POMP Model Analysis

Sijia Wang

2025-05-28

```
library(conflicted)
conflict_prefer("map", "pomp")
conflict_prefer("filter", "dplyr")

library(pomp)
library(ggplot2)
library(tidyverse)
library(dplyr)
library(doFuture)
library(doRNG)
library(foreach)
knitr::opts_chunk$set(echo = TRUE)
```

POMP Model Fitting

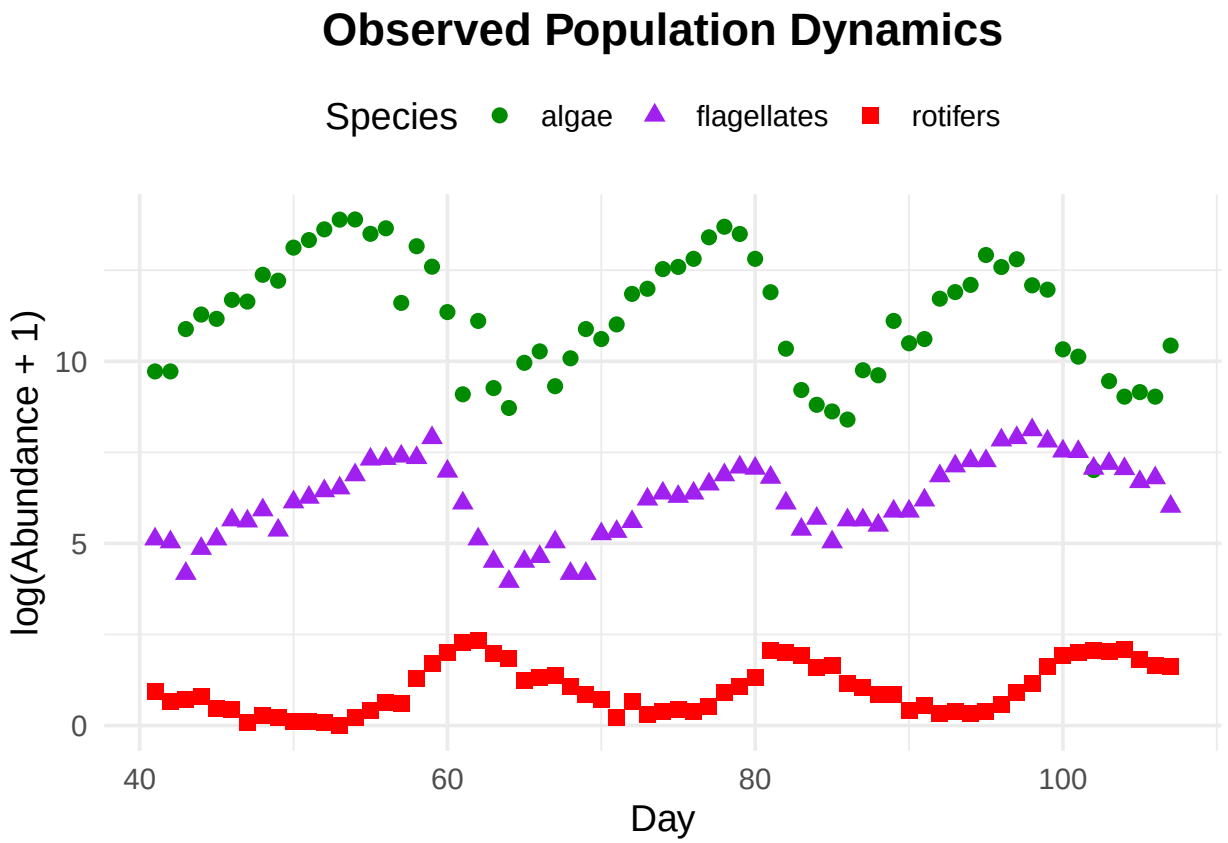
We use the **orange panel data between days 41 and 107** to build and analyze a Partially Observed Markov Process (POMP) model for the three-species food web system.

