# Install required packages and source("sr.value.R").

source("packages.R")

# Create a grid of local communities with coordinate (*x*, *y*) andenvironmental gradient *E* (matrix **landscape)**.Construct the first and second-order orthogonal environmental variables for the model communities in a grid of 10 x 10 local communities in the center of system. For large system, the environmental gradient of small system was quadrupled.

source("landscape.hump16.R")

source("landscape.rand.R")

source("landscape.wave.R")

source("landscape.hump64.R")

source("landscape.rand.large.R")

source("landscape.wave2.R")

# Function to creating the initial species pool of system; individuals code, species code, guilds code, and their environmental niche (i.e. environmental optimum and tolerance) (matrix **pool.t0)**.

source("fn.pool.t0.R")

# To ensuring the replicability of initial conditions, set the seeds of random number generator in systematic manner.

# In small system; for *g* = 1 and n.samples.g.sizes = 1, the seeds from one to 25 were used; for *g* >= 8 and n.samples.g.sizes = 5, the seeds from one to five were used. In large system, the initial conditions of small system generated by the seeds from one to five and the first population size of guilds were quadrupled. The results of each scenario was saved in different working directory.

# In small system; for *g* = 1, and the tenth environmental niche.

set.seed(10)

pool.t0 <- fn.pool.t0(JM=6400, g=1, diversity=0, n.samples.g.sizes=1)

write.csv(pool.t0, file = "pool.t0.csv")

# In small system; for *g* = 8, the second environmental niche of guilds, and the second population size of guilds.

set.seed(2)

pool.t0 <- fn.pool.t0(JM=6400, g=8, diversity=0, n.samples.g.sizes=5)

write.csv(pool.t0[[2]], file = "pool.t0.csv")

# In large system; for *g* = 1, and the first environmental niche.

set.seed(1)

pool.t0 <- fn.pool.t0(JM=6400, g=1, diversity=0, n.samples.g.sizes=1)

pool.t0.large <- pool.t0[rep(seq\_len(nrow(pool.t0)), 4), ]

row.names(pool.t0.large) <- 1:25600

write.csv(pool.t0.large, file = "pool.t0.large.csv")

# In large system; for *g* = 8, the second environmental niche of guilds, and the first population size of guilds.

set.seed(2)

pool.t0 <- fn.pool.t0(JM=6400, g=8, diversity=0, n.samples.g.sizes=5)

pool.t0 <- pool.t0[[1]]

pool.t0.large <- pool.t0[rep(seq\_len(nrow(pool.t0)), 4), ]

row.names(pool.t0.large) <- 1:25600

write.csv(pool.t0.large, file = "pool.t0.large.csv")

# Function to calculating the species compositions of system forward in time.

source("fn.forward.R")

# Check the convergence of system from alternative diversities; monodominance and infinite, for one scenario of *g* = 40 in the small system with dispersal rate *m* = 0.09 and 16 humps environmental gradient.

n.sites = 400

JM = 6400

J = 16

nu = 0.001

nb <- cell2nb(nrow=20, ncol=20, type="queen", torus=TRUE)

nb.mat <- nb2mat(nb, style="W")

landscape.hump16 <- read.csv("landscape.hump16.csv", header=TRUE, row.names=1)

set.seed(1) # Use the first environmental niche of guilds and monodominance.

pool.t0 <- fn.pool.t0(JM, g=40, diversity=0, n.samples.g.sizes=5)

pool.t0 <- pool.t0[[1]] # Use the first population size of guilds.

View(pool.t0)

set.seed(1) # Set the same seeds in random number generator, and assign the individuals of system to initial locations at random.

pool <- data.frame(ind = 1:JM, locations = complete\_ra(N = JM, num\_arms = n.sites))

LC.t0 <- dcast(pool, locations~ind, length) [,-1]

Conditions.t0 <- list(n.new.sp = 0, n.extinction = 0, species.richness = length(unique(pool.t0$species)), n.ancestors = nrow(pool.t0), n.groups = length(unique(pool.t0$groups)), LC.t0=LC.t0, pool.t0=pool.t0)

sim.result <- fn.forward(Conditions.t0=Conditions.t0, E = landscape.hump16$E, nu=nu, m=0.09, nb.mat=nb.mat, n.timestep=10000, keep = FALSE, stop = TRUE)

n.groups.mono <- sim.result$n.groups

species.richness.mono <- sim.result$species.richness

pool.out.mono <- sim.result$Conditions$pool.t0

LC.out.mono <- sim.result$Conditions$LC.t0

View(n.groups.mono)

View(species.richness.mono)

View(pool.out.mono)

set.seed(1) # Use the first environmental niche of guilds and infinite diversities.

pool.t0 <- fn.pool.t0(JM, g=40, diversity=1, n.samples.g.sizes=5)

pool.t0 <- pool.t0[[1]] # Use the first population size of guilds.

