HOKKAIDO UNIVERSITY

Title	Genetic consequences of habitat fragmentation in a perennial plant Trillium camschatcense are subjected to its slow-paced life history
Author(s)	Tsuzuki, Yoichi; Sato, Mitsuhiko P.; Matsuo, Ayumi et al.
Citation	Population Ecology, 64(1), 5-18 https://doi.org/10.1002/1438-390X.12093
Issue Date	2022-01
Doc URL	https://hdl.handle.net/2115/87602
Rights	This is the peer reviewed version of the following article: Yoichi Tsuzuki, Mitsuhiko P. Sato, Ayumi Matsuo, Yoshihisa Suyama, Masashi Ohara. Genetic consequences of habitat fragmentation in a perennial plant Trillium camschatcense are subjected to its slow-paced life history. Population Ecology, 2022; 64(1): 5-18, which has been published in final form at https://doi.org/10.1002/1438-390X.12093. This article may be used for non-commercial purposes in accordance with Wiley Terms and Conditions for Use of Self-Archived Versions.
Type	journal article
File Information	SupportingInformation.pdf, Supporting Information

Hokkaido University Collection of Scholarly and Academic Papers : HUSCAP

Supporting Information 1 Threshold for the classification of 3L stage

To estimate the leaf size range of 3L stage, we sampled leaves of 3L individuals from both FUJ and SAK population. We particularly sampled extremely small and large ones to clarify the range limit. As a result, leaf size (leaf length × leaf width) ranged from 8.67 to 289.15 cm^2, with its center at approximately 150 cm^2. Therefore, we used the threshold of 150 cm^2 to divide 3L into small (3L-1) and large (3L-2) stages.

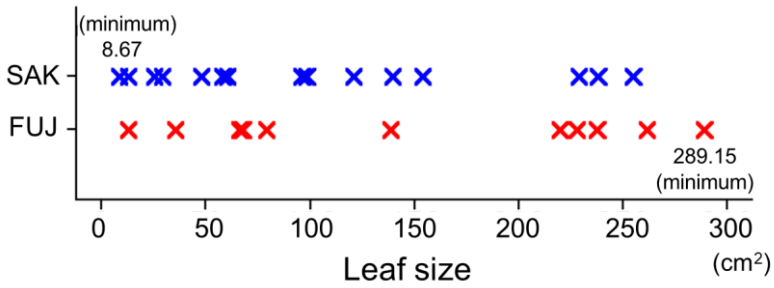

Figure S1. Leaf size of 3L stage in FUJ and SAK population.

Supporting Information 2 Analysis results for various parameter settings in Stacks

2.1 Comparison of genetic diversity and inbreeding coefficient among stages

We tested the parameter dependence of inter-stage difference in genetic diversity and inbreeding coefficient to m, M, N, r, and the missing rate of SNPs per sample. It was shown that for a large part of the parameter settings, our main two results hold: (1) genetic diversity of SD was lower than that of FL and 3L-2 in SAK population, and (2) there was no statistically significant difference in FUJ population.

(a) SAK

m	M	N	SD vs 1L	SD vs 3L-1	SD vs 3L-2	SD vs FL	1L vs 3L-1	1L vs 3L-2	1L vs FL	3L-1 vs 3L-2	3L-1 vs FL	3L-2 vs FL
3	1	3			*1	*1						
3	2	4			*2	*2		*3		*3		
3	3	5			*1	*1				*4		
5	1	3			*5	*6						
5	2	4			*1	*1						
5	3	5			*7	*5						
8	1	3			*8	*9						
8	2	4			*10	*5		*5				
8	3	5			*8	*5		*11				

(b) FUJ

m	M	N	SD vs 1L	SD vs 3L-1	SD vs 3L-2	SD vs FL	1L vs 3L-1	1L vs 3L-2	1L vs FL	3L-1 vs 3L-2	3L-1 vs FL	3L-2 vs FL
3	1	3										
3	2	4										
3	3	5										
5	1	3										
5	2	4										
5	3	5										
8	1	3			*12							
8	2	4			*1	*13						
8	3	5										

Table S1 Parameter dependence of the inter-stage comparisons of genetic diversity and inbreeding coefficient to m, M, and N. r was kept to 0.8. Pairs of life history stages which showed statistically significant difference in at least one index are marked with asterisk in (a) SAK and (b) FUJ population. The asterisks were labelled with number to indicate which indices were significantly

different. 1: A_E, H_O, H_S, H_E, sMLH; 2: A_R, A_E, H_O, H_S, H_E, sMLH; 3: H_S, H_E; 4: A_E, H_S; 5: H_O, sMLH, F_{IS}; 6: H_O, sMLH, F_{IS}; 7: H_O, H_E, sMLH; 8: A_R, A_E, H_O, H_S, H_E, sMLH, F_{IS}; 9: A_E, H_O, H_S, H_E, sMLH, F_{IS}; 10: A_R, H_O, H_S, H_E, sMLH, F_{IS}; 11: sMLH; 12: A_E, H_S, H_E; 13: H_E

(a) SAK											
r	missing rate	SD vs 1L	SD vs 3L-1	SD vs 3L-2	SD vs FL	1L vs 3L-1	1L vs 3L-2	1L vs FL	3L-1 vs 3L-2	3L-1 vs FL	3L-2 vs FL
0.7	-			*1	*1		*2		*2		
0.8	-			*1	*1		*3		*3		
0.9	-										
0.8	0.3			*1	*4						
(b) FUJ											
r	missing rate	SD vs 1L	SD vs 3L-1	SD vs 3L-2	SD vs FL	1L vs 3L-1	1L vs 3L-2	1L vs FL	3L-1 vs 3L-2	3L-1 vs FL	3L-2 vs FL
0.7	-										
0.8	-										
0.9	-										
0.8	0.3										

Table S2 Parameter dependence of the inter-stage comparisons of genetic diversity and inbreeding coefficient to r and the missing rate. As for the missing rate, samples that failed to be genotypes for more than 0.3 of the total SNPs were excluded from the analysis. m, M, N were kept to 3, 2, 4, respectively, which are the default of Stacks *de novo* assembly. Pairs of life history stages which showed statistically significant difference in at least one index are marked with asterisk. The asterisks were labelled with number to indicate which indices were significantly different. 1: A_R, A_E, H_O, H_S, H_E, sMLH; 2: H_E; 3: H_S, H_E; 4: A_E, H_O, H_S, H_E, sMLH

2.2 Estimates of N_b

We estimated N_b for each parameter setting (Table S3). It was shown that for a large part of the settings, Nb was smaller in SAK than in FUJ population.

m	M	N	r	missing rate	SAK		FUJ	
					estimates	95 % CI	estimates	95 % CI
3	1	3	0.8	-	16.8	13.2–22.3	37.4	24.9–69.3
3	2	4	0.8	-	18.0	14.0–24.5	25.8	18.9–39.1
3	3	5	0.8	-	17.2	13.4–23.3	30.7	21.3–51.6
5	1	3	0.8	-	30.8	18.2–80.2	28.9	17.5–68.4
5	2	4	0.8	-	17.5	12.0–29.5	32.6	18.9–88.9
5	3	5	0.8	-	19.1	12.6–34.9	31.5	18.3–86.1
8	1	3	0.8	-	11.0	10.0–20.9	36.5	15.2–10^7
8	2	4	0.8	-	15.7	10.0–42.9	25.4	12.2–195.3
8	3	5	0.8	-	16.5	10.0–52.0	26.7	12.3–381.0
3	2	4	0.7	-	18.0	14.0–24.5	25.8	18.9–39.1
3	2	4	0.9	-	29.9	18.6–64.9	43.7	23.8–165.3
3	2	4	0.8	0.3	20.2	15.2–28.8	30.4	21.2–50.5

Table S3 Parameter dependence of N_b and its 95 % confidence interval (95 % CI) in both SAK and FUJ population.

