Lesson 4: Particle Filtering in POMP

Spencer J. Fox Qianying (Ruby) Lin Jesse Wheeler

Particle filtering in pomp I

Recall the measles-outbreak example and the stochastic SIR model that we construct in the previous lesson, we can using the `pfilter` function to compute the likelihood using particle filtering method, given the parameters chosen by looking at simulations. `R` code to build the model is available here. We can execute this code by sourcing the file and check the parameters:

```
source("scripts/model_measSIR.R")
measSIR@params
```

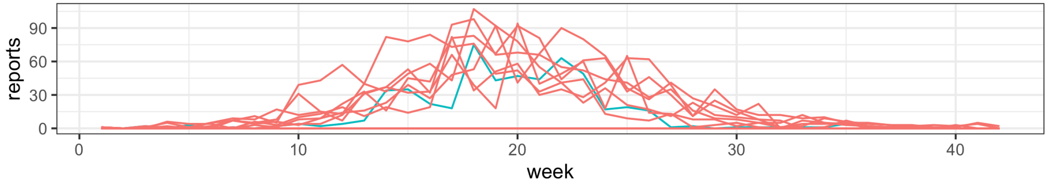
```
   Beta   Gamma     Rho       k     Eta       N 
1.5e+01 5.0e-01 5.0e-01 1.0e+01 6.0e-02 3.8e+04 
```

Particle filtering in pomp II

The particle filtering method relies heavily on the state process and the measurement model. Therefore, it is necessary to make sure that the basic particle filter is working.

1. Check the `rprocess` and the `rmeasure` by simulation, as shown in Lesson 2:

```
measSIR |>
  simulate(nsim=20,format="data.frame",include.data=TRUE) |>
  ggplot(aes(x=week,y=reports,group=.id,color=.id=="data")) +
  geom_line() + guides(color="none")
```



Particle filtering in pomp III

In pomp, we can compute the likelihood using the particle filtering method, implemented by function `pfilter`. The argument `Np` assigns the number of particles used:

```
library(pomp)
pf <- measSIR |> pfilter(Np=5000)
logLik(pf)
```

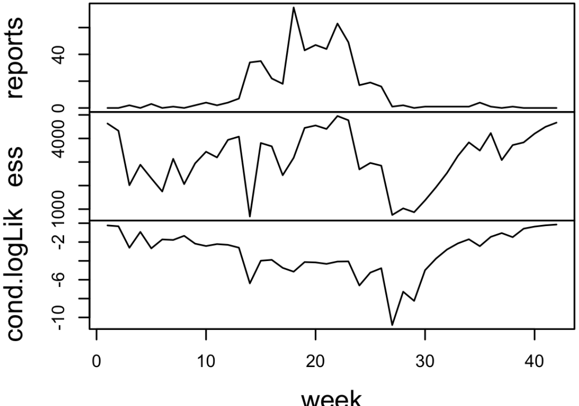
```
[1] -134.2366
```

Particle filtering in pomp IV

2. A diagnostic plot to check the `rprocess` and the `dmeasure`:

```
plot(pf)
```

- The data, `reports`;
- The **effective sample size** of the particle filter, `ess`;
- The log-likelihood of each observation conditioned on the preceding ones, `cond.logLik`.



Particle filtering in pomp V

3. The Monte Carlo variability of the likelihood:

- The `pfilter` function provides a single Monte-Carlo approximation of the log-likelihood for a fixed parameter value. That is, there is some randomness in the parameter estimate.
- Like all Monte Carlo evaluations, it makes sense to obtain multiple estimates to check the variance of the Monte Carlo algorithm.
- Because each call to `pfilter` is an independent Monte Carlo, we can leverage parallel computing to redo the evaluation multiple times.

Particle filtering in pomp VI

```r
cores <- parallel::detectCores()
registerDoParallel(cores-1)  # MacOS / Linux
registerDoRNG(seed = 123456)
# cl <- makePSOCKcluster(cores-1)  # Windows
# registerDoParallel(cl)  # Windows
foreach (
  i=1:10, .combine=c
) %dopar% {
    measSIR |> pfilter(Np=5000)
} -> pf
# stopCluster(cl)  # Windows
logLik(pf) -> ll
logmeanexp(ll,se=TRUE)
```

Particle filtering in pomp VII

```
         est           se
-130.6718080    0.3657197
```

Note that registerDoRNG sets random seed for parallel computing, converting `%dopar%` loops into `%doRNG%` loops.

