Appendix 5: SpatialDemography Example with Stochasticity

Alexander C. Keyel skeyel@gmail.com

2016-04-02

Citation

Keyel, A.C., Gerstenlauer, J.L.K. and Wiegand, K. YEAR. SpatialDemography: a new, spatially explicit, stage-structured, metacommunity model for studying plant demography. Ecography 000: 000-000

Introduction

In this example, we have added environmental stochasticity to Example 1. We illustrate two forms of environmental stochasticity that can be modeled. In the first stochastic environmental layer (called “environmental stochasticity”), the environmental conditions vary in time but are synchronized over the entire landscape. In the other stochastic layer (called “microsite stochasticity”), the environmental conditions vary within each cell asynchronously across the landscape. Note that other options exist (type ?env.change.type into the R console for more details after loading the spatialdemography package). Here, we model environmental stochasticity in the abstract (as a generic “environmental stochasticity” environmental layer) but in real applications, this could be replaced with one or more demographically-relevant stochastic environmental layers (e.g., climate). Here, the environmental conditions change at every time step. Species’ vital rates then change based on the species’ response traits.

Only the probability of transitioning from a juvenile to an adult responds to the stochastic environmental layer (to keep the example simple). Each species has a base matrix transition coeffiecent, which determines reproduction under optimal conditions. The base matrix transition coefficient is multiplied by a modifier that decreases quadratically as the environmental value diverges from the species’ optimal value. At each time step, the values of the environmental stochasticity layer are drawn from a normal distribution with a mean and standard deviation of 1. Here, species had optimal values for the environmental stochasticity layer of 1, 1.1, and 0.9. Thus, if the environmental stochasticity layer had a value of 1.1 for a time step, Species 2 would experience no reduction in the juvenile-to-adult transition rate, Species 1 would experience a reduction, and Species 3 would experience the greatest reduction in the transition rate. In the next time step, the value could be (e.g.) 0.8, then favoring Species 3 (but still with some reduction to the vital rate). In addition to stochasticity across the entire landscape, the microhabitat also changes. This allows modeling of variation within cells, without affecting other cells. We note that through the use of the copula package, environmental layers can be generated that change stochastically but are still correlated to one another (not demonstrated).

In addition, we have added an extinction threshold option to the model. When a species’ life stage drops below 1 it is rounded to 0 to prevent “resurrections”. To keep the example simple, the species and landscapes do not correspond to real landscapes and species.

Example 2

```
# Uncommented model steps were described in greater detail in Example 1 (see Appendix 3).

# Load the spatialdemography package
library(spatialdemography)


## Loading required package: Matrix


run.name = "spdem_ex2"
# Optionally set the working directory.
#setwd("C:/docs/beplants/Scripts/spatialdemography/vignettes")
scn = 1
s.lbl = ""
file.ending = sprintf("_%s", run.name)
DispPath = "dt/"
run.path = sprintf("%s/",run.name)
opath = sprintf("%soutputs/",run.path)
dir.create(opath,showWarnings = F, recursive = T)

ResultsFile = sprintf("%sResults%s.csv",opath,file.ending)
# Code to (optionally) delete previous versions of the ResultsFile and TimeFile.
do.del = 1
if (do.del == 1){
  unlink(ResultsFile) #Delete any previous ResultsFile
  }

# Read in inputs
input.path = sprintf("%sinputs/",run.path)
# initial.conditions.file contains many important model settings
initial.conditions.file = sprintf("%sInitial_conditions%s.csv"
                                  ,input.path,file.ending)
# Option to visualize file in R
#ic = read.csv(initial.conditions.file)
#View(ic) # Examine the initial conditions file to check that the correct file was read in

# settings.file A file to specify overall model settings.
settings.file = sprintf("%sSettings%s.csv",input.path,file.ending)
# Option to visualize file in R
#set = read.csv(settings.file)
#View(set)

# env.file A file to describe environmental layers and how they change.
env.file = sprintf("%sEnvironmental_layers%s.csv",input.path,file.ending)
# Option to visualize file in R
#ef = read.csv(env.file)
#View(ef)

# Set up landscape inputs
lpath = "spdem_ex2/landscape/"
```

```
#lc.file = sprintf("%slandcover_scn1.csv",lpath)
#h.file = sprintf("%sherbicide_scn1.csv",lpath)
#lc = read.csv(lc.file)

# Set up species file
spfile = sprintf("%sSpecies/Species%s.csv",run.path, file.ending)
sp.instr.file = NA
sp.resp.instr.rile = NA
#spf = read.csv(spfile)
#View(spf)

# Set up initial species locations
locations.file = sprintf("%slocations%s.csv", input.path, file.ending)
#locs = read.csv(locations.file)
#View(locs)

# Run ten times because each run will be slightly different due to stochasticity
set.seed(567891234)
n_examples = 10
for (scn in 1:n_examples){
  SpatialDemography(scn,s.lbl,file.ending,DispPath,run.path,opath,
                    ResultsFile,initial.conditions.file,settings.file,env.file,
                    spfile,sp.instr.file,sp.resp.instr.file,lpath,locations.file)
}


## Loading required package: multirich
```