Supporting Information 3 R code used for calculating genetic diversity and inbreeding coefficient

```
##-------------------------------------------------------
## rarefaction used for calculating Ar
##
## x: genotype data for all samples (data.frame)
## min: minimum number of observations per class
##-------------------------------------------------------

raref <- function(x,min) {
  nn <- sum(x)
  dum <- choose(nn - x, min)/choose(nn, min)
  dum[is.na(dum)] <- 0
  return(sum(1 - dum))
}

##-------------------------------------------------------
## calculating Ar, Ae, Ho, Hs, Fis
##
## data: genotype data for all samples (data.frame)
##-------------------------------------------------------

gstats2 <- function (data, diploid = TRUE){
  nloc <- dim(data[, -1])[2]
  all.count <- allele.count(data)
  in.count <- ind.count(data)
  n <- t(in.count)
  min.n <- 2 * min(in.count, na.rm = TRUE)
  a <- lapply(all.count, fun1 <- function(x) apply(x, 2, raref, min=min.n))
  Ar <- matrix(unlist(a), nrow = nloc, byrow = TRUE)
  rownames(Ar) <- names(data)[-1]
  mynas <- which(is.na(n))
  Ar[mynas] <- NA
  p <- pop.freq(data, diploid)
  dum <- getal.b(data[, -1])
  Ho <- dum[, , 1] == dum[, , 2]
  sHo <- (1 - t(apply(Ho, 2, fun <- function(x) tapply(x,
                                                data[, 1], mean, na.rm =
TRUE))))
  sp2 <- lapply(p, fun <- function(x) apply(x, 2, fun2 <- function(x)
sum(x^2)))
  sp2 <- matrix(unlist(sp2), nrow = nloc, byrow = TRUE)
  Hs <- (1 - sp2 - sHo/2/n)
  Hs <- n/(n - 1) * Hs
  Fis <- 1 - sHo/Hs
  Fis[is.nan(Fis)] <- NA
  Ae <- 1/sp2
  lbind <- list(Ar,Ae,Fis,sHo,Hs)
  res <- sapply(lbind, function(y) apply(y,2,mean,na.rm=T))
  rownames(res) <- levels(data[,1])
  colnames(res) <- c("Ar","Ae","Fis","Ho","Hs")
  all.res <- list(n.ind.samp = n, pop.freq = p, Ho = sHo, Hs = Hs,
               Fis = Fis, Ae = Ae, Ar = Ar,
               result = res)
  all.res
}
```

Supporting Information 4 R code used for permutation test

```
##-------------------------------------------------------
## permutation test for Ar, Ae, Ho, Hs, Fis
##
## x: genotype data for all samples (data.frame)
## y: observed values (matrix or data.frame)
## rep: the number of iterations
##-------------------------------------------------------
permtest_gstats2 <- function(x,y,rep){
  start <- Sys.time()
  pair <- combn(levels(factor(x$pop)),2)
  p <- NULL
  for(i in 1:dim(pair)[2]){
    x.i1 <- x[x$pop==pair[1,i],]
    x.i2 <- x[x$pop==pair[2,i],]
    x.i <- rbind(x.i1,x.i2)
    dif.i <- matrix(0,rep+1,5)
    dif.i[1,] <- y[rownames(y)==pair[1,i],]-y[rownames(y)==pair[2,i],]
    pb <- txtProgressBar(min=1,max=rep*dim(pair)[2],style=3)
    for(j in 1:rep){
      x.i$pop <- factor(sample(x.i$pop,length(x.i$pop)))
      result.ij <- gstats2(x.i)
      dif.i[j+1,] <- result.ij$result[1,]-result.ij$result[2,]
      setTxtProgressBar(pb,rep*(i-1)+j)
    }
    dif.i[,3] <- -1*dif.i[,3]
    p.i <- apply(dif.i,2,function(z) length(z[z>=z[1]])/(rep+1))
    p <- rbind(p,p.i)
    print(c(t(pair)[i,],p.i))
  }
  p <- cbind(t(pair),p)
  colnames(p) <- c("pair1","pair2",colnames(y))
  end <- Sys.time()
  print(paste("Start:",start,"; End:",end,sep=" "))
  p
}

##-------------------------------------------------------
## permutation test for He
##
## x: genotype data for all samples (genind)
##-------------------------------------------------------
permtest_He <- function(x){
  start <- Sys.time()
  pair <- combn(levels(x$pop),2)
  sep.x <- seppop(x)
  p <- NULL
  test.result <- NULL;test.result <- as.list(test.result)
  for(i in 1:dim(pair)[2]){
    pb <- txtProgressBar(min=0,max=10,style=3)
    x.i1 <- sep.x[[c(1:length(sep.x))[names(sep.x)==pair[1,i]]]]
    x.i2 <- sep.x[[c(1:length(sep.x))[names(sep.x)==pair[2,i]]]]
    Hs.test.i <- Hs.test(x.i1,x.i2,alter="greater")
    p.i <- Hs.test.i$pvalue
    p <- c(p,p.i)
    test.result[[i]] <- Hs.test.i
    setTxtProgressBar(pb,i)
    print(c(t(pair)[i,],p.i))
  }
  names(test.result) <- paste(pair[1,],pair[2,],sep="_")
  end <- Sys.time()
  print(paste("Start:",start,"; End:",end,sep=" "))
  result <- list(pvalue = cbind(t(pair),p), overall.result = test.result)
```

```
  result
}

##-------------------------------------------------------
## permutation test for sMLH
##
## x: sMLH of all samples (vector)
## y: stages to which each sample belong (vector)
##-------------------------------------------------------
permtest_smlh <- function(x,y){
  start <- Sys.time()
  pair <- combn(levels(y),2)
  p <- NULL
  for(i in 1:dim(pair)[2]){
    x.i1 <- x[y==pair[1,i]]
    x.i2 <- x[y==pair[2,i]]
    x.i <- c(x.i1,x.i2)
    dif.i <- rep(0,1000)
    dif.i[1] <- mean(x.i1)-mean(x.i2)
    pb <- txtProgressBar(min=1,max=999,style=3)
    for(j in 1:999){
      x.ij <- sample(x.i,length(x.i))
      dif.i[j+1] <- mean(x.ij[1:length(x.i1)])-
mean(x.ij[length(x.i1)+1:length(x.i2)])
      setTxtProgressBar(pb,j)
    }
    p.i <- length(dif.i[dif.i>=dif.i[1]])/1000
    p <- c(p,p.i)
    print(i)
  }
  p <- as.data.frame(cbind(t(pair),p))
  p$p <- as.numeric(as.character(p$p))
  end <- Sys.time()
  print(paste("Start:",start,"; End:",end,sep=" "))
  p
}
```

