Stima & Filtraggio: Lab 2

Giacomo Baggio

Dipartimento di Ingegneria dell'Informazione
Università degli Studi di Padova

✉ `baggio@dei.unipd.it`

`baggio.dei.unipd.it/~teaching`

April 19, 2017

Today's Lab

Kalman Filtering & Applications

1. Recap on Systems Theory

2. Kalman Filter & Predictor

Today's Lab

Kalman Filtering & Applications

1. Recap on Systems Theory (in **MATLAB**®)
2. Kalman Filter & Predictor (in **MATLAB**®)

Today's Lab

Kalman Filtering & Applications

1. Recap on Systems Theory (⏱ **45 min**)
2. Kalman Filter & Predictor (⏱ **45 min**)

Today's Lab

Kalman Filtering & Applications

1. Recap on Systems Theory
   - State space representation
   - Internal/external stability
   - Reachability/Stabilizability & Observability/Detectability

State Space systems
(continuous-time)

state vector

input vector

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t) \end{cases} \qquad x(0) = x_0$$

output vector

initial condition

State Space systems
(continuous-time)

state vector ***input vector***

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) + Du(t) \end{cases} \qquad x(0) = x_0$$

output vector ***initial condition***

State **S**pace

$$\left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right]$$

Transfer **F**unction

$$W(s) = C(sI - A)^{-1}B + D$$

State Space systems

(discrete-time)

state vector ***input vector***

$$\begin{cases} x(t+1) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{cases} \qquad x(0) = x_0$$

output vector ***initial condition***

State Space systems
(discrete-time)

state vector · *input vector*

$$\begin{cases} x(t+1) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{cases} \qquad x(0) = x_0$$

output vector · *initial condition*

State **S**pace

$$\left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right]$$

Transfer **F**unction

$$W(z) = C(zI - A)^{-1}B + D$$

State Space systems *in MATLAB®*

(from Control System Toolbox)

Continuous-time case

```
>> sys_c = ss(mA,mB,mC,mD)
```

Discrete-time case

```
>> sys_d = ss(mA,mB,mC,mD,dTs)
```

State Space systems *in MATLAB®*

(from Control System Toolbox)

Continuous-time case

```
>> sys_c = ss(mA,mB,mC,mD)
```

Discrete-time case

```
>> sys_d = ss(mA,mB,mC,mD,dTs)
```

sampling period
`dTs = -1`: not specified

State Space systems *in MATLAB®*

(from Control System Toolbox)

Continuous-time case	`>> sys_c = ss(mA,mB,mC,mD)`
Discrete-time case	`>> sys_d = ss(mA,mB,mC,mD,dTs)`
Recover A, B, C, D	`>> [mA,mB,mC,mD] = ssdata(sys)`
From SS to TF	`>> sys_tf = tf(sys_ss)`
From SS to ZPK	`>> sys_zpk = zpk(sys_ss)`

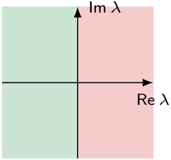
Stability
(continuous-time)

$$\Sigma : \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] \qquad \begin{array}{l} A \in \mathbb{R}^{n\times n},\ B \in \mathbb{R}^{n\times m}, \\ C \in \mathbb{R}^{p\times n},\ D \in \mathbb{R}^{p\times m} \end{array}$$

$$W(s) = C(sI - A)^{-1}B + D \xmapsto{\text{after zeros/poles cancellations}} \tilde{W}(s)$$

Σ Hurwitz stable

internally — Re $\lambda(A) < 0$



externally — Re poles$\{\tilde{W}(s)\} < 0$

Stability
(discrete-time)

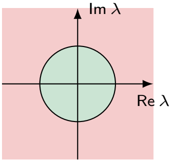
$$\Sigma : \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] \qquad \begin{array}{l} A \in \mathbb{R}^{n\times n},\ B \in \mathbb{R}^{n\times m}, \\ C \in \mathbb{R}^{p\times n},\ D \in \mathbb{R}^{p\times m} \end{array}$$

$$W(z) = C(zI - A)^{-1}B + D \xmapsto{\text{after zeros/poles cancellations}} \tilde{W}(z)$$

