Variation in thermal pressures and resource availability drives disease dynamics

Cole Brookson David Vasseur

Table of contents

0.1 Model

Vinton and Vasseur (2022) provide a basic model for temperature-dependent consumer-resource dynamics with a chemostat model of resource, R, supply, given by

$$\frac{dR}{dt} = D(S-R) - f(R,T)C, \tag{1}$$

where there is an inflow density S and an outflow rate D and $f(R,T)$ i the functional response of R with respect to temperature T. The biomass change of the consumer C is given by

$$\frac{dC}{dt} = (1-\delta)f(R,T) - m(T)C, \tag{2}$$

where $(1-\delta)$ is the consumption efficiency (denoted as a fraction), and m is the rate of respiration. We assume a Boltzmann-Arrhenius relationship and approximate that function $m(T)$ with

$$m(T) = m_a e^{m_b T} + m_c, \tag{3}$$

with m_a, m_b and m_c all > 0. The functional response is a standard type II, with attack rate a, handling rate $(1/h)$, but re-state according to the Michaelis-Menten form of

$$f(R,T) = I_{max}(T) \times \frac{R}{R + R_{\text{half}}}, \tag{4}$$

and $I_{max}T$ is the maximum uptake rate which we state as equivalent to the handling rate $I_{max}T \equiv 1/h$, and then the half-saturation density is made equivalnt via $R_{\text{half}} \equiv \frac{1}{a \times h}$. Note that the resource saturation is reached by $R/(R_{\text{half}} + R)$, which is independent of I_{max}. Last,

$$I_{max}(T) = e^{-(T-T_I)^2 \gamma}, \tag{5}$$

is the equation governing the relationship with temperature, where T_I is the optimum temperature for consumption, and γ is the breadth of response.

0.1.1 Equilibrium Solutions

Vinton and Vasseur (2022) showed that there is a coexistence equilibrium solution where

$$R_e = \frac{mR_{\text{half}}}{(1-\delta)I_{max}(T) - m}, \tag{6}$$

and

$$C_e = \frac{D(S - R_e)(R_e - R_{\text{half}})}{I_{max}(T)R_e}, \tag{7}$$

0.1.2 Brute-force Computation

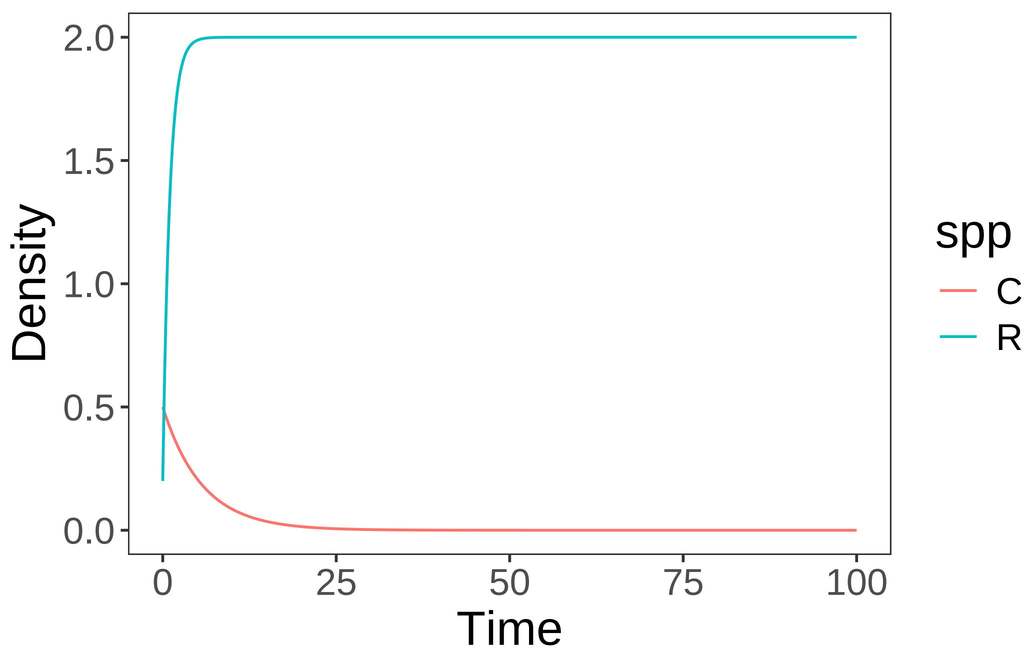
We use the standard set of model parameters for the chemostat dynamics, $R_{\text{half}} = 2, T_I = 25, \gamma = 150, m_a = 0.01, m_b = 0.1, m_c = 0.05, D = 1, S = 1$.

