MFBO-SSM: Multi-Fidelity Bayesian Optimization for Fast Inference in State-Space Models

Mahdi Imani
Texas A&M University
m.imani88@tamu.edu

Seyede Fatemeh Ghoreishi
Texas A&M University
f.ghoreishi88@tamu.edu

Douglas Allaire
Texas A&M University
dallaire@tamu.edu

Ulisses M. Braga-Neto
Texas A&M University
ulisses@ece.tamu.edu

Abstract

Nonlinear state-space models are ubiquitous in modeling real-world dynamical systems. Sequential Monte Carlo (SMC) techniques, also known as particle methods, are a well-known class of parameter estimation methods for this general class of state-space models. Existing SMC-based techniques rely on excessive sampling of the parameter space, which makes their computation intractable for large systems or tall data sets. Bayesian optimization techniques have been used for fast inference in state-space models with intractable likelihoods. These techniques aim to find the maximum of the likelihood function by sequential sampling of the parameter space through a single SMC approximator. Various SMC approximators with different fidelities and computational costs are often available for sample-based likelihood approximation. In this paper, we propose a multi-fidelity Bayesian optimization algorithm for the inference of general nonlinear state-space models (MFBO-SSM), which enables simultaneous sequential selection of parameters and approximators. The accuracy and speed of the algorithm are demonstrated by numerical experiments using synthetic gene expression data from a gene regulatory network model and real data from the VIX stock price index.

Introduction

Nonlinear state-space models are a popular class of time series models with numerous applications in fields such as statistics, economics, biology and more [1, 2, 3]. Sequential Monte Carlo (SMC) techniques, also known as particle methods [1, 4, 5], are the most well-known class of techniques for estimation of parameters of general nonlinear state space models from data. Several particle-based inference methodologies have been developed in recent years. The methods can be divided into two main categories: maximum-likelihood (ML) and Bayesian approaches. The techniques based on the ML perspective either try to maximize the regular ("incomplete") log-likelihood function [6] or the "complete" log-likelihood function; in the latter case, they are known as expectation maximization (EM) techniques [7, 8]. Bayesian techniques include particle marginal Metropolis-Hastings (PMMH) and particle-based Gibbs samplers [9, 10]. All aforementioned methods rely on excessive sampling of the parameter space to avoid local optimum traps. However, for large systems and tall data sets, the cost of approximation of the inference function per parameter sample point can be computationally expensive, rendering intractable the computation of existing particle-based techniques.

Several techniques employ Bayesian optimization to cope with intractable likelihood functions, mostly in the context of approximate Bayesian computing (ABC) [11, 12, 13]. The idea of these techniques is to construct a surrogate model representing the log-likelihood function over the parameter space, and efficiently search for its maximum through a single approximator (e.g., a particle filter). The speed and accuracy of these methods are directly impacted by the particle sample size used by the SMC approximator. Large particle sample sizes produce more accurate (high-fidelity) approximators at larger computational time/cost, while small particle sample sizes result in fast but less accurate (low-fidelity) approximators.

In this paper, we introduce the MFBO-SSM algorithm, a multi-fidelity Bayesian optimization method for the inference of general nonlinear state-space models. The proposed algorithm employs the *knowledge gradient* (KG) policy [14, 15, 16] for the simultaneous sequential selection of parameters and approximators, in such a way to achieve the largest single-period expected increase in the maximum of the inference function per unit cost.

The MFBO-SSM algorithm offers several benefits:

- Fast and accurate inference due to the efficient simultaneous sequential selection of parameters and approximators;
- Applicability to arbitrary point-based estimators, such as ML and MAP techniques;
- Possibility of considering risk in the inference process;
- Scalability to high-dimensional parameter spaces, due to the closed-form solution provided by the KG policy.

The accuracy and speed of the MFBO-SSM algorithm are demonstrated empirically by applying it to synthetic gene expression data from a gene regulatory network model and real data from the VIX stock price index.

Background

Nonlinear State-Space Models:

We assume the general nonlinear state-space model:

$$\begin{aligned} \mathbf{x}_k &= \mathbf{f}_k(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{n}_k, \theta)\,, \\ \mathbf{y}_k &= \mathbf{g}_k(\mathbf{x}_k, \mathbf{v}_k, \theta)\,, \end{aligned} \tag{1}$$

for $k = 1, 2, \ldots$ where $\mathbf{x}_k \in \mathbb{X}$ is the state variable, $\mathbf{u}_k \in \mathbb{U}$ is the input to the system, and $\mathbf{y}_k \in \mathbb{Y}$ is the output of the system. The nonlinear functions $\mathbf{f}_k(.)$ and $\mathbf{g}_k(.)$ model the state and measurement dynamics, which are assumed to be partially-known with the unknown parameter vector $\theta \in \Theta$, where Θ denotes the parameter space. Finally, $\{\mathbf{n}_k, \mathbf{v}_k; k = 1, 2 \ldots .\}$ are mutually independent i.i.d. processes, which are also independent of $\mathbf{x}_0$. The parameter vector θ models uncertainty in both state and measurement processes. Equivalently, $\mathbf{x}_k \sim p_\theta(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{u}_{k-1})$ and $\mathbf{y}_k \sim p_\theta(\mathbf{y}_k \mid \mathbf{x}_k)$, where $p_\theta(.)$ is a probability density or probability mass function. Without loss of generality and for the sake of simplicity, we will drop the input $\mathbf{u}_{k-1}$ in what follows.

