# Kalman Filter Theory and Applications Equation Drilldown: Taking the Equations Apart

https://github.com/musicarroll/kalman_course

Michael L. Carroll

June 16, 2023

©2023 by Michael L. Carroll

- Understanding the Equations: Heuristic Introduction

- Understanding the Equations: Heuristic Introduction
- **Equation Drilldown: Taking the Equations Apart**

- Understanding the Equations: Heuristic Introduction
- **Equation Drilldown: Taking the Equations Apart**
- State Space Concepts

- Understanding the Equations: Heuristic Introduction
- **Equation Drilldown: Taking the Equations Apart**
- State Space Concepts

- Mathematical Formulation of the Problem

- Mathematical Formulation of the Problem
- **Drilldown on State Dynamics and Covariance Extrapolation Equations**

- Mathematical Formulation of the Problem
- **Drilldown on State Dynamics and Covariance Extrapolation Equations**
- The Five Kalman Filter Equations

- Mathematical Formulation of the Problem
- **Drilldown on State Dynamics and Covariance Extrapolation Equations**
- The Five Kalman Filter Equations
- Examples

- Mathematical Formulation of the Problem
- **Drilldown on State Dynamics and Covariance Extrapolation Equations**
- The Five Kalman Filter Equations
- Examples
- Exercises

- Mathematical Formulation of the Problem
- **Drilldown on State Dynamics and Covariance Extrapolation Equations**
- The Five Kalman Filter Equations
- Examples
- Exercises

# Understanding Kalman Filter Notation

## Drilldown on State Dynamics

- State Dynamics Model: $x(k) = \Phi(k)x(k-1) + w(k)$

- State Dynamics Model: $x(k) = \Phi(k)x(k-1) + w(k)$
- $x(k)$ is the state vector with $n$ elements, written as a column vector:

- State Dynamics Model: $x(k) = \Phi(k)x(k-1) + w(k)$
- $x(k)$ is the state vector with $n$ elements, written as a column vector:
  - $x(k) = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}(k) = \begin{bmatrix} x_1(k) \\ \vdots \\ x_n(k) \end{bmatrix}$

# Understanding Kalman Filter Notation

## Drilldown on State Dynamics

- State Dynamics Model: $x(k) = \Phi(k)x(k-1) + w(k)$
- $x(k)$ is the state vector with $n$ elements, written as a column vector:
  - $x(k) = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} (k) = \begin{bmatrix} x_1(k) \\ \vdots \\ x_n(k) \end{bmatrix}$
- The process noise is also a column vector of size $n$:

# Understanding Kalman Filter Notation

## Drilldown on State Dynamics

- State Dynamics Model: $x(k) = \Phi(k)x(k-1) + w(k)$
- $x(k)$ is the state vector with $n$ elements, written as a column vector:
  - $x(k) = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}(k) = \begin{bmatrix} x_1(k) \\ \vdots \\ x_n(k) \end{bmatrix}$
- The process noise is also a column vector of size $n$:
  - $w(k) = \begin{bmatrix} w_1(k) \\ \vdots \\ w_n(k) \end{bmatrix}$

# Understanding Kalman Filter Notation

## Drilldown on State Dynamics

- State Dynamics Model: $x(k) = \Phi(k)x(k-1) + w(k)$
- $x(k)$ is the state vector with $n$ elements, written as a column vector:
  - $x(k) = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} (k) = \begin{bmatrix} x_1(k) \\ \vdots \\ x_n(k) \end{bmatrix}$
- The process noise is also a column vector of size $n$:
  - $w(k) = \begin{bmatrix} w_1(k) \\ \vdots \\ w_n(k) \end{bmatrix}$
- Some others prefer to make $k$ a subscript; we choose not to do so, in order to avoid clashes with vector and matrix elements which do by convention use subscripts

# Understanding Kalman Filter Notation

## Drilldown on State Dynamics

- Discrete State Dynamics

# Understanding Kalman Filter Notation

## Drilldown on State Dynamics

- Discrete State Dynamics
- KF doesn't work for nonlinear dynamics function $f[x(k-1)]$

- Discrete State Dynamics
- KF doesn't work for nonlinear dynamics function $f[x(k-1)]$
  - But nonlinear models can be linearized (more about that later)

# Understanding Kalman Filter Notation

## Drilldown on State Dynamics

- Discrete State Dynamics
- KF doesn't work for nonlinear dynamics function $f[x(k-1)]$
    - But nonlinear models can be linearized (more about that later)
- Restriction to matrices: $x(k) = \Phi(k)x(k-1) + w(k)$

# Understanding Kalman Filter Notation

## Drilldown on State Dynamics

- Discrete State Dynamics
- KF doesn't work for nonlinear dynamics function $f[x(k-1)]$
  - But nonlinear models can be linearized (more about that later)
- Restriction to matrices: $x(k) = \Phi(k)x(k-1) + w(k)$
- $\Phi(k)$ is called the **state transition matrix**. It is $n \times n$ in size.

