

# STATE-SPACE MODELS: UNRAVELLING HIDDEN DYNAMICS IN DATA

## Abstract

The chapter encapsulates a sophisticated framework for analyzing complex systems by revealing latent patterns and dynamics within observed data. State-space models (SSMs) have emerged as a fundamental tool in various disciplines, including engineering, economics, biology, and neuroscience, due to their capacity to unearth underlying dynamics that generate observed measurements. SSMs operate on the premise that observed data is a manifestation of both observable states and hidden variables, with a temporal evolution influenced by intricate dynamics. By disentangling these hidden dynamics, SSMs enable researchers to gain deeper insights into the underlying mechanisms that govern system behavior. This is achieved through a two-fold process: the observation model and the state-transition model. The utilization of SSMs is particularly valuable in scenarios where direct measurements of underlying phenomena are unattainable or noisy. In summary, "State-Space Models: Unraveling Hidden Dynamics in Data" underscores the importance of these models as a versatile approach for capturing the intricate relationships between observed data and the latent processes driving it. By harnessing SSMs, researchers are empowered to uncover the concealed dynamics that shape various phenomena, fostering a deeper comprehension of complex systems across diverse domains.

**Keywords:** State-space models, Dynamics, components,

## Authors

### **P. R Vekariya**

B. A. College of Agriculture  
Anand Agricultural University  
Anand, Gujarat, India

### **Prity Kumari**

College of Horticulture  
Anand Agricultural University  
Anand, Gujarat, India

### **Y. A Garde**

N.M. College of Agriculture  
Navsari Agricultural University  
Navsari, Gujarat, India

## I. INTRODUCTION

State-space models are a strong framework in statistics for unraveling complicated and hidden processes in various datasets. These models offer a versatile way to identify temporal or sequential data, making them useful tools in disciplines ranging from economics to engineering to biology and beyond. State-space models effectively decompose and clarify complex data structures, providing cleaned and structured inputs that enhance the learning and predictive accuracy of ANNs in identifying intricate patterns within the data in order to do forecasting (Satish and Kumari, 2021; Kumari et. al., 2022, 2023). Let us investigate the complexities of state-space models, learning how they reveal expose hidden variables, improve forecasting, and revolutionize our understanding of dynamic systems.

## II. UNVEILING THE STATE-SPACE CONCEPT

A state-space model's fundamental function is to depict how a system changes over time (Newman *et al.* 2014). The two essential elements that define its evolution are the "state" and the "observation." While the observation relates to the quantifiable results we can immediately perceive, the state represents the underlying, unobservable variables that determine the system's behavior. These models resemble a secret story that is being played out in the background, a story that state-space models aim to reveal (McClintock *et al.*, 2014).

A state space model (SSM) is a time series model in which the time series  $Y_t$  is interpreted as the result of a noisy observation of a stochastic process  $X_t$ . The values of the variables  $X_t$  and  $Y_t$  can be continuous (scalar or vector) or discrete. SSMs belong to the realm of Bayesian inference, and they have been successfully applied in many fields to solve a broad range of problems. It is usually assumed that the state process  $X_t$  is Markovian, *i.e.*,  $X_t$  depends on the history only through  $X_{t-1}$ , and  $Y_t$  depends only on  $X_t$ :

$$\begin{aligned} X_t &\sim p(X_t | X_{t-1}) \\ Y_t &\sim p(Y_t | X_t) \end{aligned}$$

## III. COMPONENTS & STRUCTURES

- 1. State Variables:** State variables are an array of internal variables that describe a dynamic system's existing state. They represent the fundamental elements of the system, which are critical for predicting how it will behave over time but cannot be observed directly. A vector, often written as  $X(t)$ , is used to represent state variables, where "t" stands for the current time step. The scale of the system being modeled determines how many state variables are needed.
- 2. Observation / Measurement Variables:** The measurable quantities or system outputs are known as observation variables. Unlike state variables, observation variables can be seen or measured immediately. They are typically denoted by a vector, typically written as  $Y(t)$ , where "t" stands for the current time step. Depending on the information provided by the system in terms of measurements, the number of observation variables may change.

3. **Control Variables:** Control variables are applied to a system as external inputs or control signals to modify its behavior. These inputs can be regulated or optimized by an outside agent to maintain system performance. Control variables are commonly expressed as a vector,  $u(t)$ , where "t" stands for the current time step.
4. **System Dynamics Equations:** The state variable evolution over time is described by the system dynamics equations. They stand in for the fundamental rules or precepts that direct how the system behaves. According to the needs of the modeling, these equations are often differential equations, either in continuous-time or discrete-time form.
5. **Observation Equations:** The state variables and the observation variables are related by the observation equations. They specify the mapping between the system's quantifiable outputs and the state variables. The measurement noise and uncertainties are also taken into consideration using observation equations.