The inference problem consists of estimating the parameter vector θ given the sequence of observed measurements $\mathbf{y}_{1:T} = (\mathbf{y}_1, ..., \mathbf{y}_T)$. The maximum-likelihood (ML) and the maximum a posteriori (MAP) estimators are given by:

$$\begin{aligned} \hat{\theta}^{\mathrm{ML}} &= \arg\max_{\theta \in \Theta} \log p_\theta(\mathbf{y}_{1:T})\,, \\ \hat{\theta}^{\mathrm{MAP}} &= \arg\max_{\theta \in \Theta} p(\theta \mid \mathbf{y}_{1:T}) \\ &= \arg\max_{\theta \in \Theta} \left[\log p(\theta) + \log p_\theta(\mathbf{y}_{1:T})\right]\,, \end{aligned} \tag{2}$$

where $p(\theta)$ denotes the prior distribution of the parameter. Both ML and MAP estimators require the exact computation of the data log-likelihood function:

$$\begin{aligned} L_T(\theta) &= \log p_\theta(\mathbf{y}_{1:T}) \\ &= \log p_\theta(\mathbf{y}_1) + \sum_{k=2}^{T} \log p_\theta(\mathbf{y}_k \mid \mathbf{y}_{1:k-1})\,, \end{aligned} \tag{3}$$

where

$$p_\theta(\mathbf{y}_k \mid \mathbf{y}_{1:k-1}) = \int p_\theta(\mathbf{y}_k \mid \mathbf{x}_k)\, p_\theta(\mathbf{x}_k \mid \mathbf{y}_{1:k-1})\, d\mathbf{x}_k, \tag{4}$$

and

$$p_\theta(\mathbf{x}_k \mid \mathbf{y}_{1:k-1}) = \int p_\theta(\mathbf{x}_k \mid \mathbf{x}_{k-1})\, p_\theta(\mathbf{x}_{k-1} \mid \mathbf{y}_{1:k-1})\, d\mathbf{x}_{k-1}. \tag{5}$$

The integrals in (4) and (5) need to be replaced by summations in the case of a discrete state space.

Sequential Monte Carlo: Auxiliary Particle Filter

The exact computation of (4) and (5) is not tractable in general and approximate methods, such as sequential Monte-Carlo (SMC), also known as particle filtering, must be used. SMC methods comprise a general class of techniques for inference of nonlinear state-space models [1, 4]. The idea of these techniques is to approximate the target distribution using a finite sample drawn from a proposal distribution, using the fact that sampling from the proposal distribution is easier than from the target. The basic algorithm to perform particle filtering is called *sequential importance resampling* (SIR). Here, we briefly review a variation of the SIR technique called the *Auxiliary Particle Filter* (APF) [17].

The APF is an SMC method that efficiently predicts the location of particles with high probability at time step k using information up to time step $k-1$ via an auxiliary variable ζ_k. The method first draws a sample of points (particles) from the joint distribution $p_\theta(\mathbf{x}_k, \zeta_k \mid \mathbf{y}_{1:k})$, then drops the auxiliary variable to obtain particles from $p_\theta(\mathbf{x}_k \mid \mathbf{y}_{1:k})$.

Let $\{\tilde{\mathbf{x}}_{k-1,i}, w_{k-1,i}\}_{i=1}^N$ be N particles and their associated weights at time $k-1$ approximating $p_\theta(\mathbf{x}_{k-1} \mid \mathbf{y}_{1:k-1})$. The process is divided into two stages. The first stage weights can be computed as:

$$v_{k,i} = p_\theta(\mathbf{y}_k \mid \mu_{k,i}) w_{k-1,i}\,, \tag{6}$$

for $i = 1, \ldots, N$; where $\mu_{k,i}$ is a characteristic of $\mathbf{x}_k$ given $\tilde{\mathbf{x}}_{k-1,i}$, which can be the mean, the mode or even a sample from $p_\theta(\mathbf{x}_k \mid \tilde{\mathbf{x}}_{k-1,i})$ [17]. The auxiliary variables $\{\zeta_{k,i}\}_{i=1}^N$ are obtained by sampling from a discrete distribution:

$$\{\zeta_{k,i}\}_{i=1}^N \sim \mathrm{Cat}(\{\tilde{v}_{k,i}\}_{i=1}^N)\,, \tag{7}$$

where $\{\tilde{v}_{k,i}\}_{i=1}^N$ are the normalized first-stage weights, and $\mathrm{Cat}(a_1, ..., a_N)$ represents a categorical distribution with probability mass function $f(\zeta = i) = a_i$. Finally, the new particles $\{\tilde{\mathbf{x}}_{k,i}\}_{i=1}^N$ and associated *second-stage* weights $\{w_{k,i}\}_{i=1}^N$ can be obtained as follows:

$$\tilde{\mathbf{x}}_{k,i} \sim p_\theta(\mathbf{x}_k \mid \tilde{\mathbf{x}}_{k-1,\zeta_{k,i}})\,,\; w_{k,i} = \frac{p_\theta(\mathbf{y}_k \mid \tilde{\mathbf{x}}_{k,i})}{p_\theta(\mathbf{y}_k \mid \mu_{k,\zeta_{k,i}})}\,. \tag{8}$$

It is shown in [18] that:

$$p_\theta(\mathbf{y}_k \mid \mathbf{y}_{1:k-1}) \approx \left(\frac{1}{N}\sum_{i=1}^N v_{k,i}\right)\left(\frac{1}{N}\sum_{i=1}^N w_{k,i}\right)\,, \tag{9}$$

where the above quantity can be used for approximating the log-likelihood function in (3).

Related Work

Particle-Based Maximum-Likelihood (ML) Techniques: Existing particle-based ML techniques for inference of general nonlinear state-space models can be divided into three main categories:

Direct Gradient-Based ML Techniques: The idea here is to maximize the log-likelihood function using gradient-ascent or quasi-Newton techniques [6]. These methods start by drawing an initial sample point from the parameter space, approximating the log-likelihood function and moving to another sample point based on the approximated gradient at the current sample. The computational complexity of approximating the log-likelihood is of order $O(N(T+1))$, where N is the number of particles and T is the length of the time series data. However, the unavoidable "resampling" step of particle filtering renders the approximated log-likelihood function discontinuous in θ even if the exact

log-likelihood function $L_T(\theta)$ is continuous [6]. Several importance-sampling methods have been introduced for approximation of the gradient function [19, 20, 21]. While some of these have computational complexity of order $O(N(T+1)\log N)$, successful methods are $O(N^2(T+1))$ [4, 6]. In addition, these techniques require extensive sampling of the parameter space to avoid local optimum traps.