Supporting Information 5 R code used for simulation analysis. We first defined function needed for simulating temporal genetic changes. We then defined function "`rep.geneticchange.model#`", which is used for simulation for scenario #.

```
# ----------------------------------------
#   0. functions
# ----------------------------------------

# convert "NaN" to 0

NaN0 <- function(y){
  y[is.na(y)] <- 0
  y
}


# rounding off

roundoff <- function(y){sapply(y, function(x) if(x%%1>=0.5) {x%/%1+1} else
  {x%/%1})}


# convert "list" of matrices to "matrix"

listofmat2mat <- function(y){
  n.elem <- dim(y[[1]])
  res <- matrix(unlist(y),n.elem[1]*n.elem[2])
  res
}


# calculate genetic diversity (expected heterozygosity)

He <- function(x,n.stage){
  freq0 <- lapply(x, function(y) (y[1:n.stage*3-2,]+y[1:n.stage*3-
1,]/2)/(y[1:n.stage*3-2,]+y[1:n.stage*3-1,]+y[1:n.stage*3,]))
  freq1 <- lapply(x, function(y) (y[1:n.stage*3,]+y[1:n.stage*3-
1,]/2)/(y[1:n.stage*3-2,]+y[1:n.stage*3-1,]+y[1:n.stage*3,]))
  ae <- mapply(function(x,y) 2*x*y,freq0,freq1)
  x4.mean <- matrix(apply(ae,1,mean,na.rm=TRUE),n.stage)
  x4.lower <- matrix(apply(ae,1,quantile,p=0.025,na.rm=TRUE),n.stage)
  x4.upper <- matrix(apply(ae,1,quantile,p=0.975,na.rm=TRUE),n.stage)
  res <- list(mean=x4.mean,CIlower=x4.lower,CIupper=x4.upper)
}

# ---------------------------------------------------
#   Senario 1 : Null Model
#
#   parameters
#   n.stage : number of stages
#   pop : number of flowerings [/25m^2]
#   mat : transition matrix
#   year : the length of simulation
#   reptime : number of repetitions
#   name : identical name used for output file
# ---------------------------------------------------

geneticchange.model1 <- function(n.stage, pop, mat, year){
  lambda <- Re(eigen(mat)$values[1]) # population growth rate
  stable <-
abs(Re(eigen(mat)$vectors[,Re(eigen(mat)$values)==max(Re(eigen(mat)$values))])
) # stable stage distribution
```

```
  n <- matrix(sapply(stable*pop/stable[n.stage], function(y)
c(y/4,y/2,y/4)),ncol=1) # initial population
  t0 <- lapply(mat, function(x) diag(x,3))
  t <- matrix(sapply(c(0:(n.stage-1)), function(x)
rbind(t0[[n.stage*x+1]],t0[[n.stage*x+2]],t0[[n.stage*x+3]],

t0[[n.stage*x+4]],t0[[n.stage*x+5]])),n.stage*3,n.stage*3) # t : transition
matrix
  f <- mat[1,n.stage]   # fecundity
    for(i in 1:year){
      pA <- (n[(3*(n.stage-1)+1),i]+n[(3*(n.stage-
1)+2),i]/2)/sum(n[(3*(n.stage-1)+1:3),i]) # frequency of A
      t.i <- t
      t.i[1:3,(3*(n.stage-1)+1):(3*n.stage)] <- matrix(c(pA*f,(1-pA)*f,0,
                                                 pA*f/2,f/2,(1-pA)*f/2,
                                                 0,pA*f,(1-pA)*f),3,3) #
transition matrix from year i to i+1
      n.i <- t.i %*% n[,i]
      n <- cbind(n,n.i/lambda) # scaling and combining population size data
    }
  colnames(n) <- c(0:year)
  n
}

rep.geneticchange.model1 <- function(n.stage, pop, mat, year,reptime,name){
  N.allrep <- NULL; N.allrep <- as.list(N.allrep)
  for(j in 1:reptime){
    N.allrep[[j]] <- geneticchange.model1(n.stage,pop,mat,year)
  }

write.csv(He(N.allrep,n.stage),paste("model1_He_",name,".csv",sep=""),row.name
s=F)

write.csv(listofmat2mat(N.allrep),paste("model1_",name,".csv",sep=""),row.name
s=F)
}

# ----------------------------------------------------------------
#   Senario 2 : Genetic Drift
#
#   parameters
#   n.stage : number of stages
#   Nb : effective number of breeders
#   pop : number of flowerings [/25m^2]
#   mat : transition matrix
#   year : the length of simulation
#   reptime : number of repetitions
#   name : identical name used for output file
# ----------------------------------------------------------------

geneticchange.model2 <- function(n.stage, Nb, pop, mat, year){
  lambda <- Re(eigen(mat)$values[1]) # population growth rate
  stable <-
abs(Re(eigen(mat)$vectors[,Re(eigen(mat)$values)==max(Re(eigen(mat)$values))])
) # stable stage distribution
  n <- matrix(sapply(stable*pop/stable[n.stage], function(y)
c(y/4,y/2,y/4)),ncol=1) # initial population
  t0 <- lapply(mat, function(x) diag(x,3))
  t <- matrix(sapply(c(0:(n.stage-1)), function(x)
rbind(t0[[n.stage*x+1]],t0[[n.stage*x+2]],t0[[n.stage*x+3]],

t0[[n.stage*x+4]],t0[[n.stage*x+5]])),n.stage*3,n.stage*3) # t : transition
matrix
  f <- mat[1,n.stage]   # fecundity
  for(i in 1:year){
```

```
    pA <- (n[(3*(n.stage-1)+1),i]+n[(3*(n.stage-1)+2),i]/2)/sum(n[(3*(n.stage-
1)+1:3),i])
    n.i.trans <- t[-c(1:3),] %*% n[,i]
    n.i.offsp <- rmultinom(1,Nb,c(pA^2,2*pA*(1-pA),(1-
pA)^2))/Nb*sum(n[(3*(n.stage-1)+1:3),i])*f
    n.i <- c(n.i.offsp,n.i.trans)
    n <- cbind(n,n.i/lambda)
  }
  colnames(n) <- c(0:year)
  n
}

rep.geneticchange.model2 <- function(n.stage, Nb, pop, mat,
year,reptime,name){
  N.allrep <- NULL; N.allrep <- as.list(N.allrep)
  for(j in 1:reptime){
    N.allrep[[j]] <- geneticchange.model2(n.stage,Nb,pop,mat,year)
  }

write.csv(He(N.allrep,n.stage),paste("model2_He_",name,".csv",sep=""),row.name
s=F)

write.csv(listofmat2mat(N.allrep),paste("model2_",name,".csv",sep=""),row.name
s=F)
}

# -------------------------------------------------------
#   Senario 3 : Demographic Stochasticity
#
#   parameters
#   n.stage : number of stages
#   pop : number of flowerings [/25m^2]
#   mat : transition matrix
#   year : the length of simulation
#   reptime : number of repetitions
#   name : identical name used for output file
# -------------------------------------------------------

geneticchange.model3 <- function(n.stage, pop, mat, year){
  lambda <- Re(eigen(mat)$values[1]) # population growth rate
  stable <-
abs(Re(eigen(mat)$vectors[,Re(eigen(mat)$values)==max(Re(eigen(mat)$values))])
) # stable stage distribution
  n <- matrix(roundoff(sapply(stable*pop/stable[n.stage], function(y)
c(y/4,y/2,y/4))),ncol=1) # initial population
  t0 <- lapply(mat, function(x) diag(x,3))
  t <- matrix(sapply(c(0:(n.stage-1)), function(x)
rbind(t0[[n.stage*x+1]],t0[[n.stage*x+2]],t0[[n.stage*x+3]],

t0[[n.stage*x+4]],t0[[n.stage*x+5]])),n.stage*3,n.stage*3) # t : transition
matrix
  f <- mat[1,n.stage]   # fecundity
  t.death <- rbind(t[-c(1:3),],1-apply(t[-c(1:3),],2,sum))
  matrix.year <- NULL
  for(i in 1:year){
    transition.i <- sapply(c(1:(n.stage*3)), function(x)
rmultinom(1,n[x,i],t.death[,x])[1:(3*(n.stage-1))])
    transition.i.sum <- apply(transition.i,1,sum)
    trans.prop.i <- sapply(c(1:(n.stage*3)), function(x)
if(n[x,i]>0){transition.i[,x]/n[x,i]}else{rep(0,3*(n.stage-1))})
    pA <- NaN0((n[(3*(n.stage-1)+1),i]+n[(3*(n.stage-
1)+2),i]/2)/sum(n[(3*(n.stage-1)+1:3),i]))
    t.f.i <-  c(pA*f,(1-pA)*f,0,pA*f/2,f/2,(1-pA)*f/2,0,pA*f,(1-pA)*f)
    offs.i <- matrix(mapply(rpois,1,t.f.i*rep(n[(n.stage*3-2:0),i],each=3)),3)
    offs.mat.i <- matrix(NaN0(offs.i/rep(n[(n.stage*3-2:0),i],each=3)),3,3)
```

```
    n.i <- matrix(c(tapply(offs.i, rep(1:3,3), sum),
transition.i.sum),n.stage*3,1)
    n <-
cbind(n,roundoff(rep(tapply(n[,1],rep(1:n.stage,each=3),sum),each=3)*NaN0(n.i/
rep(tapply(n.i,rep(1:n.stage,each=3),sum),each=3))))
    t.i <- rbind(cbind(matrix(0,3,3*(n.stage-1)),offs.mat.i),trans.prop.i)
    matrix.year <- cbind(matrix.year,as.vector(t.i))
  }
  colnames(n) <- c(0:year)
  allresult <- list(N = n, matrix = matrix.year)
  allresult
}

rep.geneticchange.model3 <- function(n.stage, pop, mat, year,reptime,name){
  N.allrep <- NULL; N.allrep <- as.list(N.allrep)
  matrix.allrep <- NULL; matrix.allrep <- as.list(matrix.allrep)
  for(j in 1:reptime){
    res.i <- geneticchange.model3(n.stage,pop,mat,year)
    N.allrep[[j]] <- res.i$N
    matrix.allrep[[j]] <- res.i$matrix
  }

write.csv(He(N.allrep,n.stage),paste("model3_He_",name,".csv",sep=""),row.name
s=F)

write.csv(listofmat2mat(N.allrep),paste("model3_",name,".csv",sep=""),row.name
s=F)

write.csv(listofmat2mat(matrix.allrep),paste("model3_matrix_",name,".csv",sep=
""),row.names=F)
}

# -------------------------------------------------------
#   Senario 4 : Genetic Drift & Demographic Stochasticity
#
#   parameters
#   n.stage : number of stages
#   Nb : effective number of breeders
#   pop : number of flowerings [/25m^2]
#   mat : transition matrix
#   year : the length of simulation
#   reptime : number of repetitions
#   name : identical name used for output file
# -------------------------------------------------------

geneticchange.model4 <- function(n.stage, Nb, pop, mat, year){
  lambda <- Re(eigen(mat)$values[1]) # population growth rate
  stable <-
abs(Re(eigen(mat)$vectors[,Re(eigen(mat)$values)==max(Re(eigen(mat)$values))])
) # stable stage distribution
  n <- matrix(roundoff(sapply(stable*pop/stable[n.stage], function(y)
c(y/4,y/2,y/4))),ncol=1) # initial population
  t0 <- lapply(mat, function(x) diag(x,3))
  t <- matrix(sapply(c(0:(n.stage-1)), function(x)
rbind(t0[[n.stage*x+1]],t0[[n.stage*x+2]],t0[[n.stage*x+3]],

t0[[n.stage*x+4]],t0[[n.stage*x+5]])),n.stage*3,n.stage*3) # t : transition
matrix
  f <- mat[1,n.stage]   # fecundity
  t.death <- rbind(t[-c(1:3),],1-apply(t[-c(1:3),],2,sum))
  matrix.year <- NULL
  for(i in 1:year){
    transition.i <- sapply(c(1:(n.stage*3)), function(x)
rmultinom(1,n[x,i],t.death[,x])[1:(3*(n.stage-1))])
    transition.i.sum <- apply(transition.i,1,sum)
```

```
    trans.prop.i <- sapply(c(1:(n.stage*3)), function(x)
if(n[x,i]>0){transition.i[,x]/n[x,i]}else{rep(0,3*(n.stage-1))})
    pA <- NaN0((n[(3*(n.stage-1)+1),i]+n[(3*(n.stage-
1)+2),i]/2)/sum(n[(3*(n.stage-1)+1:3),i]))
    offsp.i.0 <- rmultinom(1,Nb,c(pA^2,2*pA*(1-pA),(1-
pA)^2))/Nb*sum(n[(3*(n.stage-1)+1:3),i])*f
    offs.i <- mapply(rpois,1,offsp.i.0)
    n.i <- matrix(c(offs.i, transition.i.sum),n.stage*3,1)
    n <-
cbind(n,roundoff(rep(tapply(n[,1],rep(1:n.stage,each=3),sum),each=3)*NaN0(n.i/
rep(tapply(n.i,rep(1:n.stage,each=3),sum),each=3))))
    matrix.year <- cbind(matrix.year,as.vector(trans.prop.i))
  }
  colnames(n) <- c(0:year)
  allresult <- list(N = n, matrix = matrix.year)
  allresult
}

rep.geneticchange.model4 <- function(n.stage, Nb, pop, mat,
year,reptime,name){
  N.allrep <- NULL; N.allrep <- as.list(N.allrep)
  matrix.allrep <- NULL; matrix.allrep <- as.list(matrix.allrep)
  for(j in 1:reptime){
    res.i <- geneticchange.model4(n.stage,Nb,pop,mat,year)
    N.allrep[[j]] <- res.i$N
    matrix.allrep[[j]] <- res.i$matrix
  }

write.csv(He(N.allrep,n.stage),paste("model4_He_",name,".csv",sep=""),row.name
s=F)

write.csv(listofmat2mat(N.allrep),paste("model4_",name,".csv",sep=""),row.name
s=F)

write.csv(listofmat2mat(matrix.allrep),paste("model4_matrix_",name,".csv",sep=
""),row.names=F)
}
```

