Statistical Rethinking

2023

12. Multilevel Models

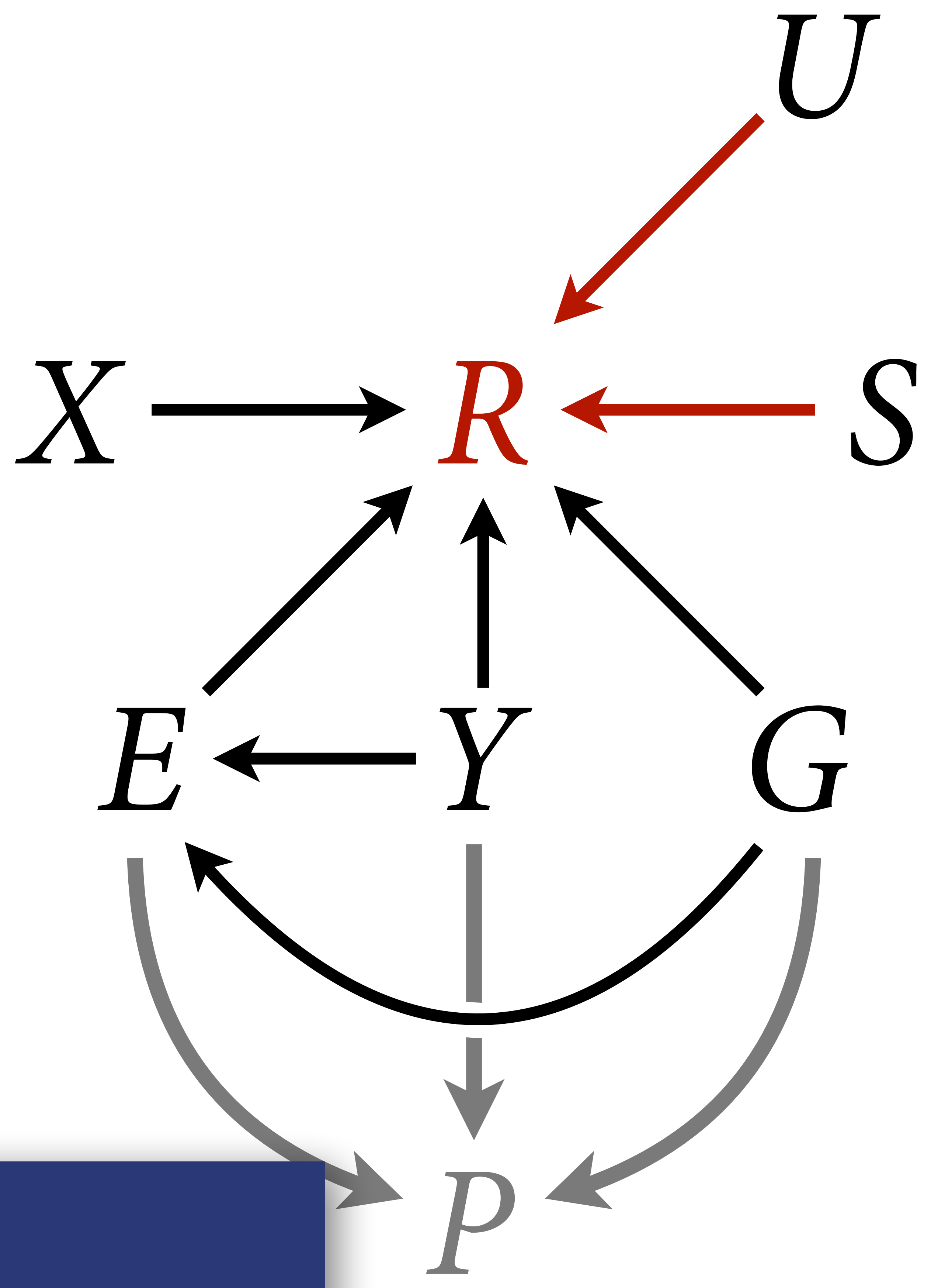
Repeat observations

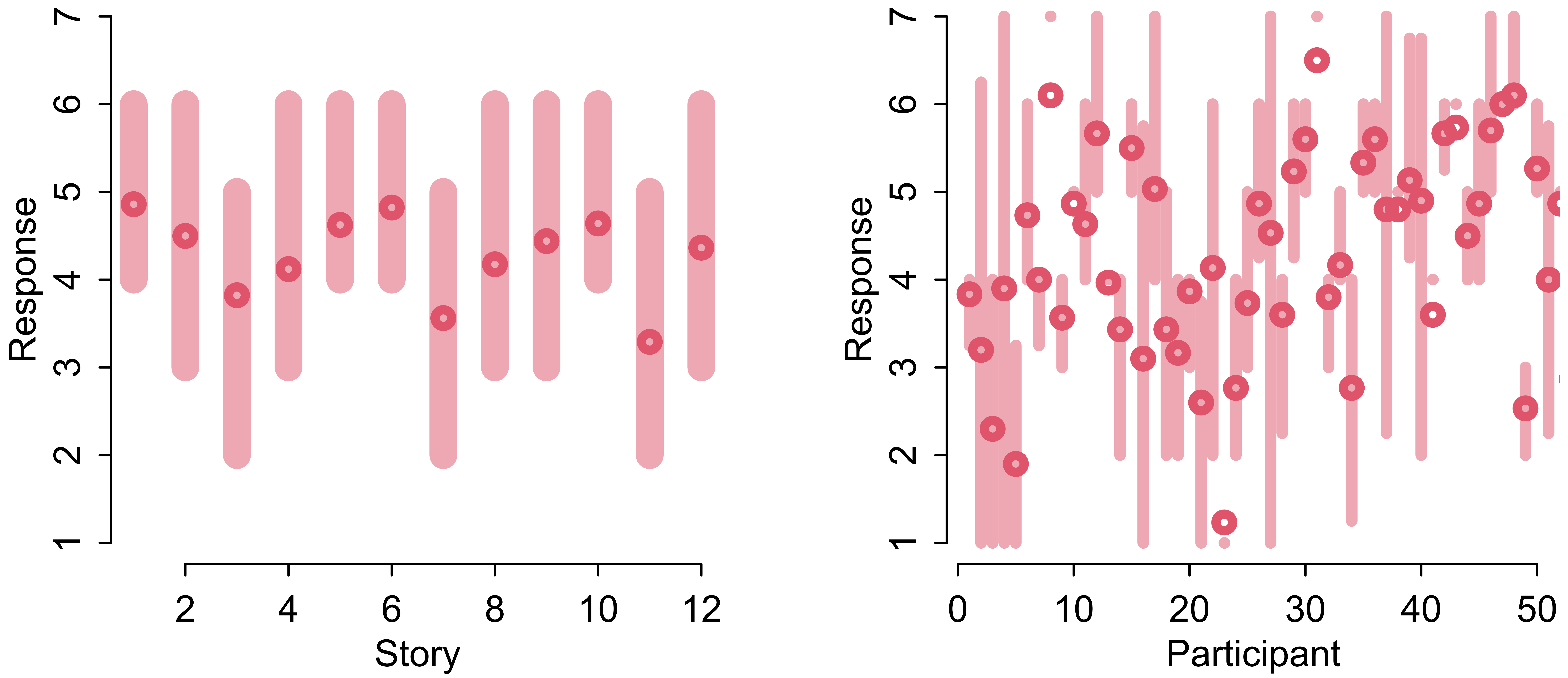
12 stories (S)

```
> table(d$story)

 aqu  boa  box  bur  car  che  pon  rub  sha  shi  spe  swi 
 662  662 1324 1324  662  662  662  662  662  662  993  993 
```

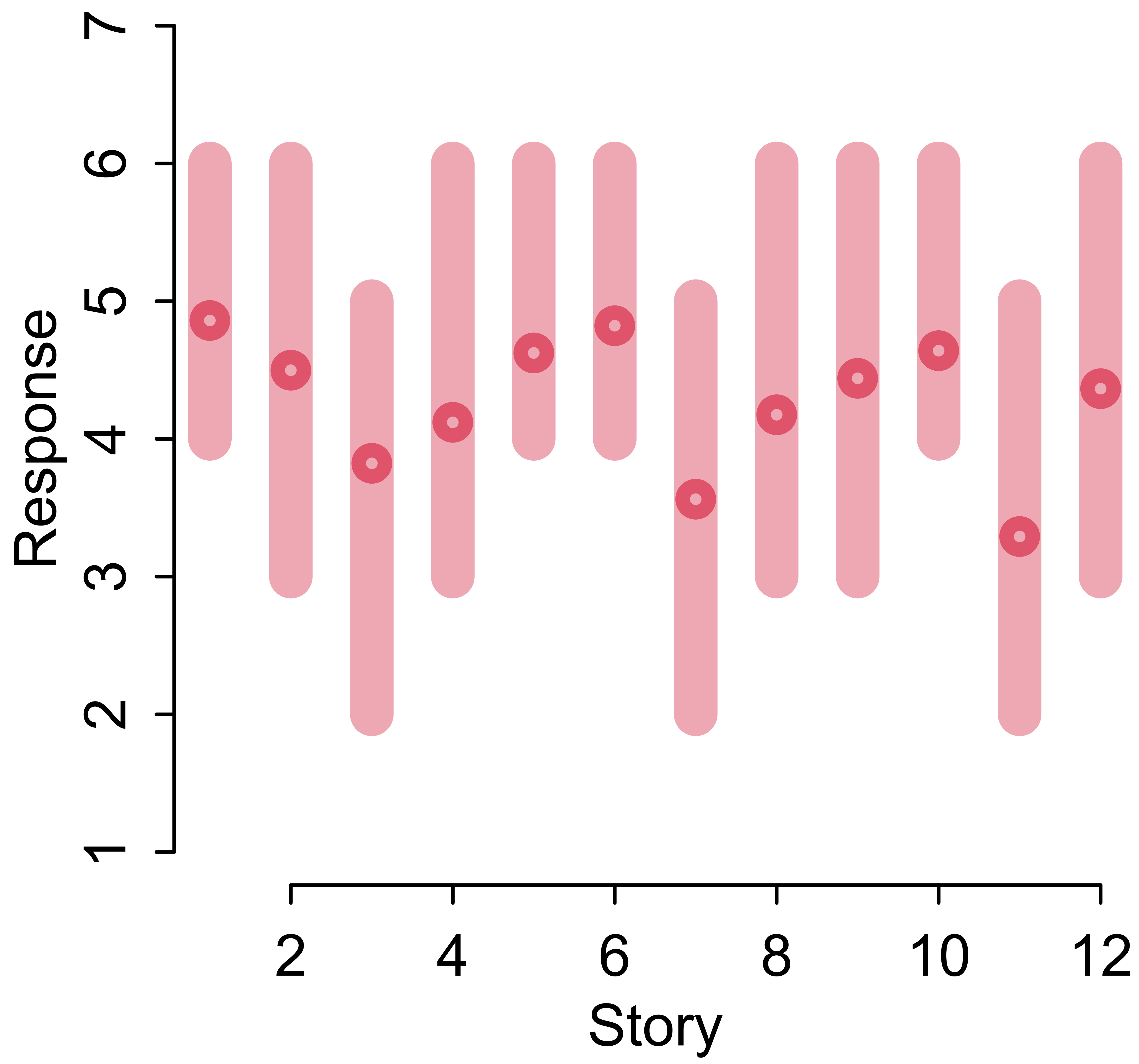
331 individuals (U)

```
> table(d$id)

96;434 96;445 96;451 96;456 96;458 96;466 96;467 96;474 96;480 96;481 96;497 
    30     30     30     30     30     30     30     30     30     30     30 
96;498 96;502 96;505 96;511 96;512 96;518 96;519 96;531 96;533 96;538 96;547 
    30     30     30     30     30     30     30     30     30     30     30 
96;550 96;553 96;555 96;558 96;560 96;562 96;566 96;570 96;581 96;586 96;591 
    30     30     30     30     30     30     30     30     30     30     30 
```





Response
1
2
3
4
5
6
7
2
4
6
8
10
12
Story
Response
1
2
3
4
5
6
7
0
10
20
30
40
50
Participant



$$R_i \sim \text{OrderedLogit}(\phi_i, \alpha)$$

$$\phi_i = \beta_{S[i]}$$

This model has anterograde amnesia

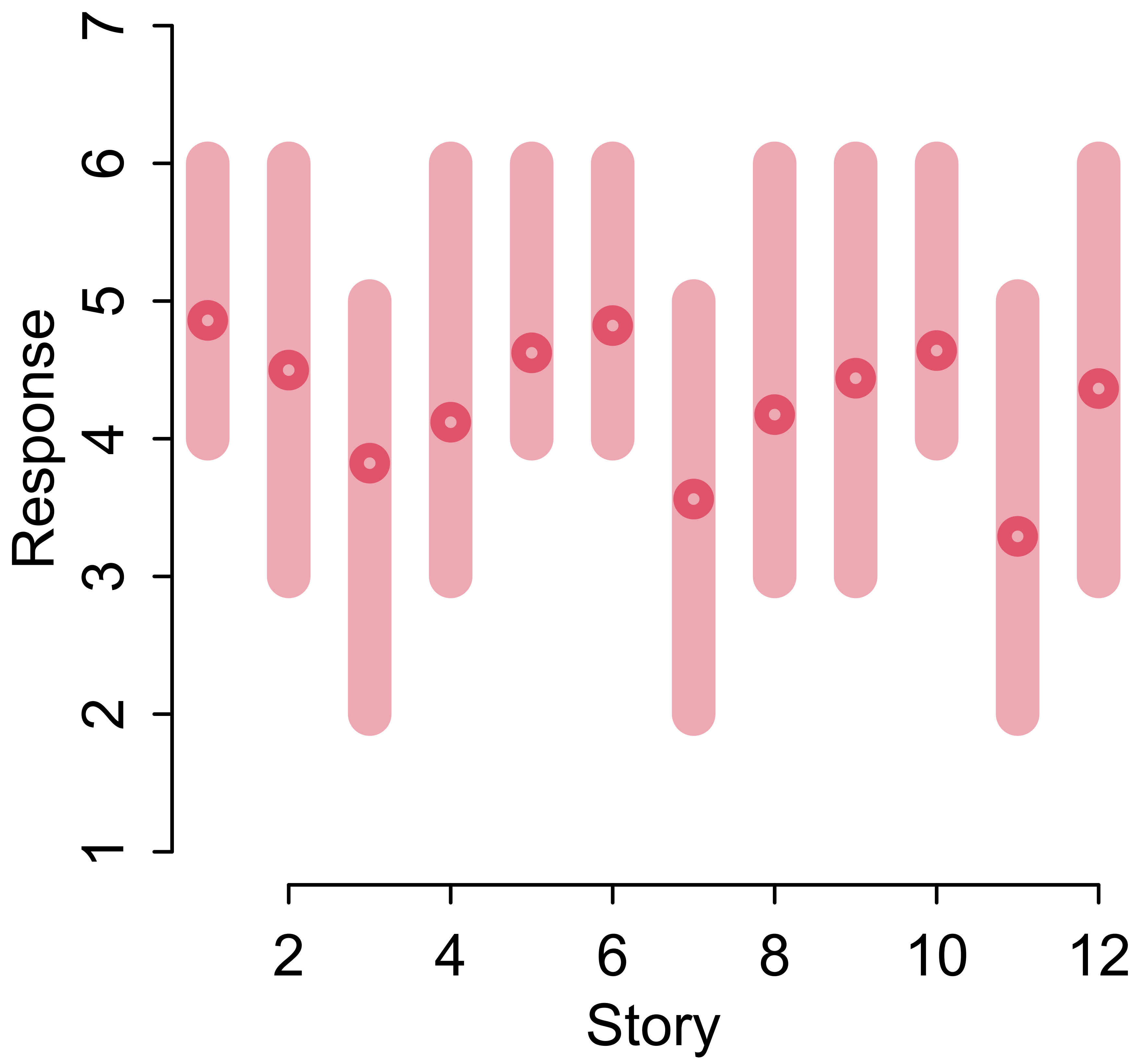
Models With Memory

Multilevel models are models within models

(1) Model observed groups/individuals

(2) Model of population of groups/individuals

The population model creates a kind of memory

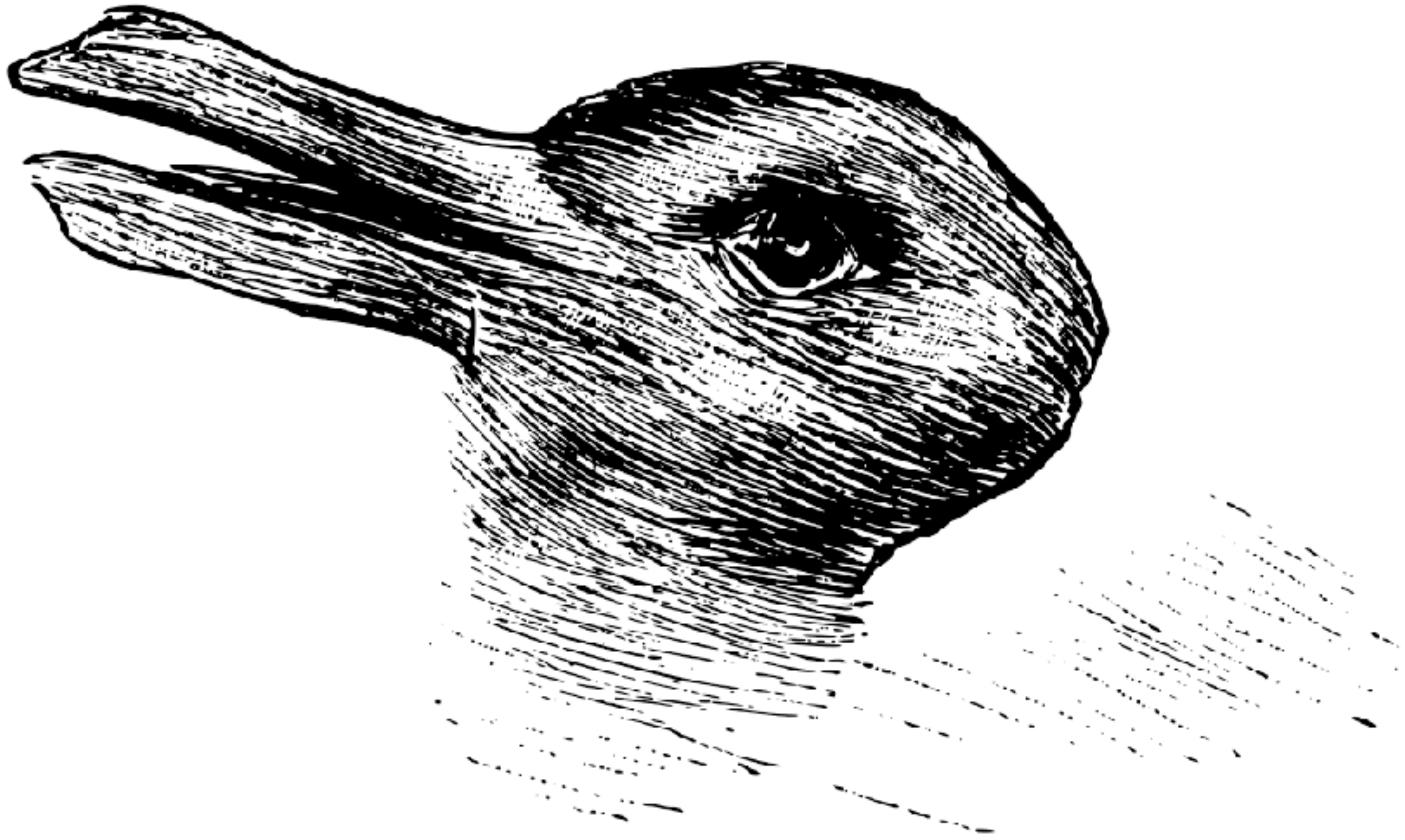


Two Perspectives

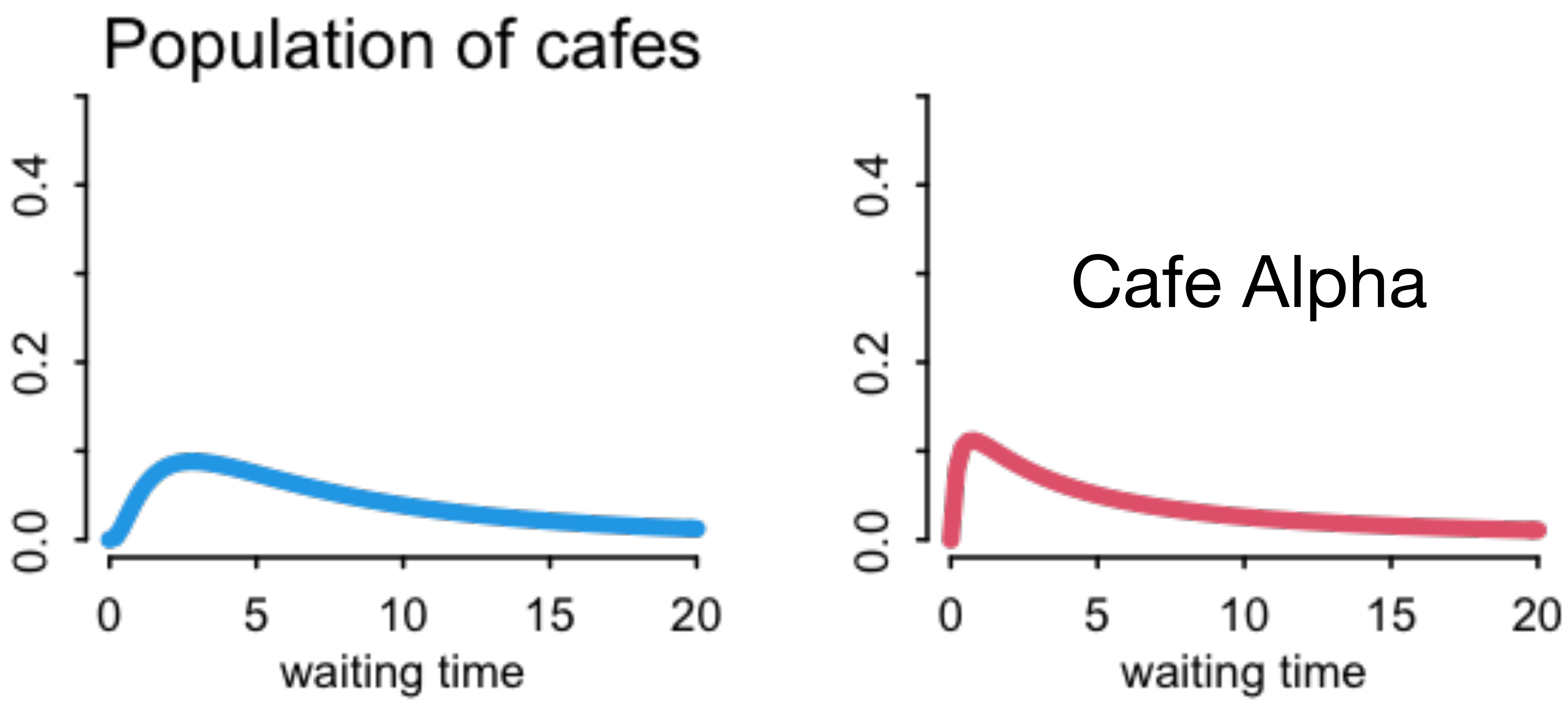
(1) Models with memory learn faster, better

(2) Models with memory resist overfitting

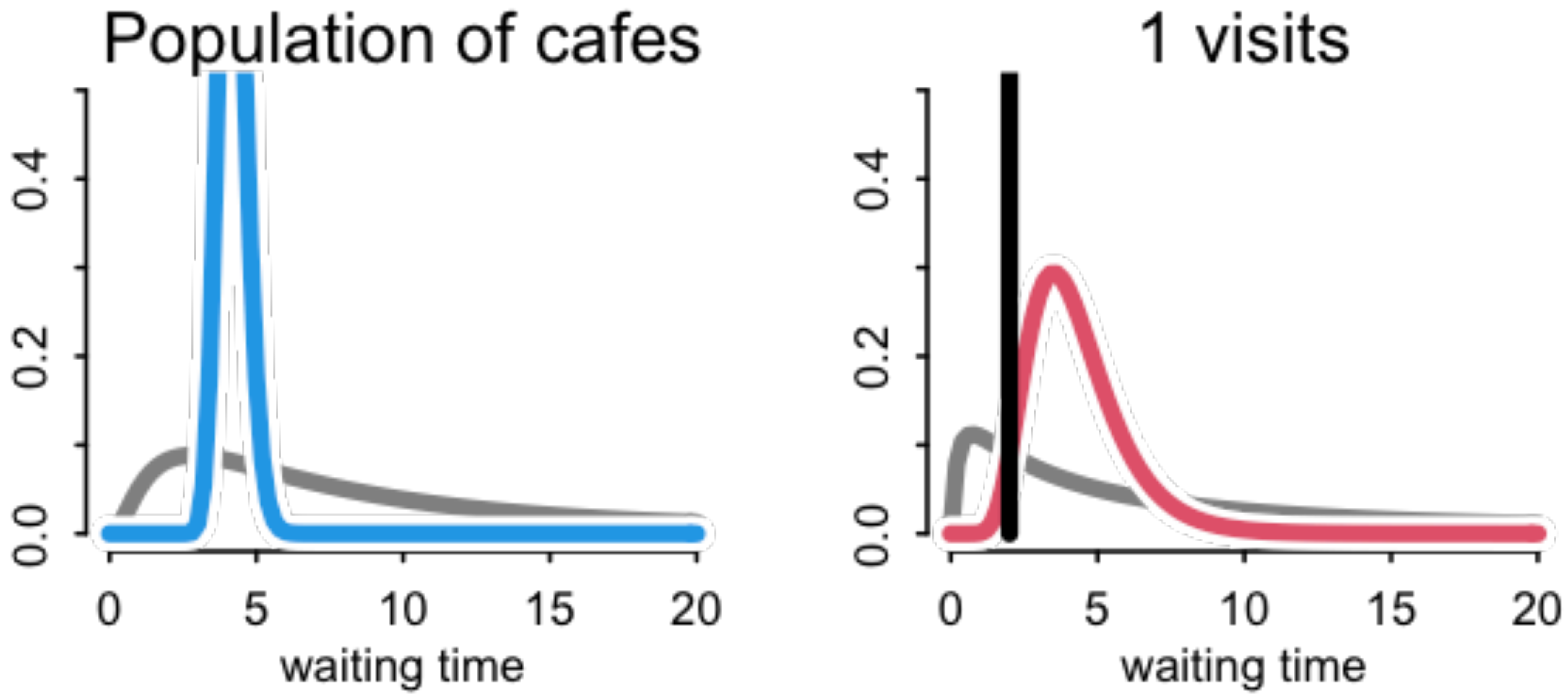
Welche Thiere gleichen einander am meiſten?

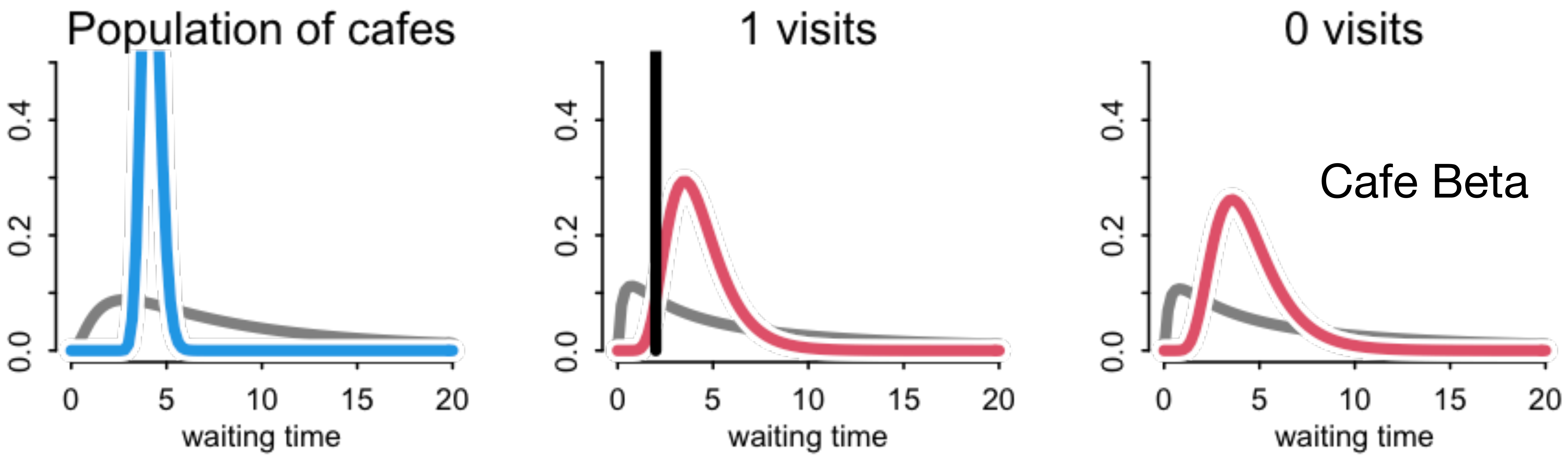


Kaninchen und Ente.

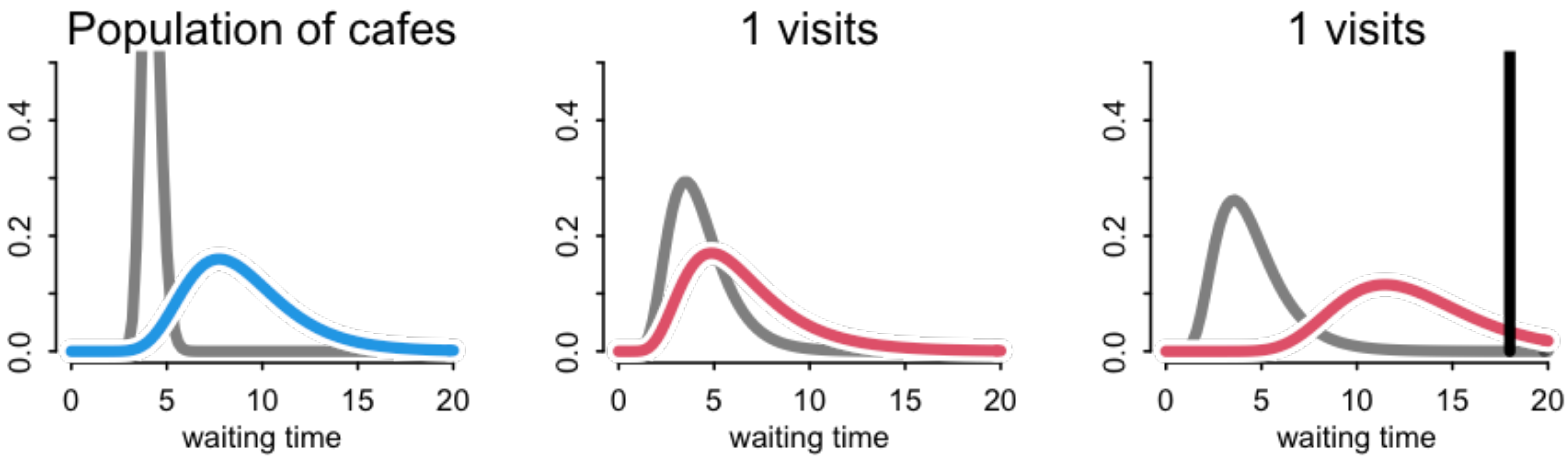


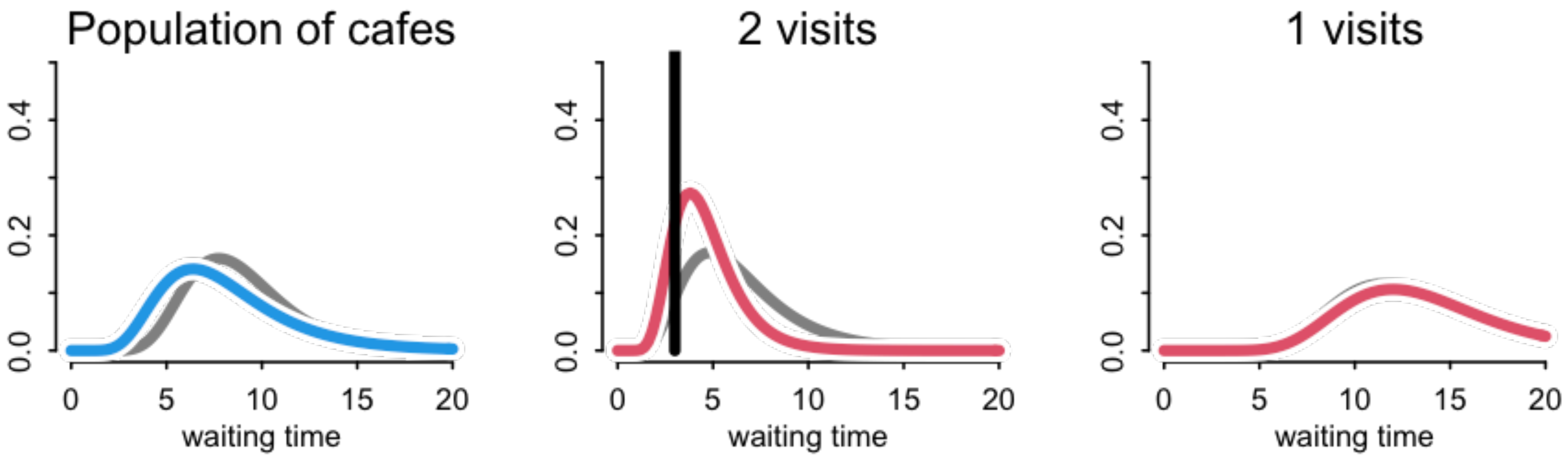
Population of cafes
Cafe Alpha
0.0
0.2
0.4
0
5
10
15
20
waiting time


Population of cafes
0.0
0.2
0.4
0
5
10
15
20
waiting time
1 visits
0.0
0.2
0.4
0
5
10
15
20
waiting time

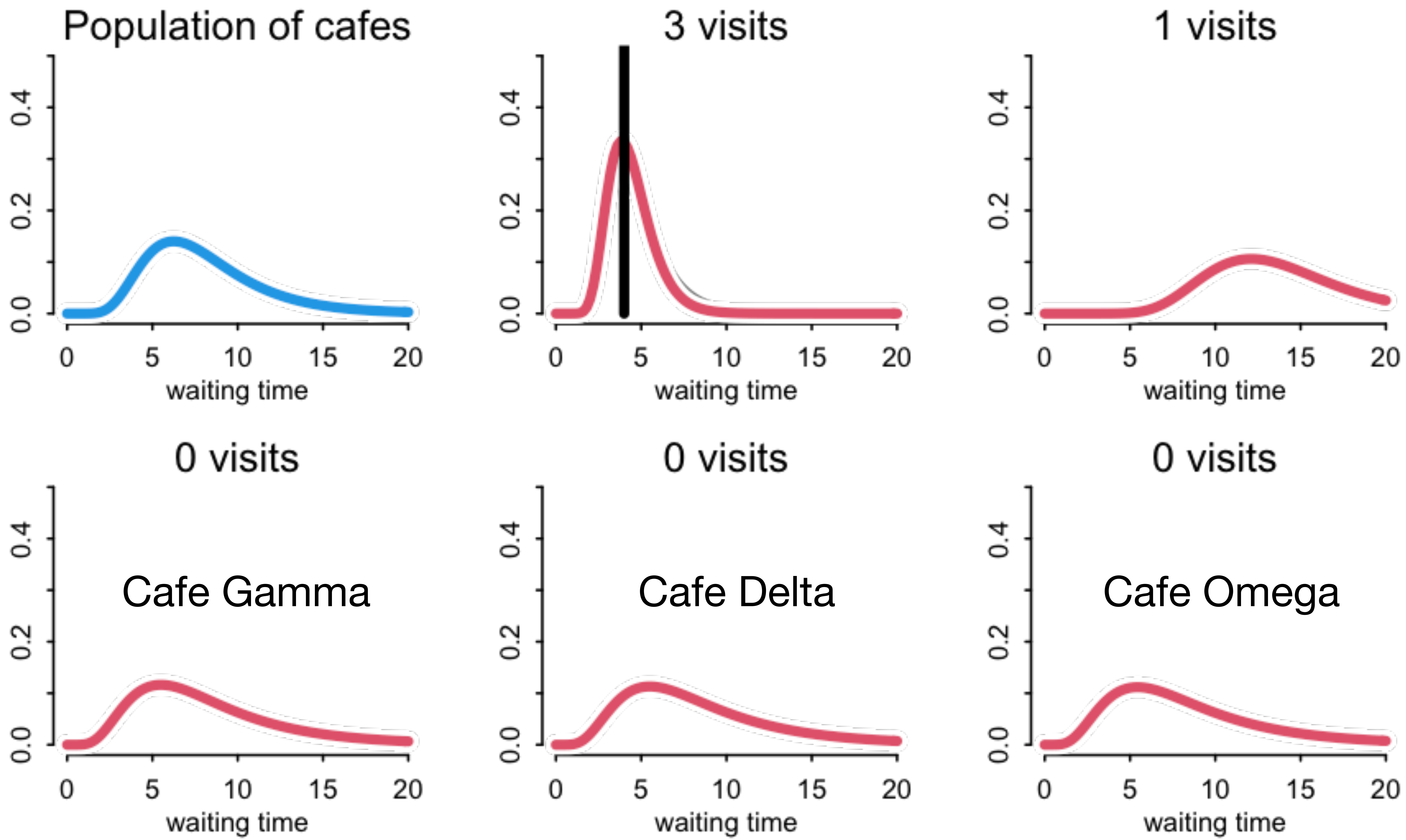


Population of cafes
1 visits
0 visits
Cafe Beta
0.0
0.2
0.4
0
5
10
15
20
waiting time

