Efficient stochastic simulation of systems with multiple time scales via statistical abstraction

Luca Bortolussi\textsuperscript{123}  Dimitrios Milios\textsuperscript{4}  Guido Sanguinetti\textsuperscript{45}

Modelling and Simulation Group, University of Saarland, Germany
Department of Mathematics and Geosciences, University of Trieste
CNR/ISTI, Pisa, Italy
School of Informatics, University of Edinburgh
SynthSys, Centre for Synthetic and Systems Biology, University of Edinburgh

16th of September 2015
Computational Methods in Systems Biology
Multiple Time-Scales in Biological Systems

The problem – Stiffness

- Existence of fast and slow time-scales
- Challenge to mathematical and computational treatment of systems

In the literature – Abstraction techniques

- Simplify some scales of the model
- Abstractions are non-trivial and model-specific

We propose:

- Model abstraction based on statistical methodologies
- Learned abstractions automatically from (few) exploratory runs of the models
Stochastic Simulation of Stiff Systems

The Gillespie algorithm is exact

- simulates every single reaction event
- High computational costs in presence of stiffness, where a small number of reactions dominate computations
The Gillespie algorithm is exact

- simulates every single reaction event
- High computational costs in presence of stiffness, where a small number of reactions dominate computations

**Enzyme-substrate example:**

\[
egin{align*}
E + S & \xrightarrow{f_1(\vec{X})} ES, \quad f_1(\vec{X}) = c_1 X_E X_S \\
ES & \xrightarrow{f_2(\vec{X})} E + S, \quad f_2(\vec{X}) = c_2 X_{ES} \\
ES & \xrightarrow{f_3(\vec{X})} E + P, \quad f_3(\vec{X}) = c_3 X_{ES}
\end{align*}
\]

Assuming \( c_1, c_2 \gg c_3 \):

- too many reaction events for \( R_1 \) and \( R_2 \),
- while \( R_3 \) progresses very slowly
Model Reduction

Reaction partitioning into \( R_{fast} \) and \( R_{slow} \):
  
  • based on their kinetic constants

System Variables: \( \vec{X} = (\vec{Y}, \vec{Z}) \)

**Fast Variables:** \( \vec{Y} = Y_1, \ldots, Y_m \)
  
  • Affected by either fast or slow reactions

**Slow Variables:** \( \vec{Z} = Z_1, \ldots, Z_s \)
  
  • Affected by slow reactions only
Model Reduction

Reaction partitioning into $\mathcal{R}_{\text{fast}}$ and $\mathcal{R}_{\text{slow}}$:
- based on their kinetic constants

System Variables: $\vec{X} = (\vec{Y}, \vec{Z})$

**Fast Variables:** $\vec{Y} = Y_1, \ldots, Y_m$
- Affected by either fast or slow reactions

**Slow Variables:** $\vec{Z} = Z_1, \ldots, Z_s$
- Affected by slow reactions only

**Enzyme-substrate example:**
We assume that $c_1, c_2 \gg c_3$
- fast and slow reactions: $\mathcal{R}_{\text{fast}} = \{R_1, R_2\}$ and $\mathcal{R}_{\text{slow}} = \{R_3\}$
- fast variables $\vec{Y} = (X_E, X_S, X_{ES})$ and slow variables $\vec{Z} = (X_P)$
The Fast Subsystem

System State $\vec{Y}$

- Affected by either $R_{fast}$ or $R_{slow}$
- Slow reactions rarely occur — can be ignored
- Fast rates may depend on the slow variables
The Fast Subsystem

System State $\vec{Y}$

- Affected by either $R_{fast}$ or $R_{slow}$
- Slow reactions rarely occur — can be ignored
- Fast rates may depend on the slow variables

Conditional Fast subsystem:

- Parametrised by the concentration $\vec{z}$ of slow variables
  - $\vec{z} = \vec{Z}/V$ in a volume $V$
The Fast Subsystem

System State $\vec{Y}$

- Affected by either $R_{fast}$ or $R_{slow}$
- Slow reactions rarely occur — can be ignored
- Fast rates may depend on the slow variables

Conditional Fast subsystem:

- Parametrised by the concentration $\vec{z}$ of slow variables
  - $\vec{z} = \vec{Z}/V$ in a volume $V$

\[
E + S \quad \xrightarrow{f_1(\vec{Y}, \vec{z})} \quad ES, \quad f_1(\vec{Y}, \vec{z}) = c_1 X_E (N - X_{ES} - X_P)
\]

\[
ES \quad \xrightarrow{f_2(\vec{Y}, \vec{z})} \quad E + S, \quad f_2(\vec{Y}, \vec{z}) = c_2 X_{ES}
\]