```
# Load the data (adjust path as needed)
X123 <- read.csv("NoEvoData=SE=Feb14 2012.csv")

# Prepare orange panel data (log-transformed)
orange_data <- data.frame(
  day = X123$day,
  algae = log1p(X123$Algae.orange1),
  flagellates = log1p(X123$Flag.orange1),
  rotifers = log1p(X123$Rot.orange1)
)

# Subset data between day 41 and 107
orange_data <- orange_data[orange_data$day >= 41 & orange_data$day <= 107, ]

# Summarize log-transformed data
summary(orange_data)
```

```
##       day            algae         flagellates       rotifers
##  Min.   : 41.0   Min.   : 7.014   Min.   :3.951   Min.   :0.0000
##  1st Qu.: 57.5   1st Qu.: 9.856   1st Qu.:5.371   1st Qu.:0.4249
##  Median : 74.0   Median :11.283   Median :6.211   Median :0.8550
##  Mean   : 74.0   Mean   :11.215   Mean   :6.183   Mean   :1.0132
```

```
##  3rd Qu.: 90.5   3rd Qu.:12.589   3rd Qu.:7.057   3rd Qu.:1.6318
##  Max.   :107.0   Max.   :13.892   Max.   :8.114   Max.   :2.3373
```

```r
orange_long <- orange_data %>%
  pivot_longer(cols = c(algae, flagellates, rotifers),
               names_to = "species",
               values_to = "log_abundance")

ggplot(orange_long, aes(x = day, y = log_abundance, color = species)) +
  geom_point(size = 2.5, aes(shape = species)) +
  scale_color_manual(values = c(
    algae = "green4",
    flagellates = "purple",
    rotifers = "red"
  )) +
  scale_shape_manual(values = c(
    algae = 16, flagellates = 17, rotifers = 15
  )) +
  labs(
    title = "Observed Population Dynamics",
    x = "Day", y = "log(Abundance + 1)",
    color = "Species", shape = "Species"
  ) +
  theme_minimal(base_size = 14) +
  theme(
    legend.position = "top",
    plot.title = element_text(hjust = 0.5, face = "bold")
  )
```




```
# Define state and parameter names
statenames <- c("S", "A", "R", "F")  # substrate, algae, rotifers, flagellates

paramnames <- c(
  "delta", "kA", "kR", "r", "g", "h",
  "alphaA", "alphaF", "eta", "IF",
  "S_0", "A_0", "R_0", "F_0"
)

# Process model (Euler method)
rproc <- euler(
  step.fun = Csnippet("