Supporting Information 6 The procedures to construct random transition matrices

(1) Perturbing transition probabilities

Step 1: From the original transition matrix (Equation (7) in the main text), we extracted none-zero elements in column i except fecundity, which are transition probabilities of an individual in stage i.

Step 2: In the case of 3L and FL stage, in which more than two elements were extracted, we sequentially added the extracted values in an increasing order to calculate their cumulative sum.

Step 3: We supposed a line with length 1which has breaking points at coordinates that are equal to the extracted value (in the case of SE, SD, and 1L) or the cumulative sum (in the case of 3L and FL). The length of each broken piece corresponds to one of the transition paths (i.e., growth, stasis, and retrogression) or mortality.

Step 4: We drew a random number from the following uniform distribution to perturb the coordinate of each breaking point.

$$U\,(max(0, b - 0.1), min(1, b + 0.1))$$

b is the original coordinate of any breaking points. This enabled us to randomly shift b by up to 0.1 within the range from 0 to 1.

Step 5: The resultant lengths of broken pieces were used as randomized transition probabilities of an individual in stage i.

These procedures are graphically shown in Figure S6, taking the case of 3L stage as an example.

(2) Perturbing fecundity

Fecundity was perturbed by drawing random numbers from the following distribution.

$$U\,(6.197/g_0 - 0.1, 6.197/g_0 + 0.1)$$

We did (1) and (2) 500 times to construct 500 random transition matrices.