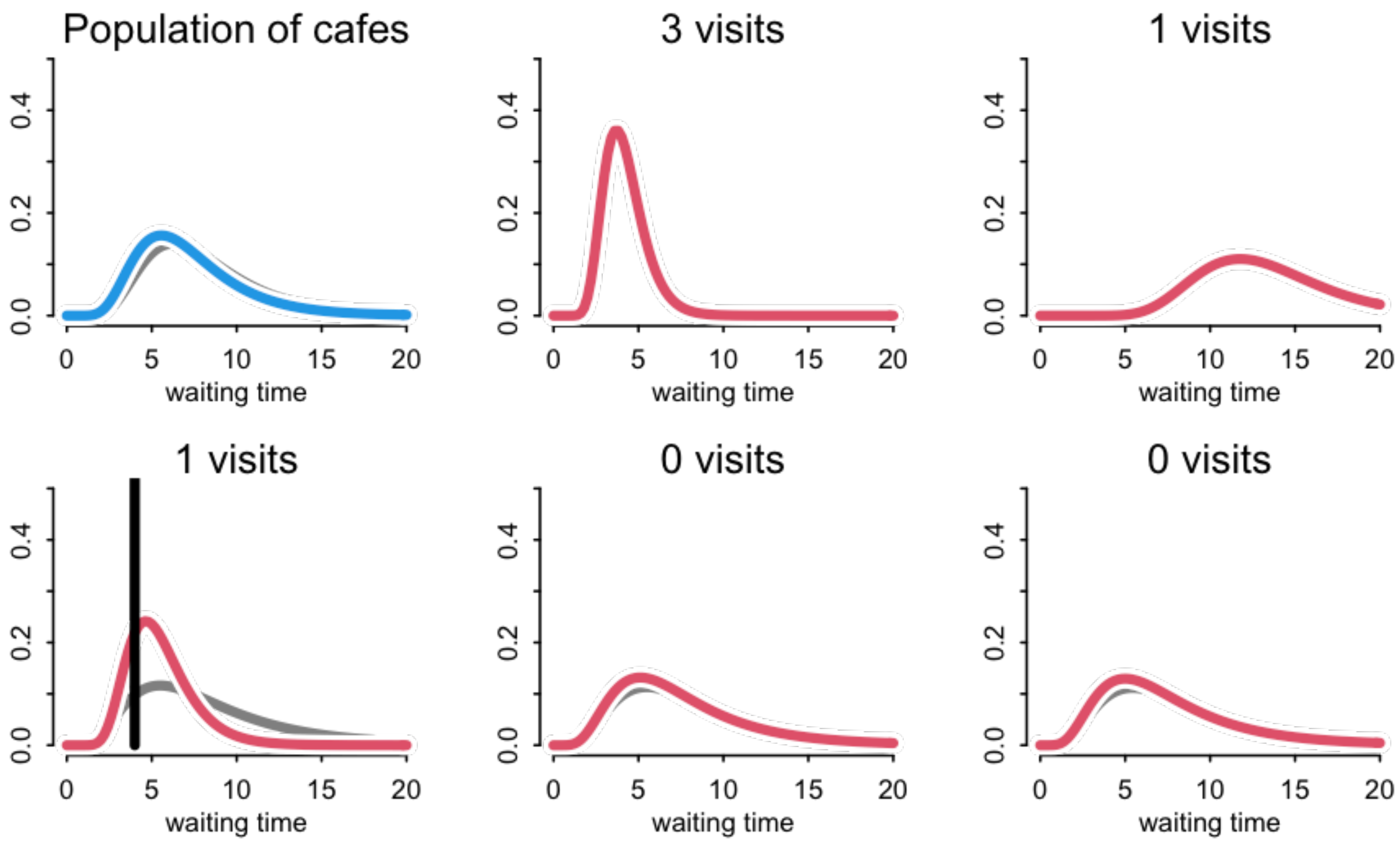

Population of cafes
1 visits
1 visits
0.0
0.2
0.4
0
5
10
15
20
waiting time



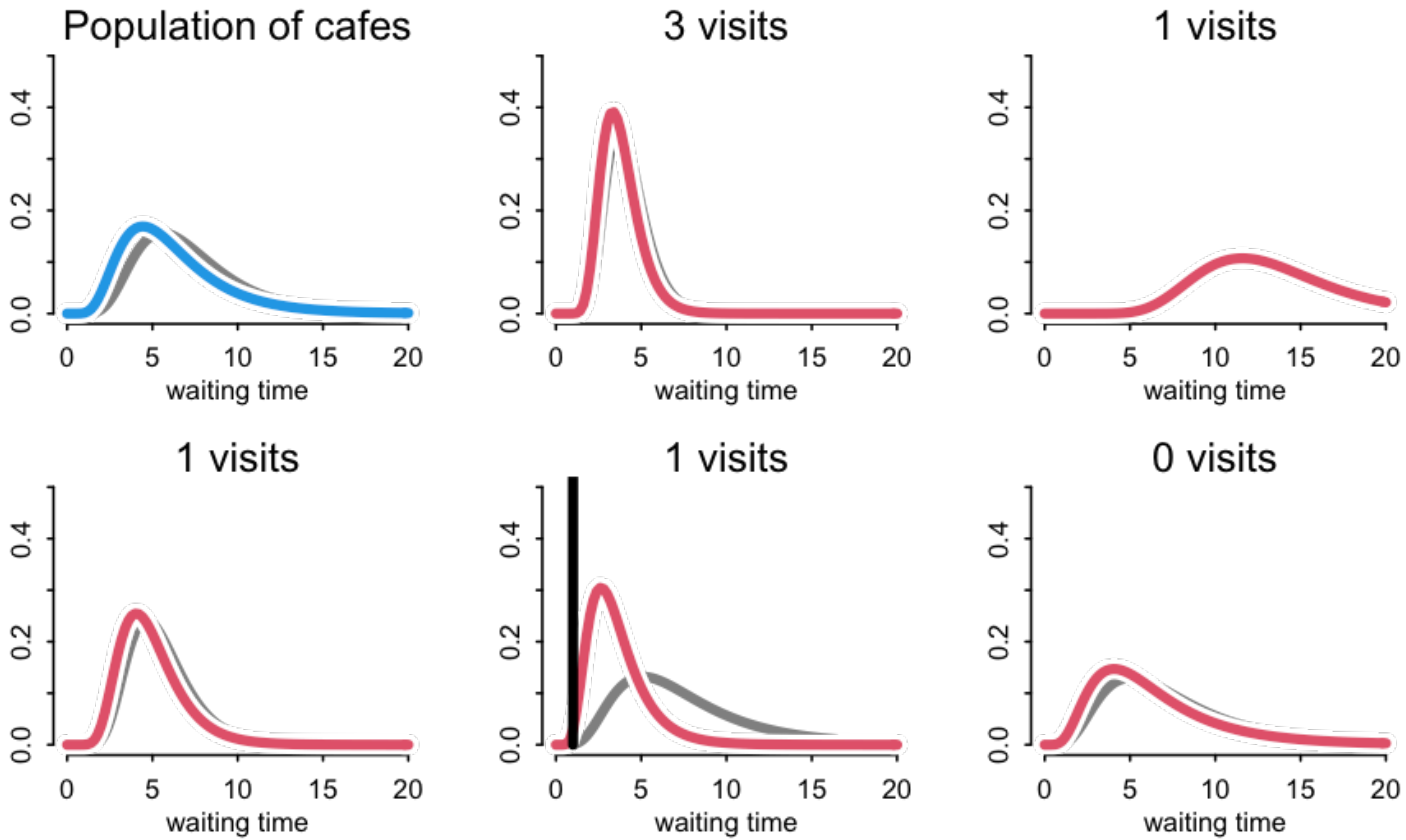
Population of cafes
2 visits
1 visits
0.0
0.2
0.4
0
5
10
15
20
waiting time



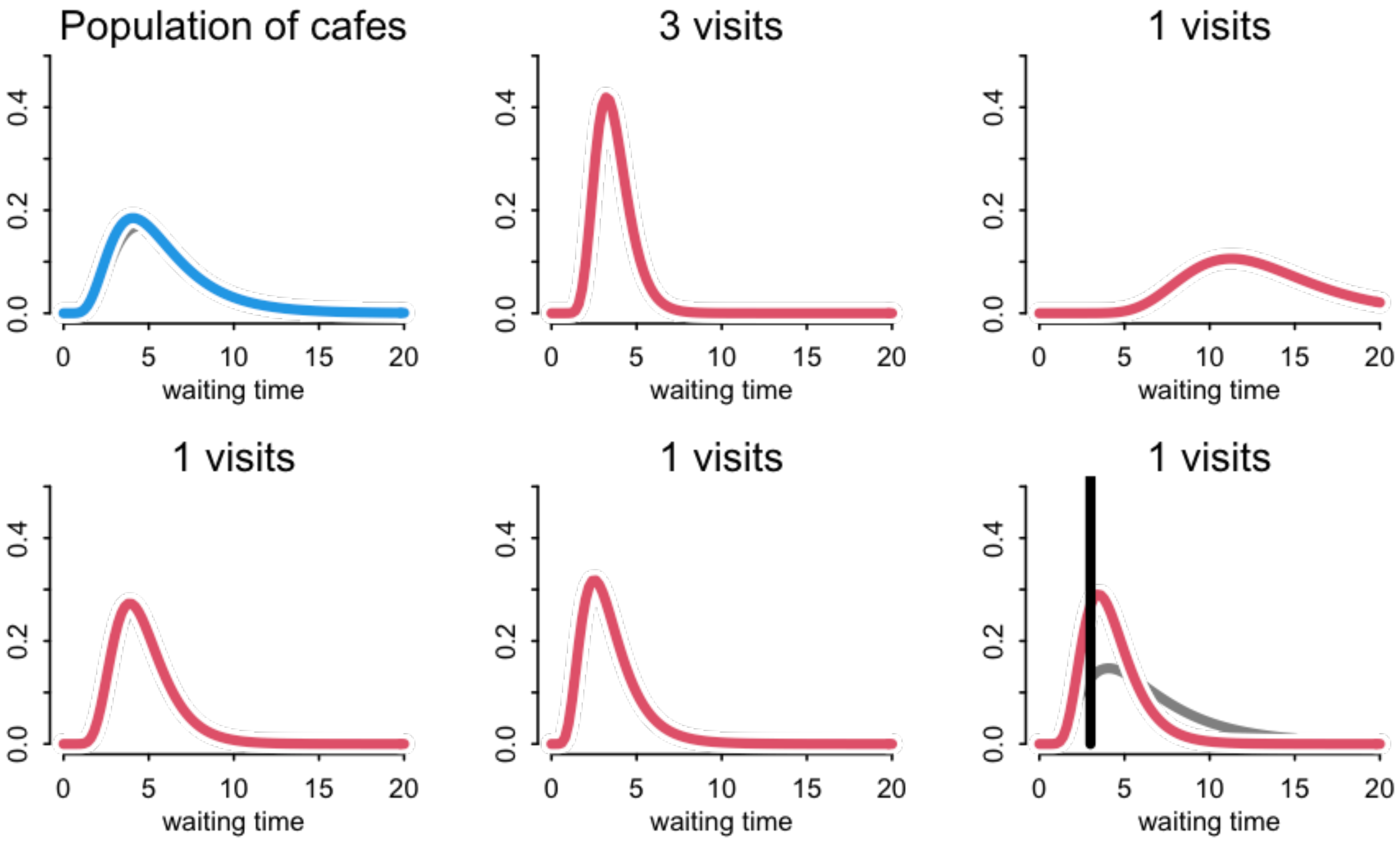
Population of cafes
3 visits
1 visits
0 visits
0 visits
0 visits
Cafe Gamma
Cafe Delta
Cafe Omega
0.0
0.2
0.4
0
5
10
15
20
waiting time



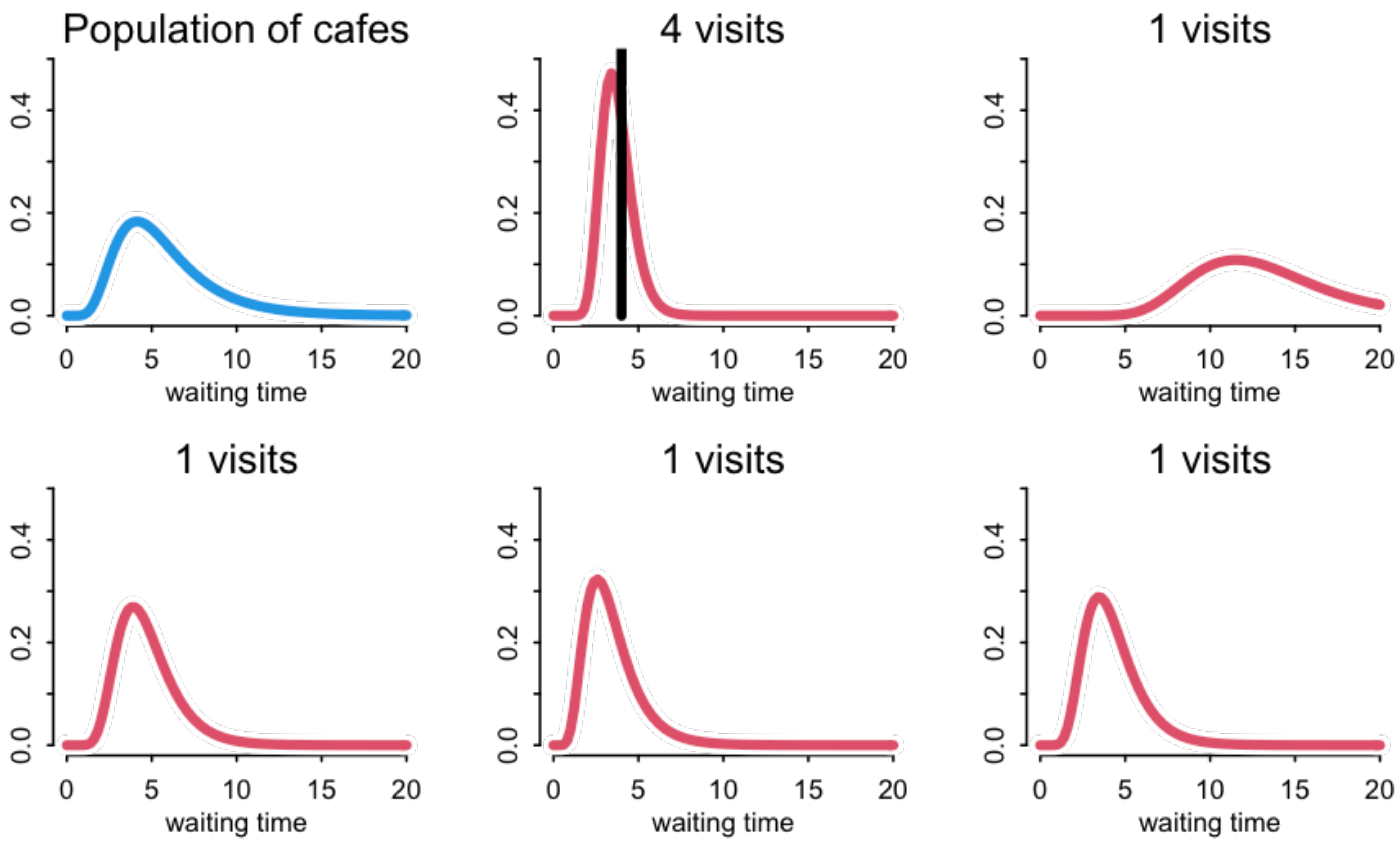
Population of cafes
3 visits
1 visits
1 visits
0 visits
0 visits
0.0
0.2
0.4
0
5
10
15
20
waiting time

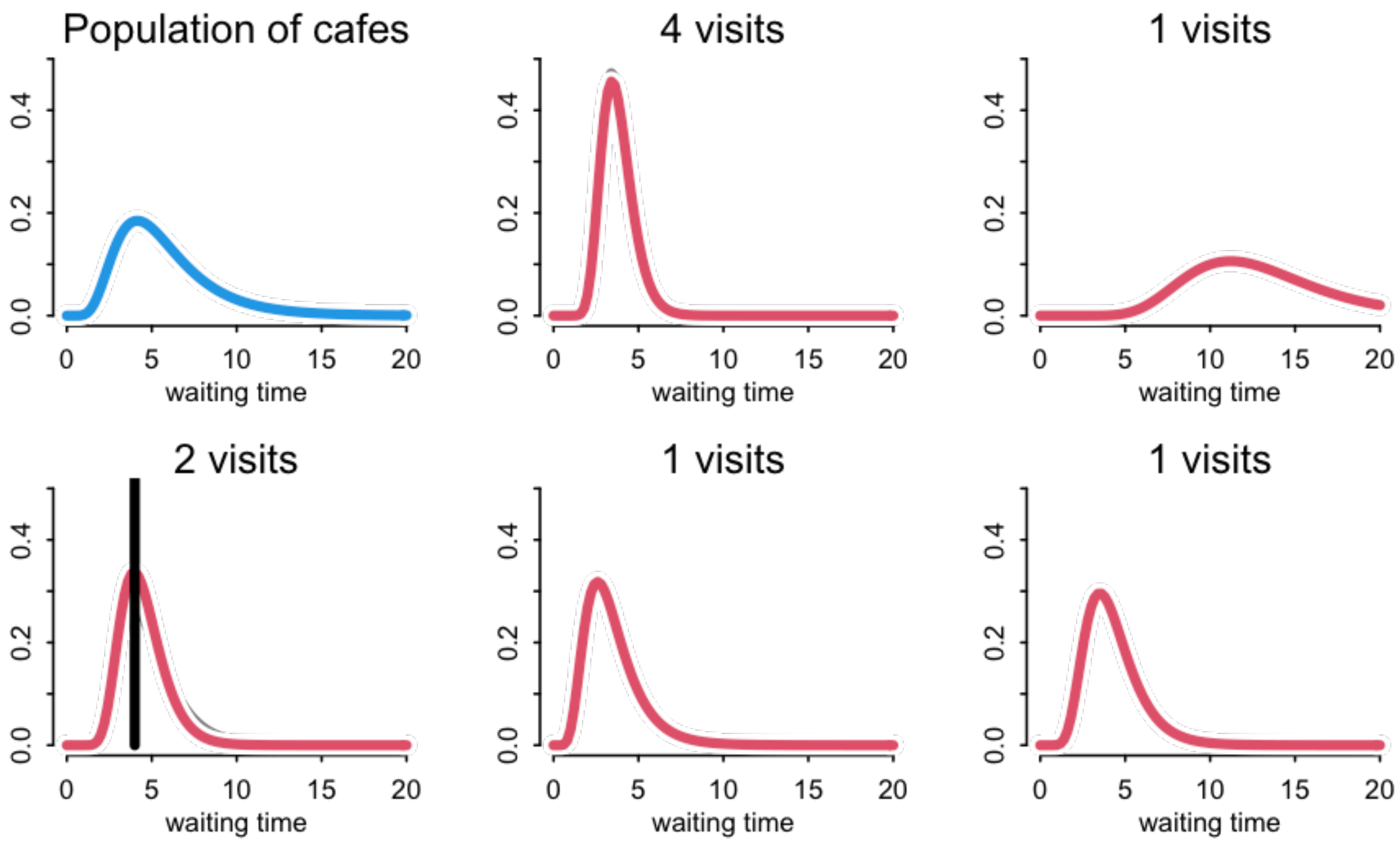


Population of cafes
3 visits
1 visits
1 visits
1 visits
0 visits
0.0
0.2
0.4
0
5
10
15
20
waiting time

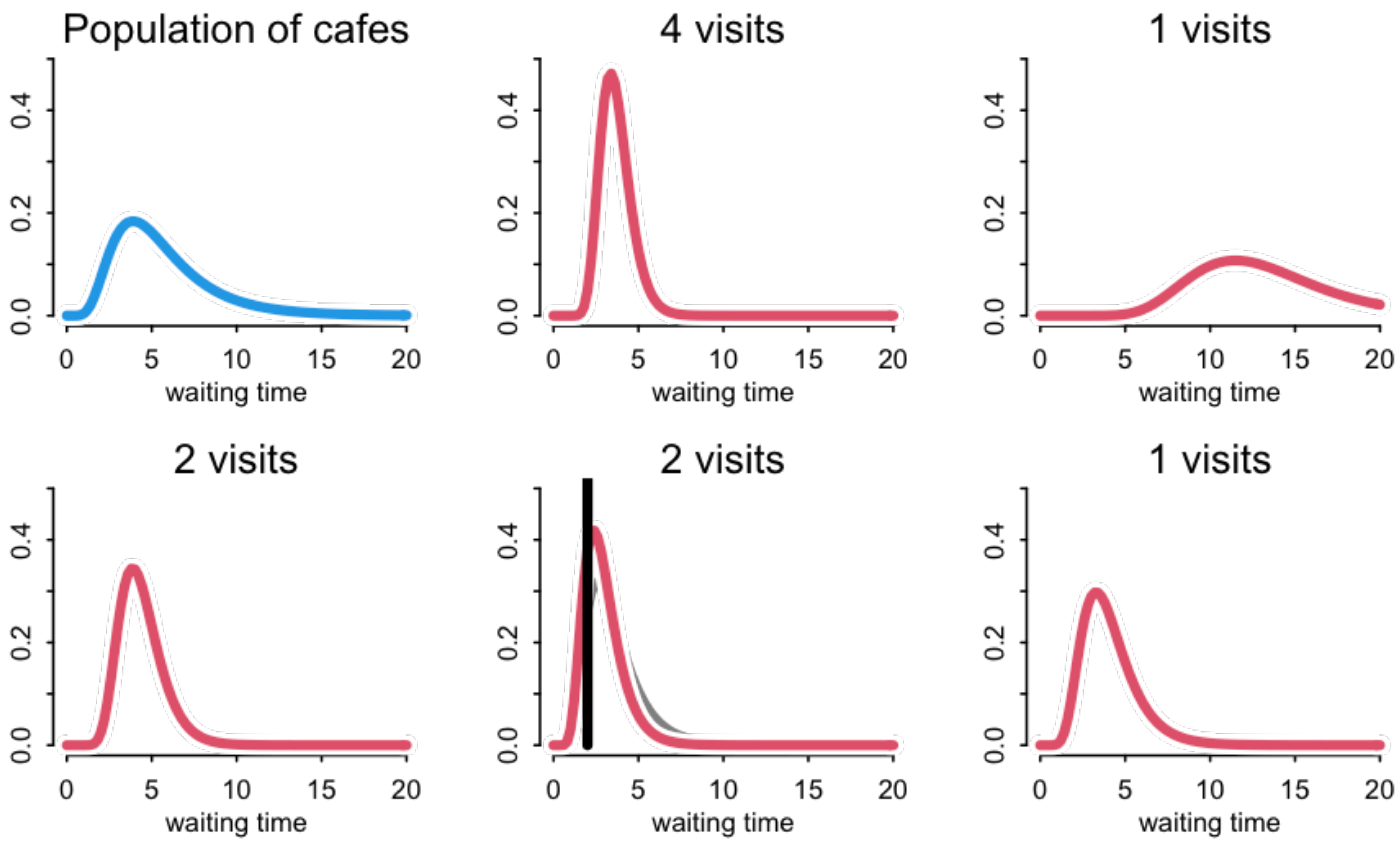


Population of cafes
3 visits
1 visits
1 visits
1 visits
1 visits
0.0
0.2
0.4
0
5
10
15
20
waiting time

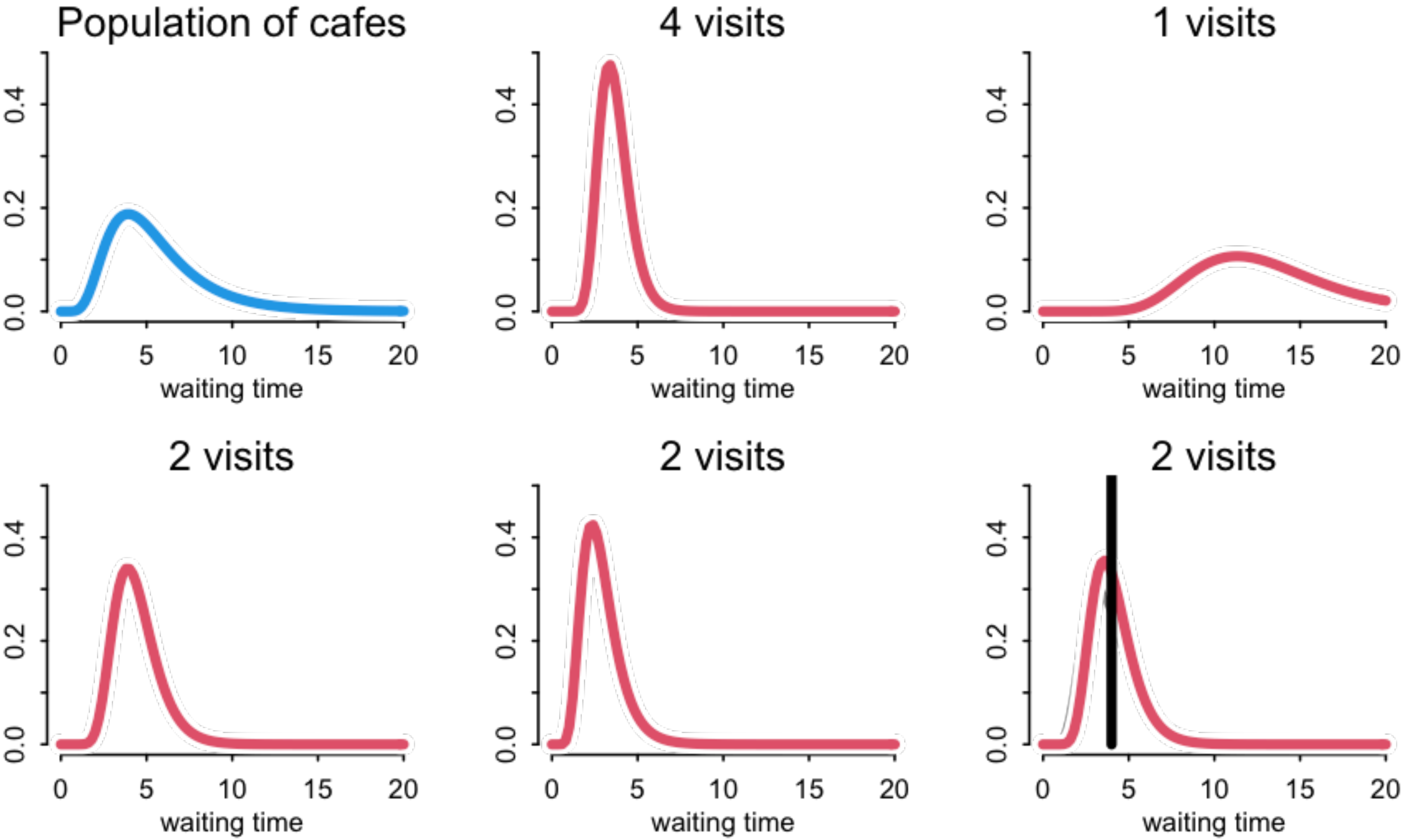

Population of cafes
4 visits
1 visits
1 visits
1 visits
1 visits
waiting time
0.0
0.2
0.4
0
5
10
15
20



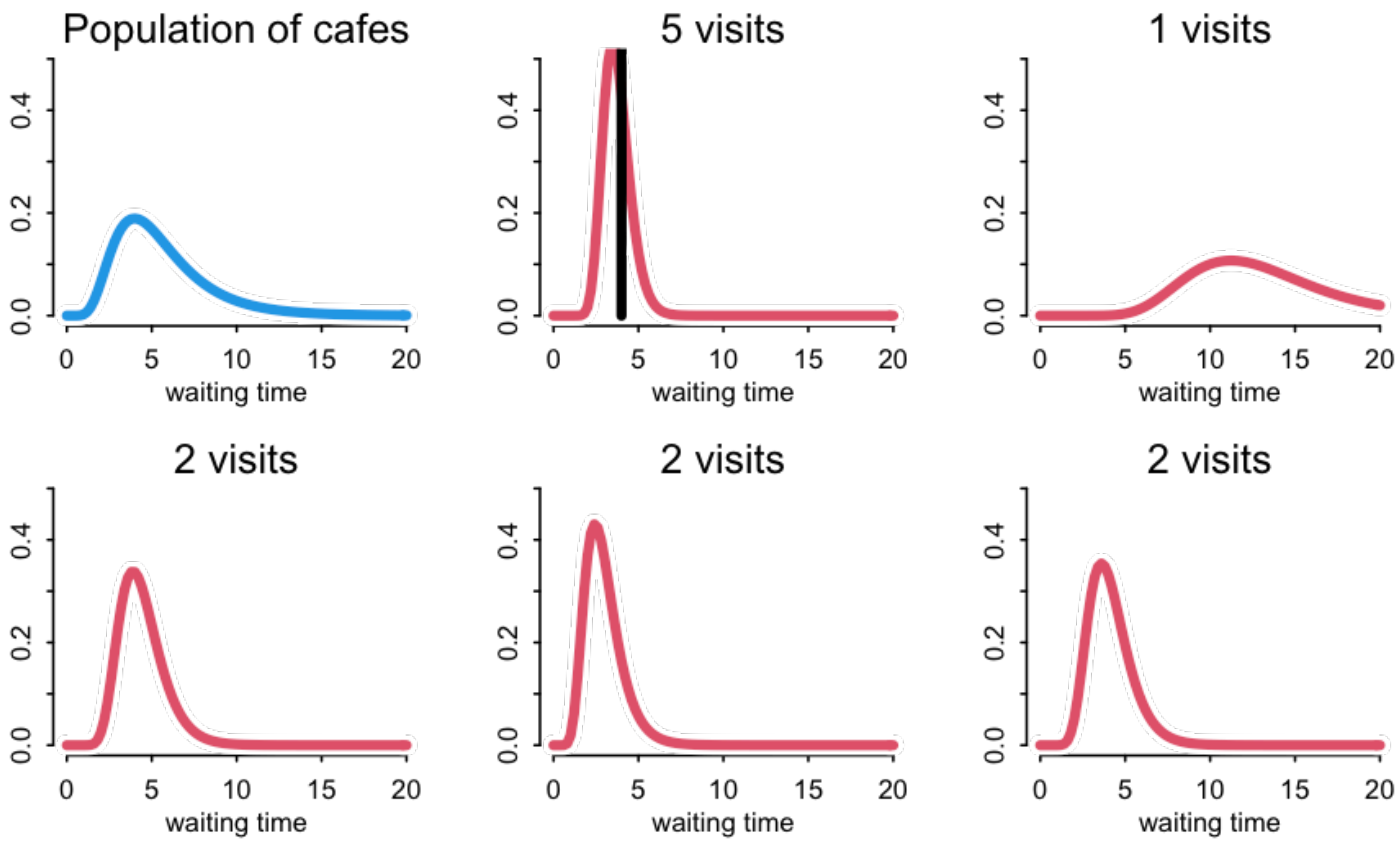
Population of cafes
4 visits
1 visits
2 visits
1 visits
1 visits
0.0
0.2
0.4
0
5
10
15
20
waiting time



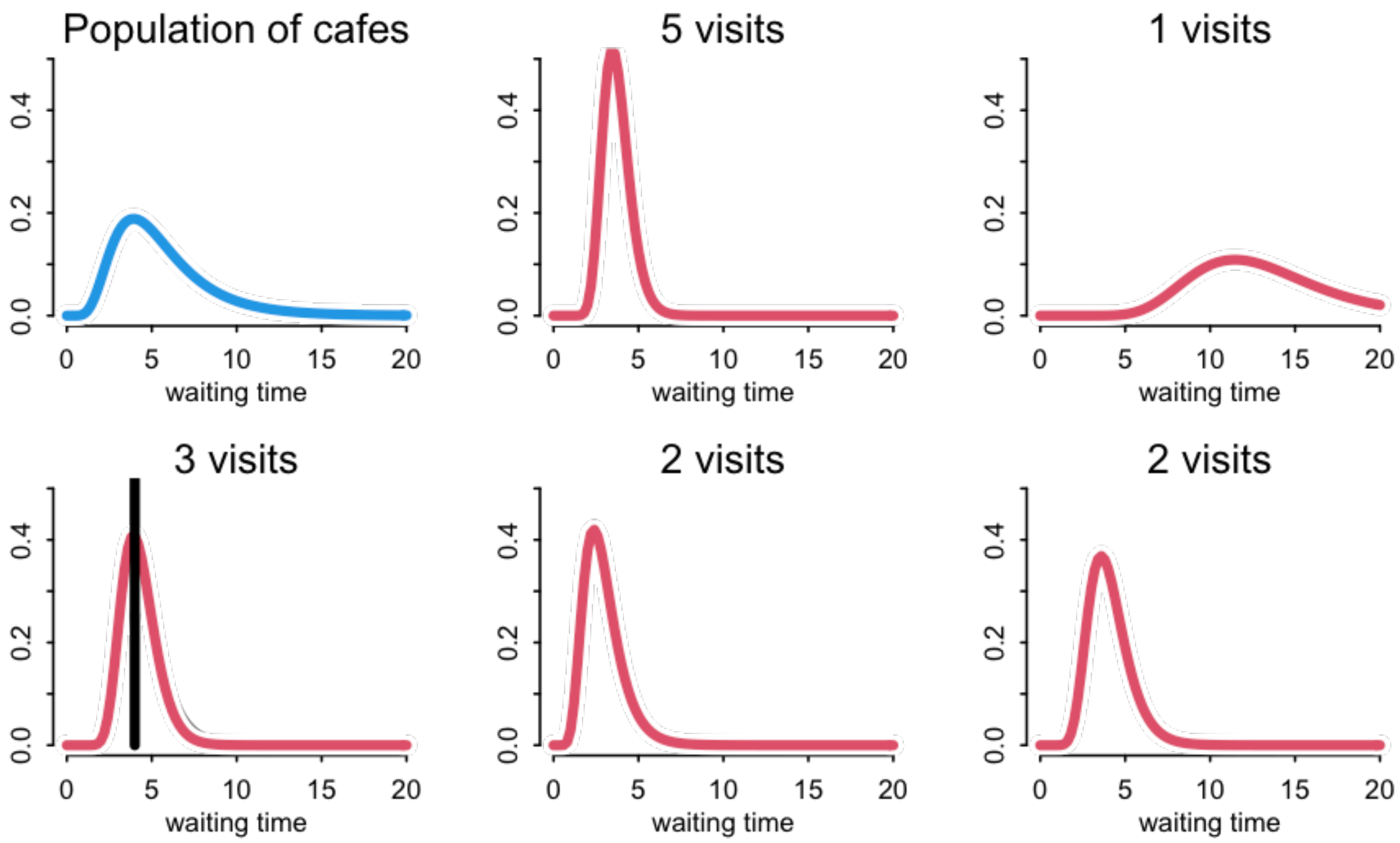
Population of cafes
4 visits
1 visits
2 visits
2 visits
1 visits
0.0
0.2
0.4
0
5
10
15
20
waiting time



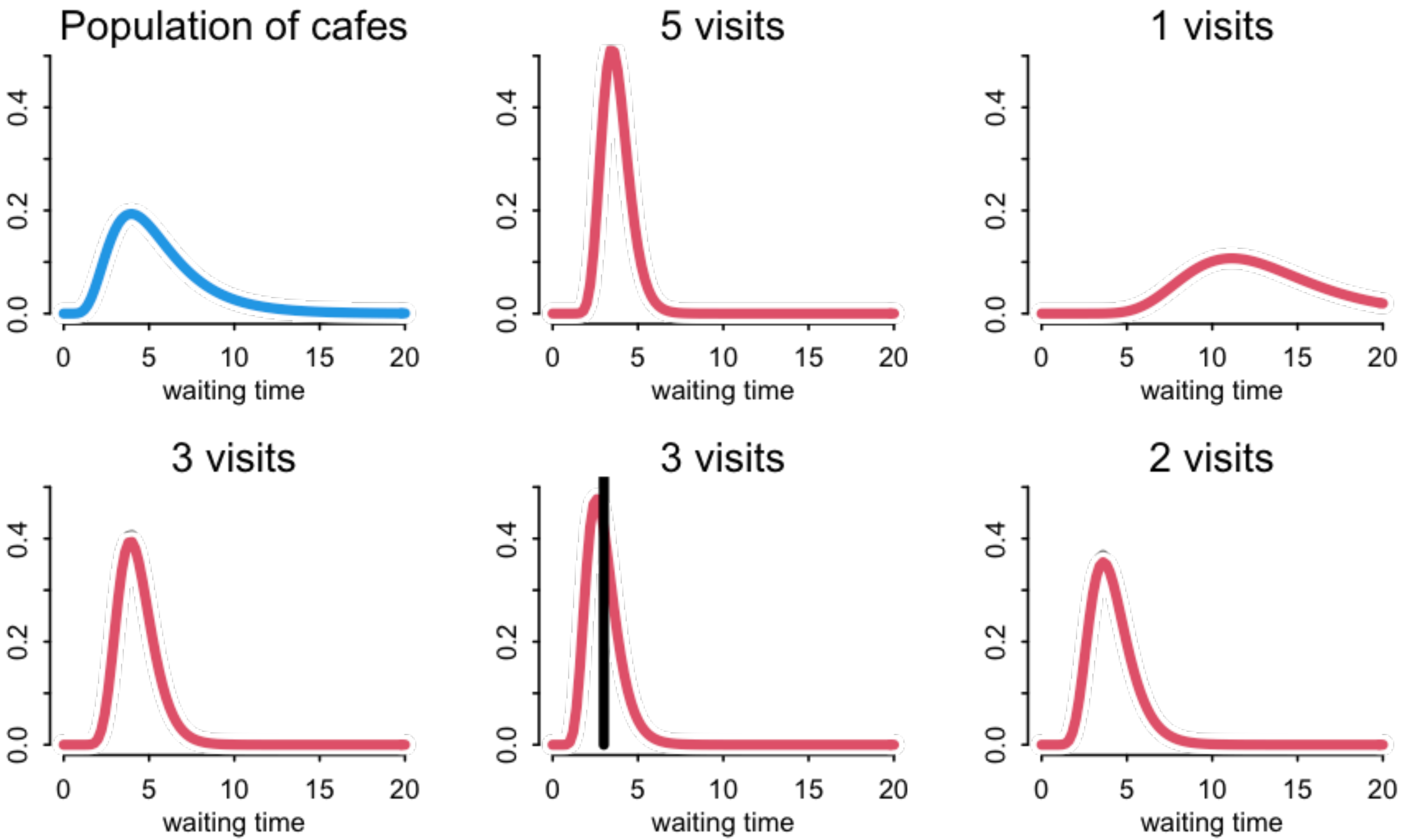
Population of cafes
4 visits
1 visits
2 visits
2 visits
2 visits
0.0
0.2
0.4
0
5
10
15
20
waiting time



