Prior SBC on the Lotka-Volterra model

Teemu Säilynoja

2024-06-20

In this notebook, we show the results of calibration assessment on the Lotka-Volterra model using prior SBC. For collected results of all our experiments, see the notebook named lotka-volterra-sbc.

1 Setup

Like the original Hudson Bay data, we simulate data sets of 21 years. We run the SBC on 250 prior predictive samples, each with a total of 4000 post warm-up posterior samples, what will be thinned down by a factor of 10, yielding 400 posterior draws for calculating the rank of the prior draws and joint log-likelihood.

2 The Model

```
functions {
  vector dz_dt(real t,       // time
               vector z,     // system state {prey, predator}
               array[] real theta // parameters
               //real[] x_r,   // unused data
               //int[] x_i
               ) {
    real u = z[1];
    real v = z[2];

    real alpha = theta[1];
    real beta = theta[2];
    real gamma = theta[3];
    real delta = theta[4];
```

```
    real du_dt = (alpha - beta * v) * u;
    real dv_dt = (-gamma + delta * u) * v;

    return to_vector({du_dt, dv_dt});
  }
}

data {
  int<lower = 0> N;          // number of measurement times exl. year 0
  array[N] real ts;                // measurement times > 0
  array[2] real y_init;            // initial measured populations
  array[N,2] real <lower = 0> y;   // measured populations
}
parameters {
  array[4] real<lower = 0> theta;   // { alpha, beta, gamma, delta }
  vector <lower = 0> [2] z_init;  // initial population
  array[2] real<lower = 0> sigma;   // measurement errors
}
transformed parameters {
  array[N] vector[2] z
    = ode_rk45_tol(dz_dt, z_init, 1.0, ts, 1e-6, 1e-5, 1000, theta);
}
model {
  theta[{1, 3}] ~ normal(1, 0.5);
  theta[{2, 4}] ~ normal(0.05, 0.05);
  sigma ~ lognormal(-1, 1);
  z_init ~ lognormal(log(10), 1);
  for (k in 1:2) {
    y_init[k] ~ lognormal(log(z_init[k]), sigma[k]);
    y[ , k] ~ lognormal(log(z[, k]), sigma[k]);
  }
}

generated quantities {
  real loglik = 0;
  vector[N + 1] log_lik;
  log_lik[1] = 0;
  for (k in 1:2) {
    log_lik[1] += lognormal_lpdf(y_init[k]|log(z_init[k]), sigma[k]);
    loglik += lognormal_lpdf(y_init[k]|log(z_init[k]), sigma[k]);
    loglik += lognormal_lpdf(y[ , k]|log(z[, k]), sigma[k]);
```

```
    }
  for (n in 1:N) {
    log_lik[n + 1] = 0;
    for (k in 1:2) {
      log_lik[n + 1] += lognormal_lpdf(y[n, k]|log(z[n, k]), sigma[k]);
      }
      }
}
```

3 Simulation-based calibration

3.1 Prior predictive samples

For implementing the SBC, we use the SBC R-package by Angie H. Moon et al. First, we define the data generating process, which in our case is generating prior predictive samples.

Next, we use the data generating process to define a data generator and construct 250 data sets for SBC.

