2016-06-04
Tokyo.Stan
Michael Betancourt's Stan Lecture

Dealing with latent discrete parameters in Stan

ITÔ, Hiroki

About me

- An end user of statistical software
- Researcher of forest ecology
  - Species composition of forests
  - Forest dynamics

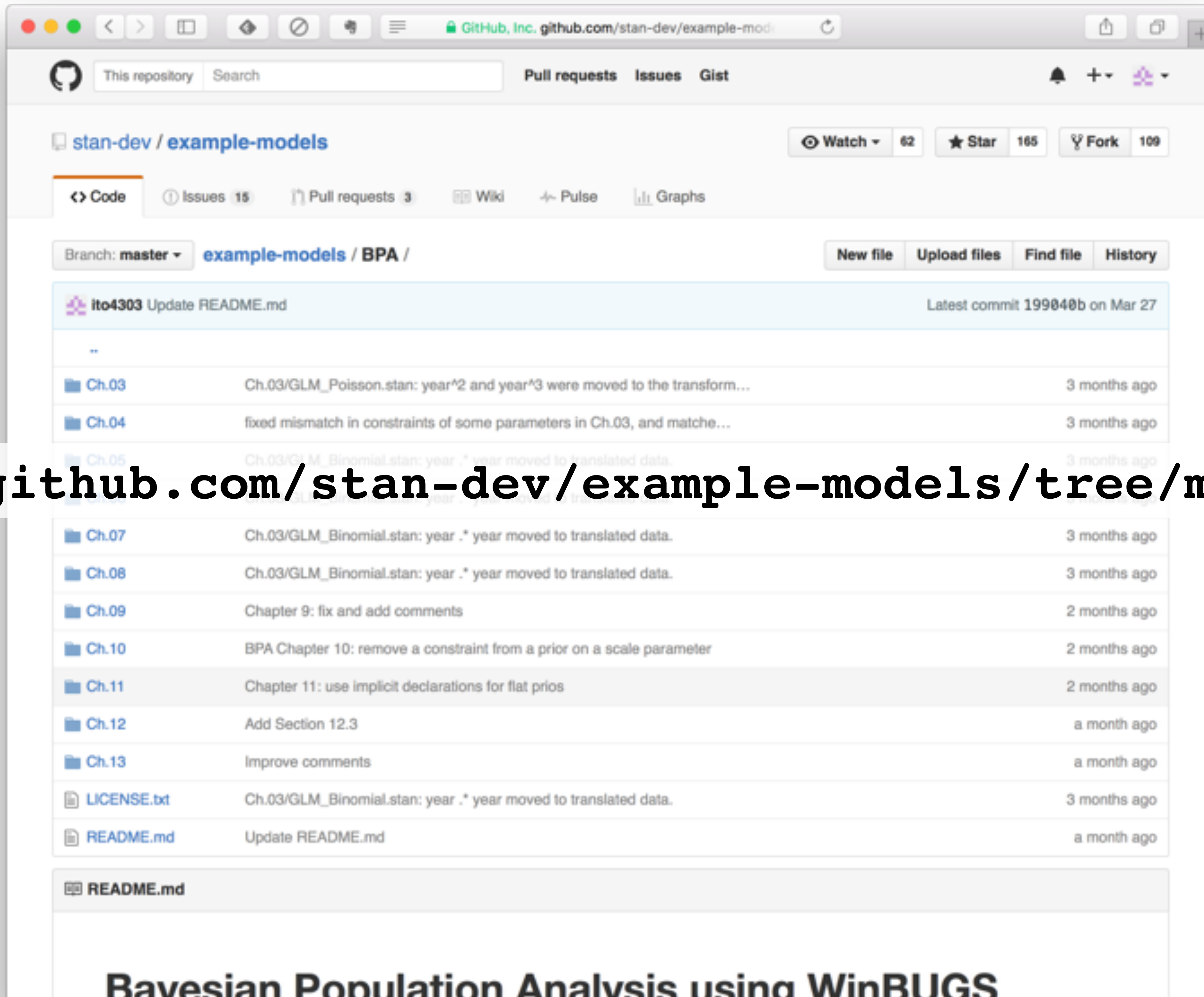
Contents

1. Introduction
   - Population Ecology
2. Examples
   1. Capture-recapture data and data augmentation
   2. Multistate model

Bayesian Population Analysis using WinBUGS (BPA)

- My colleagues and I translated “Bayesian Population Analysis using WinBUGS” by M. Kéry and M. Schaub into Japanese.
- Many practical examples for population ecology

Stan translation of BPA models



https://github.com/stan-dev/example-models/tree/master/BPA

Population ecology

- A subfield of ecology
- Studies of population
  - Changes in size (number of individuals) of animals, plants and other organisms
  - Estimation of population size, growth rate, extinction probability, etc.

