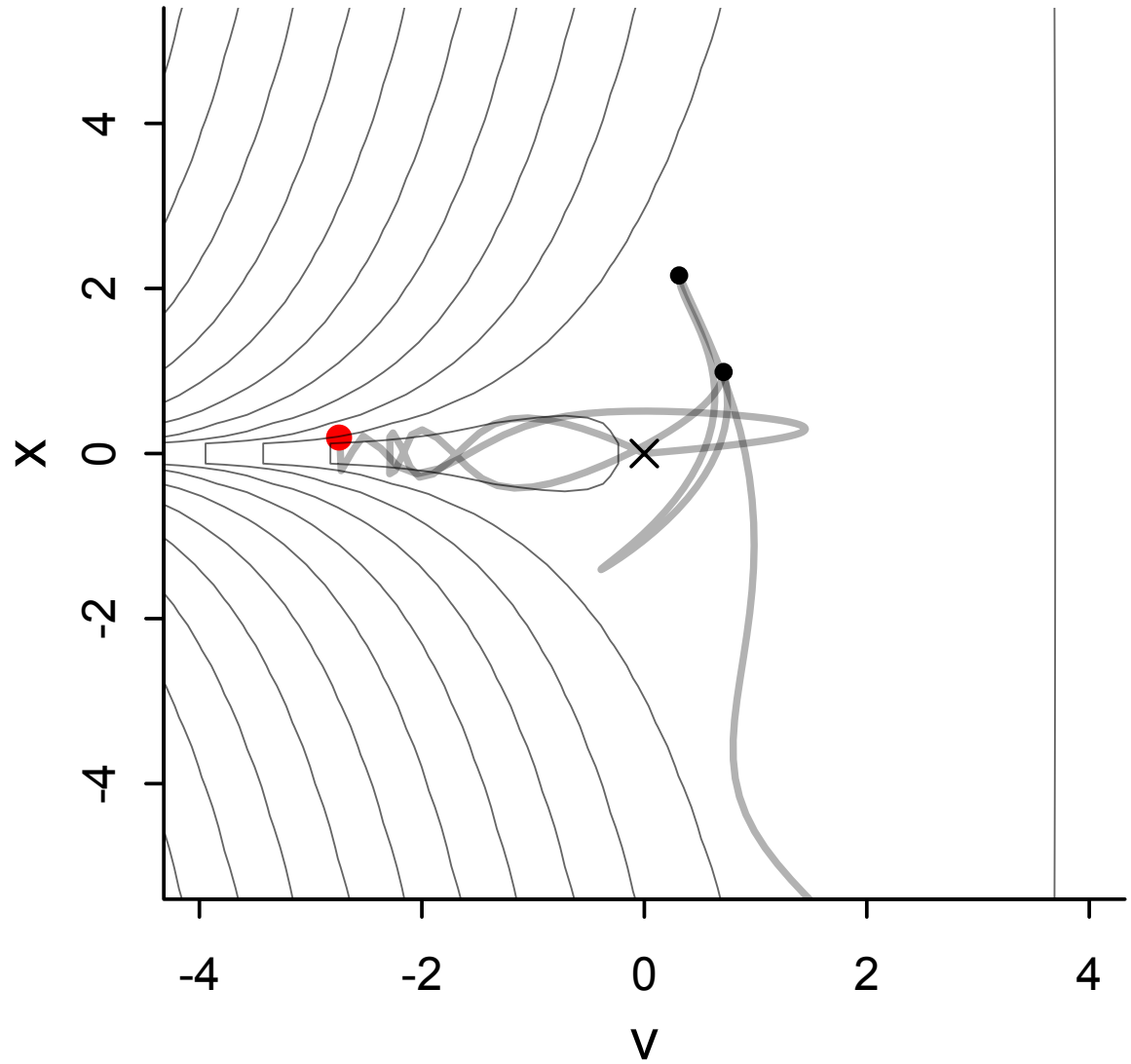
- HMC runs a physics simulation
- Each ***transition*** is a sample path
- In real physics, energy is conserved
- If energy at end of transition is not equal to energy at start, transition is ***divergent***
- Indicates inaccurate approximation
- Tends to happen in regions of strong curvature of log-posterior
- **Other sampling strategies also bad in these cases, but produce no warnings!**



