Flow-based Bayesian Estimation of Nonlinear Differential Equations for Modeling Biological Network

Nicolas Brunel and Florence d'Alché-Buc

IBISC
Université d'Evry, France
ANR Support: GD2GS Project and ODESSA Project

Nijmegen, PRIB 2010

Outline

Quantitative Models of Biological Networks

System of Ordinary Differential Equations (ODE)

$$\dot{x} = f(t, x(t); \theta)$$

- $x(t)$: state variables at time t
  - protein concentrations
  - mRNA concentrations
  - metabolite concentrations
- f : encodes the structure of the system
  - nonlinear function
  - Michaelis-Menten / Hill kinetics
  - Mass action kinetics
  - ...
- θ: parameter set (kinetic parameters, rate constants,...)

Reverse Modeling of Biological Networks

Given

- An ODE model (Initial Value Problem): $x(0) = x_0$ and

$$\dot{x}(t) = f(t, x(t), \theta)$$

- A partially and noisy observation model:

$$y(t) = h(x(t)) + \epsilon(t)$$

  where h is a nonlinear observation function, $\epsilon(t)$ is a i.i.d noise
- A sequence of observed data : $y_{0:N-1} = \{y_0, ..., y_{N-1}\}$ at time $t_0 = 0, t_1, ..., t_{N-1}$

Reverse Modeling of Biological Networks

Given

- An ODE model (Initial Value Problem): $x(0) = x_0$ and

$$\dot{x}(t) = f(t, x(t), \theta)$$

- A partially and noisy observation model:

$$y(t) = h(x(t)) + \epsilon(t)$$

  where h is a nonlinear observation function, $\epsilon(t)$ is a i.i.d noise
- A sequence of observed data : $y_{0:N-1} = \{y_0, ..., y_{N-1}\}$ at time $t_0 = 0, t_1, ..., t_{N-1}$

Goal

- Estimation of parameters θ
- Estimation of states $x(t)$ (partially observed)

$\Longrightarrow$ Need for estimation of initial conditions (usually unknown / observed with noise)

Reminder: Initial Value Problem

- Parametrized ODE:
$$\dot{x}(t) = f(t, x(t), \theta) \tag{1}$$
defined for $t \in [0, T]$ ($T > 0$), and f is a vector field indexed by a parameter $\theta \in \Theta \subset R^p$.
- Under general conditions on f (typically Lipschitz in x), there exists a unique solution to (1) for a given initial value $x(0) = x_0$ on the interval $[0, T]$.
- The solution for parameter θ and initial value x_0 is denoted $\phi : t \mapsto \phi_\theta(t, x_0)$.
- $\phi : x_0 \mapsto \phi_\theta(t, x_0)$ is the **flow** of the ODE (action of the vector field in the state space).

Proposed approach

- Search for reconstructed trajectories corresponding to a solution of the ODE with given parameter estimates
- In state-space model approach, the dependence on initial conditions is forgotten: introduction of the Flow method
- Monte Carlo approximation of the posterior distribution

Augmented initial condition

- Let's define:

$$\begin{cases} \dot{x}(t) & = f\left(t, x(t), \theta(t)\right) \\ \dot{\theta}(t) & = 0 \end{cases} \tag{2}$$

- Augmented initial condition $z_0 = (x_0, \theta)$.
- The solution is the function $t \mapsto \phi\left(t,\, z_0\right)$ from $[0,\, T]$ to R^{p+d}.
- **New setting:**

$$\begin{cases} \dot{z}(t) & = f(t, z(t)) \\ z(0) & = z_0 \end{cases} \tag{3}$$

State-Space Model interpretation

Recursive definition of the state-space model

The dynamical system is observed at discrete-time points $t_0, ..., t_{N-1}$

$$\begin{aligned} \mathbf{z}_{n+1} &= \mathbf{F}_n(\mathbf{z}_n) \\ \mathbf{y}_n &= \mathbf{h}(\mathbf{z}_n) + \epsilon_n \end{aligned}$$

with

$$\mathbf{F}_n(\mathbf{z}_n) = \phi(t_n, \mathbf{z}_0) = \mathbf{z}_n + \int_{t_n}^{t_{n+1}} \mathbf{f}(\tau, \mathbf{z}(\tau); \mathbf{z}_0) d\tau$$

