Supplement to: *Intermediate abundance promotes speciation when dispersal is limited*

S1 Computation set-up

We created a make use of a custom R package *abundolism* which can be installed from github.

```
devtools::install_github("ajrominger/abundolism")
```

The following R computing environment is used for all simulations and analyses:

```
library(abundolism)
library(dplyr)
```

```
Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(ggpointdensity)
library(knitr)

sessionInfo() |>
    print(locale = FALSE)
```

```
R version 4.5.2 (2025-10-31)
Platform: aarch64-apple-darwin20
Running under: macOS Sequoia 15.0.1

Matrix products: default
BLAS:   /System/Library/Frameworks/Accelerate.framework/Versions/A/Frameworks/vecLib.framewo
LAPACK: /Library/Frameworks/R.framework/Versions/4.5-arm64/Resources/lib/libRlapack.dylib;  

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] knitr_1.50           ggpointdensity_0.2.1 ggplot2_4.0.1
[4] dplyr_1.1.4          abundolism_0.1.0

loaded via a namespace (and not attached):
 [1] vctrs_0.6.5        cli_3.6.5          rlang_1.1.6        xfun_0.52
 [5] generics_0.1.4     S7_0.2.1           jsonlite_2.0.0     glue_1.8.0
 [9] htmltools_0.5.8.1  scales_1.4.0       rmarkdown_2.29     grid_4.5.2
[13] evaluate_1.0.4     tibble_3.3.0       MASS_7.3-65        fastmap_1.2.0
[17] yaml_2.3.10        lifecycle_1.0.4    compiler_4.5.2     RColorBrewer_1.1-3
[21] Rcpp_1.1.0         pkgconfig_2.0.3    rstudioapi_0.17.1  farver_2.1.2
[25] digest_0.6.39      R6_2.6.1           tidyselect_1.2.1   pillar_1.11.0
[29] magrittr_2.0.4     withr_3.0.2        tools_4.5.2        gtable_0.3.6
```

The quarto (Allaire et al. 2025) document used to generate this supplement can be found with the R command `system.file("ab_supp.qmd", package = "abundolism")`.

S2 Simulation details

Foo foo foo

S3 Simulation experiment details

Simulation parameters are drawn from uniform distributions with the following minimum and maximum limits:

```r
par_range <- data.frame(
    param = c("la", "mu", "g", "m_prop", "nu", "tau", "xi"),
    min   = c(1,    1,    0,   0,        0,     0,    0.5),
    max   = c(10,   10,   0.1, 0.1,      0.001, 4,    2)
)

# fixed params
np <- 2
nstep <- 10000

# number of simulation replicates
nrep <- 10000

# format param names for printing
mutate(par_range,
       param = sprintf("`%s`", param)) |>
    kable(format.args = list(scientific = FALSE))
```

param	min	max
`la`	1.0	10.000
`mu`	1.0	10.000
`g`	0.0	0.100
`m_prop`	0.0	0.100
`nu`	0.0	0.001
`tau`	0.0	4.000
`xi`	0.5	2.000

The parameters governing number of populations (`np`) and number of time steps (`nstep`) do not vary across simulations and are set to `np` = 2 and `nstep` = 10^4. We simulate a total of `nrep` = 10^4 replicates, each with a unique, randomly sampled set of parameter values.

S3.1 Running the simulation experiment

Parameter values for each simulation replicate are stored in a `data.frame` with one column for each parameter.

```r
# change or remove for different results
set.seed(123)
```

```
# data.frame of randomly sampled param values
# one param per column
pars <- sapply(par_range$param, function(p) {
    runif(nrep,
          par_range[par_range$param == p, "min"],
          par_range[par_range$param == p, "max"])
}) |>
    as.data.frame()
```

```
# now run the simulation, see `?sim_BDI_spec` for details
# on input and output
sim_dat <- sim_BDI_spec(la = pars$la, mu = pars$mu, g = pars$g,
                        m_prop = pars$m_prop, nu = pars$nu,
                        tau = pars$tau, xi = pars$xi,
                        np = np, nstep = nstep)
```