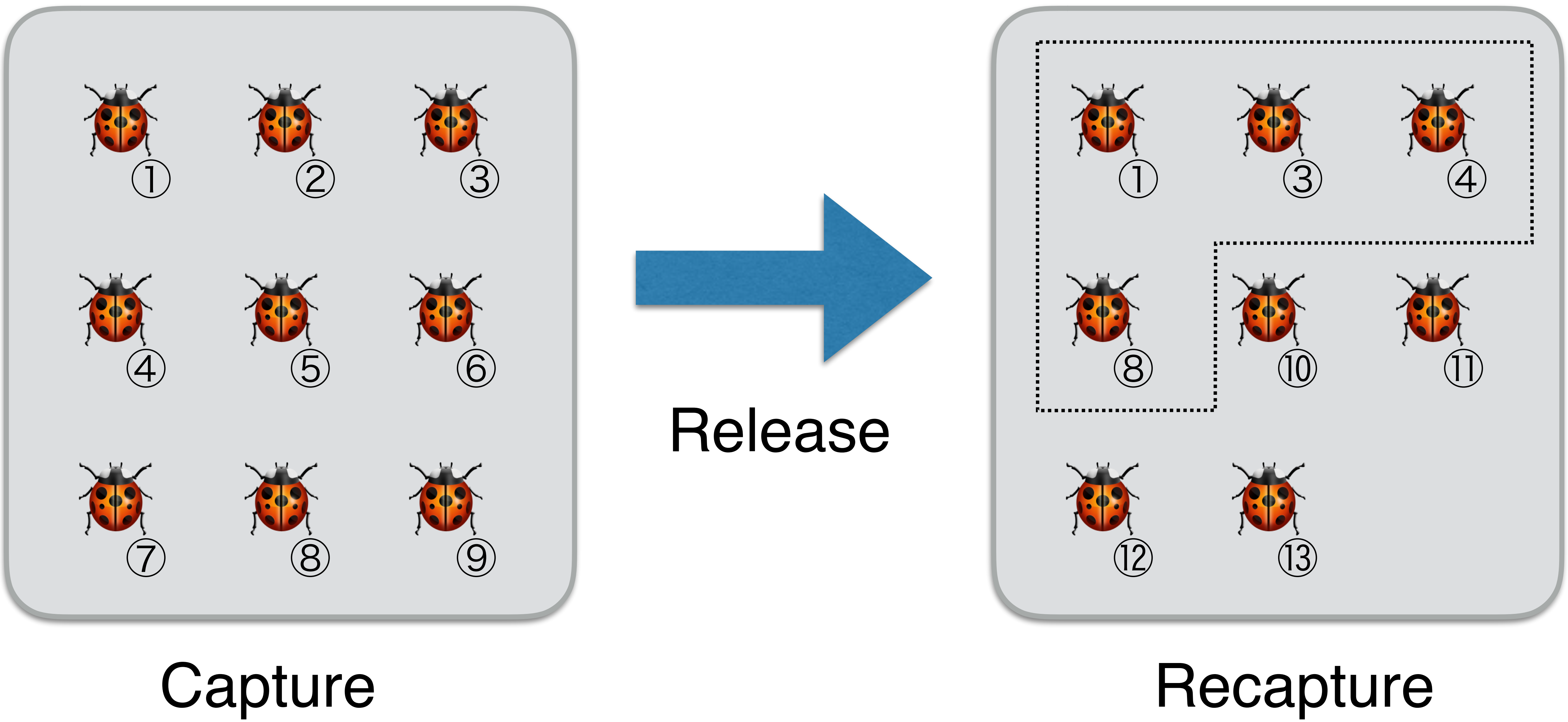
Population ecology

- **Latent discrete parameters** are often used.
  - Unobserved status
    - Present or absent
    - Dead or alive
- But Stan does **not** support discrete parameters.
- **Marginalizing out** is required to deal with discrete parameters.

Example 1

Capture-recapture data and data augmentation
(in Chapter 6 of BPA)

Capture-recapture data



Assume closed population: fixed size, no recruitment, no death, no immigration nor emigration

Data

Survey occasions

Individuals

	[,1]	[,2]	[,3]
[1,]	0	0	1
[2,]	1	1	1
[3,]	1	0	0
[4,]	1	0	1
[5,]	1	0	1
[6,]	1	0	1
	:		
[85,]	0	1	0
[86,]	0	1	1
[87,]	1	1	0

Estimation

- Population size (total number of individuals including unobserved)
- Detection (capture) probability

Data augmentation

```
         [,1] [,2] [,3]
  [1,]      0    0    1
  [2,]      1    1    1
  [3,]      1    0    0
  [4,]      1    0    1
            :
 [87,]      1    1    0
 [88,]      0    0    0
            :
[236,]      0    0    0
[237,]      0    0    0
```