Expectation-Maximization Techniques: Unlike direct ML techniques, which attempt to maximize the "incomplete" log-likelihood function $L_T(\theta) = \log p_\theta(\mathbf{y}_{1:T})$, expectation-maximization (EM) considers instead the "complete" log-likelihood function $\log p_\theta(\mathbf{x}_{0:T}, \mathbf{y}_{1:T})$. The logic behind this is that maximizing the complete log-likelihood is easier than the incomplete one. The EM algorithm thus consists of picking an initial guess $\theta = \theta^{(0)}$ and iterating two steps: 1) **E-Step:** Compute $Q(\theta, \theta^{(n)})$, 2) **M-Step:** Find $\theta^{(n+1)} = \text{argmax}_{\theta\in\Theta} Q(\theta, \theta^{(n)})$, where $Q(\theta, \theta^{(n)}) = E_{\mathbf{x}_{0:T}}\left[\log p_\theta(\mathbf{x}_{0:T}, \mathbf{y}_{1:T}) \mid \mathbf{y}_{1:T}, \theta^{(n)}\right]$. These steps need to be performed iteratively until a stopping criterion is met. Exact computation of the E-step is not possible for general nonlinear state-space models and one needs to use particle methods for its approximation. Two popular particle smoothers are the backward simulation smoother [22] and the reweighting particle smoother [23], which have lead to two different particle-based EM algorithms for general nonlinear state-space models introduced in [7] and [8] respectively. The computational complexity of both methods are of order $O(N^2(T+1))$. It should also be noted that the closed-form solution for the M-step might not be achievable in general, posing another expensive computation. Similar to direct ML techniques, this class of estimators requires several iterations to avoid local optimum traps.

Particle-Based Bayesian Techniques: There are several particle-based Bayesian techniques for the inference of general nonlinear state-space models. An important representative is the particle marginal Metropolis-Hastings (PMMH) method [9]. Given that θ is the current sample and $p_\theta(\mathbf{y}_{1:T})$ is the approximated likelihood by a particle filter (e.g., APF) associated with θ, one needs to draw a new sample parameter $\theta' \sim q(\theta' \mid \theta)$ from the proposal distribution and run a particle filter to approximate the likelihood $p_{\theta'}(\mathbf{y}_{1:T})$. Then, the new parameter θ' gets accepted with probability $\min\{1, p_{\theta'}(\mathbf{y}_{1:k})\, p(\theta')\, q(\theta \mid \theta') / p_\theta(\mathbf{y}_{1:k})\, p(\theta)\, q(\theta' \mid \theta)\}$. This process continues for a large enough (usually pre-specified) number of iterations in order to ensure good inference performance.

All the aforementioned techniques in the first two categories require extensive sampling of the parameter space to approximate the complete/incomplete log-likelihood function. For large systems, which require a large number of particles, and for tall data sets, the computational cost of approximating the log-likelihood function per parameter sample point can be prohibitive.

Surrogate-Based Techniques: This class of techniques has been developed for fast inference in SSMs with intractable likelihood functions [11, 12, 13]. The idea of these methods is to use Gaussian process regression for log-likelihood approximation, and apply Bayesian optimization techniques for efficient exploration of the maximum of the log-likelihood function using a single SMC approximator (i.e., a particle filter with a fixed particle sample size). The computational complexity of these techniques is of order $O(N(T+1))$ for each function evaluation. Despite the success of these techniques, their accuracy and speed are highly impacted by the choice of particle sample size. Large particle sample size N makes the inference process very slow, while improving accuracy. On the other hand, small particle sample size reduces accuracy, while accelerating the speed of the inference process. The main focus of the current paper is an efficient strategy for simultaneous parameter and approximator selection at each iteration of the learning process to achieve fast and accurate inference.

Proposed Algorithm

Modeling the Inference Function by a Gaussian Process:

We account for the correlation in the inference function by employing Gaussian process regression [24]. Let $z(\theta)$ be the inference function at parameter $\theta \in \Theta$ approximated by a particle filter with N particles associated with parameter θ (e.g., $z(\theta)$ refers to the approximate log-likelihood and log-a posteriori probability for ML and MAP techniques respectively). Note that $z(\theta)$ is a stochastic process, due to the uncertainty arising by the use of a particle filter.

The following model is considered here for the inference function:

$$z(\theta) \approx h(\theta) + \Delta h_N\,, \tag{10}$$

where $h(\theta)$ is a Gaussian process (GP) over the parameter space Θ, and Δh_N is a zero-mean Gaussian residual with variance σ_N^2, which models, for all parameters, the uncertainty arising from the use of a particle filter with N particles. Large particle sample sizes N correspond to smaller σ_N^2.

The following prior distribution is assumed for the GP:

$$h(\theta) = \mathcal{G}P\left(\mu(\theta), k(\theta,\theta)\right), \tag{11}$$

where $\mu(\theta)$ denotes the mean and $k(.,.)$ is a real-valued kernel function, which encodes our prior belief on the correlation between θ and θ'. A common kernel choice for a continuous parameter space is the well-known exponential kernel function [24].

Let $\boldsymbol{\theta}_m = (\theta^{(1)}, \ldots, \theta^{(m)})$ be a sample from the parameter space, with the approximated inference function computed by running m particle filters with particle sample sizes $\mathbf{n}_m = (N_1, ..., N_m)$, and evaluated objective function $\mathbf{z}_m = [z(\theta^{(1)}), \ldots, z(\theta^{(m)})]^T$. The posterior distribution of $h(\theta)$ in equation (11) can be obtained as [24]:

$$h(\theta) \mid \boldsymbol{\theta}_m, \mathbf{n}_m, \mathbf{z}_m \sim \mathcal{N}\left(\bar{h}_m(\theta), \text{cov}_m\left(\theta,\theta\right)\right), \tag{12}$$

where

$$\begin{aligned}\bar{h}_m(\theta) &= \mu(\theta) + \mathbf{K}_{\theta,\boldsymbol{\theta}_m}\left(\mathbf{K}_{\boldsymbol{\theta}_m,\boldsymbol{\theta}_m} + \boldsymbol{\Sigma}_{\mathbf{n}_m}\right)^{-1}\left(\mathbf{z}_m - \mu(\boldsymbol{\theta}_m)\right),\\ \text{cov}_m(\theta,\theta) &= k(\theta,\theta) - \mathbf{K}_{\theta,\boldsymbol{\theta}_m}\left(\mathbf{K}_{\boldsymbol{\theta}_m,\boldsymbol{\theta}_m} + \boldsymbol{\Sigma}_{\mathbf{n}_m}\right)^{-1}\mathbf{K}_{\theta,\boldsymbol{\theta}_m}^T,\end{aligned} \tag{13}$$

