Non-linear state space modeling

Jared Woolsey, Ali Akbar Mehdizadeh, Yan-Yu Chen

(Dated: December 13, 2023)

In this project we review a study on nonlinear state space modeling employing bayesian approach through the Gibbs sampler technique. Nonlinear state space models provide a versatile framework for capturing complex dynamic systems, and the Gibbs sampler serves as a powerful tool for inference in such contexts. This project explores the application of the Gibbs sampler in estimating latent states and parameters in nonlinear state space models with an emphasis on noisy logistic maps. The study delves into the methodology, emphasizing the algorithm's capability to handle nonlinearity and uncertainties in the system. The findings contribute to the advancement of nonlinear state space modeling methodologies and investigate the practical utility of the Gibbs sampler in addressing challenges posed by complex, dynamic systems near chaos.

I. INTRODUCTION

State-space modeling is a powerful technique extensively employed in the analysis and representation of dynamic systems, providing a systematic framework to describe their behavior over time. When applied to nonlinear systems, the traditional linear state-space representation is extended to accommodate the intricacies of nonlinear dynamics. In state-space modeling of nonlinear systems, variables evolve over time, and their interactions are captured through nonlinear equations. This approach enables the encapsulation of complex system behaviors that may not be accurately represented by linear models.

By employing nonlinear state-space modeling, one might can investigate the dynamics of systems exhibiting nonlinearity, allowing for more accurate predictions and facilitating the design of control strategies tailored to handle the inherent complexities of such systems, specially when we are dealing with hidden non-linear systems where we have have noisy measurements of an non-linear stochastic system.

II. NON-LINEAR SYSTEMS

Even simple nonlinear systems can display extremely complex dynamics, as pointed out by May in his influential work [1]. To demonstrate this rich complexity, we use the logistic map as an illustrative example:

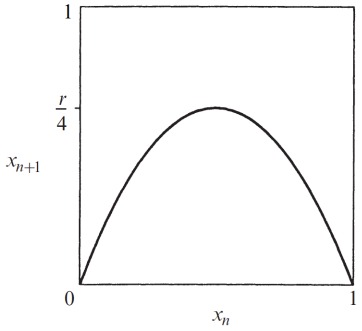

FIG. 1. Logistic Map

$$x_{n+1} = r x_n (1 - x_n) \tag{1}$$

Here, the x_n may denote a dimensionless metric for the population in the nth generation, while r can signify the intrinsic growth rate. The plot of a logistic map, represented by Eq. 1, takes the shape of a parabola, reaching its maximum value of $\frac{r}{4}$ when x is equal to $\frac{1}{2}$. To preserve meaningful dynamics, it is assumed that the parameter r is constrained within the range of $0 \leq r \leq 4$, ensuring that Eq. 1 maps the interval $0 \leq x \leq 1$ onto itself.

A. Period-Doubling

Suppose we set a constant value for r, select an initial population x_0, and subsequently employ Eq. 1 to generate the successive values

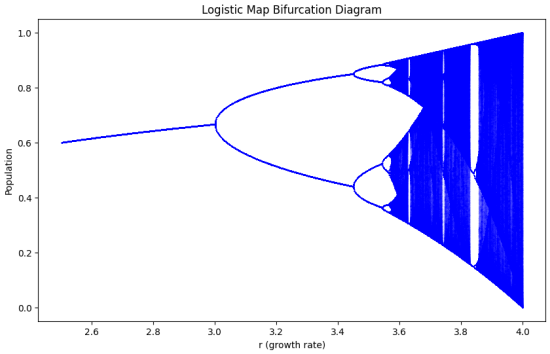

FIG. 2. Bifurcation Diagram for Logistic Map as control parameter varies.

$r_1 = 3$	(period 2 is born)
$r_2 = 3.449\ldots$	4
$r_3 = 3.54409\ldots$	8
$r_4 = 3.5644\ldots$	16
$r_5 = 3.568759\ldots$	32
$\vdots$	$\vdots$
$r_\infty = 3.569946\ldots$	∞

FIG. 3. Bifurcation Table for Logistic Map [2]

of x_n. When r is significantly small ($r \ll 1$), the population becomes extinct, converging to $x_n \to 0$ as $n \to \infty$. In the range $1 < r < 3$, the population undergoes growth and eventually attains a non-zero steady state.

We can visualize these final values by plotting them after eliminating transient initial states. The corresponding outcomes are depicted in Fig.2 (codes are included in the appendix)

With higher values of r, the population undergoes oscillations around the previous stable state, fluctuating between a large population in one generation and a smaller population in the next. This oscillation pattern, characterized by an alternation between two states, is referred to as period doubling. Upon further increasing r, the system exhibits oscillations between an increasing number of states, such as $4, 8, 16, \ldots$.

Bifurcations occur more rapidly with an increase in the value of r. Eventually, the sequence of control parameters r_n converges to a limiting value denoted as r_∞. This convergence follows a geometric pattern: as n becomes very large, the spacing between successive transitions diminishes by a constant factor.