Σ *Schur stable*

internally → $|\lambda(A)| < 1$

externally → $|\text{poles}\{\tilde{W}(z)\}| < 1$



Stability
in MATLAB®

Eigenvalues of A `>> eig(mA)`

Minimal realization `>> sys_min = minreal(sys)`

N.B. Minimal realization of Σ = state space realization of Σ with smallest possible state dimension

Reachability & Observability
(continuous-time & discrete-time)

$$\Sigma : \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] \qquad \begin{array}{l} A \in \mathbb{R}^{n\times n},\ B \in \mathbb{R}^{n\times m}, \\ C \in \mathbb{R}^{p\times n},\ D \in \mathbb{R}^{p\times m} \end{array}$$

- (A, B) ***reachable***
  - $$\operatorname{rank} \left[\begin{array}{c|c|c|c|c} A & AB & A^2B & \cdots & A^{n-1}B \end{array}\right] = n$$
    (***reachability matrix***)
  - $$\operatorname{rank} \left[\begin{array}{c|c} A - zI & B \end{array}\right] = n, \ \ \forall z \in \lambda(A)$$ (***PBH test***)

Reachability & Observability
(continuous-time & discrete-time)

$$\Sigma : \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] \qquad \begin{array}{l} A \in \mathbb{R}^{n\times n},\ B \in \mathbb{R}^{n\times m}, \\ C \in \mathbb{R}^{p\times n},\ D \in \mathbb{R}^{p\times m} \end{array}$$

- (A, C) ***observable***
  - $$\operatorname{rank}\left[\begin{array}{c} A \\ \hline CA \\ \hline CA^2 \\ \hline \vdots \\ \hline CA^{n-1} \end{array}\right] = n$$ (***observability matrix***)
  - $$\operatorname{rank}\left[\begin{array}{c} A - zI \\ \hline C \end{array}\right] = n, \ \forall z \in \lambda(A)$$ (***PBH test***)

Reachability & Observability *in MATLAB®*

Reachability matrix
```
>> mR = ctrb(mA,mB)
>> mR = ctrb(sys)
```

Observability matrix
```
>> mO = obsv(mA,mC)
>> mO = obsv(sys)
```

(Rank of a matrix X `>> iRank = rank(mX)`)

Stabilizability & Detectability
(continuous-time)

$$\Sigma : \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] \qquad \begin{array}{l} A \in \mathbb{R}^{n\times n},\ B \in \mathbb{R}^{n\times m}, \\ C \in \mathbb{R}^{p\times n},\ D \in \mathbb{R}^{p\times m} \end{array}$$

- (A, B) ***stabilizable***

$$\operatorname{rank}\left[\, A - sI \mid B \,\right] = n, \ \ \forall s \in \lambda(A) \ \text{ s.t. } \boxed{\operatorname{Re} s \geq 0}$$

Stabilizability & Detectability
(continuous-time)

$$\Sigma : \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] \qquad \begin{array}{l} A \in \mathbb{R}^{n\times n},\ B \in \mathbb{R}^{n\times m}, \\ C \in \mathbb{R}^{p\times n},\ D \in \mathbb{R}^{p\times m} \end{array}$$

- (A, C) ***detectable***

$$\operatorname{rank}\left[\begin{array}{c} A - sI \\ \hline C \end{array}\right] = n, \quad \forall s \in \lambda(A) \ \text{ s.t. } \boxed{\operatorname{Re} s \geq 0}$$

Stabilizability & Detectability
(discrete-time)

$$\Sigma : \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] \qquad \begin{array}{l} A \in \mathbb{R}^{n\times n},\ B \in \mathbb{R}^{n\times m}, \\ C \in \mathbb{R}^{p\times n},\ D \in \mathbb{R}^{p\times m} \end{array}$$

- (A, B) ***stabilizable***

  - $$\text{rank}\left[\begin{array}{c|c} A - zI & B \end{array}\right] = n, \ \ \forall z \in \lambda(A) \ \text{ s.t. } \boxed{|z| \geq 1}$$

