Supplementary Methods and Code

How sample heterogeneity can obscure the signal of microbial interactions

David W. Armitage & Stuart E. Jones

Here, we provide the methods and code to generate the figures used in the article. All code can be run using the R programming language [1], and was last tested on R version 3.5.1. We begin by loading the required libraries and functions.

```
### Load the necessary libraries
library(packcircles)
library(ggplot2)
library(Hmisc)
library(EnvStats)
library(plyr)
library(reshape2)
library(cowplot)
library(ggforce)
library(ggfortify)
library(scales)
library(reshape)
library(proto)
library(dplyr)
#devtools::install_github("johannesbjork/LaCroixColoR")
library(LaCroixColoR)
```

Methods for Simpson's Paradox Section

To assess how Simpson's paradox affects our inferences about species interactions, we conducted a simulation experiment using spatially-explicit models of two competing microbial OTUs across a heterogeneous landscape [2]. This model is a spatial version of the familiar Lotka-Volterra model for two competing populations. The non-spatial model can be written as the difference equation

$$N_{i,t+1} = \phi_i(N_i, N_j, t) = N_{i,t} \exp\left\{ r_i \left(1 - \sum_{j=1}^{2} \frac{\alpha_{ij} N_{j,t}}{K_i}) \right) \right\} \qquad \big(i = 1, 2\big), \tag{S1}$$

where $N_{i,t}$ is the population density of OTU i at time t, r_i is the OTU's growth rate, K_i is its local carrying capacity, and α_{ij} are intraspecific (when $i = j$) and interspecific (when

$i \neq j$) competition coefficients. Note that here, unlike the gLV equation in the main text, increasing the α_{ij} parameters strengthens the impact of inter- and intraspecific competition.

To spatially extend this model, let us assume that these OTUs compete and disperse across a spatially-gridded array of n discrete habitat patches described by their Cartesian coordinates, $x = (x, y)$. These grid points are bordered by 8 neighboring patches, identified by coordinates $k = (k, l)$ — to or from which individuals can move. Finally, we include a term, D_i, describing each OTU's fractional dispersal between neighboring grid cells. Our full model becomes

$$(1 - D_i)\phi_i[N_{i,x,t}, N_{j,x,t}] + \frac{D_i}{8}\sum_{j=1}^{8}\phi_i[N_{i,k,t}, N_{j,k,t}] \tag{S2}$$

.

To re-create our analyses in R, we begin by defining some ancillary functions. These are modified from [3].

```
rmvn <- function(n, mu = 0, V = matrix(1)) {
# This function draws random samples from a multivariate
# normal distribution across a spatial grid, and is needed
# for creating our heterogeneous microbial "landscapes".
  p <- length(mu)
  if (any(is.na(match(dim(V), p))))
    stop("Dimension problem!")
  D <- chol(V)
  t(matrix(rnorm(n * p), ncol = p) %*% D + rep(mu, rep(n, p)))
}

discrete<-function(Ns,rs,K,as){
##  A discrete-time Lotka-Volterra competition function
# Ns are the population abundances
# rs are the growth rates
# K is the local carrying capacity
# as are the competition coefficients
  return(as.numeric(Ns*exp(rs*(1-((as%*%Ns)/K)))))
}

find_neighbour<-function(vec,dat=dat){
# A helper function returning line numbers of all
# neighbouring cells
  lnb<-which(dat[,"x"]%in%(vec["x"]+c(-1,-1,-1,0,1,1,1,0)) & dat[,"y"]%in%(vec["y"]+c(-1,0,1,1,1,0,-1,-1)))
  lnb<-lnb[which(lnb!=vec["lnb"])]
  return(lnb)
}

# Return intergroup correlation coefficients between OTUs
corfunc <- function(xx) return(data.frame(COR = cor(xx$N1, xx$N2)))
```