When r reaches r_∞, the map enters a chaotic regime, and the set of stable points transforms from a finite set to an infinite set. Beyond r_∞, the system displays a surprising blend of order and chaos, featuring periodic windows amid chaotic regimes..

B. Lyapunov Exponent

In order for a system to be characterized as "chaotic," it must display increased sensitivity to initial conditions, with neighboring trajectories diverging exponentially quickly. Assuming an initial condition x_0, consider a nearby point $x_0 + \delta_0$. Let δ_n represent the separation after n iterations. If δ_n is approximately equal to $\delta_0 e^{n\lambda}$, then λ is referred to as the Lyapunov exponent. A positive Lyapunov exponent serves as the hallmark of chaotic behavior.

It can be seen that $\delta_n = f^n(x_0 + \delta_0) - f^n(x_0)$ so

$$\begin{aligned}\lambda &\approx \frac{1}{n}\ln\left|\frac{\delta_n}{\delta_0}\right| \\ &= \frac{1}{n}\ln\left|\frac{f^n(x_0+\delta_0) - f^n(x_0)}{\delta_0}\right| \\ &= \frac{1}{n}\ln\left|(f^n)'(x_0)\right|\end{aligned}$$

The expression within the logarithm can be expanded using the chain rule:

$$(f^n)'(x_0) = \prod_{i=0}^{n-1} f'(x_i)\,.$$

Hence

$$\begin{aligned}\lambda &\approx \frac{1}{n}\ln\left|\prod_{i=0}^{n-1} f'(x_i)\right| \\ &= \frac{1}{n}\sum_{i=0}^{n-1}\ln\left|f'(x_i)\right|\,.\end{aligned}$$

If this expression converges to a limit as $n \to \infty$, we designate that limit as the Lyapunov exponent for the orbit originating from x_0:

$$\lambda = \lim_{n\to\infty}\left\{\frac{1}{n}\sum_{i=0}^{n-1}\ln\left|f'(x_i)\right|\right\}.$$

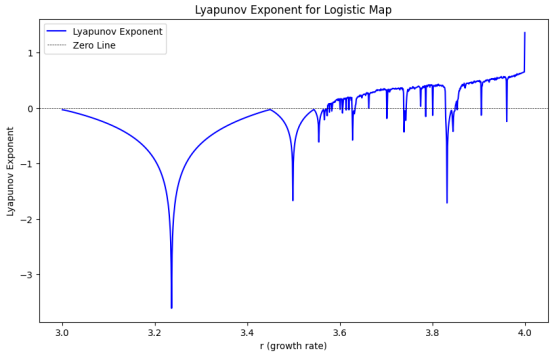

FIG. 4. Lyapunov Exponents for the Logistic Map

For stable fixed points and cycles, the Lyapunov exponent (λ) is negative, while for chaotic attractors, λ is positive. The outcomes of simulations, illustrating the calculation of the Lyapunov Exponent for the logistic map, are presented in Fig. 4, codes are included in the attachment.

C. Renormalization

In this section, I provide an intuitive introduction to the renormalization theory for period-doubling and offer insights into the mathematical derivation of r_∞, representing the onset of chaos. This section serves primarily as a review of the reference [2].

The study in [3] has shown that all unimodal maps share comparable bifurcation diagrams, depicted in FIG. 2. In essence, they each experience period-doubling routes leading to chaos, succeeded by periodic windows intertwined with chaotic bands.

Let $f(x, r)$ represent a unimodal map that undergoes a period-doubling route to chaos as r increases. Suppose x_m is the maximum of f. Let r_n denote the value of r at which a 2^n-cycle is initiated, and let R_n denote the value of r at which the 2^n-cycle becomes superstable (a superstable n-cycle is characterized by $f^n(x) = x$ and $\frac{d}{dx} f^n(x) = 0$).

Indeed, the convergence rate remains consistent across different unimodal maps, as depicted

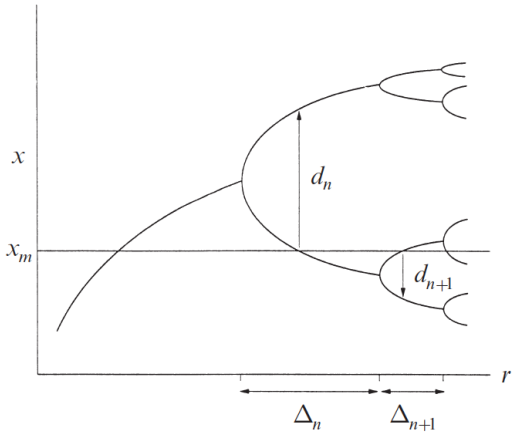

FIG. 5. Schematic Representation of Bifurcation of Unimocal Maps [2]

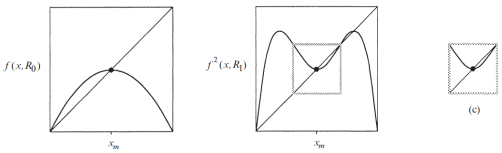

FIG. 6. Schematic representation of self-similarity between f and f^2 [2]

in FIG.5:

$$\delta = \lim_{n\to\infty} \frac{r_n - r_{n+1}}{r_{n+1} - r_n} = \frac{\Delta_n}{\Delta_{n+1}} = 4.669\cdots \quad (2)$$

$$\alpha = \lim_{n\to\infty} \frac{d_n}{d_{n+1}} = -2.5029\cdots \quad (3)$$

This phenomenon can be further elucidated through the perspective of renormalization and self-similarity in the context of chaotic maps [4, 5]. The renormalization theory draws inspiration from the self-similarity observed in the bifurcation diagram, akin to the patterns found in the branching of fig trees.