Assumption: Quickly reaches equilibrium for any $\vec{z}$
The Slow Subsystem

System State $\vec{Z}$

- Affected by $R_{slow}$
- Slow rates may depend on the fast variables
  - Senses the fast system only via its steady state distribution
The Slow Subsystem

System State $\vec{Z}$

- Affected by $\mathcal{R}_{slow}$
- Slow rates may depend on the fast variables
  - Senses the fast system only via its steady state distribution

All $R_j$ in $\mathcal{R}_{slow}$ are modified by:

1. removing the fast variables
2. replacing the rate function $f_j(\vec{Y}, \vec{z})$ by:

$$\hat{f}_j(\vec{z}) = \mathbb{E}_{\vec{z}}[f_j(\vec{Y}, \vec{z})]$$

Average out fast variables wrt their steady state distribution
The Slow Subsystem

System State $\bar{Z}$

- Affected by $\mathcal{R}_{slow}$
- Slow rates may depend on the fast variables
  - Senses the fast system only via its steady state distribution

All $R_j$ in $\mathcal{R}_{slow}$ are modified by:

1. removing the fast variables
2. replacing the rate function $f_j(\bar{Y}, \bar{z})$ by:

$$\hat{f}_j(\bar{z}) = \mathbb{E}_{\bar{z}}[f_j(\bar{Y}, \bar{z})]$$

Average out fast variables wrt their steady state distribution

$$\emptyset \xrightarrow{\hat{f}_3(\bar{z})} P, \quad \hat{f}_3(\bar{z}) = \mathbb{E}_{\bar{z}}[f_3(\bar{Y}, \bar{z})]$$
Simulation of the slow subsystem:

- Derive expectations $\hat{f}_j(\vec{z})$, $\forall R_j \in \mathcal{R}_{slow}$
- Fast reactions are ignored
Slow-scale Simulation

Simulation of the slow subsystem:
- Derive expectations $\hat{f}_j(\vec{z})$, $\forall R_j \in \mathcal{R}_{\text{slow}}$
- Fast reactions are ignored

In the literature:
- $\hat{f}_j(\vec{z})$ is given by model-dependent expressions
- Applicability is limited
- Required expertise on the modeller side
Slow-scale Simulation

Simulation of the slow subsystem:
- Derive expectations $\hat{f}_j(\vec{z})$, $\forall R_j \in \mathcal{R}_{slow}$
- Fast reactions are ignored

In the literature:
- $\hat{f}_j(\vec{z})$ is given by model-dependent expressions
- Applicability is limited
- Required expertise on the modeller side

A more generic approach:
- Construct a lookup table for the rate expectations
  - Explore the state-space of $\vec{Z}$
  - Estimate $\hat{f}_j(\vec{z})$ statistically
- **Problem:** The number of states for $\vec{Z}$ could be too large
Approximation of Rate Expectations

Theorem

The equilibrium statistics of the fast variables are a continuous function of the slow variables (rescaled to concentrations)

Our approach:

- Statistical estimate of the continuous function $\hat{f}_j(\vec{z})$
- Use a few samples from the slow state-space
- Interpolate via Gaussian Processes Regression
- Exhaustive state-space exploration is avoided
Gaussian Process Regression

- Place a GP prior over $f$
  \[ p(f) = \mathcal{N}(0, K) \]

- Assume noisy observations $y = f + \epsilon$
  \[ p(y \mid f) = \mathcal{N}(f, \sigma^2 I) \]
Gaussian Process Regression

- Place a GP prior over $f$
  \[ p(f) = \mathcal{N}(0, K) \]
- Assume noisy observations $y = f + \epsilon$
  \[ p(y \mid f) = \mathcal{N}(f, \sigma^2 I) \]

\[
p(f \mid y) = \frac{1}{Z} \underbrace{p(f)}_{\text{Gaussian Prior}} \underbrace{p(y \mid f)}_{\text{Gaussian Noise}}
\]
Initialisation Phase: For a grid of $n$ states of the slow process:

- Calculated rate expectations:

  \[ \hat{f}_j(z) = \frac{1}{t_f} \int_{t_0}^{t_0 + t_f} f_j(\tilde{Y}, \tilde{z}) dt \]

