An Efficient Forward-Reverse Expectation-Maximization Algorithm for Statistical Inference in Stochastic Reaction Networks

Christian Bayer¹, Alvaro Moraes², Raúl Tempone², and Pedro Vilanova²

¹ Weierstrass Institute for Applied Analysis and Stochastics, Berlin
² Mathematical and Computer Sciences and Engineering Division, King Abdullah University of Science and Technology (KAUST)

UQAW
KAUST, January 2016
The inference problem

**Motivation:** Estimate parameters of a Stochastic Reaction Network (SRN) from \textit{discretely observed} data. A SRN is a continuous time Markov Chain

\[ X = (X_1, \ldots, X_d) : [0, T] \times \Omega \to \mathbb{Z}_+^d \]

is described by \( J \) reaction channels, \((\nu_j, a_j)\), where

- \( \nu_j \in \mathbb{Z}^d \) (\( x \rightarrow x + \nu_j \)),
- \( a_j : \mathbb{Z}_+^d \times \Theta \to \mathbb{R}^+ \) (\( \Theta \) finite dimensional), and

\[
P(X(t+\Delta t) = x+\nu_j \mid X(t) = x) = a_j(x; \theta) \Delta t + o(\Delta t).
\]

We set \( a_j(x; \cdot) = 0 \) for those \( x \) such that \( x + \nu_j \notin \mathbb{Z}_+^d \).

**Goal:** Assuming \( a_j(x; \theta) = \theta_j g_j(x) \) for every \( j \), estimate \( \theta_j > 0 \) for \( g_j \) known functions, from a discretely observed data set

\[
\mathcal{D} := ([s_k, t_k], x(s_k), x(t_k))_{k=1}^K,
\]

where \( s_k < t_k \) are two consecutive points where the states \( x(s_k) \) and \( x(t_k) \) have been observed.
Likelihood function for SRNs

To simulate a path of $X$ from $t=0$ until $t=T$ from $X(0)=x$,

1. Simulate the next reaction to occur, $j$, with probability $a_j(x)/\sum_j a_j(x; \theta)$.

2. Simulate independently the time to the next reaction, $\tau$, as an exponential random variable with rate $\sum_j a_j(x; \theta)$.

3. Update $t \leftarrow t + \tau$ and $x \leftarrow x + \nu_j$. Repeat while $t < T$.

Given a continuously observed path $C:=(\tau_i, j_i)_{i=1}^m$ along $[0, T]$ the complete data log-likelihood is

$$
\log L_C(\theta|C) = \sum_{j=1}^J \left( \log(\theta_j) R_j, [0, T] - \theta_j F_j, [0, T] \right),
$$

where

- $R_j, [0, T]$ is the number of times channel $j$ fired in $[0, T]$.
- $F_j, [0, T] = \int_0^T g_j(X(s))ds$. 

Likelihood function for SRNs

Assume a collection of intervals, \( (I_k = [s_k, t_k])_{k=1}^K \subset [0, T] \), where we have continuously observed the process \( (X(t))_{t \in I_k} \) at each \( I_k \),

\[
\log L^c(\theta | C) = \sum_{j=1}^J \left( \log(\theta_j) \sum_{k=1}^K R_{j,I_k} - \theta_j \sum_{k=1}^K F_{j,I_k} \right).
\]

The MLE for complete data is

\[
\theta^*_j = \frac{\sum_{k=1}^K R_{j,I_k}}{\sum_{k=1}^K F_{j,I_k}}, \quad j=1, \ldots, J.
\]

**Issue:** Usually, it is only possible to observe \( X \) at discrete measurement times.
Missing data problem

Idea: Fill into the “gaps” using a Forward-Reverse technique.

Figure: Left: forward-reverse exact path simulation for a decay problem (SRN-bridges). Right: shooting method.