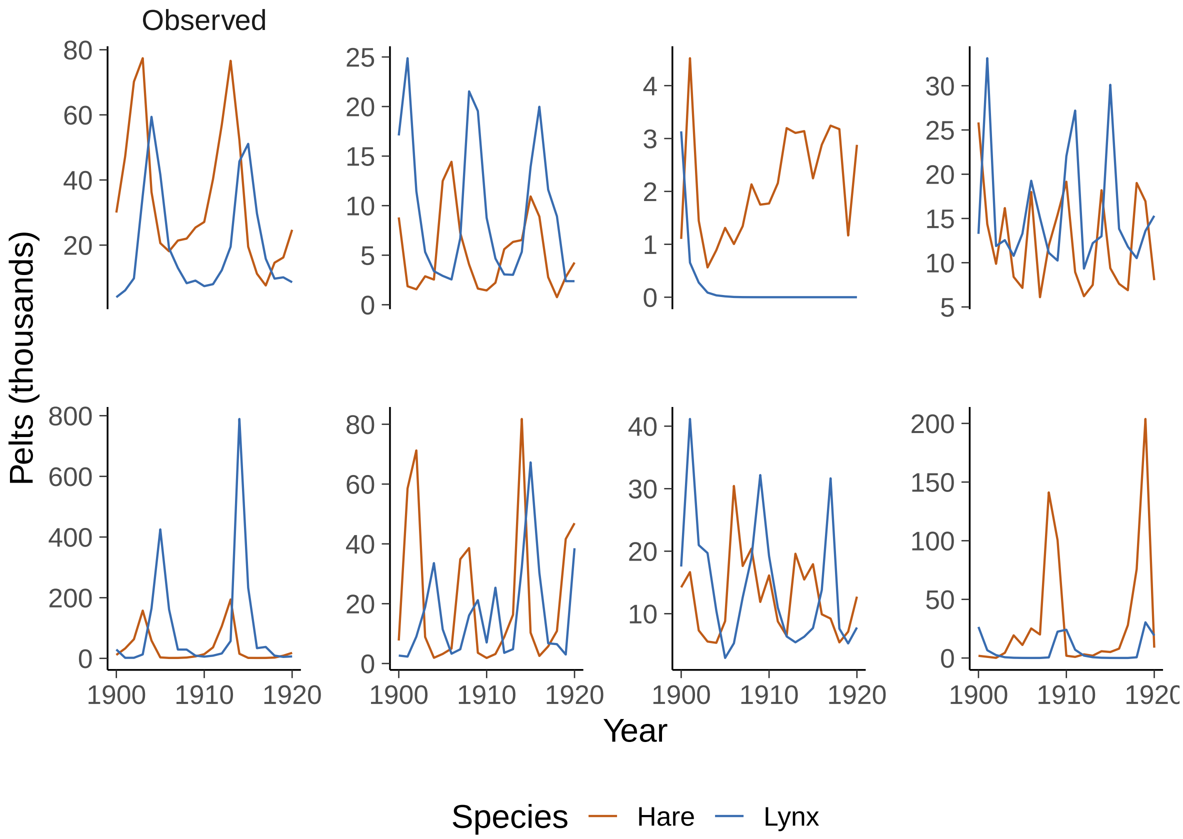


Figure 1: Observed data compared to prior predictive draws.