- $t_0$: time required to reach equilibrium (estimated by heuristic)
- Train a GP regression model
Stochastic Simulation via Statistical Abstraction
The SA-SSA Approach

**Initialisation Phase:** For a grid of $n$ states of the slow process:
- Calculated rate expectations:
  \[ \hat{f}_j(\vec{z}) = \frac{1}{t_f} \int_{t_0}^{t_0+t_f} f_j(\vec{Y}, \vec{z}) dt \]
  - $t_0$: time required to reach equilibrium (estimated by heuristic)
  - Train a GP regression model

**Simulation Phase:**
- Simulate the slow system (ignoring the fast variables/reactions)
- Using the rate expectations as given by the GP regression model
Cost of SA-SSA

**Pre-simulation Cost** (only during initialisation)
- Few samples of the slow system state-space
- Excessive simulation of the fast system is avoided

**Regression Cost** (only during initialisation)
- Dominated by the solution of a linear system — $O(n^2)$

**Cost of using the Analytical Approximation** (during simulation)
- Produce estimation from $n$ training points — $O(n)$
- For higher-dimensional slow state-spaces, sparse schemes are necessary

Note: Can learn rate expectations as functions of the system parameters
- approximate an entire family of stiff systems
Enzyme-substrate system — Parameter exploration

Let $c_1$ vary in the range $[0.01, 1]$

- The system remains stiff
- Sampled a grid of 1000 values for $X_P \in [0, 3000]$ and $c_1 \in [0.01, 1]$

**Table:** Relative mean error values for approximating the mean value of $X_P$, for $10^3$ simulation runs.

<table>
<thead>
<tr>
<th>Time</th>
<th>$c_1 = 0.01$</th>
<th>$c_1 = 0.1$</th>
<th>$c_1 = 0.5$</th>
<th>$c_1 = 1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$5 \times 10^4$</td>
<td>$1.83 \times 10^{-3}$</td>
<td>$9.08 \times 10^{-4}$</td>
<td>$2.35 \times 10^{-3}$</td>
<td>$2.17 \times 10^{-3}$</td>
</tr>
<tr>
<td>$10 \times 10^4$</td>
<td>$1.20 \times 10^{-3}$</td>
<td>$1.49 \times 10^{-3}$</td>
<td>$1.94 \times 10^{-3}$</td>
<td>$2.87 \times 10^{-3}$</td>
</tr>
<tr>
<td>$18 \times 10^4$</td>
<td>$8.04 \times 10^{-4}$</td>
<td>$3.73 \times 10^{-5}$</td>
<td>$4.49 \times 10^{-4}$</td>
<td>$3.05 \times 10^{-4}$</td>
</tr>
<tr>
<td>$20 \times 10^4$</td>
<td>$9.13 \times 10^{-4}$</td>
<td>$4.56 \times 10^{-5}$</td>
<td>$6.06 \times 10^{-5}$</td>
<td>$3.26 \times 10^{-5}$</td>
</tr>
</tbody>
</table>

**Gillespie algorithm:** 1911 sec

**SA-SSA:** 32 sec $+$ 3.562 sec for initialisation

The **Nested Stochastic Simulation Algorithm** (Nested-SSA) is proposed to approximate the steady-state of the fast subsystem
- The fast subsystem is only simulated up to a given step
  - .. assuming that steady-state is reached by then
- Completely transparent wrt the slow process

We have implemented Nested-SSA, to produce comparative results
- The step parameter for Nested-SSA has been explored experimentally such that the efficiency of both simulation approaches has been roughly the same
Enzyme-substrate system — Accuracy results

Initial state: \( \vec{X}_0 = (X_E, X_S, X_{ES}, X_P) = (220, 3000, 0, 0) \).

- The rate expectation for \( R_3 \) has been approximated via GP regression.
- Sampled 1000 states for the slow variable \( P \) between 0 and 3000.

