Inferring biological dynamics 101

2. Sequential Monte Carlo (SMC)

- SMC arose in the 1990s, simultaneously called particle filtering, bootstrap filtering, Monte Carlo filtering, and the condensation algorithm.
- Used in physics and chemistry since the 1950s: "Poor man's Monte Carlo" [2, 9].
- In 2001, the research area was unified [1, 7].
- SMC provides an alternative to MCMC for many computations. For dynamic systems, SMC is preferred.
- SMC resembles natural selection. A "swarm" of "particles" evolves according to a stochastic dynamic system. Particles consistent with data at time t are propagated to $t + 1$.
- SMC guarantees unbiased likelihood estimates (more particles gives reduced variance).

POMP model notation

- The **state process model**, $X(t)$, is a Markov process observed at times $t_1 < t_2 < \cdots < t_N$.
- $X_n = X(t_n)$ is a discrete-time Markov process, specified by transition density $f(x_n \,|\, x_{n-1}, \theta)$ and initial distribution $f(x_0, \theta)$ at some time $t_0 < t_1$.
- The **observation model** is $Y_n \sim f(y_n \,|\, x_n, \theta)$.

We write $y_{1:k} = (y_1, \ldots, y_k)$ and let $f(\cdot \,|\, \cdot)$ denote an arbitrary density, specified by its arguments. Important examples are:

- **Prediction density.** $f(x_n \,|\, y_{1:n-1}, \theta)$
- **Filtering density.** $f(x_n \,|\, y_{1:n}, \theta)$
- **Smoothing density.** $f(x_n \,|\, y_{1:N}, \theta)$
- **Likelihood evaluation.** $f(y_n \,|\, y_{1:n-1}, \theta)$

Numerical solutions (approximations) for prediction, filtering, smoothing and likelihood evaluation enable (aproximately) full-information POMP inference.

POMP recursions

Prediction:

$$f(x_n|y_{1:n-1}) = \int f(x_n|x_{n-1})\, f(x_{n-1}|y_{1:n-1})\, dx_{n-1}.$$

Filtering:

$$f(x_n|y_{1:n}) = \frac{f(x_n|y_{1:n-1}) f(y_n|x_n)}{f(y_n|y_{1:n-1})}.$$

Smoothing:

$$f(x_n|y_{1:N}) \propto f(y_{n:N}|x_n)\, f(x_n|y_{1:n-1}).$$

Likelihood:

$$f(y_n|y_{1:n-1}) = \int f(y_n|x_n) f(x_n|y_{1:n-1})\, dx_n.$$

- For Gaussian models, the integrals have a closed form solution (the Kalman filter).
- If the states are discrete, the integrals are sums.
- In general, numerical integration or Monte Carlo is required.
- All densities can depend on θ, which is suppressed here. These recursions integrate out unobserved state variables, for a fixed model.

If you want to do the algebra...

- The POMP recursion identities follow in a couple of lines of algebra, if you set off in the right direction.
- The POMP conditional independence assumptions are

$$\begin{aligned} f(x_n \,|\, x_{1:n-1}, y_{1:n-1}) &= f(x_n \,|\, x_{n-1}), \\ f(y_n \,|\, x_{1:n}, y_{1:n-1}) &= f(y_n \,|\, x_n). \end{aligned}$$

- As tools, you need conditional forms of standard identities:

$$\begin{aligned} f(u \,|\, w) &= \int f(u, v \,|\, w)\, dv, \\ f(u, v \,|\, w) &= f(v \,|\, w)\, f(u \,|\, v, w). \end{aligned}$$

- Here, $f(u \,|\, w)$ is shorthand for $f_{U|M}(u \,|\, w)$. The generic use of f is abuse of notation: the choice of dummy variables u and w should not determine the definition of a function.

Monte Carlo POMP recursions
"Basic SMC" or "Vanilla particle filter"

- The filtering distribution $f(x_n|y_{1:n})$ is represented by a **swarm** of particles, $\{X^F_{n,j}, j = 1, \ldots, J\}$.