Add 150 dummy records

Model

$$y_{ij} \sim \text{Bernoulli}(p_{ij}^{\text{eff}})$$
$$p_{ij}^{\text{eff}} = z_i p$$
$$z_i \sim \text{Bernoulli}(\Omega)$$

y_{ij}: observation of individual i at time j,

p_{ij}^{eff}: effective detection probability,

p: detection probability,

z_i: inclusion indicator (latent discrete parameter),

Ω: inclusion probability

BUGS

```
model {
  # Priors
  omega ~ dunif(0, 1)    # Inclusion probability
  p ~ dunif(0, 1)        # Detection probability

  # Likelihood
  for (i in 1:M){
    z[i] ~ dbern(omega)  # Inclusion indicators
    for (j in 1:T) {
      y[i, j] ~ dbern(p.eff[i, j])
      p.eff[i, j] <- z[i] * p
    }
  }

  # Derived quantities
  N <- sum(z[])
}
```

Stan

```
data {
  int<lower=0> M;                // Size of augmented data set
  int<lower=0> T;                // Number of sampling occasions
  int<lower=0,upper=1> y[M, T];  // Capture-history matrix
}

transformed data {
  int<lower=0> s[M];             // Totals in each row
  int<lower=0> C;                // Size of observed data set

  C <- 0;
  for (i in 1:M) {
    s[i] <- sum(y[i]);
    if (s[i] > 0)
      C <- C + 1;
  }
}

parameters {
  real<lower=0,upper=1> omega;  // Inclusion probability
  real<lower=0,upper=1> p;      // Detection probability
}
```

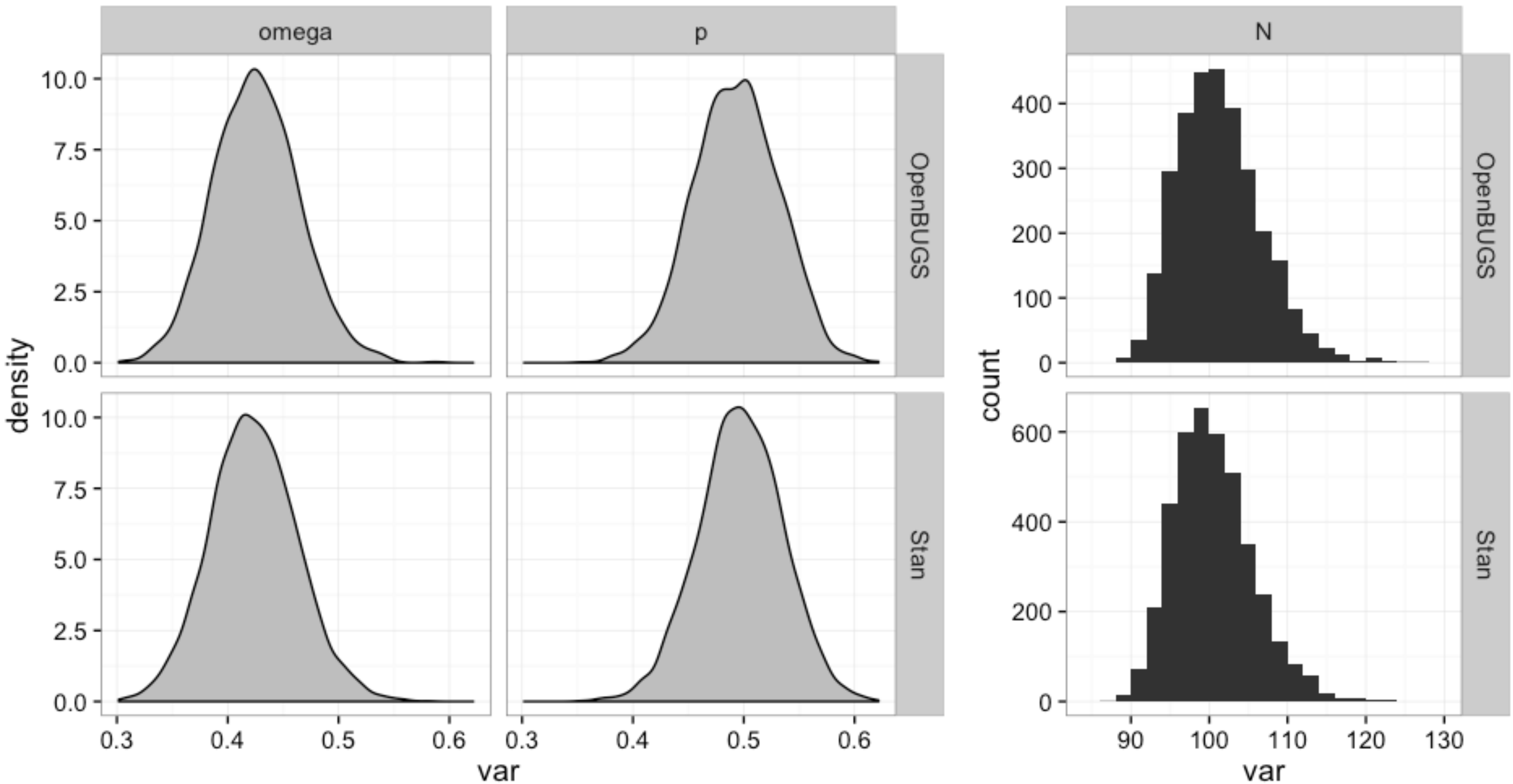
```
model {
  for (i in 1:M) {
    real lp[2];

    if (s[i] > 0) {
      // Included
      increment_log_prob(bernoulli_log(1, omega)
                         + binomial_log(s[i], T, p));
    } else {
      // Included
      lp[1] <- bernoulli_log(1, omega)
            + binomial_log(0, T, p);
      // Not included
      lp[2] <- bernoulli_log(0, omega);
      increment_log_prob(log_sum_exp(lp));
    }
  }
}
```

```
generated quantities {
  int<lower=C> N;

  N <- C + binomial_rng(M, omega * pow(1 - p, T));
}
```