- Discrete State Dynamics
- KF doesn't work for nonlinear dynamics function $f[x(k-1)]$
  - But nonlinear models can be linearized (more about that later)
- Restriction to matrices: $x(k) = \Phi(k)x(k-1) + w(k)$
- $\Phi(k)$ is called the **state transition matrix**. It is $n \times n$ in size.
  - If $\Phi$ is derived from a constant-coefficient difference or differential equation, and the time increments $\Delta t$ are constant, there there is no need for the index $k$; $\Phi$ will then be constant

# Understanding Kalman Filter Notation

## Drilldown on State Dynamics

- State Transition Matrix: $\Phi(k) = \Phi(k, k-1)$

- State Transition Matrix: $\Phi(k) = \Phi(k, k-1)$
- Extrapolates state vector from time step $k-1$ to time step $k$ (pulling from past to present)

# Understanding Kalman Filter Notation

## Drilldown on State Dynamics

- State Transition Matrix: $\Phi(k) = \Phi(k, k-1)$
- Extrapolates state vector from time step $k-1$ to time step $k$ (pulling from past to present)
    - If $\Delta t = t_k - t_{k-1}$ is constant for all $k$, then we might write $\Phi(\Delta t)$

- State Transition Matrix: $\Phi(k) = \Phi(k, k-1)$
- Extrapolates state vector from time step $k-1$ to time step $k$ (pulling from past to present)
  - If $\Delta t = t_k - t_{k-1}$ is constant for all $k$, then we might write $\Phi(\Delta t)$
- Tells you how the state would evolve **in the absence of forcing functions**

- State Transition Matrix: $\Phi(k) = \Phi(k, k-1)$
- Extrapolates state vector from time step $k-1$ to time step $k$ (pulling from past to present)
    - If $\Delta t = t_k - t_{k-1}$ is constant for all $k$, then we might write $\Phi(\Delta t)$
- Tells you how the state would evolve **in the absence of forcing functions**
    - Remember: this is still just a model

# Understanding Kalman Filter Notation

## Drilldown on State Dynamics

- State Transition Matrix: $\Phi(k) = \Phi(k, k-1)$
- Extrapolates state vector from time step $k-1$ to time step $k$ (pulling from past to present)
    - If $\Delta t = t_k - t_{k-1}$ is constant for all $k$, then we might write $\Phi(\Delta t)$
- Tells you how the state would evolve **in the absence of forcing functions**
    - Remember: this is still just a model
    - We don't really know what the true state is, because the random forcing function (process noise) generates disturbances!

- State Transition Matrix: $\Phi(k) = \Phi(k, k-1)$
- Extrapolates state vector from time step $k-1$ to time step $k$ (pulling from past to present)
    - If $\Delta t = t_k - t_{k-1}$ is constant for all $k$, then we might write $\Phi(\Delta t)$
- Tells you how the state would evolve **in the absence of forcing functions**
    - Remember: this is still just a model
    - We don't really know what the true state is, because the random forcing function (process noise) generates disturbances!

# Understanding Kalman Filter Notation

## Drilldown on State Dynamics

- Other factors and terms often play a role in the full state dynamics model

- Other factors and terms often play a role in the full state dynamics model
    - A matrix $\Gamma(k)$ to distribute the Gaussian white noise to the states

- Other factors and terms often play a role in the full state dynamics model
    - A matrix $\Gamma(k)$ to distribute the Gaussian white noise to the states
    - An additional, deterministic control variable $u(k)$, and associated distribution matrix $\Lambda(k)$

- Other factors and terms often play a role in the full state dynamics model
  - A matrix $\Gamma(k)$ to distribute the Gaussian white noise to the states
  - An additional, deterministic control variable $u(k)$, and associated distribution matrix $\Lambda(k)$
  - Then the full state dynamics equation would look like this:

$$x(k) = \Phi(k)x(k-1) + \Lambda(k)u(k-1) + \Gamma(k)w(k)$$

- Other factors and terms often play a role in the full state dynamics model
  - A matrix $\Gamma(k)$ to distribute the Gaussian white noise to the states
  - An additional, deterministic control variable $u(k)$, and associated distribution matrix $\Lambda(k)$
  - Then the full state dynamics equation would look like this:

$$x(k) = \Phi(k)x(k-1) + \Lambda(k)u(k-1) + \Gamma(k)w(k)$$

- In order to focus primarily on the estimation problem, we will generally omit the control variable