- Each particle is **updated** according to the stochastic dynamic model,

$$X^P_{n+1,j} \sim f(x_{n+1} \,|\, x_n = X^F_{n,j}, \theta).$$

  The updated swarm $\{X^P_{n+1,j}, j = 1, \ldots, J\}$ describes the prediction density, $f(x_{n+1}|y_{1:n})$. **Phenotypic variation** of the swarm is increased by stochasticity in the update.

- The prediction particles are **resampled** with weight $w_j = f(y_{n+1} \,|\, x_{n+1} = X^P_{n+1,j})$ to describe the filtering distribution $f(x_{n+1}|y_{1:n+1})$. This **natural selection** reduces phenotypic variation in the swarm.

- The $(n+1)$th conditional log likelihood is estimated by $f(y_{n+1} \,|\, y_{1:n}) \approx (1/J) \sum_{j=1}^{J} w_j$.

Mixing is needed for numerical stability

- A stochastic process $Z(\cdot)$ is **mixing** if $Z(s+t)$ has negligible information about $Z(s)$ for sufficiently large t. We do not need a more precise definition here.
- Mixing (information in the present about the past) is related to predictability (information in the past about the present).
- Filtering algorithms for mixing processes can expect to be numerically stable: small errors made at one time will have have diminishing rather than growing consequences.

An important non-mixing process:
Combine the state and parameter vectors of a POMP to give $Z(t) = \big(X(t), \theta\big)$. The filtering distribution for $Z(t_N)$ gives the posterior distribution of θ given $y_{1:N}$. Ideally, the filtering and prediction recursions could be used to compute this posterior. In practice, lack of mixing means that numerical instability is prohibitive.

Particle depletion & effective sample size

- J dependent particles carry less information about a target distribution than J independent particles. **Effective sample size (ESS)** is the equivalent number of independent particles.
- The particle filtering update step increases the ESS, if the process is mixing: adding random variation increases population diversity.
- Natural selection decreases ESS. Only a subset of the particles at time n have offspring at $n+1$. Pairs of particles with a common ancestor more recent than the mixing timescale of the process are dependent.
- Low ESS (from heavy selection or slow mixing) makes the swarm unrepresentative of its target.
- The actual ESS is usually unknown, but approximations can provide useful diagnostics of successful filtering. A one-step approximation is $ESS \approx \left\{ \sum_{j=1}^{J} w_j \right\}^2 \left\{ \sum_{j=1}^{J} w_j^2 \right\}^{-1}$.

Parameter estimation using SMC

- **Likelihood maximization**. SMC provides a noisy (Monte Carlo) likelihood estimate.

  ◊ Seed fixing (the method of common random numbers [10]) fails since resampling gives discontinuous functions of the random numbers.

  ◊ For low dimensional parameter spaces, the likelihood can be approximated directly by smoothed Monte Carlo estimates [3].

  ◊ Otherwise, a stochastic maximization algorithm [10, 8, 4] is needed.

- **Bayesian computation**. Recall that naively adding parameters to the state space gives an unstable Bayesian computation. To generate mixing and reduce depletion, dynamic noise can be added to the parameters in various ways [6, 5]. This strategy is also used for deterministic filters [11].

Fixed vs time-varying parameters

- Dynamic noise added to parameters is a way to model time-varying parameters. Arguably, most systems vary over time. If it is computationally convenient to include this, then why not?
- Sometimes time-varying parameters are what you want, but...
  - ◊ It is hard to interpret results when each parameter varies over time.
  - ◊ Often, we want to understand how parameters vary as a function of covariates. Thus, the parameter varies over time but depends on covariates in a fixed way.
  - ◊ Including dynamic noise in parameters doubles the degrees of freedom in the model (each noise intensity must be estimated).