Results



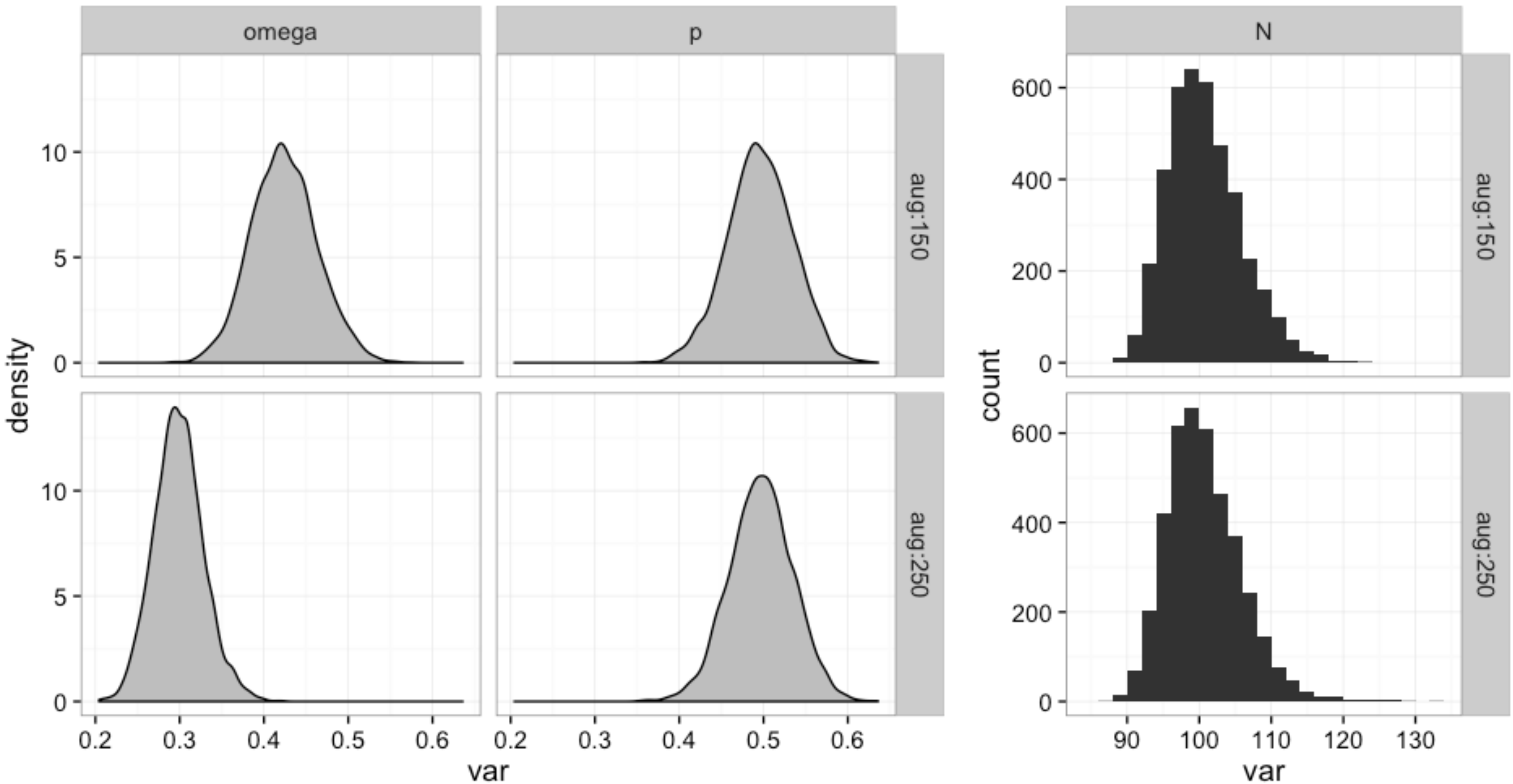
omega
p
N
OpenBUGS
Stan
OpenBUGS
Stan
density
count
var
var
10.0
7.5
5.0
2.5
0.0
0.3
0.4
0.5
0.6
400
300
200
100
0
600
90
100
110
120
130