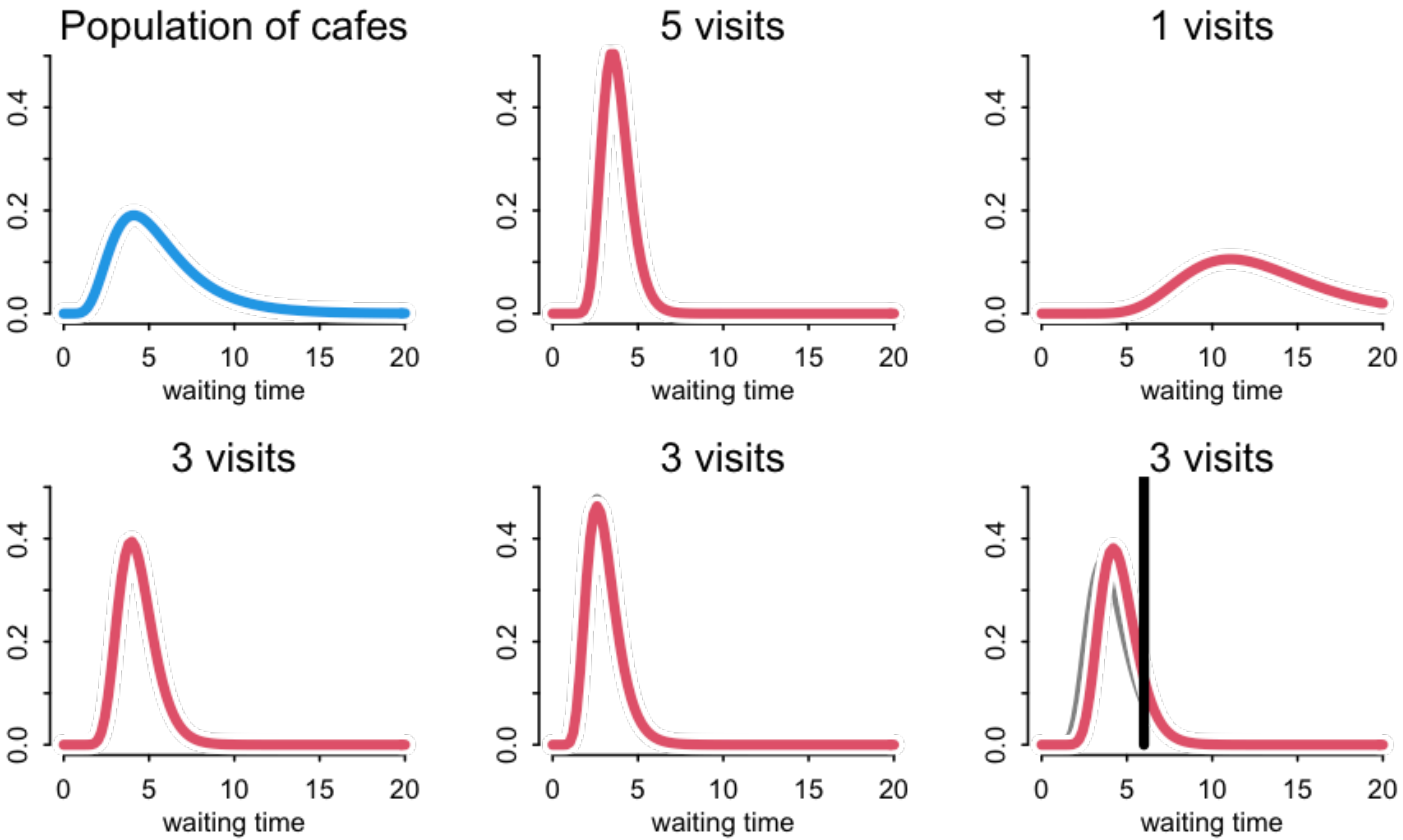
Population of cafes
5 visits
1 visits
2 visits
2 visits
2 visits
0.0
0.2
0.4
0
5
10
15
20
waiting time



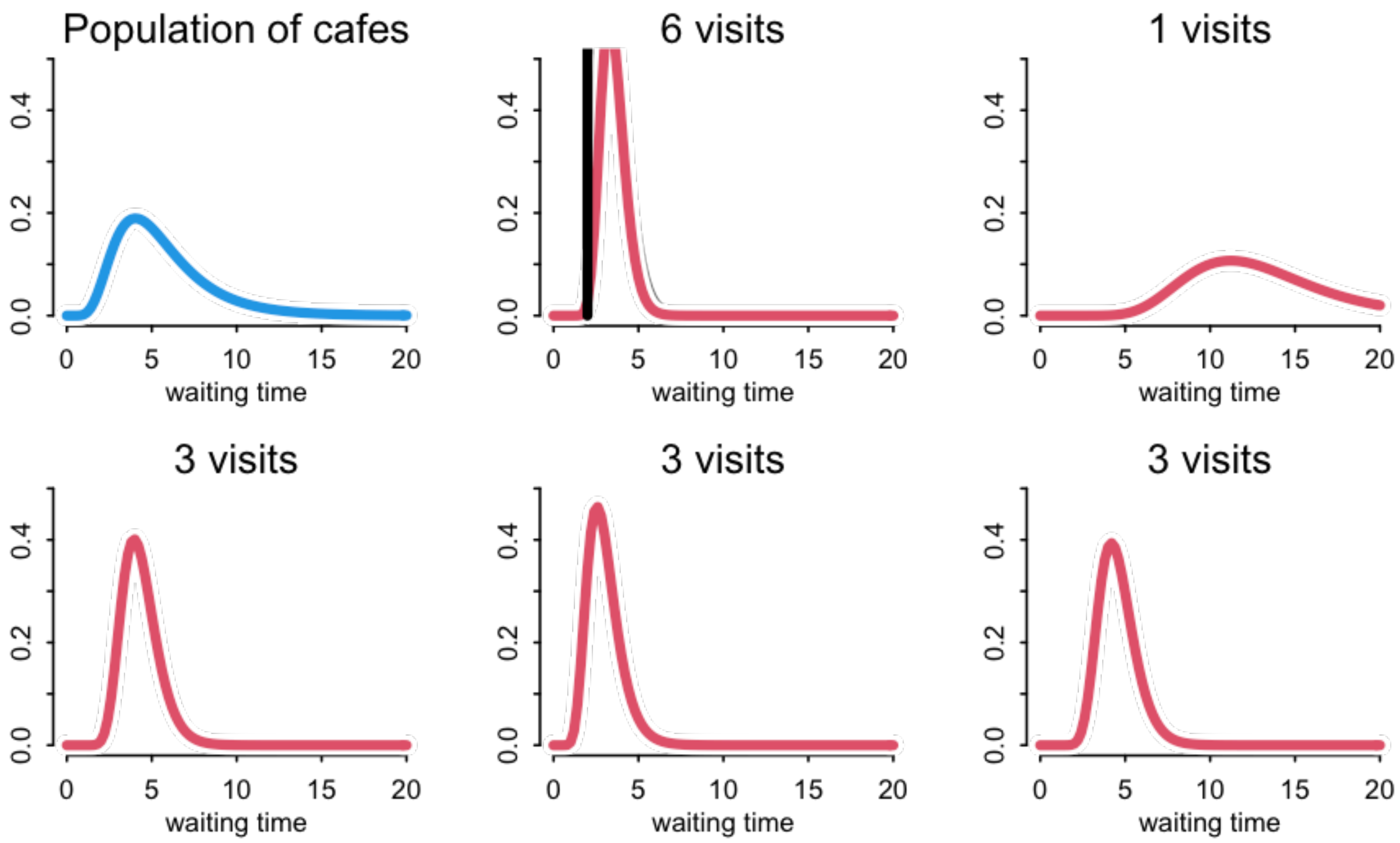
Population of cafes
5 visits
1 visits
3 visits
2 visits
2 visits
0.0
0.2
0.4
0
5
10
15
20
waiting time



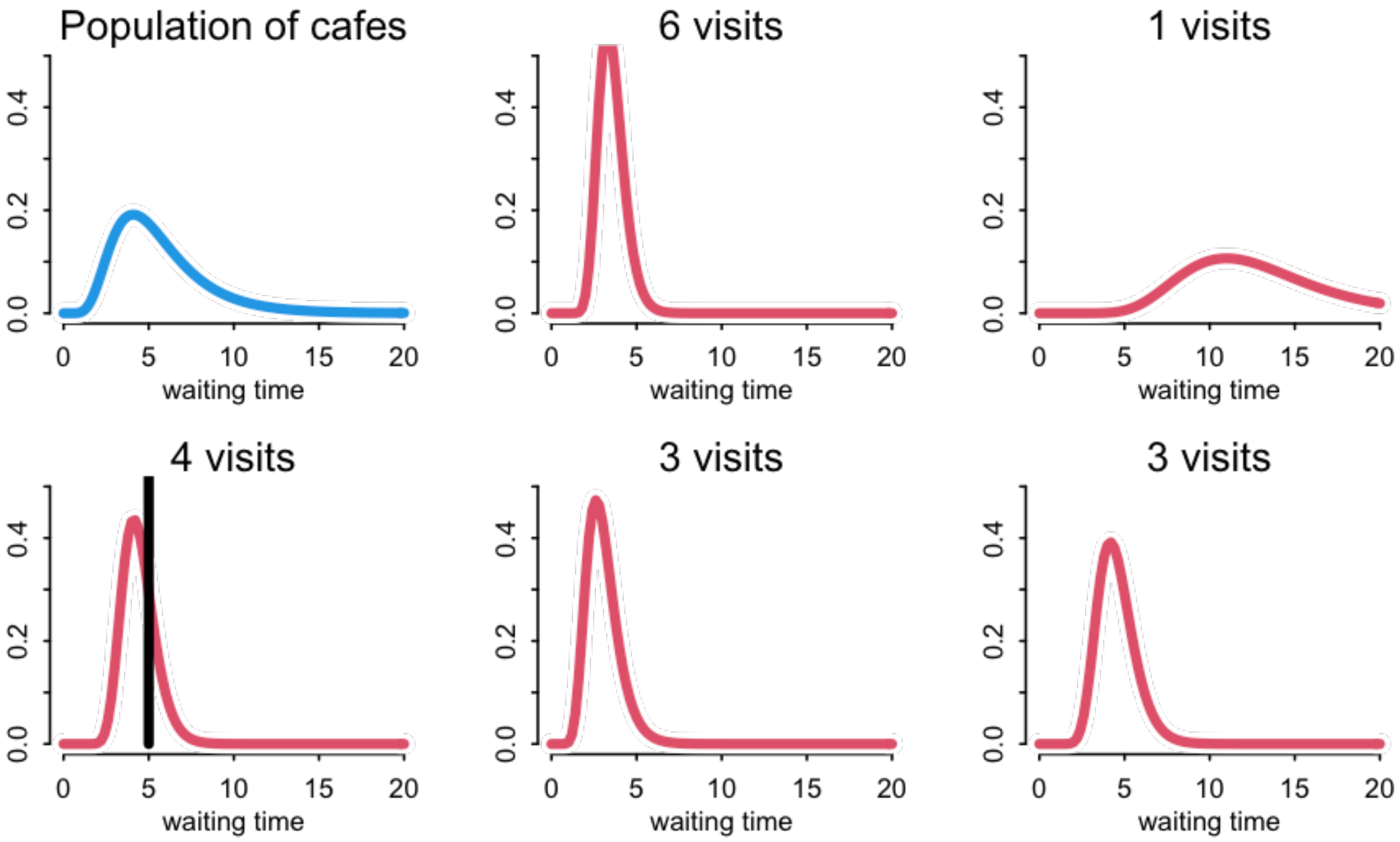
Population of cafes
5 visits
1 visits
3 visits
3 visits
2 visits
0.0
0.2
0.4
0
5
10
15
20
waiting time

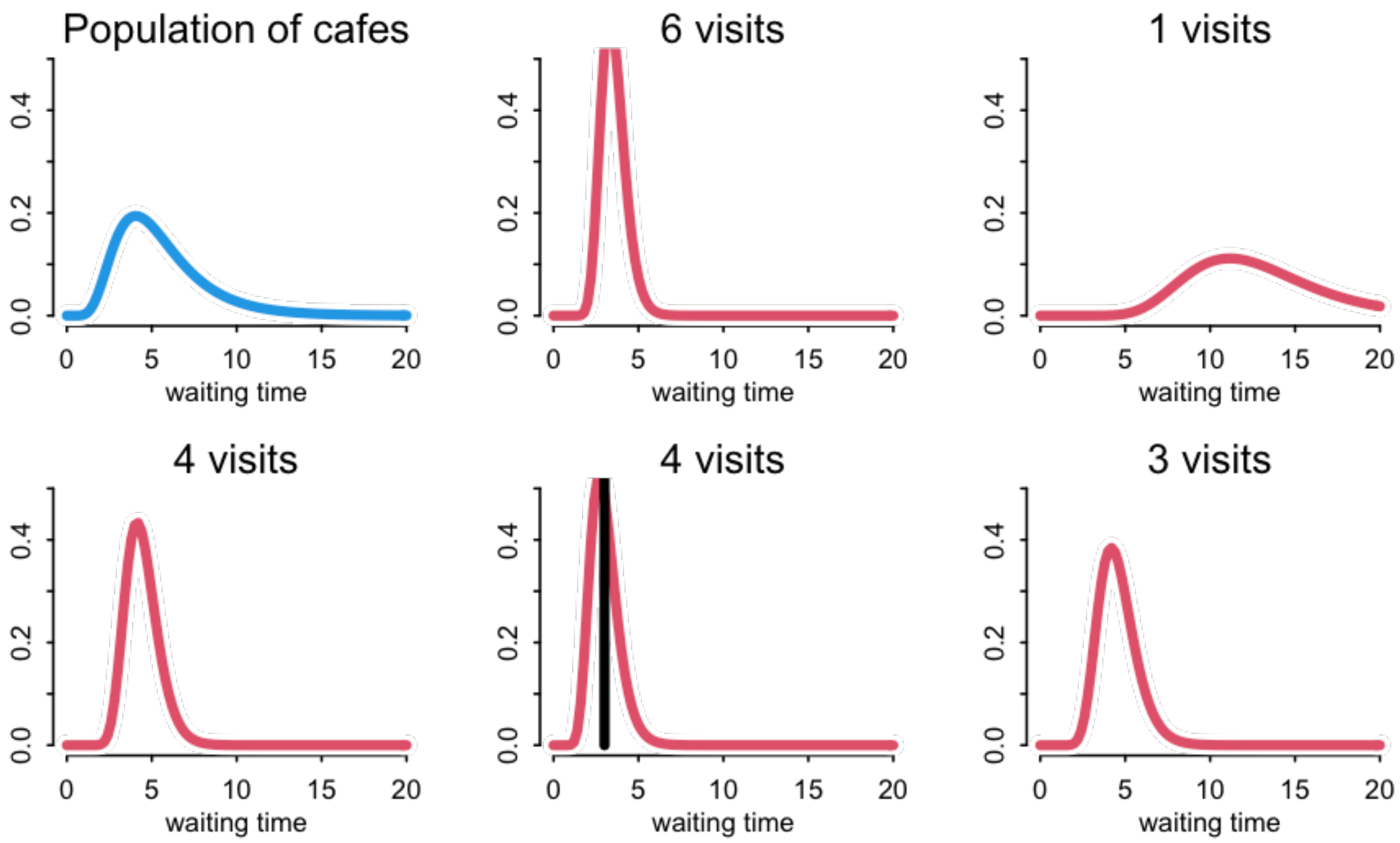


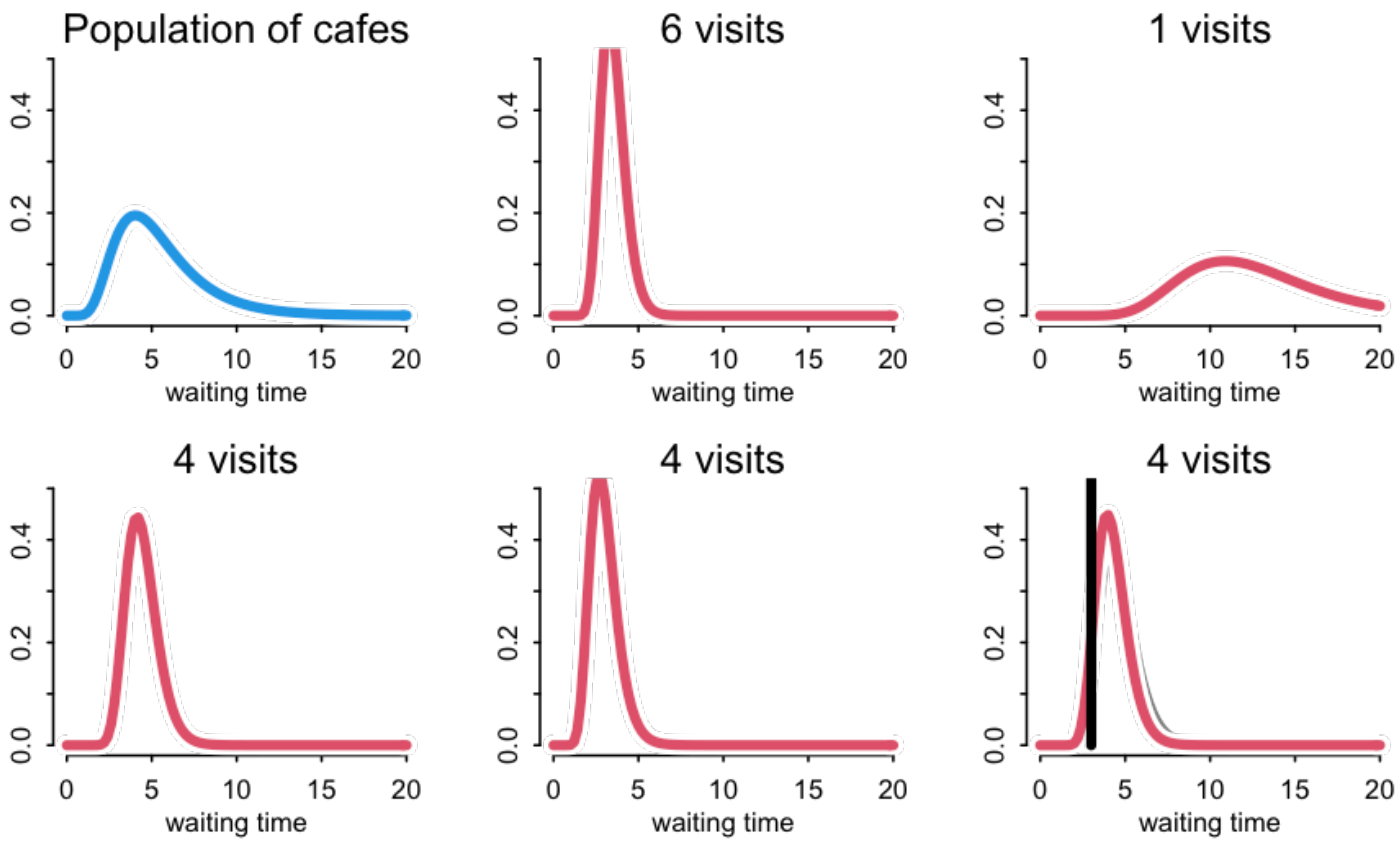
Population of cafes
5 visits
1 visits
3 visits
3 visits
3 visits
0.0
0.2
0.4
0
5
10
15
20
waiting time

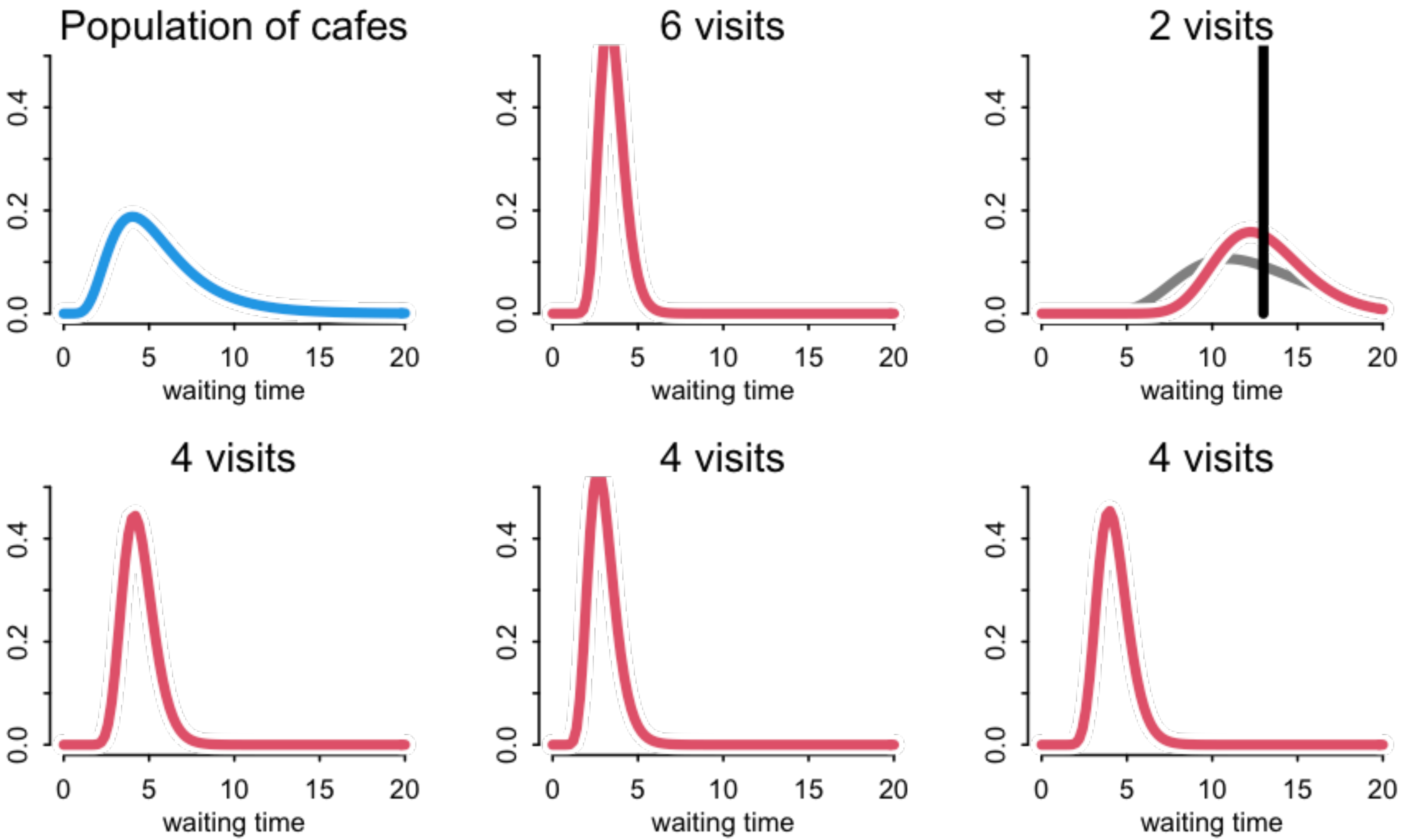


Population of cafes
6 visits
1 visits
3 visits
3 visits
3 visits
0.0
0.2
0.4
0
5
10
15
20
waiting time


Population of cafes
6 visits
1 visits
4 visits
3 visits
3 visits
0.0
0.2
0.4
0
5
10
15
20
waiting time


Population of cafes
6 visits
1 visits
4 visits
4 visits
3 visits
0.0
0.2
0.4
0
5
10
15
20
waiting time


Population of cafes
6 visits
1 visits
4 visits
4 visits
4 visits
0.0
0.2
0.4
0 5 10 15 20
waiting time


Population of cafes
6 visits
2 visits
4 visits
4 visits
4 visits
0.0
0.2
0.4
0
5
10
15
20
waiting time

Regularization

Another reason for multilevel models is that they adaptively regularize

Complete pooling: Treat all clusters as identical => underfitting

No pooling: Treat all clusters as unrelated => overfitting

Partial pooling: Adaptive compromise

Reedfrogs in peril

data(reedfrogs)

48 groups ("tanks") of tadpoles

Treatments: density, size, predation

Outcome: survival

Vonesh & Bolker (2005) Compensatory larval responses shift trade-offs associated with predator-induced hatching plasticity

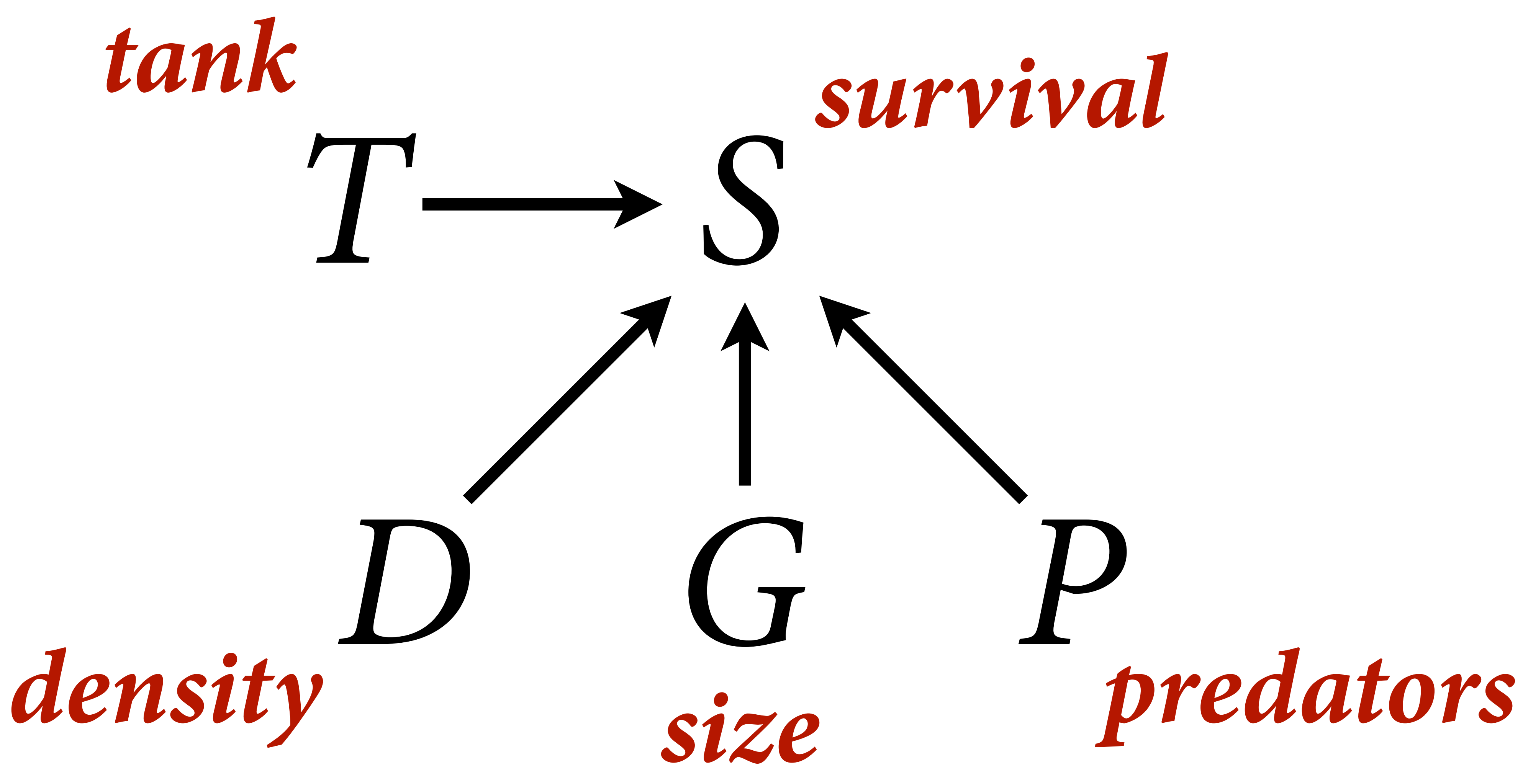
Reedfrogs in peril

data(reedfrogs)

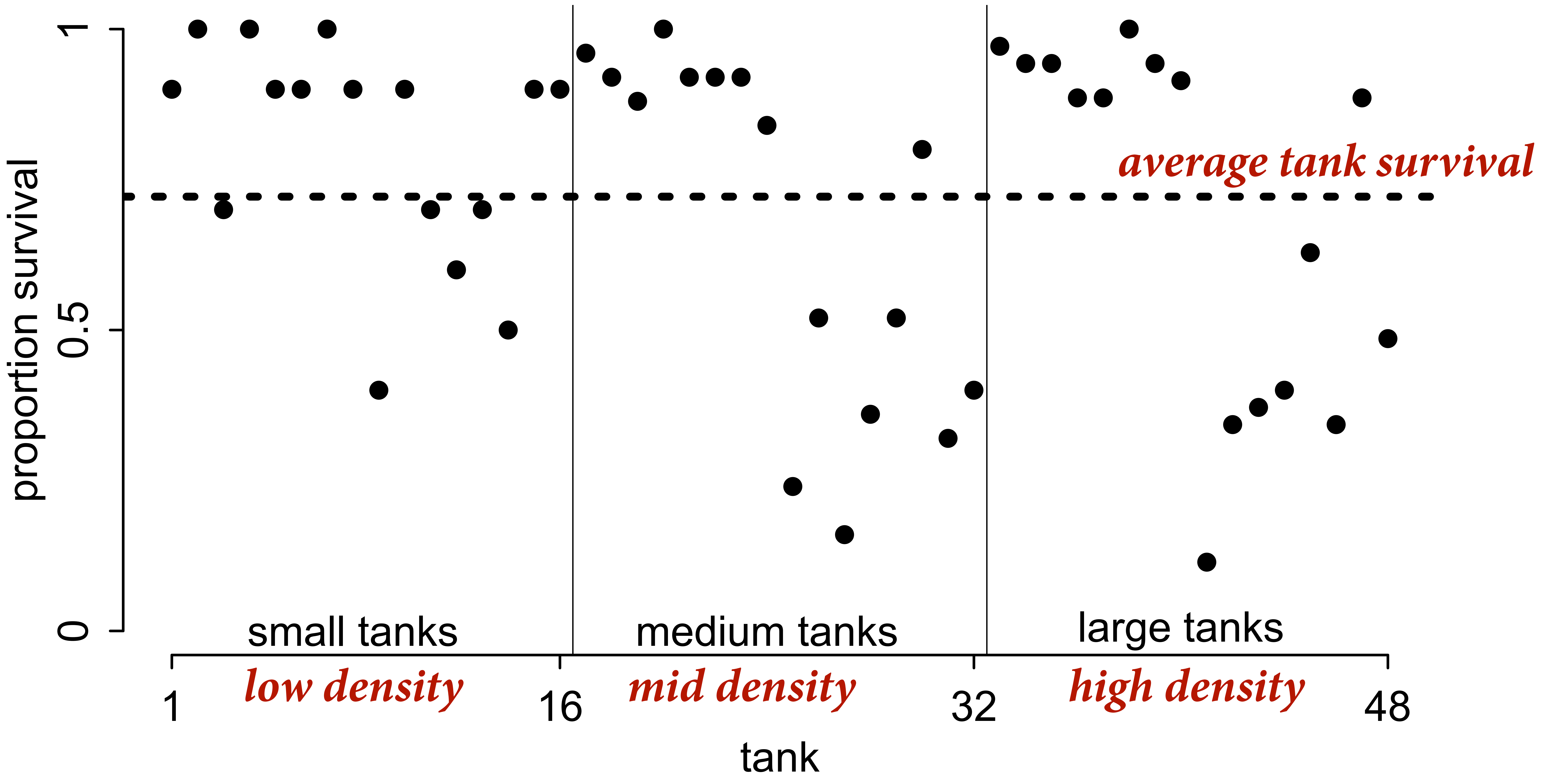
48 groups ("tanks") of tadpoles

Treatments: density, size, predation

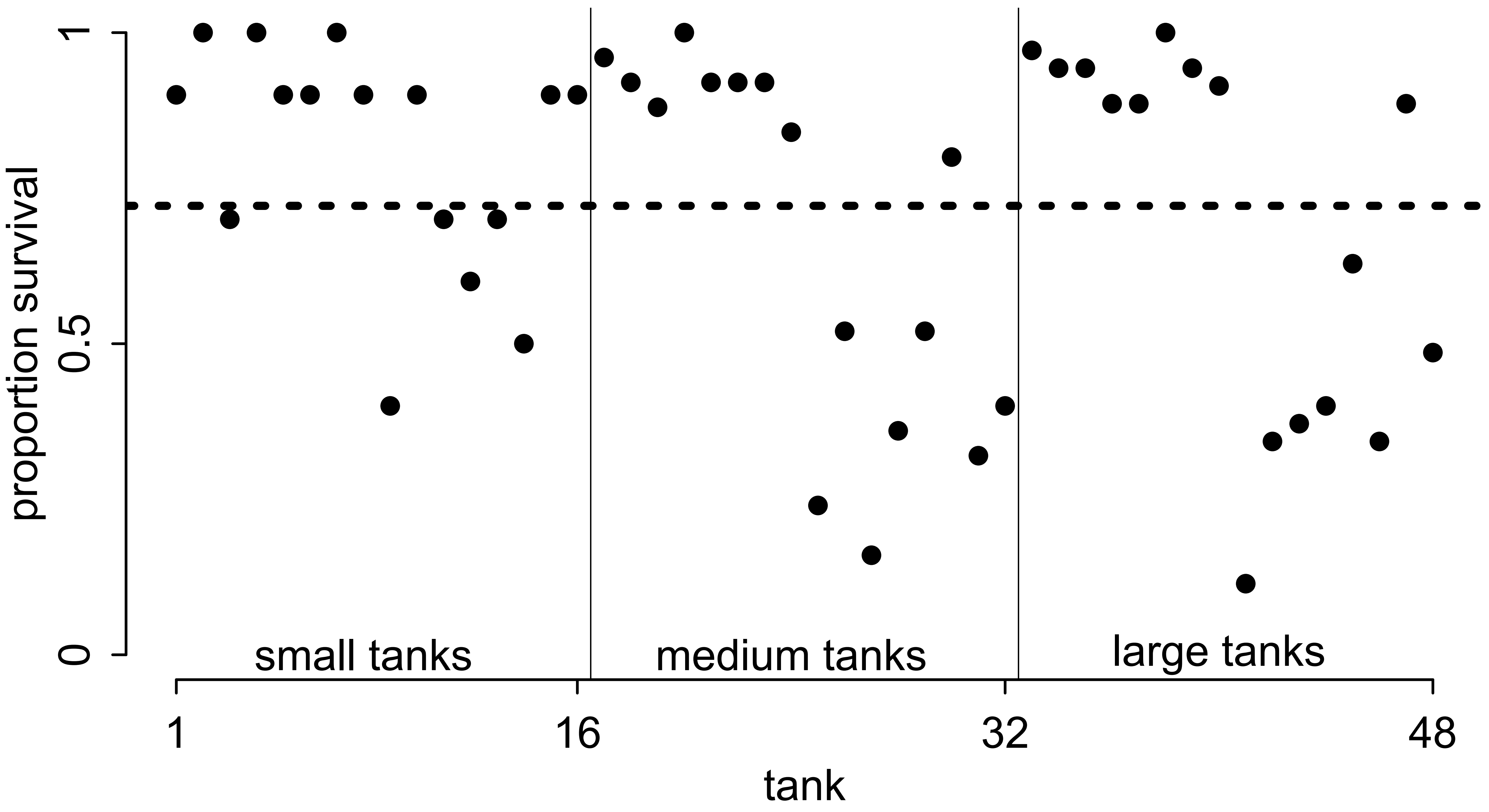
Outcome: survival



Vonesh & Bolker (2005) Compensatory larval responses shift trade-offs associated with predator-induced hatching plasticity



1
0.5
0
proportion survival
average tank survival
small tanks
medium tanks
large tanks
low density
mid density
high density
1
16
32
48
tank