#### IV. TYPES OF STATE SPACE MODELS

There are two types of state space models (SSMs), depending on the linearity of their state and observation equations:

1. **Linear State Space Models (LSSMs):** When the state and observation equations are written as linear functions of the state variables and observations, the resulting model is known as a linear state space model (LSSM).

- **State Equation (State Transition Model)**

$$X_t = A_t * X_{t-1} + B_t * u_t + w_t$$

Where,

$X_t$  : At time t, the state vector represents the system's hidden or unobservable variables

$A_t$  : A state transition matrix is a matrix that connects the state at time t to the state at time t-1. It captures the dynamics of the system

$X_{t-1}$  : State vector at time t-1

$B_t$  : At time t, the control input matrix accounts for any external control or effect on the system

$u_t$  : Control input vector at time t

$w_t$  : Process noise represents the uncertainty or random fluctuations in the state transition process

- **Observation Equation**

$$Y_t = C_t * X_t + v_t$$

Where,

$Y_t$  : At time t, the observation vector represents the system's measured or observed variables

$C_t$  : The observation matrix maps the state vector to the observation space. It expresses how the states are related to the observable quantities

$v_t$  : Observation noise, which accounts for measurement errors and uncertainty in the observed data

- **Key Characteristics**

- The linearity of the state and observation equations leads to closed-form solutions and effective computing.
- LSSMs often assume Gaussian processes and observation noise, simplifying estimation and enabling the application of Kalman filters and smoothers.
- The most well-studied SSM is the Kalman filter, for which the processes above are linear and the sources of randomness are Gaussian. Namely, a linear state space model has the form:

$$\begin{aligned} X_{t+1} &= GX_t + \varepsilon_{t+1} \\ Y_t &= HX_t + \eta_t \end{aligned}$$

- Here, the state vector  $X_t \in R^r$  is possibly unobservable and it can be observed only through the observation vector  $Y_t \in R^n$ .
- The matrices  $G \in MAT_r(R)$  and  $H \in MAT_{n,r}(R)$  are assumed to be known. For example, their values may be given by (economic) theory, or they may have been obtained through MLE estimation.
- In fact, the matrices G and H may depend deterministically on time, *i.e.*, G and H may be replaced by known matrices  $G_t$  and  $H_t$ , respectively.
- We also assume that the distribution of the initial value  $X_1$  is known and Gaussian. The vectors of residuals  $\varepsilon_t \in R^r$  and  $\eta_t \in R^n$  satisfy

$$\begin{aligned} E(\varepsilon_t \varepsilon_s^T) &= \delta_{ts} Q, \\ E(\eta_t \eta_s^T) &= \delta_{ts} R, \end{aligned}$$

- Where  $\delta_{ts}$  denotes Kronecker's delta, and where Q and R are known positive definite (covariance) matrices.
- We also assume that the components of  $\varepsilon_t$  and  $\eta_s$  are independent of each other for all t and s. The matrices Q and R may depend deterministically on time.
- The first of the equations is called the state equation, while the second one is referred to as the observation equation. Let T denote the time horizon.
- Our broad goal is to make inferences about the states  $X_t$  based on a set of observations  $Y_1, \dots, Y_T$ .

## 2. Nonlinear State Space Models

Nonlinear State Space Models (NSSMs) express the state equation, observation equation, or both as nonlinear functions of state variables and observations.

- **Nonlinear State Equation (State Transition Model)**

$$X_t = f(X_{t-1}, u_t) + w_t$$

Where,

- $X_t$  : At time t, the state vector represents the system's hidden or unobservable variables
- f : The nonlinear state transition function describes how the state at time t is affected by the state at time t-1 and any control inputs  $u_t$
- $X_{t-1}$  : State vector at time t-1
- $u_t$  : Control input vector at time t
- $w_t$  : Process noise represents the uncertainty or random fluctuations in the state transition process

- **Nonlinear Observation Equation:**

$$Y_t = h(X_t) + v_t$$

Where,

- $Y_t$  : At time  $t$ , the observation vector represents the system's measured or observed variables
- $H$  : A nonlinear observation function maps the state vector to the observation space. It describes how the states are related to the observable quantities
- $v_t$  : Observation noise, which accounts for measurement errors and uncertainty in the observed data