To illustrate self-similarity, envisioning f and f^2 involves normalizing f^2 and then mapping it back into f (see FIG. 6). Here, R_0 and R_1 represent superstable cycles of f and f^2, respectively. To transition from one to the other, a change of scale and a reversal of both axes are necessary. This transformation can be represented as:

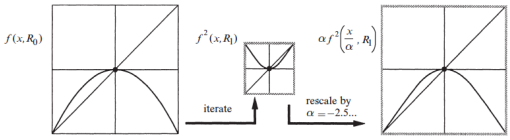

FIG. 7. Renormalizaioton between f and f^2 [2]

$$f(x,R_0) \approx \alpha f^2(\frac{x}{\alpha}, R_1) \tag{4}$$

In summary, the renormalization of f involves taking its second iterate, rescaling $x \to \frac{x}{\alpha}$, and shifting r to the next superstable value. This process can be generalized for n iterations:

$$f(x,R_0) \approx \alpha^n f^{2^n}(\frac{x}{\alpha^n}, R_n) \tag{5}$$

In this expression, the limit is denoted as $g_0(x)$, which is a universal function independent of the specific form of the mapping f. Generally, universal functions can be conceptualized as:

$$g_i(x) = \lim_{n\to\infty} \alpha^n f^{2^n}(\frac{x}{\alpha^n}, R_{n+i}) \tag{6}$$

In the state of chaos, when $R_i = R_\infty$, there is no need for a shift during renormalization. This yields the following relation:

$$g_\infty(x) = g(x) = \alpha g^2(x/\alpha) \tag{7}$$

This forms a self-referential functional equation for $g(x)$ and the universal scale factor α. By examining the boundary conditions for $g(x)$, one can solve this equation and obtain $\alpha \approx 2.5029$, a value consistent with the numerical results obtained from simulations.

D. Problem Setup

In the following sections, we investigate a Bayesian approach that can be employed to analyze a hidden noisy logistic map [6]. We make the following assumptions:

$$x_{n+1} = rx_n(1 - x_n) + u_n \tag{8}$$

$$y_n = bx_n^2 + v_n \tag{9}$$

In these equations, the first one characterizes the concealed dynamics of the system, incorporating a logistic map along with white noise. The second equation describes the noisy observations of the system, with a known constant b. A Bayesian approach is particular helpful in this problem because it models the unknown state distribution by a prior and links it to the observation by updating through a posterior. The procedure mimics the nature of state-space model and allows us to do inference from the sample of unknown state we generate.

Our goal is to assess our ability to estimate $\hat{r}$ (a marginal distribution for the actual value of r), predict one step ahead of the system given a sequence of observations (i.e., $p(x_{n+1}|\vec{y})$), and also filter through the observations on the next observable (i.e., $p(x_{n+1}|\vec{y}, y_{n+1})$). The exploration of whether the Gibbs sampler encounters success or failure is significant, particularly in regimes proximate to chaos, where internal dynamics exhibit substantial divergence.

III. STATE SPACE MODELS

The difference between a typical time series and a state space model, is that in a state space model, we have unknown states x_t, which are associated with our observed values y_t. We will start this section discussing Linear State Space models, and a filtering technique called the Kalman Filter, which can be used to estimate states in this the linear case. Then, we discuss a specific example of applying the Kalman Filter before introducing Non-Linear State Spaces.

A. Linear Vs. Non-Linear State Space Models

We start by considering the standard linear state space model given by Equation 10. Here, H_t and F_t represent known matrices of constants. The simplest case for this is when $H_t = H$ and $F_t = F$, in which case, we have a univariate linear model. In practice, these matrices, as well, must be estimated. The Technique we will

discuss for estimating the states x_t as well as forecasting future y_t is called Kalman Filtering.

$$\begin{aligned} y_t &= H_t x_t + \epsilon_t, \quad \epsilon_t \sim N(0, \Sigma_\epsilon^2) \\ x_{n+1} &= F_t x_t + \eta_t, \quad \eta_t \sim N(0, \Sigma_\eta^2) \end{aligned} \tag{10}$$