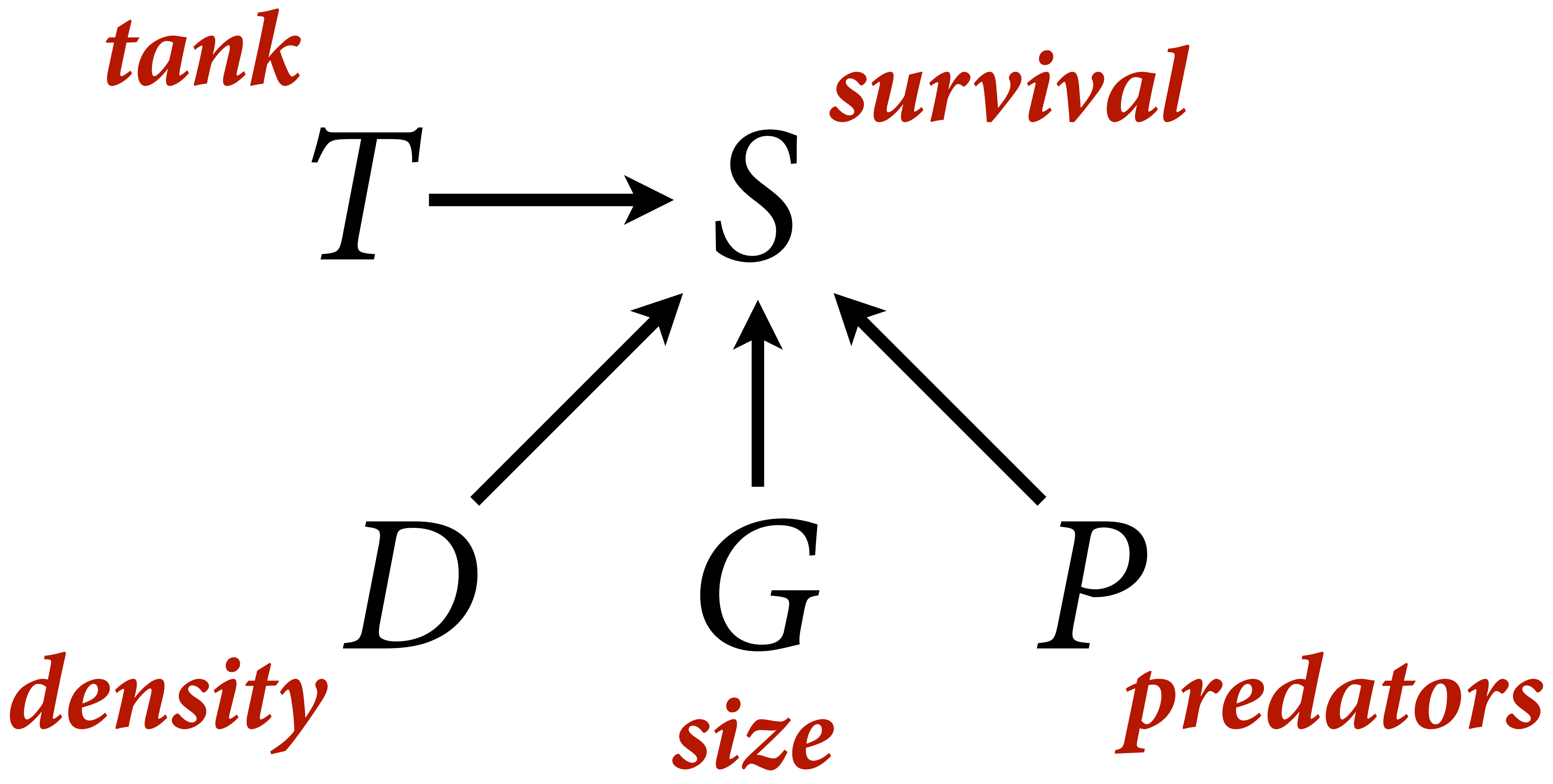


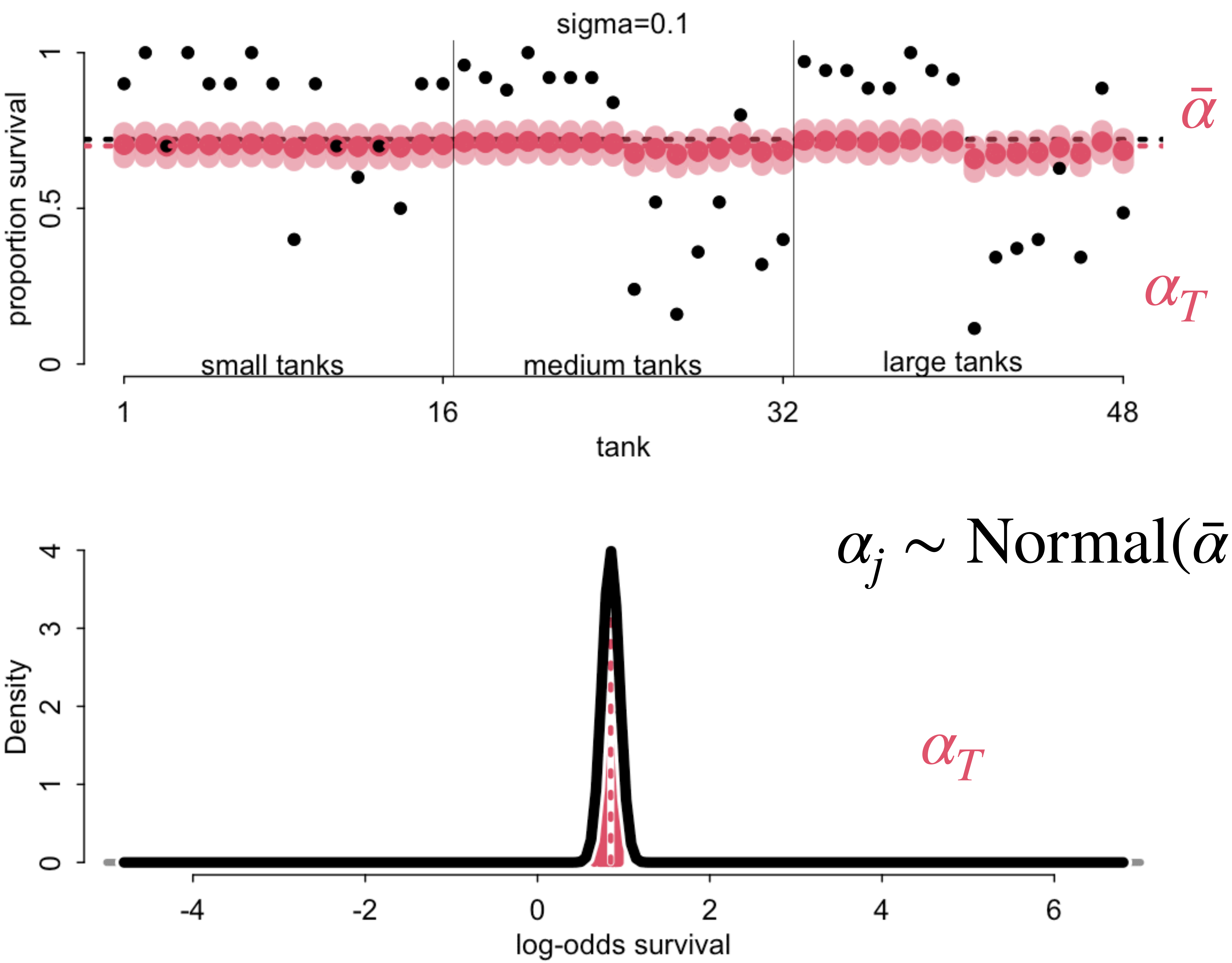
$$S_i \sim \text{Binomial}(D_i, p_i)$$

$$\text{logit}(p_i) = \alpha_{T[i]}$$

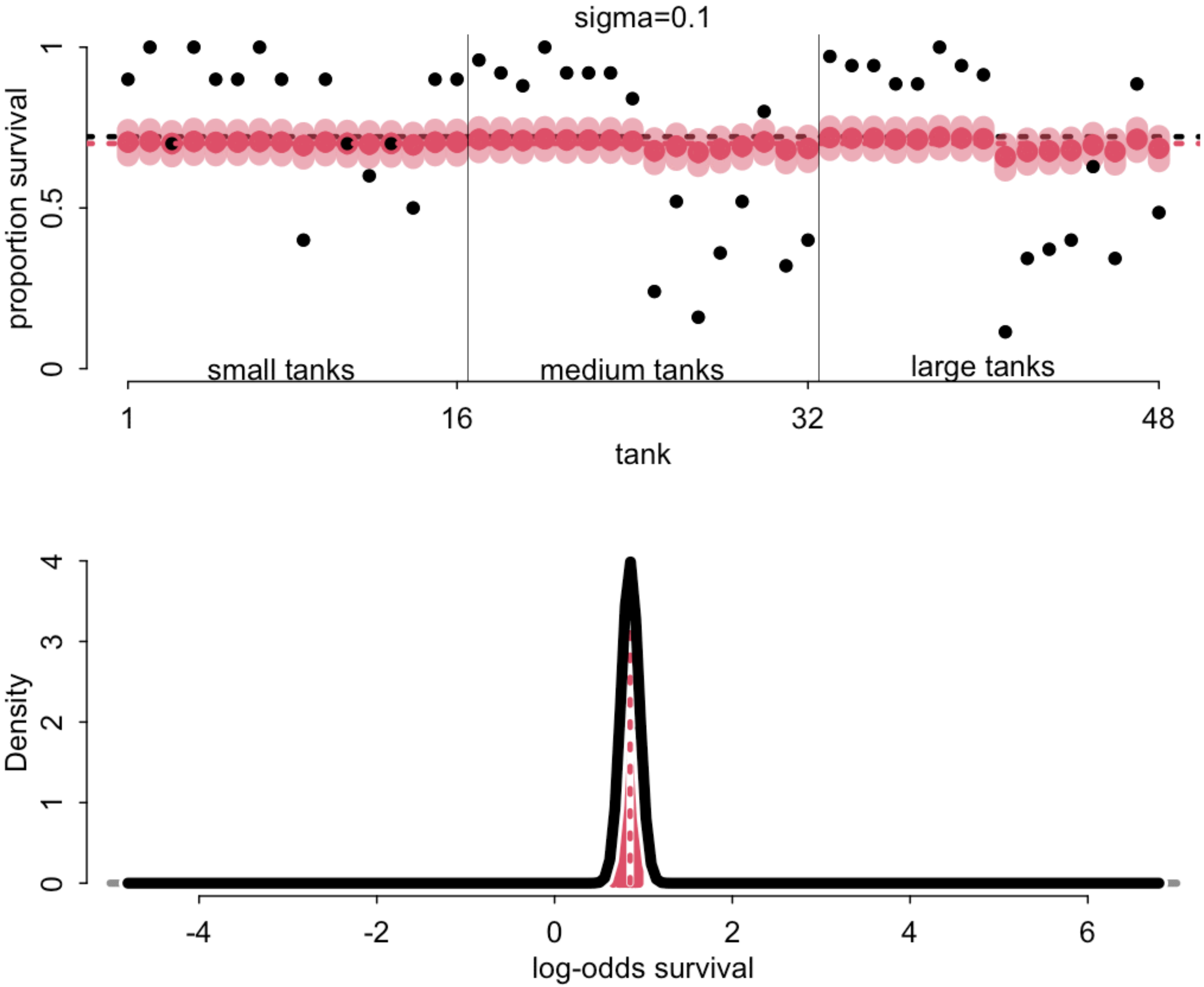
$$\alpha_j \sim \text{Normal}(\bar{\alpha}, ?)$$

$$\bar{\alpha} \sim \text{Normal}(0, 1.5)$$

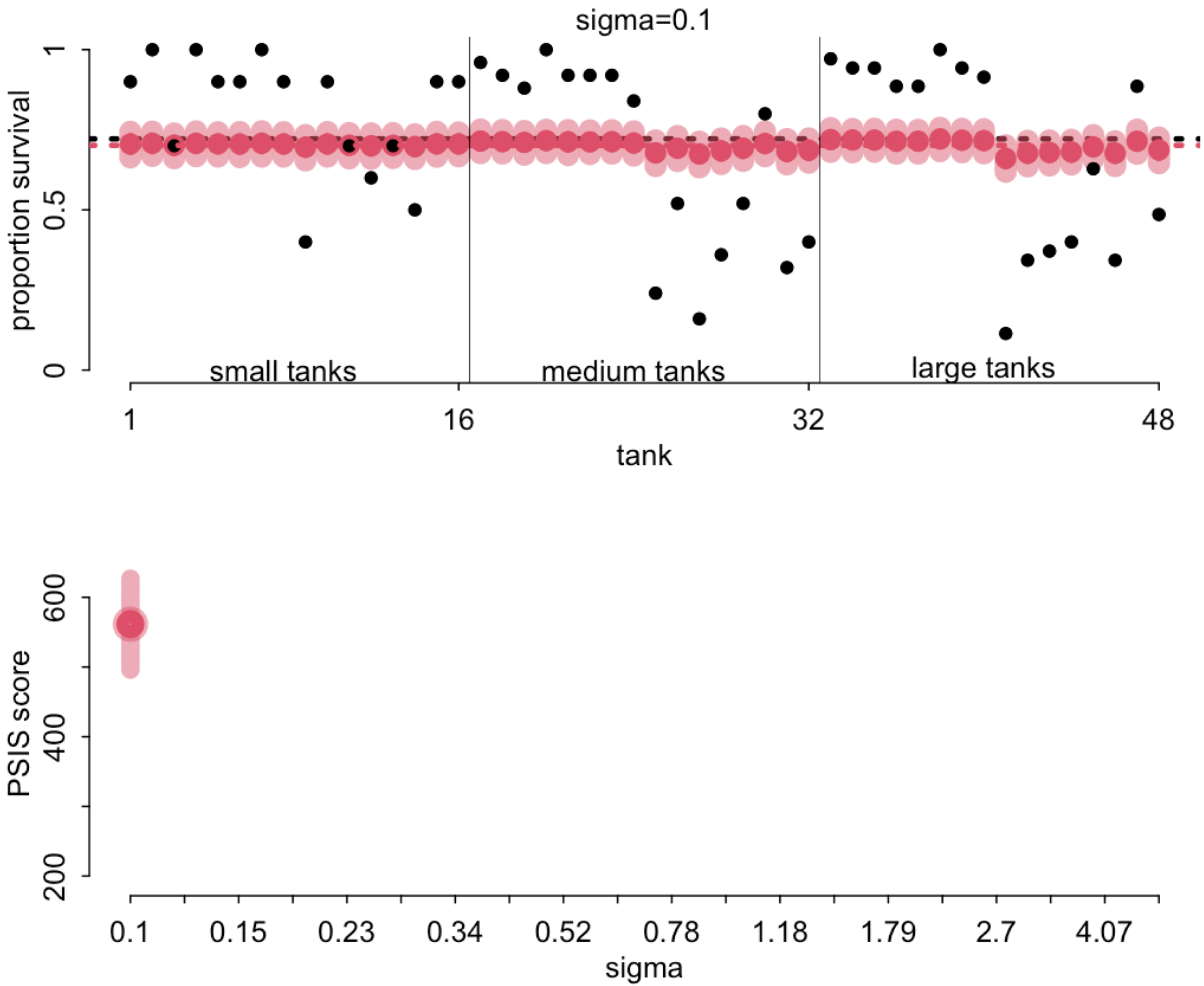




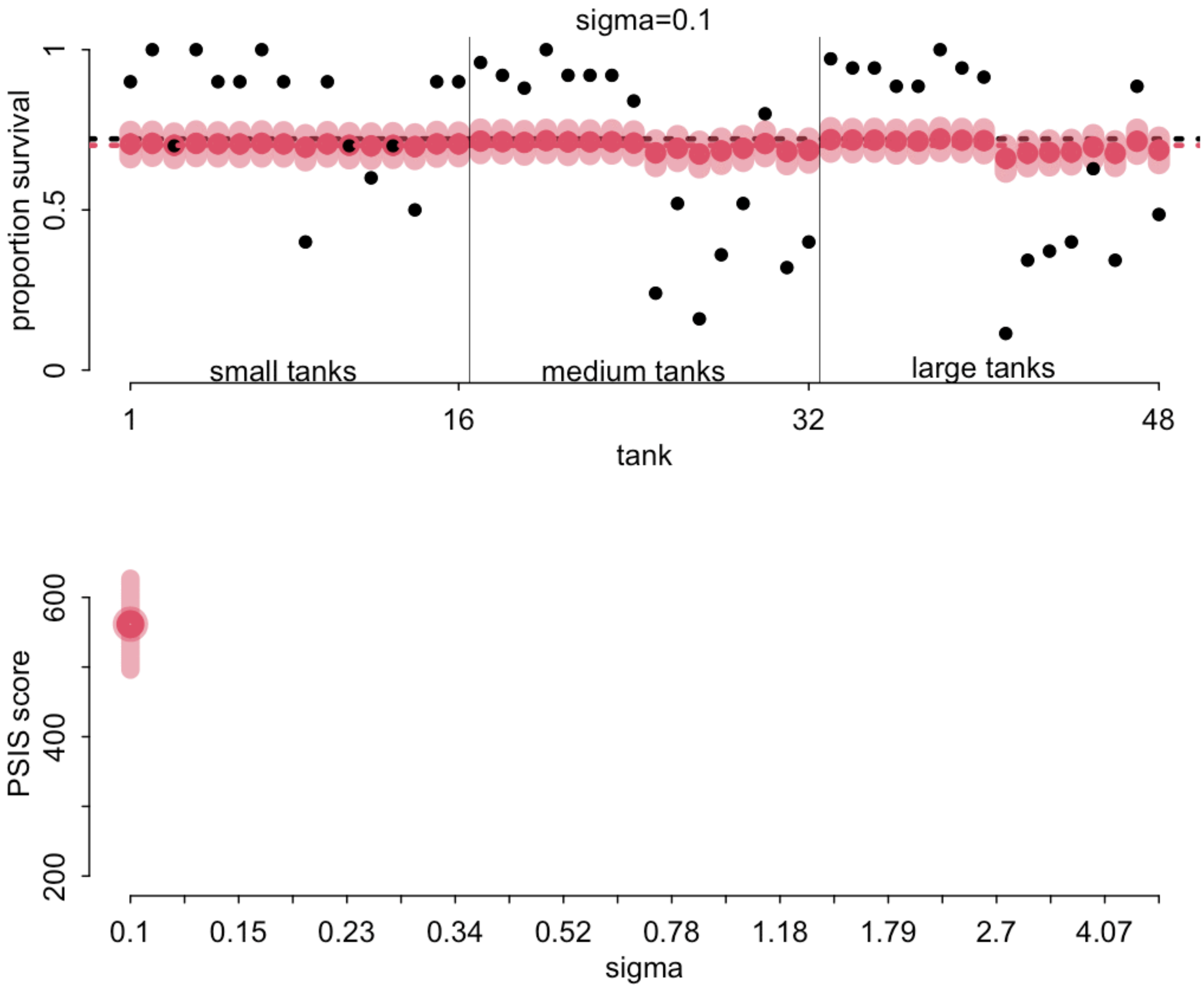
$$\alpha_j \sim \text{Normal}(\bar{\alpha}, \sigma)$$

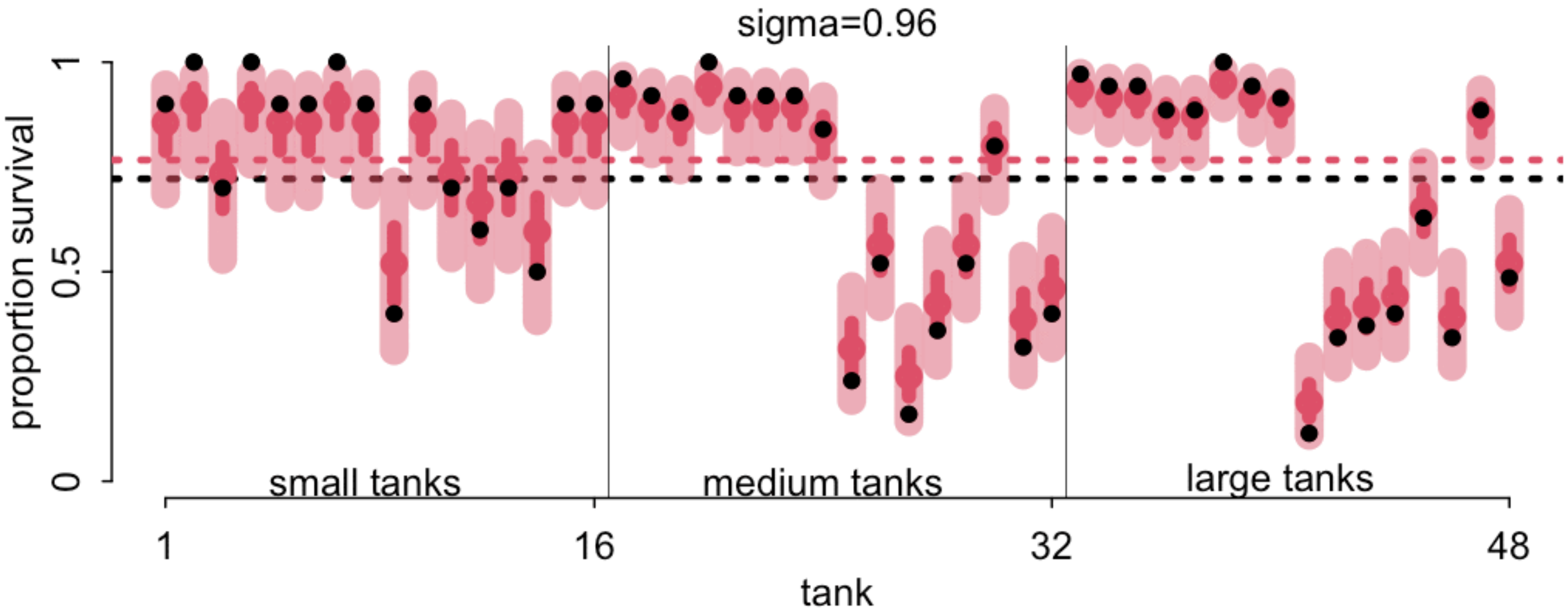


sigma=0.1
proportion survival
1
0.5
0
small tanks
medium tanks
large tanks
1
16
32
48
tank
Density
4
3
2
1
0
-4
-2
0
2
4
6
log-odds survival



sigma=0.1
proportion survival
0
0.5
1
small tanks
medium tanks
large tanks
1
16
32
48
tank
PSIS score
200
400
600
0.1
0.15
0.23
0.34
0.52
0.78
1.18
1.79
2.7
4.07
sigma


sigma=0.1
proportion survival
0
0.5
1
small tanks
medium tanks
large tanks
1
16
32
48
tank
PSIS score
200
400
600
0.1
0.15
0.23
0.34
0.52
0.78
1.18
1.79
2.7
4.07
sigma

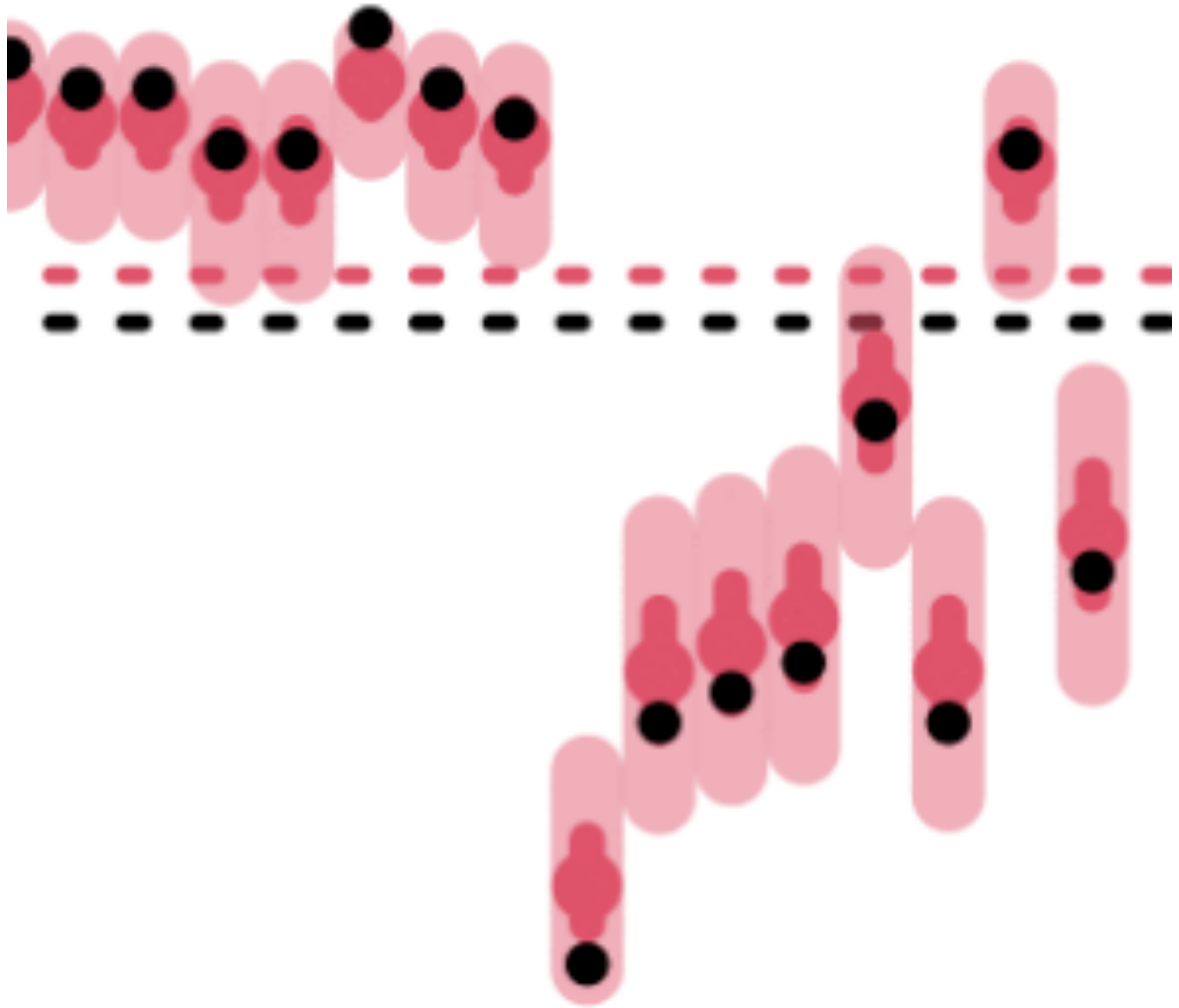

sigma=0.96
proportion survival
1
0.5
0
small tanks
medium tanks
large tanks
1
16
32
48
tank


600

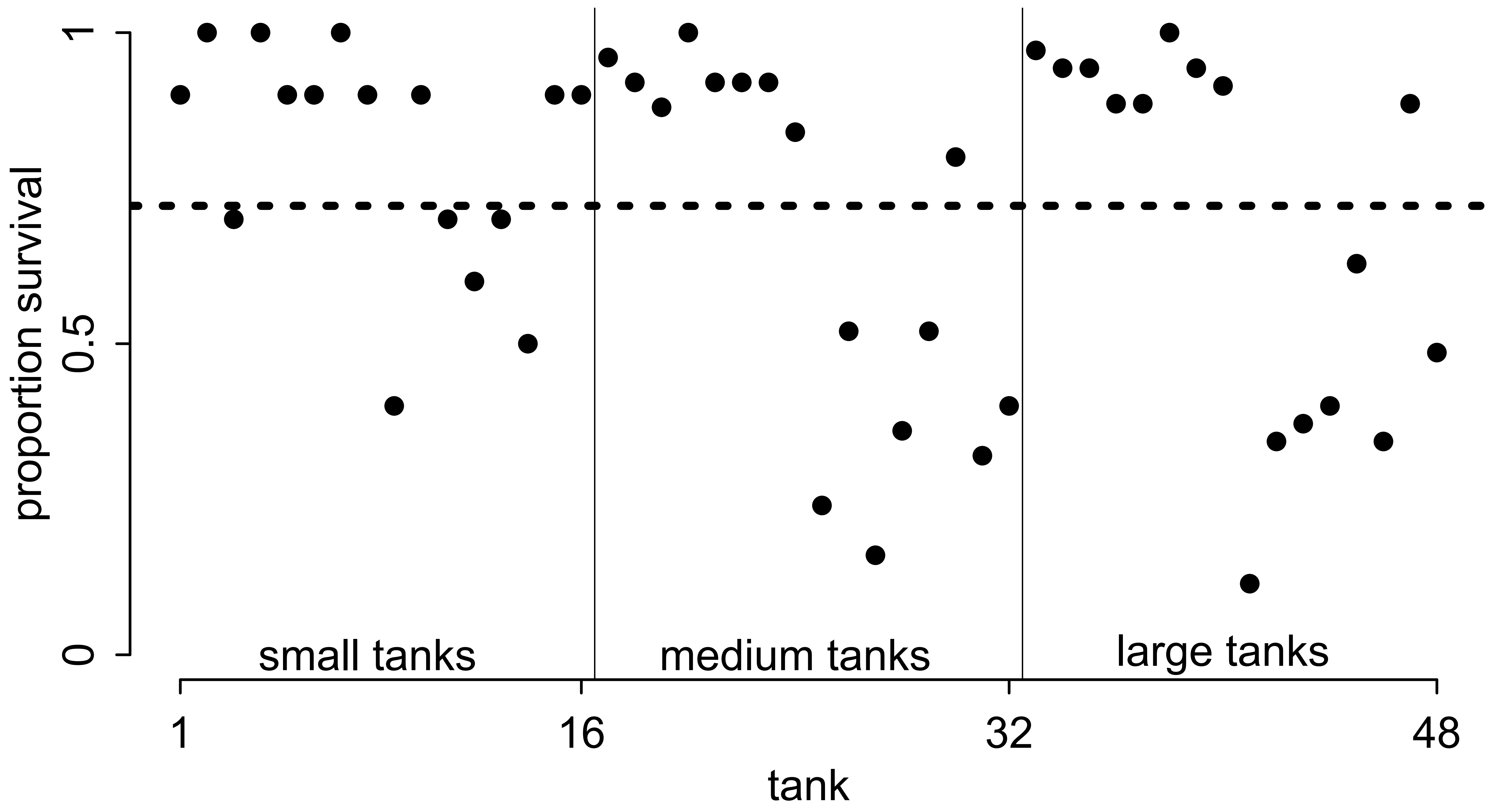
Automatic regularization

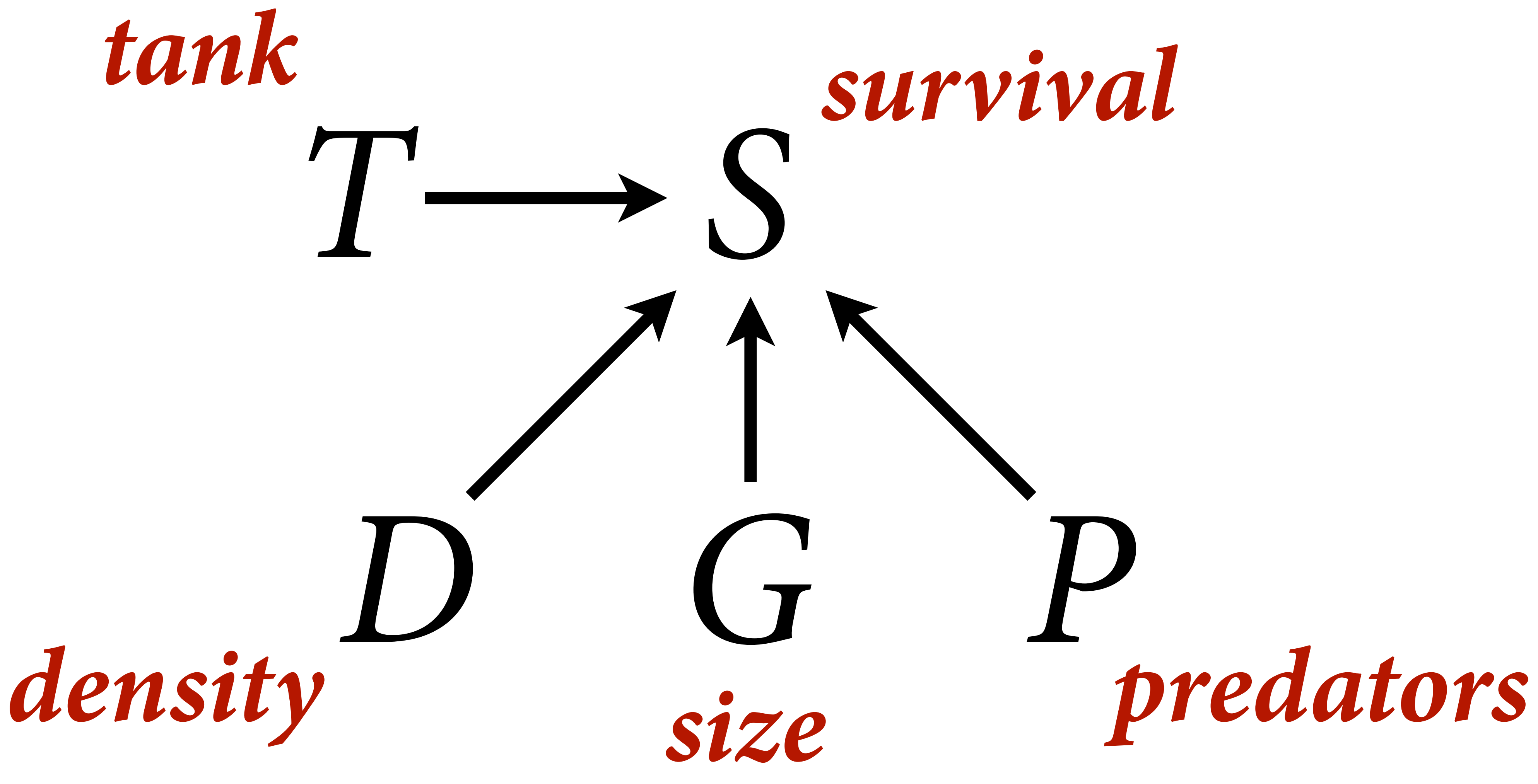
Wouldn't it be nice if we could find a good sigma without running so many models?

Maybe we could learn it from the data?



PAUSE





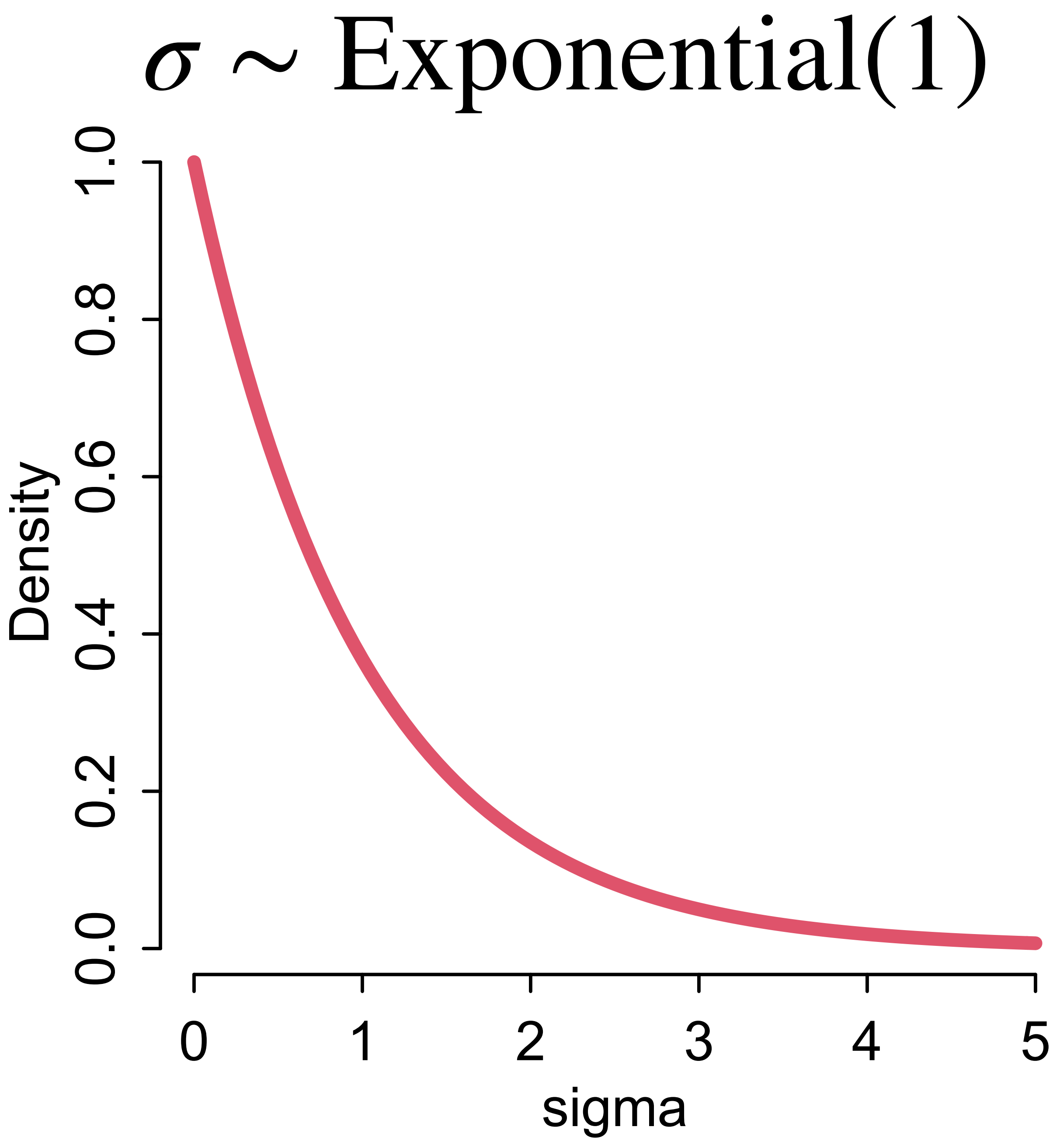
$$S_i \sim \text{Binomial}(D_i, p_i)$$

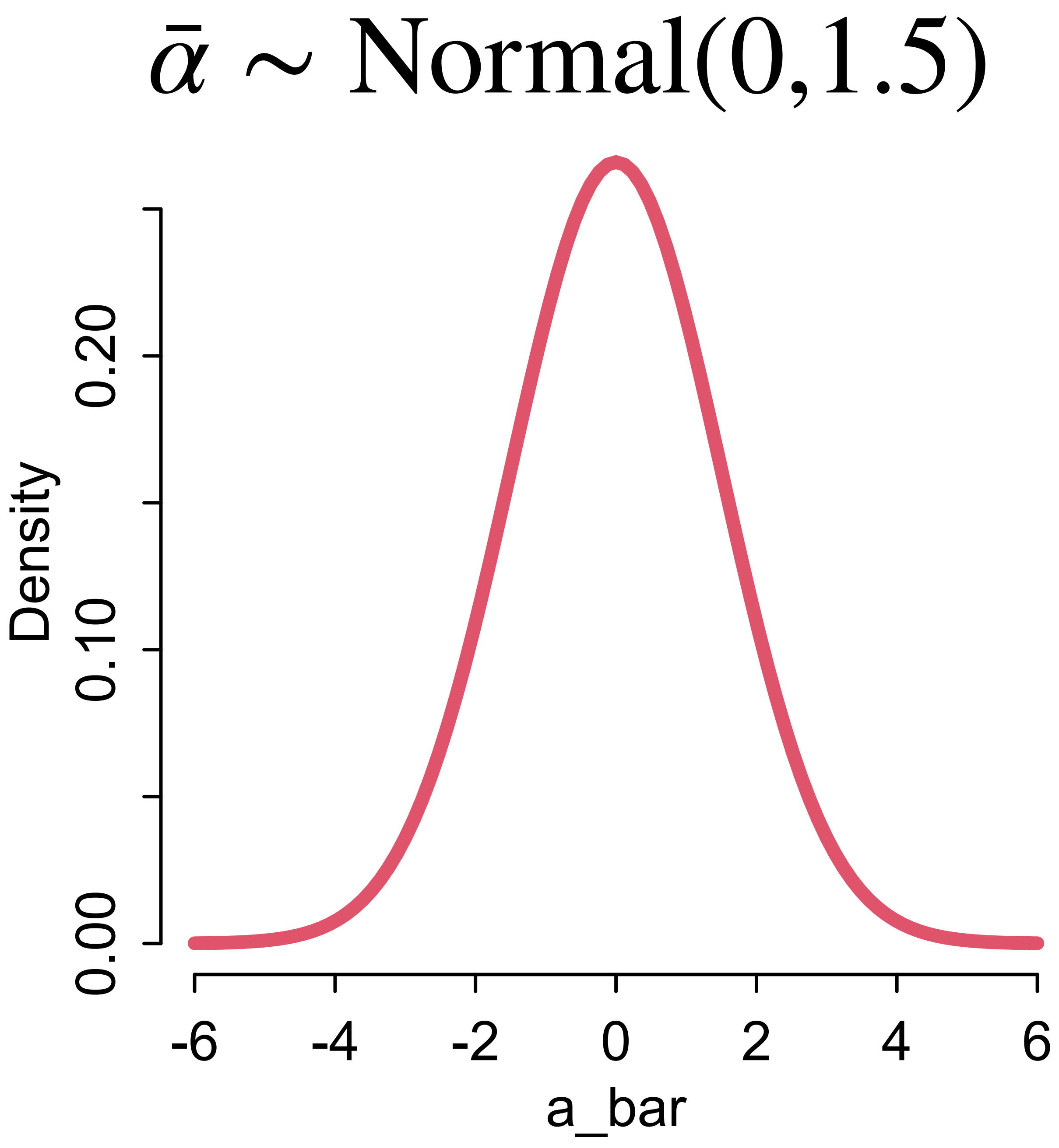
$$\text{logit}(p_i) = \alpha_{T[i]}$$

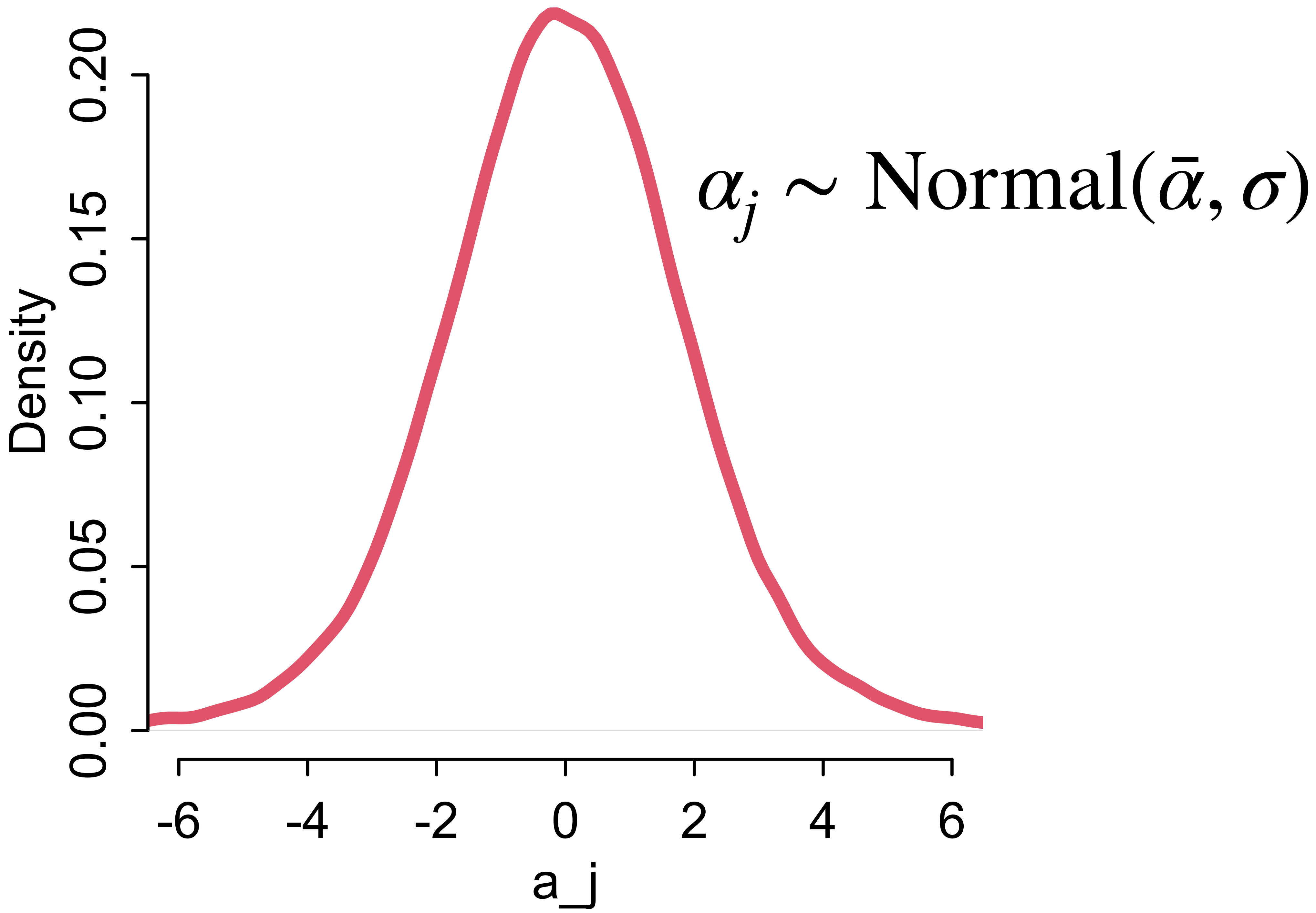
$$\alpha_j \sim \text{Normal}(\bar{\alpha}, \sigma)$$