- Other factors and terms often play a role in the full state dynamics model
  - A matrix $\Gamma(k)$ to distribute the Gaussian white noise to the states
  - An additional, deterministic control variable $u(k)$, and associated distribution matrix $\Lambda(k)$
  - Then the full state dynamics equation would look like this:

$$x(k) = \Phi(k)x(k-1) + \Lambda(k)u(k-1) + \Gamma(k)w(k)$$

- In order to focus primarily on the estimation problem, we will generally omit the control variable
- Control is obviously an important part of engineering system solutions, so we will consider it occasionally

- Other factors and terms often play a role in the full state dynamics model
  - A matrix $\Gamma(k)$ to distribute the Gaussian white noise to the states
  - An additional, deterministic control variable $u(k)$, and associated distribution matrix $\Lambda(k)$
  - Then the full state dynamics equation would look like this:

$$x(k) = \Phi(k)x(k-1) + \Lambda(k)u(k-1) + \Gamma(k)w(k)$$

- In order to focus primarily on the estimation problem, we will generally omit the control variable
- Control is obviously an important part of engineering system solutions, so we will consider it occasionally

# The Kalman Filter Solution

## State Estimate Extrapolation

- Given an estimate $\hat{x}^+(k-1)$ at time $k-1$, we extrapolate it forward in time by the state transition matrix:

$$\hat{x}^-(k) = \Phi(k)\hat{x}^+(k-1) \tag{1}$$

- Given an estimate $\hat{x}^+(k-1)$ at time $k-1$, we extrapolate it forward in time by the state transition matrix:

$$\hat{x}^-(k) = \Phi(k)\hat{x}^+(k-1) \tag{1}$$

- Q: Why isn't the white noise sequence $w$ included in this extrapolation?

- Given an estimate $\hat{x}^+(k-1)$ at time $k-1$, we extrapolate it forward in time by the state transition matrix:

$$\hat{x}^-(k) = \Phi(k)\hat{x}^+(k-1) \tag{1}$$

- Q: Why isn't the white noise sequence $w$ included in this extrapolation?
  A: Because the white noise forcing function is a **zero-mean** process. Thus, **on average**, the equation is homogeneous and we should therefore use the homogeneous solution, i.e., the state transition matrix.

- Given an estimate $\hat{x}^+(k-1)$ at time $k-1$, we extrapolate it forward in time by the state transition matrix:

$$\hat{x}^-(k) = \Phi(k)\hat{x}^+(k-1) \tag{1}$$

- Q: Why isn't the white noise sequence $w$ included in this extrapolation?
  A: Because the white noise forcing function is a **zero-mean** process. Thus, **on average**, the equation is homogeneous and we should therefore use the homogeneous solution, i.e., the state transition matrix.

1

- Q: How certain are we that our solution is right?

1

- Q: How certain are we that our solution is right?
  A: The answer is given by the **estimation error covariance matrix**, often for brevity just called the covariance matrix $P(k)$

1

- Q: How certain are we that our solution is right?
  A: The answer is given by the **estimation error covariance matrix**, often for brevity just called the covariance matrix $P(k)$[1]

1

- Q: How certain are we that our solution is right?
  A: The answer is given by the **estimation error covariance matrix**, often for brevity just called the covariance matrix $P(k)$[1]

[1] There are two other covariance matrices involved in KF, but $P$ is often referred to as THE covariance matrix.

- Q: How certain are we that our solution is right?
  A: The answer is given by the **estimation error covariance matrix**, often for brevity just called the covariance matrix $P(k)$[1]
- Like the state estimate, the covariance matrix is propagated forward in time before being corrected:

[1] There are two other covariance matrices involved in KF, but $P$ is often referred to as THE covariance matrix.

- Q: How certain are we that our solution is right?
  A: The answer is given by the **estimation error covariance matrix**, often for brevity just called the covariance matrix $P(k)$[1]
- Like the state estimate, the covariance matrix is propagated forward in time before being corrected:

$$P^{-}(k) = \Phi(k) P^{+}(k-1) \Phi(k)^{\top} + Q(k) \tag{2}$$

  where $Q(k)$ is a matrix called the **process noise covariance matrix**

[1] There are two other covariance matrices involved in KF, but $P$ is often referred to as THE covariance matrix.

# The Kalman Filter Solution

## Covariance Extrapolation

- Q: How certain are we that our solution is right?
  A: The answer is given by the **estimation error covariance matrix**, often for brevity just called the covariance matrix $P(k)$[1]
- Like the state estimate, the covariance matrix is propagated forward in time before being corrected:

$$P^{-}(k) = \Phi(k)P^{+}(k-1)\,\Phi(k)^{\top} + Q(k) \tag{2}$$

  where $Q(k)$ is a matrix called the **process noise covariance matrix**
- $Q(k)$ measures the amount of dispersion in the white noise forcing function $w(k)$

[1] There are two other covariance matrices involved in KF, but $P$ is often referred to as THE covariance matrix.