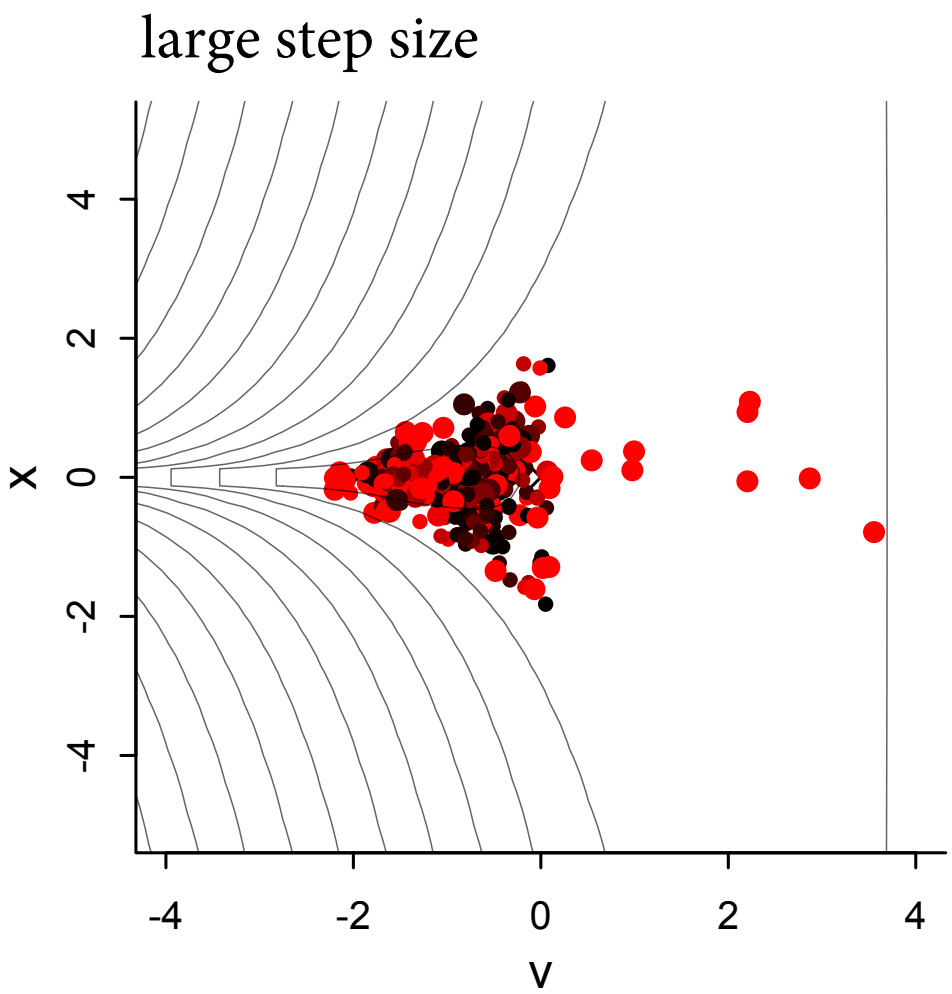
Divergent transitions

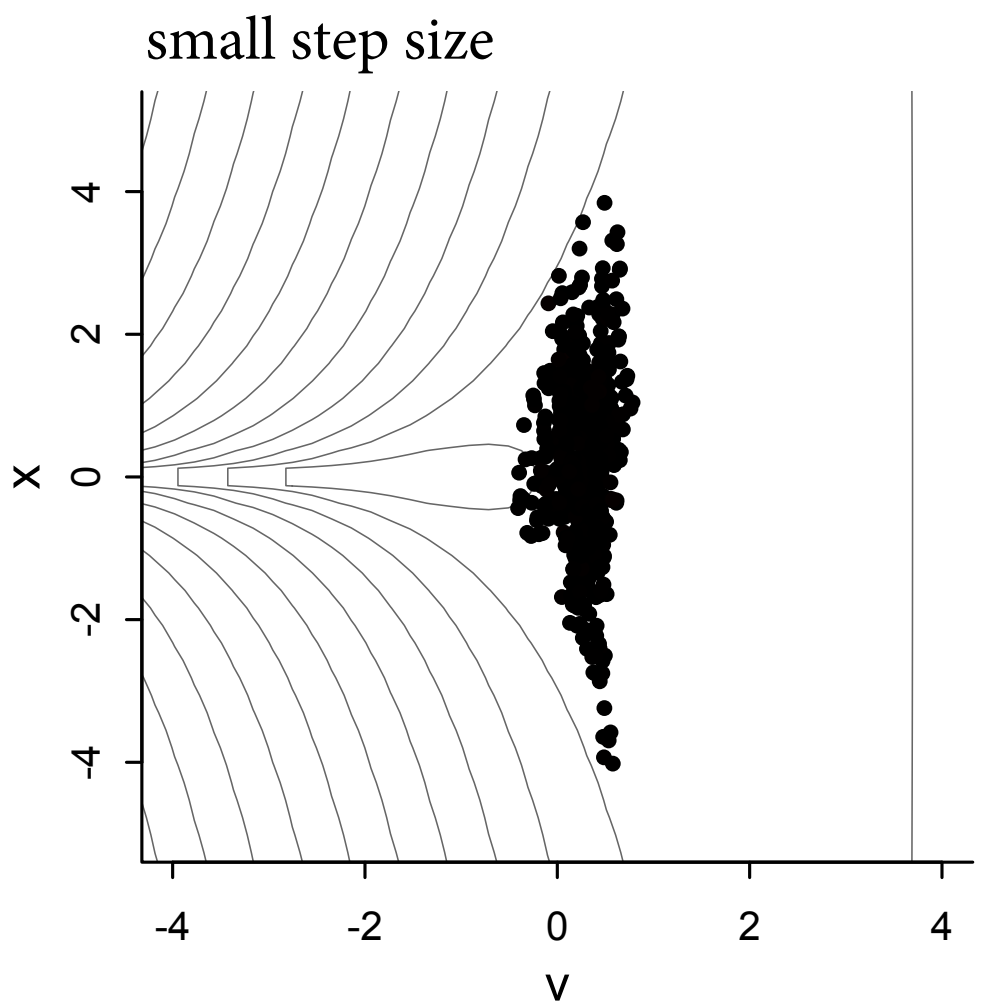
$$v \sim \text{Normal}(0, 3)$$

$$x \sim \text{Normal}(0, \exp(v))$$



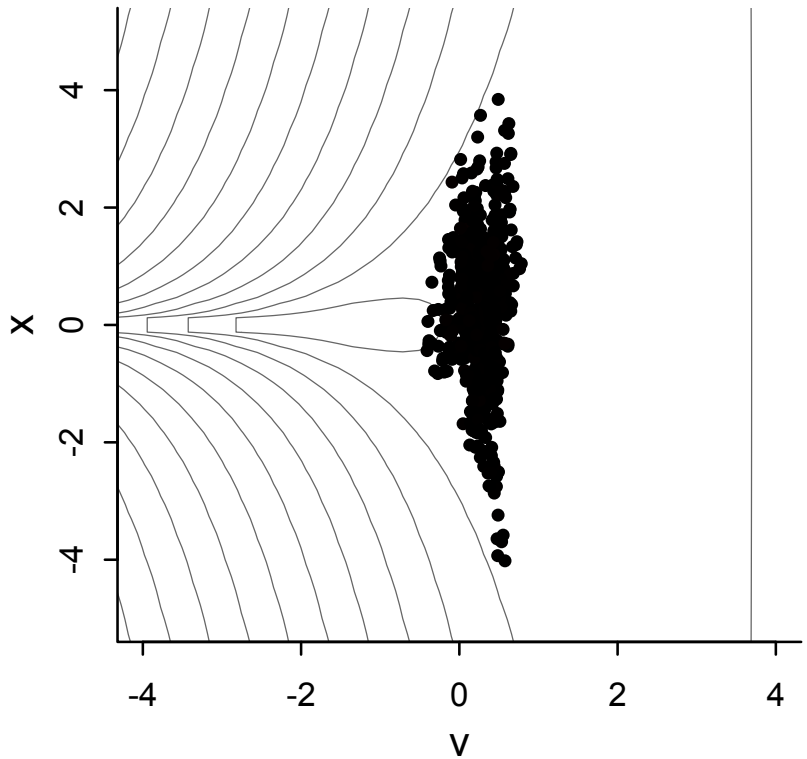
Divergent transitions





Divergent transitions

- Two basic strategies:
- (1) Increase Stan's adapt_delta control parameter => better step size adaptation + slower exploration
- (2) Re-parameterize!



Re-parameterize!

- Most any statistical model can be expressed in several mathematically identical ways

$$\alpha \sim \text{Normal}(\mu, \sigma)$$

Re-parameterize!

- Most any statistical model can be expressed in several mathematically identical ways

$$\alpha \sim \text{Normal}(\mu, \sigma)$$

$$\alpha = \mu + \beta$$
$$\beta \sim \text{Normal}(0, \sigma)$$

Re-parameterize!

- Most any statistical model can be expressed in several mathematically identical ways

$$\alpha \sim \text{Normal}(\mu, \sigma)$$ *Centered*

$$\alpha = \mu + \beta$$
$$\beta \sim \text{Normal}(0, \sigma)$$

$$\alpha = \mu + z\sigma$$
$$z \sim \text{Normal}(0, 1)$$ *Non-centered*

Re-parameterize!

- Why would this madness help with sampling?
- HMC sees a different geometry!