State-Space Model interpretation

Recursive definition of the state-space model

The dynamical system is observed at discrete-time points $t_0, ..., t_{N-1}$

$$\begin{aligned} \mathbf{z}_{n+1} &= \mathbf{F}_n(\mathbf{z}_n) \\ \mathbf{y}_n &= \mathbf{h}(\mathbf{z}_n) + \epsilon_n \end{aligned}$$

with

$$\mathbf{F}_n(\mathbf{z}_n) = \phi(t_n, \mathbf{z}_0) = \mathbf{z}_n + \int_{t_n}^{t_{n+1}} \mathbf{f}(\tau, \mathbf{z}(\tau); \mathbf{z}_0) d\tau$$

A non recursive expression for the hidden states

$$\begin{aligned} \mathbf{z}_n &= \mathbf{z}_0 + \int_0^{t_n} f(\mathbf{z}(\tau), \mathbf{z}_0) d\tau \\ &= \phi(t_n, \mathbf{z}_0) \end{aligned}$$

Bayesian estimation of the augmented initial condition 1

- Collection of the observed time series (Imperfect observation of the system):

$$y_0, \ldots, y_{N-1}$$

- Choice of a prior distribution:

$$\pi_{-1}(z_0)$$

(more or less classical choice)

- Focus on the Posterior Mean among possible Bayesian estimator (i.e. Posterior Least Square Error Estimate):

$$\hat{z}_0 = \int z_0 p(z_0 | y_{0:N-1}) dz_0$$

- Approximation of $\pi_{N-1} = p(z_0 | y_{0:N-1})$ (no closed-form) and of $\hat{z}_0$.

Bayesian estimation of the augmented initial condition 2

- Homoscedasticity assumption: $\sigma^2 \times I_m \in \mathbb{R}^{m \times m}$
- Gaussian Noise $\implies$ Gaussian Likelihood

$$L(y_{0:N-1}; z_0) \propto \sigma^{-mN} \exp\left(-e(y_{0:N-1}, z_0)\right) \tag{4}$$

  with $e(y_{0:N-1}, z_0) = \frac{1}{2\sigma^2} \sum_{i=0}^{N-1} \|y_i - h(\phi(t_i, z_0))\|^2$.
- Posterior distribution of the parameter z_0:

$$p(z_0 | y_{0:N-1}) \propto \exp\left(-e(y_{0:N-1}, z_0)\right) \pi_{-1}(z_0) \tag{5}$$

Outline

Stochastic simulation or Monte Carlo methods

- Monte Carlo = approximation of an expectation by the sample mean of a function of simulated random variables.
- Application in Bayesian Inference: approximate $p(z_0|y_{0:n-1})$ and estimate $E[z_0|y_{0:n-1}]$

Importance Sampling

- Given a target density p and a proposal density q,
- Estimate $E_p[Z] = \int z.p(z)dz = \int z.\frac{p(z)}{q(z)}q(z)dz$
- Using $(\xi_1, \ldots, \xi_M)$, a iid sample from the distribution Q of density q

$$\begin{cases} \hat{E}_p[Z] = \frac{1}{M}\sum_{i=1}^{M} w_i \xi_i \\ w_i = \frac{p(\xi_i)}{q(\xi_i)} \end{cases} \tag{6}$$

- Density p is known up to the normalizing constant: one uses self-normalized weights $\tilde{w}_i = \frac{w_i}{\sum_{j=1}^{M} w_j}$
- Necessary condition of consistency: p has its support included in the support of q
- Limit theorems (Large numbers, Central Limit Theorem) guarantee convergence of IS (and rate of convergence) when $M \longrightarrow \infty$