Treat data $\mathcal{D}$ as a part of a larger data set, $(\mathcal{D}, \tilde{\mathcal{D}})$, where the complete likelihood is $L^c(\theta|\mathcal{D}, \tilde{\mathcal{D}})$. 
The reverse process [Bayer et al., 2015]

Let $X(t)$ be a SRN with reaction channels $(\nu_j, a_j)_{j=1}^J$. Then a reverse process $Y$ with reaction channels $(-\nu_j, \tilde{a}_j)_{j=1}^J$,

$$\tilde{a}_j(y; \theta) := a_j(y-\nu_j; \theta)$$

is also a SRN such that

$$P \left( Y(\tilde{t}+\Delta\tilde{t}) = y-\nu_j \mid Y(\tilde{t}) = y \right) = \tilde{a}_j(y; \theta)\Delta\tilde{t} + o(\Delta\tilde{t}).$$

Note that $Y$ runs forward in time.

Let $[s, t]$ be a time interval, and $t^* \in [s, t]$. Then,

- $X^{(f)}$ denotes a process starting at the observed data $x \in \mathbb{Z}_+^d$, defined in $[s, t^*]$.
- $X^{(b)}$ denotes the reverse process $Y$ run backwards in time: $X^{(b)}(u) := Y(t^*+t-u)$ for $u \in [t^*, t]$, and $X^{(b)}(t) = y \in \mathbb{Z}_+^d$. 


The Forward-Reverse Formula for SRNs

**Theorem ([Bayer et al., 2015])**

Let $\Phi$ be a continuous map from $\mathbb{Z}_+^d$-valued paths to $\mathbb{R}$. Then,

$$
\mathbb{E} \left[ \Phi \left( X, [s, t] \right) \mid X(s) = x, X(t) = y \right] = 
\lim_{\epsilon \to 0} \frac{\mathbb{E} \left[ \Phi \left( X^{(f)} \circ X^{(b)}, [s, t] \right) \kappa_\epsilon \left( X^{(f)}(t^*) - X^{(b)}(t^*) \right) \Psi \left( X^{(b)}, [t^*, t] \right) \right]}{\mathbb{E} \left[ \kappa_\epsilon \left( X^{(f)}(t^*) - X^{(b)}(t^*) \right) \Psi \left( X^{(b)}, [t^*, t] \right) \right]},
$$

where $\kappa_\epsilon$ is a whole family of kernels indexed by the bandwidth $\epsilon \geq 0$,

$$
X^{(f)} \circ X^{(b)}(u) := \begin{cases} 
X^{(f)}(u), & s \leq u \leq t^*, \\
X^{(b)}(u), & t^* < u \leq t,
\end{cases}
$$

and

$$
\Psi(Z, [a, b]) := \exp \left( \int_a^b c(Z(u)) \, du \right), \quad c_j(y) := a_j(y - \nu_j) - a_j(y).
$$
The EM algorithm [Dempster et al., 1977]

The Expectation Maximization (EM) algorithm approximates a local maximum or saddle point of a likelihood function. Given \( \theta^{(0)} \), the EM algorithm maps \( \theta^{(p)} \) into \( \theta^{(p+1)} \) in 2 steps

1. **Expectation step:**
   \[ Q_{\theta(p)}(\theta|\mathcal{D}) := E_{\theta(p)} \left[ \log L^c(\theta|\mathcal{D}, \tilde{\mathcal{D}}) | \mathcal{D} \right]. \]

2. **Maximization step:**
   \[ \theta^{(p+1)} := \arg \max_{\theta} Q_{\theta(p)}(\theta|\mathcal{D}). \]

where \( E_{\theta(p)} [\cdot | \mathcal{D}] \) is the expectation conditional to the data \( \mathcal{D} \) under the parameter choice \( \theta^{(p)} \).

For the SRN case we obtain

\[
\theta_j^{(p+1)} = \frac{\sum_{k=1}^{K} E_{\theta(p)} \left[ R_{j,k} | \mathcal{D} \right]}{\sum_{k=1}^{K} E_{\theta(p)} \left[ F_{j,k} | \mathcal{D} \right]}, \quad j=1, \ldots, J.
\]
Forward-reverse EM algorithm (FREM)

Use Monte Carlo to estimate $\theta_j^{(p+1)}$ with

$$
\hat{\theta}_j^{(p+1)} := \frac{\sum_{k=1}^K A_{\hat{\theta}(p)}(R_j,l_k \mid \mathcal{D}; \cdot)}{\sum_{k=1}^K A_{\hat{\theta}(p)}(F_j,l_k \mid \mathcal{D}; \cdot)}, \quad j=1, \ldots, J.
$$

and iterate until convergence to get a final estimation.