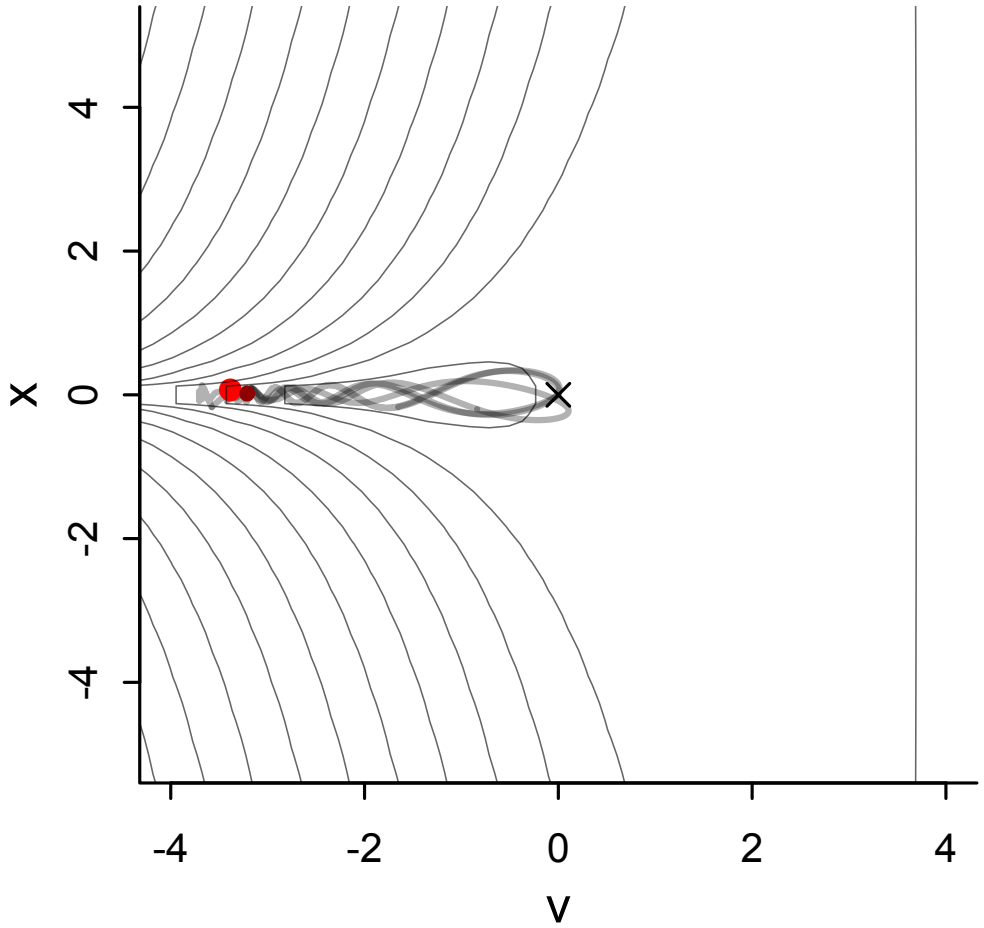
Centered

$$v \sim \text{Normal}(0, 3)$$
$$x \sim \text{Normal}(0, \exp(v))$$

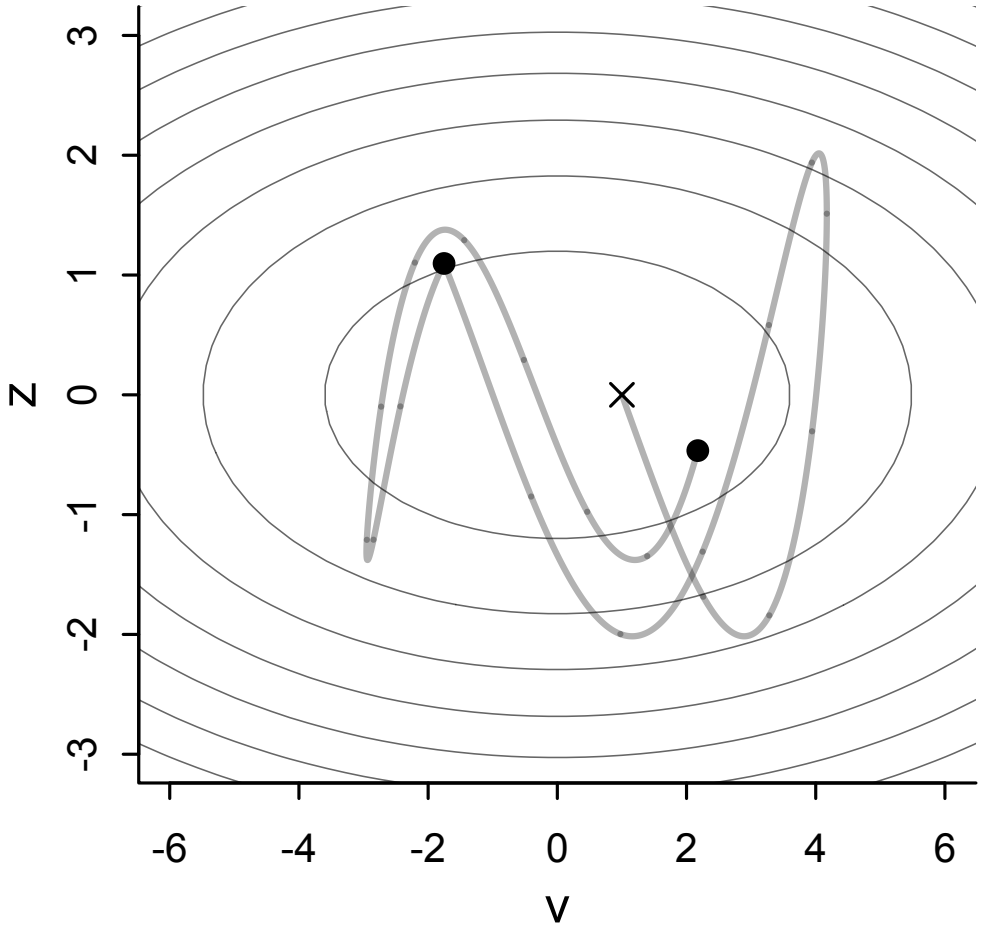
Non-centered

$$v \sim \text{Normal}(0, 3)$$
$$x = z \exp(v)$$
$$z \sim \text{Normal}(0, 1)$$

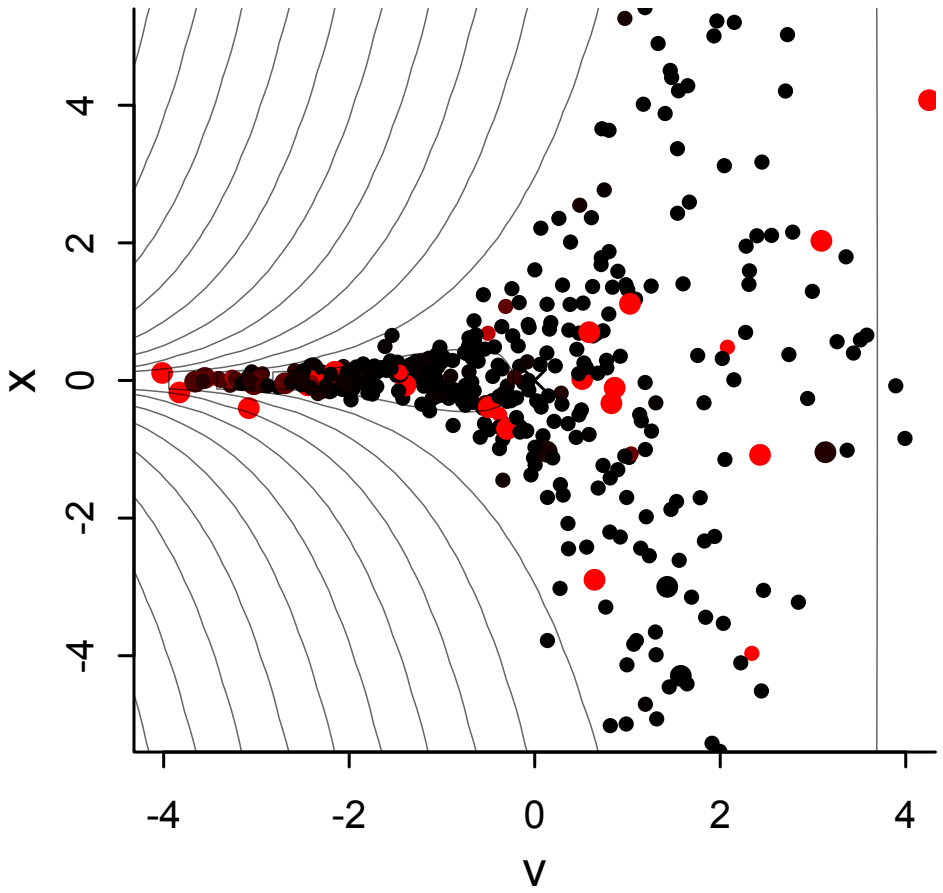
$$v \sim \text{Normal}(0, 3)$$