In contrast to the linear state-space model, we also can have a non-linear state space, as seen in equation 11, where $h_t(*)$ and $f_t(*)$ can depend on unknown parameters, but are assumed to have a known structure. The basic Kalman Filter cannot be used in this setting, although, if we still have the normal error assumption, there are slight modifications to the Kalman Filter called the Extended Kalman Filter as well as the Unscented Kalman filter. We will not discuss these algorithms in this paper, as the focus is on understanding the Gibbs sampler, which can be used in a much more general setting, without even the normal error assumption.

$$\begin{aligned} y_t &= h_t(x_t) + \epsilon_t, \\ x_{t+1} &= f_t(x_t) + \eta_t \end{aligned} \tag{11}$$

B. Kalman Filter

The Kalman Filter is a method we use to update assumptions of a system's state, after new data is observed. Similar to a Hidden Markov Model (HMM), we have a system of unknown states we are trying to estimate, and we want to forecast a probability distribution of future observations. While HMM's operate in the discrete setting, updating estimates of the probability distribution of two coins, a Kalman Filter continuously refines state estimates, as time is considered a continuous variable.

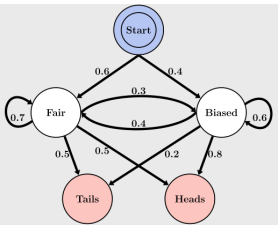

FIG. 8. Hidden Markov Model

A basic Kalman Filter can be applied to estimate the state of a system, and forecast future values in the context of a linear system with Gaussian noise. The Kalman Filter can be implemented as shown in equation 13, where $K_t = \frac{P_t}{F_t}$ and a_1, P_1 are assumed to be known. We must make some assumptions as to the structure of our system. Namely, the assumptions:

$$\begin{aligned} (Y_{t-1}|x_{t+1}) &\sim N(a_{t+1}, P_{t+1}) \\ (x_t|Y_{t-1}) &\sim N(a_t, P_t) \\ (x_t|Y_t) &\sim N(a_{t|t}, P_{t|t}) \end{aligned} \tag{12}$$

Let us explain the assumptions more intuitively. First, we assume some (normal) conditional distribution of $Y_{t-1}|x_{t+1}$, where Y_{t-1} is the vector of all our previous observations. Thus, a_{t+1} represents the one-step-ahead predictor of x_{t+1}, and P_{t+1} it's associated variance (our goal is to find both of these quantities). The previous assumption represented our Future state estimate based on current observations, and $x_t|Y_{t-1}$ is our current state estimate given past observations. Lastly, we assume $x_t|Y_t$ —which represents our Updated state estimate, which utilizes our newest observation y_t— follows a normal distribution as well. Thus, $a_{t|t}$ is referred to as the filtered estimator of the state x_t and $P_{t|t}$ it's associated variance.

Given these assumptions of conditional linearity and normality, we can determine the properties given in equation 14.

$$\begin{aligned} v_t &= y_t - a_t, \\ a_{t|t} &= a_t + K_t v_t, \\ a_{t+1} &= a_t + K_t v_t, \\ F_t &= P_t + \sigma_\epsilon^2, \\ P_{t|t} &= P_t(1 - K_t), \\ P_{t+1} &= P_t(1 - K_t) + \sigma_\eta^2. \end{aligned} \tag{13}$$

This updating scheme has many components, so I will break it down to make more sense as follows:

We start with some a_1 and P_1 which is simply assumed, but may be arbitrary. We define a variable v_t which represents the one-step ahead

prediction error of y_t and F_t it's associated variance. K_t represents the "Kalman Gain", or the relative uncertainty between the current estimate and the new observation at a time t. As K_t becomes smaller, this indicates there is less information to gain via collecting a new observation. We update from a_1 to a_2 by adding our one-step-ahead prediction error times the initial Kalman Gain (which simply measures the proportion of variance of our one step-ahead predictor to this variance plus the inherent variance of the $y_t's$) to a_1. P_1 is updated to P_2 via multiplication by the complement of the Kalman Gain and the one-step ahead prediction variance, plus the variance of the $x_t's$

$$\begin{aligned} E(v_t|Y_{t-1}) &= E(x_t + \epsilon_t - a_t|Y_{t-1}) = a_t - a_t = 0, \\ \mathrm{Var}(v_t|Y_{t-1}) &= \mathrm{Var}(x_t + \epsilon_t - a_t|Y_{t-1}) = P_t + \sigma_\epsilon^2, \\ E(v_t|x_t, Y_{t-1}) &= E(x_t + \epsilon_t - a_t|x_t, Y_{t-1}) = x_t - a_t, \\ \mathrm{Var}(v_t|x_t, Y_{t-1}) &= \mathrm{Var}(x_t + \epsilon_t - a_t|x_t, Y_{t-1}) = \sigma_\epsilon^2. \end{aligned} \tag{14}$$