To demonstrate how Simpson's paradox influences our interpretation of interspecific interactions, we can make carrying capacities a function of the local environmental quality, $K_{i,x} = f(E_x)$, such that higher-quality patches can support larger populations of each OTU. To do this, we randomly populated a spatial grid with cells assigned one of three mean carrying capacities (1 = low, 2 = medium, 3 = high), keeping the total number of each group equal across the landscape (Fig. S1). To add some realism, we introduce normally-distributed noise around these values.

```r
set.seed(680318)
# Set up a square lattice region; should be a multiple of
# 12 to allow for 3 carrying capacities and 4 samples.
dat <- expand.grid(1:48, 1:48); names(dat) <- c("x", "y")

# Set up a distance matrix
distance <- as.matrix(dist(dat))
ntot <- nrow(dat)

# Next, we assign parameters and other important values
generation <- 1:60 #number of generations
# seed landscape with randomized populations
dat$N1 <- 0.5+rnorm(ntot,0,0.1)
dat$N2 <- 0.5+rnorm(ntot,0,0.1)
dat$lnb <- 1:ntot # index value for each cell
as<-matrix(c(1,1.05,1.05,1),ncol=2,byrow=TRUE) # Competition parameters. a_jj are set to 1
rs<-c(1.5,1.5) # Growth rate for each OTU
ds<-c(0.05,0.05) # dispersal fraction to move between adjacent cells
neigh<-apply(dat,1,function(x) find_neighbour(x,dat)) # get list of neighbor cells

# Here, we generate the carrying capacities for each cell.
# To do so, we first chose an autocorrelation parameter, psi, which
# controls the degree of clustering or spatial autocorrelation in
# our samples. We set this to an intermediate value, but note that our results
# are quantitatively robust across a wide range of psi.
psi <- 0.15
qual <- rmvn(1, rep(0, ntot), exp(-psi * distance))
dat$qual <- as.numeric(cut2(qual, g=3))
dat$K <- ifelse(dat$qual == 1, dat$qual + rnorm(ntot, 0.01, 0.1), dat$qual + rnorm(ntot, 0.01, 0.25))
list_dat<-list(dat) # transform to list for updating during simulations
```

Over a time period of 60 generations, we simulated births, competition, and dispersal across this spatial grid. Note that individuals migrating across the boundary of this system are assumed lost.

```r
for(i in generation){
  list_dat[[(i+1)]]<-rbind.fill(apply(list_dat[[i]],1,function(x){
    # model population dynamics within one grid cell
    intern<-(1-ds)*discrete(Ns=x[c("N1","N2")],rs=rs, K = x["K"], as=as)
    # get information on neighboring cells
    neigh_cell<-list_dat[[i]][neigh[[x["lnb"]]],]
    # compute the number of immigrants from the 8 adjacent grid cells
    imm<-(ds/8)*rowSums(apply(neigh_cell,1,function(y) discrete(Ns=y[c("N1","N2")],rs=rs,K = y["K"], as=as)))
    # save the results to a data frame in our list
    out<-data.frame(x=x["x"],y=x["y"],N1=intern[1]+imm[1],N2=intern[2]+imm[2],lnb=x["lnb"], qual =x["qual"], K =x["K"])
    return(out)
  }))
  print(i)
}
exmpl<- list_dat[[60]]; exmpl$qual <- factor(exmpl$qual)
# Create 4 equally-sized samples from the landscape (can be done many ways besides this).
exmpl$samps <- ifelse(exmpl$x <= sqrt(ntot)/2 & exmpl$y > sqrt(ntot)/2, "A",
                ifelse(exmpl$x > sqrt(ntot)/2 & exmpl$y > sqrt(ntot)/2, "B",
                       ifelse(exmpl$x > sqrt(ntot)/2 & exmpl$y <= sqrt(ntot)/2, "C",
                              "D")))
# Calculate the mean populations per sample for correlation
means <- aggregate(.~exmpl$samps, exmpl[c("N1", "N2")], FUN=mean); names(means) <- c("samps", "N1", "N2")
# Compute the correlation coefficeints
cor(means$N1, means$N2) # Sample correlation
mean(ddply(exmpl, .(qual), corfunc)$COR) # "True" mean correlation (averaged across K)
```