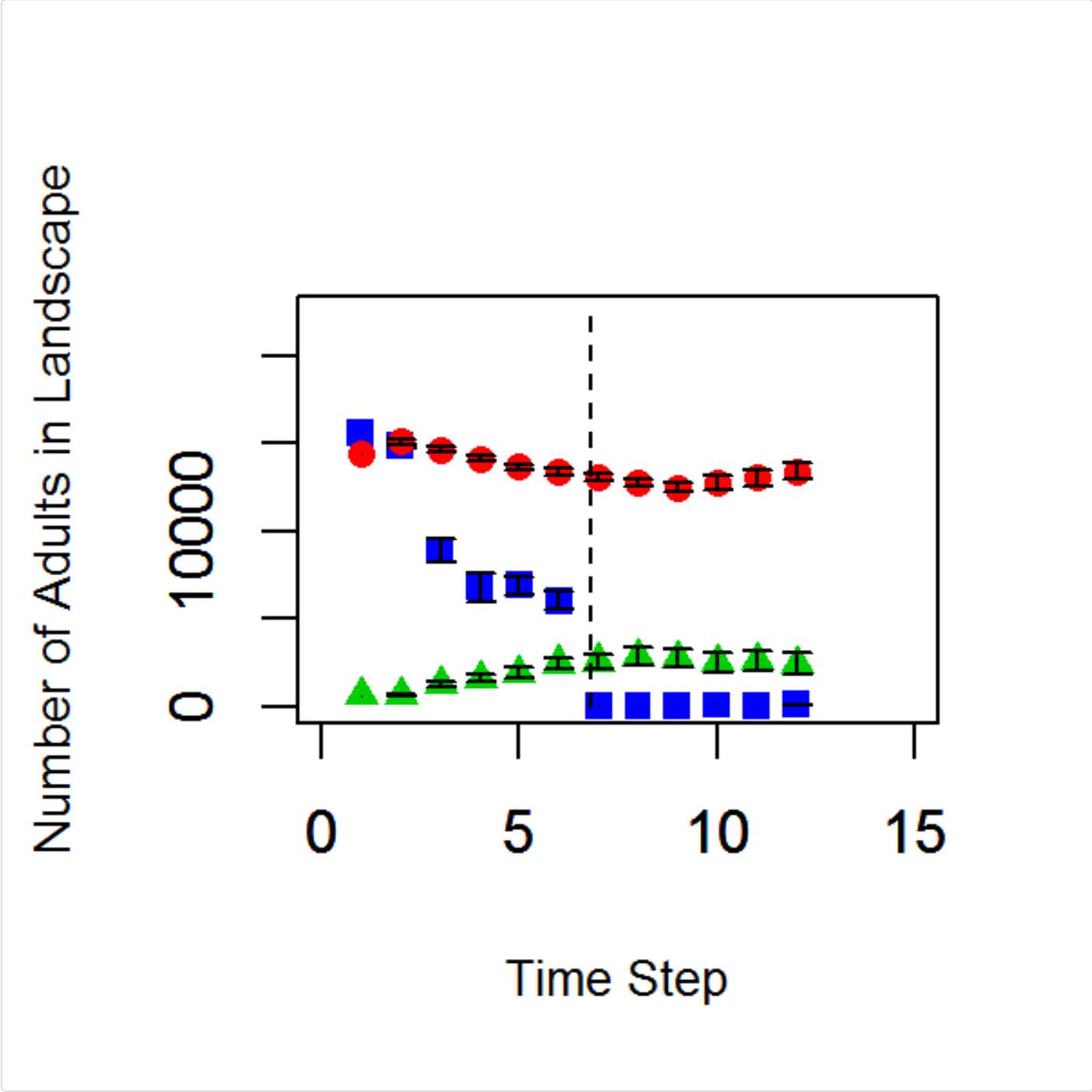
Summary figures

Now that the model has been run, we will generate summary figures similar to those presented in Fig. 1.

```
# Read in species' data

n_timesteps = 12
n_sp = 3
sp.lst = list(rep(list(NA), n_sp))
time.lst = list(rep(sp.lst, n_timesteps))
#example.lst = list(rep(time.lst, n_examples))
#example.lst = example.lst[[1]] # Get rid of single list added by R
time.lst = time.lst[[1]]
#str(time.lst)

# Loop through examples
for (i in 1:n_examples){
  spdat = read.csv(sprintf("spdem_ex2/Example2_%s/SpeciesStats_Adults_v2.csv", i))
  spdat = spdat[ , 2:4] # Drop runtime column and extra 3 columns

  # Loop through time steps
```

```
  for (j in 1:n_timesteps){

    # Loop through species
    for (k in 1:n_sp){

      if (i == 1){
        new.rec = spdat[j,k] # j is row, k is species in reformatted matrix
      }else{
        old.rec = time.lst[[j]][[k]]
        # append the current record to the vector with one entry per example
        new.rec = c(old.rec, spdat[j,k])
      }

      # Update the massive nested list
      time.lst[[j]][[k]] = new.rec
    }
  }
}

# Compute mean & sd for each time step & add to a data matrix
# two columns for each species, one for mean & one for se
my.df = rep(NA, n_timesteps * n_sp * 2)
my.df = matrix(my.df, ncol = (n_sp * 2))
rownames(my.df) = seq(1, n_timesteps)
colnames(my.df) = c("sp1.mean", "sp2.mean", "sp3.mean", "sp1.se", "sp2.se", "sp3.se")
for (j in 1:n_timesteps){
  for (k in 1:n_sp){
    this.rec = time.lst[[j]][[k]]
    new.mean = mean(this.rec)
    new.se = sd(this.rec) / sqrt(n_examples)
    my.df[j, k] = new.mean
    my.df[j, k + n_sp] = new.se
  }
}
# convert from matrix to data frame
my.df = as.data.frame(my.df)
my.df$RunTime = seq(1, n_timesteps)
# Set up graph parameters
x.limit = n_timesteps + 3
y.limit = 22500