1. *Nile River Example*

(This example was found in [7]) In the following example, we apply the Kalman Filter to data using observations from the Nile River annual flow Volume. Here, our hidden states are physical processes which affect the river flow, but are not directly observed. The existence of such features can be inferred by the fact that this is a physical system, and so linearity and normality assumptions are likely to hold reasonably well. In Figure 9, we see four graphs which describe the Nile River Data, and how well we can perform estimation and prediction using a Kalman Filter. In graph (i), we see the data: On the x axis is the year, and on the y axis we see the annual flow volume of the Nile River. We apply the iterative procedure discussed in this section, where $a_1 = 0, P_1 = 10^7, \sigma_\epsilon^2 = 15099, \sigma_\eta^2 = 1469.1$ with a_1 and P_1 were chosen arbitrarily. Going through the first few points we would see:

$$\begin{aligned} &y = [1120, 1160, 963, ..., 718, 714, 740] \\ &v_1 = y_1 - a_1 = 1120 - 0 = 1120 \\ &F_1 = P_1 + \sigma_\epsilon^2 = 10^7 + 15099 \\ &K_1 = P_1/F_1 = 0.9985 \\ &a_2 = a_1 + K_1 v_1 = 0 + 0.9985 * 1120 = 1118.3 \\ &P_2 = P_1(0.0015) + 1469.1 = 16469.1, \\ &... \end{aligned} \tag{15}$$

And we would end up finding $a = [1118, 1140, ..., 820, 798]$ These points are all shown in (i), as well as their 90% Confidence intervals, which helps us get more of a smooth, more accurate representation of the underlying state of the system across time. (ii) shows our $P_t's$, which clearly, very rapidly converge to a constant value, and (iv), which shows our prediction variances $F_t's$, a similar statement can be seen. (iii) simply shows the error in our predictions, which are generally less than about 250 in absolute value. As we can see, the Kalman Filter provides a smooth function which extracts much of the meaningful information hidden in our data.

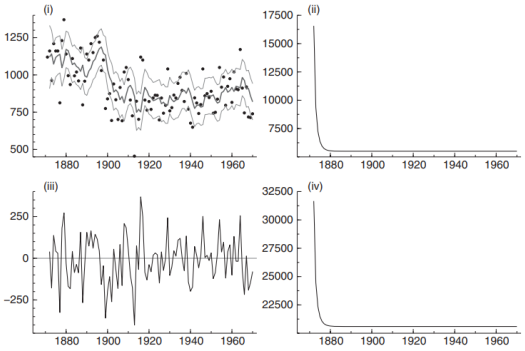

FIG. 9. Nile River Data with Kalman Filter Applied

IV. GIBBS SAMPLER

A. Monte Carlo Simulation

Before we talk about the Gibbs sampler, it is very important to understand what Monte carlo Simulation is, and how Monte Carlo methods

are used to solve problems for which traditional numeric techniques cannot be used. So in this section, we will introduce a brief overview of Monte Carlo Simulation.

Monte Carlo Simulation, as the name may imply, is a technique in which we use randomization to simulate data from a probability distribution many times to find many types of numeric results. This can be best understood with a simple example with the game of blackjack. We are interested in answering the question of whether or not a player can make money playing blackjack, which we will measure with metrics such as expected value, variance, and risk of ruin (AKA, losing all your money at some point over the course of play). Figure 10 demonstrates the results of a particular implementation of Monte Carlo simulation in which the user defines a particular playing and betting strategy which will be strictly adhered to, as well as a particular set of game rules. One billion random hands are simulated with these parameters, and we can see the following results for a player perfectly following a this strategy:

- Starting Bankroll = $20,000
- 100 Rounds are dealt per hour
- Expected profit = $140 per hour,
- Variance = $1153.34 per hour,
- Risk of Ruin = 1.5%

1. *BlackJack Simulation*

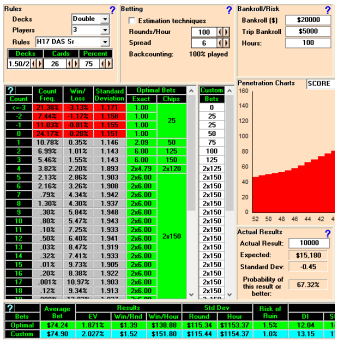

FIG. 10. BlackJack Monte Carlo Simulation

This problem would be very difficult to solve outright with probability theory, and a closed-form analytical solution is impossible, so Monte Carlo techniques are employed. This technique is very important to this paper, as the Gibbs sampler is actually an adaptive Monte Carlo integration technique. Very importantly, Monte Carlo Methods do not require any assumption about the state of the system being modeled, which is very important to solving non-linear state space problems.

B. Gibbs Sampler

The Gibbs sampler is a Markov Chain Monte Carlo (MCMC) algorithm used for statistical inference and Bayesian sampling. The primary purpose of the Gibbs sampler is to approximate the joint distribution of multiple random variables by iteratively sampling from their conditional distributions. In principle, if we generate a large sample and update the sampling equations many times, the sampler should have the same law to the underlying true state-space model. Therefore, we could do forecasting and filtering from the sample we generate.

Here's a basic overview of how the Gibbs sampler works:

1. Setup: Consider a set of random variables, $U = (U_1, \ldots, U_k)$, for which you want to sample from the joint distribution $P(U)$.