    double dS = delta * (1 - S) - S * r * A / (kA + S);
    double dA = A * (
      r * S / (kA + S)
      - g * R / (kR + A + alphaF * F)
      - h * F / (1 + alphaA * A)
      - delta
    );
    double dR = R * (
      g * A / (kR + A + alphaF * F)
      + eta * F / (kR + A + alphaF * F)
      - delta
    );
    double dF = F * (
      h * A / (1 + alphaA * A)
```

```
      - eta * R / (kR + A + alphaF * F)
      - delta
    ) + IF;

    S += dS * dt;
    A += dA * dt;
    R += dR * dt;
    F += dF * dt;

    if (S > 1) S = 1;
    if (S < 0) S = 0;

    if (A > 1e5) A = 1e5;
    if (A < 10) A = 10;

    if (R > 1e4) R = 1e4;
    if (R < 0.1) R = 0.1;

    if (F > 1e4) F = 1e4;
    if (F < 10) F = 10;
  "),
  delta.t = 0.05
)

rmeasure <- Csnippet("
  double sigma_A = 0.1;
  double sigma_F = 0.1;
  double sigma_R = 0.1;

  algae = rnorm(log1p(fmax(A, 1e-6)), sigma_A);
  flagellates = rnorm(log1p(fmax(F, 1e-6)), sigma_F);
  rotifers = rnorm(log1p(fmax(R, 1e-6)), sigma_R);
")

dmeasure <- Csnippet("
  double sigma_A = 0.2;
  double sigma_F = 0.2;
  double sigma_R = 0.2;

  lik = dnorm(algae, log1p(fmax(A, 1e-6)), sigma_A, 1) +
      dnorm(flagellates, log1p(fmax(F, 1e-6)), sigma_F, 1) +
      dnorm(rotifers, log1p(fmax(R, 1e-6)), sigma_R, 1);


  if (!give_log) lik = exp(lik);
")


rinit <- Csnippet("
  S = S_0;
  A = A_0;
  R = R_0;
  F = F_0;
```

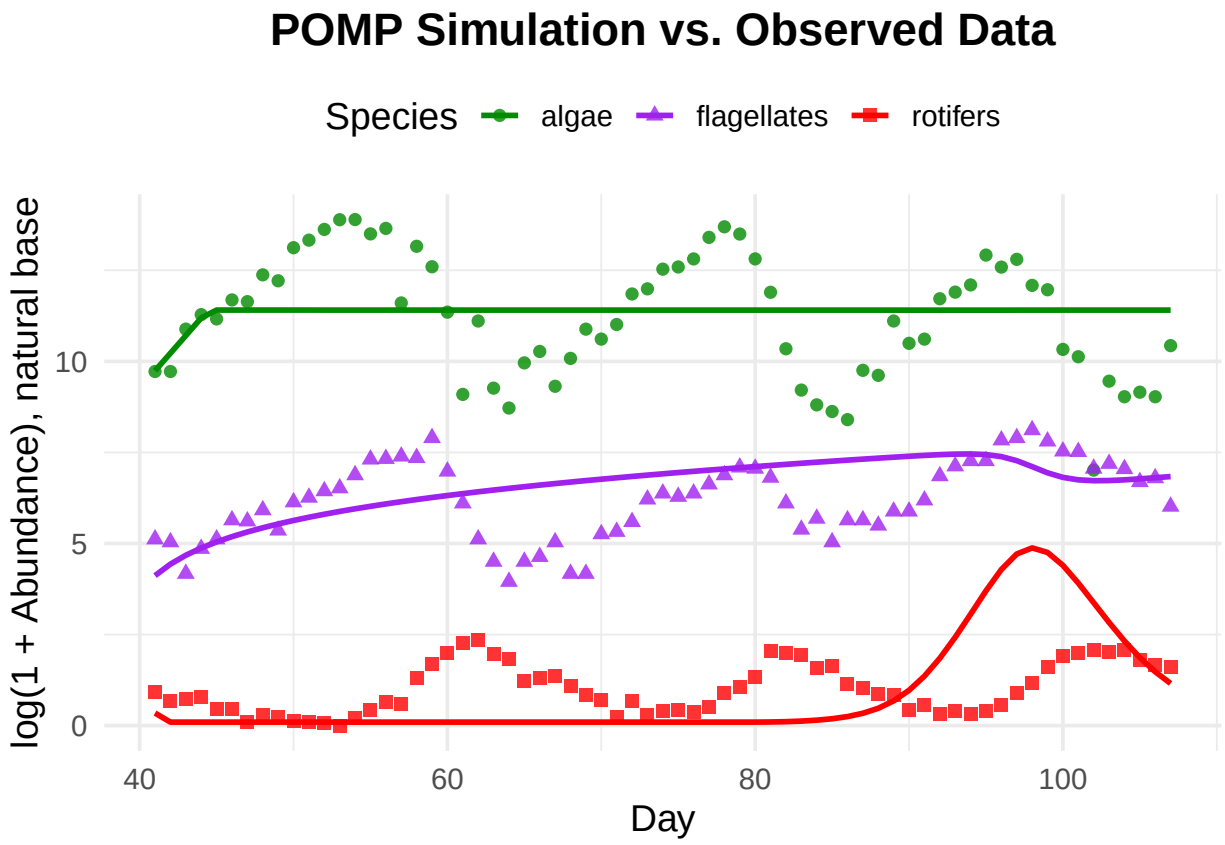
```
")

# Parameter transformation (log scale for positive parameters)
pt <- parameter_trans(
  log = c("delta", "kA", "kR", "r", "g", "h",
          "alphaA", "alphaF", "eta", "IF",
          "S_0", "A_0", "R_0", "F_0")
)

# Create POMP model
pomp_model <- pomp(
  data = orange_data,
  times = "day",
  t0 = 40,
  rprocess = rproc,
  rmeasure = rmeasure,
  dmeasure = dmeasure,
  rinit = rinit,
  statenames = statenames,
  paramnames = paramnames,
  obsnames = c("algae", "flagellates", "rotifers"),
  partrans = pt
)
```

```
# Get initial conditions from day 40
init_row <- X123[X123$day == 40, ]

params <- c(