- **Key Characteristics**

- The model is more powerful and able to handle complex system interactions since at least one of the state or observation equations incorporates nonlinear functions.
- When estimating the states and parameters of NSSMs, numerical techniques like the Extended Kalman Filter (EKF), Unscented Kalman Filter (UKF), or Particle Filter (PF) are frequently used.
- To estimate the posterior distribution of states in NSSMs, particle filters are frequently used. This enables more accurate inference in non-Gaussian and highly nonlinear environments.
- NSSMs can model more real-world systems than LSSMs since linearity assumptions do not restrict them.

```
import numpy as np
import matplotlib.pyplot as plt
from statsmodels.tsa.statespace.kalman_filter import KalmanFilter
from statsmodels.tsa.statespace import tools

# Generate synthetic data
np.random.seed(0)
n_samples = 100
true_values = np.sin(np.linspace(0, 4 * np.pi, n_samples))
noisy_values = true_values + np.random.normal(0, 0.5, n_samples)

# Define the state space model
class SinStateSpace(KalmanFilter):
    def __init__(self, endog):
        super().__init__(k_states=2, k_obs=1)

        self['design', 0, 0] = 1.0
        self['transition', 0, 0] = 1.0
        self['transition', 0, 1] = 1.0
        self['selection', 0, 0] = 1.0
        self['obs_intercept', 0, 0] = 0.0
        self.initialize_known([0.0, 0.0], [[1.0, 0.0], [0.0, 1.0]])
        self.loglikelihood_burn = 1

# Create the state space model
model = SinStateSpace(noisy_values)
```

```
# Fit the model
results = model.smooth(noisy_values)

# Plot the true values, noisy measurements, and smoothed estimates
plt.plot(true_values, label='True Values')
plt.plot(noisy_values, label='Noisy Measurements')
plt.plot(results.filtered_state[0], label='Smoothed Estimates')
plt.xlabel('Time Step')
plt.ylabel('Value')
plt.title('State Space Model Example')
plt.legend()
plt.show()
```

## Kalman Filter

- Kalman filtering, formerly referred to as linear quadratic estimation (LQE), is a technique used in statistics and control theory that uses a collection of evaluations seen over time, such as statistical noise and other errors, to produce estimates of unobserved factors that are typically more accurate than those based on one measurement alone. It does this by estimating a joint probability distribution over the variables for every interval of time.
- Rudolf E. Kálmán (1960), one of its main theorists, is honored by having his name associated with the filter.
- By employing the dynamic model of the framework (such as physical laws of motion), existing control inputs to the system, and numerous consecutive observations from sensors, the state of the system can be calculated more precisely using Kalman filtering than it would by using only one measurement. As a result, it is a standard sensor fusion and data fusion algorithm.
- Numerous factors, such as noisy sensor data, estimations in the equations that describe the system's natural selection, and unaccounted-for outside variables, limit the precision of estimating the system's state.
- The Kalman filter effectively manages the uncertainty brought on by noisy sensor data and, to a lesser extent, by unforeseen external impacts. The Kalman filter creates an estimate of the system's state by blending the system's projected state and the current observation using a weighted average.
- For the sake of the weights, values with higher (*i.e.*, smaller) expected uncertainty are "trusted" more. The covariance, a metric reflecting the anticipated level of unpredictability in state prediction for the system, is used to establish the weights.
- The result of the weighted average is a new state prediction that is between the predicted and actual states and has a better-estimated uncertainty than each separately.
- This procedure is repeated for each time step, with the new estimate and its covariance affecting the prediction used in the following iteration.
- In order to avoid using the entire history of the system's state, the Kalman filter only uses the most recent "best guess" when calculating a new state.
- The current-state assessment and certainty grade of the measures are significant considerations. The reaction of the filter is typically described in terms of the gain of the Kalman filter.