Population Monte Carlo (PMC) and Iterative Importance Sampling Resampling

- Importance Sampling Resampling: Resample $\xi_i, i = 1, \ldots, M$ with a multinomial $\mathcal{M}(\tilde{w}_i)$ (with replacement)
- Population Monte Carlo: Iterative construction of new samples (populations) by "improving" the proposal distribution $q_t,\ t = 1, \ldots, T$:
- At each step t:
  1. Generate $(\xi_{i,t})_{1 \leq i \leq M} \sim q_t$ (i.i.d sampling) and compute normalized weights $\tilde{w}_{i,t}$,
  2. Resample $(\tilde{\xi}_{i,t})_{1 \leq i \leq M}$ by multinomial sampling with weights $\tilde{w}_{i,t}, i = 1, \ldots, M$
  3. Construct q_{t+1} from $((\tilde{\xi}_{i,t'}, \tilde{w}_{i,t'}))_{1 \leq i \leq M, 0 \leq t' \leq t}$

PMC and Adaptive Importance Sampling Resampling:

- Proposal densities? *D*-**kernel PMC** (Douc et al., AOS, 2007) uses a mixture of *D* transition kernels

$$\begin{cases} \xi_{i,t} = \xi_{i,t-1} + \epsilon_{i,t} \\ \epsilon_{i,t} \sim \sum_{d=1}^{D} \alpha_d N(0, \Sigma_{d,t}) \end{cases} \tag{7}$$

- Adaptivity ?
  - Possible sizes of jumps represented by different Σ_d
  - Adaptivity by augmenting the probability of components $N(0, \Sigma_{d,t})$ which ameliorate the likelihood
  - The idea is to make the proposal density closer to the target density

$D-$ *kernel* PMC: Learning algorithm

- Minimization of Kullback-Leibler divergence w.r.t the objective function
- EM-like algorithm used for the estimation of the mixture parameters $\alpha_d, d = 1, \ldots, D$
- At $t = 0$: generate $(\tilde{\xi}_{i,0})$ by IS-Resampling and start with $\alpha_d^0 = 1/D$.
- For $t = 1, \ldots, T$
  1. Generate index $J_{i,t} \in [1, .., D]$ with $J_{i,t} \sim \mathcal{M}(M, (\alpha_d^t))$
  2. Generate independent moves $\xi_{i,t} \sim K_{J_{i,t}}(\xi_{i,t-1}, z)$
  3. Compute the normalized importance weights:

$$\tilde{w}_{i,t} \propto \frac{\pi_{N-1}(\xi_{i,t})}{\sum_d \alpha_d^t K_d(\tilde{\xi}_{i,t-1}, \xi_{i,t})}$$

  4. Resample $(\xi_{i,t})$ into $(\tilde{\xi}_{i,t})$ by multinomial sampling $\mathcal{M}(M, \tilde{w}_{i,t})$
  5. Update mixture weights:

$$\alpha_d^{t+1} = \sum_{i/J_{i,t}=d} \tilde{w}_{i,t}$$

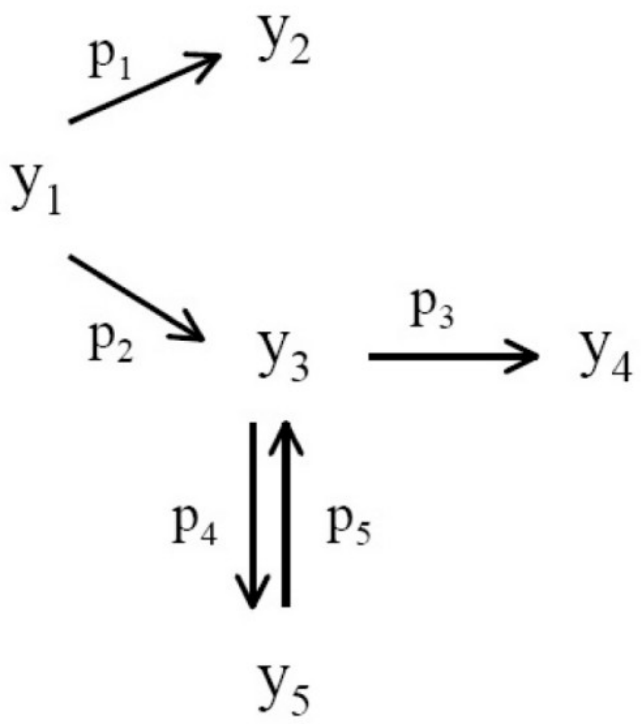
Outline

Numerical simulations

- Estimate parameters in a biochemical network (Isomerization of $\alpha-$pinene network)
  - Noninformative uniform prior on an interval centered in true parameters
  - $N = 8, M = 5000, D = 7, T = 20$
  - Only parameters p_i's to be learnt (initial conditions are known)
  - $\sigma_d = 10^{-11}, \ldots, 10^{-5}$
- Estimate parameters in a regulatory network (Repressilator network)
  - $N = 25$ (simulated date), $M = 1000, D = 7, T = 10$
- Comparison of several Monte Carlo (Bayesian) methods: Unscented Kalman Smoothing, Flow-Based Estimate with Importance Sampling, Flow-Based Estimate with PMC

Isomerization of $\alpha-$pinene network

- Biochemical reactions
- Isomerization of $\alpha-$pinene (y1) to dipentene (y2) and allo-ocimen (y3) which in turn yields $\alpha-$ and $\beta-$ *pyronene* (y4) and a dimer (y5)



Differential equations for $\alpha-$pinene network

- Linear model with 5 equations

$$\begin{cases} \dot{x}_1 &= -(p_1 + p_2)x_1 \\ \dot{x}_2 &= p_1 x_1 \\ \dot{x}_3 &= p_2 x_1 - (p_3 + p_4)x_3 + p_5 x_5 \\ \dot{x}_4 &= p_3 x_3 \\ \dot{x}_5 &= p_4 x_3 + p_5 x_5 \end{cases} \tag{8}$$

- 5 rate constants $\theta = (p_1, \ldots, p_5)$ to be estimated,
- Initial conditions are known.

Parameter estimation

- Results and comparisons on real dataset (Banga, 2006)

$\times 10^{-5}$	Reference	UKS	FBE–IS	FBE–PMC
p_1	5.92	3.66 ± 5.6	$5.98 \pm 1.3 \times 10^{-2}$	$5.93 \pm 4.7 \times 10^{-2}$
p_2	2.96	2.5 ± 4.8	$2.92 \pm 1.3 \times 10^{-2}$	$2.96 \pm 5 \times 10^{-2}$
p_3	2.04	1.78 ± 20.4	$2.05 \pm 5.69 \times 10^{-2}$	$2.06 \pm 2 \times 10^{-2}$
p_4	27.44	27.3 ± 31.1	$26.7 \pm 5.69 \times 10^{-2}$	$27.89 \pm 10 \times 10^{-2}$
p_5	3.99	4.24 ± 26	$3.53 \pm 13.1 \times 10^{-2}$	$4.11 \pm 5.2 \times 10^{-2}$
$\lVert \hat{\theta} - \theta^{ref} \rVert$	0	2.3×10^{-5}	8.2×10^{-6}	4.5×10^{-6}

Table: Estimated parameter values with UKS, IS and PMC with standard deviation.