Computing times

	OpenBUGS	Stan
Number of chains	3	4
Burn-in or warmup + iterations / chain	500 + 1000	500 + 1000
Computing time (sec)	5.9	5.8
Effective sample size of *p*	320	1867
Eff. sample size / time (sec^{-1})	54.6	319.5

Environment: 2.8 GHz Xeon W3530, Ubuntu 14.04, No parallell computing.
Compilation time is not included in Stan.
Times were measured using `system.time()`. The values are mean of 3 measurements.

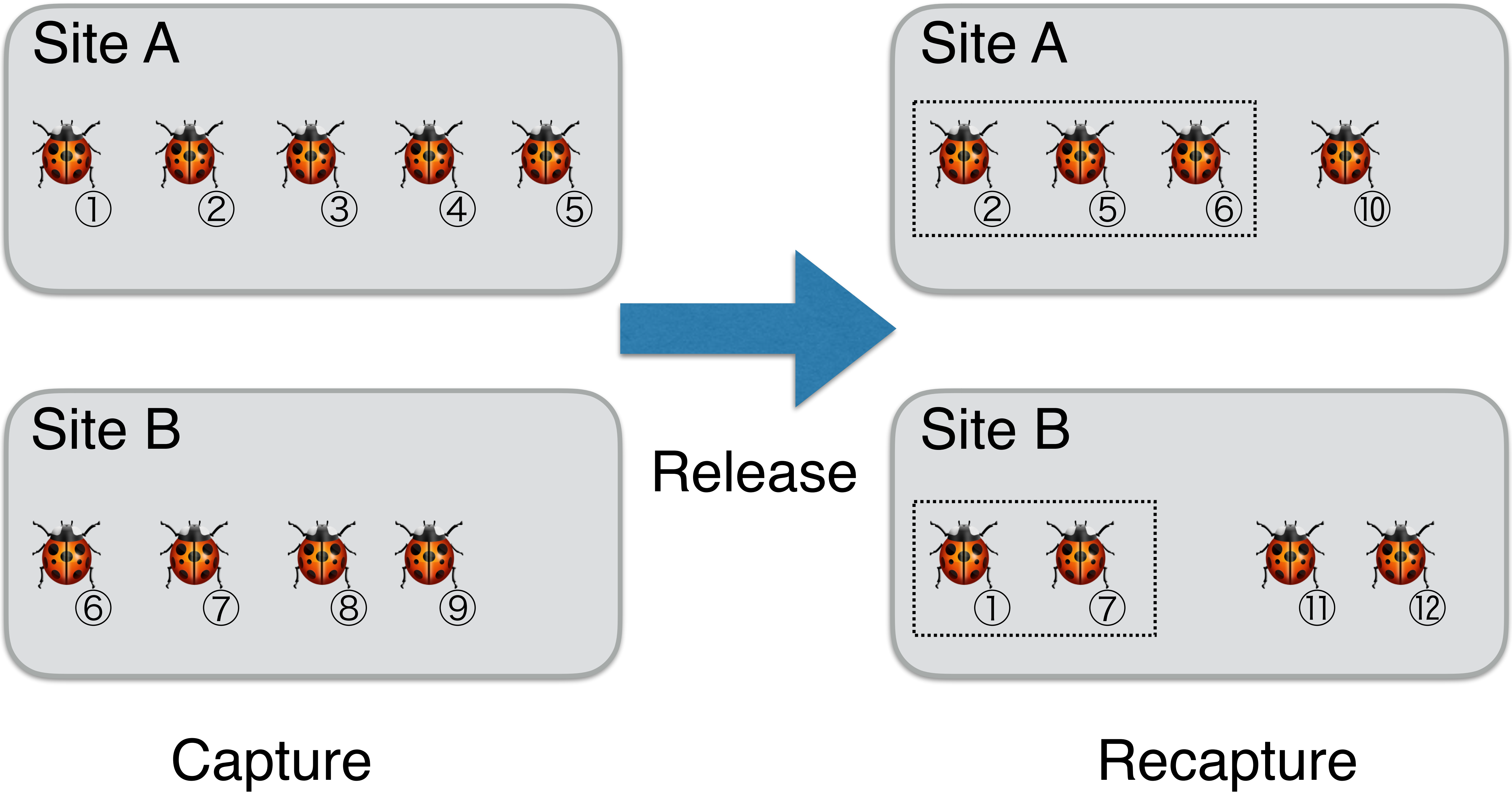
Data augmentation



Example 2

Multitstate model
(in chapter 9 of BPA)