$$\bar{\alpha} \sim \text{Normal}(0, 1.5)$$

$$\sigma \sim \text{Exponential}(1)$$







$$S_i \sim \text{Binomial}(D_i, p_i)$$

$$\text{logit}(p_i) = \alpha_{T[i]}$$

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

$$\bar{\alpha} \sim \text{Normal}(0, 1.5)$$

$$\sigma \sim \text{Exponential}(1)$$

```
library(rethinking)
data(reedfrogs)
d <- reedfrogs
d$tank <- 1:nrow(d)
dat <- list(
    S = d$surv,
    D = d$density,
    T = d$tank )

mST <- ulam(
    alist(

        S ~ dbinom( D , p ) ,
        logit(p) <- a[T] ,
        a[T] ~ dnorm( a_bar , sigma ) ,
        a_bar ~ dnorm( 0 , 1.5 ) ,
        sigma ~ dexp( 1 )

    ), data=dat , chains=4 , log_lik=TRUE )
```

$$S_i \sim \text{Binomial}(D_i, p_i)$$

$$\text{logit}(p_i) = \alpha_{T[i]}$$

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

$$\bar{\alpha} \sim \text{Normal}(0, 1.5)$$

$$\sigma \sim \text{Exponential}(1)$$

```r
library(rethinking)
data(reedfrogs)
d <- reedfrogs
d$tank <- 1:nrow(d)
dat <- list(
    S = d$surv,
    D = d$density,
    T = d$tank )

mST <- ulam(
    alist(

        S ~ dbinom( D , p ) ,
        logit(p) <- a[T] ,
        a[T] ~ dnorm( a_bar , sigma ) ,
        a_bar ~ dnorm( 0 , 1.5 ) ,
        sigma ~ dexp( 1 )

    ), data=dat , chains=4 , log_lik=TRUE )
```

```
> precis(mST,depth=2)
       mean   sd  5.5% 94.5% n_eff Rhat4
a[1]   2.13 0.85  0.89  3.54  2992     1
a[2]   3.06 1.04  1.57  4.82  2716     1
a[3]   1.01 0.67 -0.01  2.11  5635     1
a[4]   3.08 1.07  1.53  4.88  2441     1
a[5]   2.14 0.87  0.85  3.61  3460     1
a[6]   2.11 0.85  0.88  3.61  3628     1
a[7]   3.05 1.08  1.54  4.90  3603     1
a[8]   2.14 0.89  0.83  3.69  3190     1
a[9]  -0.17 0.64 -1.20  0.88  5424     1
a[10]  2.15 0.90  0.83  3.72  2559     1
a[11]  1.00 0.66 -0.03  2.09  3265     1
a[12]  0.57 0.63 -0.44  1.60  6602     1
a[13]  1.01 0.67 -0.02  2.13  3618     1
a[14]  0.21 0.62 -0.75  1.21  4147     1
a[15]  2.10 0.85  0.84  3.51  4563     1
a[16]  2.12 0.85  0.89  3.58  3030     1
a[17]  2.88 0.77  1.82  4.22  3888     1
a[18]  2.38 0.65  1.42  3.46  3645     1
a[19]  2.01 0.58  1.16  2.95  4029     1
a[20]  3.65 1.04  2.17  5.47  2750     1
a[21]  2.39 0.65  1.43  3.47  3585     1
a[22]  2.39 0.66  1.41  3.51  3607     1
a[23]  2.40 0.66  1.45  3.49  3312     1
a[24]  1.71 0.53  0.92  2.58  3395     1
a[25] -0.99 0.43 -1.69 -0.32  3187     1
a[26]  0.16 0.39 -0.47  0.80  4611     1
a[27] -1.43 0.49 -2.23 -0.69  3289     1
a[28] -0.47 0.41 -1.15  0.15  5525     1
a[29]  0.17 0.40 -0.46  0.82  5628     1
a[30]  1.44 0.50  0.68  2.26  4925     1
a[31] -0.62 0.41 -1.29  0.03  5449     1
a[32] -0.30 0.39 -0.95  0.32  4039     1
a[33]  3.19 0.80  2.06  4.60  2357     1
a[34]  2.71 0.63  1.79  3.80  3117     1
a[35]  2.71 0.62  1.82  3.71  3185     1
a[36]  2.07 0.53  1.28  2.95  3616     1
a[37]  2.05 0.47  1.34  2.82  4705     1
a[38]  3.87 0.91  2.56  5.42  3022     1
a[39]  2.71 0.66  1.76  3.80  3479     1
a[40]  2.37 0.59  1.47  3.38  2981     1
a[41] -1.79 0.45 -2.54 -1.13  4340     1
a[42] -0.58 0.35 -1.14  0.00  4354     1
a[43] -0.45 0.36 -1.01  0.10  5360     1
a[44] -0.33 0.34 -0.89  0.22  4541     1
a[45]  0.58 0.35  0.02  1.15  5803     1
a[46] -0.56 0.37 -1.14  0.02  4696     1
a[47]  2.06 0.51  1.27  2.91  3955     1
a[48]  0.01 0.35 -0.55  0.56  5353     1
a_bar  1.35 0.26  0.93  1.78  2746     1
sigma  1.61 0.21  1.32  1.96  1557     1
```

$$\sim \text{Binomial}(D_i, p_i)$$

$$= \alpha_{T[i]}$$

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

$$\sim \text{Normal}(0, 1.5)$$

$$\sim \text{Exponential}(1)$$

```r
library(rethinking)
data(reedfrogs)
d <- reedfrogs
d$tank <- 1:nrow(d)
dat <- list(
    S = d$surv,
    D = d$density,
    T = d$tank )

mST <- ulam(
    alist(

        S ~ dbinom( D , p ) ,
        logit(p) <- a[T] ,
        a[T] ~ dnorm( a_bar , sigma ) ,
        a_bar ~ dnorm( 0 , 1.5 ) ,
        sigma ~ dexp( 1 )

    ), data=dat , chains=4 , log_lik=TRUE )
```

```
> precis(mST,depth=2)
```

	mean	sd	5.5%	94.5%	n_eff	Rhat4
a[1]	2.13	0.85	0.89	3.54	2992	1
a[2]	3.06	1.04	1.57	4.82	2716	1
a[3]	1.01	0.67	-0.01	2.11	5635	1
a[4]	3.08	1.07	1.53	4.88	2441	1
a[5]	2.14	0.87	0.85	3.61	3460	1
a[6]	2.11	0.85	0.88	3.61	3628	1
a[7]	3.05	1.08	1.54	4.90	3603	1
a[8]	2.14	0.89	0.83	3.69	3190	1
a[9]	-0.17	0.64	-1.20	0.88	5424	1
a[10]	2.15	0.90	0.83	3.72	2559	1
a[11]	1.00	0.66	-0.03	2.09	3265	1
a[12]	0.57	0.63	-0.44	1.60	6602	1
a[13]	1.01	0.67	-0.02	2.13	3618	1
a[14]	0.21	0.62	-0.75	1.21	4147	1
a[15]	2.10	0.85	0.84	3.51	4563	1
a[16]	2.12	0.85	0.89	3.58	3030	1
a[17]	2.88	0.77	1.82	4.22	3888	1
a[18]	2.38	0.65	1.42	3.46	3645	1
a[19]	2.01	0.58	1.16	2.95	4029	1
a[20]	3.65	1.04	2.17	5.47	2750	1
a[29]	0.17	0.40	-0.46	0.82	5628	1
a[30]	1.44	0.50	0.68	2.26	4925	1
a[31]	-0.62	0.41	-1.29	0.03	5449	1
a[32]	-0.30	0.39	-0.95	0.32	4039	1
a[33]	3.19	0.80	2.06	4.60	2357	1
a[34]	2.71	0.63	1.79	3.80	3117	1
a[35]	2.71	0.62	1.82	3.71	3185	1
a[36]	2.07	0.53	1.28	2.95	3616	1
a[37]	2.05	0.47	1.34	2.82	4705	1
a[38]	3.87	0.91	2.56	5.42	3022	1
a[39]	2.71	0.66	1.76	3.80	3479	1
a[40]	2.37	0.59	1.47	3.38	2981	1
a[41]	-1.79	0.45	-2.54	-1.13	4340	1
a[42]	-0.58	0.35	-1.14	0.00	4354	1
a[43]	-0.45	0.36	-1.01	0.10	5360	1
a[44]	-0.33	0.34	-0.89	0.22	4541	1
a[45]	0.58	0.35	0.02	1.15	5803	1
a[46]	-0.56	0.37	-1.14	0.02	4696	1
a[47]	2.06	0.51	1.27	2.91	3955	1
a[48]	0.01	0.35	-0.55	0.56	5353	1
a_bar	1.35	0.26	0.93	1.78	2746	1
sigma	1.61	0.21	1.32	1.96	1557	1

	mean	sd	5.5%	94.5%	n_eff	Rhat4
a_bar	1.35	0.26	0.93	1.78	2746	1
sigma	1.61	0.21	1.32	1.96	1557	1

$\sim \text{Binomial}(D_i, p_i)$

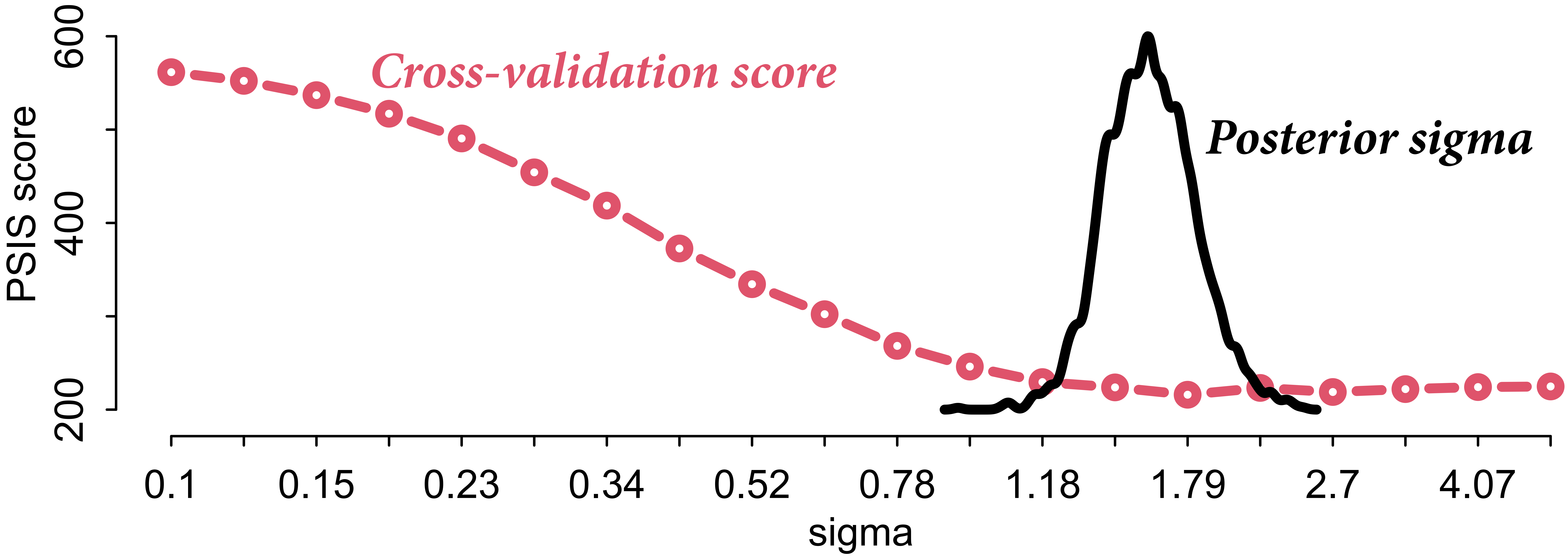
$= \alpha_{T[i]}$

$(\bar{\alpha}, \sigma)$

$\sim \text{Normal}(0,1.5)$

$\sim \text{Exponential}(1)$

	mean	sd	5.5%	94.5%	n_eff	Rhat4
a_bar	1.35	0.26	0.93	1.78	2746	1
sigma	1.61	0.21	1.32	1.96	1557	1



```
mST <- ulam(
    alist(

        S ~ dbinom( D , p ) ,
        logit(p) <- a[T] ,
        a[T] ~ dnorm( a_bar , sigma ) ,
        a_bar ~ dnorm( 0 , 1.5 ) ,
        sigma ~ dexp( 1 )

    ), data=dat , chains=4 , log_lik=TRUE )
```

$$S_i \sim \text{Binomial}(D_i, p_i)$$
$$\text{logit}(p_i) = \alpha_{T[i]}$$
$$\alpha_j \sim \text{Normal}(\bar{\alpha}, \sigma)$$
$$\bar{\alpha} \sim \text{Normal}(0, 1.5)$$
$$\sigma \sim \text{Exponential}(1)$$

```
mSTnomem <- ulam(
    alist(

        S ~ dbinom( D , p ) ,
        logit(p) <- a[T] ,
        a[T] ~ dnorm( a_bar , 1 ) ,
        a_bar ~ dnorm( 0 , 1.5 )

    ), data=dat , chains=4 , log_lik=TRUE )

compare( mST , mSTnomem , func=WAIC )
```

$$S_i \sim \text{Binomial}(D_i, p_i)$$
$$\text{logit}(p_i) = \alpha_{T[i]}$$
$$\alpha_j \sim \text{Normal}(\bar{\alpha}, 1)$$
$$\bar{\alpha} \sim \text{Normal}(0, 1.5)$$

```
mST <- ulam(
    alist(

        S ~ dbinom( D , p ) ,
        logit(p) <- a[T] ,
        a[T] ~ dnorm( a_bar , sigma ) ,
        a_bar ~ dnorm( 0 , 1.5 ) ,
        sigma ~ dexp( 1 )

    ), data=dat , ch
```

```
> compare( mST , mSTnomem , func=WAIC )
          WAIC   SE dWAIC  dSE pWAIC weight
mST      200.6 7.52   0.0   NA  21.1      1
mSTnomem 217.4 7.80  16.8 4.35  25.6      0
```

$$S_i \sim \text{Binomial}(D_i, p_i)$$
$$\text{logit}(p_i) = \alpha_{T[i]}$$
$$\alpha_j \sim \text{Normal}(\bar{\alpha}, \sigma)$$
$$\bar{\alpha} \sim \text{Normal}(0, 1.5)$$
$$\sigma \sim \text{Exponential}(1)$$

```
mSTnomem <- ulam(
    alist(

        S ~ dbinom( D , p ) ,
        logit(p) <- a[T] ,
        a[T] ~ dnorm( a_bar , 1 ) ,
        a_bar ~ dnorm( 0 , 1.5 )

    ), data=dat , chains=4 , log_lik=TRUE )

compare( mST , mSTnomem , func=WAIC )
```

inomial(D_i, p_i)
$$\text{logit}(p_i) = \alpha_{T[i]}$$
$$\alpha_j \sim \text{Normal}(\bar{\alpha}, 1)$$
$$\bar{\alpha} \sim \text{Normal}(0, 1.5)$$

```
mST <- ulam(
    alist(

        S ~ dbinom( D , p ) ,
        logit(p) <- a[T] ,
        a[T] ~ dnorm( a_bar , sigma ) ,
        a_bar ~ dnorm( 0 , 1.5 ) ,
        sigma ~ dexp( 1 )

    ), data=dat , ch
```

$$S_i \sim \text{Binomial}(D_i, p_i)$$
$$\text{logit}(p_i) = \alpha_{T[i]}$$
$$\alpha_j \sim \text{Normal}(\bar{\alpha}, \sigma)$$
$$\bar{\alpha} \sim \text{Normal}(0, 1.5)$$
$$\sigma \sim \text{Exponential}(1)$$

```
> compare( mST , mSTnomem , func=WAIC )
```

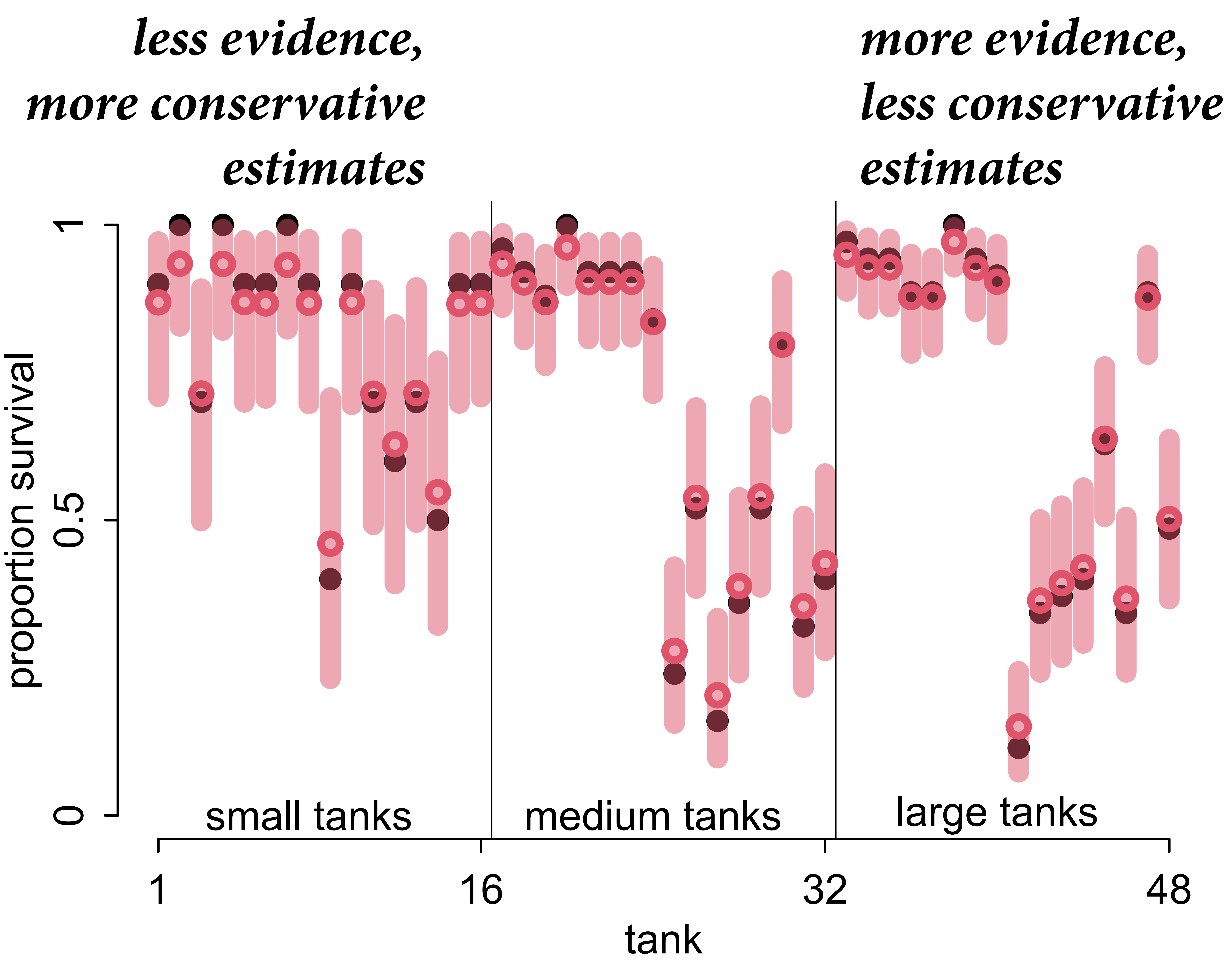
	WAIC	SE	dWAIC	dSE	pWAIC	weight
mST	200.6	7.52	0.0	NA	21.1	1
mSTnomem	217.4	7.80	16.8	4.35	25.6	0

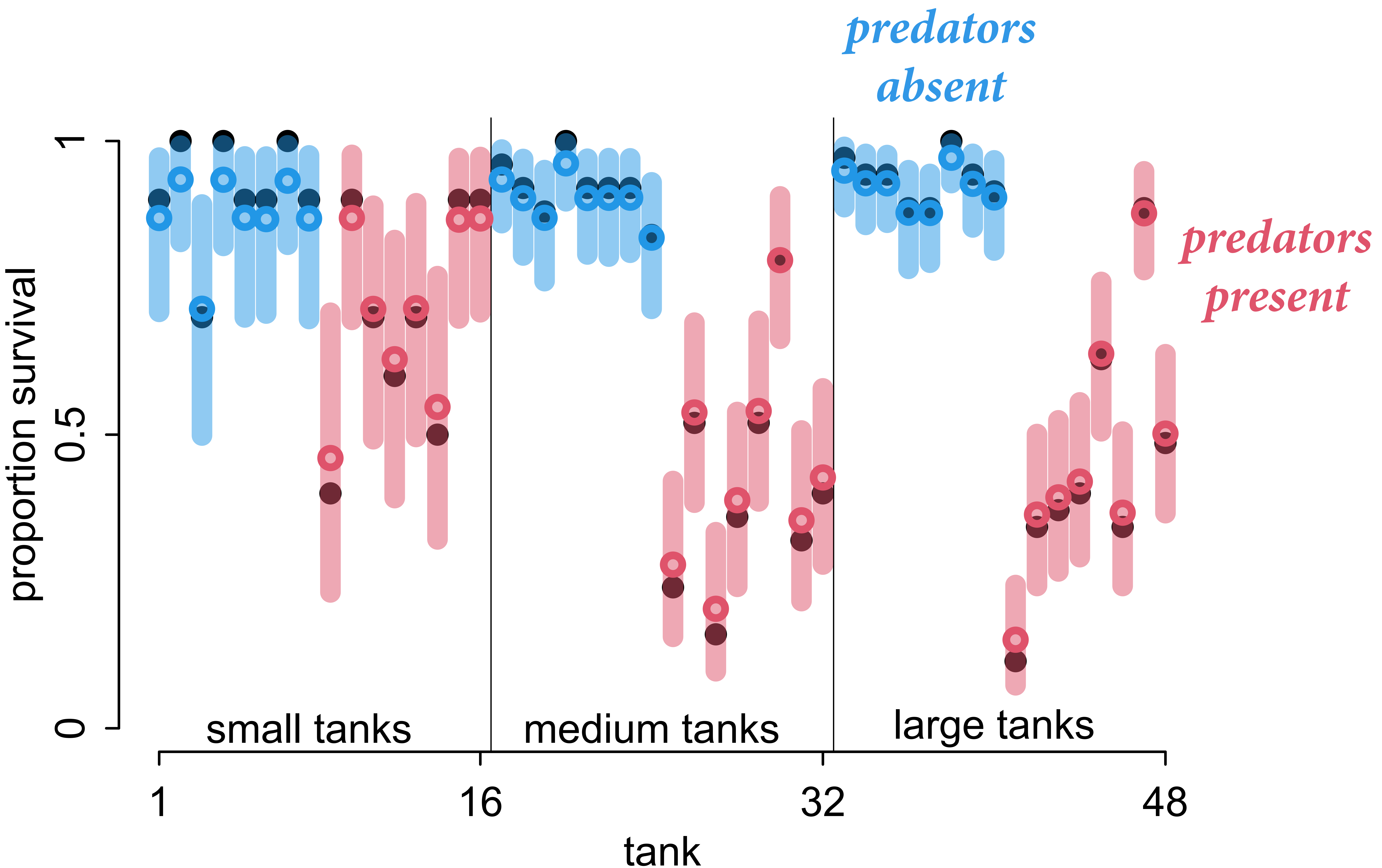
```
mSTnomem <- ulam(
    alist(

    ), d

compare( mST , mSTnomem , func=WAIC )
```

$$\text{Binomial}(D_i, p_i)$$
$$\text{logit}(p_i) = \alpha_{T[i]}$$

Adding parameters can reduce overfitting

What matters is structure, not number


less evidence,
more conservative
estimates
more evidence,
less conservative
estimates
proportion survival
1
0.5
0
small tanks
medium tanks
large tanks
1
16
32
48
tank


predators absent
predators present
proportion survival
1
0.5
0
small tanks
medium tanks
large tanks
1
16
32
48
tank

Stratify mean by predators

$$S_i \sim \text{Binomial}(D_i, p_i)$$

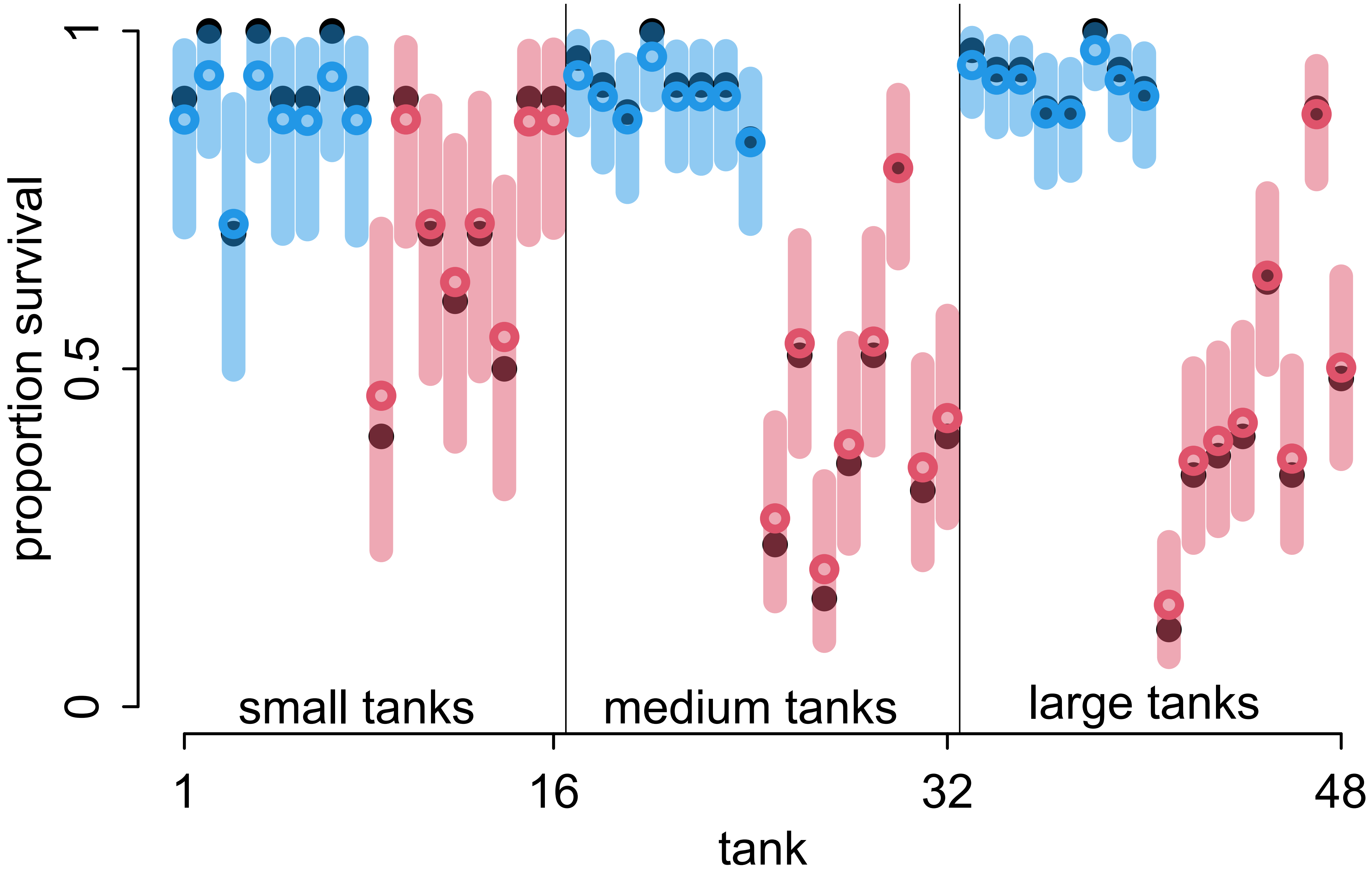
$$\text{logit}(p_i) = \alpha_{T[i]} + \beta_P P_i$$

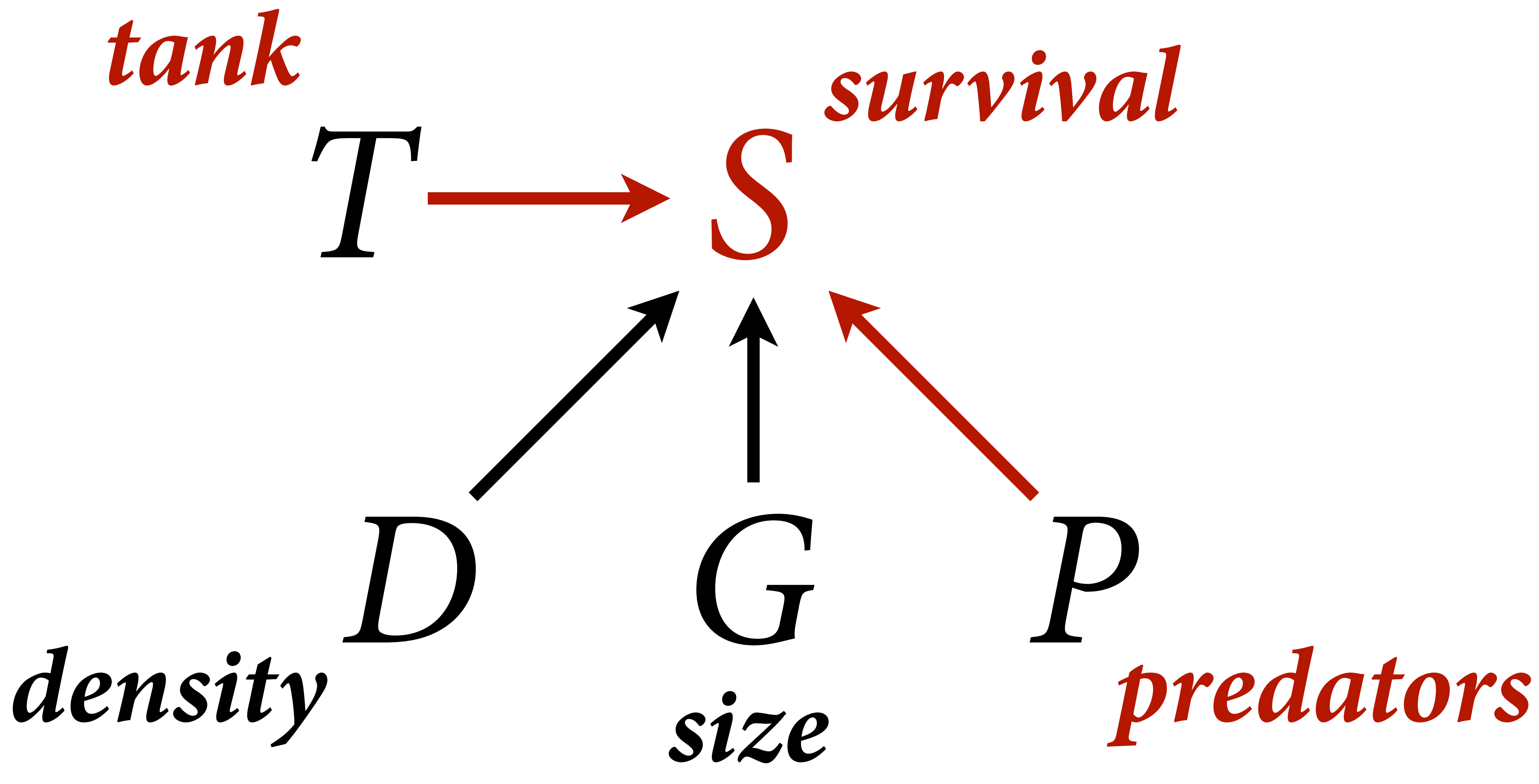
$$\beta_P \sim \text{Normal}(0, 0.5)$$

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

$$\bar{\alpha} \sim \text{Normal}(0, 1.5)$$

$$\sigma \sim \text{Exponential}(1)$$





Stratify mean by predators

$$S_i \sim \text{Binomial}(D_i, p_i)$$

$$\text{logit}(p_i) = \alpha_{T[i]} + \beta_P P_i$$

$$\beta_P \sim \text{Normal}(0,0.5)$$

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

$$\bar{\alpha} \sim \text{Normal}(0,1.5)$$

$$\sigma \sim \text{Exponential}(1)$$

```
dat$P <- ifelse(d$pred=="pred",1,0)
mSTP <- ulam(
    alist(
        S ~ dbinom( D , p ) ,
        logit(p) <- a[T] + bP*P ,
        bP ~ dnorm( 0 , 0.5 ),
        a[T] ~ dnorm( a_bar , sigma ) ,
        a_bar ~ dnorm( 0 , 1.5 ) ,
        sigma ~ dexp( 1 )
    ), data=dat , chains=4 , log_lik=TRUE )
```