With these data, we can calculate interspecific correlation coefficients for both local samples and bulk samples, and plot our results. Here, we use Pearson's correlation coefficient, ρ, but note that our results are robust to other correlation metrics. From this simulation, across most random seeds and levels of spatial autocorrelation, we find that the estimated directions of aggregated and local interspecific correlations rarely match. To assess the generality of this phenomenon, we re-ran this simulation 1000 times, each time re-randomizing our microbial landscapes across a range of autocorrelation parameters. After each run, we computed interspecific correlations for both bulk samples, as well as the true interactions. In doing so, we find that in the vast majority of cases, we fail to uncover the true interaction term or even its correct sign (Fig. 1c). Furthermore, we also find that there is no single habitat configuration (based on values of ψ) that allows for consistently accurate correlation inference. Even well-mixed particulate samples (i.e., when $\psi > 1$), which might be assumed homogeneous with regard to sample composition, still have a high probability of returning incorrect correlation estimates (Fig. S2). R code for these simulations can be found below.

```r
## Code for simulating spatial Lotka-Volterra models over different randomized landscapes.
reps <- 1000 # Set number of different randomizations
correl <- matrix(NA, ncol = 2, nrow = reps) # initialize matrix to store values
psis <- seq(0.0001,1,length = reps) # create vector of parameters for landscape generation

for(j in 1:reps){
  set.seed(sample(1:1000000,1)) # random starting seed
  generation <- 1:60
  dat$N1 <- 0.5+rnorm(ntot,0,0.1); dat$N2 <- 0.5+rnorm(ntot,0,0.1); dat$lnb <- 1:ntot
  neigh<-apply(dat,1,function(x) find_neighbour(x,dat))
  psi <- psis[j]
  qual <- rmvn(1, rep(0, ntot), exp(-psi * distance))
  dat$qual <- as.numeric(cut2(qual, g=3))
  dat$K <- dat$qual + rnorm(ntot, 0.01, 0.25)
  list_dat<-list(dat)
  # Run the model for a single landscape
  for(i in generation){
    list_dat[[(i+1)]]<-rbind.fill(apply(list_dat[[i]],1,function(x){
      #population dynamics within one grid cell
      intern<-(1-ds)*discrete(Ns=x[c("N1","N2")],rs=rs, K = x["K"], as=as)
      #grab the neighbouring cells
      neigh_cell<-list_dat[[i]][neigh[[x["lnb"]]],]
      #the number of immigrant coming into the focal grid cell
      imm<-(ds/8)*rowSums(apply(neigh_cell,1,function(y) discrete(Ns=y[c("N1","N2")],rs=rs,K = y["K"], as=as)))
      out<-data.frame(x=x["x"],y=x["y"],N1=intern[1]+imm[1],N2=intern[2]+imm[2],lnb=x["lnb"], qual =x["qual"], K =x["K"])
      return(out)
    }))
  }
  #
  exmpl<- list_dat[[60]]; exmpl$qual <- factor(exmpl$qual)
  exmpl$samps <- ifelse(exmpl$x <= sqrt(ntot)/2 & exmpl$y <= sqrt(ntot)/2, "A",
                        ifelse(exmpl$x > sqrt(ntot)/2 & exmpl$y <= sqrt(ntot)/2, "B",
                               ifelse(exmpl$x <= sqrt(ntot)/2 & exmpl$y > sqrt(ntot)/2, "C",
                                      "D")))
# Aggregate and compute sample and true means
  means <- aggregate(.~exmpl$samps, exmpl[c("N1", "N2")], FUN=mean); names(means) <- c("samps", "N1", "N2")

  correl[j,1] <- mean(ddply(exmpl, .(qual), corfunc)$COR) # "True" mean correlation averaged over K
  correl[j,2] <- cor(means$N1, means$N2)  # Correlation of samples
  print(j)
}

correl_df <- data.frame(correl); names(correl_df) <- c("Local", "Aggregate")

# Plot the results in a histogram
ggplot() + geom_histogram(data = correl_df, aes(x = Aggregate),
                          colour="gray", fill="gray", binwidth = 0.1) +
  geom_vline(xintercept = mean(correl_df$Aggregate),lwd = 1, lty = "dashed") +
  geom_vline(xintercept = mean(correl_df$Local), lwd = 1, lty = "dotted", col = "red") +
  scale_y_continuous("Count", expand = c(0, 0)) +
  scale_x_continuous("Pearson correlation coefficient", expand = c(0, 0), limits = c(-1.1,1.1))
```