  # Higher washout rate creates instability
  delta = 2,          # Increase from 0.05 to 0.15

  # Algae growth - needs to be fast enough to recover
  kA = 0.05,             # Lower threshold for faster growth
  r = 8,               # Higher growth rate

  # Rotifer predation - needs to be strong enough to create crashes
  kR = 200,              # Lower threshold (was 1000)
  g = 0.08,               # Higher predation rate (was 0.05)

  # Flagellate predation - moderate strength
  h = 3,               # Higher predation rate (was 0.02)
  alphaA = 1.49,          # Higher handling time creates delays
  alphaF = 10,          # Competition coefficient
  eta = 170,             # Interaction strength

  # Lower external input - let dynamics drive the system
  IF = 22,               # Reduce from 50 to 10

  # Initial conditions
  S_0 = 80,                          # Higher substrate
  A_0 = init_row$Algae.orange1,
  R_0 = init_row$Rot.orange1,
```

```
  F_0 = init_row$Flag.orange1
)

# Simulate the model
sim <- simulate(pomp_model, params = params, nsim = 1)
sim_df <- as.data.frame(sim, include.data = FALSE)
```

```
# Prepare observed data
obs_long <- orange_data %>%
  pivot_longer(cols = c(algae, flagellates, rotifers),
               names_to = "species",
               values_to = "log_abundance") %>%
  mutate(type = "Observed")

# Prepare simulation data (log1p to match observed)
sim_long <- sim_df %>%
  mutate(
    A = log1p(A),
    R = log1p(R),
    F = log1p(F)
  ) %>%
  pivot_longer(cols = c(A, R, F),
               names_to = "species",
               values_to = "log_abundance") %>%
  mutate(
    species = recode(species,
                     A = "algae",
                     R = "rotifers",
                     F = "flagellates"),
    type = "Simulated"
  )


# Combine both
combined_data <- bind_rows(obs_long, sim_long)

# Unified plot
ggplot(combined_data, aes(x = day, y = log_abundance, color = species)) +
  geom_point(data = filter(combined_data, type == "Observed"),
             aes(shape = species), size = 2, alpha = 0.8) +