`pfilter` in `pomp`: Summary

What have we done?

- After building a model, giving it parameters, we can approximate the log-likelihood using `pfilter`.
- There are tools to check things are working properly. For instance, `plot` simulations to check `rprocess` and `rmeas` components are working.
- Plotting `pfiltered` objects helps us check how good the Monte-Carlo approximation is.
- We can use parallel computing to do multiple `pfilter` evaluations, and combine results to get a single estimate.

We have so far used fixed parameter values. What about estimation?

Review of likelihood-based inference

- Likelihood-based inference (meaning statistical tools based on the likelihood function) provides tools for parameter estimation, standard errors, hypothesis tests and diagnosing model misspecification.
- Likelihood-based inference often (but not always) has favorable theoretical properties. Here, we are not especially concerned with the underlying theory of likelihood-based inference. On any practical problem, we can check the properties of a statistical procedure by simulation experiments.

The maximum likelihood estimate (MLE)

- A maximum likelihood estimate (MLE) is

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}}\, \ell(\theta),$$

  where $\operatorname{argmax}_\theta g(\theta)$ means a value of argument θ at which the maximum of the function g is attained, so $g\left(\operatorname{argmax}_\theta g(\theta)\right) = \max_\theta g(\theta)$.
- If there are many values of θ giving the same maximum value of the likelihood, then an MLE still exists but is not unique.
- Note that $\operatorname{argmax}_\theta \mathcal{L}(\theta)$ and $\operatorname{argmax}_\theta \ell(\theta)$ are the same. Why?

Naive Likelihood Maximizaiton

In other areas of statistics and machine learning, if we have some type of *objective function* that we want to maximize (or minimize), we can use numeric solvers to find the maximum (or minimum). For instance, the Nelder-Mead, BFGS, or other gradient descent algorithms (using a function like `stats::optim`.
This famously doesn't work well with the particle filter..
Some problems:

- Evaluation is very slow.
- Evaluation is stochastic.
- Likelihood surface is multi-modal (many local maximums).

We will explore this idea by looking at *slices* of the likelihood surface.

The likelihood surface

- It is extremely useful to visualize the geometric surface defined by the likelihood function.
- If Θ is two-dimensional, then the surface $\ell(\theta)$ has features like a landscape.
- Local maxima of $\ell(\theta)$ are peaks.
- Local minima are valleys.
- Peaks may be separated by a valley or may be joined by a ridge. If you go along the ridge, you may be able to go from one peak to the other without losing much elevation. Narrow ridges can be easy to fall off, and hard to get back on to.
- In higher dimensions, one can still think of peaks and valleys and ridges. However, as the dimension increases it quickly becomes hard to imagine the surface.

Exploring the likelihood surface: slices

- To get an idea of what the likelihood surface looks like in the neighborhood of a point in parameter space, we can construct some likelihood *slices*.
- A likelihood slice is a cross-section through the likelihood surface.
- We'll make slices for our Consett measles POMP model, in the β and μ_{IR} directions.
- Both slices will pass through our current candidate parameter vector, stored in the `pomp` model object.

Slicing the measles SIR likelihood I

- We first construct a data frame to explore the parameter slice, with $40 \times 3 + 40 \times 3 = 240$ rows:

```r
slice_design(
  center = coef(measSIR),
  Beta = rep(seq(from=5,to=30,length=40),each=3),
  Gamma = rep(seq(from=0.2,to=2,length=40),each=3)
) -> param_slice

dim(param_slice)
```

```
[1] 240   7
```