With these values, we set up a brute-force computation that draws values of R

Here's a single run for R:

```r
# Load the deSolve package
library(deSolve)
library(ggplot2)
library(magrittr)
library(dplyr)
```

```r
# Define the differential equations
consumer_resource_model <- function(time, state, parameters) {
    R <- state[1]
    C <- state[2]
    T <- parameters["T"]
    D <- parameters["D"]
    S <- parameters["S"]
    delta <- parameters["delta"]
    m_a <- parameters["m_a"]
    m_b <- parameters["m_b"]
    m_c <- parameters["m_c"]
    a <- parameters["a"]
    h <- parameters["h"]
    e <- parameters["e"]
    gamma <- parameters["gamma"]
    T_I <- parameters["T_I"]
    R_half <- parameters["R_half"]

    # Functional response f(R, T)
    I_max_T <- exp(-(T - T_I)^2 * gamma)
    f_R_T <- I_max_T * R / (R + R_half)

    # Respiration rate m(T)
    m_T <- m_a * exp(m_b * T) + m_c

    # Differential equations
    dR_dt <- D * (S - R) - f_R_T * C
    dC_dt <- (1 - delta) * f_R_T - m_T * C

    return(list(c(dR_dt, dC_dt)))
}

# Define the parameters
parameters <- c(
    T = rnorm(1, mean = 25, sd = 1), # Random normal temperature
    D = 1, # Outflow rate
    S = 2, # Inflow resource density
    delta = 0.2, # Consumption efficiency
    e = 0.5,
    m_a = 0.01,
    m_b = 0.1,
    m_c = 0.05,
    a = 0.5, # Attack rate
    h = 0.1, # Handling rate
    gamma = 150, # Breadth of response
    T_I = 25, # Optimum temperature for consumption
    R_half = 0.5 # Half-saturation density
)

# Define initial conditions for R and C
state <- c(R = 0.2, C = 0.5)

# Set time points for the simulation
times <- seq(0, 100, by = 0.1)

# Run the simulation using deSolve's ode function
output <- data.frame(deSolve::ode(
    y = state, times = times,
    func = consumer_resource_model,
    parms = parameters
)) |>
    tidyr::pivot_longer(
        cols = c(R, C),
        names_to = "spp"
    )


# Plot the results
ggplot2::ggplot(
    data = output
) +
    geom_line(aes(x = time, y = value, colour = spp)) +
    theme_bw() +
    theme(
        text = element_text(size = 18),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        strip.background = element_blank()
    ) +
    labs(x = "Time", y = "Density")
```