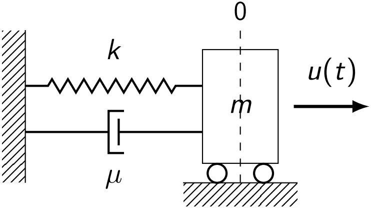
Stabilizability & Detectability
(discrete-time)

$$\Sigma : \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] \qquad \begin{array}{l} A \in \mathbb{R}^{n\times n},\ B \in \mathbb{R}^{n\times m}, \\ C \in \mathbb{R}^{p\times n},\ D \in \mathbb{R}^{p\times m} \end{array}$$

- (A, C) ***detectable***

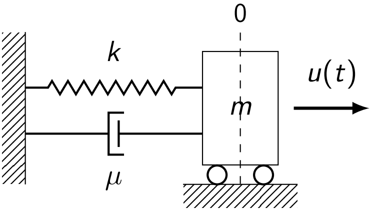
$$\operatorname{rank} \left[\begin{array}{c} A - zI \\ \hline C \end{array}\right] = n, \ \ \forall z \in \lambda(A) \ \text{ s.t. } \boxed{|z| \geq 1}$$

A worked example



- Dynamical equation:
  $m\ddot{x}(t) = -kx(t) - \mu\dot{x}(t) + u(t)$
- Measured output:
  Position $x(t)$

A worked example



- Dynamical equation:
  $m\ddot{x}(t) = -kx(t) - \mu\dot{x}(t) + u(t)$
- Measured output:
  Position $x(t)$

$$
\begin{cases}
\begin{bmatrix} \dot{x}(t) \\ \ddot{x}(t) \end{bmatrix} = \overbrace{\begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{\mu}{m} \end{bmatrix}}^{A} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} + \overbrace{\begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix}}^{B} u(t) \\
y(t) = \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{C} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix}
\end{cases}
$$

A worked example

Pick $m = 1$, $\mu = 0.5$, $k = 2$

$$\begin{cases} \begin{bmatrix} \dot{x}(t) \\ \ddot{x}(t) \end{bmatrix} = \underbrace{\overset{A}{\begin{bmatrix} 0 & 1 \\ -2 & -0.5 \end{bmatrix}}}_{} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} + \overset{B}{\begin{bmatrix} 0 \\ 1 \end{bmatrix}} u(t) \\ y(t) = \underset{C}{\begin{bmatrix} 1 & 0 \end{bmatrix}} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} \end{cases}$$

A worked example

Pick $m = 1$, $\mu = 0.5$, $k = 2$

$$\begin{cases} \begin{bmatrix} \dot{x}(t) \\ \ddot{x}(t) \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 \\ -2 & -0.5 \end{bmatrix}}^{A} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} + \overbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}^{B} u(t) \\ y(t) = \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{C} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} \end{cases}$$

In MATLAB®...

```
>> sys = ss(mA,mB,mC,[]);
```

A worked example

Pick $m = 1$, $\mu = 0.5$, $k = 2$

$$\begin{cases} \begin{bmatrix} \dot{x}(t) \\ \ddot{x}(t) \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 \\ -2 & -0.5 \end{bmatrix}}_{A} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} + \underbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}_{B} u(t) \\ y(t) = \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{C} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} \end{cases}$$

Is the system internally stable?

```
>> eig(mA)
ans =
   -0.2500 + 1.3919i
   -0.2500 - 1.3919i
```

Yes!

A worked example

Pick $m = 1$, $\mu = 0.5$, $k = 2$

$$\begin{cases} \begin{bmatrix} \dot{x}(t) \\ \ddot{x}(t) \end{bmatrix} = \overbrace{\begin{bmatrix} 0 & 1 \\ -2 & -0.5 \end{bmatrix}}^{A} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} + \overbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}^{B} u(t) \\ y(t) = \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{C} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} \end{cases}$$

Is the system externally stable?

```
>> tf(sys)
ans =
              1
    ----------------
     s^2 + 0.5 s + 2
```

Yes!

A worked example

Pick $m = 1$, $\mu = 0.5$, $k = 2$

$$\begin{cases} \begin{bmatrix} \dot{x}(t) \\ \ddot{x}(t) \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 \\ -2 & -0.5 \end{bmatrix}}_{A} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} + \underbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}_{B} u(t) \\ y(t) = \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{C} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} \end{cases}$$

Is the system reachable?

```
>> rank(ctrb(sys))
ans =
      2
```

Yes! (...and stabilizable!)