Slicing the measles SIR likelihood II

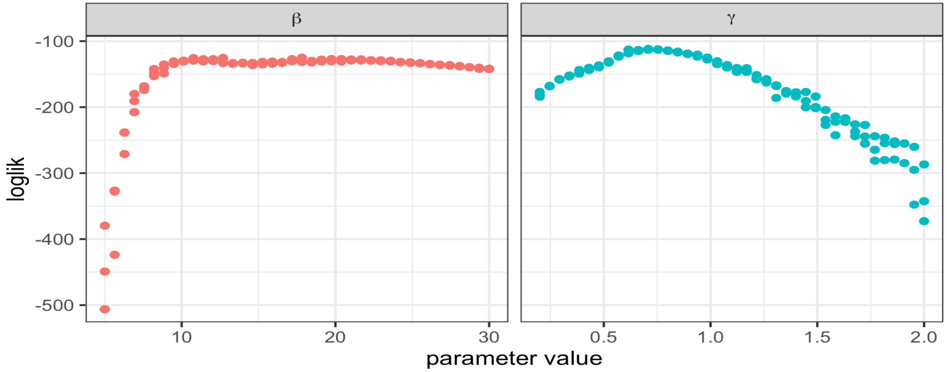
- We compute the likelihoods 3 times for each combination (i.e., row) in `param_slice`:

Slicing the measles SIR likelihood III

```
library(iterators)
# Doesn't need to happen again, but included as reminder.
registerDoParallel(cores-1)  # For MacOS / Linux
registerDoRNG(seed = 654321)

# cl <- makePSOCKcluster(cores-1)  # Windows
# registerDoParallel(cl)  # Windows
foreach (theta=iter(param_slice,"row"), .combine=rbind) %dopar% {
  measSIR |> pfilter(params=theta,Np=5000) -> pf
  theta$loglik <- logLik(pf)
  theta
} -> lik_slice
# stopCluster(cl)  # Windows
```

Slicing the measles SIR likelihood IV



- Slices offer a very limited perspective on the geometry of the likelihood surface.
- When there are only one or two unknown parameters, we can evaluate the likelihood at a grid of points and visualize the surface directly.

Two-dimensional likelihood slice I

- We first construct the parameters grid data frame `param_grid`, with $40 \times 3 \times 40 \times 3 = 14,400$ rows:

```
expand.grid(
  Beta = rep(seq(from=10,to=30,length=40), each=3),
  Gamma = rep(seq(from=0.4,to=1.5,length=40), each=3),
  Rho = 0.5, k=10, Eta=0.06, N=38000
) -> param_grid

dim(param_grid)
```

```
[1] 14400     6
```

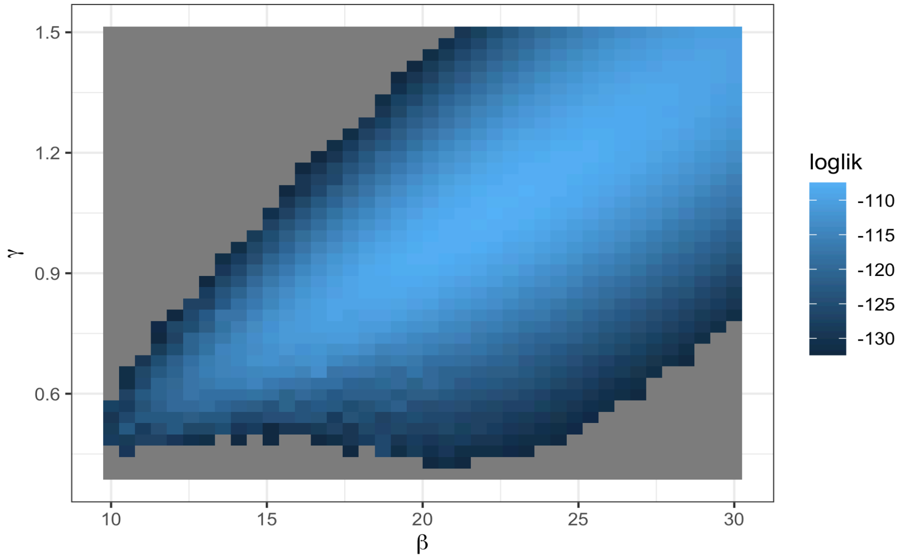
Two-dimensional likelihood slice II

- We then compute likelihoods for each of the combinations:

```
# Doesn't need to happen again, but included as reminder.
registerDoParallel(cores-1)  # For MacOS / Linux
registerDoRNG(seed = 111111)

# cl <- makePSOCKcluster(cores-1)  # Windows
# registerDoParallel(cl)  # Windows
foreach (theta=iter(param_grid,"row"), .combine=rbind) %dopar% {
  measSIR |> pfilter(params=theta,Np=5000) -> pf
  theta$loglik <- logLik(pf)
  theta
} -> lik_grid
# stopCluster(cl)  # Windows
```

Two-dimensional likelihood slice: Figure



Two-dimensional likelihood: Discussion

In the above, all points with log-likelihoods less than 25 units below the maximum are shown in grey.