2. Initialization: Start with an initial set of values for each variable in U. This serves as the starting point for the algorithm.

3. Iteration: For each iteration, select one variable from U and update its value by sampling from its conditional distribution given the current values of the other variables. Repeat this process for each variable in U.

4. Convergence: After a sufficient number of iterations, the sampler reaches a point where the samples drawn closely approximate the joint distribution of the variables.

5. Repeat: Steps 3 and 4 are repeated until the algorithm converges to the desired precision or a predetermined number of iterations.

In particular, the variables U in our case are nuisance parameters and state $\{x_n\}_{n=1}^N$.

The key idea behind the Gibbs sampler is that, at each iteration, it updates one variable while keeping the others fixed. This conditional sampling allows the algorithm to explore the joint distribution of the variables in a way that converges to the true distribution.

The Gibbs sampler is particularly useful in Bayesian statistics when dealing with complex posterior distributions. It simplifies the sampling process by breaking it down into more manageable steps, as each variable is updated based on the values of the others. This approach is especially beneficial when direct sampling from the joint distribution is challenging. In our case, Gibbs sampler helps us obtain the joint distribution from a nonlinear state-space model.

While the Gibbs sampler is a powerful tool, it's important to note that its performance can be influenced by factors such as the choice of conditional distributions and the initialization of the algorithm. In particular, to obtain an analytic form of posterior distributions for Gibbs sampler to work, the prior distributions are chosen to be normal distributions if the marginal distribution is modeled by a normal or an inverse gamma distribution if that is modeled by a non-normal distribution. Both prior distributions with their parameters are wish to be chosen in the way that the corresponding mean is as close as possible to the true value for a fast convergence and stable numerical result. We will see a concrete example to choose a prior for Gibbs sampler in the next section.

V. SIMULATION & EXAMPLES

A. A Non-linear and Non-normal Example

Consider a non-linear and non-normal example from [6]. For $n = 1, \ldots, 100$,

$$x_n = \alpha x_{n-1} + \beta \frac{x_{n-1}}{1 + x_{n-1}^2} + \gamma \cos(1.2(n-1)) + u_n \tag{16}$$

$$y_n = x_n^2/20 + v_n, \tag{17}$$

where $\alpha = 0.5$, $\beta = 25$, $\gamma = 8$, $x_0 = 0$ are given but unknown to the analyzer. The distributions of $u_n \sim t_v$ and $v_n \sim N(0, 1)$ and $v = 10$ are also given and known to the analyzer. The visualization of a simulated data and its prediction is shown in Figure 11. The posterior distribution of the nuisance parameters obtained from Gibbs sampler are shown in Figure 12. Both the prediction and peak of their distribution is very close to the true values, suggesting the practical benefit of Gibbs sampler to a non-linear state-space model. A complete simulation detail could be found in [6].

The only thing left here is how to handle non-normal errors in Equation 16. According to the paper[6], they modeled a non-normal error by a mixture of normal distributions. To be more specific, they assumed there exists a latent variable λ_n such that the conditional distribution of x_n given λ_n is normal, i.e.,

$$p(x_n|x_{n-1}, \sigma^2) = \int_\Lambda p(x_n|x_{n-1}, \lambda_n, \sigma^2) p(\lambda_n) d\lambda_n,$$

where $x_n|x_{n-1}, \lambda_n, \sigma^2 \sim N(f(x_{n-1}), \lambda_n \sigma^2)$ with $f(\cdot)$ being a known state-space function of x_n. The prior distribution $p(\lambda_n)$ again is chosen from inverse gamma distributions to obtain an analytic form of posterior distributions for Gibbs sampler to work. The intuition to model non-normal distribution by the mixture of normal is to approximate the underlying distribution by multiple normal distributions with the common mean centered around the state-space model, yet each contributing to the overall observed distribution with its own standard deviation $\sqrt{\lambda_n}\sigma$, and weight $p(\lambda_n)$. Obvious, if x_n itself was normally distributed, one just set

$\lambda_n = 1$ with probability one to cover the normal error case.

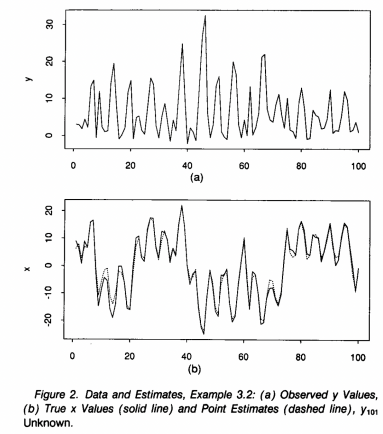

Figure 2. Data and Estimates, Example 3.2: (a) Observed y Values, (b) True x Values (solid line) and Point Estimates (dashed line), y_{101} Unknown.