A worked example

Pick $m = 1$, $\mu = 0.5$, $k = 2$

$$\begin{cases} \begin{bmatrix} \dot{x}(t) \\ \ddot{x}(t) \end{bmatrix} = \overset{A}{\begin{bmatrix} 0 & 1 \\ -2 & -0.5 \end{bmatrix}} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} + \overset{B}{\begin{bmatrix} 0 \\ 1 \end{bmatrix}} u(t) \\ y(t) = \underset{C}{\begin{bmatrix} 1 & 0 \end{bmatrix}} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} \end{cases}$$

Is the system observable?

```
>> rank(obsv(sys))
ans =
      2
```

Yes! (...and detectable!)

Other useful functions from CST

Impulse response	`>> [cvY,cvT] = impulse(sys)`
Step response	`>> [cvY,cvT] = step(sys)`
Bode plot	`>> bode(sys)`
Zero/pole plot	`>> pzmap(sys)`
Output response	`>> cvY = lsim(sys,cvU,cvT,cvX0)`

✎ *Practice time 1!*

Ex 1.1. Create a function

```
[bInt,bExt] = checkStability(mA,mB,mC,mD,strSysType)
```

that has as inputs matrices $\mathtt{mA} \in \mathbb{R}^{n \times n}$, $\mathtt{mB} \in \mathbb{R}^{n \times m}$, $\mathtt{mC} \in \mathbb{R}^{p \times n}$, $\mathtt{mD} \in \mathbb{R}^{p \times m}$, and a string `strSysType` that can be set to either `‘continuous’` or `‘discrete’` depending on the type of system considered. The function returns

- boolean `bInt = true` if the system ***internally stable*** and `bInt = false` otherwise.
- boolean `bExt = true` if the system is ***externally stable*** and `bExt = false` otherwise.

✎ *Practice time 1!*

Ex 1.2. Create a function

```
[bReach,bStab] = checkReachStab(mA,mB,strSysType)
```

that has as inputs matrices mA $\in \mathbb{R}^{n\times n}$, mB $\in \mathbb{R}^{n\times m}$ and a string strSysType that can be set to either `'continuous'` or `'discrete'` depending on the type of system considered.
The function returns

- boolean `bReach = true` if $(\mathtt{mA}, \mathtt{mB})$ is ***reachable*** and `bReach = false` otherwise.
- boolean `bStab = true` if $(\mathtt{mA}, \mathtt{mB})$ is ***stabilizable*** and `bStab = false` otherwise.

Practice time 1!

Ex 1.3. Create a function

```
[bObs,bDetec] = checkObsDetec(mA,mC,strSysType)
```

that has as inputs matrices $\mathtt{mA} \in \mathbb{R}^{n\times n}$, $\mathtt{mC} \in \mathbb{R}^{p\times n}$ and a string `strSysType` that can be set to either `‘continuous’` or `‘discrete’` depending on the type of system considered.
The function returns

- boolean `bObs = true` if $(\mathtt{mA}, \mathtt{mC})$ is ***observable*** and `bObs = false` otherwise.
- boolean `bDetec = true` if $(\mathtt{mA}, \mathtt{mC})$ is ***detectable*** and `bDetec = false` otherwise.

Today's Lab

Kalman Filtering & Applications

2 Kalman Filter & Predictor

- Quick recap
- Steady state behavior
- **MATLAB®** tools

Setup

The model

$$\begin{cases} x(t+1) &= Ax(t) + v(t) \\ y(t) &= Cx(t) + w(t) \end{cases} \qquad x(0) = x_0$$

Setup

The model

$$\begin{cases} x(t+1) &= Ax(t) + v(t) \\ y(t) &= Cx(t) + w(t) \end{cases} \qquad x(0) = x_0$$