- Notice some features of the log-likelihood surface, and its estimate from the particle filter, that can cause difficulties for numerical methods:
    1. The surface is wedge-shaped, so its curvature varies considerably. By contrast, asymptotic theory predicts a parabolic surface that has constant curvature.
    2. Monte Carlo noise in the likelihood evaluation makes it hard to pick out exactly where the likelihood is maximized. Nevertheless, the major features of the likelihood surface are evident despite the noise.
- Wedge-shaped relationships between parameters, and nonlinear relationships, are common features of epidemiological dynamic models.

Cost of a particle-filter calculation

- How much computer processing time does a particle filter take?
- How does this scale with the number of particles?

Form a conjecture based upon your understanding of the algorithm. Test your conjecture by running a sequence of particle filter operations, with increasing numbers of particles (`Np`), measuring the time taken for each one using `system.time`. Plot and interpret your results.

Worked solution to the Exercise

log-likelihood estimation I

Here are some desiderata for a Monte Carlo log-likelihood approximation:

- It should have low Monte Carlo bias and variance.
- It should be presented together with estimates of the bias and variance so that we know the extent of Monte Carlo uncertainty in our results.
- It should be computed in a length of time appropriate for the circumstances.

Set up a likelihood evaluation for the measles model, choosing the numbers of particles and replications so that your evaluation takes approximately one minute on your machine.

- Provide a Monte Carlo standard error for your estimate.
- Comment on the bias of your estimate.
- Use `foreach` and `doParallel` to take advantage of multiple cores on your computer to improve your estimate.

Exercises

1. **One-dimensional likelihood slice**: Compute several likelihood slices in the η direction.
2. **Two-dimensional likelihood slice**: Compute a slice of the likelihood in the β-η plane.

Maximizing the particle filter likelihood

- Likelihood maximization is key to profile intervals, likelihood ratio tests and AIC as well as the computation of the MLE.
- An initial approach to likelihood maximization might be to stick the particle filter log-likelihood estimate into a standard numerical optimizer, such as the Nelder-Mead algorithm.
- In practice this approach is unsatisfactory on all but the smallest POMP models. Standard numerical optimizers are not designed to maximize noisy and computationally expensive Monte Carlo functions.
- Further investigation into this approach is available as a supplement.
- We'll present an *iterated filtering algorithm* for maximizing the likelihood in a way that takes advantage of the structure of POMP models and the particle filter.
- First, let's think a bit about some practical considerations in interpreting the MLE for a POMP.

Likelihood-based model selection and model diagnostics

- For nested hypotheses, we can carry out model selection by likelihood ratio tests.
- For non-nested hypotheses, likelihoods can be compared using Akaike's information criterion (AIC) or related methods.

Likelihood ratio tests for nested hypotheses I

- The whole parameter space on which the model is defined is $\Theta \subset \mathbb{R}^D$.
- Suppose we have two **nested** hypotheses

$$H^{\langle 0 \rangle} : \theta \in \Theta^{\langle 0 \rangle},$$
$$H^{\langle 1 \rangle} : \theta \in \Theta^{\langle 1 \rangle},$$

  defined via two nested parameter subspaces, $\Theta^{\langle 0 \rangle} \subset \Theta^{\langle 1 \rangle}$, with respective dimensions $D^{\langle 0 \rangle} < D^{\langle 1 \rangle} \leq D$.
- We consider the log-likelihood maximized over each of the hypotheses,

$$\ell^{\langle 0 \rangle} = \sup_{\theta \in \Theta^{\langle 0 \rangle}} \ell(\theta),$$
$$\ell^{\langle 1 \rangle} = \sup_{\theta \in \Theta^{\langle 1 \rangle}} \ell(\theta).$$