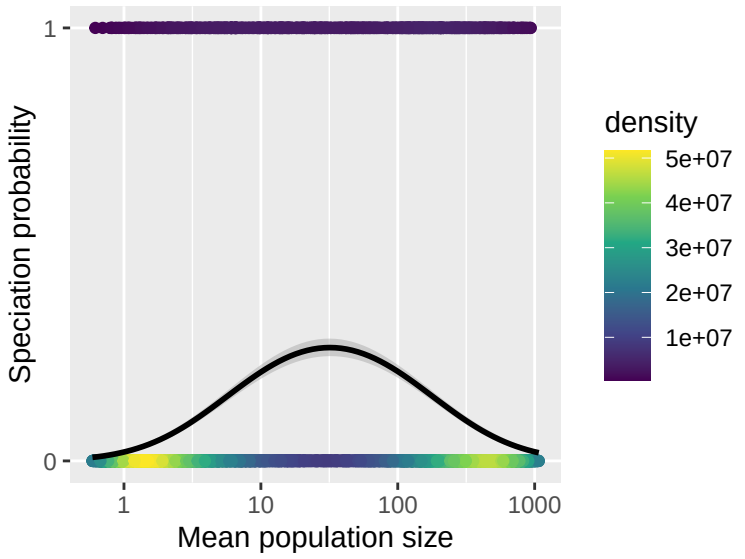
Now we can plot the results and find out how abundance relates to speciation in this model

```
ggplot(sim_dat,
       aes(x = tau, speciation)) +
    geom_pointdensity(method = "kde2d") +
    scale_shape_binned() +
    scale_color_viridis_c() +
    # scale_x_log10() +
    geom_smooth(method = "glm",
                method.args = list(family = "binomial"),
                color = "black")

ggplot(sim_dat, aes(niter, time)) +
    geom_pointdensity(method = "kde2d") +
    scale_color_viridis_c() +
    # scale_x_log10() +
    scale_y_log10() +
    facet_wrap(vars(speciation))


ggplot(sim_dat, aes(time, mean_pop_size)) +
    geom_point(data = select(sim_dat, !speciation),
               mapping = aes(time, mean_pop_size), color = "gray50") +
    geom_pointdensity() +
    facet_wrap(vars(speciation)) +
    scale_x_log10() +
```

```
    scale_y_log10() +
    scale_color_viridis_c(trans = "log10")

ggplot(sim_dat, aes(time, mean_pop_size)) +
    geom_point(data = select(sim_dat, !speciation),
               mapping = aes(time, mean_pop_size), color = "gray50") +
    geom_pointdensity() +
    facet_wrap(vars(speciation)) +
    scale_x_log10() +
    scale_y_log10() +
    scale_color_viridis_c(trans = "log10")

ggplot(sim_dat[sim_dat$time < 100, ], aes(time, mean_pop_size)) +
    geom_pointdensity() +
    facet_wrap(vars(speciation)) +
    scale_x_log10() +
    scale_y_log10() +
    scale_color_viridis_c(trans = "log10")

ggplot(sim_dat, aes(time)) +
    geom_histogram() +
    facet_wrap(vars(speciation), ncol = 1) +
    scale_x_log10()
```

This is the main figure

```
sim_fig <- ggplot(sim_dat,
                  aes(x = mean_pop_size, y = speciation)) +
    geom_pointdensity(method = "kde2d") +
    scale_color_viridis_c() +
    # scale_size_continuous(range = c(0.001, 4)) +
    scale_x_log10() +
    scale_y_continuous(breaks = c(0, 1)) +
    xlab("Mean population size") +
    ylab("Speciation probability") +
    theme(panel.grid.minor.y = element_blank()) +
    geom_smooth(method = glm, formula = y ~ x + I(x^2),
                method.args = list(family = "binomial"), color = "black")

sim_fig
```



References

Allaire, J. J., Teague, C., Scheidegger, C., Xie, Y., Dervieux, C. and Woodhull, G. 2025. Quarto.