**Table:** Enzyme-substrate model: histogram distances for \( 10^3 \) simulation runs (estimated self-distance: 0.252).

<table>
<thead>
<tr>
<th>Time</th>
<th>Nested-SSA</th>
<th>SA-SSA</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 ( \times ) 10^4</td>
<td>0.290</td>
<td>0.246</td>
</tr>
<tr>
<td>10 ( \times ) 10^4</td>
<td>0.250</td>
<td>0.204</td>
</tr>
<tr>
<td>18 ( \times ) 10^4</td>
<td>1.016</td>
<td>0.160</td>
</tr>
<tr>
<td>20 ( \times ) 10^4</td>
<td>0.940</td>
<td>0.142</td>
</tr>
</tbody>
</table>
Viral Infection model

Reactions: $\mathcal{R}_{\text{fast}} = \{R_3, R_5\}$ and $\mathcal{R}_{\text{slow}} = \{R_1, R_2, R_4, R_6\}$

Fast variables $\vec{Y} = (X_S)$, and slow variables $\vec{Z} = (X_G, X_T)$

- $\emptyset \xrightarrow{f_3(\vec{Y}, \vec{z})} S,$ \hspace{1em} $f_3(\vec{Y}, \vec{z}) = k_3 X_T$
- $S \xrightarrow{f_5(\vec{Y}, \vec{z})} \emptyset,$ \hspace{1em} $f_5(\vec{Y}, \vec{z}) = k_5 X_S$
- $T \xrightarrow{f_1(\vec{z})} G + T,$ \hspace{1em} $f_1(\vec{z}) = k_1 X_T$
- $G \xrightarrow{f_2(\vec{z})} T,$ \hspace{1em} $f_2(\vec{z}) = k_2 X_G$
- $T \xrightarrow{f_4(\vec{z})} \emptyset,$ \hspace{1em} $f_4(\vec{z}) = k_4 X_T$
- $G \xrightarrow{\hat{f}_6(\vec{z})} V,$ \hspace{1em} $\hat{f}_6(\vec{z}) = \mathbb{E}_{\vec{z}}[f_6(\vec{Y}, \vec{z})]$

The rate $\hat{f}_6(\vec{z})$ depends on $X_G$ directly, and on $X_T$ indirectly

- $T$ affects the steady-state of the fast process
Viral Infection model — Accuracy results

Random grid of 256 uniformly distributed population values for $G$ and $T$,
• given upper bounds of 500 and 100 molecules correspondingly
Naïve exploration of the rate expectation would require $50000$ evaluations

Table: Viral infection model: histogram distances for $10^3$ simulation runs
(estimated self-distance: 0.252).

<table>
<thead>
<tr>
<th>Time</th>
<th>$G$</th>
<th></th>
<th>$T$</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Nested-SSA</td>
<td>SA-SSA</td>
<td>Nested-SSA</td>
<td>SA-SSA</td>
</tr>
<tr>
<td>50</td>
<td>0.988</td>
<td>0.308</td>
<td>0.548</td>
<td>0.242</td>
</tr>
<tr>
<td>100</td>
<td>0.244</td>
<td>0.414</td>
<td>0.154</td>
<td>0.226</td>
</tr>
<tr>
<td>200</td>
<td>0.388</td>
<td>0.406</td>
<td>0.156</td>
<td>0.204</td>
</tr>
<tr>
<td>500</td>
<td>0.346</td>
<td>0.432</td>
<td>0.198</td>
<td>0.238</td>
</tr>
</tbody>
</table>
Viral Infection model — Accuracy results

Distribution of $X_G$ at $t = 50$
# Efficiency results

Table: Execution times in seconds for $10^3$ simulation runs.

<table>
<thead>
<tr>
<th>Method</th>
<th>Enzyme-substrate</th>
<th>Viral model</th>
</tr>
</thead>
<tbody>
<tr>
<td>SA-SSA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pre-simulation</td>
<td>0.291</td>
<td>26.11</td>
</tr>
<tr>
<td>Hyperparam. opt.</td>
<td>1.484</td>
<td>1.68</td>
</tr>
<tr>
<td>Training</td>
<td>0.080</td>
<td>0.05</td>
</tr>
<tr>
<td>Total initialisation</td>
<td>1.855</td>
<td>27.84</td>
</tr>
<tr>
<td>Simulation</td>
<td>153</td>
<td>316</td>
</tr>
<tr>
<td>Exact SSA</td>
<td>6947</td>
<td>2410</td>
</tr>
</tbody>
</table>
Conclusions

Time-scale separation

- In the literature: exploit structure to produce estimations for the rate expectations for the slow process
- We proposed SA-SSA: rate expectations are approximated via machine learning
- Learn the rate expectations as functions of the parameters as well
- Similar or better accuracy than Nested-SSA

Future Work

- Efficient simulation in presence of multiple spatio-temporal scales
- Abstraction of intra-cellular dynamics for cell population models
Acknowledgements...