Stratify mean by predators

$$S_i \sim \text{Binomial}(D_i, p_i)$$

$$\text{logit}(p_i) = \alpha_{T[i]} + \beta_P P_i$$

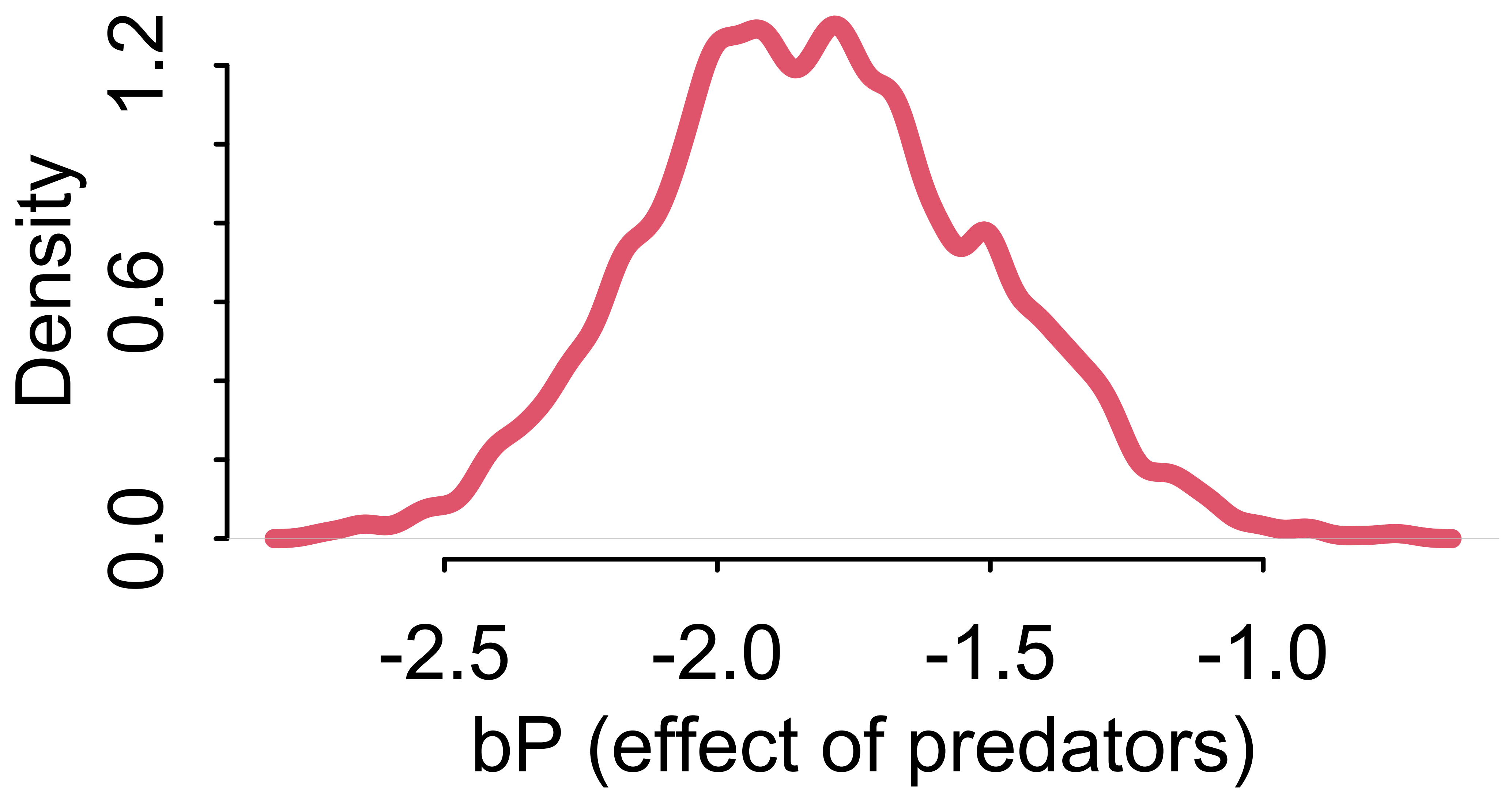
$$\beta_P \sim \text{Normal}(0, 0.5)$$

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

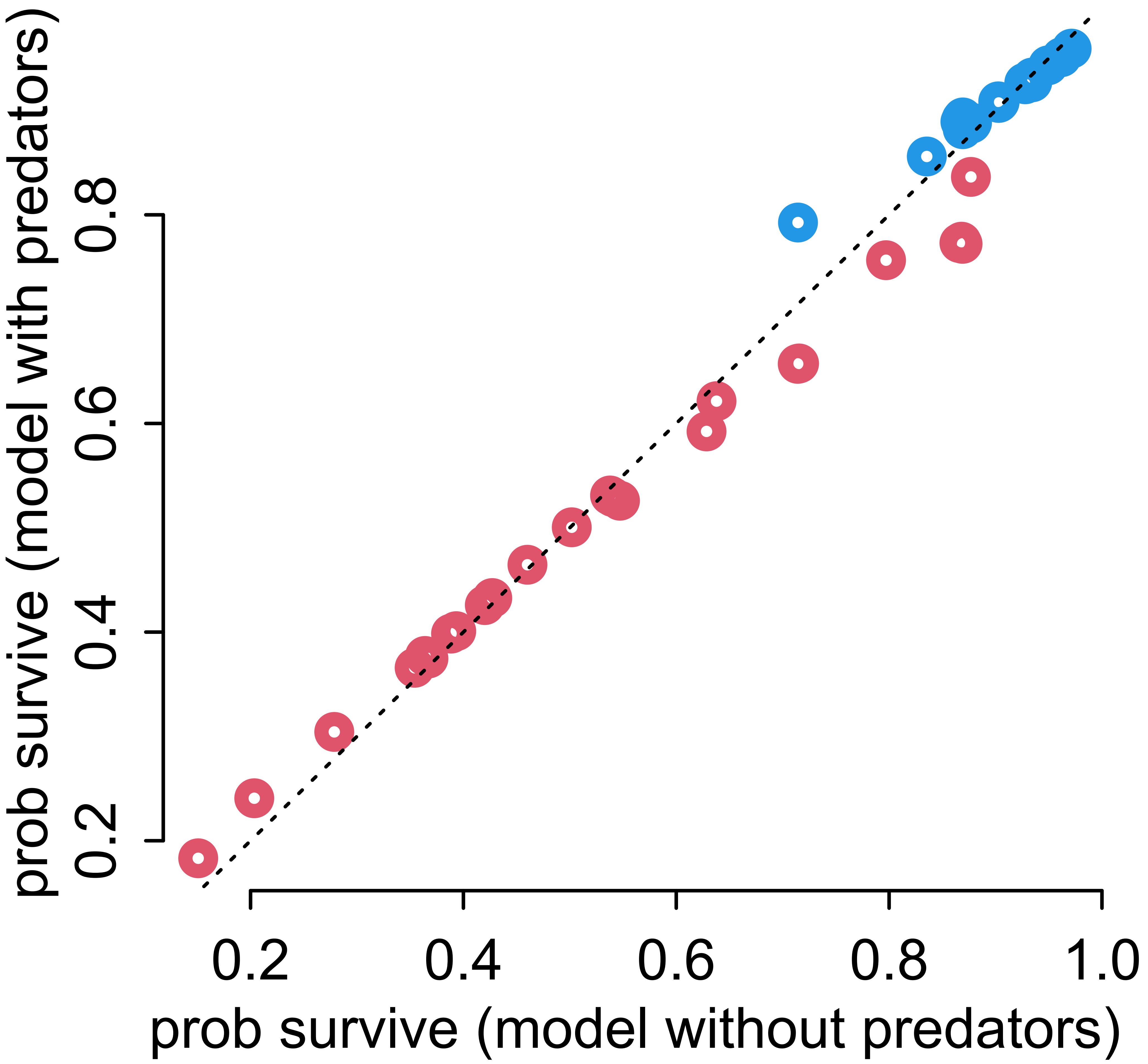
$$\bar{\alpha} \sim \text{Normal}(0, 1.5)$$

$$\sigma \sim \text{Exponential}(1)$$

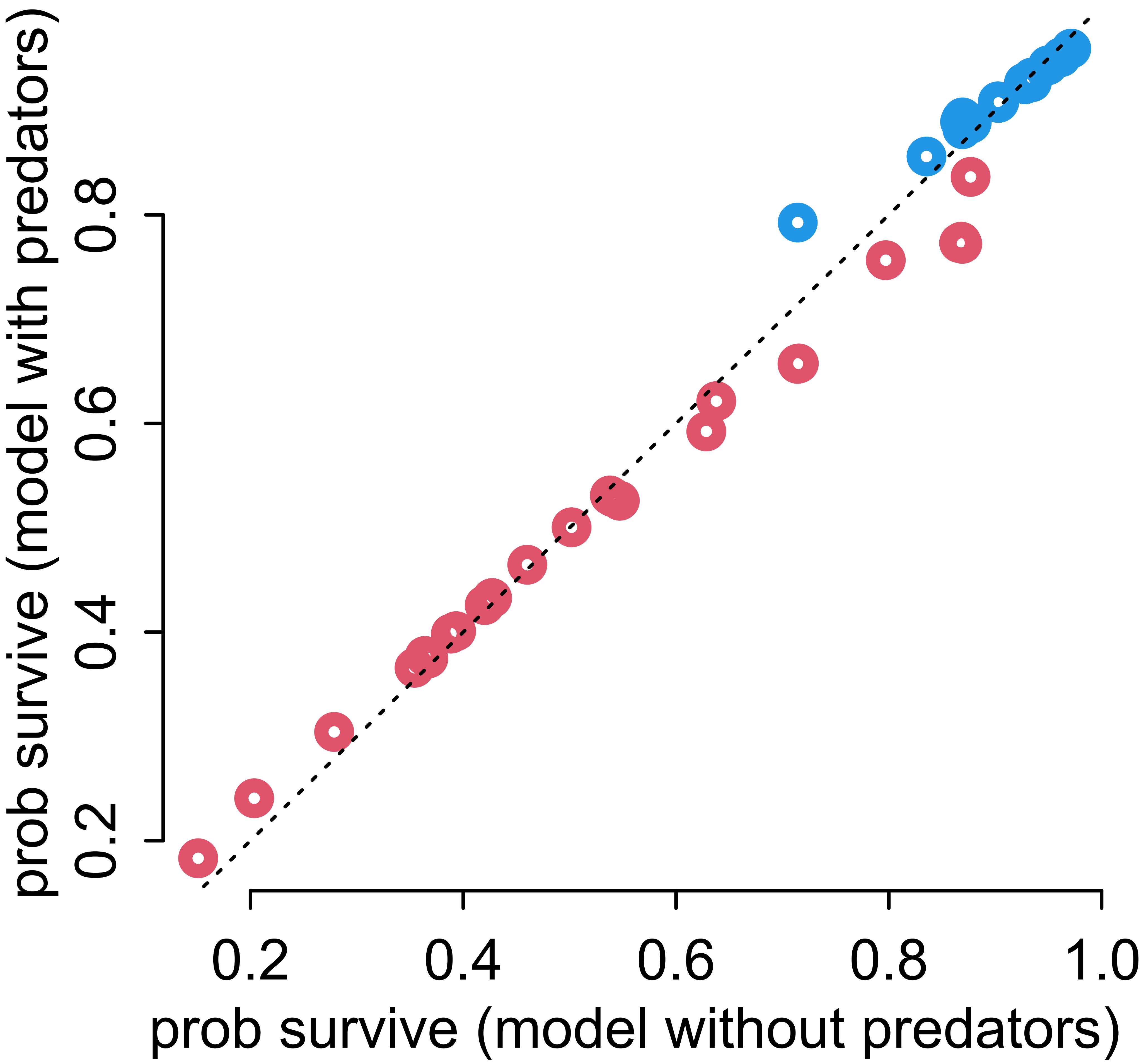
```
dat$P <- ifelse(d$pred=="pred",1,0)
mSTP <- ulam(
    alist(
        S ~ dbinom( D , p ) ,
        logit(p) <- a[T] + bP*P ,
        bP ~ dnorm( 0 , 0.5 ),
        a[T] ~ dnorm( a_bar , sigma ) ,
        a_bar ~ dnorm( 0 , 1.5 ) ,
        sigma ~ dexp( 1 )
    ), data=dat , chains=4 , log_lik=TRUE )
```



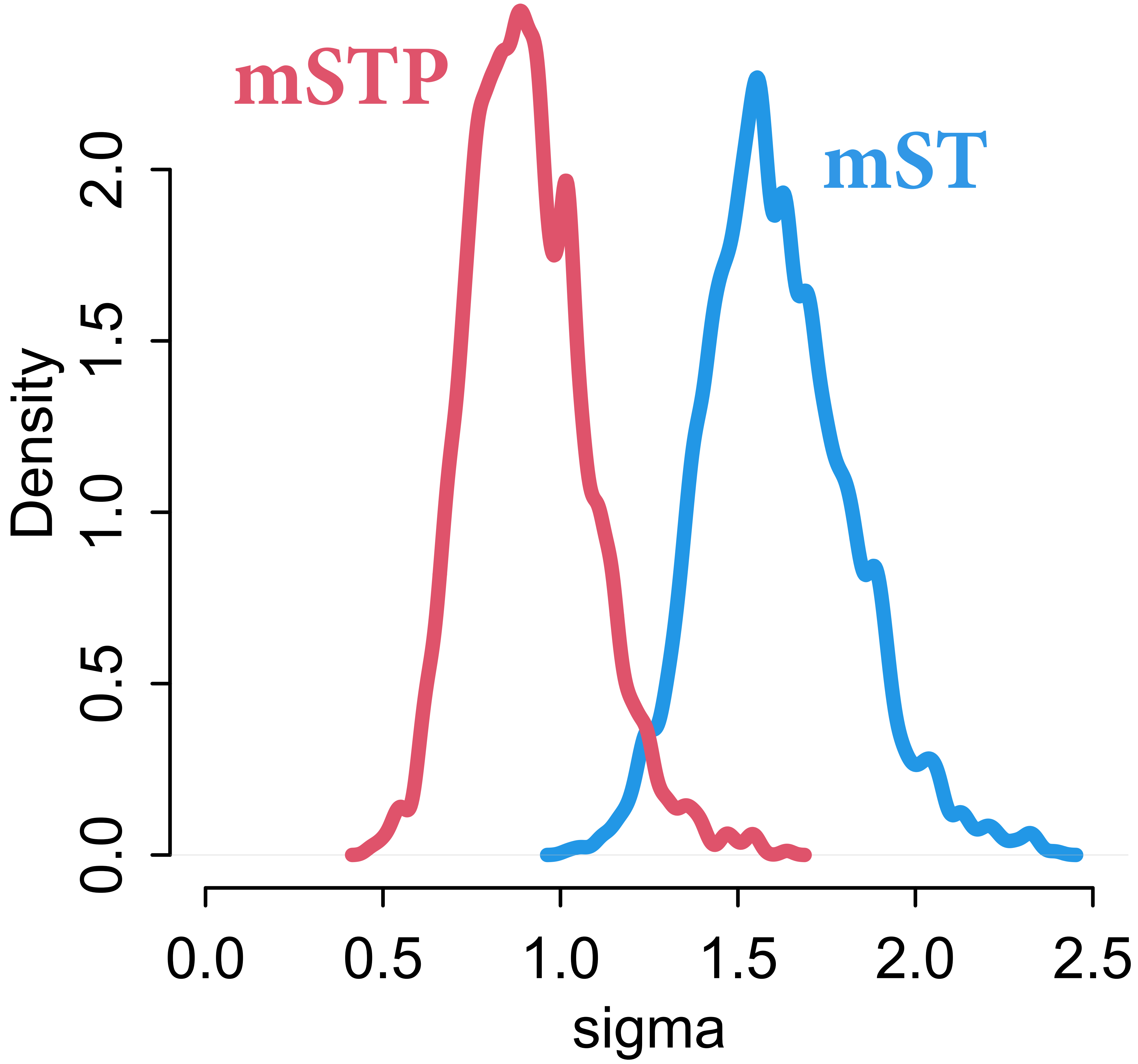
Extremely similar predictions



Extremely similar predictions



Very different sigma values



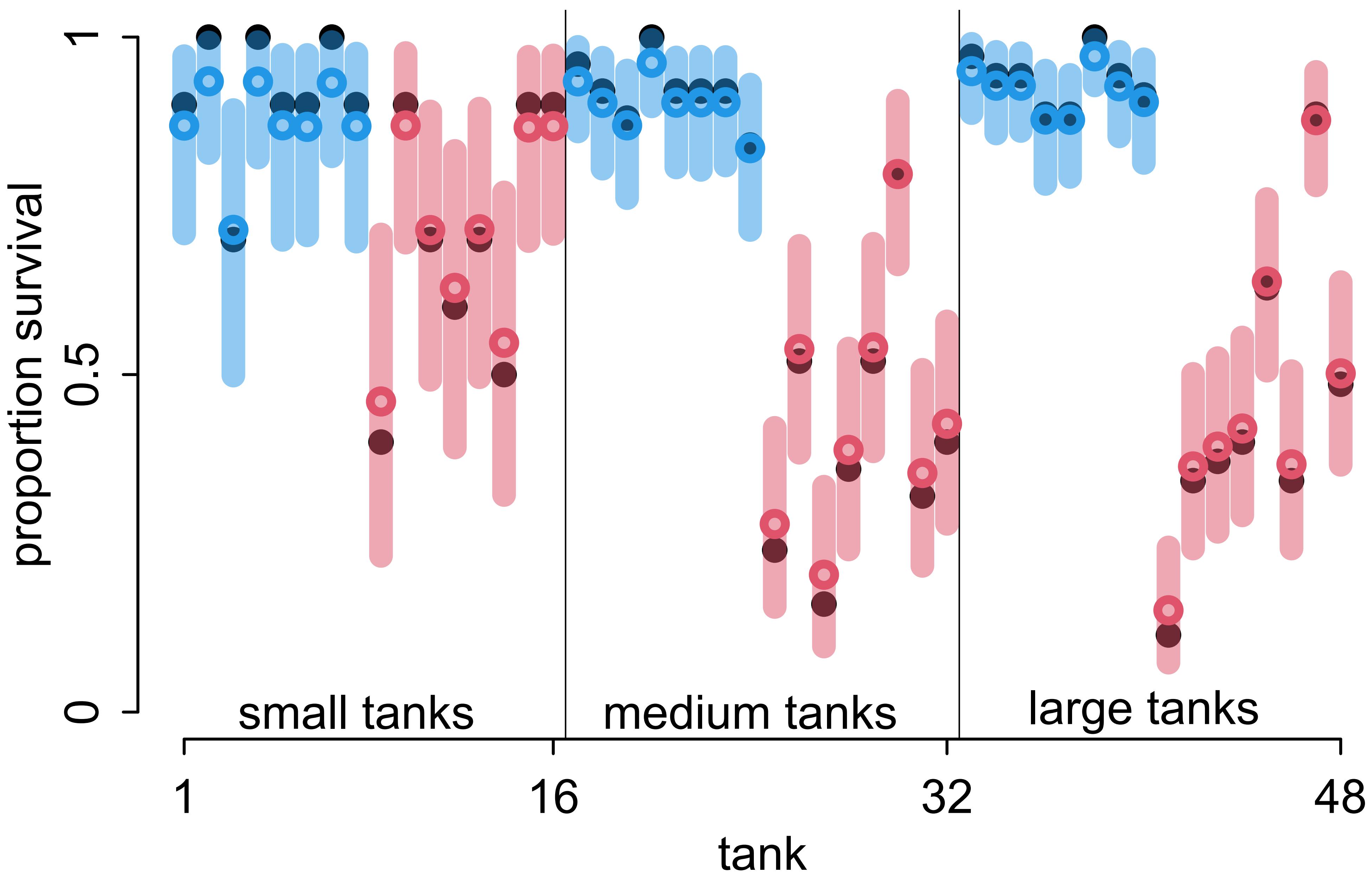
Multilevel Tadpoles

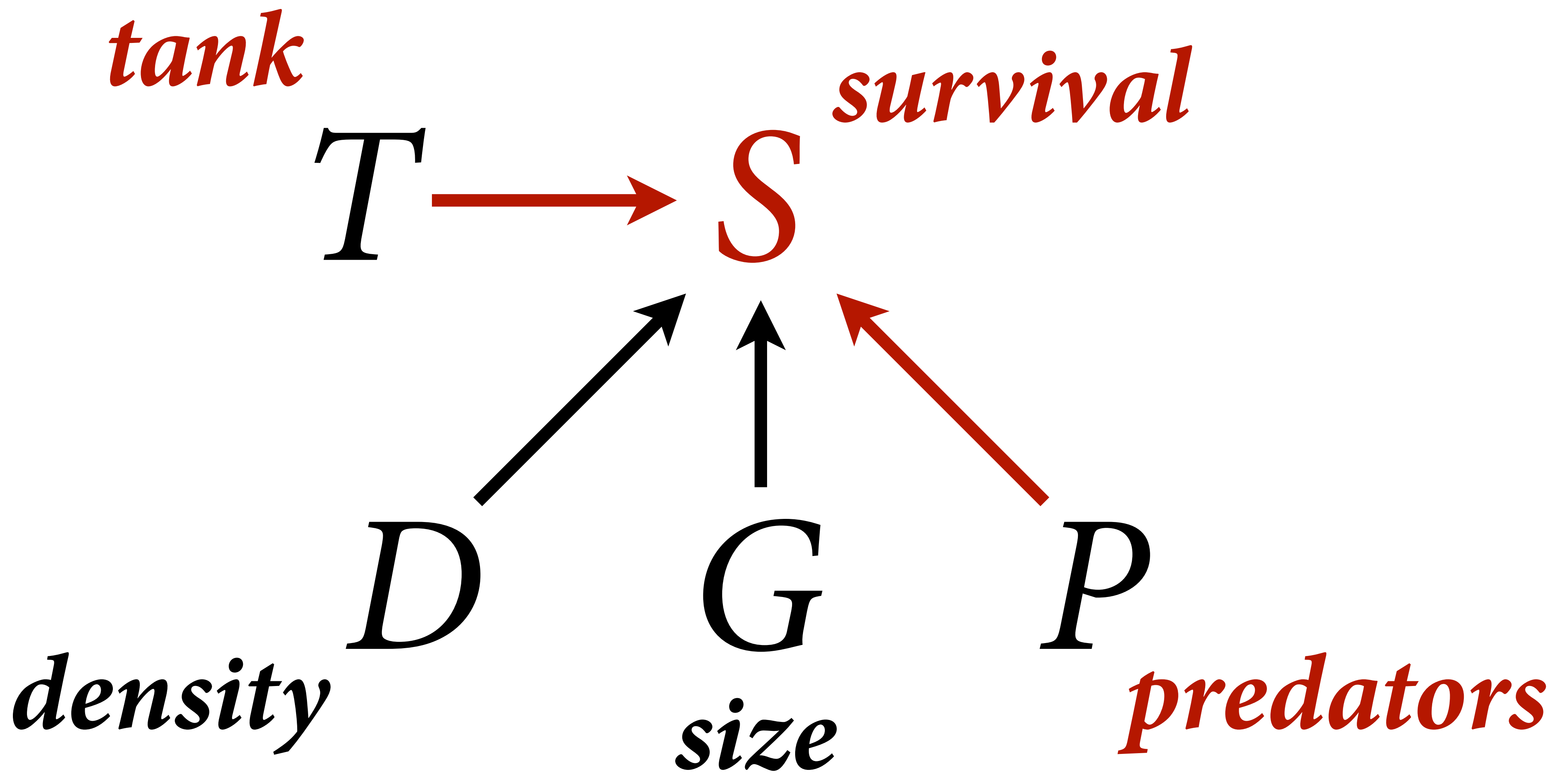
Model of unobserved population helps learn about observed units

Use data efficiently, reduce overfitting

Varying effects: Unit-specific partially pooled estimates

What about *D* and *G*? Homework





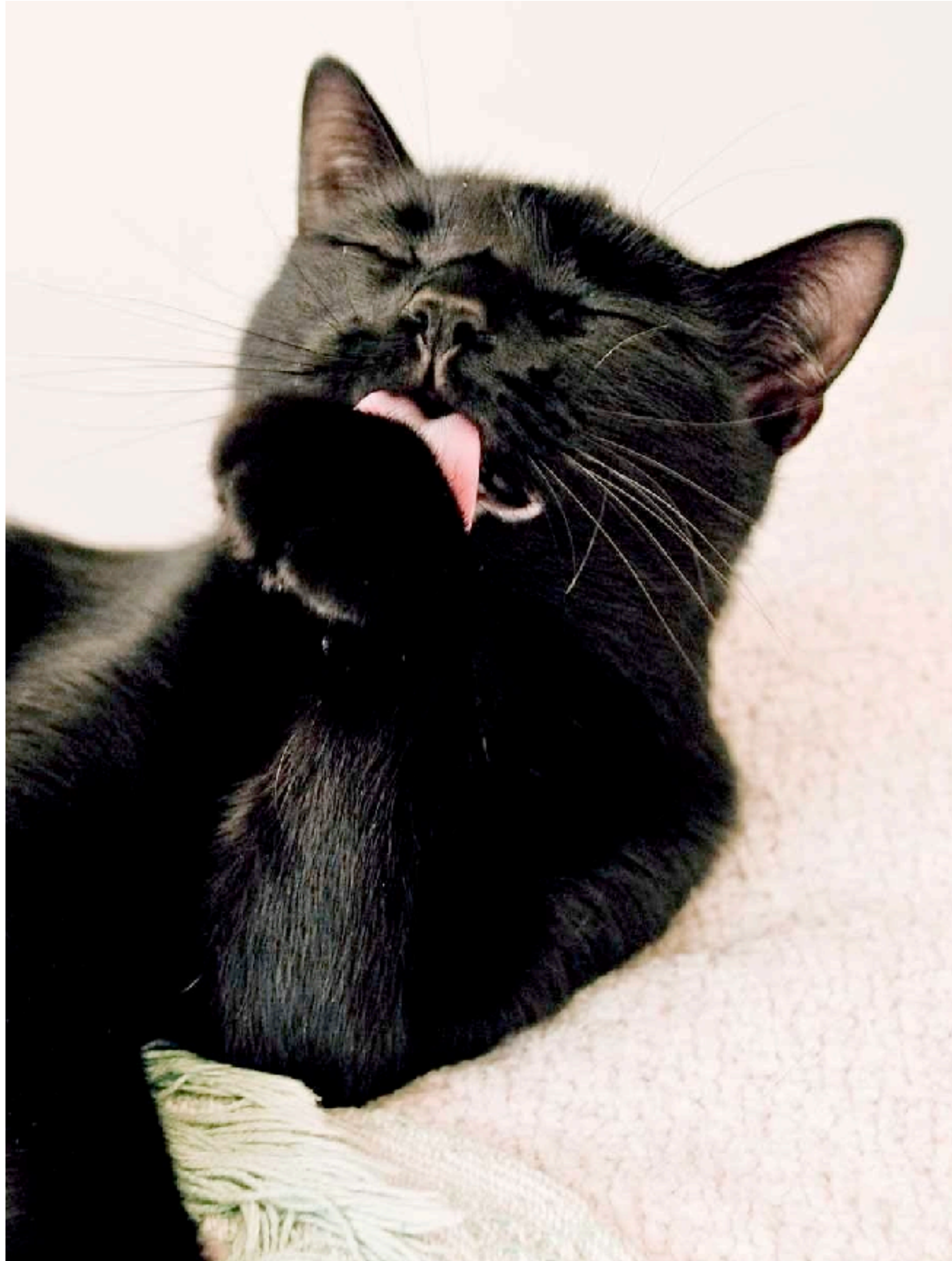
Varying Effect Superstitions

Varying effect models are plagued by superstition

(1) Units must be sampled at random

(2) Number of units must be large

(3) Assumes Gaussian variation



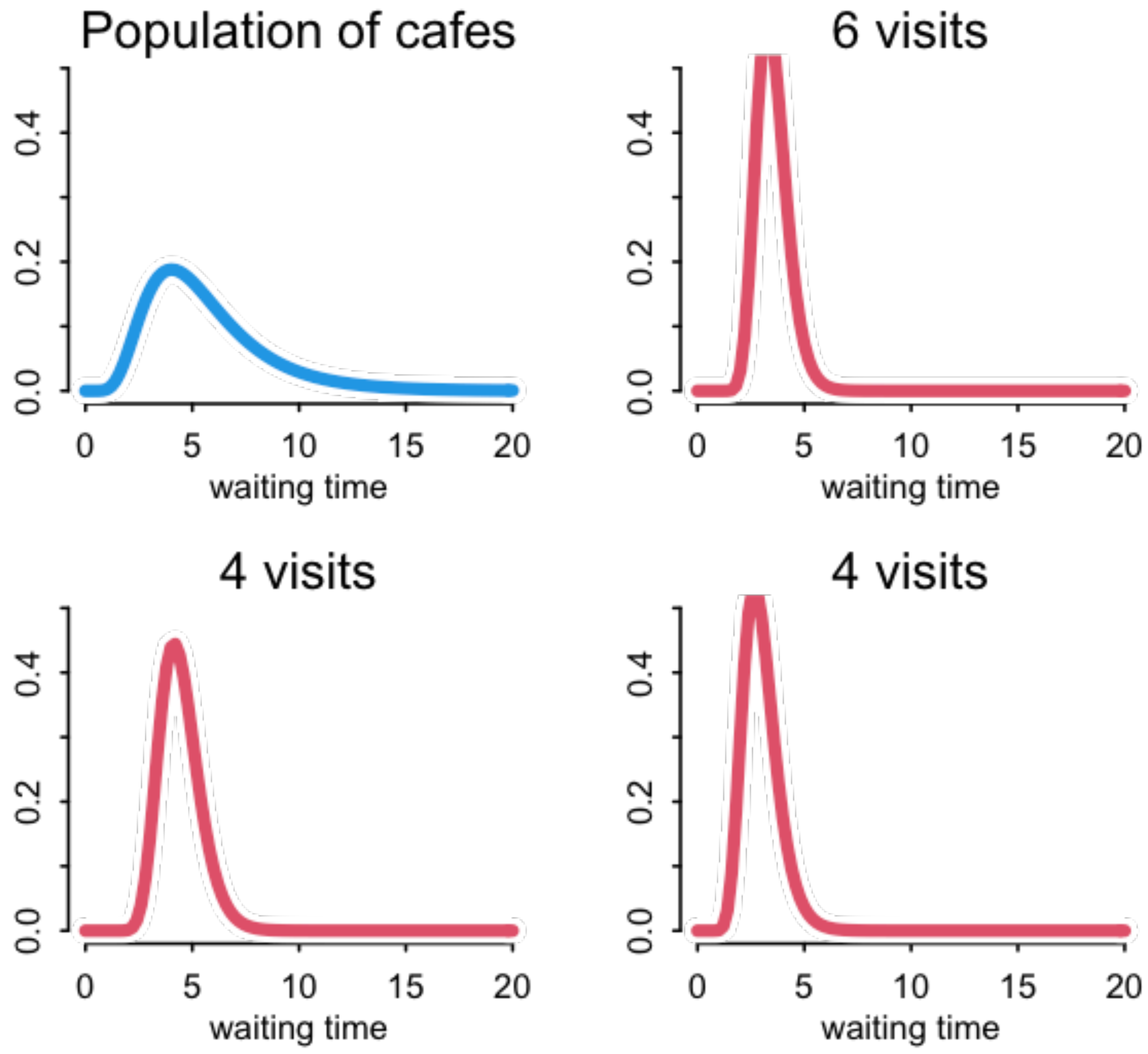
Varying Effect Superstitions

Varying effect models are plagued by superstition

(1) Units must be sampled at random

(2) Number of units must be large

(3) Assumes Gaussian variation



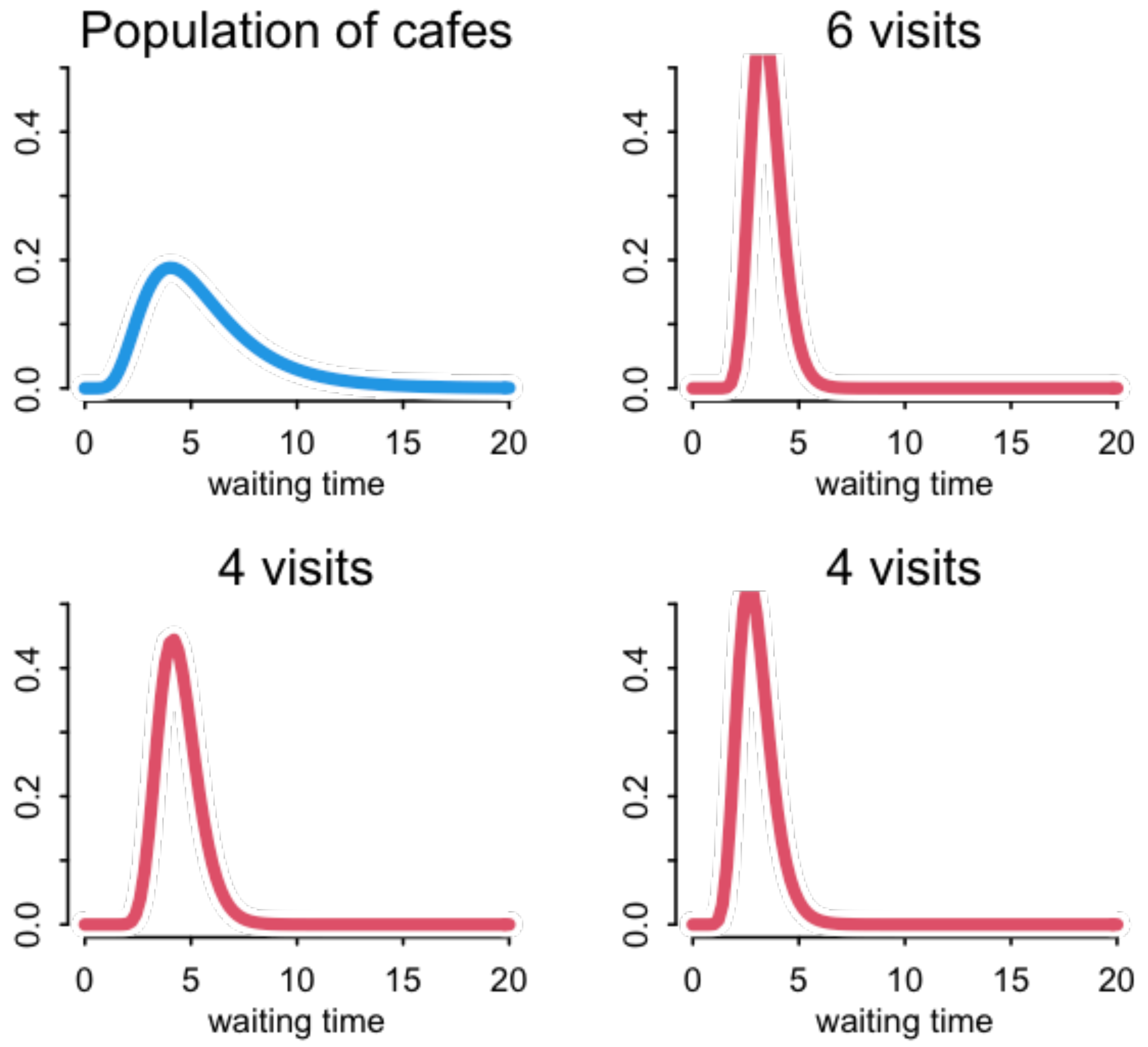
Varying Effect Superstitions

Varying effect models are plagued by superstition

~~(1) Units must be sampled at random~~

(2) Number of units must be large

(3) Assumes Gaussian variation



Varying Effect Superstitions

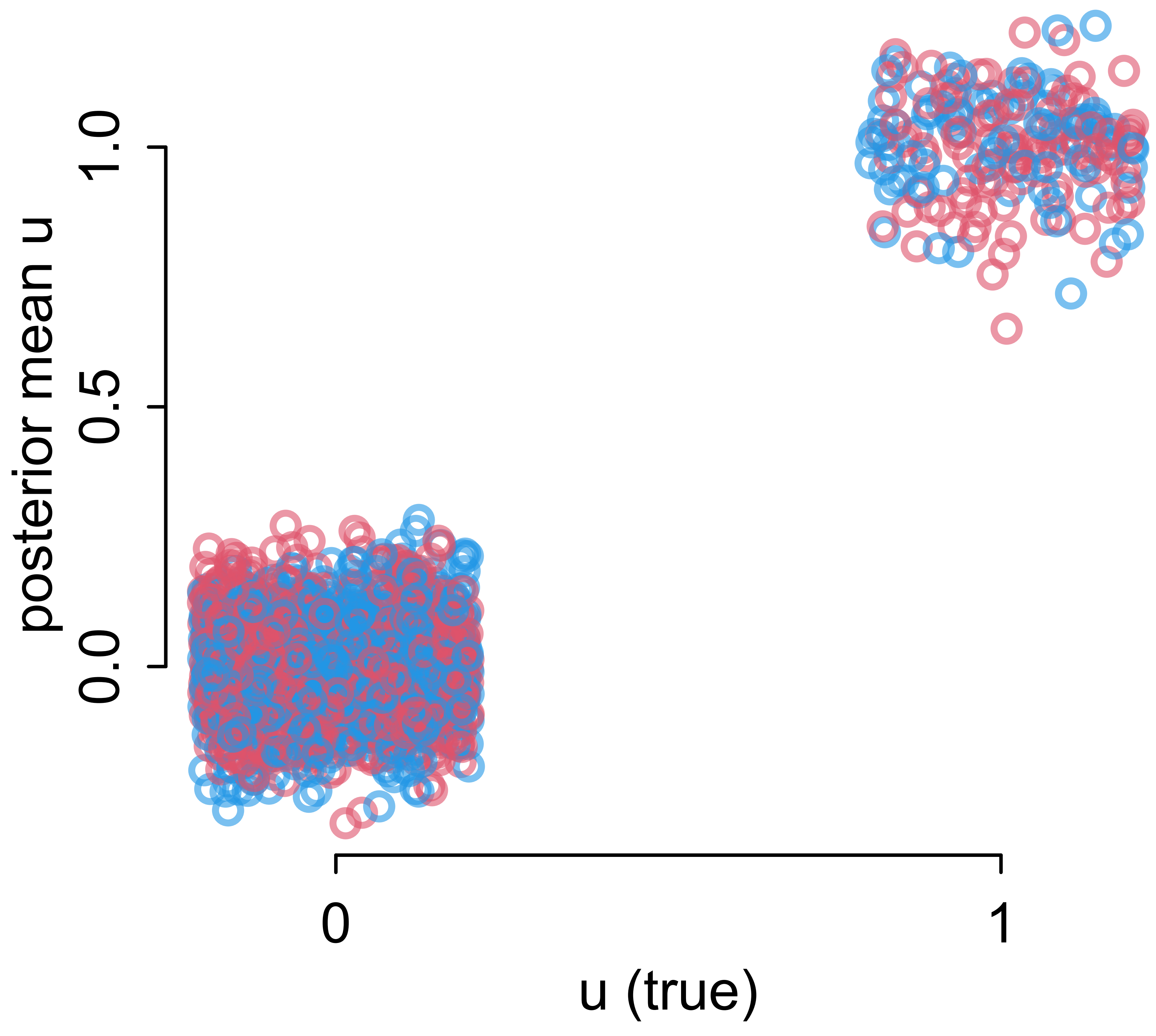
Varying effect models are plagued by superstition

~~(1) Units must be sampled at random~~

~~(2) Number of units must be large~~

(3) Assumes Gaussian variation

$u_k \sim \text{Normal}(0,1)$



Varying Effect Superstitions

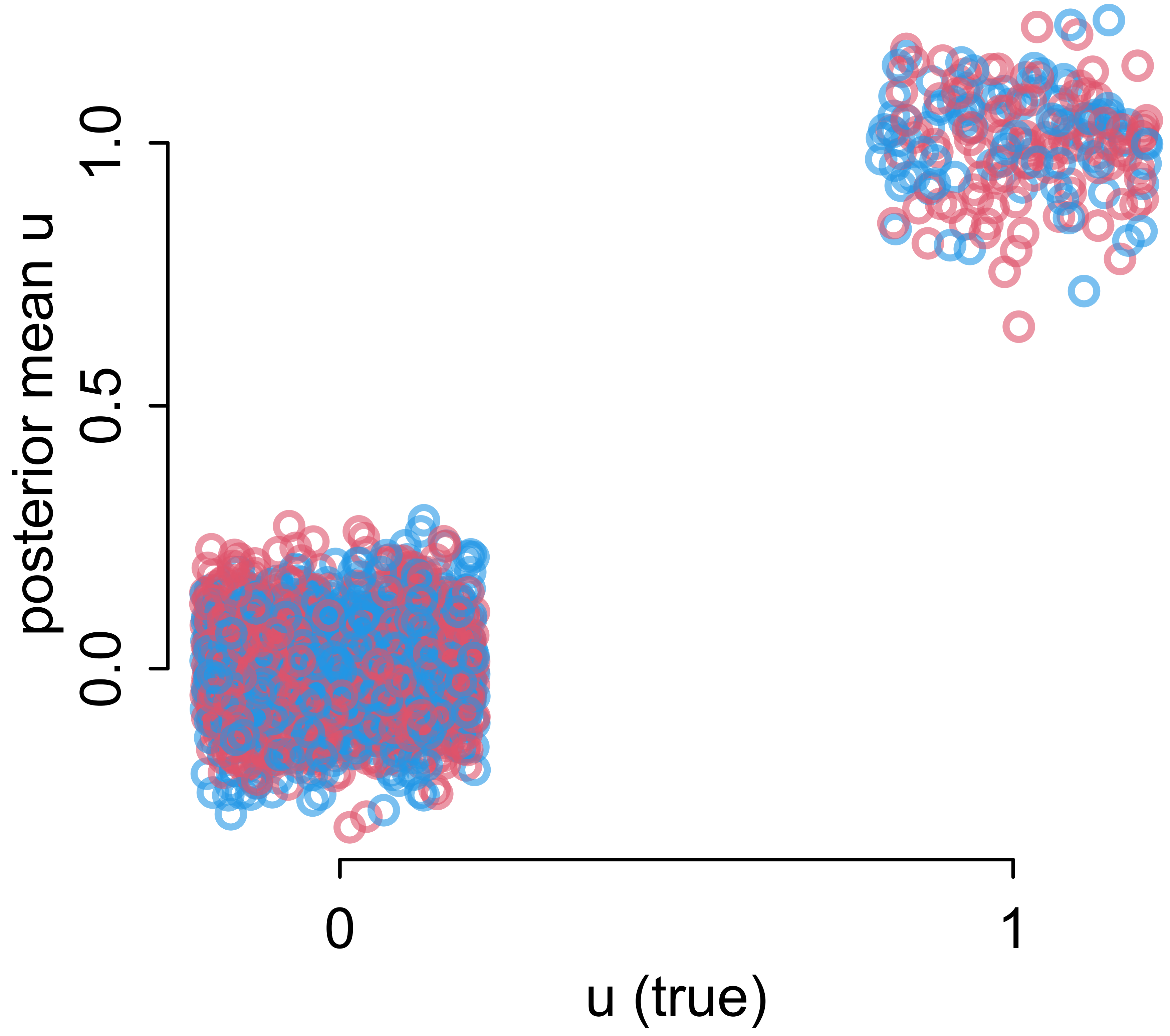
Varying effect models are plagued by superstition

~~(1) Units must be sampled at random~~

~~(2) Number of units must be large~~

~~(3) Assumes Gaussian variation~~

$u_k \sim \text{Normal}(0,1)$



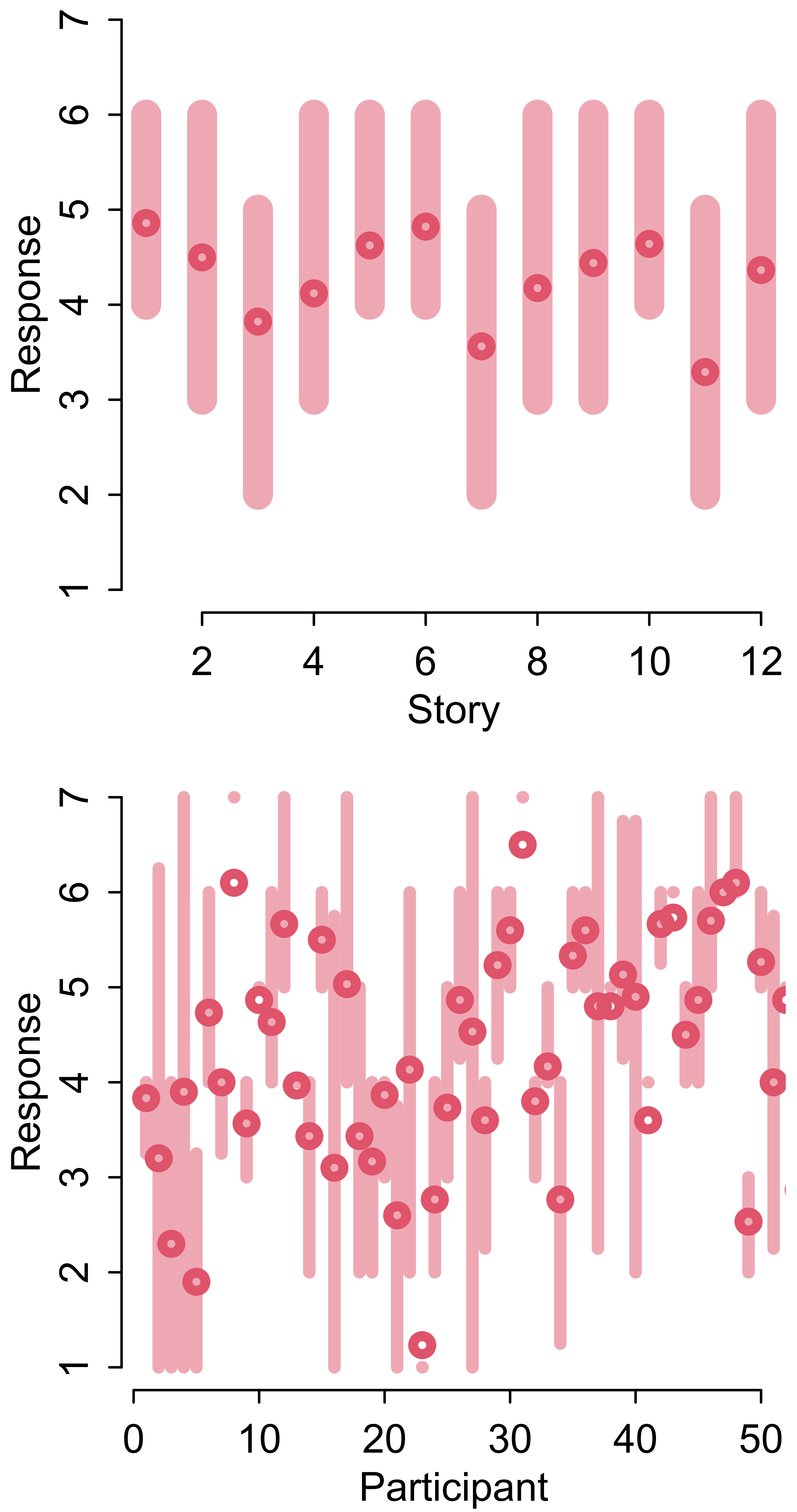
Practical Difficulties

Varying effects are a good default, but…

(1) How to use more than one cluster type at the same time? For example stories and participants

(2) How to sample efficiently

(3) What about slopes? Confounds?