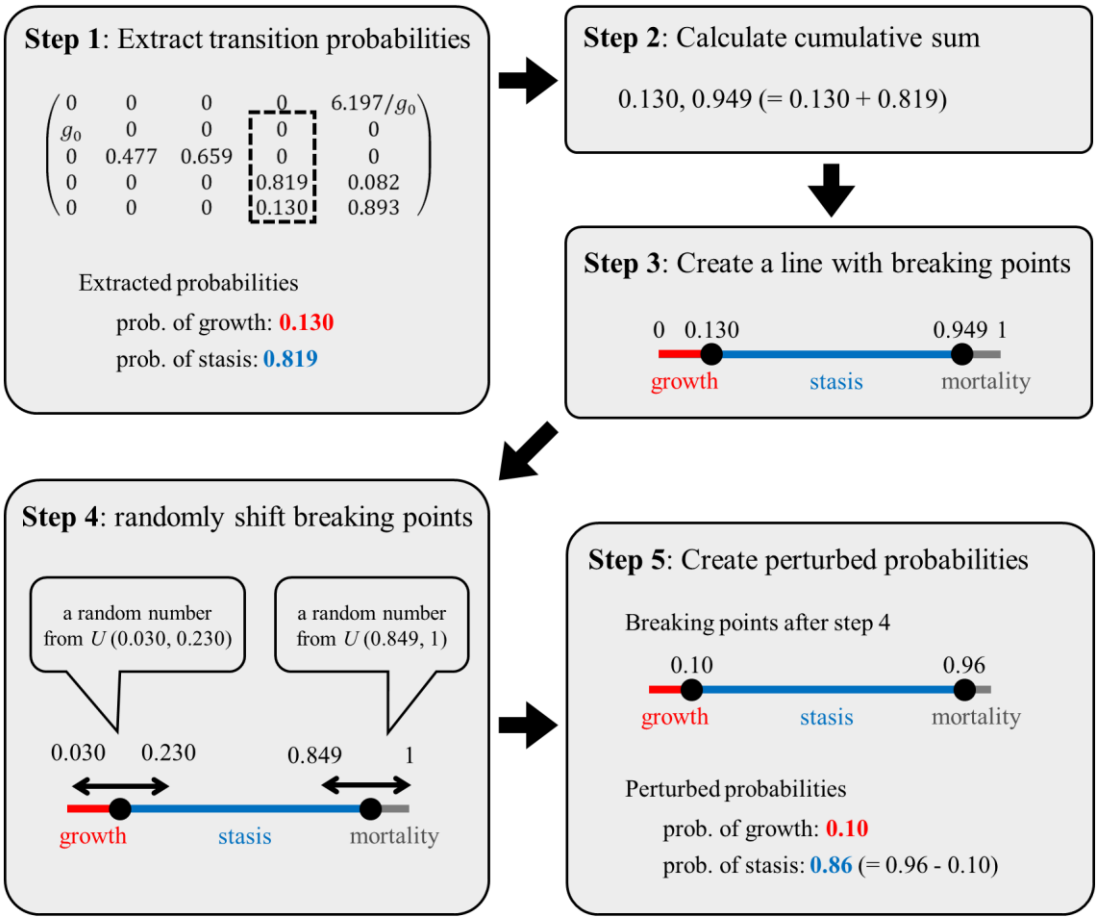

Figure S2 Procedures to randomize transition probabilities of 3L stage. In step 2, transition probabilities were sequentially added in an increasing order so that the range of the two uniform distributions would not overlap in step 4

Supporting Information 7 Temporal dynamics of expected heterozygosity in SAK (Sakuragi) and FUJ (Fuji) populations under different germination rate (g_0)