$\mathbf{\Sigma}_{\mathbf{n}_m}$ is a diagonal matrix of size m with ith diagonal element $(\mathbf{\Sigma}_{\mathbf{n}_m})_{ii} = \sigma^2_{N_i}$, and

$$\mathbf{K}_{\boldsymbol{\theta},\boldsymbol{\theta}'} = \begin{bmatrix} k(\theta_1,\theta'_1) & \dots & k(\theta_1,\theta'_n) \\ \vdots & \ddots & \vdots \\ k(\theta_l,\theta'_1) & \dots & k(\theta_l,\theta'_n) \end{bmatrix}, \qquad (14)$$

for $\boldsymbol{\theta} = \{\theta_1, ..., \theta_l\}, \boldsymbol{\theta}' = \{\theta'_1, ..., \theta'_n\}$. Using the above formulation, the inference function before observing any data is modeled by a zero-mean Gaussian process with covariance $k(\theta,\theta)$, while at iteration m, the inference function is predicted based on the sequence of queried samples $\boldsymbol{\theta}_m$, the approximate inference function values $\mathbf{z}_m$, and the particle sample sizes $\mathbf{n}_m$ used for these approximations. The uncertainty in the inference function, which is modeled by the covariance function in equation (12), decreases as more points are sampled from the parameter space and added to the GP.

The hyperparameters of the Gaussian process, such as the parameters of the kernel function or the mean function, can be estimated at each time point using the marginal likelihood function [24]:

$$\mathbf{z}_m \mid \boldsymbol{\theta}_m, \mathbf{n}_m \sim \mathcal{N}\left(\mu(\boldsymbol{\theta}_m), \mathbf{K}_{\boldsymbol{\theta}_m,\boldsymbol{\theta}_m} + \mathbf{\Sigma}_{\mathbf{n}_m}\right). \qquad (15)$$

Notice that, due to the difficulty of choosing a proper model for the mean function and its impact on the inference accuracy, a possible option is to use $\mu(\theta) = \min_{i=1,...,m} \mathbf{z}_m(i)$. This adaptive constant mean avoids the challenging task of picking a proper parametric model for the mean function, and also prevents over-estimation of the objective function over regions that have not been explored well.

Simultaneous Sequential Selection of Parameter and Particle Sample Size:

To boost the speed of the inference process and overcome the computational intractability of existing techniques, we introduce a multi-fidelity Bayesian optimization algorithm for the inference of general nonlinear state-space models (MFBO-SSM).

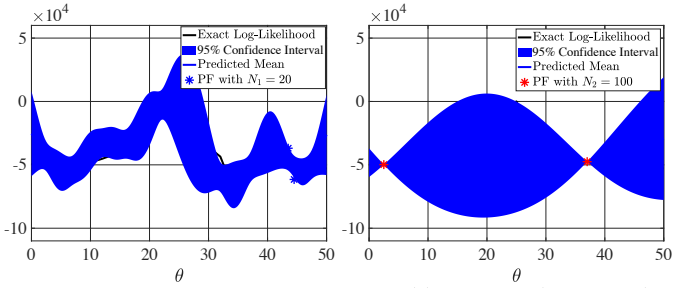
Let σ^2_N be the variance of the objective function approximated by a particle filter with N particles. The value of the noise statistics σ^2_N mainly depends on the particle sample size N for the log-likelihood approximation. This value can be quantified in three possible ways: 1) running multiple particle filters with a fixed particle sample size at an arbitrary sample of the parameter space and computing the sample variance of the approximated log-likelihood values; 2) using available theoretical upper bounds on the approximation error of a particle filter with a fixed particle sample size [25, 26]; 3) treating the noise parameters as hyperparameters and learning them on the fly according to the marginal likelihood in (15).

The variance issue is linked to the computational complexity c_N of particle filtering algorithms, which increases linearly with the number of particles. Thus, large N corresponds to an approximator with smaller variance (high- fidelity) and high computational complexity, whereas small N models an approximator with large uncertainty (low-fidelity) but low computational complexity.

To better understand the intuition behind the proposed algorithm, let us consider a simple nonlinear continuous state-space model [5, 22]:

$$\begin{aligned} x_{k+1} &= 0.5\,x_k + \theta\frac{x_k}{1+x_k^2} + 8\cos(1.2k) + n_k\,, \\ y_k &= 0.05x_k^2 + v_k, \end{aligned} \qquad (16)$$

where $n_k \sim \mathcal{N}(0, 0.001), v_k \sim \mathcal{N}(0, 0.01)$, and θ is the only parameter of the system, with true value $\theta^* = 25$. For a time series of length 100, the exact log-likelihood is plotted in black in Figure 1. The Gaussian process approximated by 10 sample points from the parameter space with particle sample size $N_1 = 20$ is plotted in Figure 1(a). This low-fidelity approximator has variance $\sigma_{N_1} = 10{,}000$. It can be seen that 10 sample points from the parameter space have properly captured the log-likelihood function with this approximator. Figure 1(b) displays the constructed GP with $N_2 = 100$ and two sample points from the parameter space. For this high-fidelity approximator, $\sigma_{N_2} = 500$. These two approximations have the same computational complexity, since $c_{N_1}/c_{N_2} = N_1/N_2 = 20/100$. However, the mean of the GP corresponding to the high-fidelity approximator is far from the exact log-likelihood function.



(a) $N_1 = 20$ ($\sigma_{N_1} = 10{,}000$) (b) $N_2 = 100$ ($\sigma_{N_2} = 500$)

Figure 1: GP approximation for the log-likelihood function using a) 10 parameter samples and a low-fidelity particle filter with $N_1 = 20$, and b) 2 parameter samples and a high-fidelity particle filter with $N_2 = 100$, for the system in (16).

We conclude from the previous example that, in order to achieve a solution that is both accurate and fast, one needs to select simultaneously good sample points in the parameter space and good approximators of the log-likelihood function during the learning process. Finding an optimal (finite or infinite horizon) strategy is a challenging task. In the next paragraphs, we describe how a Bayesian optimization framework, in particular the knowledge gradient policy [14, 15], can be employed for tackling this problem.