Multistate model



Data

Survey occasion

Individuals

```
         [,1] [,2] [,3] [,4] [,5] [,6]
  [1,]      1    3    3    2    3    1
  [2,]      1    1    1    1    1    2
  [3,]      1    1    1    1    3    3
  [4,]      1    3    3    3    3    3
  [5,]      1    3    3    3    3    3
  [6,]      1    1    2    3    3    3
            :
[798,]      3    3    3    3    2    3
[799,]      3    3    3    3    2    3
[800,]      3    3    3    3    2    3
```

Values

1: seen (captured) at site A, 2: seen (captured) at site B,

3: not seen (not captured)

Estimation

- Survival probability (at site A and B)
- Movement probability (from site A to B and B to A)
- Detection (capture) probability (at site A and B)

State transition

		State at time t+1		
		Site A	Site B	Dead
State at time t	Site A	$\phi_A(1-\psi_{AB})$	$\phi_A\psi_{AB}$	$1-\phi_A$
	Site B	$\phi_B\psi_{BA}$	$\phi_B(1-\psi_{BA})$	$1-\phi_B$
	Dead	0	0	1

ϕ_A: survival probability at site A, ϕ_B: survival probability at site B,
ψ_{AB}: movement probability from A to B,
ψ_{BA}: movement probability from B to A

Observation

		Observation at time t		
		Site A	Site B	Not seen
State at time t	Site A	p_A	0	$1-p_A$
	Site B	0	p_B	$1-p_B$
	Dead	0	0	1

p_A: detection probability at site A,
p_B: detection probability at site B

Model

$$z_{i,t} \mid z_{i,t-1} \sim \text{Categorical}(S[z_{i,t-1},\ ])$$
$$y_{i,t} \mid z_{i,t} \sim \text{Categorical}(O[z_{i,t},\ ])$$

$z_{i,t}$: latent state of individual i at time t, $z \in \{1, 2, 3\}$

$y_{i,t}$: observation of individual i at time t, $y \in \{1, 2, 3\}$

$S[,]$: state transition probability matrix

$O[,]$: observation probability matrix

BUGS

```
model {
  :
  # Define probabilities of state S(t+1) given S(t)
  ps[1, 1] <- phiA * (1 - psiAB)
  ps[1, 2] <- phiA * psiAB
  ps[1, 3] <- 1 - phiA
  ps[2, 1] <- phiB * psiBA
  ps[2, 2] <- phiB * (1 - psiBA)
  ps[2, 3] <- 1 - phiB
  ps[3, 1] <- 0
  ps[3, 2] <- 0
  ps[3, 3] <- 1

  # Define probabilities of O(t) given S(t)
  po[1, 1] <- pA
  po[1, 2] <- 0
  po[1, 3] <- 1 - pA
  po[2, 1] <- 0
  po[2, 2] <- pB
  po[2, 3] <- 1 - pB
  po[3, 1] <- 0
  po[3, 2] <- 0
  po[3, 3] <- 1
  :
```

```
for (i in 1:nind) {
   # Define latent state at first capture
   z[i, f[i]] <- y[i, f[i]]
   for (t in (f[i] + 1):n.occasions) {
     # State process
     z[i, t] ~ dcat(ps[z[i, t - 1], ])
     # Observation process
     y[i, t] ~ dcat(po[z[i, t], ])
   }
}
```