View(pool.t0)

set.seed(1) # Set the same seeds in random number generator.

pool <- data.frame(ind = 1:JM, locations = complete\_ra(N = JM, num\_arms = n.sites))

LC.t0 <- dcast(pool, locations~ind, length) [,-1]

Conditions.t0 <- list(n.new.sp = 0, n.extinction = 0, species.richness = length(unique(pool.t0$species)), n.ancestors = nrow(pool.t0), n.groups = length(unique(pool.t0$groups)), LC.t0=LC.t0, pool.t0=pool.t0)

sim.result.inf <- fn.forward(Conditions.t0=Conditions.t0, E = landscape.hump16$E, nu=nu, m=0.09, nb.mat=nb.mat, n.timestep=3687, keep = FALSE, stop = FALSE)

n.groups.inf <- sim.result$n.groups

species.richness.inf <- sim.result$species.richness

pool.out.inf <- sim.result$Conditions$pool.t0

LC.out.inf <- sim.result$Conditions$LC.t0

View(n.groups.inf)

View(species.richness.inf)

View(pool.out.inf)

identical(LC.out.mono, LC.out.inf)

# Function to calculating the diversity statics of initial conditions and emergent biodiversity patterns.

source("fn.divpart.in.R")

source("fn.divpart.out.R")

# Construct the principal coordinates of neighbour matrices (PCNM) of overall model for the model communities in the center of small and large system.

source("PCNM.pos.center.R")

source("PCNM.pos.large.center.R")

# Function to calculating the summary statics of variation partitioning for initial conditions and emergent biodiversity patterns.

source("fn.varpart.in.R")

source("fn.varpart.out.R")

# Implement individual-based model, and calculate the summary statics of model communities. The results of each scenario was saved in different working directory.

# To ensuring the replicability of individual-based model, set the seeds of random number generator in systematic manner. The replicability of fn.varpart.out was not ensured.

# In small system; for *g* = 1, set the same seeds as initial species pool of system; for *g* >= 8, set the seeds from one to five for the initial conditions generated by seed one, set the seeds from six to ten for the initial conditions generated by seed two; then, repeat subsequently for three other numbers of guilds until 25. In large system; for *g* = 1, set the seeds from one to five for the initial conditions generated by the seeds from one to five; for *g* >= 8, set the seeds from one to five using only the first population size of guilds in each environmental niche of guilds.

# In the small system with dispersal rate *m* = 0.09 and 16 humps environmental gradient. For *g* = 40, using the third environmental niche of guilds, and the third population size of guilds.

n.sites = 400

JM = 6400

J = 16

nu = 0.001

nb <- cell2nb(nrow=20, ncol=20, type="queen", torus=TRUE)

nb.mat <- nb2mat(nb, style="W")

landscape <- read.csv("landscape.hump16.csv", header=TRUE, row.names=1)

landscape.center <- landscape[landscape$x >= 6 & landscape$x <= 15 & landscape$y >= 6 & landscape$y <= 15, ]

PCNM.pos.center <- read.csv("PCNM.pos.center.csv", header=TRUE, row.names=1)

E.trans.center <- read.csv("E.trans.hump16.center.csv", header=TRUE, row.names=1)

set.seed(13) # The seeds from one to 25 were used. The seed was 13 for this scenario.

pool.t0 <- read.csv("pool.t0.csv", header=TRUE, row.names=1)

pool <- data.frame(ind = 1:JM, locations = complete\_ra(N = JM, num\_arms = n.sites))

LC.t0 <- dcast(pool, locations~ind, length) [,-1]

write.csv(pool, file = "pool.csv")

write.csv(LC.t0, file = "LC.t0.csv")

LC.t0.center <- LC.t0[which(landscape$x >= 6 & landscape$x <= 15 & landscape$y >= 6 & landscape$y <= 15), ]

LC.t0.center.no0 <- LC.t0.center[ ,which(!colSums(LC.t0.center)==0)]

pool.t0.center <- pool.t0[which(!colSums(LC.t0.center)==0), ]

pool.center <- pool[which(!colSums(LC.t0.center)==0), ]

divpart.in <- fn.divpart.in(pool.t0=pool.t0.center, pool=pool.center)

write.csv(divpart.in, file = "divpart.in.csv")

varpart.in <- fn.varpart.in(PCNM.pos=PCNM.pos.center, E.trans=E.trans.center, pool=pool.center, pool.t0=pool.t0.center)

write.csv(varpart.in, file = "varpart.in.csv")

Conditions.t0 <- list(n.new.sp = 0, n.extinction = 0, species.richness = length(unique(pool.t0$species)), n.ancestors = nrow(pool.t0), n.groups = length(unique(pool.t0$groups)), LC.t0=LC.t0, pool.t0=pool.t0)

system.time(sim.result <- fn.forward(Conditions.t0=Conditions.t0, E = landscape$E, nu=nu, m=0.09, nb.mat=nb.mat, n.timestep=10000, keep = FALSE, stop = TRUE))

write.csv(sim.result$Conditions$pool.t0, file = "pool.out.csv")

write.csv(sim.result$Conditions$LC.t0, file = "LC.out.csv")

write.csv(sim.result$species.richness, file = "n.species.csv")

write.csv(sim.result$n.groups, file = "n.groups.csv")

set.seed(13) # Although the replicability of fn.varpart.out was not ensured, the same seed was used as above.