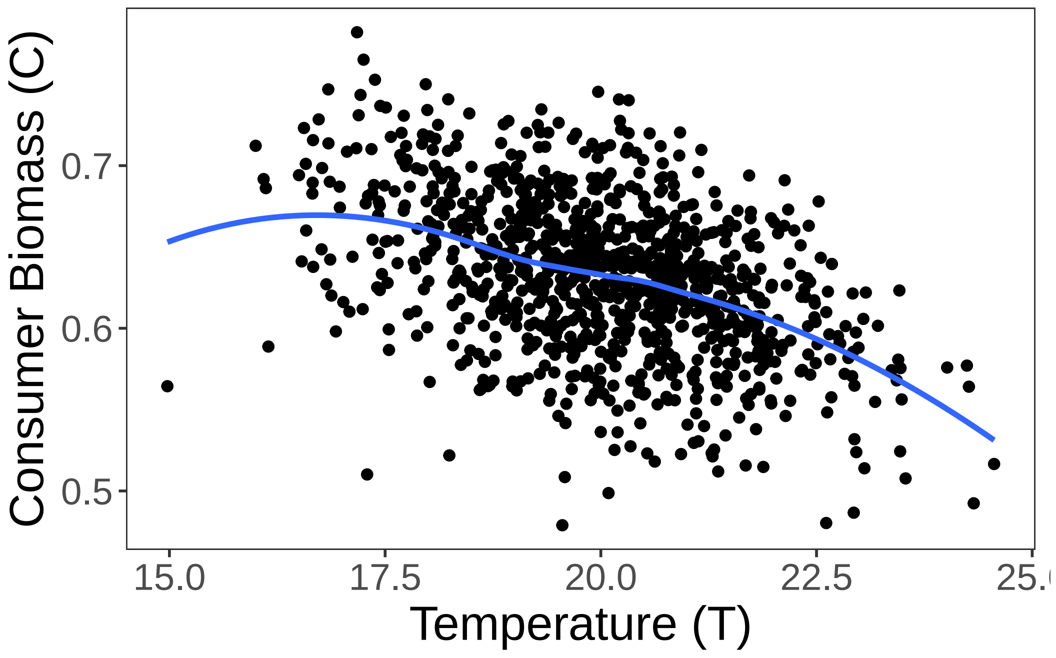
Source: Article Notebook

Now we do it drawing distributions and we can look at the equilibrium conditions:

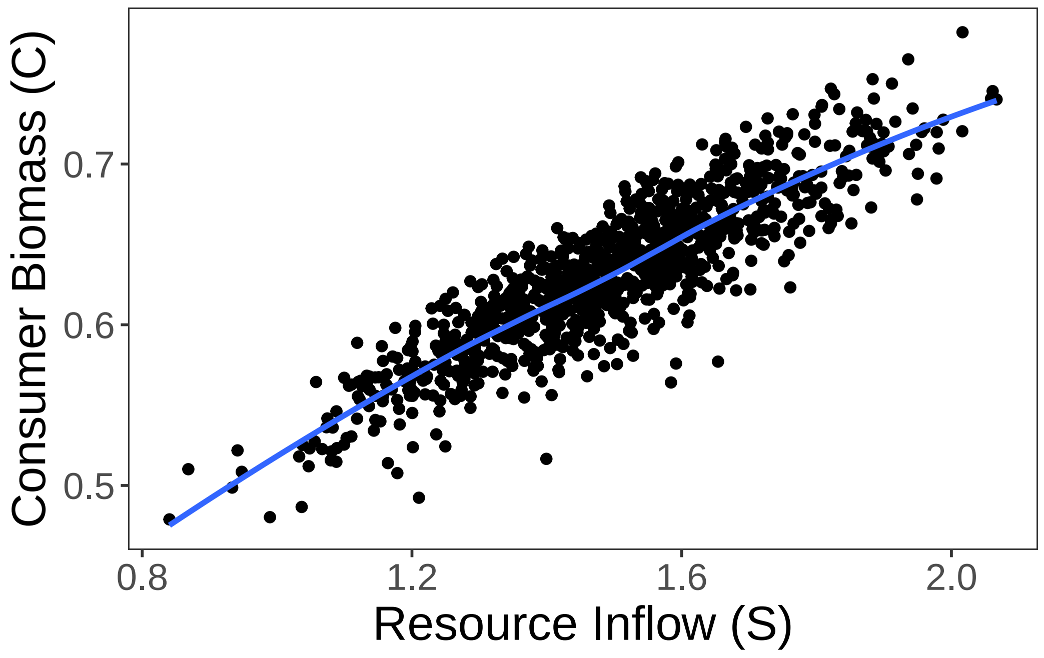
```r
library(plot3D)
library(reshape2)
library(plotly)
```

```r
consumer_equil_resource_model <- function(time, state, parameters) {
    R <- state[1]
    C <- state[2]
    T <- parameters["T"]
    D <- parameters["D"]
    S <- parameters["S"]
    delta <- parameters["delta"]
    m_a <- parameters["m_a"]
    m_b <- parameters["m_b"]
    m_c <- parameters["m_c"]
    a <- parameters["a"]
    h <- parameters["h"]
    gamma <- parameters["gamma"]
    T_I <- parameters["T_I"]
    R_half <- parameters["R_half"]
    # Functional response f(R, T)
    I_max_T <- exp(-(T - T_I)^2 * gamma)
    f_R_T <- I_max_T * R / (R + R_half)

    # Respiration rate m(T)
    m_T <- m_a * exp(m_b * T) + m_c

    # Differential equations
    dR_dt <- D * (S - R) - f_R_T * C
    dC_dt <- (1 - delta) * f_R_T - m_T * C

    return(list(c(dR_dt, dC_dt)))
}

# Define the parameter ranges (means and sds for S and T)
S_mean <- 1.5
S_sd <- 0.2
T_mean <- 20
T_sd <- 1.5

# Set other model parameters (fixed values)
parameters_base <- c(
    D = 0.1, # Outflow rate
    delta = 0.2, # Consumption efficiency
    m_a = 0.1,
    m_b = 0.05,
    m_c = 0.02,
    a = 0.5, # Attack rate
    h = 0.1, # Handling rate
    gamma = 0.05, # Breadth of response
    T_I = 20, # Optimum temperature for consumption
    R_half = 0.1 # Half-saturation density
)

# Define initial conditions for R and C
state <- c(R = 1, C = 0.5)

# Time points for the simulation
times <- seq(0, 500, by = 0.1) # Extended time for equilibrium

# Number of draws for S and T
num_draws <- 1000
```