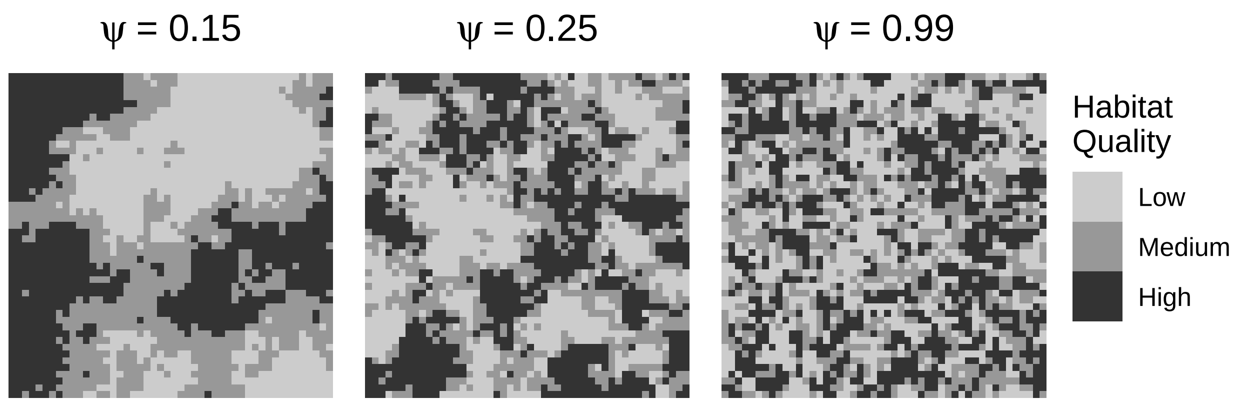

Figure S1: Examples of the artificial landscapes that we generated to assess Simpson's paradox. The parameter ψ, which can take values between 0 and $+\infty$, describes how quickly spatial autocorrelation declines with distance.