  geom_line(data = filter(combined_data, type == "Simulated"),
            linewidth = 1) +
  scale_color_manual(values = c(
    algae = "green4",
    flagellates = "purple",
    rotifers = "red"
  )) +
  scale_shape_manual(values = c(
    algae = 16, flagellates = 17, rotifers = 15
  )) +
  labs(
    title = "POMP Simulation vs. Observed Data",
    x = "Day",
```

```
    y = "log(1 + Abundance), natural base",
    color = "Species",
    shape = "Species"
  ) +
  theme_minimal(base_size = 14) +
  theme(
    legend.position = "top",
    plot.title = element_text(hjust = 0.5, face = "bold")
  )
```




```
library(future)
library(doParallel)
```

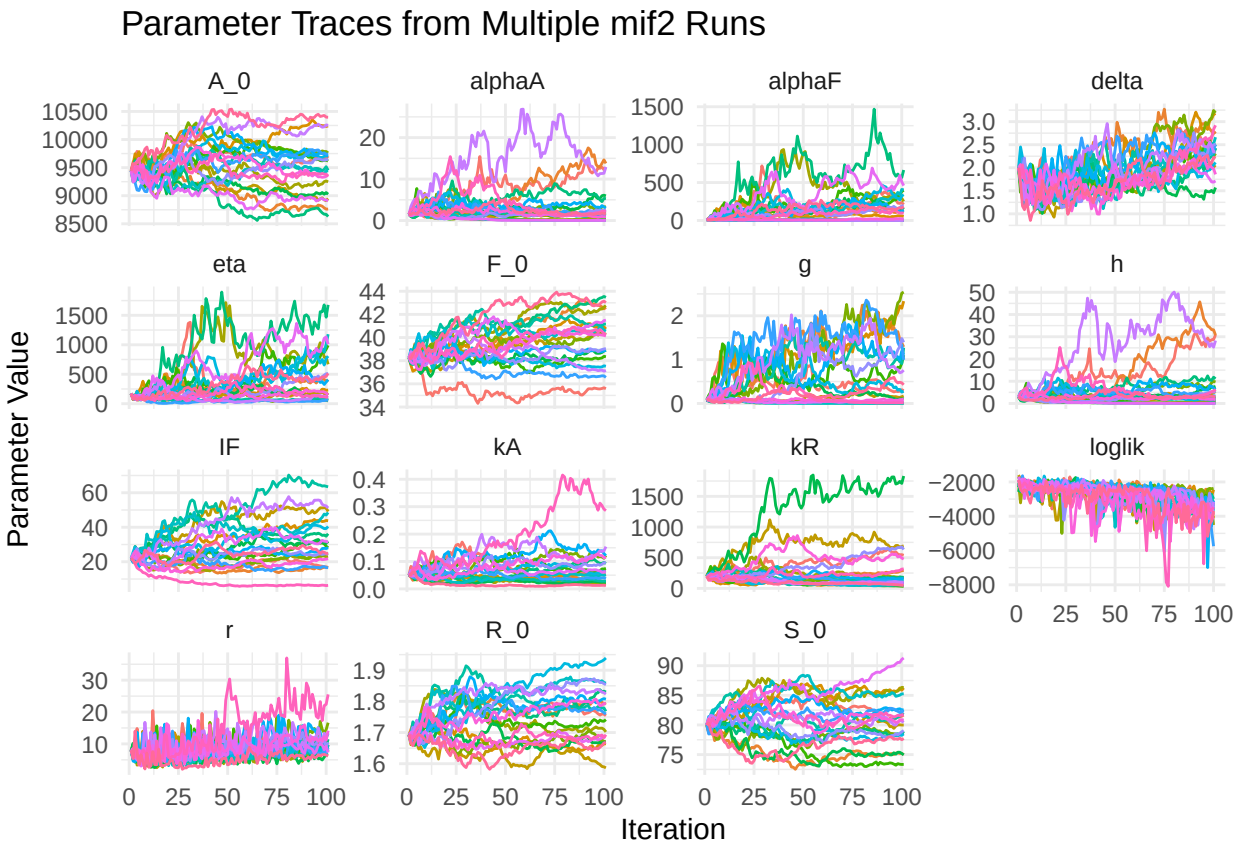
```
## Loading required package: iterators
```

```
## Loading required package: parallel
```

```
library(foreach)
plan(multisession)

# Step 1: initial guess
init_params <- params

# Step 2: define random walk standard deviations for each parameter
```

```r
rw_sd_values <- rw_sd(
  delta    = 0.02,
  kA       = 0.02,
  kR       = 0.02,
  r        = 0.05,
  g        = 0.05,
  h        = 0.05,
  alphaA   = 0.05,
  alphaF   = 0.05,
  eta      = 0.05,
  IF       = 0.01,
  S_0      = ivp(0.01),
  A_0      = ivp(0.01),
  R_0      = ivp(0.01),
  F_0      = ivp(0.01)
)

cores <-  as.numeric(Sys.getenv('SLURM_NTASKS_PER_NODE', unset=NA))
if(is.na(cores)) cores <- detectCores()
registerDoParallel(cores)
ggplot2::theme_set(ggplot2::theme_bw())

stopifnot(packageVersion("pomp")>="5.0")

# Perform multiple mif2 fits in parallel to obtain multiple parameter estimates
mifs_local <- foreach(
  i = 1:20,
  .combine = c,
  .options.future = list(seed = 89898975),
  .packages = "pomp"
) %dopar% {
  mif2(
    pomp_model,
    params = init_params,
    Np = 5000,              # Number of particles
    Nmif = 100,              # Number of mif iterations
    cooling.fraction.50 = 0.5,  # Cooling fraction
    rw.sd = rw_sd_values    # Random walk standard deviations
  )
}

#parameter traces
mifs_local |>
  traces() |>
  melt() |>
  ggplot(aes(x = iteration, y = value, group = .L1, color = factor(.L1))) +