Let $\boldsymbol{\theta}_m = (\theta^{(1)}, ..., \theta^{(m)})$ be a sample from the parameter space with associated approximated inference function values $\mathbf{z}_m = [z(\theta^{(1)}), ..., z(\theta^{(m)})]^T$ computed by particle filters with particle sample sizes $\mathbf{n}_m = (N_1, ..., N_m)$. By constructing a GP given all available information up to iteration m, a given sample point $\theta \in \Theta$ has expected inference function $\bar{h}_m(\theta) = \mathbb{E}[h(\theta) \mid \boldsymbol{\theta}_m, \mathbf{n}_m, \mathbf{z}_m]$, according to (13), and

$$\hat{\theta}^*_{\mathrm{GP}} = \operatorname*{argmax}_{\theta\in\Theta} \bar{h}_m(\theta). \qquad (17)$$

If an additional pair of parameter and approximator is to be selected from the parameter space and set of approximators, we would like to choose the pair with the highest single-period expected increase in the maximum of the inference function per unit cost. This policy, which is referred to as a multi-fidelity knowledge gradient policy, can be formulated as:

$$(\theta^{(m+1)}, N_{m+1}) = \underset{(\theta,N)\in(\Theta,\mathbb{N})}{\operatorname{argmax}} \frac{1}{c_N}$$
$$\mathbb{E}_m \left[\max_{\theta'\in\Theta} \mathbb{E}\left[h(\theta') \mid \boldsymbol{\theta}_m, \mathbf{n}_m, \mathbf{z}_m, \theta^{(m+1)} = \theta, N_{m+1} = N \right] - \max_{\theta'\in\Theta} \mathbb{E}\left[h(\theta') \mid \boldsymbol{\theta}_m, \mathbf{n}_m, \mathbf{z}_m \right] \right], \tag{18}$$

where $\mathbb{N}$ denotes a finite set of particle sample sizes corresponding to a finite number of approximators, and $\mathbb{E}_m$ denotes expectation over the unobserved inference function at point $\theta^{(m+1)}$ approximated by a particle filter (approximator) with N_{m+1} particles, given all available information up to iteration m.

Exact computation of (18) is not possible over an infinite parameter space. However, given a finite set of alternatives $\mathbb{A} \subset \Theta$, we can write the approximation:

$$(\theta^{(m+1)}, N_{m+1}) = \underset{(\theta,N)\in(\mathbb{A},\mathbb{N})}{\operatorname{argmax}} \frac{1}{c_N} \mathrm{EI}\,(\theta, N), \tag{19}$$

where

$$\mathrm{EI}\,(\theta, N) =$$
$$\mathbb{E}_m \left[\max_{\theta'\in\mathbb{A}} \mathbb{E}\left[h(\theta') \mid \boldsymbol{\theta}_m, \mathbf{n}_m, \mathbf{z}_m, \theta^{(m+1)} = \theta, N_{m+1} = N \right] - \max_{\theta'\in\mathbb{A}} \mathbb{E}\left[h(\theta') \mid \boldsymbol{\theta}_m, \mathbf{n}_m, \mathbf{z}_m \right] \right], \tag{20}$$

for $\theta \in \mathbb{A}$ and $N \in \mathbb{N}$. The knowledge gradient algorithm [14, 15] provides a closed-form solution for computation of the expected increase $\mathrm{EI}\,(\theta, N)$ in the maximum of the objective function in (20). The feature of the knowledge gradient policy to account for the uncertainty in the log-likelihood function approximation is a reason of choosing this acquisition function over other Bayesian optimization techniques, such as expected improvement [27] and entropy search [28].

The finite set of alternatives $\mathbb{A} \subset \Theta$ should be selected based on the goal of the inference process. In particular, the set of alternatives for ML estimation can be obtained using hypercube sampling [29], whereas for MAP estimation, the alternative set could be provided by a sample drawn from the prior distribution. The set of particle sample sizes (number of approximators) needs to be chosen based on the size of the system and the amount of data. However, the algorithm is fairly robust against this choice: in our numerical experiments, we observed that "small", "medium" and "large" particle sample sizes all lead to good inference accuracy and speed.

After selection of $\theta^{(m+1)}$ and N_{m+1} using (19), a particle filter with N_{m+1} particles associated with parameter $\theta^{(m+1)}$ is run and the Gaussian process is updated based on $\boldsymbol{\theta}_{m+1} = (\boldsymbol{\theta}_m, \theta^{(m+1)})$, $\mathbf{n}_{m+1} = (\mathbf{n}_m, N_{m+1})$, and $\mathbf{z}_{m+1} = [\mathbf{z}_m, z(\theta^{(m+1)})]^T$. This procedure continues until a stopping criterion is met, which might be a threshold for the change in the maximum of the mean of the constructed GP in consecutive iterations, or a pre-specified limit on the number of iterations. It should be noted that the complexity of the MFBO-SSM algorithm at iteration m is approximately $O(\max\{N_{m+1}(T+1), m^3\})$, where N_{m+1} is the number of particles used for the objective function approximation, and T is the length of the time series data.

The Gaussian processes constructed over the log-likelihood function for the system in (16) at different iterations of the proposed algorithm are plotted in Figure 2. It can be seen that all 9 initial sample parameters prescribed by MFBO-SSM are from the low-fidelity approximator ($N_1 = 20$), but the last three sample parameters are evaluated by the high fidelity approximator ($N_2 = 100$). Indeed, the low-fidelity approximator with 5 times less computational complexity has been initially used for proper exploration of the parameter space. Then, as the expected increase in the maximum of the inference function per unit cost using the use of first approximator becomes small, the high-fidelity approximator is performed for proper exploitation. As we will see in the next section, this key feature of the MFBO-SSM algorithm results in an inference process that is both fast and accurate, as compared to the existing techniques.

Algorithm 1 MFBO-SSM Algorithm

1: Set the alternative set $\mathbb{A}$, and the particle set $\mathbb{N}$, the cost c_N and variance σ_N^2 of the particle filter with $N \in \mathbb{N}$ particles.
2: Construct a GP over parameter θ.
3: $m = -1$, $\boldsymbol{\theta}_0 = \{\}$, $\mathbf{n}_0 = \{\}$, $\mathbf{z}_0 = \{\}$.
While stopping criterion is not met
4: $m = m + 1$.
5: $(\theta^{(m+1)}, N_{m+1}) = \operatorname{argmax}_{(\theta,N)\in(\mathbb{A},\mathbb{N})} \frac{1}{c_N} \mathrm{EI}\,(\theta, N)$ where $\mathrm{EI}\,(\theta, N)$ is defined in Eq. (20).
6: Run a particle filter tuned to $\theta^{(m+1)}$ with particle size N_{m+1} to get $z(\theta^{(m+1)})$.
7: $\boldsymbol{\theta}_{m+1} = \{\boldsymbol{\theta}_m, \theta^{(m+1)}\}$, $\mathbf{n}_{m+1} = \{\mathbf{n}_m, N_{m+1}\}$, $\mathbf{z}_{m+1} = \{\mathbf{z}_m, z(\theta^{(m+1)})\}$.
8: Update GP according to ($\boldsymbol{\theta}_{m+1}$, $\mathbf{n}_{m+1}$, $\mathbf{z}_{m+1}$).
End While
9: $\hat{\theta}^*_{\mathrm{GP}} = \operatorname{argmax}_{\theta\in\Theta} \bar{h}_m(\theta)$, where $\bar{h}_m(\theta)$ is the mean of the final GP.