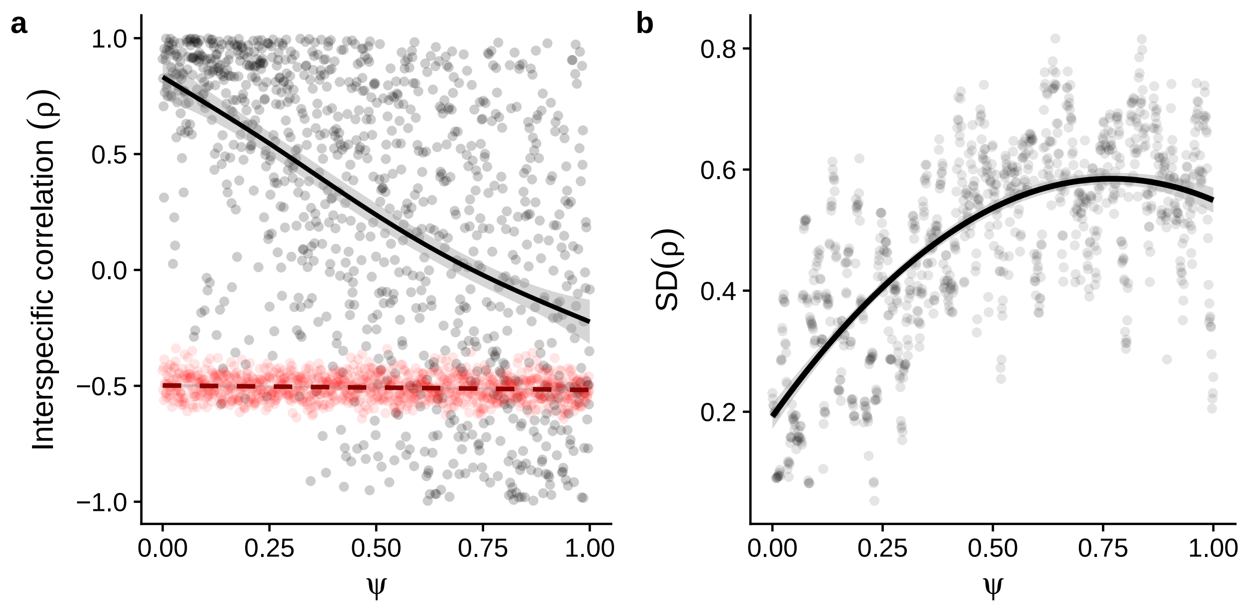

Figure S2: (a) As we increase the parameter ψ and thereby reduce the distance of spatial autocorrelation in our simulated landscapes, we find that the average estimated interspecific correlation coefficient (solid black line) approaches its "true" values (dashed red line). (b) However, the standard deviation of these correlation estimates (measured using a rolling window across ψ) is increases with ψ. Consequently, this tradeoff between accuracy and precision impacts our ability to correctly estimate interspecific correlations across a wide variety of habitat configurations.

Nonlinear spatial averaging

We begin this section by presenting an overview of scale transition theory for calculating nonlinear spatial averages, but refer interested readers to the original literature for a more in-depth treatment [4, 5, 6, 7, 8, 9]. Note that for notational clarity, we use overlines and angle brackets interchangeably to denote spatial averages.

As outlined in the main text, our goal is to take a spatial average over some nonlinear population dynamic function, $f(N)$, where N is heterogeneous across space, in order to quantify the scaled-up dynamics of the system, $\overline{f(N)}$. Chesson *et. al* [8] derives the general scale transition terms for nonlinear functions, which we illustrate here using the logistic growth model from the main text (eq. 2):

$$\frac{dN_x}{dt} = G(N_x) = \mu N_x \left(1 - \frac{N_x}{K}\right), \qquad (x = 1, \ldots, n) \tag{S3}$$

This equation can be approximated by a second degree Taylor polynomial, $P_2(N_x)$, expanded about the spatially-averaged density, $\bar{N} = [\sum_{x=1}^{n} N_x]/n$. This results in the following approximation:

$$\frac{dN_x}{dt} \approx P_2(N_x) \equiv G(\bar{N}) + G'(\bar{N})[N_x - \bar{N}] + \frac{1}{2}G''(\bar{N})[N_x - \bar{N}]^2. \tag{S4}$$

We can use this approximation to estimate the spatially-averaged nonlinear dynamics, $\overline{G(N)}$, by taking the spatial average of equation S4: $\big\langle P_2(N) \big\rangle = \big[\sum_{x=1}^{n} P_2(N_x)\big]/n$. Recalling that $\big\langle N_x - \overline{N} \big\rangle = 0$, and $\big\langle (N_x - \overline{N})^2 \big\rangle = \mathrm{Var}(N)$, the expression simplifies to

$$\frac{d\bar{N}}{dt} \equiv \overline{G(N)} \approx G(\bar{N}) + \frac{1}{2}G''(\bar{N})\mathrm{Var}(N), \tag{S5}$$

where the second term on the right hand side is a combined measure of nonlinearity, $\frac{1}{2}G''(\bar{N}) = -\mu/K$, and spatial variation, $\mathrm{Var}(N)$.