Standing assumptions

- $\mathbb{E}\left\{ \begin{bmatrix} v(t) \\ w(t) \end{bmatrix} \begin{bmatrix} v^\top(s) & w^\top(s) \end{bmatrix} \right\} = \begin{bmatrix} Q & S \\ S^\top & R \end{bmatrix} \delta(t-s), \quad R > 0$
- $\mathbb{E}\left\{ x_0 \begin{bmatrix} v^\top(t) & w^\top(t) \end{bmatrix} \right\} = 0, \ \forall t \geq 0$
- $\mathbb{E}\{x_0\} = \mu_0, \quad \mathrm{Var}\{x_0\} = P_0$

Setup

An equivalent model...

$$\begin{cases} x(t+1) & = Fx(t) + SR^{-1}y(t) + \tilde{v}(t) \\ y(t) & = Cx(t) + w(t) \end{cases} \qquad x(0) = x_0$$

- $F := A - SR^{-1}C$
- $\tilde{v}(t) := v(t) - \hat{\mathbb{E}}[v(t) \,|\, w(t)] = v(t) - SR^{-1}(y(t) - Cx(t))$
- $\tilde{v}(t) \perp w(t), \quad \mathrm{Var}\, \tilde{v}(t) = \tilde{Q} := Q - SR^{-1}S^\top$

Kalman Filtering equations

Initial definitions

$$P(t|t-1) := \mathrm{Var}\,\tilde{x}(t|t-1), \qquad P(t|t) := \mathrm{Var}\,\tilde{x}(t|t)$$

(prediction error covariance) ***(estimation error covariance)***

$$\Lambda(t) := CP(t|t-1)C^\top + R, \qquad L(t) := P(t|t-1)C^\top\Lambda^{-1}(t)$$

(innovation process covariance) ***(filter gain)***

Initial conditions

$$\hat{x}(0|-1) := \mu_0, \quad P(0|-1) := P_0$$

Kalman Filtering equations

• *Estimation* •

$$\hat{x}(t|t) = \hat{x}(t|t-1) + L(t)(y(t) - C\hat{x}(t|t-1))$$

$$\begin{aligned} P(t|t) &= P(t|t-1) - P(t|t-1)C^\top \Lambda(t)^{-1} C P(t|t-1) \\ &= (I - L(t)C)P(t|t-1)(I - L(t)C)^\top + L(t)RL^\top(t) \end{aligned}$$

• *Prediction* •

$$\hat{x}(t+1|t) = F\hat{x}(t|t) + SR^{-1}y(t)$$

$$P(t+1|t) = FP(t|t)F^\top + \tilde{Q}$$

One-step Kalman predictor

By decoupling the previous equations...

- $\hat{x}(t+1|t) = A\hat{x}(t|t-1) + G(t)(y(t) - C\hat{x}(t|t-1))$
- $P(t+1|t) = \Gamma(t)P(t|t-1)\Gamma^\top(t) + K(t)RK^\top(t) + \tilde{Q}$

where...

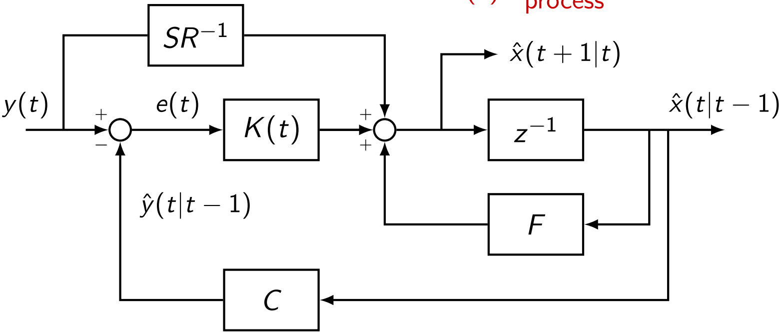
- $K(t) := FL(t)$ ***(Kalman gain)***
- $G(t) := K(t) + SR^{-1}$ ***(predictor gain)***
- $\Gamma(t) := A - G(t)C = F - K(t)C = F - (I - L(t)C)$ ***(closed-loop matrix)***

One-step Kalman predictor

Block diagram representation

★ $\hat{x}(t+1|t) = F\hat{x}(t|t-1) + K(t)\underbrace{(y(t) - C\hat{x}(t|t-1))}_{e(t)} + SR^{-1}y(t)$