The FREM generates a sequence $(\hat{\theta}^{(p)})_{p=1}^{+\infty}$ in two phases:

**Phase I:** find a suitable $\hat{\theta}^{(0)}$ to reduce the computational work of the Monte Carlo EM (robustness).

**Phase II:** simulate F-R paths to compute

$$A_{\hat{\theta}(p)}(R_j,l_k \mid \mathcal{D}; \cdot) \quad \text{and} \quad A_{\hat{\theta}(p)}(F_j,l_k \mid \mathcal{D}; \cdot).$$
Forward-reverse EM algorithm (FREM)

**Phase I:** find a suitable initial estimation $\hat{\theta}(0)$ to reduce the computational work of phase II:

$$\hat{\theta}(0) := \arg\min_{\theta \geq 0} \sum_{k=1}^{K} w_k \left\| Z^{(f)}(t^*_k; \theta) - Z^{(b)}(t^*_k; \theta) \right\|_2^2,$$

where $Z^{(f)}$ and $Z^{(b)}$ are the ODE approximations to the average of $X^{(f)}$ and $X^{(b)}$, for a suitable $t^*_k \in I_k$, and a suitable weight $w_k$ (for example $(t_k - s_k)^{-1}$).

**Goal:** Increase the number of joined forward-reverse paths for all time intervals.
Forward-reverse EM algorithm (FREM)

**Phase II:** Given the running guess $\hat{\theta}(p)$:

- simulate $M_k$ forward paths in $[s_k, t_k^*]$ and record $R_{j,l_k}^{(f)}(\tilde{\omega}_m)$ and $F_{j,l_k}^{(f)}(\tilde{\omega}_m)$ for all $j$ and $m$.
- same for $R_{j,l_k}^{(b)}(\tilde{\omega}_{m'})$ and $F_{j,l_k}^{(b)}(\tilde{\omega}_{m'})$ in $[t_k^*, t_k]$.

Compute:

\[
A_{\hat{\theta}(p)}(R_{j,l_k} \mid D; \kappa) := \frac{\sum_{1 \leq m, m' \leq M_k} \left( R_{j,l_k}^{(f)}(\tilde{\omega}_m) + R_{j,l_k}^{(b)}(\tilde{\omega}_{m'}) \right) \kappa_{m,m',k} \psi_{m',k}}{\sum_{1 \leq m, m' \leq M_k} \kappa_{m,m',k} \psi_{m',k}},
\]

\[
A_{\hat{\theta}(p)}(F_{j,l_k} \mid D; \kappa) := \frac{\sum_{1 \leq m, m' \leq M_k} \left( F_{j,l_k}^{(f)}(\tilde{\omega}_m) + F_{j,l_k}^{(b)}(\tilde{\omega}_{m'}) \right) \kappa_{m,m',k} \psi_{m',k}}{\sum_{1 \leq m, m' \leq M_k} \kappa_{m,m',k} \psi_{m',k}},
\]

where $\kappa$ is a kernel function, and $\psi$ is the weighting factor of the reverse process.
Forward-reverse EM algorithm (FREM)

Example: Kronecker kernel,

\[ \kappa_{m,m',k} = \begin{cases} 1 & \text{if } X^{(f)}(t_k^*, \tilde{\omega}_m) = X^{(b)}(t_k^*, \tilde{\omega}_{m'}) \\ 0 & \text{otherwise} \end{cases} \]

Note 1: \( \hat{\theta}(p) \) may not provide a large number of joined paths when using the Kronecker kernel. An Epanechnikov kernel may help.

Note 2: To reduce the work of computing the double sums, from \( \mathcal{O}(M_k^2) \) to up to \( \mathcal{O}(M_k) \) a space partition procedure is proposed.
The Epanechnikov kernel

Motivation: Relax the Kronecker kernel to get in average $O(M)$ joined paths.