Experiments

All experiments have been conducted on a PC with an Intel Core i7-4790 CPU@3.60-GHz clock and 16 GB of RAM.

Stochastic volatility model: To assess the performance of the proposed algorithm in the presence of tall data, the time-varying stochastic volatility model is considered. This model, which is often used as a benchmark for inference of nonlinear state-space models, describes the behavior of

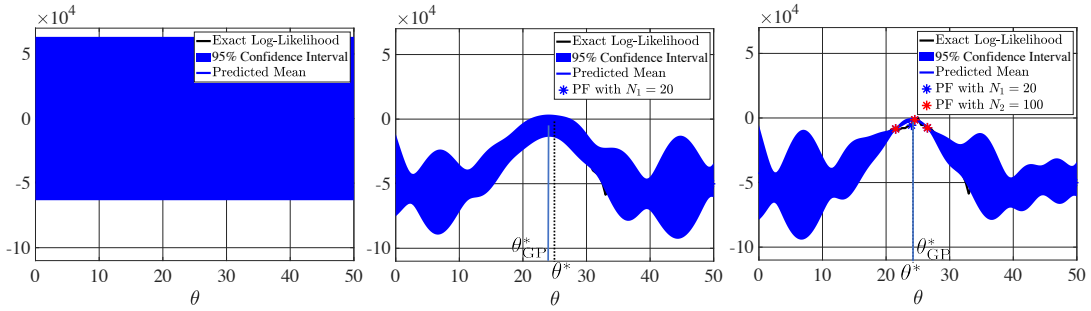


Figure 2: Constructed GP and estimated parameter at different iterations of MFBO-SSM for the system in (16).

moderate to high-frequency financial data [30, 31]. The state and observation processes of this model are:

$$\begin{aligned} x_k &= \phi x_{k-1} + (1-\phi)\mu + \sigma n_k, \quad n_k \sim \mathcal{N}(0,1) \\ y_k &= \beta \exp(x_k/2)\, v_k, \quad v_k \sim \mathcal{N}(0,1). \end{aligned} \tag{21}$$

The performance of the MFBO-SSM algorithm is assessed using both synthetic and real data. For synthetic data, 100 time series of length 4000 are drawn from the model in (21) with initial state $x_0 \sim \mathcal{N}(0, \sigma^2/1-\phi^2)$. While there is only a single state variable, the time-varying process necessitates the use of a large number of particles. MFBO-SSM algorithm uses $N_1 = 100$, $N_2 = 1000$, $N_3 = 5000$, corresponding to "small," "medium," and "large" particle sample sizes. Other methods use a fixed particle sample size $N = 1000$. We are interested in estimating the true parameter $\theta^* = (\sigma^*, \phi^*, \beta^*, \mu^*) = (0.97, 0.55, 0.95, 0.1)$ from synthetic data, where $\Theta = [0,2] \times [-1,1] \times [0,10] \times [0,5]$. The results of the MFBO-SSM algorithm are compared with the Bayesian optimization (expected improvement) method with fixed particle sample size proposed in [11], the PMMH algorithm [9], a particle-based ML algorithm [6], and particle-based EM algorithms [8]. A Gaussian proposal distribution is used with the PMMH algorithm. The MFBO-SSM, BO, EM, and ML algorithms all stop when the change in the estimated value of all parameters over a window of length 20 falls bellow 5% of their range, whereas the PMMH algorithm continues over a fixed number of 6,000 iterations. Figure 3 displays the average MSE of estimation of the different parameters against running time in minutes. One can observe that the accuracy at the same speed achieved by MFBO-SSM is significantly higher than for the other methods.

Real data with length 3273 from the VIX stock price index recorded between March 2005 and March 2018 have also been used in our analysis. The data are displayed in the left panel of Figure 4. The average maximum of the log-likelihood function against running time is displayed in the right panel of Figure 4. One can observe that the log-likelihood is maximized faster by the MFBO-SSM algorithm than by the other methods. The average number of particles used by the MFBO-SSM algorithm against iteration number is displayed in Figure 5. It can be seen that the MFBO-SSM algorithm picks the low-fidelity approximator (i.e., the particle filter with $N_1 = 100$) most of the time at early iterations, for cheap exploration, while selecting the two other expensive approximators with particle sample sizes $N_2 = 1000$ and $N_3 = 5000$ only at later iterations, for better exploitation.

Cell-Cycle Gene Regulatory Network Example: In the second experiment, we consider the cell cycle gene regulatory network model in [32]. This is a Boolean network consisting of 14 genes, where each gene can be activated or inactivated. Hence, there are $2^{14} = 16384$ different possible system states. The pathway diagram for this gene regulatory network is displayed in Figure 6. Normal arrows represent activating regulations and dash arrows represent suppressive regulations. With vector $\mathbf{x}_k$ containing the expression state of all 14 genes at time k, the Boolean state process can be written as:

$$\mathbf{x}_k = \overline{A\,\mathbf{x}_{k-1}} \oplus \mathbf{n}_k\,, \tag{22}$$

where $\overline{\mathbf{v}}$ maps the positive elements of vector $\mathbf{v}$ to 1 and others to 0, $\mathbf{n}_k$ is the process noise, and $A = [a_{ij}]$ is the connectivity matrix. Parameter a_{ij} specifies the type of regulation from gene j to gene i: it is equal to $+1$ for the activating regulation, -1 for the inactivating regulation and 0 for no regulation. The process noise $\mathbf{n}_k$ is assumed to have independent components distributed as Bernoulli(p), where the noise parameter p gives the amount of "perturbation" to the Boolean state process. In our simulation, we set $p = 0.01$.