$e(t)$ innovation process

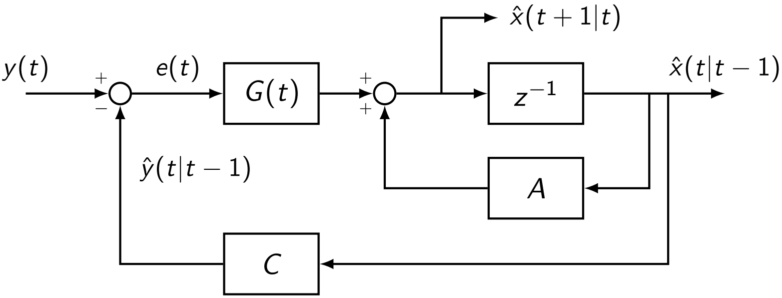


One-step Kalman predictor

Block diagram representation

★ $\hat{x}(t+1|t) = A\hat{x}(t|t-1) + G(t)(y(t) - C\hat{x}(t|t-1))$

$G(t) = K(t) + SR^{-1}$

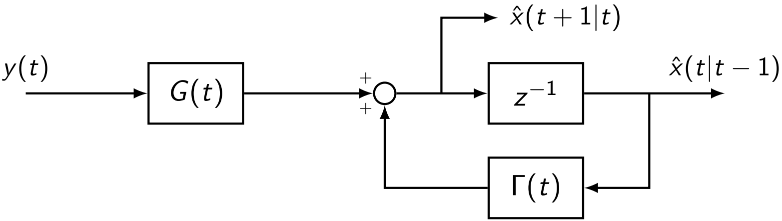


One-step Kalman predictor

Block diagram representation

★ $\hat{x}(t+1|t) = \Gamma(t)\hat{x}(t|t-1) + G(t)y(t)$

$$\Gamma(t) = A - G(t)C$$



Steady-state one-step Kalman predictor

N.B. The steady-state prediction error covariance satisfies

$$\bar{P} = F\bar{P}F^\top - F\bar{P}C^\top(C\bar{P}C^\top + R)^{-1}C\bar{P}F^\top + \tilde{Q} \quad \textbf{(DARE)}$$

Steady-state one-step Kalman predictor

N.B. The steady-state prediction error covariance satisfies

$$\bar{P} = F\bar{P}F^\top - F\bar{P}C^\top(C\bar{P}C^\top + R)^{-1}C\bar{P}F^\top + \tilde{Q} \quad \textbf{(DARE)}$$

Fundamental Theorem of KF Theory:

(F, C) detectable & $(F, \tilde{Q}^{\frac{1}{2}})$ stabilizable

$$\Updownarrow$$

- $\exists!\ \bar{P} = \bar{P}^\top$ of (DARE)
- $\bar{P}$ stabilizing
- $\lim_{t\to\infty} P(t) = \bar{P}$, $\forall\, P_0 = P_0^\top \geq 0$

Steady-state one-step Kalman predictor

- ★ $\hat{x}_\infty(t+1|t) = A\hat{x}_\infty(t|t-1) + \bar{G}(y(t) - C\hat{x}_\infty(t|t-1))$
- ★ $\bar{P} = \bar{\Gamma}\bar{P}\bar{\Gamma}^\top + \bar{K}R\bar{K} + \tilde{Q}$

where...

- $\bar{K} := F\bar{P}C^\top(C\bar{P}C^\top + R)^{-1}$ ***(steady-state Kalman gain)***
- $\bar{G} := \bar{K} + SR^{-1}$ ***(steady-state predictor gain)***
- $\bar{\Gamma} := A - \bar{G}C = F - \bar{K}C$ ***(steady-state closed-loop matrix)***

Steady-state one-step Kalman predictor

★ $\hat{x}_\infty(t+1|t) = A\hat{x}_\infty(t|t-1) + \bar{G}(y(t) - C\hat{x}_\infty(t|t-1))$

★ $\bar{P} = \bar{\Gamma}\bar{P}\bar{\Gamma}^\top + \bar{K}R\bar{K} + \tilde{Q}$

where...