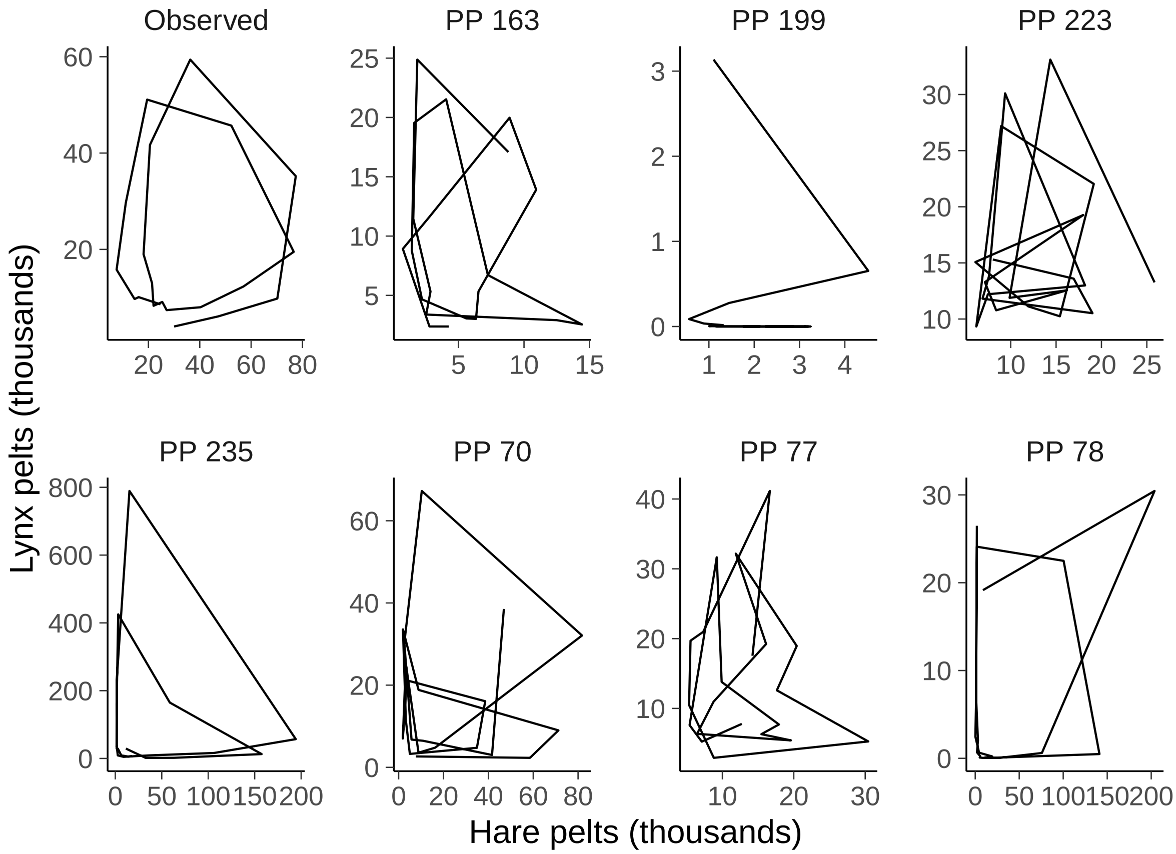


Figure 2: Trajectories of the population dynamics in the observation and the prior predictive draws from above.

3.2 Posterior samples

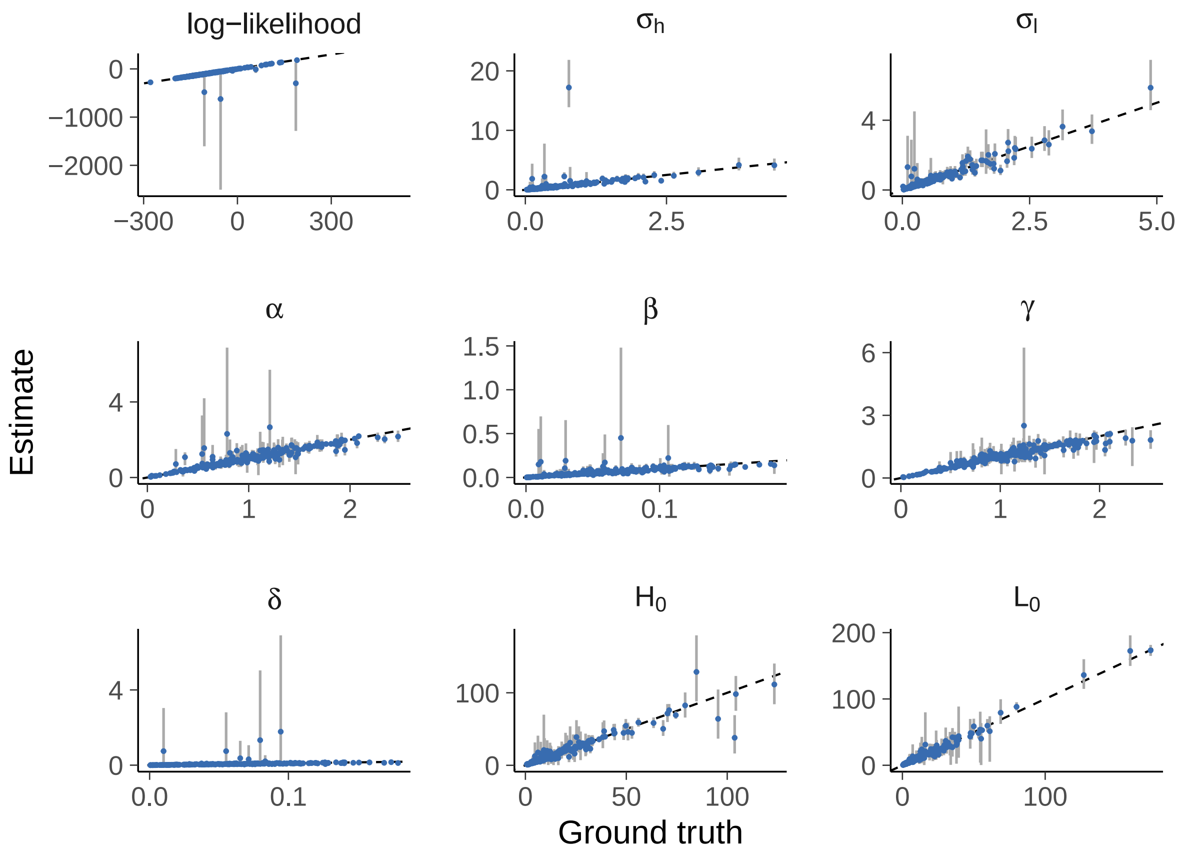
We use `cmdstanr` to obtain posterior draws via MCMC sampling.