- The weight assigned to the observations and current-state prediction is called the Kalman gain, and it can be "tuned" to achieve a particular performance.
- The filter reacts more rapidly and gives the most attention to the most recent data when the gain is high. When the gain is low, the filter more accurately reflects the model predictions.
- A high gain near one will produce an estimated direction that jumps around more at the extremes, while a low gain near zero will decrease noise but increase responsiveness.
- Due to the several dimensions required in a single set of procedures when performing the actual calculations for the filter (as described below), the state estimate and covariances are coded into matrices.
- This makes it possible to model linear relationships between a number of state parameters (such as position, velocity, and acceleration) in any of the changing models or covariances. Here's a basic example of how to implement a Kalman filter using Python and the filterpy library:

```
import numpy as np
from filterpy.kalman import KalmanFilter
import matplotlib.pyplot as plt

# Generate some synthetic data
np.random.seed(0)
n_samples = 100
true_values = np.linspace(0.1, 10.0, n_samples)
noisy_values = true_values + np.random.normal(0, 1, n_samples)

# Initialize the Kalman filter
kf = KalmanFilter(dim_x=2, dim_z=1)
kf.x = np.array([0.0, 1.0]) # Initial state [position, velocity]
kf.F = np.array([[1.0, 1.0], [0.0, 1.0]]) # State transition matrix
kf.H = np.array([[1.0, 0.0]]) # Measurement matrix
kf.P *= 1000.0 # Initial state covariance
kf.R = 1.0 # Measurement noise covariance
kf.Q = np.array([[0.001, 0.001], [0.001, 0.001]]) # Process noise covariance

# Store the filtered results
filtered_states = []
# Kalman filtering
for z in noisy_values:
    kf.predict()
    kf.update(z)
    filtered_states.append(kf.x[0]) # Estimated position

# Plot the true values, noisy measurements, and filtered estimates
plt.plot(true_values, label='True Values')
plt.plot(noisy_values, label='Noisy Measurements')
plt.plot(filtered_states, label='Filtered Estimates')
plt.xlabel('Time Step')
```

```
plt.ylabel('Value')
plt.title('Kalman Filter Example')
plt.legend()
plt.show()
```

- Before running the code, make sure you have the filterpy library installed in your Python environment:  
pip install filterpy

## V. APPLICATIONS OF STATE SPACE MODEL

- 1. Control Engineering:** SSMs are used by dynamic systems to simulate and predict behavior, allowing for the efficient implementation of control and feedback mechanisms. In fields like process control, robotics, and aerospace, they are essential.
- 2. Finance and Economics:** Models using state spaces are crucial for simulating financial time series, asset pricing, and economic variables. Their use is necessary for risk management and portfolio optimization. The ability to predict stock prices, interest rates, and economic indicators is another benefit.
- 3. Time Series Analysis:** State space models are useful for studying time-varying data, such as variations in temperature, traffic patterns, and economic trends. They include prediction, computation of missing variables, and the discovery of underlying patterns.
- 4. Signal Processing:** SSMs are essential for generating trustworthy and beneficial information from unstable signals and observations in applications involving signal processing. Systems for communication, speech recognition, and image processing all make use of them.
- 5. Ecology and Environmental Studies:** State Space Models (SSMs) are used by scientists to study ecological systems, animal populations, and environmental variables. They aid in the analysis of species interaction dynamics, ecosystem modeling, and climate change.
- 6. Health and Medicine:** Researchers utilize state space models to forecast disease transmission, optimize medication doses, and examine patient health trajectories in the fields of epidemiology, pharmacokinetics, and disease modeling.
- 7. Robotics and Autonomous Systems:** SSMs are used by researchers in robotics for behavior strategy, mapping, and localization. They enable robots to locate themselves and navigate hazy and dynamic situations.
- 8. Economics and Finance:** SSMs forecast economic indicators, model financial time series, and forecast asset values for use in economic analysis and decision-making.



## REFERENCES

- [1] Kalman, R. (1960). A New Approach to Linear Filtering and Prediction Problems. *ASME Journal of Basic Engineering*, 82, 35-45.
- [2] Kumari, P. and M. Sathish Kumar. Forecasting area, production and productivity of Citrus in Gujarat-An application of artificial neural network. *International Journal of Agricultural Sciences* 2021; 13 (10):10913-10916
- [3] Kumari P., Parmar D. J., M. Sathish Kumar, Lad Y. A. and Mahera A. B. An artificial neural network approach for predicting area, production and productivity of Banana in Gujarat. *The Pharma Innovation Journal* 2022; 11(4):816-821.
- [4] Kumari P, Goswami V, Harshith N, Pundir RS. Recurrent neural network architecture for forecasting banana prices in Gujarat, India. *Plos one* 2023; 18(6): e0275702.
- [5] McClintock, B. T., Johnson, D. S., Hooten, M. B., Ver Hoef, J. M. & Morales, J. M. (2014). When to be discrete: the importance of time formulation in understanding animal movement. *Movement Ecology*, 2-21.
- [6] Newman, K. B., Buckland, S. T., Morgan, B. J. T., King, R., Borchers, D. L., Cole, D. J., Besbeas, P., Gimenez, O. & Thomas, L. (2014). Modelling population dynamics: model formulation, fitting and assessment using state-space methods. *Springer*, New York.
- [7] Satish MK and Kumari P. Artificial neural network model for predicting area, production and productivity of sapota in Gujarat. *International journal of Agriculture Sciences* 2021; 13(10):10909-10912.