$$x \sim \text{Normal}(0, \exp(v))$$



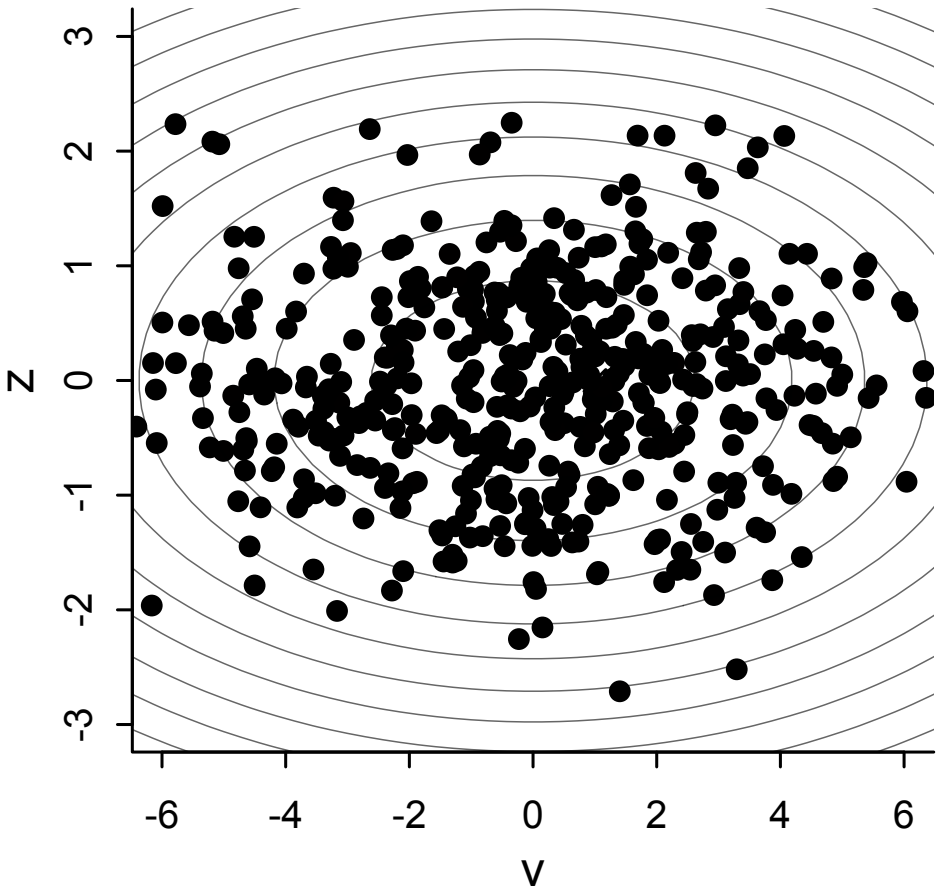
$$v \sim \text{Normal}(0, 3)$$
$$x = z \exp(v)$$
$$z \sim \text{Normal}(0, 1)$$



$$v \sim \text{Normal}(0, 3)$$
$$x \sim \text{Normal}(0, \exp(v))$$



$$v \sim \text{Normal}(0, 3)$$
$$x = z \exp(v)$$
$$z \sim \text{Normal}(0, 1)$$



Re-parameterize!