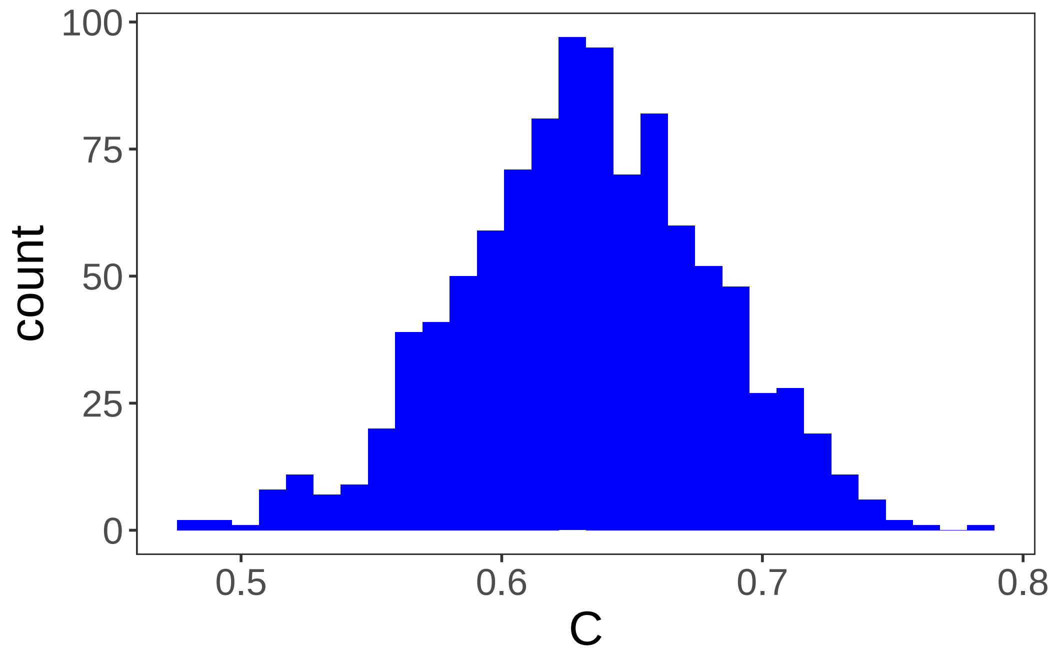
```r
# Storage for equilibrium C values, and the corresponding T and S values
results <- data.frame(
    S = numeric(num_draws),
    T = numeric(num_draws),
    C = numeric(num_draws)
)

# Run the simulation for each draw
for (i in 1:num_draws) {
    # Draw random values for S and T from normal distributions
    parameters <- parameters_base
    parameters["S"] <- rnorm(1, mean = S_mean, sd = S_sd)
    parameters["T"] <- rnorm(1, mean = T_mean, sd = T_sd)

    # Run the simulation
    output <- ode(y = state, times = times,
    func = consumer_resource_model, parms = parameters)

    # Store the final values of S, T, and the equilibrium value of C
    results$S[i] <- parameters["S"]
    results$T[i] <- parameters["T"]
    results$C[i] <- tail(output[, "C"], n = 1) # Equilibrium value of C
}

# Plot the relationship between C and T
ggplot(results, aes(x = T, y = C)) +
    geom_point() +
    geom_smooth(method = "loess", se = FALSE) +
    labs(x = "Temperature (T)", y = "Consumer Biomass (C)") +
    theme_bw() +
    theme(
        text = element_text(size = 18),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        strip.background = element_blank()
    )
```