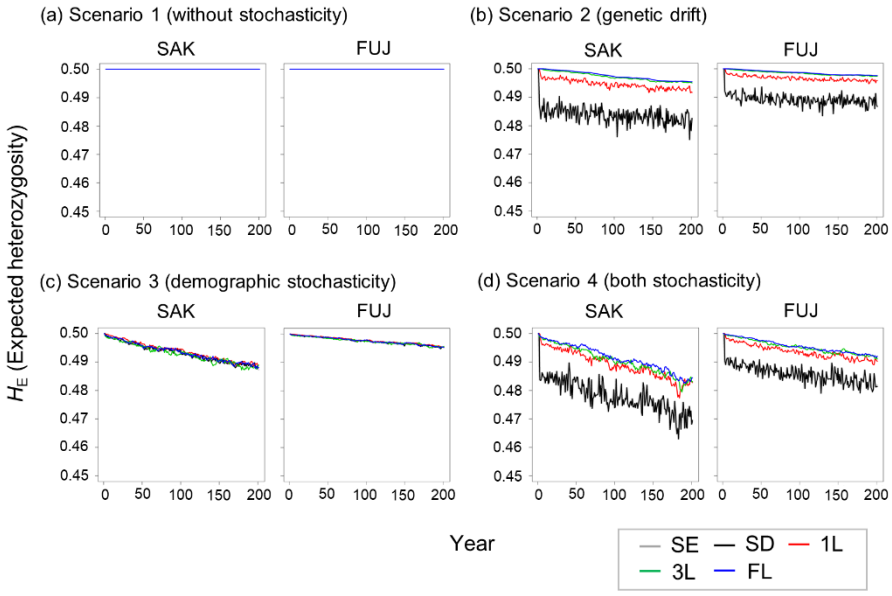

Figure S3-1 $g_0 = 1$

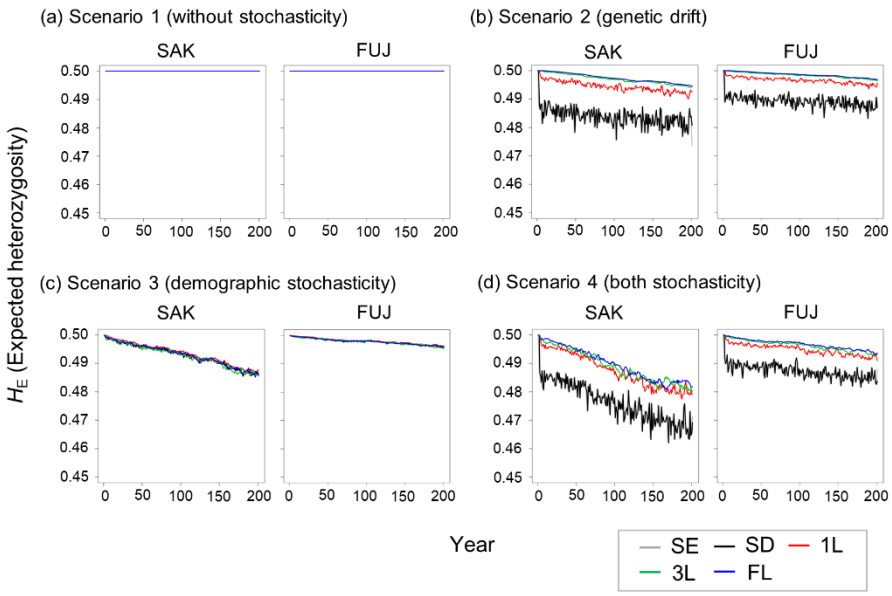

Figure S3-2 $g_0 = 0.8$

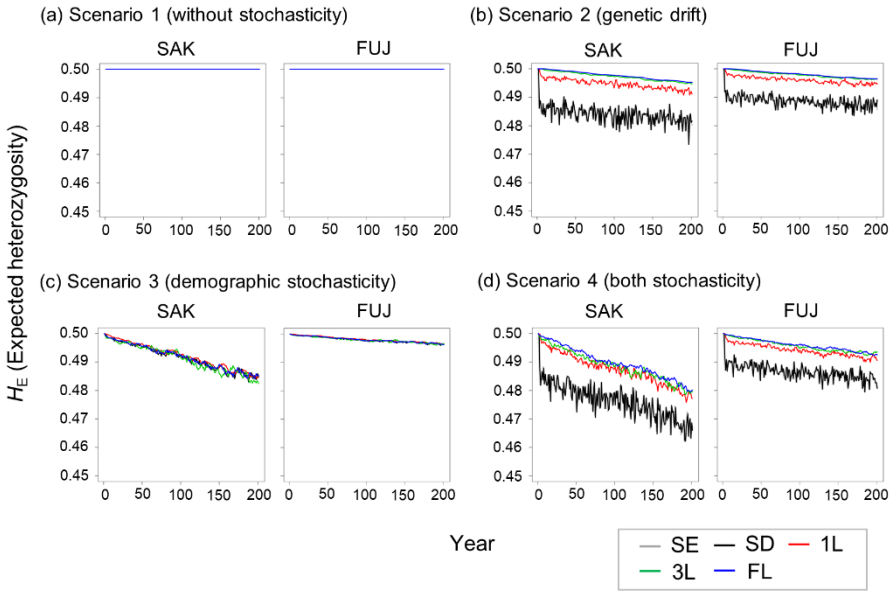

Figure S3-3 $g_0 = 0.6$

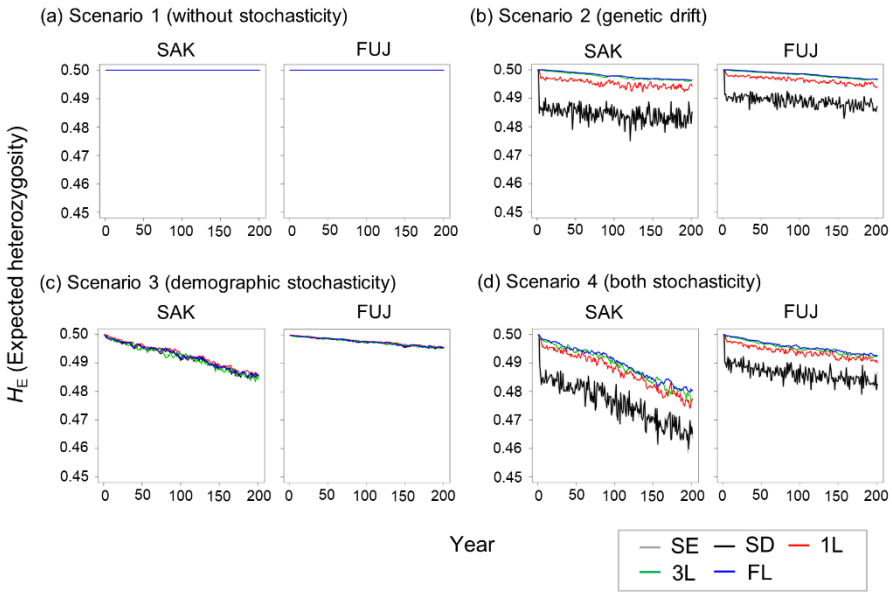

Figure S3-4 $g_0 = 0.4$

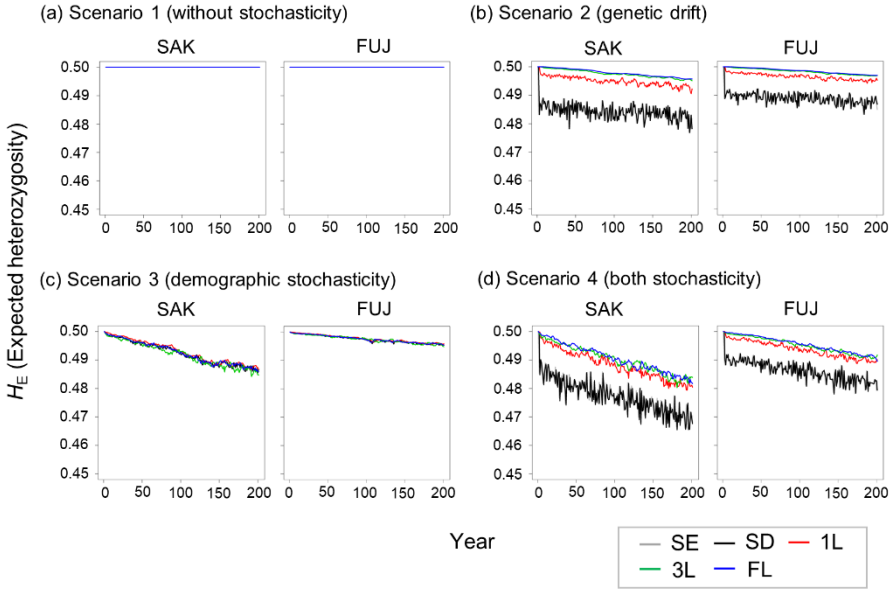

Figure S3-5 $g_0 = 0.2$

Supporting Information 8 RDA triplots of expected heterozygosity and demographic rates in SAK (Sakuragi) and FUJ (Fuji) populations under different germination rate (g_0)