3.3 Run SBC

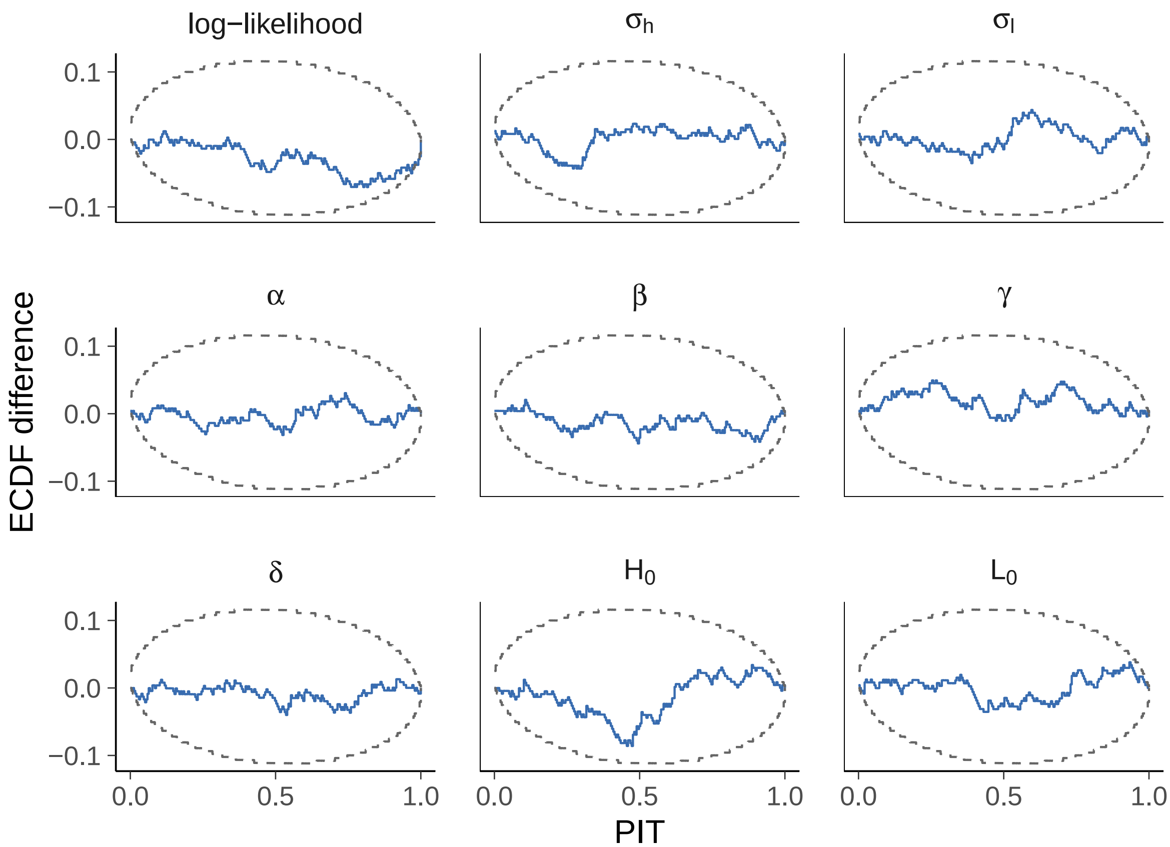
This is the step requiring a lot of computation. We run MCMC on each of the predictive samples and store statistics of the results.

3.4 Results

A simple plot to compare the posterior samples and the ground truth values.



The ECDF of the PIT values of the prior draw with regards to the posterior sample gives a principled way to check for the calibration of the model, as we can draw simultaneous 95% confidence intervals for the PIT-ECDFs. The joint log-likelihood is a good test quantity for overall calibration of the inference, and shows here some possible calibration issues, although the number of SBC iterations is quite low.



log−likelihood
σh
σl
α
β
γ
δ
H0
L0
0.1
0.0
−0.1
ECDF difference
0.0
0.5
1.0
PIT