pool.out <- read.csv("pool.out.csv", header=TRUE, row.names=1)

LC.out <- read.csv("LC.out.csv", header=TRUE, row.names=1)

LC.out.center <- LC.out[which(landscape$x >= 6 & landscape$x <= 15 & landscape$y >= 6 & landscape$y <= 15), ]

LC.out.center.no0 <- LC.out.center[ ,which(!colSums(LC.out.center)==0)]

pool.out.center <- pool.out[which(!colSums(LC.out.center)==0), ]

divpart.out <- fn.divpart.out(pool.t0=pool.out.center, LC.t0=LC.out.center.no0)

write.csv(divpart.out, file = "divpart.out.csv")

system.time(varpart.out <- fn.varpart.out(PCNM.pos=PCNM.pos.center, landscape=landscape.center, E.trans=E.trans.center, pool.t0=pool.out.center, LC.t0=LC.out.center.no0))

write.csv(varpart.out, file = "varpart.out.csv")

# In the large system with dispersal rate *m* = 0.81 and 64 humps environmental gradient. For *g* = 40, using the the first population size of guilds in the third environmental niche of guilds.

n.sites = 1600

JM = 25600

J = 16

nu = 0.001

nb <- cell2nb(nrow=40, ncol=40, type="queen", torus=TRUE)

nb.mat <- nb2mat(nb, style="W")

landscape <- read.csv("landscape.hump64.csv", header=TRUE, row.names=1)

landscape.center <- landscape[landscape$x >= 16 & landscape$x <= 25 & landscape$y >= 16 & landscape$y <= 25, ]

PCNM.pos.center <- read.csv("PCNM.pos.large.center.csv", header=TRUE, row.names=1)

E.trans.center <- read.csv("E.trans.hump64.center.csv", header=TRUE, row.names=1)

set.seed(3) # The seeds from one to five were used. The seed was three for this scenario.

pool.t0 <- read.csv("pool.t0.large.csv", header=TRUE, row.names=1)

pool <- data.frame(ind = 1:JM, locations = complete\_ra(N = JM, num\_arms = n.sites))

LC.t0 <- dcast(pool, locations~ind, length) [,-1]

write.csv(pool, file = "pool.csv")

write.csv(LC.t0, file = "LC.t0.csv")

LC.t0.center <- LC.t0[which(landscape$x >= 16 & landscape$x <= 25 & landscape$y >= 16 & landscape$y <= 25), ]

LC.t0.center.no0 <- LC.t0.center[ ,which(!colSums(LC.t0.center)==0)]

pool.t0.center <- pool.t0[which(!colSums(LC.t0.center)==0), ]

pool.center <- pool[which(!colSums(LC.t0.center)==0), ]

divpart.in <- fn.divpart.in(pool.t0=pool.t0.center, pool=pool.center)

write.csv(divpart.in, file = "divpart.in.csv")

varpart.in <- fn.varpart.in(PCNM.pos=PCNM.pos.center, E.trans=E.trans.center, pool=pool.center, pool.t0=pool.t0.center)

write.csv(varpart.in, file = "varpart.in.csv")

Conditions.t0 <- list(n.new.sp = 0, n.extinction = 0, species.richness = length(unique(pool.t0$species)), n.ancestors = nrow(pool.t0), n.groups = length(unique(pool.t0$groups)), LC.t0=LC.t0, pool.t0=pool.t0)

system.time(sim.result <- fn.forward(Conditions.t0=Conditions.t0, E = landscape$E, nu=nu, m=0.81, nb.mat=nb.mat, n.timestep=100000, keep = FALSE, stop = TRUE))

write.csv(sim.result$Conditions$pool.t0, file = "pool.out.csv")

write.csv(sim.result$Conditions$LC.t0, file = "LC.out.csv")

write.csv(sim.result$species.richness, file = "n.species.csv")

write.csv(sim.result$n.groups, file = "n.groups.csv")

set.seed(3) # Although the replicability of fn.varpart.out was not ensured, the same seed was used as above.

pool.out <- read.csv("pool.out.csv", header=TRUE, row.names=1)

LC.out <- read.csv("LC.out.csv", header=TRUE, row.names=1)

LC.out.center <- LC.out[which(landscape$x >= 16 & landscape$x <= 25 & landscape$y >= 16 & landscape$y <= 25), ]

LC.out.center.no0 <- LC.out.center[ ,which(!colSums(LC.out.center)==0)]

pool.out.center <- pool.out[which(!colSums(LC.out.center)==0), ]

divpart.out <- fn.divpart.out(pool.t0=pool.out.center, LC.t0=LC.out.center.no0)

write.csv(divpart.out, file = "divpart.out.csv")

system.time(varpart.out <- fn.varpart.out(PCNM.pos=PCNM.pos.center, landscape=landscape.center, E.trans=E.trans.center, pool.t0=pool.out.center, LC.t0=LC.out.center.no0))

write.csv(varpart.out, file = "varpart.out.csv")