`f[ ]`: array containing first capture occasion

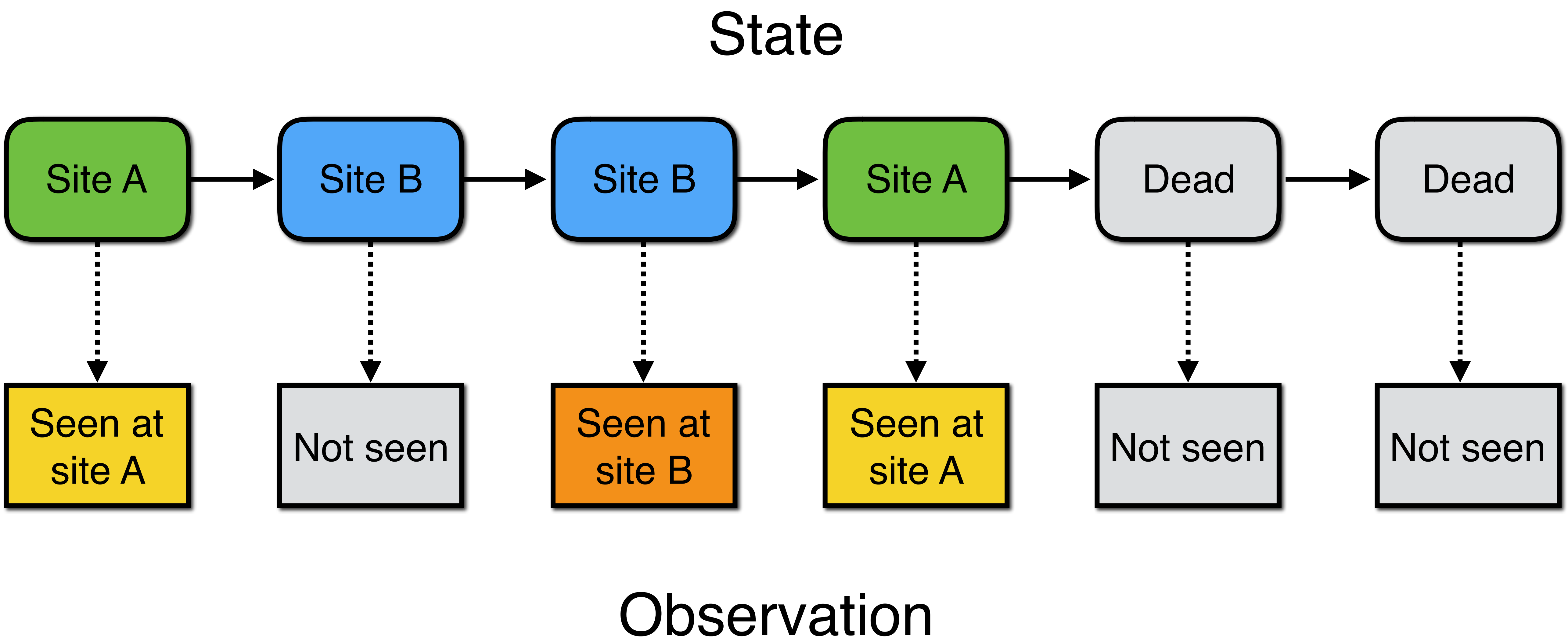
`ps[ , ]`: state transition matrix

`po[ , ]`: observation matrix

Stan

Treating as a Hidden Markov model

Hidden Markov model

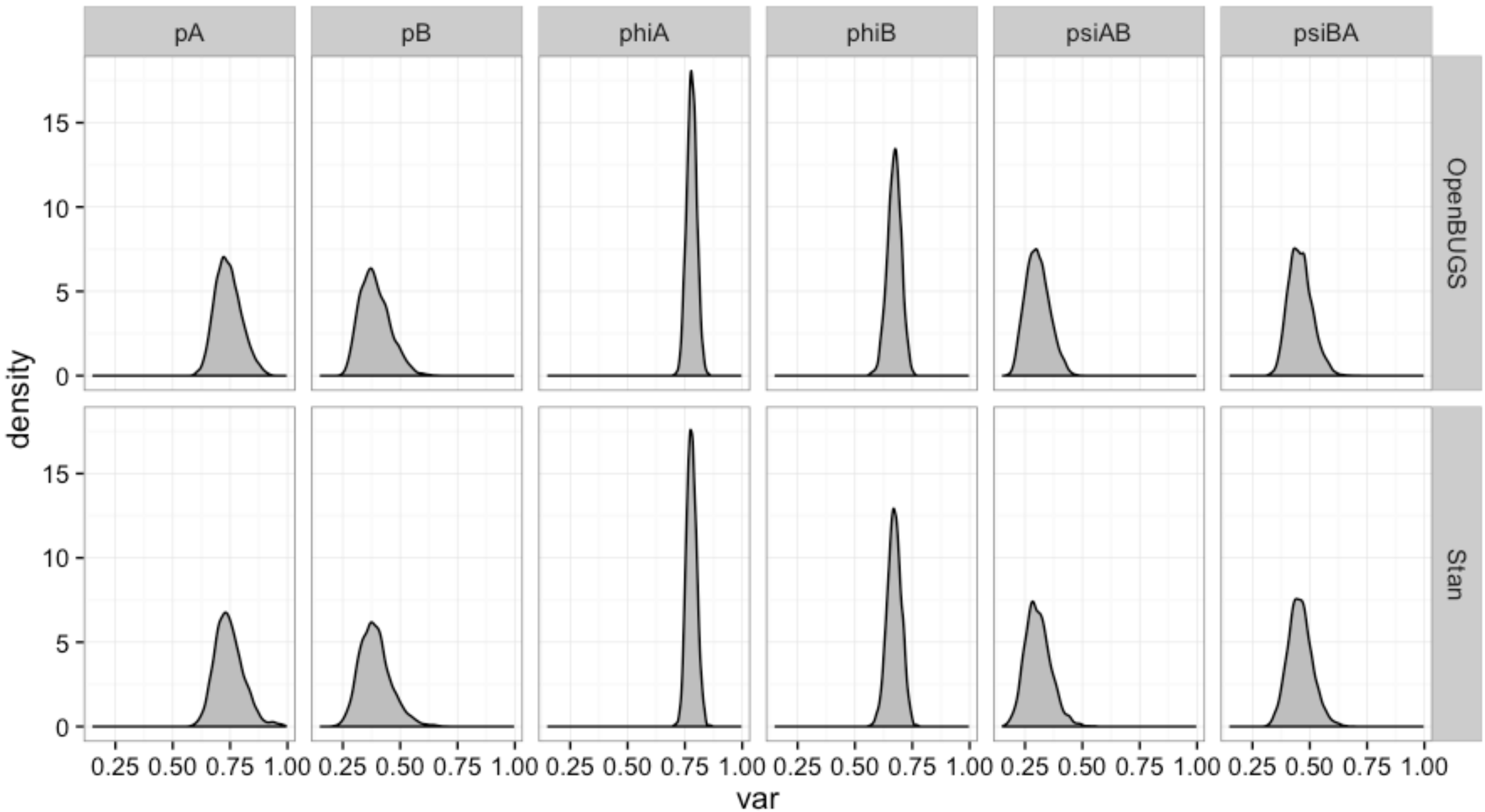


```
model {
  real acc[3];
  vector[3] gamma[n_occasions];

  // See Stan Modeling Language User's Guide and Reference Manual
  for (i in 1:nind) {
    if (f[i] > 0) {
      for (k in 1:3)
        gamma[f[i], k] <- (k == y[i, f[i]]);

      for (t in (f[i] + 1):n_occasions) {
        for (k in 1:3) {
          for (j in 1:3)
            acc[j] <- gamma[t - 1, j] * ps[j, k]
                    * po[k, y[i, t]];
          gamma[t, k] <- sum(acc);
        }
      }
      increment_log_prob(log(sum(gamma[n_occasions])));
    }
  }
}
```

Results



pA
pB
phiA
phiB
psiAB
psiBA
OpenBUGS
Stan
density
0
5
10
15
0.25 0.50 0.75 1.00
var

Computing times

	OpenBUGS	Stan
Number of chains	3	4
Burn-in or warmup + iterations / chain	500 + 2000	500 + 1000
Computing time (sec)	327.9	193.5
Effective sample size of *p*A	140	649
Eff. sample size / time (sec^{-1})	0.4	3.4

Environment: 2.8 GHz Xeon W3530, Ubuntu 14.04, No parallell computing.
Compilation time is not included in Stan.
Times were measured using `system.time()`. The values are mean of 3 measurements.

Binomial-mixture model (in Chapter 12 of BPA)

$$N_i \sim \text{Poisson}(\lambda)$$