The spatial average $\overline{G(N)}$ can also be written as $\overline{rN}$ — the spatially-averaged product of local *per capita* growth rates and population abundances. Chesson *et al.* [8] show that this product is itself a function of two separate spatial averages plus a covariance term:

$$\overline{rN} = \tilde{r} \times \bar{N} = [\bar{r} + \mathrm{Cov}(r, v)]\bar{N}. \tag{S6}$$

Here, $\tilde{r}$ represents the individual average growth rate, and v_x are the relative local population densities, $N_x/\overline{N_x}$. Importantly, the value $\tilde{r}$ represents the local *per capita* growth rates averaged over all individuals in the population, accounting for the effects of dispersal between patches. This value is different than $\bar{r}$, which is the patch-averaged *per capita* growth rate. In mathematical terms:

$$\bar{r} = \frac{\sum_{x=1}^{n} r_x}{n}, \tag{S7}$$

and

$$\tilde{r} = \frac{\sum_{x=1}^{n} r_x N_x}{\sum_{x=1}^{n} N_x} = \frac{\sum_{x=1}^{n} r_x v_x}{n} = \overline{rv}. \tag{S8}$$

Following the linearity property of expectations, and since $\bar{v}$ necessarily equals one, $\bar{r}$ can also be expressed as

$$\tilde{r} = \overline{rv} = \bar{r} \times \bar{v} + \mathrm{Cov}(r, v) = \bar{r} + \mathrm{Cov}(r, v). \tag{S9}$$

The covariance term in equation S9 is called the *growth-density covariance*, and relates the local density of a population to the growth rates individuals experience within a local patch. It is an important quantity that can be used to understand and predict species coexistence in spatially-variable environments [8].

In many cases, local *per capita* growth rates, r_x, can themselves be nonlinear functions of interspecific and intraspecific densities, local environmental conditions, and any number of other state variables. To illustrate this, consider our logistic model of population growth (eq. 2). Replacing $\overline{G(N)}$ with $\overline{g(N)N}$ and following equation 6, we can write

$$\frac{d\bar{N}}{dt} = \overline{G(N)} = \overline{g(N)N} = \left[\overline{g(N)} + \mathrm{Cov}(g(N), v)\right]\bar{N} \tag{S10}$$

$$= \left[\overline{g(N)} + \overline{g'(N)}\mathrm{Cov}(N, v)\right]\bar{N} \tag{S11}$$

which can be approximated by expanding $g'(N)$ about $\bar{N}$ and taking the spatial average:

$$\overline{g'(N)} \approx \left\langle 0 + g'(\bar{N}) + g''(\bar{N})(N - \bar{N})\right\rangle. \tag{S12}$$

Substituting eq. S12 into eq. S11 and simplifying yields the expression

$$\overline{g(N)N} \approx \overline{g(N)} \times \bar{N} + g'(\bar{N})\mathrm{Var}(N). \tag{S13}$$

Finally, following equation S5, we can use the spatially-averaged Taylor polynomial to approximate $\overline{g(N)}$. Combining this with with equation S13, we arrive at the approximation for the nonlinear spatial average, $\overline{g(N)N}$:

$$\overline{g(N)N} \approx \left[g(\bar{N}) + \frac{1}{2}g''(\bar{N})\mathrm{Var}(N)\right]\bar{N} + g'(\bar{N})\mathrm{Var}(N). \tag{S14}$$

This equation is equivalent to $\overline{G(N)}$, but shows more generally how these terms are derived.

We note that the example above represents the special case of a model containing only one spatially-varying quantity — the abundances of a single OTU. However, this approach generalizes, with slight modification, to models containing more than one state variable. Such variables might include measures of environmental quality, resource availability, or the densities of competitors or predators. In the main text (equation 4), we show how to calculate a scale transition for the generalised Lotka-Volterra model describing M interacting OTUs (equation 1). This is done by first defining

$$r_{ix} = g(W_{ix}) = \mu_i + W_{ix} = \mu_i + \sum_{j=1}^{M} \alpha_{ij} N_{jx}, \tag{S15}$$

and then following the recipe for calculating the scale transition terms derived by Chesson *et. al* [8] and outlined above for $d\bar{N}_i/dt = \overline{g(W_i)N_i}$.