# warning = FALSE suppresses the warnings that arise because some points
# have a se of 0 and this causes the arrows command to throw a warning.

# replicate Fig 5a with se bars
col.vec = c(4,2,3)
pch.vec = c(15, 16, 17)
x.label = "Time Step"
y.label = "Number of Adults in Landscape"
```

```
xaxt.val = yaxt.val = 's'
# Loop through species
for (m in 1:n_sp){
  if (m != 1){
    x.label = ""
    y.label = ""
    xaxt.val = 'n'
    yaxt.val = 'n'
  }

  # Plot means
  sp.name = names(my.df)[m]
  sp.means = my.df[[sp.name]]
  x = my.df$RunTime
  plot(x, sp.means, xlab = x.label, ylab = y.label,
     xlim = c(0,x.limit), ylim = c(0, y.limit), pch = pch.vec[m], col = col.vec[m], cex.lab = 0.8)

  # Plot se's
  se.name = names(my.df)[m + n_sp]
  sp.se = my.df[[se.name]]

  arrows(x,sp.means,x,(sp.means + sp.se),length = 0.04, angle = 90, xpd = T)
  arrows(x,sp.means,x,(sp.means - sp.se),length = 0.04, angle = 90, xpd = T)
  par(new = T) #Tell R to add next plot to existing plot
}
segments(6.8,0,6.8,y.limit, lty = 2)
```



In the above, blue squares indicate species 1, red circles species 2, and green triangles represent species 3. Error bars indicate standard error based on the 10 simulations. The dashed line indicates the onset of herbicide application. Note that Species 1's shows a greater reduction in abundance prior to the herbicide than it did in Example 1 (cf. Fig. 5a). When stochasticity was added to the model, it penalized each species if the species' optimum did not match the value for the stochastic layer. Consequently, stochasticity in this case could only hurt a species (note that transition matrix coefficients could have been increased to offset the decreases due to stochasticity). It is possible that Species 1's much poorer performance relative to Species 2 is due to an unequal penalty when adding stochasticity to the model. We leave the exact cause of this discrepancy as an exercise for the reader.