1. Transform the endpoints of the forward and backward paths generated in a given interval $k$,

$$ X_k := (X^{(f)}(t^*_k, \tilde{\omega}_1), X^{(f)}(t^*_k, \tilde{\omega}_2), \ldots, X^{(f)}(t^*_k, \tilde{\omega}_{M_k}),$$

$$ X^{(b)}(t^*_k, \tilde{\omega}_{M_k+1}), X^{(b)}(t^*_k, \tilde{\omega}_{M_k+2}), \ldots, X^{(b)}(t^*_k, \tilde{\omega}_{2M_k}))$$

into

$$ T(X_k) := (Y^{(f)}(t^*_k, \tilde{\omega}_1), Y^{(f)}(t^*_k, \tilde{\omega}_2), \ldots, Y^{(f)}(t^*_k, \tilde{\omega}_{M_k}),$$

$$ Y^{(b)}(t^*_k, \tilde{\omega}_{M_k+1}), Y^{(b)}(t^*_k, \tilde{\omega}_{M_k+2}), \ldots, Y^{(b)}(t^*_k, \tilde{\omega}_{2M_k}))$$

using the linear transformation $T(x) := (\text{cov}(X_k))^{-1/2} x$. (Hopefully covariance matrix is close to $\alpha \text{Id}$).
The Epanechnikov kernel

Figure: Left: Example: a bivariate Gaussian cloud, \( Z \). Right: Its corresponding decorrelated and scaled version \( H(Z) \). On average, there is one point of the cloud in each \( d \)-dimensional cube (with sides parallel to the coordinate axis).
The Epanechnikov kernel

2 Choose $\alpha$ such that

$$M = (3\alpha)^d V_d,$$

we obtain $\alpha = \frac{1}{3} \left( \frac{M}{V_d} \right)^{1/d}$, where $V_d$ is the volume of the unitary sphere in $\mathbb{R}^d$. Finally,

$$H(x) := \alpha T(x).$$

3 Use a $d$-dimensional kernel (Epanechnikov)

$$\kappa(\eta) := \left( \frac{3}{4} \right)^d \prod_{i=1}^{d}(1 - \eta_i^2)1_{|\eta_i| \leq 1},$$

where $\eta$ is defined as

$$\eta \equiv \eta(m, m', k) := H(X^{(f)}(t^*_k, \tilde{\omega}_m)) - H(X^{(b)}(t^*_k, \tilde{\omega}_{m'})).$$
On the stopping criterion

We adapt the $\hat{R}$ criterion [Gelman and Rubin, 1992]: Monitor the convergence of $N$ parallel random sequences $(\hat{\theta}_i^{(p)})_{p=1}^{+\infty}$, $i=1, 2, \ldots, N$ starting from over-dispersed initial points.

Compute:

- **Between variance:**
  \[
  B_p := \frac{1}{N-1} \sum_{i=1}^{N} \left( \bar{\psi}_{p,i} - \bar{\psi}_p \right)^2,
  \]
  where
  \[
  \bar{\psi}_{p,i} := \frac{1}{p} \sum_{k=1}^{p} \hat{\theta}_i^{(k)} \quad \text{and} \quad \bar{\psi}_p := \frac{1}{N} \sum_{i=1}^{N} \bar{\psi}_{p,i}.
  \]

- **Within variance:**
  \[
  W_p := \frac{1}{N} \sum_{i=1}^{N} s_{p,i}^2, \quad \text{where} \quad s_{p,i}^2 := \frac{1}{p-1} \sum_{k=1}^{p} \left( \hat{\theta}_i^{(k)} - \bar{\psi}_{p,i} \right)^2.
  \]
On the stopping criterion

Then, define

\[ V_p := \frac{p - 1}{p} W_p + B_p, \text{ and} \]

\[ \hat{R}_p := \sqrt{\frac{V_p}{W_p}}. \]

It is expected that \( \hat{R}_p \) declines to \( 1 \) as \( p \to +\infty \).

If \( \bar{\psi}_{p,i} \approx \bar{\psi}_p \) (we have essentially only one Markov chain) then

\[ \hat{R}_p \approx \sqrt{\frac{p-1}{p}} \to 1 \text{ as } p \to +\infty \text{ independently of the behavior of the chain.} \]

Then observe also the behavior of the moving averages of order \( L \), that is,

\[ \bar{\psi}_p := \frac{1}{N} \sum_{i=1}^{N} \left( \bar{\psi}_{p,i} - \bar{\psi}_{p-1,i} \right)^2 \text{ where } \bar{\psi}_{p,i} := \frac{1}{L} \sum_{\ell=0}^{L-1} \hat{\theta}_{i}^{(p-\ell)}. \]
Example 1: Birth-death process

One dimensional, two reactions example [Daigle et al., 2012]:

\[
\emptyset \overset{\theta_1}{\rightarrow} X, \quad X \overset{\theta_2}{\rightarrow} \emptyset
\]

\[
\nu = \begin{pmatrix} 1 \\ -1 \end{pmatrix}, \quad a(X) = \begin{pmatrix} \theta_1 \\ \theta_2 X \end{pmatrix}
\]

Since we are not continuously observing the paths of \( X \), the estimation of the parameters is non-trivial.