$$
\begin{aligned}
L_i &\sim \text{Binomial}(1, p_i) \\
\text{logit}(p_i) &= \alpha_{\text{ACTOR}[i]} + \gamma_{\text{BLOCK}[i]} + \beta_{\text{TREATMENT}[i]} \\
\beta_j &\sim \text{Normal}(0, 0.5) \quad \text{, for } j = 1..4 \\
\alpha_j &\sim \text{Normal}(\bar{\alpha}, \sigma_\alpha) \quad \text{, for } j = 1..7 \\
\gamma_j &\sim \text{Normal}(0, \sigma_\gamma) \quad \text{, for } j = 1..6 \\
\bar{\alpha} &\sim \text{Normal}(0, 1.5) \\
\sigma_\alpha &\sim \text{Exponential}(1) \\
\sigma_\gamma &\sim \text{Exponential}(1)
\end{aligned}
$$

$$
\begin{aligned}
L_i &\sim \text{Binomial}(1, p_i) \\
\text{logit}(p_i) &= \alpha_{\text{ACTOR}[i]} + \gamma_{\text{BLOCK}[i]} + \beta_{\text{TREATMENT}[i]}
\end{aligned}
$$

$$
\begin{aligned}
L_i &\sim \text{Binomial}(1, p_i) \\
\text{logit}(p_i) &= \bar{\alpha} + z_{\text{ACTOR}[i]}\sigma_\alpha + x_{\text{BLOCK}[i]}\sigma_\gamma + \beta_{\text{TREATMENT}[i]}
\end{aligned}
$$

$$
\begin{aligned}
L_i &\sim \text{Binomial}(1, p_i) \\
\text{logit}(p_i) &= \alpha_{\text{ACTOR}[i]} + \gamma_{\text{BLOCK}[i]} + \beta_{\text{TREATMENT}[i]}
\end{aligned}
$$

$$
\begin{aligned}
L_i &\sim \text{Binomial}(1, p_i) \\
\text{logit}(p_i) &= \bar{\alpha} + z_{\text{ACTOR}[i]}\sigma_\alpha + x_{\text{BLOCK}[i]}\sigma_\gamma + \beta_{\text{TREATMENT}[i]} \\
\beta_j &\sim \text{Normal}(0, 0.5) \quad \text{, for } j = 1..4 \\
z_j &\sim \text{Normal}(0, 1) \quad \text{, for } j = 1..7 \\
x_j &\sim \text{Normal}(0, 1) \quad \text{, for } j = 1..6 \\
\bar{\alpha} &\sim \text{Normal}(0, 1.5) \\
\sigma_\alpha &\sim \text{Exponential}(1) \\
\sigma_\gamma &\sim \text{Exponential}(1)
\end{aligned}
$$

$$
\begin{aligned}
L_i &\sim \text{Binomial}(1, p_i) \\
\text{logit}(p_i) &= \alpha_{\text{ACTOR}[i]} + \gamma_{\text{BLOCK}[i]} + \beta_{\text{TREATMENT}[i]}
\end{aligned}
$$

$$
\begin{aligned}
L_i &\sim \text{Binomial}(1, p_i) \\
\text{logit}(p_i) &= \bar{\alpha} + z_{\text{ACTOR}[i]}\sigma_\alpha + x_{\text{BLOCK}[i]}\sigma_\gamma + \beta_{\text{TREATMENT}[i]} \\
\beta_j &\sim \text{Normal}(0, 0.5) \quad , \text{for } j = 1..4 \\
z_j &\sim \text{Normal}(0, 1) \quad , \text{for } j = 1..7 \\
x_j &\sim \text{Normal}(0, 1) \quad , \text{for } j = 1..6 \\
\bar{\alpha} &\sim \text{Normal}(0, 1.5) \\
\sigma_\alpha &\sim \text{Exponential}(1) \\
\sigma_\gamma &\sim \text{Exponential}(1)
\end{aligned}
$$

$$L_i \sim \text{Binomial}(1, p_i)$$

$$\text{logit}(p_i) = \bar{\alpha} + z_{\text{ACTOR}[i]}\sigma_\alpha + x_{\text{BLOCK}[i]}\sigma_\gamma + \beta_{\text{TREATMENT}[i]}$$

R code 13.28

```
set.seed(13)
m13.4nc <- ulam(
    alist(
        pulled_left ~ dbinom( 1 , p ) ,
        logit(p) <- a_bar + z[actor]*sigma_a + x[block_id]*sigma_g + b[treatment] ,
        b[treatment] ~ dnorm( 0 , 0.5 ),
        z[actor] ~ dnorm( 0 , 1 ),
        x[block_id] ~ dnorm( 0 , 1 ),
        a_bar ~ dnorm( 0 , 1.5 ),
        sigma_a ~ dexp(1),
        sigma_g ~ dexp(1)
    ) , data=dat_list , chains=4 , cores=4 )
```

Non-centered vs centered

R code 13.29

```
neff_c <- precis( m13.4 , depth=2 )[['n_eff']]
neff_nc <- precis( m13.4nc , depth=2 )[['n_eff']]
par_names <- rownames( precis( m13.4 , depth=2 ) )
neff_table <- cbind( neff_c , neff_nc )
rownames(neff_table) <- par_names
round(t(neff_table))
```

```
        b[1] b[2] b[3] b[4] a[1] a[2] a[3] a[4] a[5] a[6] a[7] g[1] g[2] g[3]
neff_c   584  572  587  523  562  759  541  547  582  601  677  418  793  784
neff_nc 1144 1233 1144 1115  592  962  614  614  611  627  772 1811 2137 1748
        g[4] g[5] g[6] a_bar sigma_a sigma_g
neff_c   605  934  603   824     898     243
neff_nc 1575 2127 1429   573     777     970
```

Posterior predictions

- Predictions more subtle: Same clusters or new clusters?
- Same clusters: proceed as usual
- New clusters: should average over distribution of varying effects
- In this case:
  - Same clusters: Predictions for these chimpanzees
  - New clusters: Prediction for a new chimpanzee or rather for population of chimpanzees