- Q: How certain are we that our solution is right?
  A: The answer is given by the **estimation error covariance matrix**, often for brevity just called the covariance matrix $P(k)$[1]
- Like the state estimate, the covariance matrix is propagated forward in time before being corrected:

$$P^{-}(k) = \Phi(k) P^{+}(k-1) \Phi(k)^{\top} + Q(k) \tag{2}$$

  where $Q(k)$ is a matrix called the **process noise covariance matrix**
- $Q(k)$ measures the amount of dispersion in the white noise forcing function $w(k)$
- Later we'll learn how to define $P(k)$ (at least initially) in terms of the expected error in the estimation $\tilde{x}^{+}(k) = \hat{x}^{+}(k) - x(k)$

[1] There are two other covariance matrices involved in KF, but $P$ is often referred to as THE covariance matrix.

- Q: How certain are we that our solution is right?
  A: The answer is given by the **estimation error covariance matrix**, often for brevity just called the covariance matrix $P(k)$[1]
- Like the state estimate, the covariance matrix is propagated forward in time before being corrected:

$$P^-(k) = \Phi(k)P^+(k-1)\,\Phi(k)^\top + Q(k) \tag{2}$$

  where $Q(k)$ is a matrix called the **process noise covariance matrix**
- $Q(k)$ measures the amount of dispersion in the white noise forcing function $w(k)$
- Later we'll learn how to define $P(k)$ (at least initially) in terms of the expected error in the estimation $\tilde{x}^+(k) = \hat{x}^+(k) - x(k)$

[1] There are two other covariance matrices involved in KF, but $P$ is often referred to as THE covariance matrix.

- What does Eq. (2) really mean?

$$P^{-}(k) = \Phi(k) P^{+}(k-1) \Phi(k)^{\top} + Q(k)$$

- What does Eq. (2) really mean?

$$P^{-}(k) = \Phi(k) P^{+}(k-1)\,\Phi(k)^{\top} + Q(k)$$

- The term $\Phi(k) P^{+}(k-1)\,\Phi(k)^{\top}$ represents the effect that the state dynamics has on the estimation error covariance

# The Kalman Filter Solution

## Covariance Extrapolation

- What does Eq. (2) really mean?

$$P^{-}(k) = \Phi(k) P^{+}(k-1)\, \Phi(k)^{\top} + Q(k)$$

- The term $\Phi(k) P^{+}(k-1)\, \Phi(k)^{\top}$ represents the effect that the state dynamics has on the estimation error covariance
  - P(k) is a positive definite matrix and $\Phi$ tends to increase its norm

- What does Eq. (2) really mean?

$$P^{-}(k) = \Phi(k)P^{+}(k-1)\,\Phi(k)^{\top} + Q(k)$$

- The term $\Phi(k)P^{+}(k-1)\,\Phi(k)^{\top}$ represents the effect that the state dynamics has on the estimation error covariance
    - P(k) is a positive definite matrix and $\Phi$ tends to increase its norm
- The other term $Q(k)$ represents the increased uncertainty added at each step due to the process noise inherent in the system (like all covariance matrices, it is also positive definite)

- What does Eq. (2) really mean?

$$P^{-}(k) = \Phi(k)P^{+}(k-1)\,\Phi(k)^{\top} + Q(k)$$

- The term $\Phi(k)P^{+}(k-1)\,\Phi(k)^{\top}$ represents the effect that the state dynamics has on the estimation error covariance
  - P(k) is a positive definite matrix and $\Phi$ tends to increase its norm
- The other term $Q(k)$ represents the increased uncertainty added at each step due to the process noise inherent in the system (like all covariance matrices, it is also positive definite)

- Where are we?

- Where are we?
  - Drilled down on state dynamics

- Where are we?
  - Drilled down on state dynamics
  - Introduced the $\Phi$ and $Q$ matrices and discussed time extrapolation of $P$

- Where are we?
    - Drilled down on state dynamics
    - Introduced the $\Phi$ and $Q$ matrices and discussed time extrapolation of $P$
- What's next?

- Where are we?
  - Drilled down on state dynamics
  - Introduced the $\Phi$ and $Q$ matrices and discussed time extrapolation of $P$
- What's next?
  - Drill down on the Kalman gain equation as a function of $P, Q, R$

- Where are we?
  - Drilled down on state dynamics
  - Introduced the $\Phi$ and $Q$ matrices and discussed time extrapolation of $P$
- What's next?
  - Drill down on the Kalman gain equation as a function of $P, Q, R$
  - Present the 5 Kalman equations for the general solution

- Where are we?
  - Drilled down on state dynamics
  - Introduced the $\Phi$ and $Q$ matrices and discussed time extrapolation of $P$
- What's next?
  - Drill down on the Kalman gain equation as a function of $P, Q, R$
  - Present the 5 Kalman equations for the general solution