FIG. 11. Simulated state-space model and its prediction

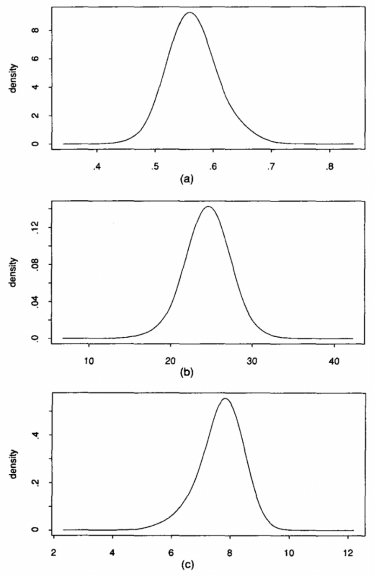

Figure 3. Estimated Marginal Posteriors, Example 3.2: (a) Marginal Posterior for Alpha, G = 500; Mode = .56, (b) Marginal Posterior for Beta, G = 500; Mode = 24.641, (c) Marginal Posterior for Gamma, G = 500; Mode = 7.826.

FIG. 12. Posterior distribution of nuisance parameters

B. Revisit Logistic Map

In this section, we consider the hidden noisy logistic map given in Equation 8 for $n = 1, \ldots, 25$, where $r = 0.5, 1, 5, 3, 3.8$, $x_0 = 0.5$, $b = 20$, $u_n \sim N(0, \sigma^2)$ with $\sigma = 0.003$, and $v_n \sim N(0, \tau^2)$ with $\tau = 0.1$ are all known to the analyzer. The different choose of r allows us to observe different dynamic behaviors of the logistic map. The visualization of simulated data and its prediction are shown in Figure 13 and 14 for $r = 0.5$, Figure 15 and 16 for $r = 1.5$, Figure 17 and 18 for $r = 3$, and Figure 19 and 20 for $r = 3.8$. For x_n without noise, we know that if $0 < r < 1$, $x_n \to 0$, leading to the extinction; if $1 < r < 3$, $x_n \to (r-1)/r$; if $3 < r < 3.57$, x_n behaves oscillation; if $3.57 < r < 4$, x_n becomes chaos. In our simulation, we found that Gibbs sampler works for the first three cases, but seems to fail when r lies in the range of chaos. This suggests that the system of the hidden logistic map is sensitive to the value of r. In particular, we also try to make r unknown to the analyzer, but our numerical experiment found that it is very likely to sample r across different phases and thus make its numerical result unstable to converge to a reasonable value. To sum up, if a r goes from 0 to 3.83, the prediction of Gibbs sampler will become more and more sensitive to the previous prediction due to the nature of logistic map, thus making the prediction unreliable.