Same clusters, new clusters

- Same actors:
  - Really same as before: varying effects are just parameters; you know the model; push samples back through the model
  - link() and sim() obey this rule
- New actors (counterfactual):
  - which actor (cluster) to use for counterfactual predictions?
    - average actor
    - marginal of actor
    - show sample of actors from posterior

Average actor

- "average actor" means actor with population average intercept, "alpha"
- Strategy:
  - replace varying intercept samples with zeros => all actors have average intercept now
  - compute predictions as usual

```
# replace varying intercept samples with zeros
# 1000 samples by 7 actors
a_actor_zeros <- matrix(0,1000,7)
```

R code 12.32

```
# fire up link
# note use of replace list
link.m12.4 <- link( m12.4 , n=1000 , data=d.pred ,
    replace=list(a_actor=a_actor_zeros) )
```

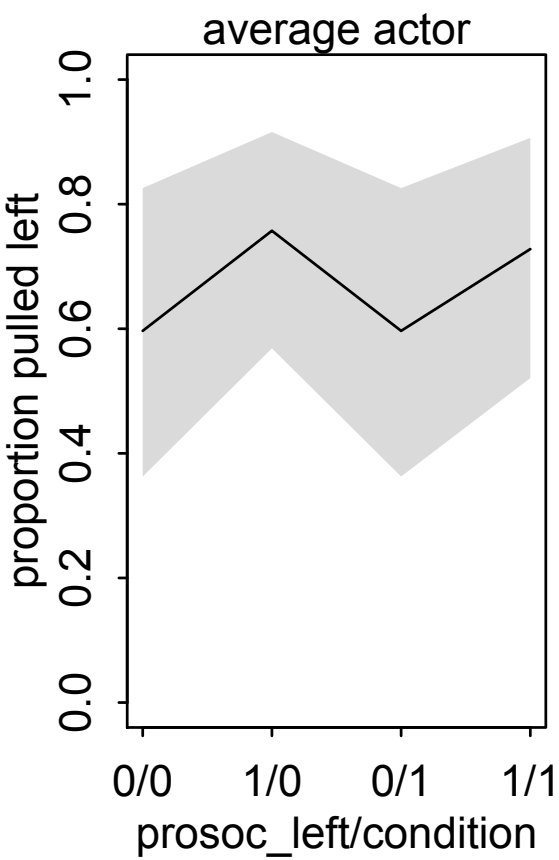
R code 12.33

R code 12.32

```
# replace varying intercept samples with zeros
# 1000 samples by 7 actors
a_actor_zeros <- matrix(0,1000,7)
```

R code 12.33

```
# fire up link
# note use of replace list
link.m12.4 <- link( m12.4 , n=1000 , data=d.pred ,
    replace=list(a_actor=a_actor_zeros) )
```



Marginal of actor

- "Marginal of" means "averaging over variation in actors" => shows variation arising from variation across actors

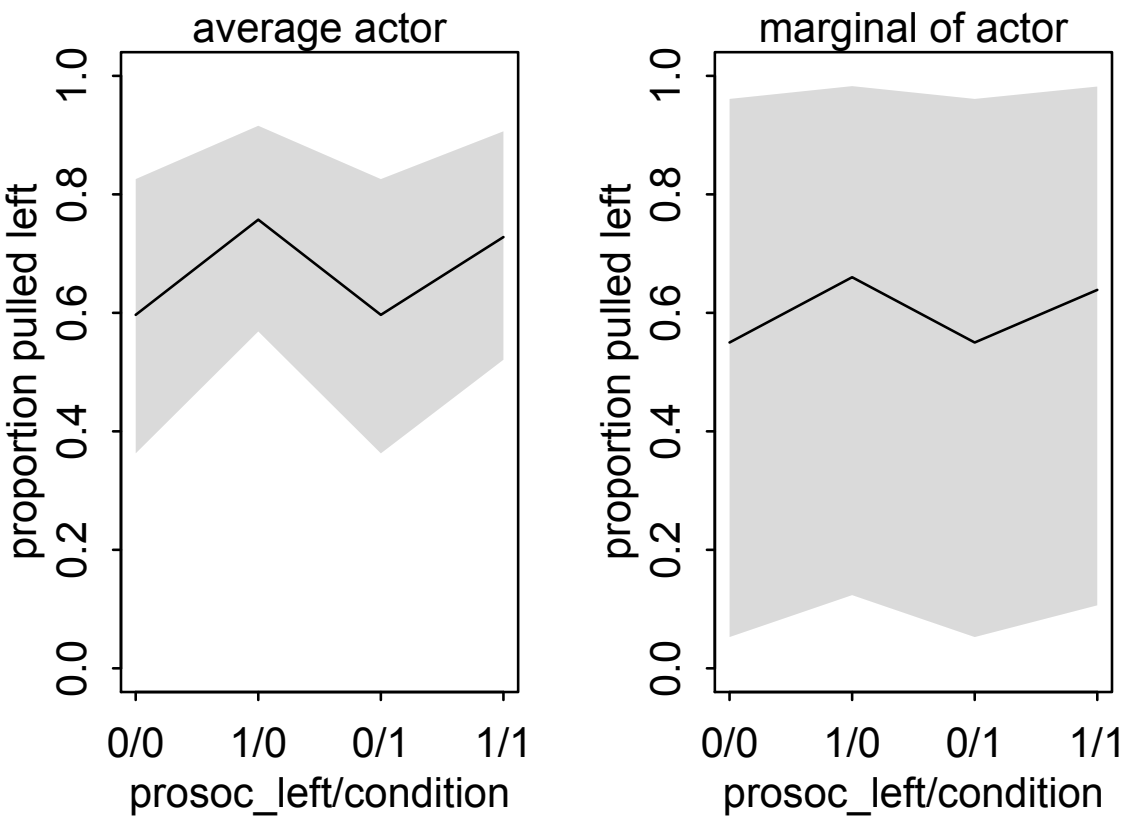
$$\alpha_{\text{ACTOR}} \sim \text{Normal}(0, \sigma_{\text{ACTOR}})$$