  geom_line() +
  guides(color = "none") +
  facet_wrap(~ name, scales = "free_y") +
  theme_minimal() +
  labs(
    title = "Parameter Traces from Multiple mif2 Runs",
    x = "Iteration",
```

```
    y = "Parameter Value"
  )
```




```
local_search <- foreach(
  mf = mifs_local,
  .combine = rbind
) %dopar% {
  evals <- replicate(50, logLik(pfilter(mf, Np = 5000)))
  ll <- logmeanexp(evals, se = TRUE)
  mf %>% coef() %>% bind_rows() %>%
    bind_cols(loglik = ll[1], loglik.se = ll[2])
}

bind_rows(local_search) %>%
  filter(is.finite(loglik)) %>%
  filter(loglik.se < .5) %>%
  arrange(-loglik) -> best_searches

head(best_searches)
```

```
## # A tibble: 6 x 16
##   delta     kA     r    kR       g       h  alphaA  alphaF    eta     IF   S_0
##   <dbl>  <dbl> <dbl> <dbl>   <dbl>   <dbl>   <dbl>   <dbl>  <dbl> <dbl> <dbl>
## 1  2.01 0.0243  8.20 693.   0.170    1.21    0.622    1.08   147.   49.1  80.0
## 2  2.34 0.154   13.4   98.1 0.759   28.3    12.9      0.410  106.   51.2  80.4
```

```
## 3  1.71 0.0632  7.28  77.2 0.0561   0.0382  0.0250 460.    1010.   31.2  91.3
## 4  1.86 0.116  14.1  478.  0.104    1.69    0.979   18.0    150.   21.8  80.8
## 5  2.66 0.0510  9.71 125.  0.00639  0.0949  0.0416 403.    1180.   48.6  85.3
## 6  2.43 0.0282  7.49  84.4 1.27     2.33    1.22     7.04    64.6  63.4  81.7
## # i 5 more variables: A_0 <dbl>, R_0 <dbl>, F_0 <dbl>, loglik <dbl>,
## #   loglik.se <dbl>
```

```
summary(best_searches$loglik)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -106984   -7881   -6353  -11693   -5609   -3998
```

```
library(doFuture)
plan(multisession)

paramnames <- c(
  "delta", "kA", "kR", "r", "g", "h",
  "alphaA", "alphaF", "eta", "IF",
  "S_0", "A_0", "R_0", "F_0"
)

# Step 1: define random parameter ranges (log scale if applicable)
param_range <- runif_design(
  lower = c(
    delta = 0.01, kA = 0.01, kR = 10, r = 0.1, g = 0.01, h = 0.1,
    alphaA = 0.5, alphaF = 1, eta = 1, IF = 5,
    S_0 = 20, A_0 = 10, R_0 = 0.1, F_0 = 10
  ),
  upper = c(
    delta = 2, kA = 1, kR = 500, r = 10, g = 1, h = 10,