Course Schedule

Week 1	Bayesian inference	Chapters 1, 2, 3
Week 2	Linear models & Causal Inference	Chapter 4
Week 3	Causes, Confounds & Colliders	Chapters 5 & 6
Week 4	Overfitting / MCMC	Chapters 7, 8, 9
Week 5	Generalized Linear Models	Chapters 10, 11
Week 6	Ordered categories & Multilevel models	Chapters 12 & 13
Week 7	More Multilevel models	Chapters 13 & 14
Week 8	Multilevel models & Gaussian processes	Chapter 14
Week 9	Measurement & Missingness	Chapter 15
Week 10	Generalized Linear Madness	Chapter 16

https://github.com/rmcelreath/stat_rethinking_2023

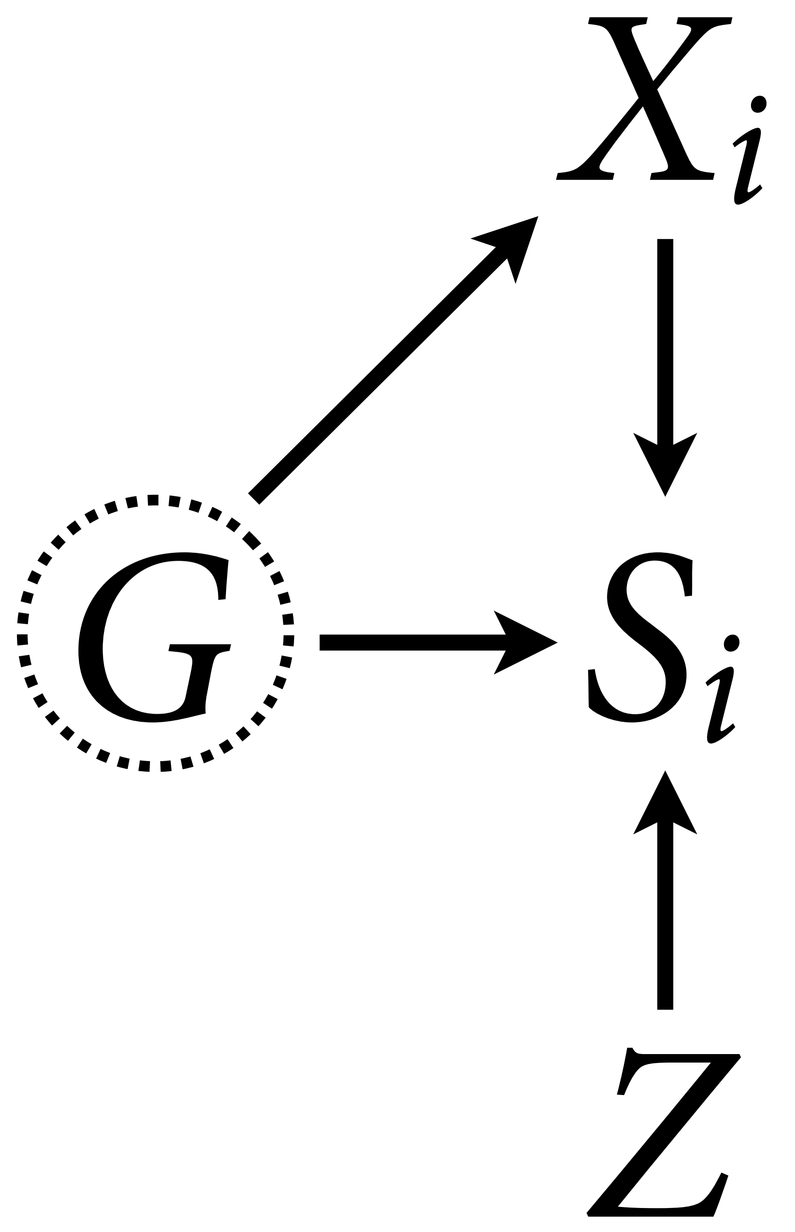
BONUS

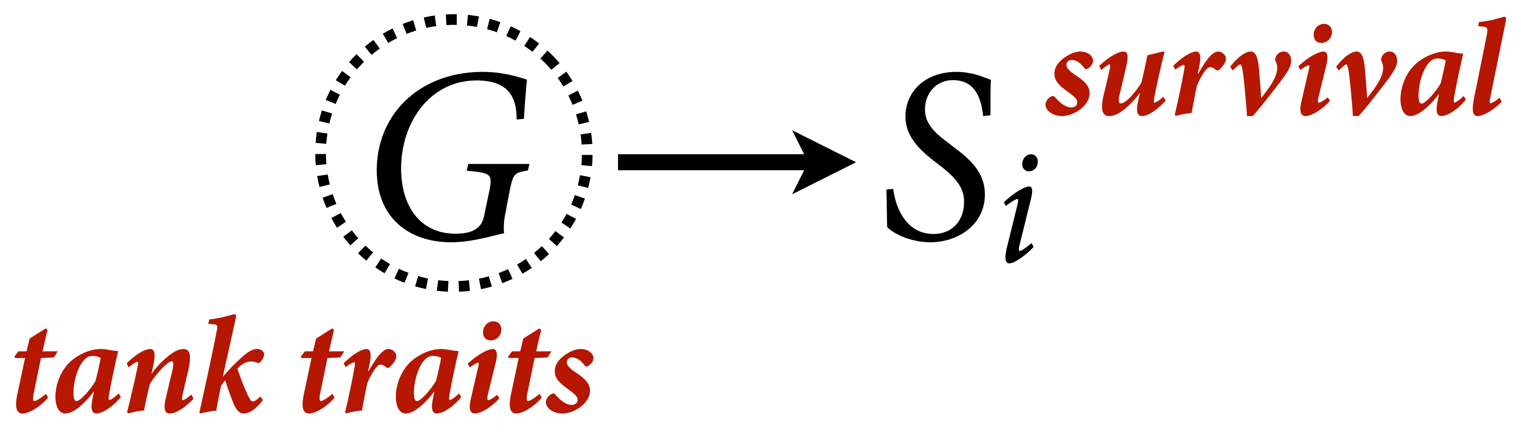
Random confounds

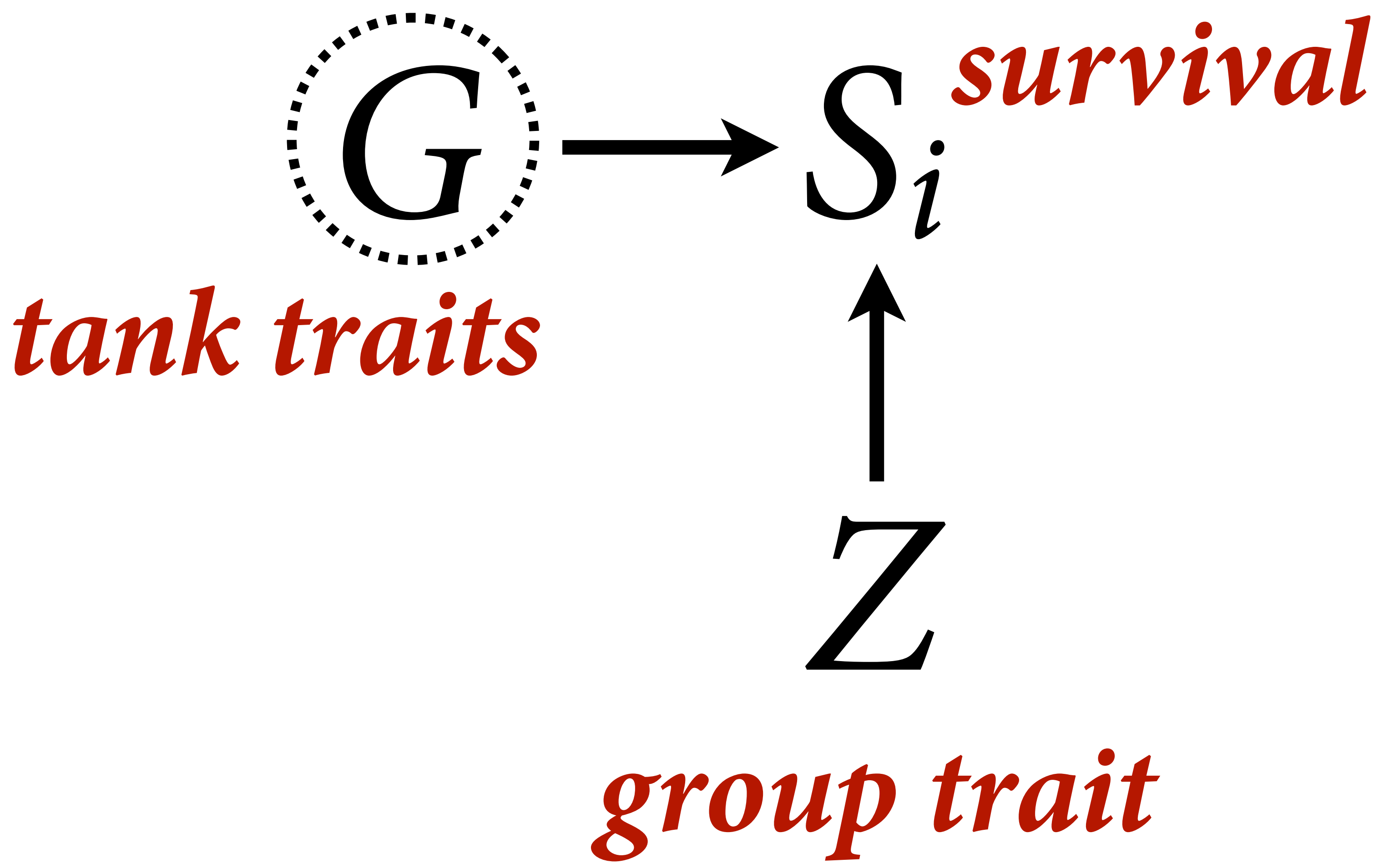
When unobserved group features influence individually-varying causes

Dizzying terminology: group-level confounding, endogeneity, correlated errors, econometrics

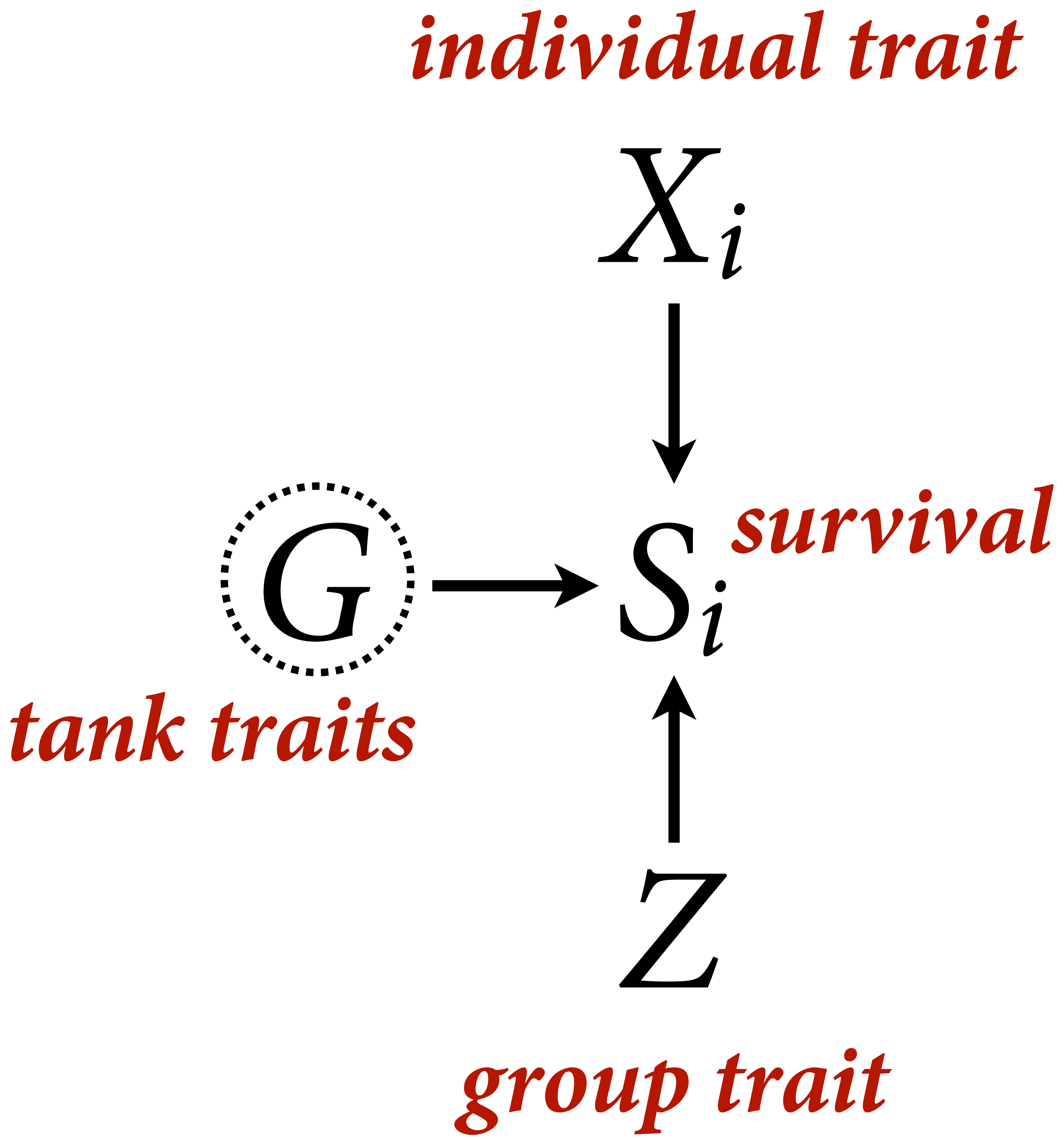
Group-level variables have direct and indirect influences

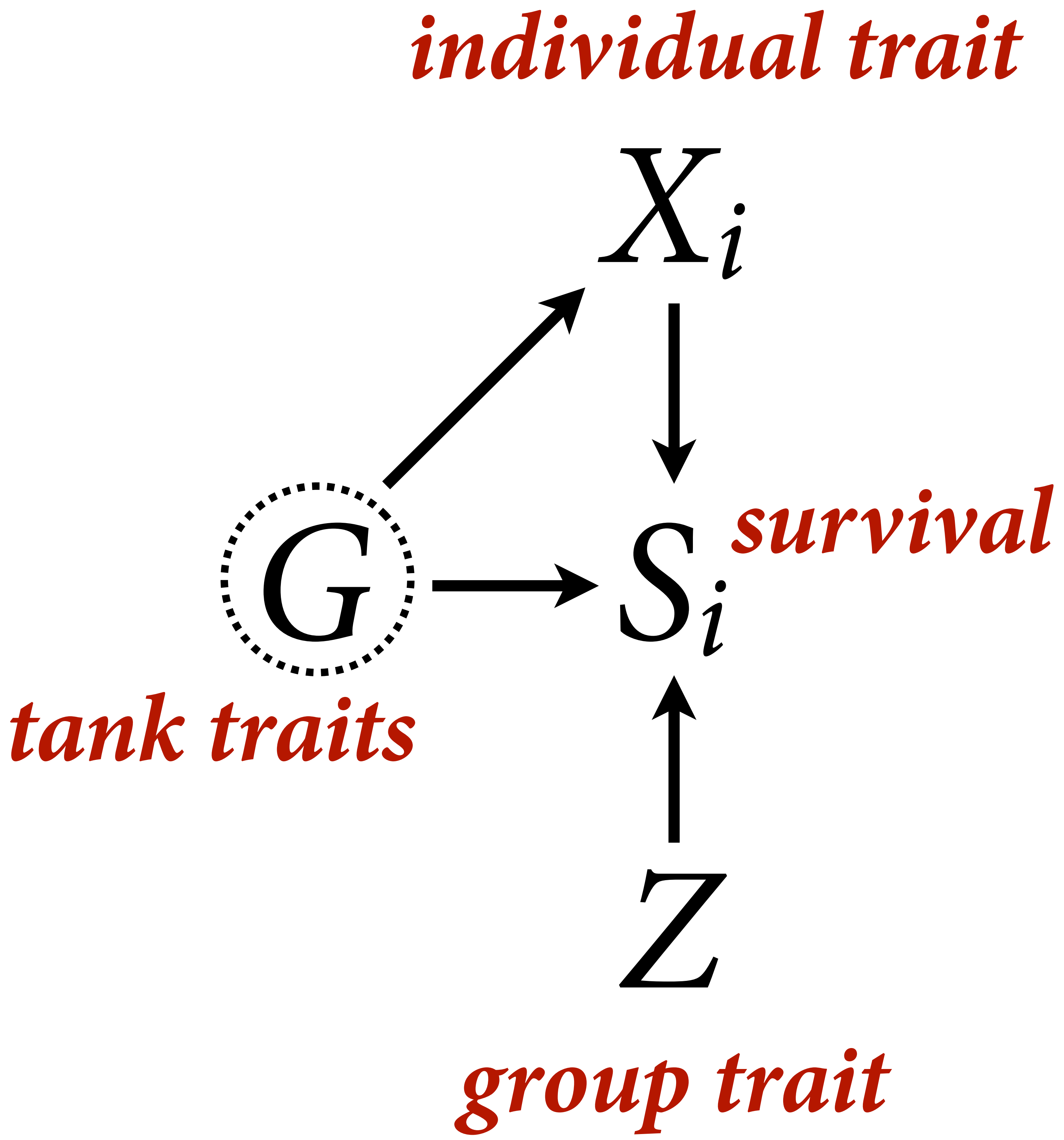



G
S_i
survival
tank traits



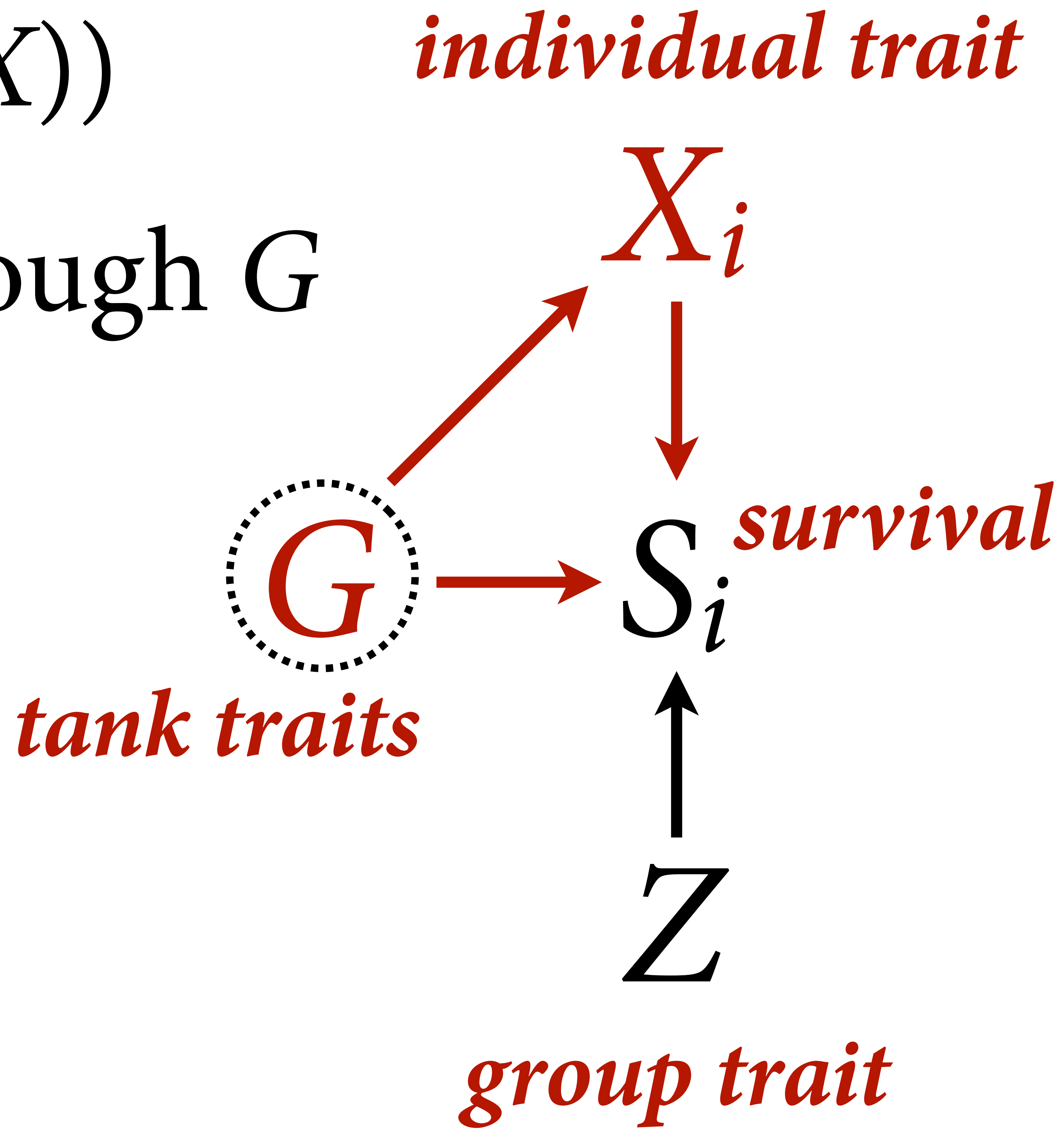
G
S_i
survival
tank traits
Z
group trait

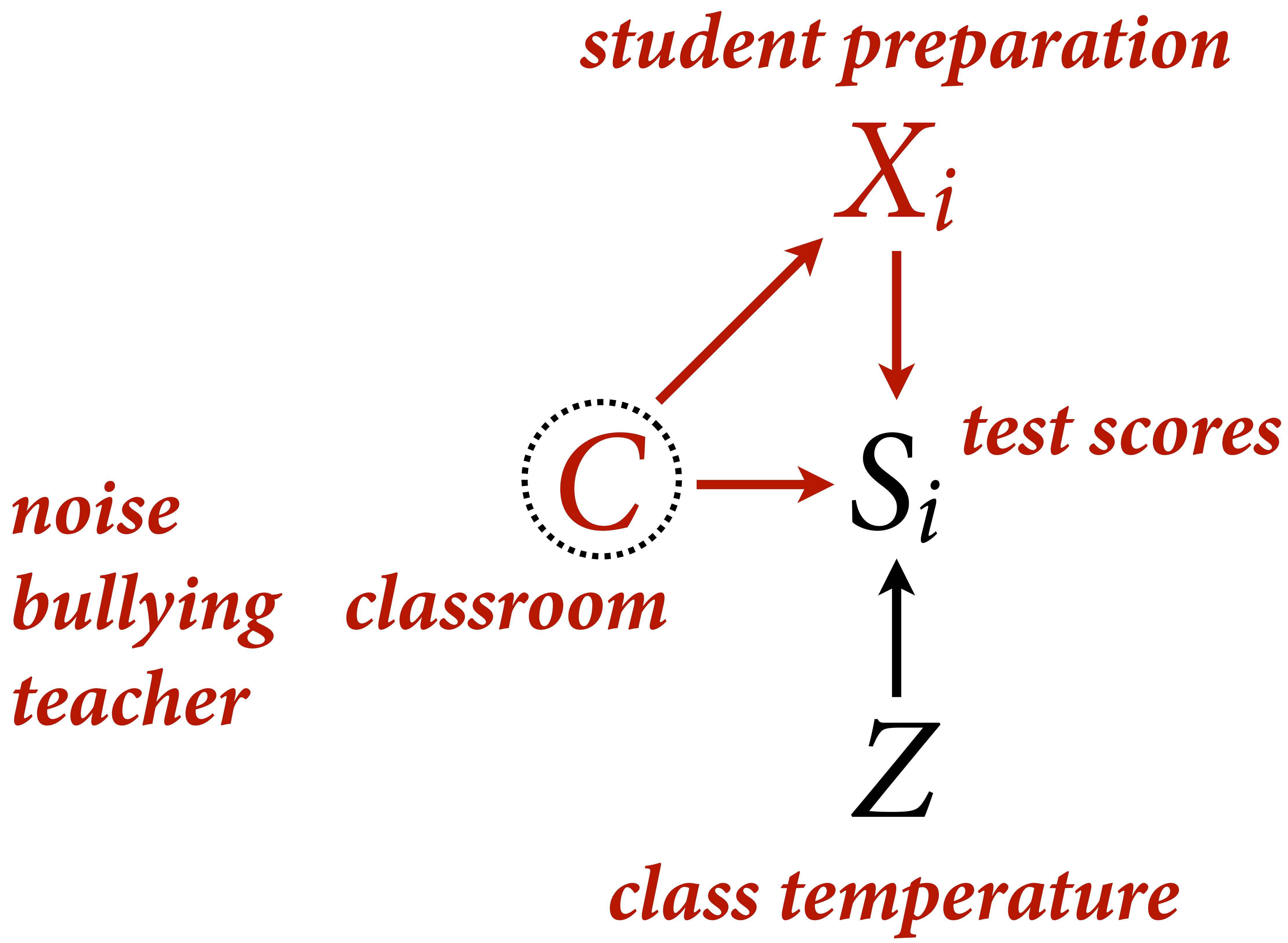

individual trait
X_i
G
S_i
survival
tank traits
Z
group trait

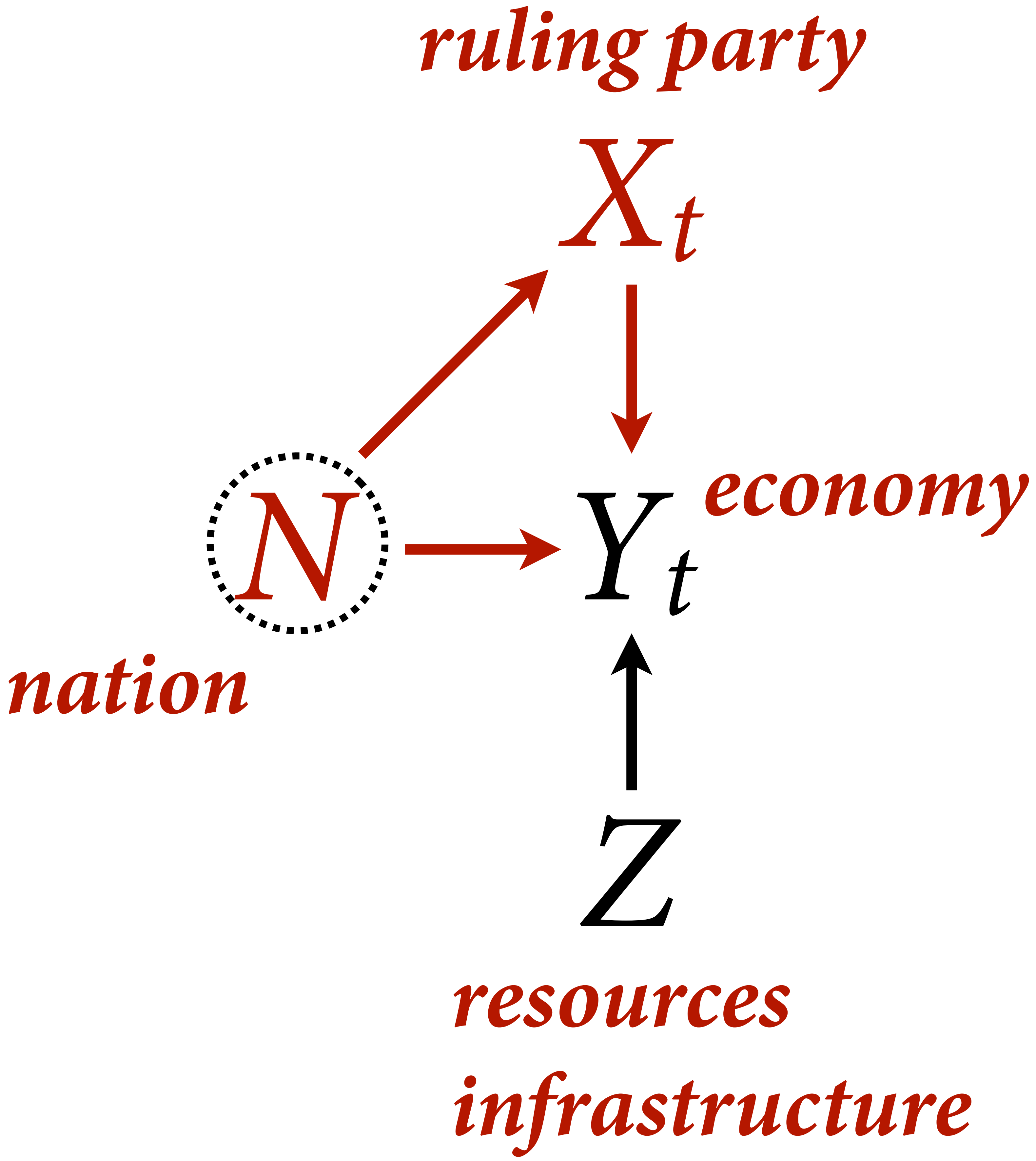

individual trait
X_i
G
S_i
survival
tank traits
Z
group trait

Estimand: $p(S|\text{do}(X))$

Backdoor path through G




student preparation
X_i
C
S_i
test scores
noise
bullying
teacher
classroom
Z
class temperature


ruling party
X_t
N
Y_t
economy
nation
Z
resources
infrastructure

Random confounds

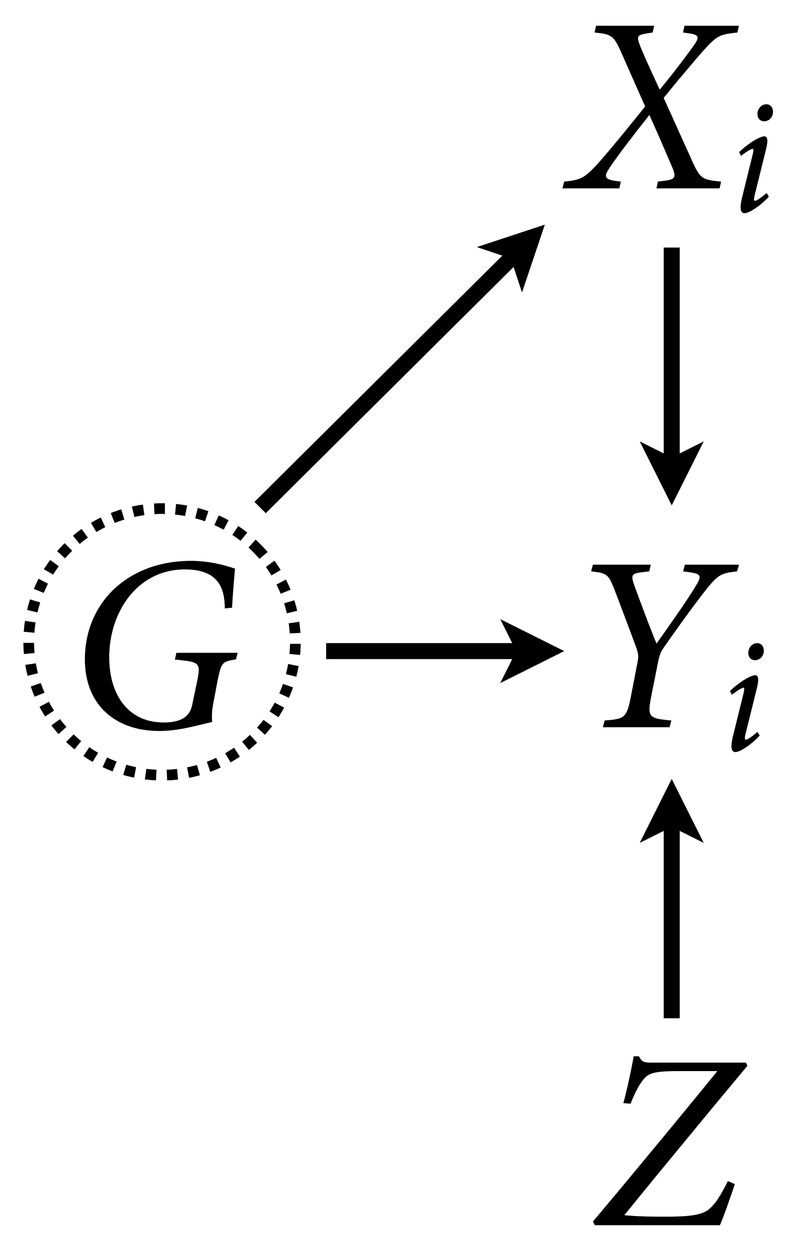
Estimand: Influence of X on Y

Estimator?