$$y_{i,j} \sim \text{Binomial}(N_i, p)$$

N_i: latent population size at site i

$y_{i,j}$: observed counts at site i at time j

λ: mean population size

p: detection probability

Stan code

```
data {
  int<lower=0> R;        // Number of sites
  int<lower=0> T;        // Number of temporal replications
  int<lower=0> y[R, T]; // Counts
  int<lower=0> K;        // Upper bounds of population size
}
model {
  :
  // Likelihood
  for (i in 1:R) {
    vector[K+1] lp;

    for (n in max_y[i]:K) {
      lp[n + 1] <- binomial_log(y[i], n, p)
        + poisson_log(n, lambda);
    }
    increment_log_prob(log_sum_exp(lp[(max_y[i] + 1):(K + 1)]));
  }
  :
}
```

$$L(\lambda, p \mid y) = \sum_{n=0}^{\infty} \text{Binomial}(y \mid n, p)\text{Poisson}(n \mid \lambda)$$

Computing times

	OpenBUGS	Stan (K=100)
Number of chains	3	4
Burn-in or warmup + iterations / chain	200 + 1000	500 + 1000
Computing time (sec)	25.4	1007.8
Effective sample size of λ	3000	401
Eff. sample size / time (sec^{-1})	118.3	0.4

Environment: 2.8 GHz Xeon W3530, Ubuntu 14.04, No parallell computing.
Compilation time is not included in Stan.
Times were measured using `system.time()`. The values are mean of 3 measurements.

Summary

- Stan can deal with models including discrete parameters by marginalizing.
  - Divide and sum up every cases
  - However, the formulation used are less straightforward.