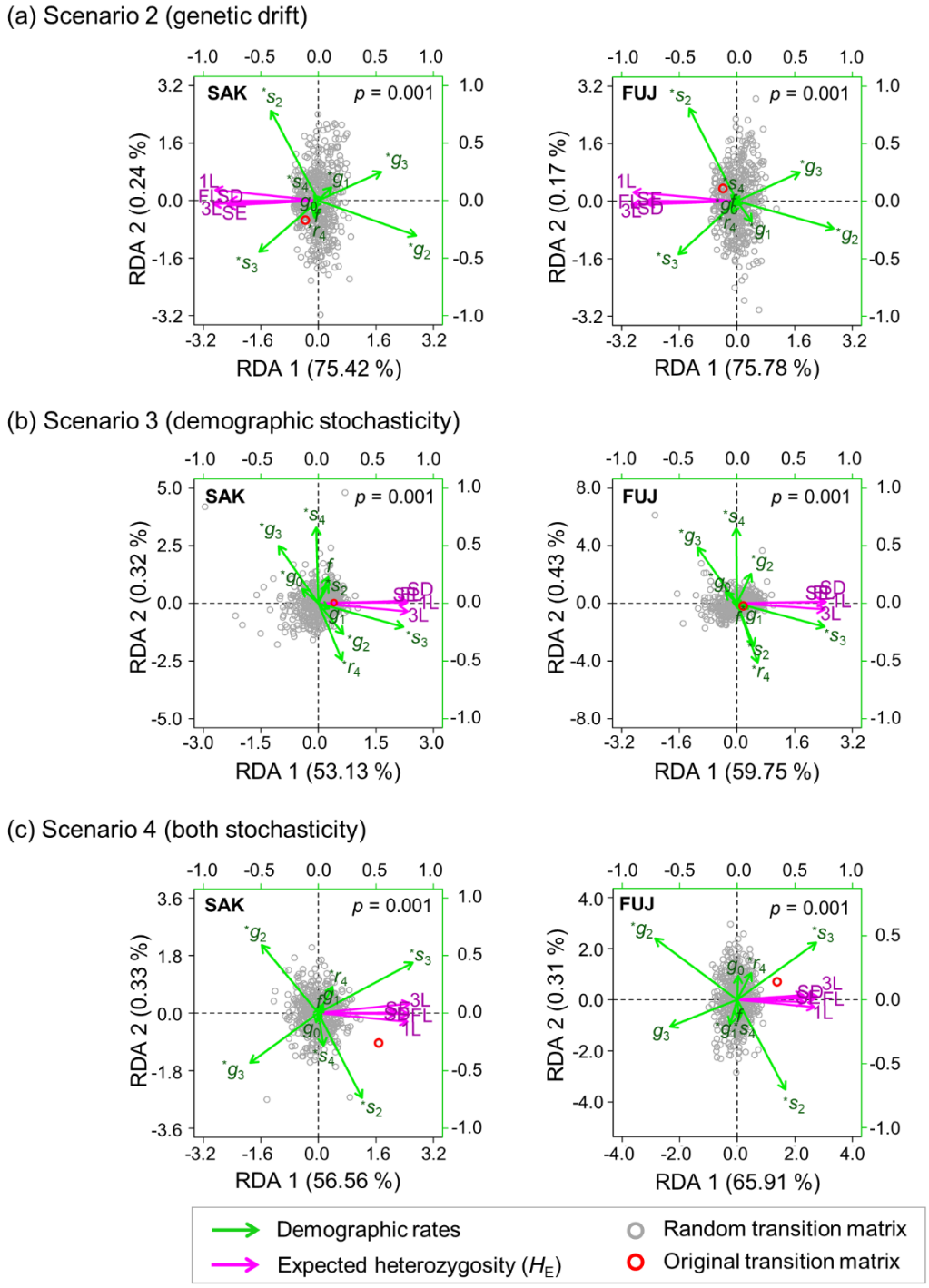

Figure S4-1 $g_0 = 1$

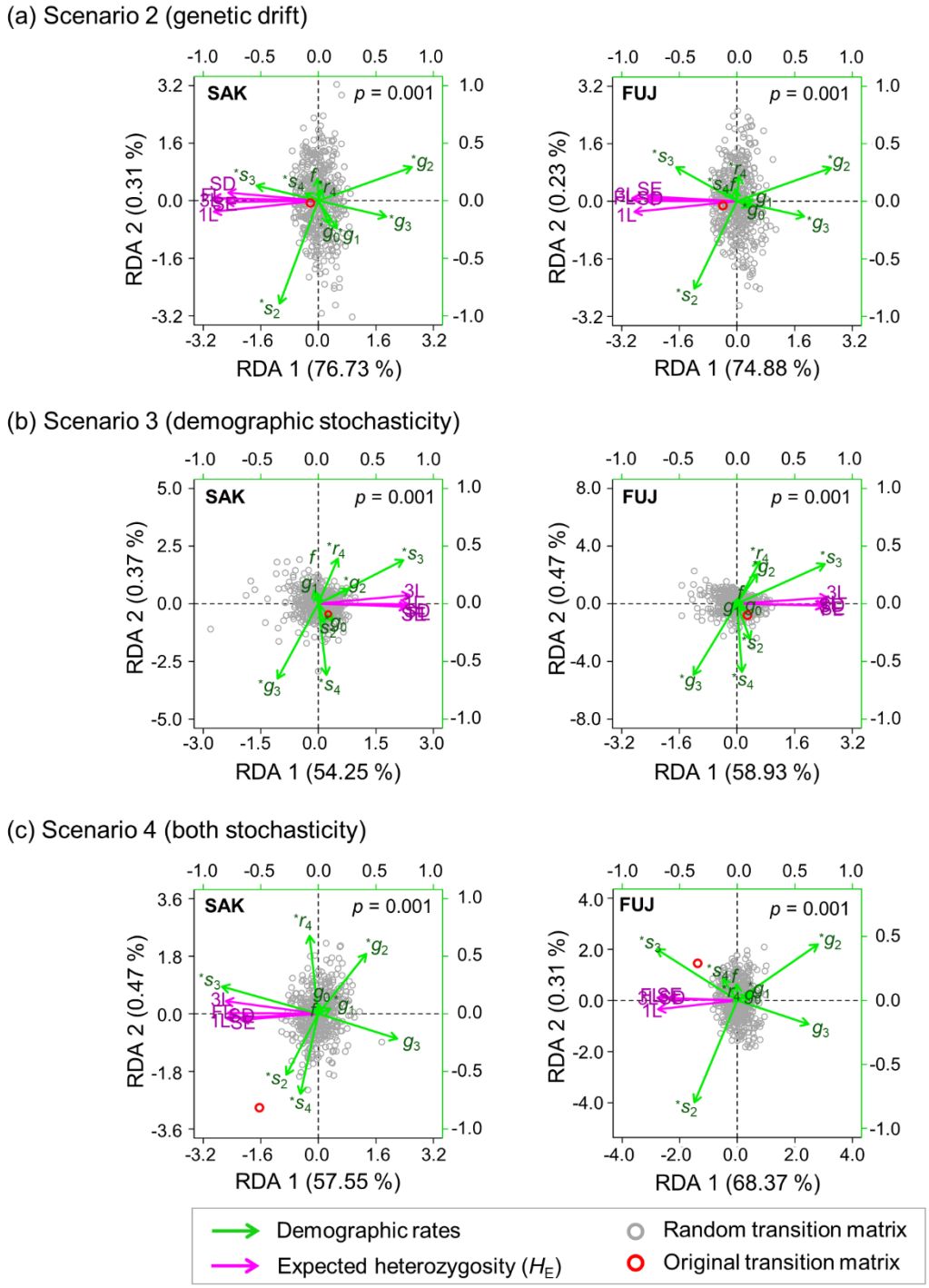

Figure S4-2 $g_0 = 0.8$

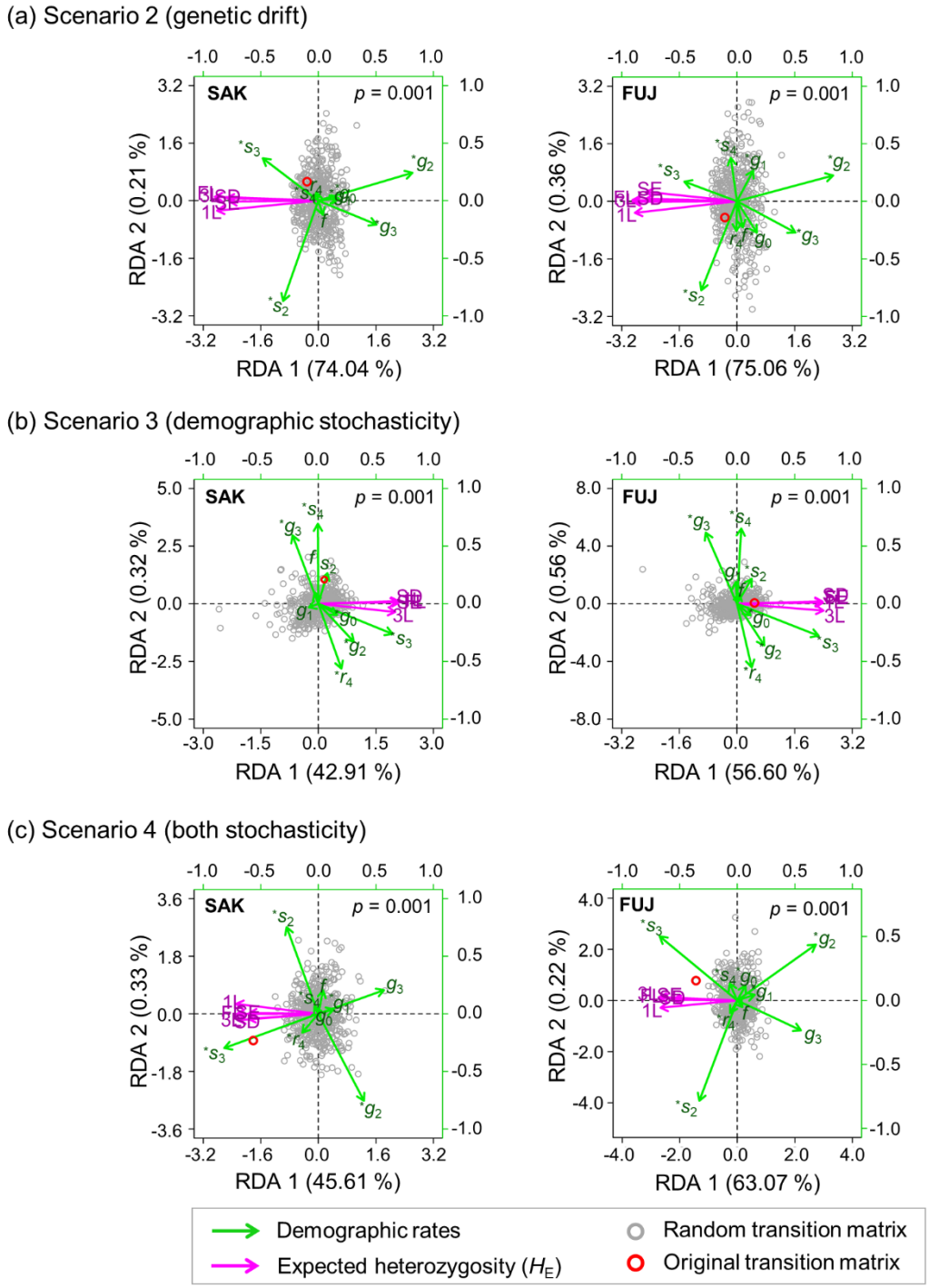