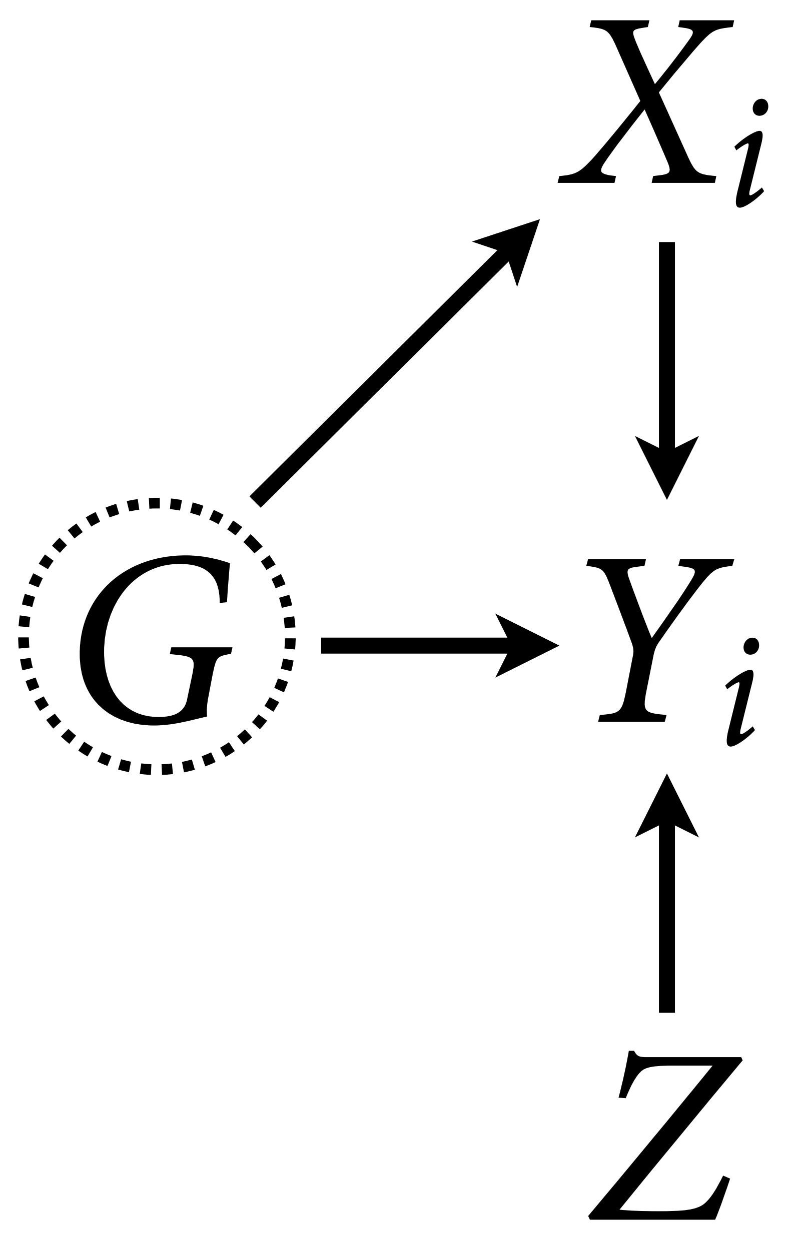
(1) Fixed effects model

(2) Multilevel model

(3) Mundlak Machines



```
N_groups <- 30
N_id <- 200
a0 <- (-2)
bZY <- (-0.5)
g <- sample(1:N_groups,size=N_id,replace=TRUE) # sample into groups
Ug <- rnorm(N_groups,1.5) # group confounds
X <- rnorm(N_id, Ug[g] ) # individual varying trait
Z <- rnorm(N_groups) # group varying trait (observed)
Y <- rbern(N_id, p=inv_logit( a0 + X + Ug[g] + bZY*Z[g] ) )

table(g)
```




```
> table(g)
g
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
11  5  8  3 11  7  6  4  9  8  4  5  3  7  7  7  4  3  8 10  4  7  6 12  5 12

27 28 29 30
 7  7  5  5
```

Fixed effects model

Estimate a different average rate for each group, without pooling

Inefficient, but soaks up group-level (fixed) confounding (G)

Problem: Cannot identify any group-level effects (Z)

$$Y_i \sim \text{Bernoulli}(p_i)$$

$$\text{logit}(p_i) = \alpha_{G[i]} + \beta_X X_i + \beta_Z Z_{G[i]}$$

$$\alpha_j \sim \text{Normal}(0,10)$$

$$\beta_{X,Z} \sim \text{Normal}(0,1)$$

Fixed effects model

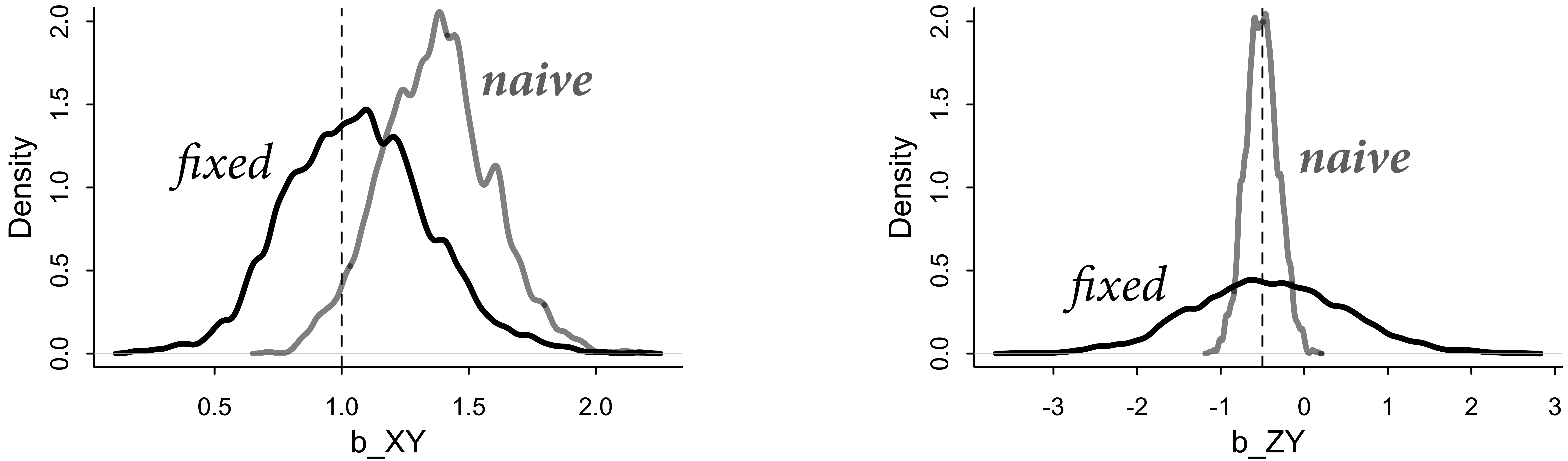
```
dat <- list(Y=Y,X=X,g=g,Ng=N_groups,Z=Z)

# fixed effects
mf <- ulam(
    alist(
        Y ~ bernoulli(p),
        logit(p) <- a[g] + bxy*X + bzy*Z[g],
        a[g] ~ dnorm(0,10),
        c(bxy,bzy) ~ dnorm(0,1)
    ) , data=dat , chains=4 , cores=4 )
```

$$Y_i \sim \text{Bernoulli}(p_i)$$

$$\text{logit}(p_i) = \alpha_{G[i]} + \beta_X X_i + \beta_Z Z_{G[i]}$$

$$\alpha_j \sim \text{Normal}(0,10)$$

$$\beta_{X,Z} \sim \text{Normal}(0,1)$$



Multilevel model

Estimate a different average rate for each group, partial pooling

Better estimates for G, worse estimate for X

Bonus: Can identify Z

$$Y_i \sim \text{Bernoulli}(p_i)$$

$$\text{logit}(p_i) = \alpha_{G[i]} + \beta_X X_i + \beta_Z Z_{G[i]}$$

$$\alpha_j \sim \text{Normal}(\bar{\alpha}, \tau)$$

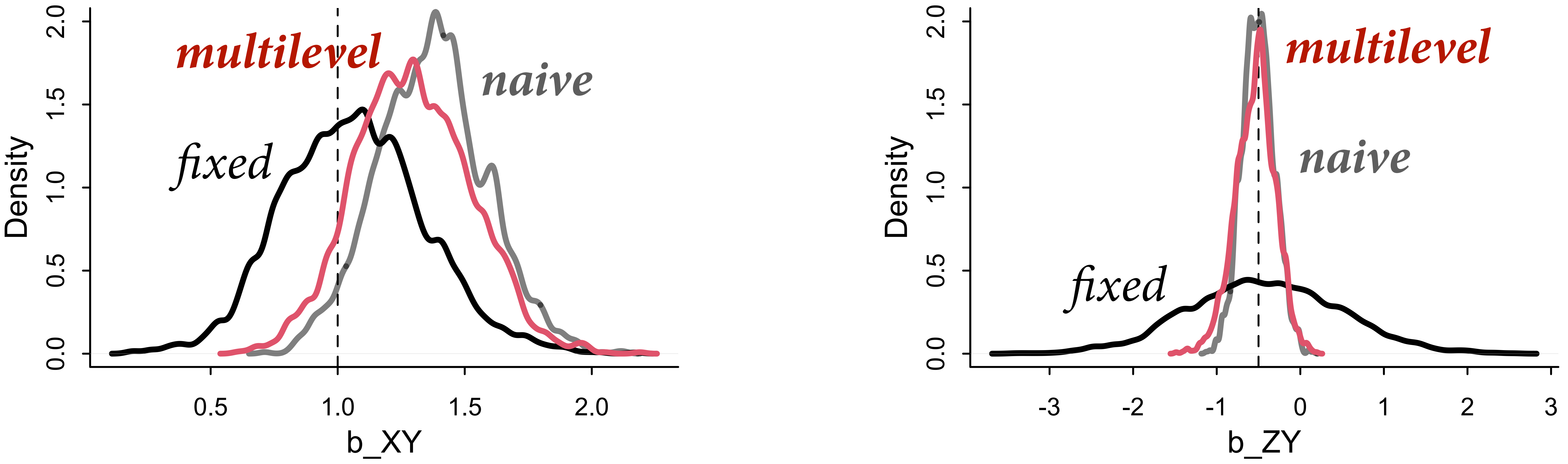
$$\beta_{X,Z} \sim \text{Normal}(0,1)$$

$$\bar{\alpha} \sim \text{Normal}(0,1)$$

$$\tau \sim \text{Exponential}(1)$$

```
# varying effects (non-centered - next week!)
mr <- ulam(
    alist(
        Y ~ bernoulli(p),
        logit(p) <- a[g] + bxy*X + bzy*Z[g],
        transpars> vector[Ng]:a <<- abar + z*tau,
        z[g] ~ dnorm(0,1),
        c(bxy,bzy) ~ dnorm(0,1),
        abar ~ dnorm(0,1),
        tau ~ dexp(1)
    ) , data=dat , chains=4 , cores=4 , sample=TRUE )
```

$$
\begin{aligned}
Y_i &\sim \text{Bernoulli}(p_i) \\
\text{logit}(p_i) &= \alpha_{G[i]} + \beta_X X_i + \beta_Z Z_{G[i]} \\
\alpha_j &\sim \text{Normal}(\bar{\alpha}, \tau) \\
\beta_{X,Z} &\sim \text{Normal}(0,1) \\
\bar{\alpha} &\sim \text{Normal}(0,1) \\
\tau &\sim \text{Exponential}(1)
\end{aligned}
$$

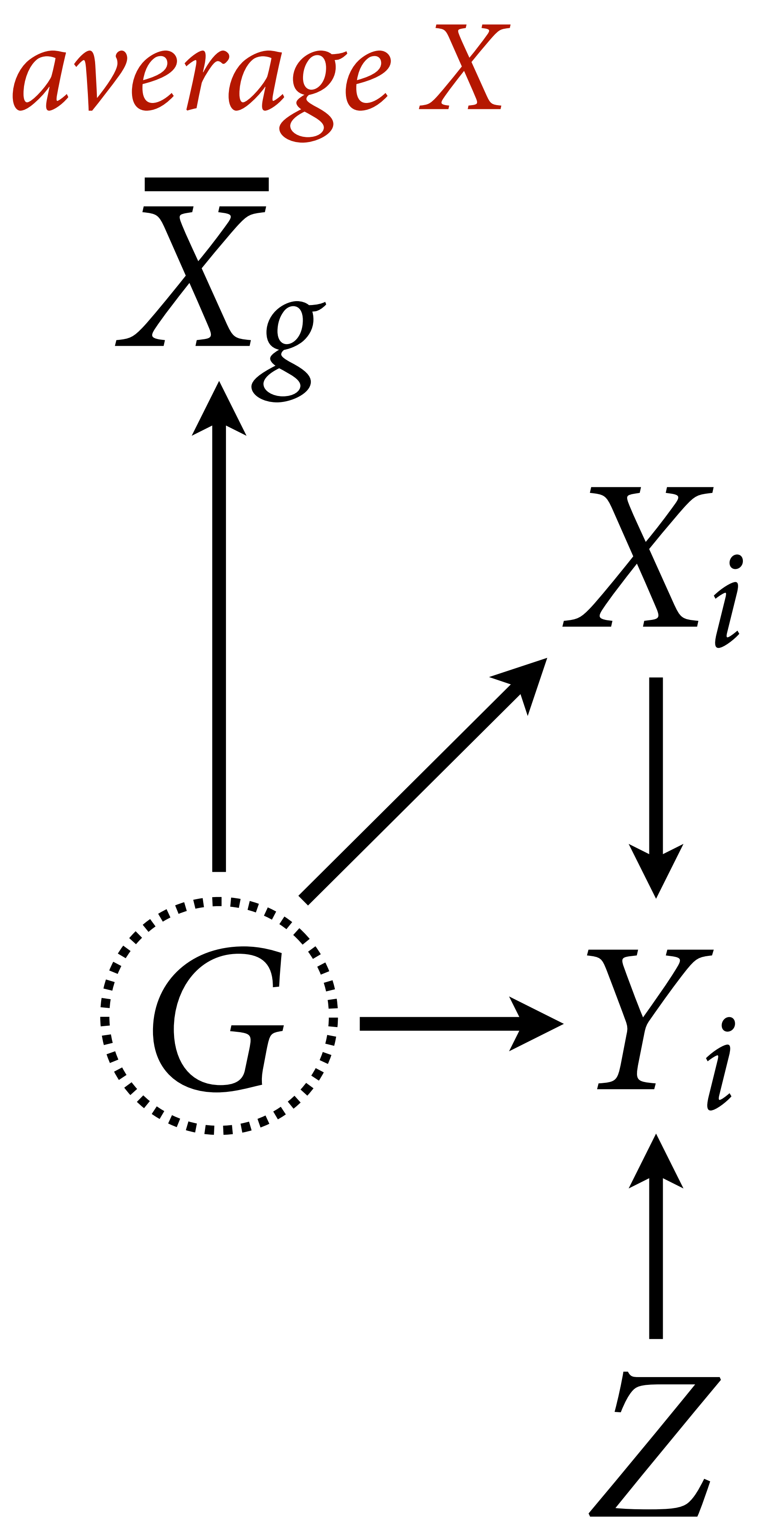


Mundlak machine

Estimate a different average rate for each group, partial pooling

Include group average X

Better X, but improper respect for uncertainty in X-bar

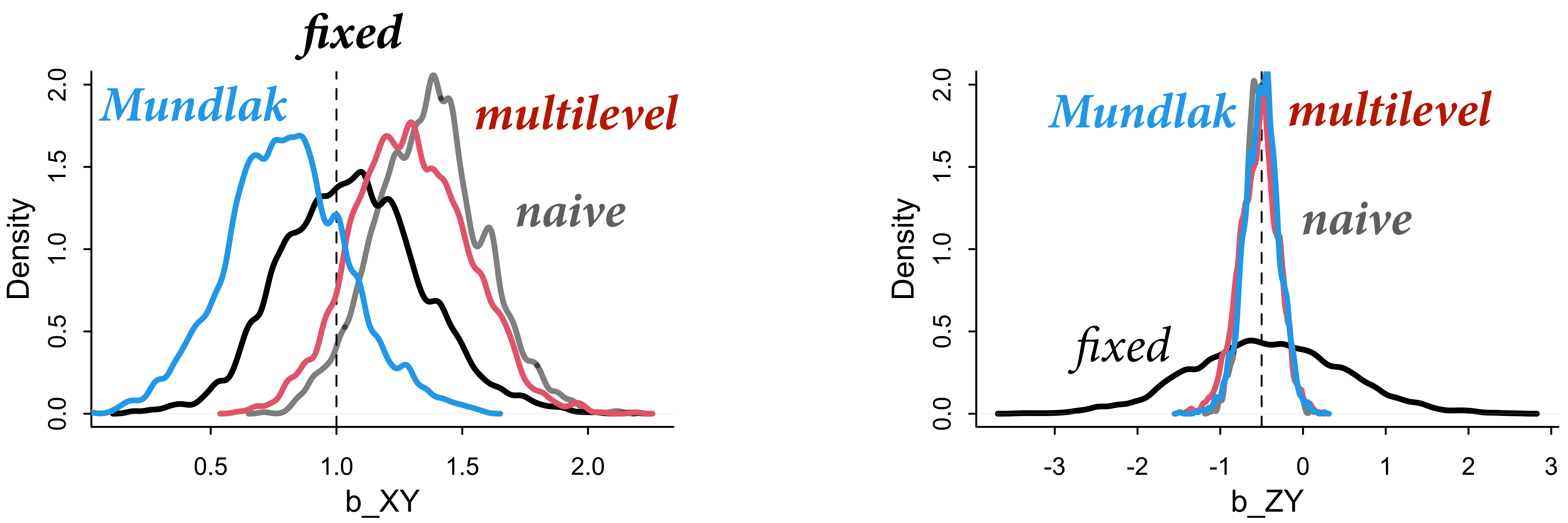


$$\text{logit}(p_i) = \alpha_{G[i]} + \beta_X X_i + \beta_Z Z_{G[i]} + \beta_{\bar{X}} \bar{X}_{G[i]}$$

Mundlak 1978 On the pooling of time series and cross-section data

```r
# The Mundlak Machine
xbar <- sapply( 1:N_groups , function(j) mean(X[g==j]) )
dat$Xbar <- xbar
mrx <- ulam(
    alist(
        Y ~ bernoulli(p),
        logit(p) <- a[g] + bxy*X + bzy*Z[g] + buy*Xbar[g],
        transpars> vector[Ng]:a <<- abar + z*tau,
        z[g] ~ dnorm(0,1),
        c(bxy,buy,bzy) ~ dnorm(0,1),
        abar ~ dnorm(0,1),
        tau ~ dexp(1)
    ) , data=dat , chains=4 , cores=4 , sample=TRUE )
```

$$Y_i \sim \text{Bernoulli}(p_i)$$
$$\text{logit}(p_i) = \alpha_{G[i]} + \beta_X X_i + \beta_Z Z_{G[i]} + \beta_{\bar{X}} \bar{X}_{G[i]}$$
$$\alpha_j \sim \text{Normal}(\bar{\alpha}, \tau)$$
$$\beta_{X,Z,\bar{X}} \sim \text{Normal}(0,1)$$
$$\bar{\alpha} \sim \text{Normal}(0,1)$$
$$\tau \sim \text{Exponential}(1)$$



Full Luxury Bayes

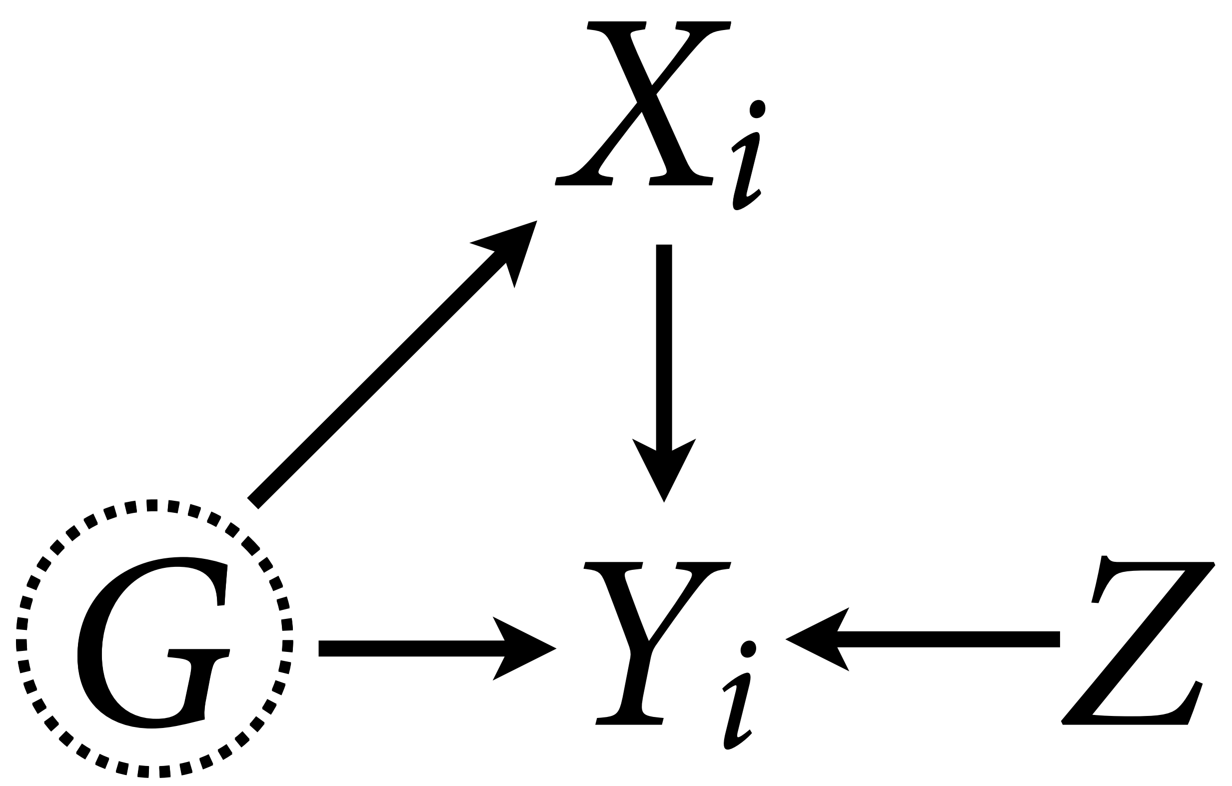
aka Latent Mundlak Machine

Just express the generative model

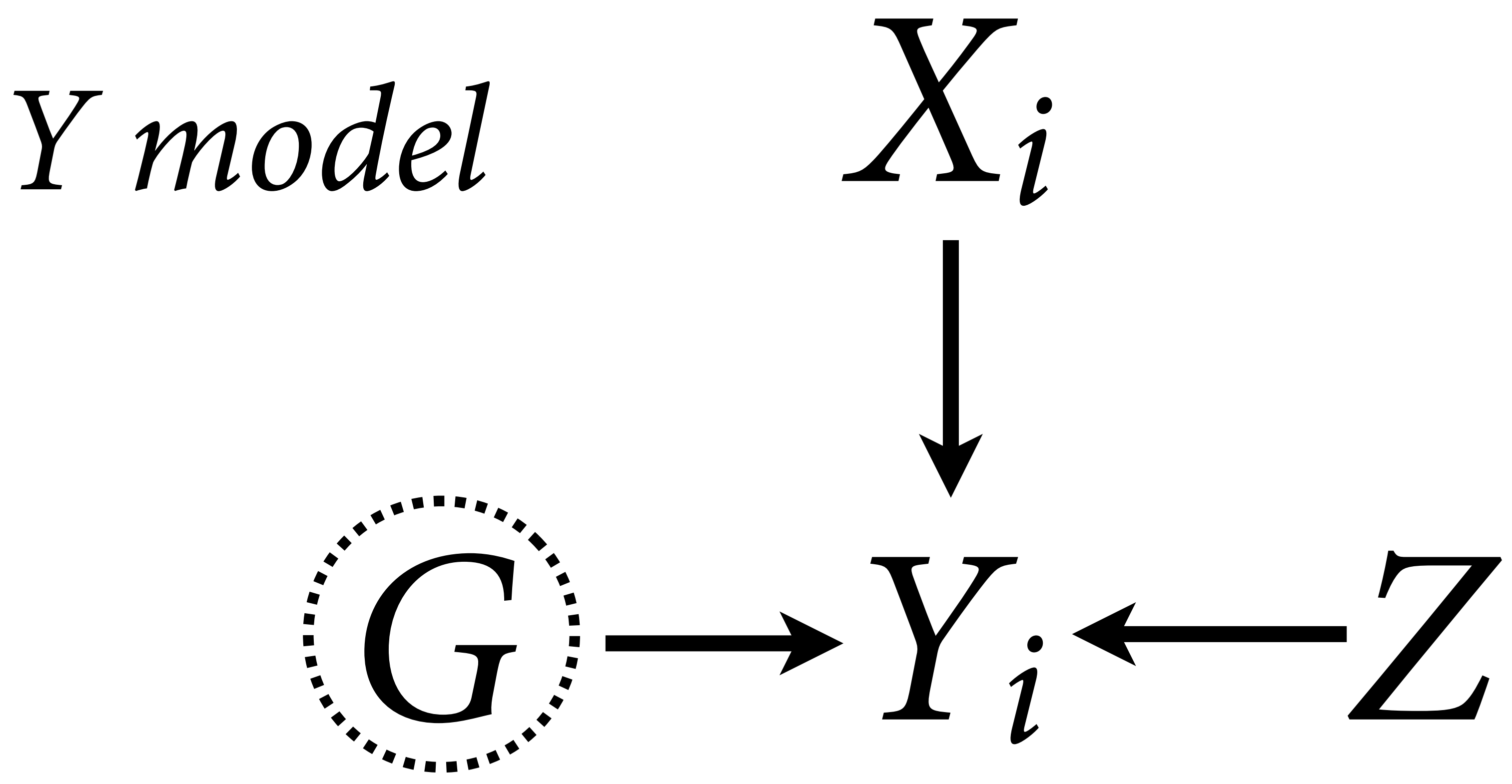
Treat G as unknown and use Xi to estimate

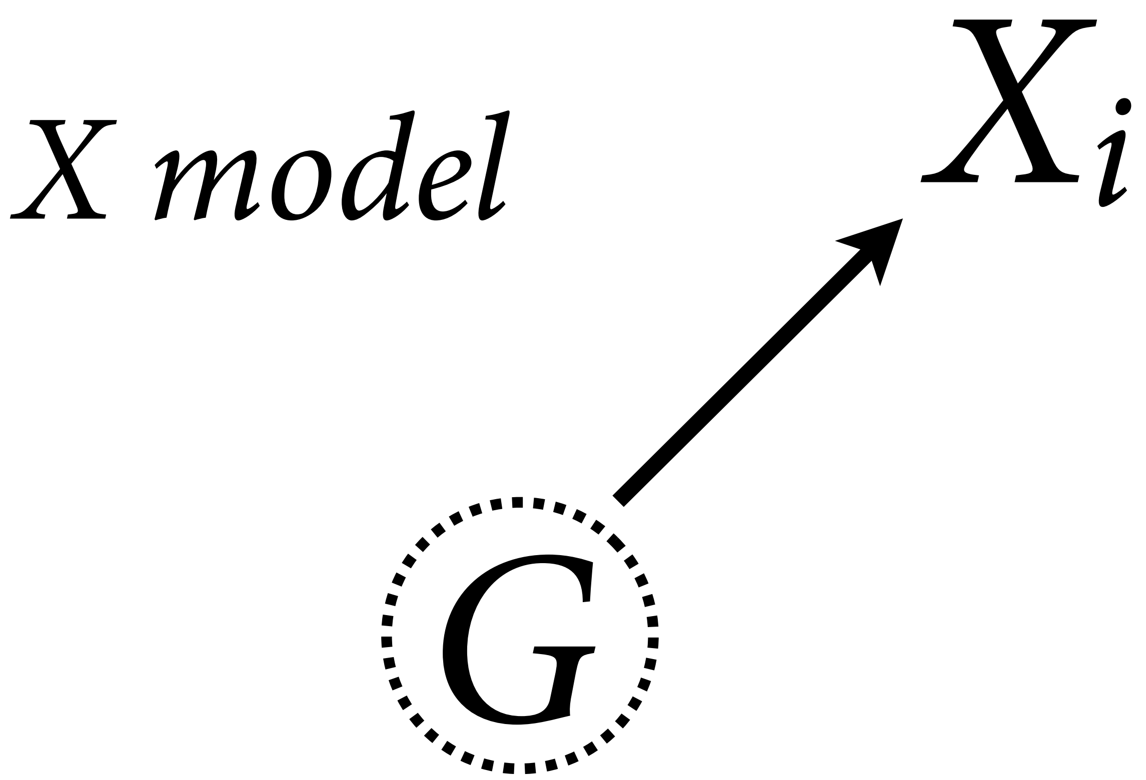
Two simultaneous regressions
(1) Estimate $Xi|\text{do}(G)$
(2) Estimate $Yi|\text{do}(Xi)$

Respects uncertainty in G

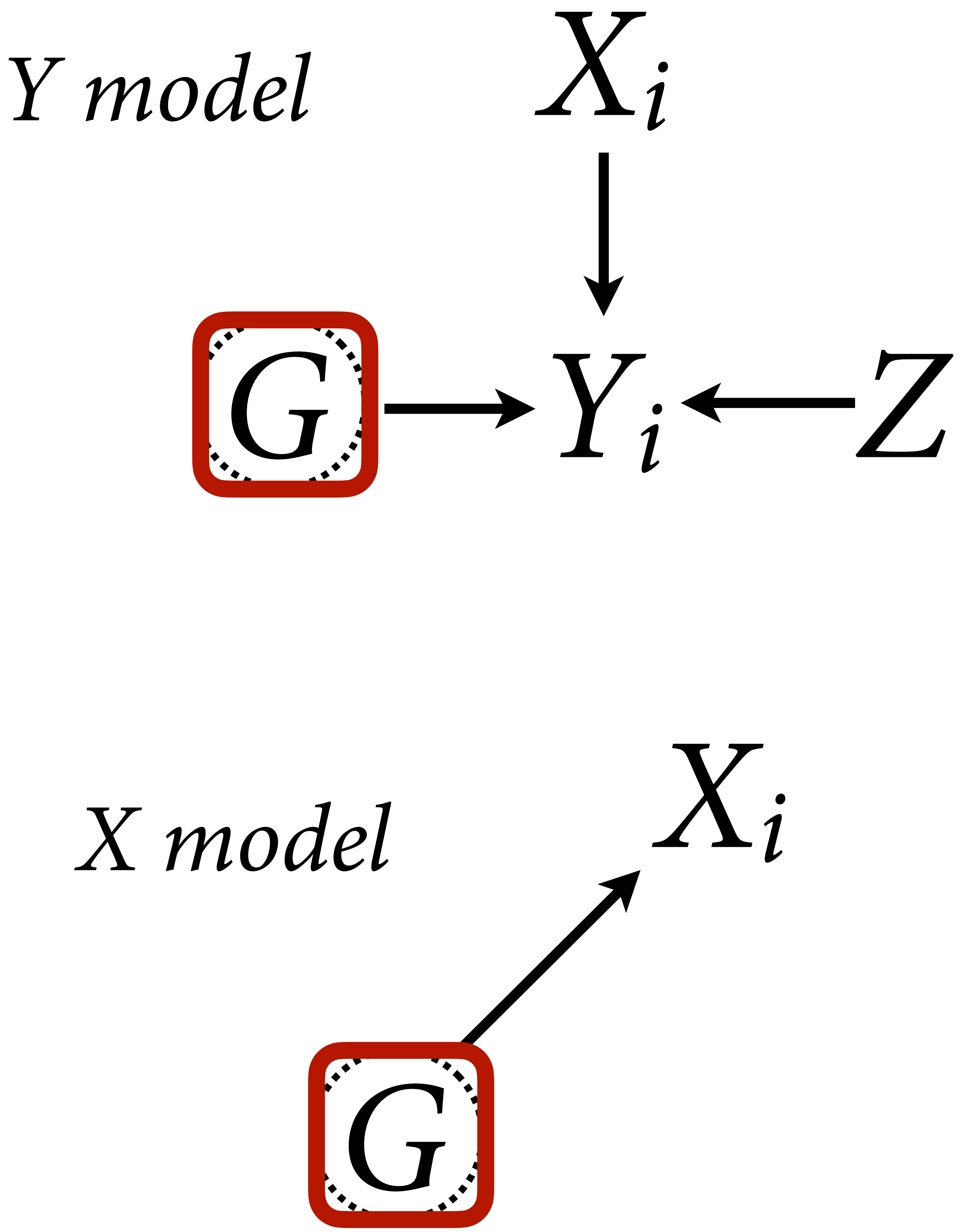


```
# The Latent Mundlak Machine
mru <- ulam(
    alist(
        # Y model
        Y ~ bernoulli(p),
        logit(p) <- a[g] + bxy*X + bzy*Z[g] + buy*u[g],
        transpars> vector[Ng]:a <<- abar + z*tau,

        # X model
        X ~ normal(mu,sigma),
        mu <- aX + bux*u[g],
        vector[Ng]:u ~ normal(0,1),

        # priors
        z[g] ~ dnorm(0,1),
        c(aX,bxy,buy,bzy) ~ dnorm(0,1),
        bux ~ dexp(1),
        abar ~ dnorm(0,1),
        tau ~ dexp(1),
        sigma ~ dexp(1)
    ) , data=dat , chains=4 , cores=4 , sample=TRUE )
```

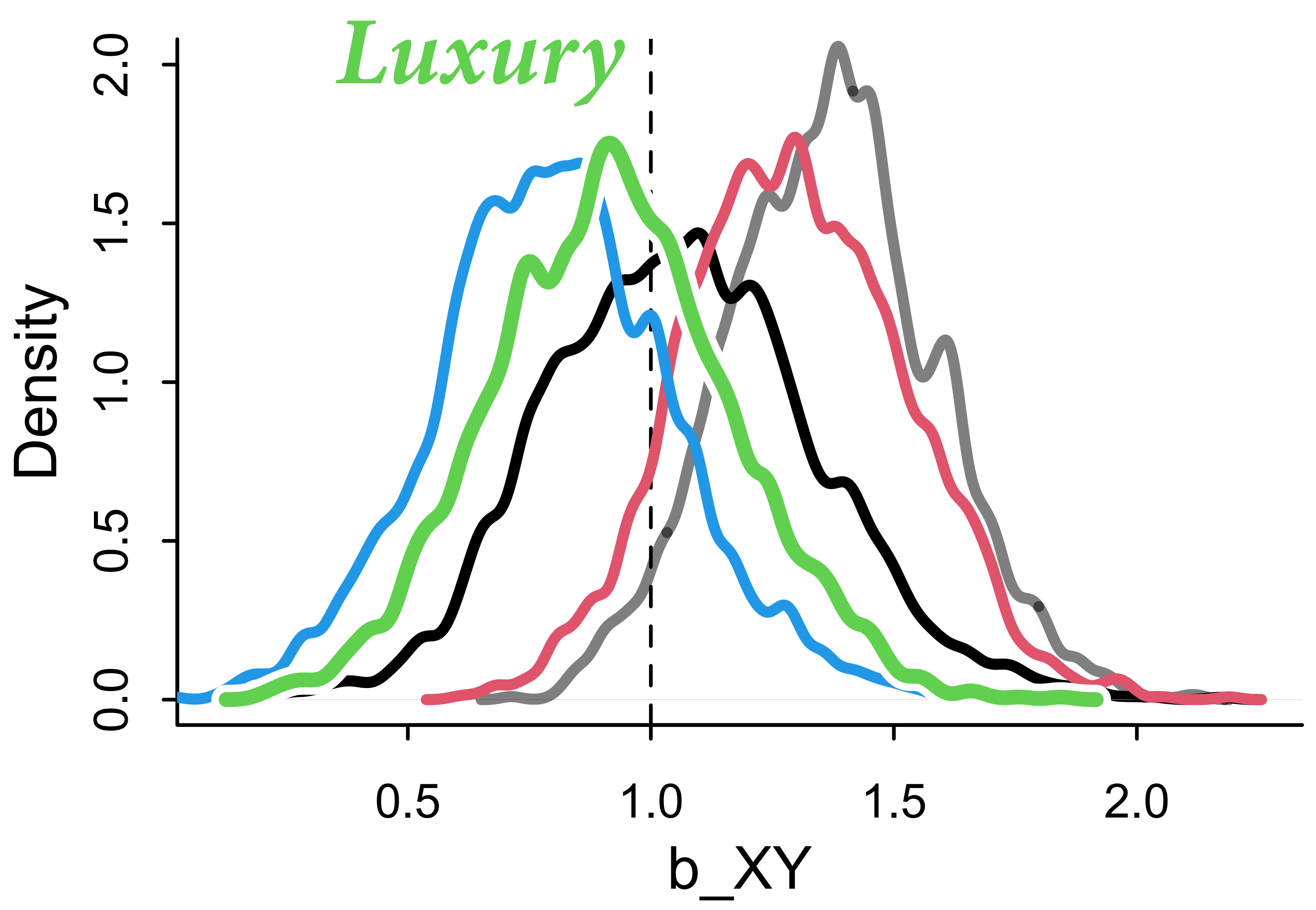


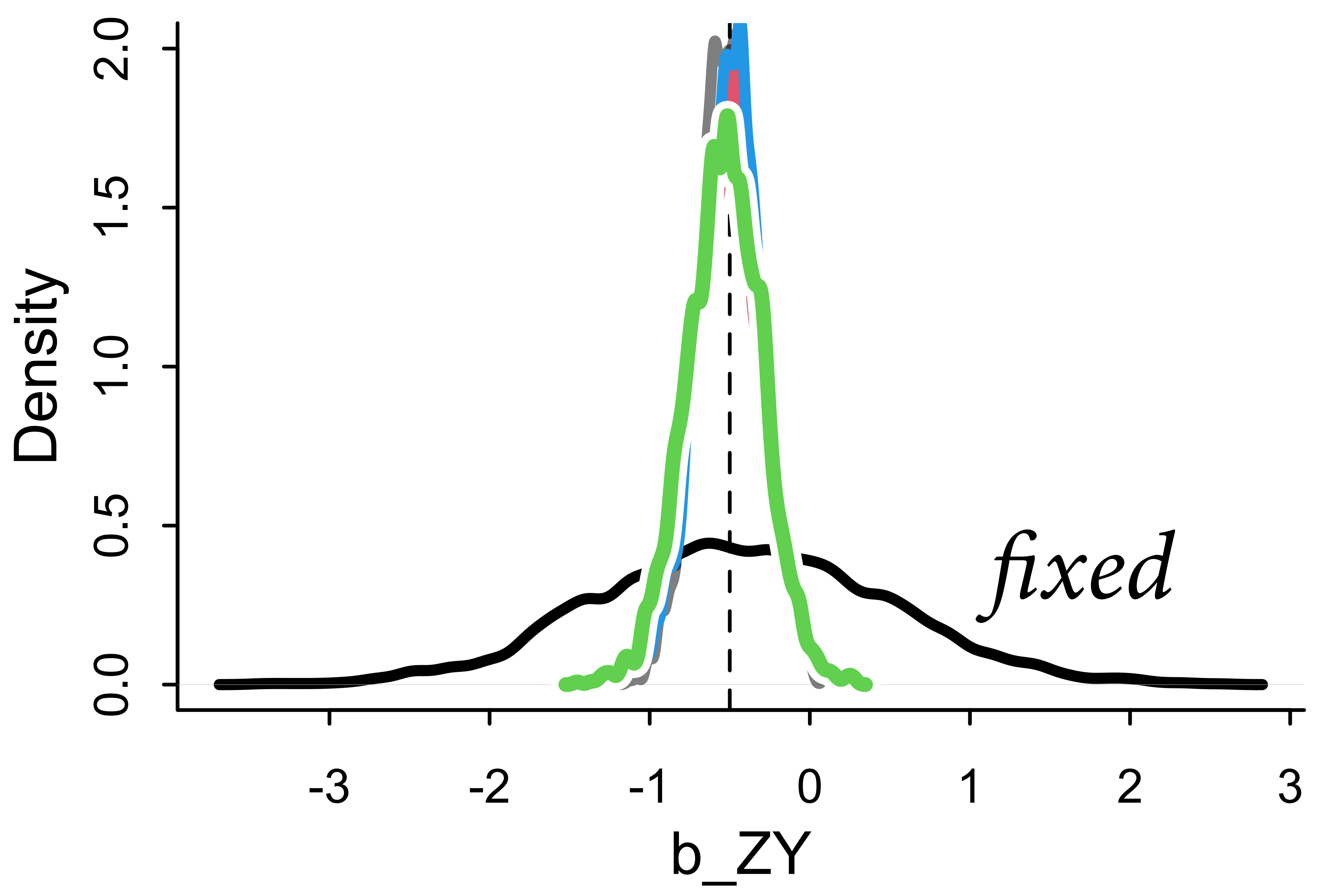


```
# The Latent Mundlak Machine
mru <- ulam(
    alist(
        # Y model
        Y ~ bernoulli(p),
        logit(p) <- a[g] + bxy*X + bzy*Z[g] + buy*u[g],
        transpars> vector[Ng]:a <<- abar + z*tau,

        # X model
        X ~ normal(mu,sigma),
        mu <- aX + bux*u[g],
        vector[Ng]:u ~ normal(0,1),

        # priors
        z[g] ~ dnorm(0,1),
        c(aX,bxy,buy,bzy) ~ dnorm(0,1),
        bux ~ dexp(1),
        abar ~ dnorm(0,1),
        tau ~ dexp(1),
        sigma ~ dexp(1)
    ) , data=dat , chains=4 , cores=4 , sample=TRUE )
```



```
# The Latent Mundlak Machine
mru <- ulam(
    alist(
        # Y model
        Y ~ bernoulli(p),
        logit(p) <- a[g] + bxy*X + bzy*Z[g] + buy*u[g],
        transpars> vector[Ng]:a <<- abar + z*tau,

        # X model
        X ~ normal(mu,sigma),
        mu <- aX + bux*u[g],
        vector[Ng]:u ~ normal(0,1),

        # priors
        z[g] ~ dnorm(0,1),
        c(aX,bxy,buy,bzy) ~ dnorm(0,1),
        bux ~ dexp(1),
        abar ~ dnorm(0,1),
        tau ~ dexp(1),
        sigma ~ dexp(1)
    ) , data=dat , chains=4 , cores=4 , sample=TRUE )
```





Random confounds

Should you use fixed effects?

Should you include average *X*?

Use a generative model, model the confound

Confounds also vary at individual level — no single solution