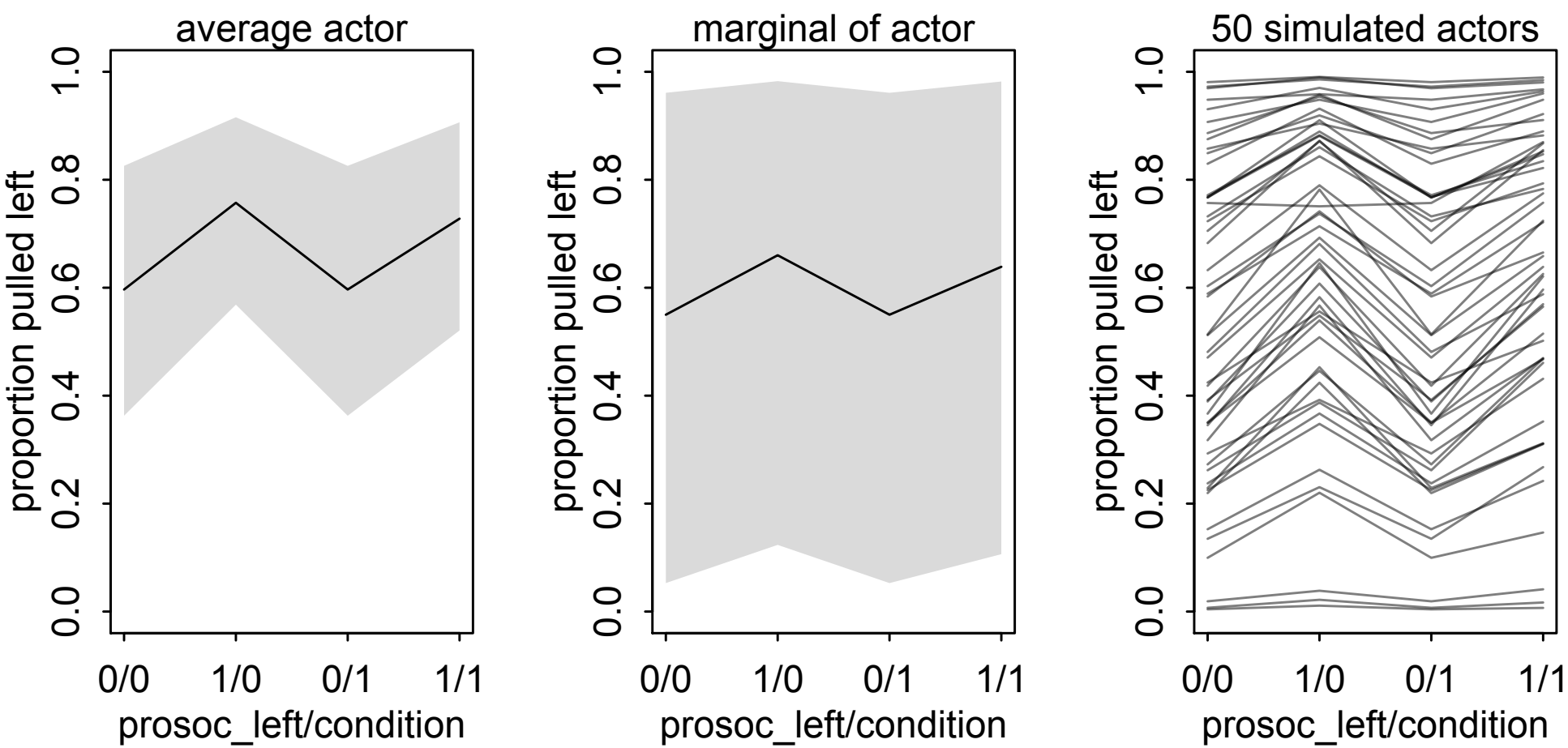
- Strategy:
  - Extract samples for sigma_actor
  - Simulate new varying intercepts
  - Use simulated intercepts to simulate predictions

R code 12.34

```
# replace varying intercept samples with simulations
post <- extract.samples(m12.4)
a_actor_sims <- rnorm(7000,0,post$sigma_actor)
a_actor_sims <- matrix(a_actor_sims,1000,7)

# fire up link
# note use of replace list
link.m12.4 <- link( m12.4 , n=1000 , data=d.pred ,
    replace=list(a_actor=a_actor_sims) )
```




average actor
marginal of actor
50 simulated actors
proportion pulled left
0.0
0.2
0.4
0.6
0.8
1.0
0/0
1/0
0/1
1/1
prosoc_left/condition

Homework

- Frogs & contraception
- Next week: Chapter 14 — varying slopes and other wonders