Figure S4-3 $g_0 = 0.6$

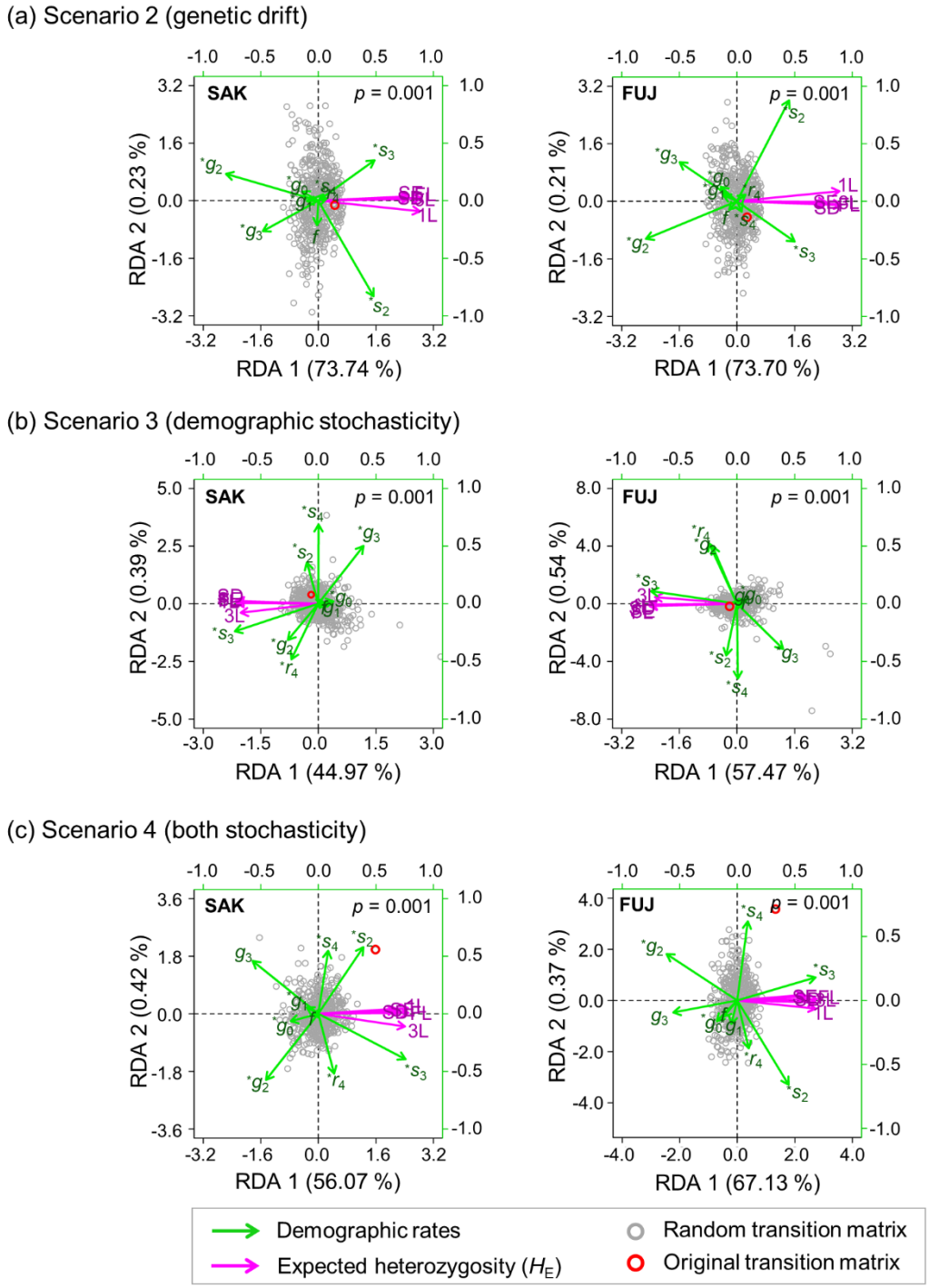

Figure S4-4 $g_0 = 0.4$

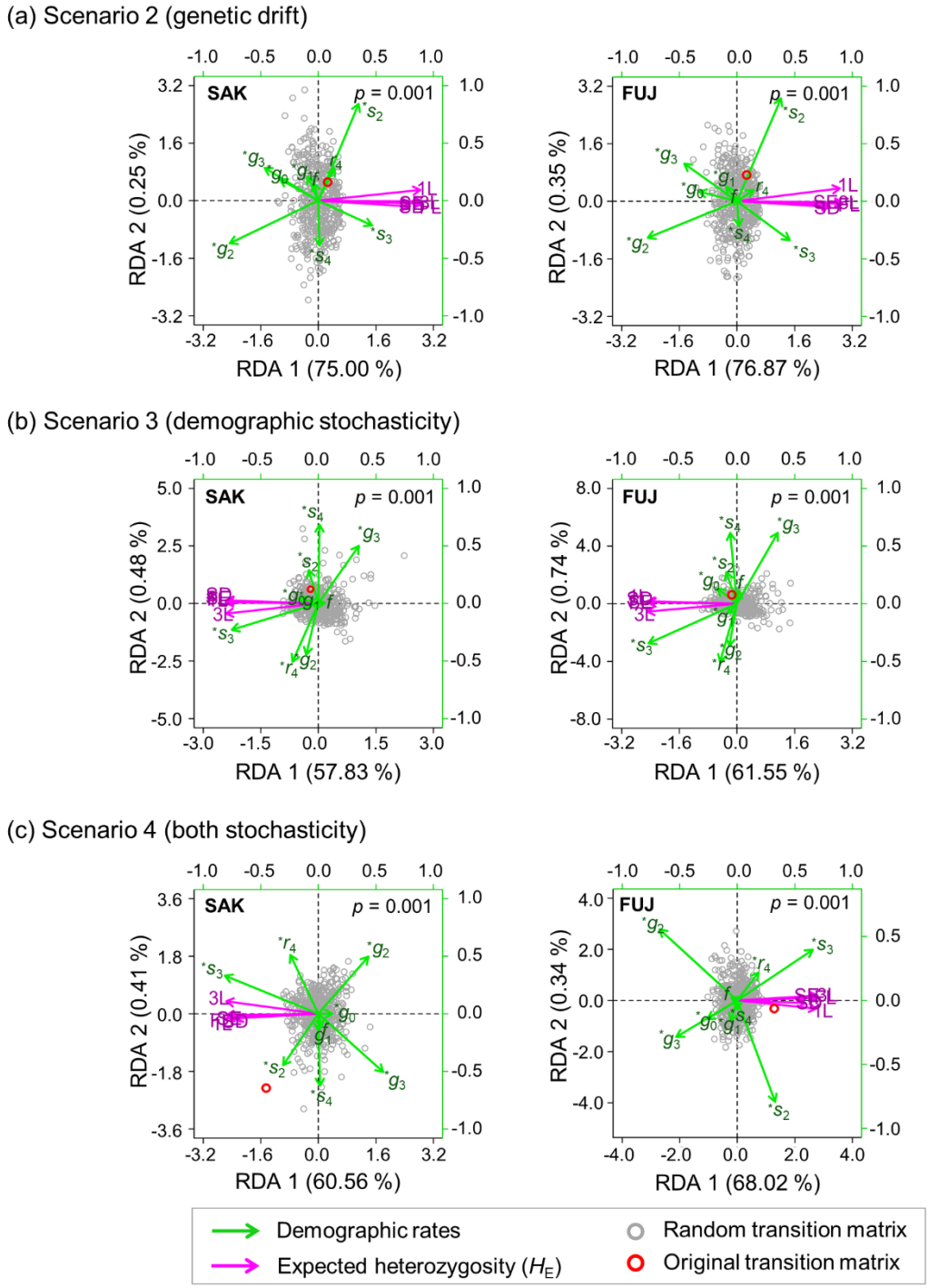

Figure S4-5 $g_0 = 0.2$