Population evolution and adaptativity in $\alpha-$pinene

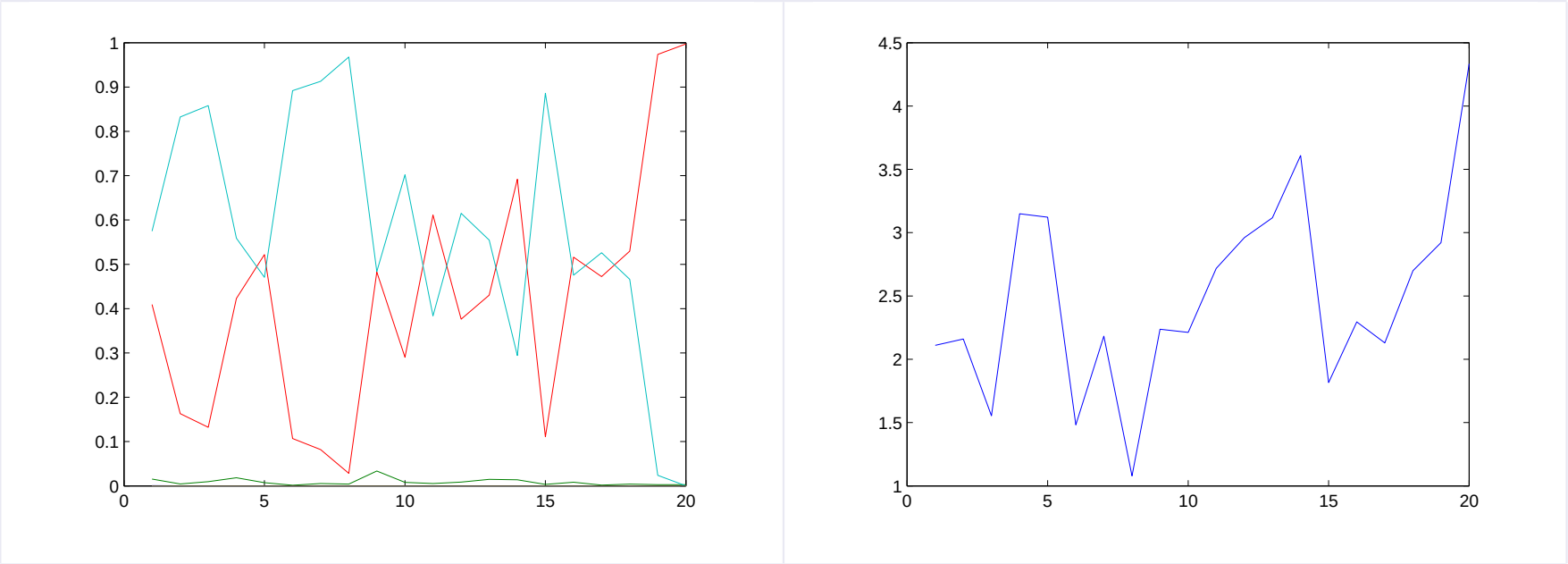
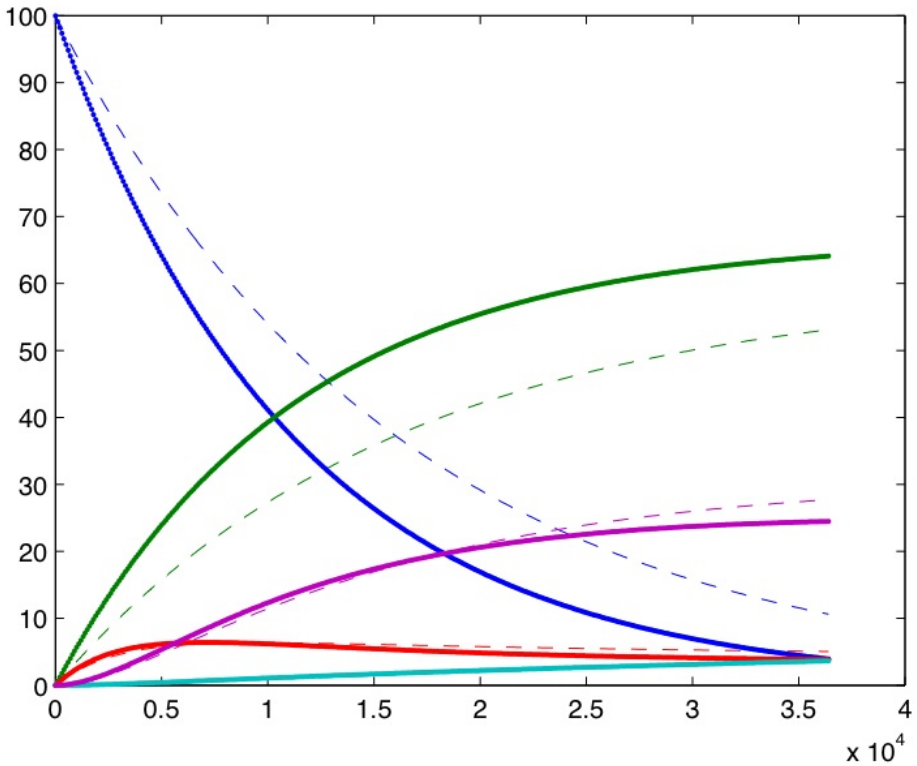
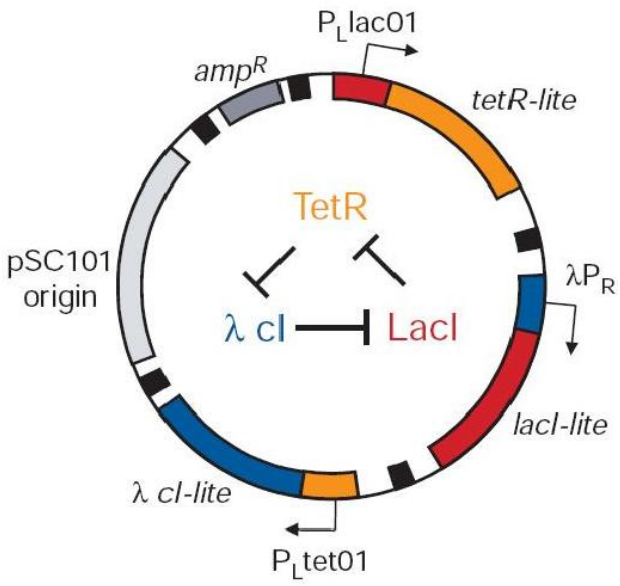