Next we examine just the plot for Species 3.

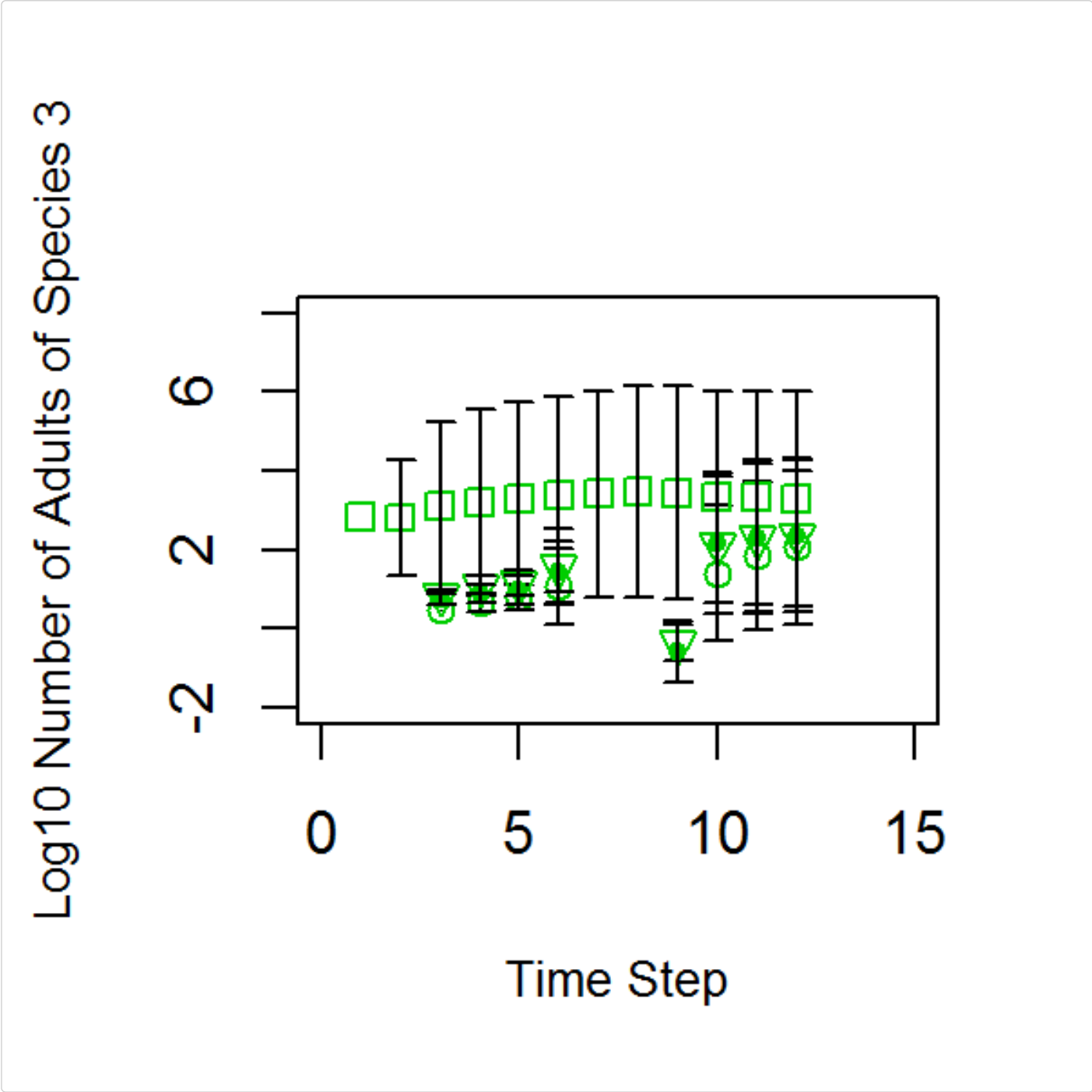
```
# replicate Fig 5b with standard error bars
```

```
n_cells = 4
cell.lst = list(rep(list(NA), n_cells))
timestep.lst = list(rep(cell.lst, n_timesteps))
timestep.lst = timestep.lst[[1]] # Remove extraneous outer list

# Read in data from each example
for (i in 1:n_examples){
  cell.dat = read.csv(sprintf("spdem_ex2/Example2_%s/SpeciesData.csv", i))
  # Restrict to species 3
  sp3.dat = cell.dat[cell.dat$Species == 3, ]
  # Restrict to adults
  sp3.dat = sp3.dat[sp3.dat$LifeStage == "Adults", ]

  for (j in 1:n_timesteps){
    for (k in 1:n_cells){
      if (i == 1){
        new.rec = sp3.dat[j , 3 + k] # 3 offsets for 1st 3 columns
      }else{
        old.rec = timestep.lst[[j]][[k]]
        new.rec = c(old.rec, sp3.dat[j , 3 + k])
      }

      timestep.lst[[j]][[k]] = new.rec # 3 offsets for 1st 3 columns
    }
  }
}

# Compute mean & sd for each time step & add to a data matrix
# two columns for each species, one for mean & one for se
my.df = rep(NA, n_timesteps * n_cells * 2)
my.df = matrix(my.df, ncol = (n_cells * 2))
rownames(my.df) = seq(1, n_timesteps)
colnames(my.df) = c("cell1.mean", "cell2.mean", "cell3.mean",
                    "cell4.mean", "cell1.se", "cell2.se", "cell3.se", "cell4.se")
for (j in 1:n_timesteps){
  for (k in 1:n_cells){
    this.rec = timestep.lst[[j]][[k]]
    new.mean = mean(this.rec)
    new.se = sd(this.rec) / sqrt(n_examples)
    my.df[j, k] = new.mean
    my.df[j, k + n_cells] = new.se
  }
}
my.df = as.data.frame(my.df)

## Examine Species 3's performance by cell
x.limit = 15
y.min = -2
y.limit = 8
```

```
# One point type for each cell
pch.vec = c(1, 6,20,0)
# Same color for each cell to make it clear it is Species 3
col.vec = c(3,3,3,3)

nam = names(my.df)
x = seq(1,n_timesteps)