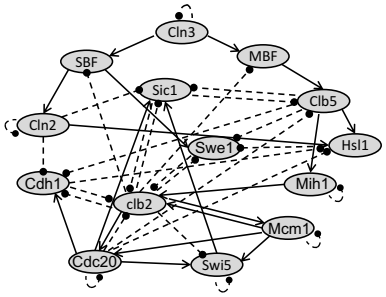


Figure 6: Pathway diagram for the cell-cycle gene regulatory network model.

We assume a Gaussian linear observation model:

$$\mathbf{y}_k = \boldsymbol{\mu} + D\mathbf{x}_k + \mathbf{v}_k\,, \quad k = 1, 2, \ldots \tag{23}$$

where $\mathbf{v}_k \sim \mathcal{N}(0, \sigma^2 I)$ is an uncorrelated zero-mean Gaussian noise vector, $\boldsymbol{\mu}$ is a vector of baseline gene expressions (corresponding to the "zero" state for each gene) and D is a diagonal matrix containing differential expression values for

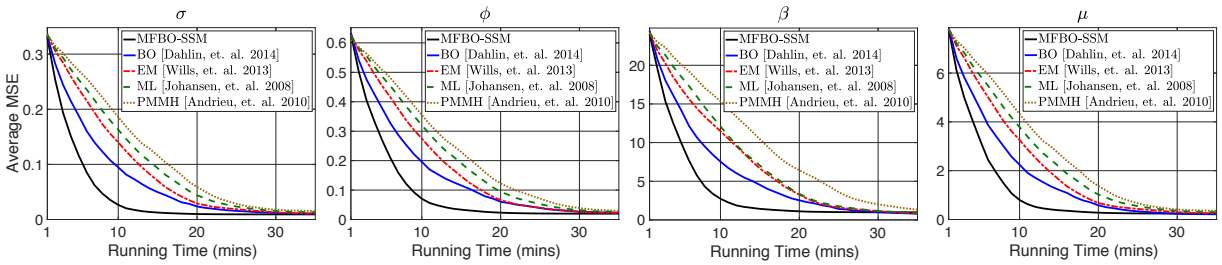


Figure 3: Average MSE against running time in minutes for different inference algorithms using synthetic data from the stochastic volatility model.

Table 1: Results for the cell-cycle gene regulatory network.

Average MSE (running time per minutes)					
N	BO [11]	EM [8]	ML [6]	PMMH [9]	MFBO-SSM
20,000	9.412 (88.15)	9.473 (166.98)	10.238 (199.39)	10.991 (354.45)	6.904 (35.59)
40,000	6.902 (189.24)	6.862 (341.92)	7.827 (418.09)	7.618 (722.90)	

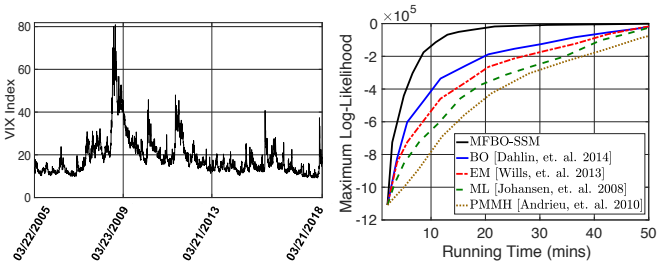


Figure 4: Daily return data for the VIX stock price index and average maximum log-likelihood against running time in minutes for different inference algorithms using these data.

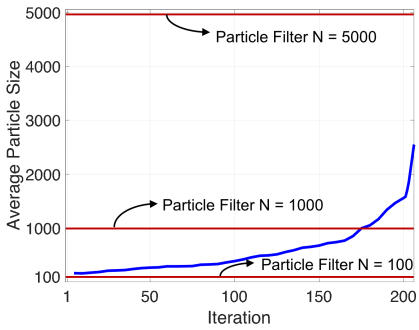


Figure 5: Average number of particles used by the MFBO-SSM algorithm against iteration number for the daily return data for the VIX stock price index.

each gene along the diagonal (these indicate by how much the activated state of each gene is over-expressed over the inactivated state). Such a Gaussian linear model is an appropriate model for many important gene-expression measurement technologies, such as cDNA microarrays [33] and live cell imaging-based assays [34]. Here, we assume that $\boldsymbol{\mu} = [\mu, \ldots, \mu]^T$ and $D = \delta I$, so that the parameter vector is $\theta = (\mu, \delta, \sigma)$. The true value of the parameter is $\theta^* = (30, 20, 10)$ and the parameter space is assumed to be $\Theta = [5, 40] \times [5, 40] \times [5, 20]$. The prior distribution is uniform over Θ. Given 100 time series of length 100, the sum of the average MSE of estimation for all parameters and running time for different algorithms are displayed in Table 1. Three approximators with $N_1 = 1{,}000$, $N_2 = 10{,}000$ and $N_3 = 20{,}000$ are used with the MFBO-SSM algorithm, whereas the particle sample size $N = 10{,}000$ is used with the other methods. The MFBO-SSM, EM and ML algorithms stop when the change in the estimated value of all parameters over a window of 20 consecutive iterations falls bellow 5% of their range, whereas the PMMH algorithm runs over a fixed number of 10,000 iterations.

In Table 1, we can observe that for both particle sample sizes, the MFBO-SSM algorithm achieves good accuracy with a much smaller running time than the other algorithms. Indeed, MFBO-SSM is 2.5 times faster than the fastest competitor for $N = 20{,}000$ and 5 times faster for $N = 40{,}000$. These results demonstrate the ability of MFBO-SSM in speeding up inference for both large systems and tall data sets, at similar or better accuracy levels.

Conclusion

In this paper, we introduced the MFBO-SSM algorithm for fast and accurate inference of parameters of nonlinear state-space models. The proposed algorithm can handle large system sizes and tall data sets. MFBO-SSM alleviates the computational expense associated with sample-based approximations of the inference function by constructing a Gaussian process for modeling the correlation in the inference function, and learns the maximum of this surrogate model by simultaneous selection of sample parameters and particle sample sizes for its SMC approximation. In numerical experiments using real and synthetic data, the proposed al-

gorithm performed significantly faster than competing algorithms, at similar or better accuracy levels.

References

[1] D. Arnaud, N. de Freitas, and N. Gordon, *Sequential Monte Carlo methods in practice*. New York: Springer Science+ Business Media, Inc. LLC, 2001.