In the main text, we use a graphical representation of a logistic growth model to show how nonlinear averaging impacts our assessment of the scaled-up dynamics. Below, we outline how to recreate this figure using R.

```
## Code for re-creating figure 3 in the main text
samples <- 100 # Number of individual particles
Size <- EnvStats::rnormTrunc(samples, mean = 10, sd = 6, min = 0, max = 100) # For aesthetic purposes, generate a
     lognormally- distributed particle sizes
# Define local logistic growth function (dN/dt) following equation 2 in the main text.
pop.growth.rate <- function(mu,N,K){N*mu*(1-(N/K))}
# Define scale transition term. Here, it is exact and follows derivations in ref. 7.
scale.trans <- function(mu,N,K,S) {-(mu/K)*(exp(var(log(S)))-1)*(N^2)}
# Set the parameters for our logistic growth model
mu <- 1; K <- 900
# Select the different levels of variation for generating our simulated local populations
sds <- c(0.1, 0.25, 0.5, 1.25)
# Create a vector for N_bar
N <- seq(0,999,length.out = samples)
# Create vectors to store local simulated populations and regional growth rates
Ns <- matrix(NA, ncol = length(sds), nrow = samples)
regional <- matrix(NA, ncol = length(sds), nrow = samples)
# Calculate the regional population dynamics, iterating over different distributions of local population sizes
for(i in 1:length(sds)){
  Ns[,i] <- rlnormTrunc(samples, 6.5, sds[i], min = 1E-16, max = 1000) # simulate truncated lognormal population sizes
  S <- Ns[,i]
  regional[,i] <- pop.growth.rate(mu,N,K) + scale.trans(mu,N,K,S) # calculate scaled-up per capita growth rates, dN_bar/dt
}
```

Next, we prepare the data for plotting.

```
# Generate estimates and true values for instantaneous scaled-up dynamics
estimates <- pop.growth.rate(mu,N=mean(Ns[,4]),K) - (mu/K)*(var(Ns[,4]))
exact <- mean(pop.growth.rate(mu,N=Ns[,4],K))
# We can similarly estimate the scaled-up carrying capacity following the equation K/[1+var(N)]
K_scaled <- K/(1+(exp(var(log(Ns[,4])))-1))
# prepare data for plotting
scaletransition <- data.frame(N,pop.growth.rate(mu,N,K)); names(scaletransition) <- c("N", "dNdt")
samplerates <- data.frame(Ns[,4],pop.growth.rate(mu,N=(Ns[,4]),K))
names(samplerates) <- c("N", "dNdt")
regional <- as.data.frame(regional); names(regional) <- sds
regional <- reshape2::melt(regional)
regional$N <- rep(N,length(sds)); names(regional) <- c("sd", "dNdt","N")
regional$variance <- factor(as.numeric(levels(regional$sd))[regional$sd]^2)
local <- data.frame(N,pop.growth.rate(mu,N,K)); names(local) <- c("N", "dNdt")
```