# Plot Cell 1
sp.means = log(my.df$cell1.mean, 10)
plot(x,sp.means,
    xlim = c(0,x.limit), ylim = c(y.min,y.limit),
    pch = pch.vec[1], col = col.vec[1],
    xlab = "Time Step", ylab = "Log10 Number of Adults of Species 3", cex.lab = 0.8 )

    # Plot se's
    se.name = names(my.df)[m + n_sp]
    sp.se = log(my.df[[se.name]], 10)

    arrows(x,sp.means,x,(sp.means + sp.se),length = 0.04, angle = 90, xpd = T)


## Warning in arrows(x, sp.means, x, (sp.means + sp.se), length = 0.04, angle
## = 90, : zero-length arrow is of indeterminate angle and so skipped


    arrows(x,sp.means,x,(sp.means - sp.se),length = 0.04, angle = 90, xpd = T)


## Warning in arrows(x, sp.means, x, (sp.means - sp.se), length = 0.04, angle
## = 90, : zero-length arrow is of indeterminate angle and so skipped


# Plot Cells 2 - 4
for (i in 2:4){
    par(new = T) #Tell R to add next plot to existing plot
    this.cell = sprintf("cell%s.mean", i)
    sp.means = log(my.df[[this.cell]],10)
    plot(x, sp.means,
         xlim = c(0,x.limit), ylim = c(y.min, y.limit),
         pch = pch.vec[i], col = col.vec[i], xlab = "", ylab = "",
         xaxt = 'n', yaxt = 'n')

    # Plot se's
    se.name = names(my.df)[i + n_cells]
    sp.se = log(my.df[[se.name]], 10)
    arrows(x,sp.means,x,(sp.means + sp.se),length = 0.04, angle = 90, xpd = T)
    arrows(x,sp.means,x,(sp.means - sp.se),length = 0.04, angle = 90, xpd = T)

}


## Warning in arrows(x, sp.means, x, (sp.means + sp.se), length = 0.04, angle
```

```
## = 90, : zero-length arrow is of indeterminate angle and so skipped
```

```
## Warning in arrows(x, sp.means, x, (sp.means - sp.se), length = 0.04, angle
## = 90, : zero-length arrow is of indeterminate angle and so skipped
```



Example 2 shows a similar overall pattern as in Example 1, but with substantial variation in species' abundances due to the the added stochasticity.