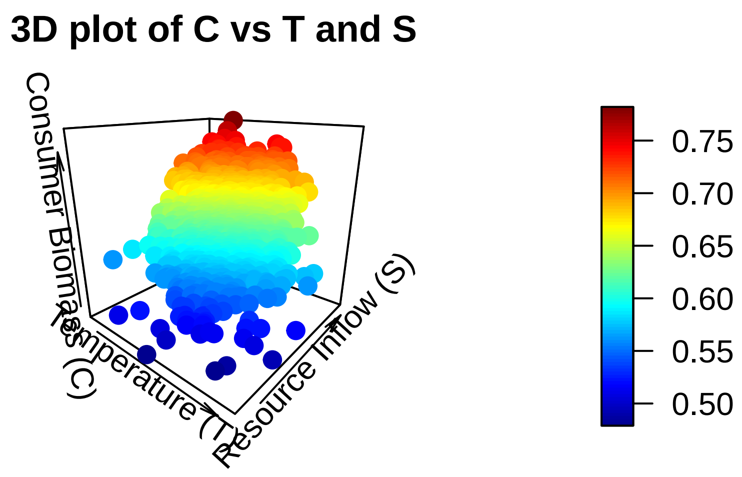
```r
# Plot the relationship between C and S
ggplot(results, aes(x = S, y = C)) +
    geom_point() +
    geom_smooth(method = "loess", se = FALSE) +
    labs(x = "Resource Inflow (S)", y = "Consumer Biomass (C)") +
    theme_bw() +
    theme(
        text = element_text(size = 18),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        strip.background = element_blank()
    )
```



```r
ggplot(results)  +
    geom_histogram(aes(x = C), fill = "blue") +
    theme_bw() +
    theme(
        text = element_text(size = 18),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        strip.background = element_blank()
    )
```



```
# 3D plot of C with respect to both T and S
scatter3D(
    x = results$T, y = results$S, z = results$C,
    pch = 19, cex = 1.2, colkey = TRUE,
    main = "3D plot of C vs T and S",
    xlab = "Temperature (T)", ylab = "Resource Inflow (S)",
    zlab = "Consumer Biomass (C)",
    theta = 40, phi = 20
)
```



```
# # Create a 2D grid of T and S
# plot_matrix <- t(reshape2::acast(results, S~T, value.var="C"))
# d_df <- as.matrix(results[, c("C", "S", "T")], rownames.force = NA)
# plot_ly(z=~d_df) %>% add_surface()

# plotly::plot_ly(
#       z=~d_df
#     ) %>%
#   add_surface() %>%
#   layout(
#     title = "",
#     scene = list(
#       xaxis = list(title = "S"),
#       yaxis = list(title = "T"),
#       zaxis = list(title = "C"),
#       camera = list(eye = list(x = 1.95, y = -1.25, z = 1.25))
#     ))
```

Source: Article Notebook

Vinton, Anna C., and David A. Vasseur. 2022. "Resource Limitation Determines Realized Thermal Performance of Consumers in Trophodynamic Models." *Ecology Letters* 25 (10): 2142–55. https://doi.org/10.1111/ele.14086.