    alphaA = 5, alphaF = 10, eta = 200, IF = 50,
    S_0 = 120, A_0 = 100, R_0 = 1000, F_0 = 100
  ),
  nseq = 100
)

# Step 2: choose a starting mif2 object from local search
mf_start <- mifs_local[[1]]

# Step 3: run mif2 for each random parameter set
bake(file = "mif2_global.rds", seed = 888888, {
  foreach(p = iter(param_range, "row"), .combine = c,
          .options.future = list(seed = TRUE)) %dofuture% {
    mif2(
      mf_start,
      Nmif = 200,
      params = p,
      Np = 1000,
      cooling.fraction.50 = 0.5,
      rw.sd = rw_sd_values
    )
  }
}) -> mif2_results
```

```
# Step 4: evaluate loglikelihood of each fitted model
bake(file = "pfilter_ll.rds", seed = 999999, {
  foreach(m = mif2_results, .combine = rbind,
          .options.future = list(seed = TRUE)) %dofuture% {
    replicate(20, logLik(pfilter(m, Np = 5000))) |>
      logmeanexp(se = TRUE)
  }
}) -> loglik_results

# Step 5: organize into table
coef(mif2_results) |>
  melt() |>
  tidyr::pivot_wider(names_from = name, values_from = value) |>
  bind_cols(loglik = loglik_results[, 1], loglik.se = loglik_results[, 2]) |>
  arrange(-loglik) -> mif2_global_summary

# Step 6: inspect
summary(mif2_global_summary$loglik)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -72926  -69141   -4493  -27427   -4083   -3983
```

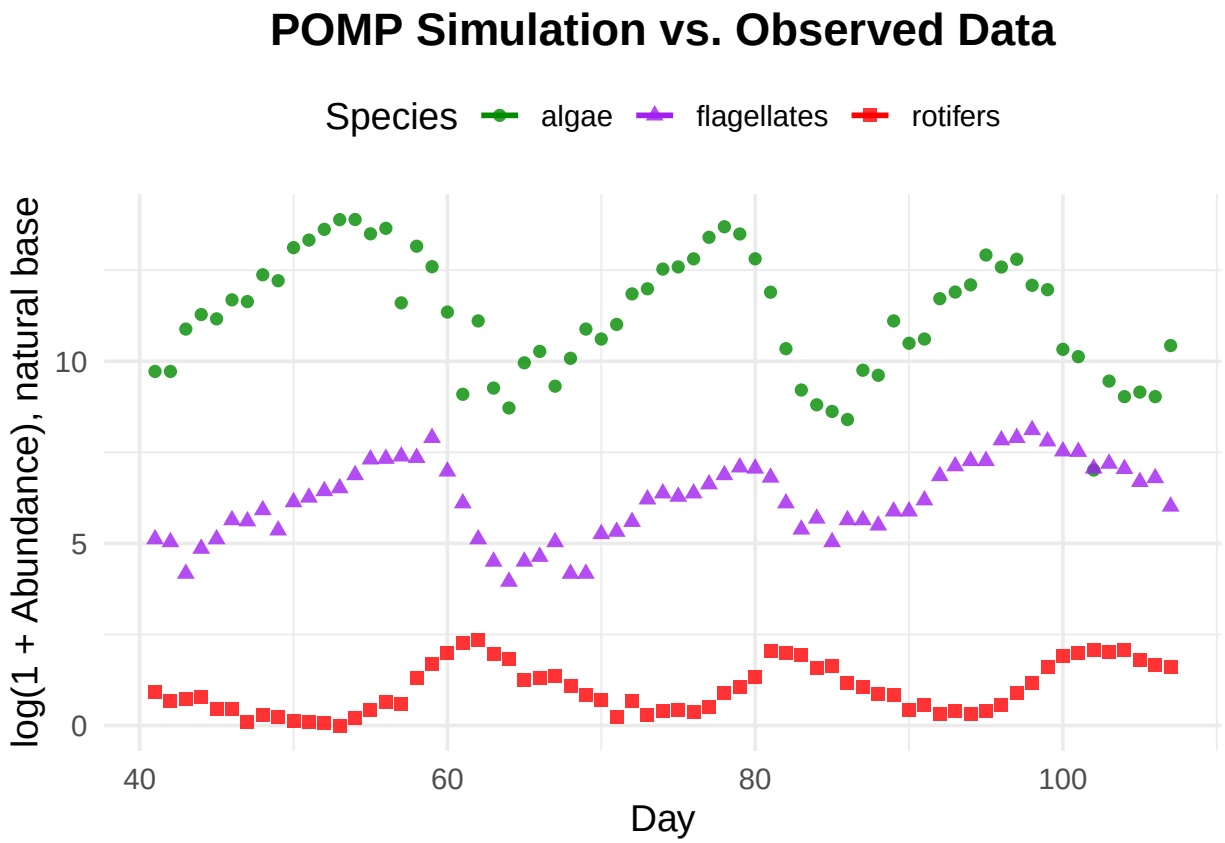
```
head(mif2_global_summary)
```

```
## # A tibble: 6 x 17
##     .id delta    kA     kR     r       g     h alphaA  alphaF    eta    IF   S_0
##   <int> <dbl> <dbl>  <dbl> <dbl>   <dbl> <dbl>  <dbl>   <dbl>  <dbl> <dbl> <dbl>
## 1    87  3.65 0.208   48.8  58.0 0.0234  6.62  1.85   363.     8.61   45.2  21.3
## 2    48  3.83 0.533 1112.   82.0 0.00212 0.527 0.141  137.     1.15   51.1  53.7
## 3    58  4.47 1.19   310.  154.  0.00309 3.83  0.884    3.06   0.657  78.9  32.5
## 4    77  4.62 0.272   69.4  65.3 0.0227  0.397 0.0900 200.    12.4   114.   29.8
## 5    32  4.95 1.38   412.  201.  0.0468  0.642 0.135    9.19   9.76  120.   22.4
## 6    68  4.32 0.942 1598.  154.  0.0731  5.94  1.44     0.823  1.57  100.   27.6
## # i 5 more variables: A_0 <dbl>, R_0 <dbl>, F_0 <dbl>, loglik <dbl>,
## #   loglik.se <dbl>
```