Figure: Adaptive ISR and reduction of weight degeneracy with $D = 3$ and $T = 20$: mixture coefficients α_d and entropy of the population weights

ODE solution after parameter estimation



Repressilator, [Elowitz, Nature 2000]



$$\frac{dr_1}{dt} = v_1^{max} \frac{k_{12}^n}{k_{12}^n + p_2^n} - k_1^{mRNA} r_1$$

$$\frac{dr_2}{dt} = v_2^{max} \frac{k_{23}^n}{k_{23}^n + p_3^n} - k_2^{mRNA} r_2$$

$$\frac{dr_3}{dt} = v_3^{max} \frac{k_{31}^n}{k_{31}^n + p_1^n} - k_3^{mRNA} r_3$$

$$\frac{dp_1}{dt} = k_1 r_1 - k_1^{protein} p_1$$

$$\frac{dp_2}{dt} = k_2 r_2 - k_2^{protein} p_2$$

$$\frac{dp_3}{dt} = k_3 r_3 - k_3^{protein} p_3$$

- mRNAs are observed, proteins are hidden
- mRNA and protein degradation rate constants are supposed to be known
- Estimate 9 parameters and initial conditions

Results for the Repressilator for $N = 25$ observations

- Parameter estimation

Parameter	True Parameter	UKS	FBE–IS	FBE–PMC
v_1^{max}	150	147.3 ± 0.9	150.2 ± 0.09	150.0 ± 0.46
v_2^{max}	80	81.9 ± 1.7	80.7 ± 0.49	80.2 ± 0.66
v_3^{max}	100	102.2 ± 1.7	100.7 ± 0.25	100.1 ± 0.91
k_1	50	53.0 ± 0.9	50.7 ± 0.05	50.1 ± 0.35
k_2	30	37.1 ± 0.94	30.9 ± 0.08	29.9 ± 0.38
k_3	40	47.6 ± 0.8	40.72 ± 0.03	40.0 ± 0.36

- Initial conditions for hidden variables

Parameter	True	UKS	FBE–IS	FBE–PMC
$p_1(0)$	1	97.8 ± 5.9	3.11 ± 0.21	2.86 ± 0.09
$p_2(0)$	2	143.6 ± 3.0	3.83 ± 0.21	3.51 ± 0.10
$p_3(0)$	3	148.5 ± 8.6	4.76 ± 0.17	4.75 ± 0.27

Conclusion and perspective

- Introduction of the Flow for retrieving exact solution of the ODE (specific to deterministic hidden process driven by ODE)
- First results $\rightarrow$ better estimate and better ODE equation at the end
  - Population Monte Carlo to approximate posterior probability
  - D-PMC of Douc et al. implemented and tested
- **Current and future work**
  - Study the impact of non recursive definition e.g. integration on all the time interval (and recursive version of a learning algorithm?)
  - Impact of prior choice
  - Other proposal densities and other rules for adaptive ISR: M-PMC (mixture of Gaussian distributions)
  - Combine structure and parameter learning in a single PMC scheme