And finally, we are ready to plot the simulated results for figure 3.

```
# First, we plot Fig. 3B, showing correspondence between the true mean (X)
# and the estimated mean (white diamond).
jensenplot = ggplot() +
  geom_point(data = samplerates, aes(x = N, y = dNdt, fill = N),
             size = 4, alpha = 0.6, pch = 21, color = "black") +
  geom_line(data = scaletransition, aes(x = N, y = dNdt), lwd = 1) +
  annotate("point", x = mean(Ns[,4]), y = pop.growth.rate(mu,N=mean(Ns[,4]),K),
           size = 6, pch = 23, fill = "black") +
  annotate("text", x = 0.05, y = pop.growth.rate(mu,N=mean(Ns[,4]),K),
           size = 4, colour = "black", label = expression(italic(paste("f(",bar("N"),")")))) +
  annotate("text", x = 0.05, y = estimates,
           size = 4, colour = "black", label = expression(italic(paste(bar("f(N)"))))) +
  annotate("text", x = mean(Ns[,4]), y = 10,
           size = 4, colour = "black", label = expression(italic(paste(bar(N))))) +
  scale_fill_gradientn(expression(paste(italic(N[x]))),
                       colors = rev(lacroix_palette("Apricot", n = 8, type = "continuous")[-8])) +
  geom_segment(aes(x = 50, y = pop.growth.rate(mu,N=mean(Ns[,4]),K),
                   xend = mean(Ns[,4]), yend = pop.growth.rate(mu,N=mean(Ns[,4]),K)),
               color = "black") +
  geom_segment(aes(x = 50, y = estimates,
                   xend = mean(Ns[,4]), yend = estimates),
               color = "black") +
  geom_segment(aes(x = mean(Ns[,4]), y = 20,
                   xend = mean(Ns[,4]), yend = pop.growth.rate(mu,N=mean(Ns[,4]),K)),
               color = "black", lty = "dashed") +
  annotate("point", x = mean(Ns[,4]), y = estimates,
           size = 5, stroke = 1.3, pch = 23, colour = "black", fill = "white") +
  annotate("point", x = mean(Ns[,4]), y = exact,
           size = 5, stroke = 1.3, pch = 4, colour = "black") +
  xlab(expression(paste("Density per particle ", italic((N[x]))))) +
  ylab(expression(italic(paste(f(N[x]))))) +
  geom_hline(yintercept = 0) + ylim(0,250)

# Plot figure 3C; where we have temporally extended the approximation by
# approximating Var(N) as [exp(var(log N)) - 1] * mean(N)^2
scaletrans = ggplot() +
  geom_line(data = local, aes(x = N, y = dNdt), lwd = 2) +
  geom_line(data = regional, aes(x = N, y = dNdt, color = variance)) +
  scale_y_continuous(limits = c(-15,250), expand = c(0,0)) +
  scale_x_continuous(limits = c(0,1000), expand = c(0,0)) +
  geom_hline(yintercept = 0) +
  ylab(expression(paste(italic("d"),bar(italic(N)),italic("/dt")))) +
  xlab(expression(paste(italic(bar(N))))) +
  scale_color_manual(expression(paste("Spatial\nvariance (",sigma^2, ")")), values = rev(lacroix_palette("Apricot", n = 5,
        type = "continuous")[-5]))

plot_grid(jensenplot, scaletrans, ncol = 2, align = "vh")
```

References

[1] Team RDC. R: A language and environment for statistical computing. Vienna, Austria: R Foundation for Statistical Computing; 2019. Available from: http://www.R-project.org.

[2] Solé RV, Bascompte J. Self-organization in complex ecosystems. Princeton, NJ: Princeton University Press; 2006.

[3] Hertzog LR. Exploring spatial patterns and coexistence; 2016. Available from: http://rpubs.com/hughes/188614.

[4] Chesson P. Matters of scale in the dynamics of populations and communities. In: Floyd RB, Sheppard AW, de Barro PJ, editors. Frontiers of Population Ecology. Melbourne, AU: CSIRO Publishing; 1996. 353–368.

[5] Chesson P. General theory of competitive coexistence in spatially-varying environments. Theoretical Population Biology. 2000;58:211–237.

[6] Chesson P. Scale transition theory with special reference to species coexistence in a variable environment. Journal of Biological Dynamics. 2009;3:149–163.

[7] Chesson P. Scale transition theory: Its aims, motivations and predictions. Ecological Complexity. 2012;10:52–68.

[8] Chesson P, Donahue MJ, Melbourne BA, Sears ALW. Scale transition theory for understanding mechanisms in metacommunities. In: Holyoak M, Leibold MA, Holt RD, editors. Metacommunities: spatial dynamics and ecological communities. Chicago: University Of Chicago Press; 2005. 279–306.

[9] Melbourne BA, Chesson P. The scale transition: Scaling up population dynamics with field data. Ecology. 2006;87:1478–1488.