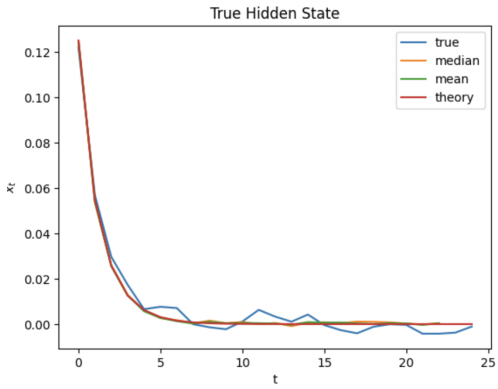

FIG. 13. State for $r = 0.5$

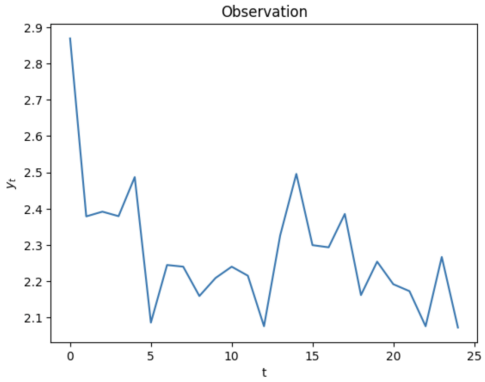

FIG. 14. Observation for $r = 0.5$

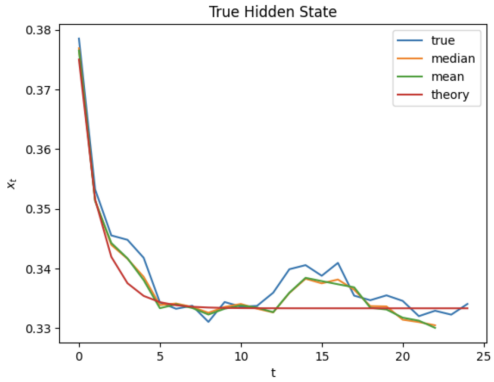

FIG. 15. State for $r = 1.5$

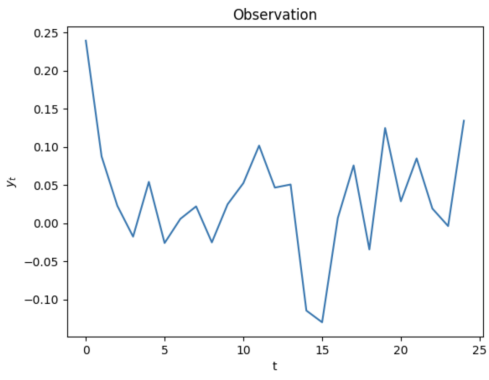

FIG. 16. Observation for $r = 1.5$

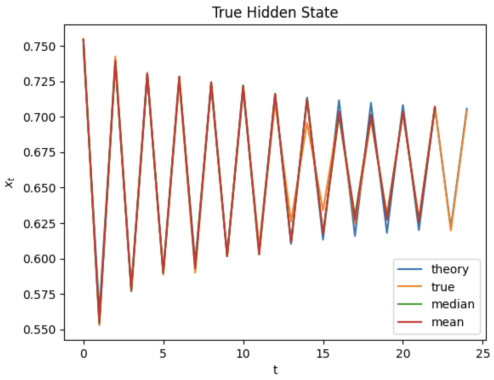

FIG. 17. State for $r = 3$

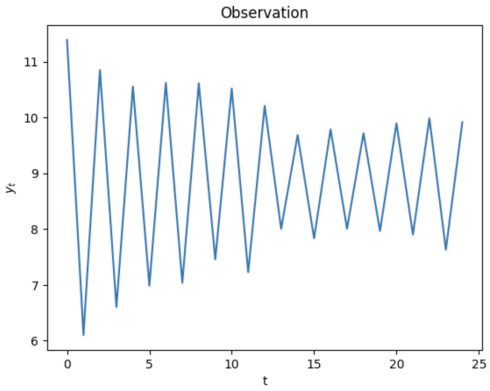

FIG. 18. Observation for $r = 3$

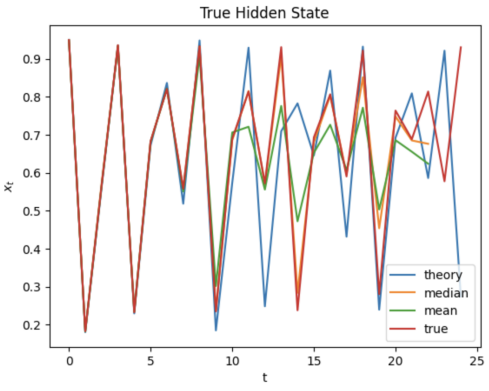

FIG. 19. State for $r = 3.8$

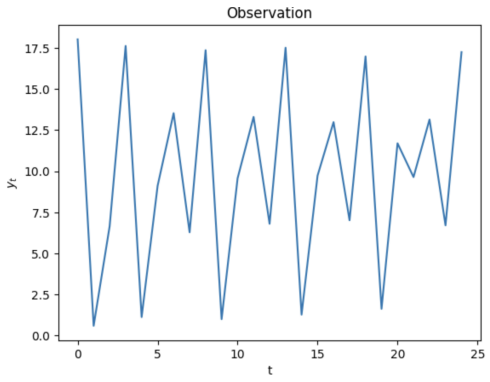

FIG. 20. Observation for $r = 3.8$

VI. CONCLUSION

The logistic map exhibits diverse dynamic behaviors determined by the parameter r. Gibbs sampler is a valuable tool for non-linear and non-normal state space models, especially when Kalman Filtering, a linear model, proves inadequate. However, Gibbs sampler requires intensive computation and knowledge of the functional forms of $h_n(x)$ and $f_n(x)$ in a state-space model, as well as the distributions of noise. The second problem may be partially alleviated by employing Baum-Welch algorithm from hidden Markov models as a nonparametric approach to obtain an initial fit and guess the form of $h_n(x)$ and $f_n(x)$. Careful prior selection is also crucial, and the method may be potentially inefficient in sampling. Furthermore, sampling r in the logistic map can be challenging due to its dynamic phases.

[1] R. M. May, Simple mathematical models with very complicated dynamics, Nature **261**, 459 (1976).
[2] S. H. Strogatz, *Nonlinear dynamics and chaos with student solutions manual: With applications to physics, biology, chemistry, and engineering* (CRC press, 2018).
[3] N. Metropolis, M. Stein, and P. Stein, On finite limit sets for transformations on the unit interval, in *Universality in Chaos, 2nd edition* (Routledge, 2017) pp. 185–206.
[4] M. J. Feigenbaum, The universal metric properties of nonlinear transformations, Journal of Statistical Physics **21**, 669 (1979).
[5] M. J. Feigenbaum, The onset spectrum of turbulence, Physics Letters A **74**, 375 (1979).
[6] B. P. Carlin, N. G. Polson, and D. S. Stoffer, A monte carlo approach to nonnormal and nonlinear state-space modeling, Journal of the american Statistical association **87**, 493 (1992).
[7] J. Durbin and S. J. Koopman, *Time Series Analysis by State Space Methods* (Oxford University Press, 2012).

VII. APPENDIX

Codes used for generating logistic map plots can be found here: `https://github.com/AliakbarMehdizadeh/STA-237-Project`