[2] R. J. Elliott, L. Aggoun, and J. B. Moore, *Hidden Markov models: estimation and control*, vol. 29. Springer Science & Business Media, 2008.

[3] O. Cappé, E. Moulines, and T. Rydén, *Inference in Hidden Markov Models*. Springer, 2005.

[4] N. Kantas, A. Doucet, S. S. Singh, J. Maciejowski, and N. Chopin, "On particle methods for parameter estimation in state-space models," *Statistical science*, vol. 30, no. 3, pp. 328–351, 2015.

[5] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statistics and computing*, vol. 10, no. 3, pp. 197–208, 2000.

[6] A. M. Johansen, A. Doucet, and M. Davy, "Particle methods for maximum likelihood estimation in latent variable models," *Statistics and Computing*, vol. 18, no. 1, pp. 47–57, 2008.

[7] T. B. Schön, A. Wills, and B. Ninness, "System identification of nonlinear state-space models," *Automatica*, vol. 47, no. 1, pp. 39–49, 2011.

[8] A. Wills, T. B. Schön, L. Ljung, and B. Ninness, "Identification of hammerstein–wiener models," *Automatica*, vol. 49, no. 1, pp. 70–81, 2013.

[9] C. Andrieu, A. Doucet, and R. Holenstein, "Particle Markov chain Monte Carlo methods," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 72, no. 3, pp. 269–342, 2010.

[10] F. Lindsten, M. I. Jordan, and T. B. Schön, "Particle Gibbs with ancestor sampling," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 2145–2184, 2014.

[11] J. Dahlin, M. Villani, and T. B. Schön, "Bayesian optimisation for fast approximate inference in state-space models with intractable likelihoods," *arXiv preprint arXiv:1506.06975*, 2015.

[12] J. Dahlin and F. Lindsten, "Particle filter-based Gaussian process optimisation for parameter inference," *arXiv preprint arXiv:1311.0689*, 2013.

[13] M. U. Gutmann and J. Corander, "Bayesian optimization for likelihood-free inference of simulator-based statistical models," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 4256–4302, 2016.

[14] P. I. Frazier, W. B. Powell, and S. Dayanik, "A knowledge-gradient policy for sequential information collection," *SIAM Journal on Control and Optimization*, vol. 47, no. 5, pp. 2410–2439, 2008.

[15] P. Frazier, W. Powell, and S. Dayanik, "The knowledge-gradient policy for correlated normal beliefs," *INFORMS journal on Computing*, vol. 21, no. 4, pp. 599–613, 2009.

[16] W. B. Powell and I. O. Ryzhov, *Optimal learning*, vol. 841. John Wiley & Sons, 2012.

[17] M. K. Pitt and N. Shephard, "Filtering via simulation: Auxiliary particle filters," *Journal of the American statistical association*, vol. 94, no. 446, pp. 590–599, 1999.

[18] M. K. Pitt, *Smooth particle filters for likelihood evaluation and maximisation*. University of Warwick, Department of Economics, 2002.

[19] D. N. DeJong, R. Liesenfeld, G. V. Moura, J.-F. Richard, and H. Dharmarajan, "Efficient likelihood evaluation of state-space representations," *Review of Economic Studies*, vol. 80, no. 2, pp. 538–567, 2012.

[20] S. Malik and M. K. Pitt, "Particle filters for continuous likelihood evaluation and maximisation," *Journal of Econometrics*, vol. 165, no. 2, pp. 190–209, 2011.

[21] E. L. Ionides, C. Bretó, and A. A. King, "Inference for nonlinear dynamical systems," *Proceedings of the National Academy of Sciences*, vol. 103, no. 49, pp. 18438–18443, 2006.

[22] S. J. Godsill, A. Doucet, and M. West, "Monte Carlo smoothing for nonlinear time series," *Journal of the american statistical association*, vol. 99, no. 465, pp. 156–168, 2004.

[23] M. Hürzeler and H. R. Künsch, "Monte Carlo approximations for general state-space models," *Journal of Computational and Graphical Statistics*, vol. 7, no. 2, pp. 175–193, 1998.

[24] C. E. Rasmussen and C. Williams, *Gaussian processes for machine learning*. MIT Press, 2006.

[25] N. Whiteley, "Stability properties of some particle filters," *The Annals of Applied Probability*, vol. 23, no. 6, pp. 2500–2537, 2013.

[26] J. Olsson and T. Rydén, "Asymptotic properties of particle filter-based maximum likelihood estimators for state space models," *Stochastic Processes and their Applications*, vol. 118, no. 4, pp. 649–680, 2008.

[27] D. R. Jones, "A taxonomy of global optimization methods based on response surfaces," *Journal of global optimization*, vol. 21, no. 4, pp. 345–383, 2001.

[28] J. M. Hernández-Lobato, M. W. Hoffman, and Z. Ghahramani, "Predictive entropy search for efficient global optimization of black-box functions," in *Advances in neural information processing systems*, pp. 918–926, 2014.

[29] M. D. McKay, R. J. Beckman, and W. J. Conover, "Comparison of three methods for selecting values of input variables in the analysis of output from a computer code," *Technometrics*, vol. 21, no. 2, pp. 239–245, 1979.

[30] S. Chib, F. Nardari, and N. Shephard, "Markov chain Monte Carlo methods for stochastic volatility models," *Journal of Econometrics*, vol. 108, no. 2, pp. 281–316, 2002.

[31] Y. Aı and R. Kimmel, "Maximum likelihood estimation of stochastic volatility models," *Journal of financial economics*, vol. 83, no. 2, pp. 413–452, 2007.

[32] E. Radmaneshfar and M. Thiel, "Recovery from stress–a cell cycle perspective," *Journal of computational interdisciplinary sciences*, vol. 3, no. 1-2, p. 33, 2012.

[33] Y. Chen, E. R. Dougherty, and M. L. Bittner, "Ratio-based decisions and the quantitative analysis of cDNA microarray images," *Journal of Biomedical optics*, vol. 2, no. 4, pp. 364–374, 1997.

[34] J. Hua, C. Sima, M. Cypert, G. C. Gooden, S. Shack, L. Alla, E. A. Smith, J. M. Trent, E. R. Dougherty, and M. L. Bittner, "Dynamical analysis of drug efficacy and mechanism of action using GFP reporters," *Journal of Biological Systems*, vol. 20, no. 04, pp. 403–422, 2012.