```
best_searches <- mif2_global_summary %>%
  filter(is.finite(loglik)) %>%
  filter(loglik.se < .5) %>%
  arrange(-loglik)


param_names <- paramnames

params <- best_searches[1, param_names] |> unlist()

simulated_data <- simulate(
  pomp_model,
  params = unlist(best_searches[1,]),
  nsim = 1,
  format = "data.frame",
  include.data = TRUE
```

```
)

sim_df <- as.data.frame(simulated_data, include.data = FALSE)

sim_long <- sim_df %>%
  mutate(
    A = log1p(A),
    R = log1p(R),
    F = log1p(F)
  ) %>%
  pivot_longer(cols = c(A, R, F),
               names_to = "species",
               values_to = "log_abundance") %>%
  mutate(
    species = recode(species,
                     A = "algae",
                     R = "rotifers",
                     F = "flagellates"),
    type = "Simulated"
  )


# Combine both
combined_data <- bind_rows(obs_long, sim_long)

ggplot(combined_data, aes(x = day, y = log_abundance, color = species)) +
  geom_point(data = filter(combined_data, type == "Observed"),
             aes(shape = species), size = 2, alpha = 0.8) +
  geom_line(data = filter(combined_data, type == "Simulated"),
            linewidth = 1) +
  scale_color_manual(values = c(
    algae = "green4",
    flagellates = "purple",
    rotifers = "red"
  )) +
  scale_shape_manual(values = c(
    algae = 16, flagellates = 17, rotifers = 15
  )) +
  labs(
    title = "POMP Simulation vs. Observed Data",
    x = "Day",
    y = "log(1 + Abundance), natural base",
    color = "Species",
    shape = "Species"
  ) +
  theme_minimal(base_size = 14) +
  theme(
    legend.position = "top",
    plot.title = element_text(hjust = 0.5, face = "bold")
  )
```

```
## Warning: Removed 3 rows containing missing values or values outside the scale range
## (`geom_line()`).
```


POMP Simulation vs. Observed Data
Species
algae
flagellates
rotifers
log(1 + Abundance), natural base
0
5
10
40
60
80
100
Day