Initial value parameters (IVPs)

- The **initial value parameter vector** ϕ is a sub-vector of θ which determines the values of $X(t_0)$.
- The identity parameterization $\phi = X(t_0)$ is often used.
- Initial values are distinct from *starting values* required to initialize an estimation algorithm.
- **Are IVPs needed?**
  - ◊ If the dynamics are **stationary**, $X(t_0)$ can be modeled as a random draw from the stationary distribution. This avoids the need for IVPs.
  - ◊ A system with dynamic covariates (i.e., external forcing) is non-stationary.
  - ◊ Biological systems are often non-stationary.
- Information on IVPS from data is concentrated in time. This hinders numerical stabilization by time-varying parameters. Non-IVP parameters, such as break points, can share this issue.

Heuristics of hard maximization

- Attaining and certifying global maximization is infeasible for large multimodal surfaces.
- Theorems guaranteeing global convergence are useful (they suggest decent local behavior) but should not be taken literally (unless you have infinite computational resources).
- There is no substitute for trying many starting values.
- Many algorithms move from wide search (high temperature) to local search (cold temperature). Tempering (non-monotone cooling) can work better than theoretically justified annealing (monotone cooling).
- These considerations also broadly apply to Monte Carlo estimates of posterior distributions.

Overview of iterated filtering

- Uses the trick of adding noise to stabilize computations.
- Reduces the added noise in a sequence of filtering iterations, approaching the fixed parameter limit.
- Produces a non-Bayesian estimate (the MLE).
- Each iteration looks like a Bayesian computation with time-varying parameters, though with a "prior" which contracts toward the fixed parameter MLE.
- A suitable average of the time-varying parameters is used to update the current approximation to the MLE.
- Algorithmically, the SMC **recursion** over time is nested within filtering **iterations** having diminishing parameter noise. "Recursion" and "iteration" are synonymous here, but conventionally refer to their respective loops.

Evolution analogy for iterated filtering

- For SMC with time-varying parameters, every particle has a parameter value (the **genotype**) and a state value (the **phenotype**).
- Each resampling event (each observation time for vanilla SMC) leads to a new generation.
  - ◊ Particles consistent with the new observation are preferentially propagated. This **natural selection** operates on the phenotype.
  - ◊ Genotype diversity is generated by random perturbation of the parameter vector (**genetic mutation**).
  - ◊ The phenotype for the next generation depends on both the genotype and the current phenotype (**epigenetics**).
- Iterated filtering uses the output of one SMC operation to create a new founder population for the subsequent SMC operation.

References

[1] Doucet, A., de Freitas, N., and Gordon, N. J., editors (2001). *Sequential Monte Carlo Methods in Practice*. Springer, New York.

[2] Hammersley, J. M. and Morton, K. W. (1954). Poor man's Monte Carlo. *Journal of the Royal Statistical Society, Series B (Statistical Methodology)*, 16:23–38.

[3] Ionides, E. L. (2005). Maximum smoothed likelihood estimation. *Statistica Sinica*, 15:1003–1014.

[4] Kiefer, J. and Wolfowitz, J. (1952). Stochastic estimation of the maximum of a regression function. *Annals of Mathematical Statistics*, 23:462–466.

[5] Kitagawa, G. (1998). A self-organising state-space model. *Journal of the American Statistical Association*, 93:1203–1215.

[6] Liu, J. and West, M. (2001). Combining parameter and state estimation in simulation-based filtering. In Doucet, A., de Freitas, N., and Gordon, N. J., editors, *Sequential Monte Carlo Methods in Practice*, pages 197–224. Springer, New York.

[7] Liu, J. S. (2001). *Monte Carlo Strategies in Scientific Computing*. Springer, New York.

[8] Robbins, H. and Monro, S. (1951). A stochastic approximation method. *Annals of Mathematical Statistics*, 22:400–407.

[9] Rosenbluth, M. N. and Rosenbluth, A. W. (1955). Monte Carlo calculation of the average extension of molecular chains. *Journal of Chemical Physics*, 23:356–359.

[10] Spall, J. C. (2003). *Introduction to Stochastic Search and Optimization*. Wiley, Hoboken.

[11] Wan, E. and Van Der Merwe, R. (2000). The unscented kalman filter for nonlinear

estimation. In *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000*, pages 153 –158.