Likelihood ratio tests for nested hypotheses II

- **Wilks approximation**: under the hypothesis $H^{\langle 0 \rangle}$,
$$\ell^{\langle 1 \rangle} - \ell^{\langle 0 \rangle} \approx \frac{1}{2}\,\chi^2_{D^{\langle 1 \rangle} - D^{\langle 0 \rangle}},$$
where χ^2_d is a chi-squared random variable on d degrees of freedom and $\approx$ means "is approximately distributed as".
- The Wilks approximation can be used to construct a hypothesis test of the null hypothesis $H^{\langle 0 \rangle}$ against the alternative $H^{\langle 1 \rangle}$.
- This is called a **likelihood ratio test** since a difference of log-likelihoods corresponds to a ratio of likelihoods.
- When the data are IID, $N \to \infty$, and the hypotheses satisfy suitable regularity conditions, this approximation can be derived mathematically and is known as **Wilks' theorem**.

Likelihood ratio tests for nested hypotheses III

- The chi-squared approximation to the likelihood ratio statistic may be useful, and can be assessed empirically by a simulation study, even in situations that do not formally satisfy any known theorem.

Wilks' theorem and profile likelihood I

- Suppose we have an MLE, written $\hat{\theta} = (\hat{\phi}, \hat{\psi})$, and a profile log-likelihood for ϕ, given by $\ell^{\text{profile}}(\phi)$.
- Consider the likelihood ratio test for the nested hypotheses

$$\begin{aligned} H^{\langle 0 \rangle} &: \phi = \phi_0, \\ H^{\langle 1 \rangle} &: \phi \text{ unconstrained.} \end{aligned}$$

- We can compute the 95%-ile for a chi-squared distribution with one degree of freedom: `qchisq(0.95,df=1)`$= 3.841$.
- Wilks' theorem then gives us a hypothesis test with approximate size 5% that rejects $H^{\langle 0 \rangle}$ if $\ell^{\text{profile}}(\hat{\phi}) - \ell^{\text{profile}}(\phi_0) < 3.84/2$.

Wilks' theorem and profile likelihood II

- It follows that, with probability 95%, the true value of ϕ falls in the set
$$\{\phi : \ell^{\text{profile}}(\hat{\phi}) - \ell^{\text{profile}}(\phi) < 1.92\}.$$
So, we have constructed a profile likelihood confidence interval, consisting of the set of points on the profile likelihood within 1.92 log units of the maximum.
- This is an example of a general duality between confidence intervals and hypothesis tests.

Akaike's information criterion (AIC) I

- Likelihood ratio tests provide an approach to model selection for nested hypotheses, but what do we do when models are not nested?
- A more general approach is to compare likelihoods of different models by penalizing the likelihood of each model by a measure of its complexity.
- Akaike's information criterion **AIC** is given by

$$\mathrm{AIC} = -2\,\ell(\hat{\theta}) + 2\,D$$

  "Minus twice the maximized log-likelihood plus twice the number of parameters."
- We are invited to select the model with the lowest AIC score.
- AIC was derived as an approach to minimizing prediction error. Increasing the number of parameters leads to additional **overfitting** which can decrease predictive skill of the fitted model.

Akaike's information criterion (AIC) II

- Viewed as a hypothesis test, AIC may have weak statistical properties. It can be a mistake to interpret AIC by making a claim that the favored model has been shown to provide a superior explanation of the data. However, viewed as a way to select a model with reasonable predictive skill from a range of possibilities, it is often useful.
- AIC does not penalize model complexity beyond the consequence of reduced predictive skill due to overfitting. One can penalize complexity by incorporating a more severe penalty than the $2D$ term above, such as via BIC.
- A practical approach is to use AIC, while taking care to view it as a procedure to select a reasonable predictive model and not as a formal hypothesis test.

License, acknowledgments, and links

- This lesson is prepared for the Simulation-based Inference for Epidemiological Dynamics module at the Summer Institute in Statistics and Modeling in Infectious Diseases, SISMID.
- The materials build on previous versions of this course and related courses.

- Produced with R version 4.5.1 and pomp version 6.3.
- Compiled on 2024-07-24.

Back to Lesson
R code for this lesson