**Synthetic data:** Start from \( X_0=17 \) until \( T=200 \), and observe a single path of \( X \) at regular time intervals of size \( \Delta t=5 \), using

\[
\theta_1=1, \quad \theta_2=0.06
\]
Example 1: Birth-death process
Example 1: Birth-death process

Figure: Cluster average of one FREM run: \( \hat{\theta} = (1.22, 0.065) \). Took 95 MCEM iterations to converge (runtime in the order of hours). In [Daigle et al., 2012], \( \hat{\theta} = (1.446, 0.093) \) with 234 MCEM iterations.
Example 1: Birth-death process

<table>
<thead>
<tr>
<th></th>
<th>Average</th>
<th>Average CI at 95%</th>
<th>Min Value</th>
<th>Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\hat{\theta}_1$</td>
<td>1.243</td>
<td>(1.237, 1.249)</td>
<td>1.213</td>
<td>1.284</td>
</tr>
<tr>
<td>$\hat{\theta}_2$</td>
<td>0.0659</td>
<td>(0.0655, 0.0663)</td>
<td>0.0643</td>
<td>0.0681</td>
</tr>
</tbody>
</table>

Table: Values computed for an ensemble of 30 independent runs of the FREM Algorithm. In each run, we obtain a cluster average as an MLE point estimate. The MLE is $(1.218, 0.0646)$. 
Example 1: Birth-death process

<table>
<thead>
<tr>
<th>Interval</th>
<th>Method</th>
<th>M 50</th>
<th>M 100</th>
<th>M 200</th>
<th>M 400</th>
</tr>
</thead>
<tbody>
<tr>
<td>([0, 5.12], 17, 16)</td>
<td>F-R</td>
<td>272</td>
<td>1094</td>
<td>4831</td>
<td>19809</td>
</tr>
<tr>
<td></td>
<td>S</td>
<td>6</td>
<td>13</td>
<td>17</td>
<td>50</td>
</tr>
<tr>
<td>([66.67, 71.79], 15, 15)</td>
<td>F-R</td>
<td>354</td>
<td>1178</td>
<td>5282</td>
<td>20214</td>
</tr>
<tr>
<td></td>
<td>S</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>([102.56, 107.69], 24, 19)</td>
<td>F-R</td>
<td>151</td>
<td>743</td>
<td>2466</td>
<td>11408</td>
</tr>
<tr>
<td></td>
<td>S</td>
<td>5</td>
<td>6</td>
<td>16</td>
<td>38</td>
</tr>
</tbody>
</table>

Table: Forward-Reverse and Shooting joined paths, with $\theta_1=1, \theta_2=0.06$
Example 2: Auto-Regulatory Gene Network

This example [Daigle et al., 2012] has 5 dimensions and 8 reactions,

\[
\begin{align*}
DNA + P_2 \xrightarrow{\theta_1} DNA - P_2, \\
DNA \xrightarrow{\theta_3} DNA + mRNA, \\
P + P \xrightarrow{\theta_5} P_2, \\
mRNA \xrightarrow{\theta_7} mRNA + P,
\end{align*}
\]

\[
\begin{align*}
DNA - P_2 \xrightarrow{\theta_2} DNA + P_2, \\
mRNA \xrightarrow{\theta_4} \emptyset, \\
P_2 \xrightarrow{\theta_6} P + P, \\
P \xrightarrow{\theta_8} \emptyset
\end{align*}
\]

**Data:** Start from \(X_0=(7, 3, 10, 10, 10)\) until \(T=50\), and observe a single path of \(X\) at regular time intervals of size \(\Delta t=1/2\), using \(\theta = (0.1, 0.7, 0.35, 0.3, 0.1, 0.9, 0.2, 0.1)\).

Run 2 FREM sequences starting at \((0.5, 0.5, ..., 0.5)\) and \((1, 1, ..., 1)\).
Example 2: Auto-Regulatory Gene Network

Data trajectory

Species count

Time

DNA
DNA-P2
mRNA
P
P2
Example 2: Auto-Regulatory Gene Network

<table>
<thead>
<tr>
<th></th>
<th>Average</th>
<th>Average CI at 95%</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\hat{\theta}_1)</td>
<td>0.1011</td>
<td>(0.1001, 0.1021)</td>
<td>0.0984</td>
<td>0.1033</td>
</tr>
<tr>
<td>(\hat{\theta}_2)</td>
<td>0.6207</td>
<td>(0.6135, 0.6279)</td>
<td>0.6005</td>
<td>0.6328</td>
</tr>
<tr>
<td>(\hat{\theta}_3)</td>
<td>0.3398</td>
<td>(0.3380, 0.3416)</td>
<td>0.3358</td>
<td>0.3441</td>
</tr>
<tr>
<td>(\hat{\theta}_4)</td>
<td>0.3182</td>
<td>(0.3166, 0.3198)</td>
<td>0.3139</td>
<td>0.3213</td>
</tr>
<tr>
<td>(\hat{\theta}_5)</td>
<td>0.0637</td>
<td>(0.0622, 0.0652)</td>
<td>0.0595</td>
<td>0.0687</td>
</tr>
<tr>
<td>(\hat{\theta}_6)</td>
<td>0.5891</td>
<td>(0.5742, 0.6040)</td>
<td>0.5485</td>
<td>0.6357</td>
</tr>
<tr>
<td>(\hat{\theta}_7)</td>
<td>0.1444</td>
<td>(0.1426, 0.1462)</td>
<td>0.1392</td>
<td>0.1483</td>
</tr>
<tr>
<td>(\hat{\theta}_8)</td>
<td>0.0630</td>
<td>(0.0623, 0.0637)</td>
<td>0.0618</td>
<td>0.0652</td>
</tr>
</tbody>
</table>

**Table:** Values computed for an ensemble of 30 independent runs of the FREM algorithm. Took 169 MCEM iterations to converge (runtime in the order of two days). In [Daigle et al., 2012], \(\hat{\theta}=(0.043, 0.538, 0.302, 0.377, 0.301, 3.103, 0.494, 0.243)\).
### Example 2: Auto-Regulatory Gene Network

<table>
<thead>
<tr>
<th>Interval</th>
<th>Method</th>
<th>M 500</th>
<th>M 1000</th>
<th>M 2000</th>
<th>M 4000</th>
</tr>
</thead>
<tbody>
<tr>
<td>[2.02, 2.52]</td>
<td>F-R</td>
<td>53</td>
<td>230</td>
<td>1040</td>
<td>4369</td>
</tr>
<tr>
<td></td>
<td>S</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>[22.22, 22.72]</td>
<td>F-R</td>
<td>23</td>
<td>160</td>
<td>705</td>
<td>2891</td>
</tr>
<tr>
<td></td>
<td>S</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>[44.44, 44.94]</td>
<td>F-R</td>
<td>780</td>
<td>2672</td>
<td>11427</td>
<td>48982</td>
</tr>
<tr>
<td></td>
<td>S</td>
<td>8</td>
<td>22</td>
<td>34</td>
<td>77</td>
</tr>
</tbody>
</table>

**Table:** Forward-Reverse and Shooting joined paths, with \( \theta = (0.1, 0.7, 0.35, 0.3, 0.1, 0.9, 0.2, 0.1) \).
Conclusions / Future work

- Extension to SRNs of the forward-reverse technique for expectations of functionals of bridges by [Bayer and Schoenmakers, 2014] with an application to the statistical problem for efficiently inferring coefficients of propensity functions.
- A two-phase Forward-Reverse Expectation-Maximization (FREM) algorithm.
- This method provides a clear computational work advantage over current shooting-like methods and others based on acceptance rejection techniques.
- Accelerated techniques for the EM algorithm to reduce the number of steps.
- Use higher-order kernels for higher dimensional problems.
- Instead of using the direct method to simulate paths, use an approximate one.
References