- $\bar{K} := F\bar{P}C^\top(C\bar{P}C^\top + R)^{-1}$ ***(steady-state Kalman gain)***
- $\bar{G} := \bar{K} + SR^{-1}$ ***(steady-state predictor gain)***
- $\bar{\Gamma} := A - \bar{G}C = F - \bar{K}C$ ***(steady-state closed-loop matrix)***

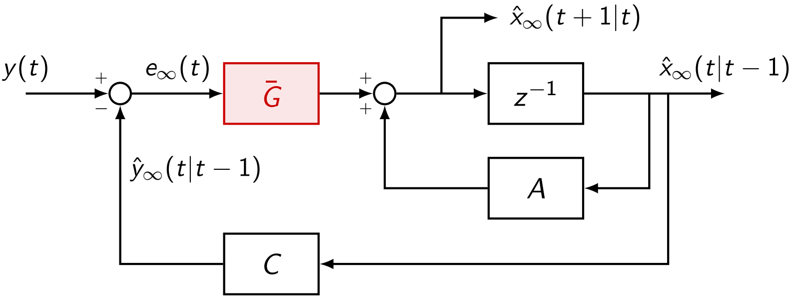
N.B. If A stable, $\bar{P} = \bar{\Sigma} - \hat{\Sigma}_\infty$ with $\hat{\Sigma}_\infty := \mathrm{Var}\,\hat{x}_\infty(t|t-1)$ and $\bar{\Sigma}$ sol. of

$$\bar{\Sigma} = A\bar{\Sigma}A^\top + Q \qquad \textbf{(DALE)}$$

Steady-state one-step Kalman predictor

★ $\hat{x}_\infty(t+1|t) = A\hat{x}_\infty(t|t-1) + \bar{G}\big(y(t) - C\hat{x}_\infty(t|t-1)\big)$

$e_\infty(t)$



MATLAB® tools for Kalman Filtering

DALE `>> X = dlyap(A,Q)`

```
>> help dlyap

dlyap Solve discrete Lyapunov equations.

X = dlyap(A,Q) solves the discrete Lyapunov matrix
equation:

    A*X*A' - X + Q = 0
```

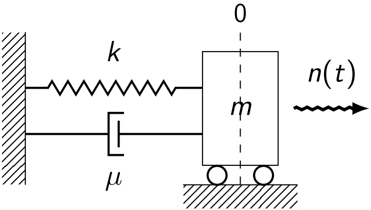
MATLAB® tools for Kalman Filtering

DARE `>> [mX,mL,mG] = dare(mA,mB,mQ,mR,mS,mE)`

```
>> help dare
dare Solve discrete-time algebraic Riccati
equations.
[X,L,G] = dare(A,B,Q,R,S,E) computes the unique
stabilizing solution X of the discrete-time
algebraic Riccati equation

  E'XE = A'XA - (A'XB + S)(B'XB + R)^-1 (A'XB + S)' + Q
```

A worked example



- Dynamical equation:

  $m\ddot{x}(t) = -kx(t) - \mu\dot{x}(t) + n(t)$

  $\mathbb{E}\{n(t)n(s)\} = \sigma_n^2\,\delta(t-s)$

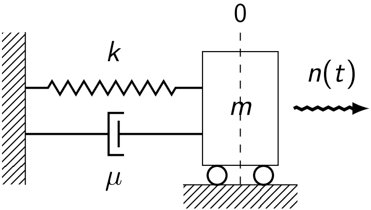
- Measured output:

  Noisy position $x(t) + w(t)$

  $\mathbb{E}\{w(t)w(s)\} = \sigma_R^2\,\delta(t-s)$

  $v(t) \perp w(s)$, $\forall t, s \geq 0$

A worked example



- Dynamical equation:

  $m\ddot{x}(t) = -kx(t) - \mu\dot{x}(t) + n(t)$

  $\mathbb{E}\{n(t)n(s)\} = \sigma_n^2\,\delta(t-s)$

- Measured output:

  Noisy position $x(t) + w(t)$

  $\mathbb{E}\{w(t)w(s)\} = \sigma_R^2\,\delta(t-s)$

  $v(t) \perp w(s)$, $\forall t, s \geq 0$

Task: W.r.t. the ***sampled system*** (sampling period $T_s = 1\,\mathrm{s}$), **(i)** write down the ***steady-state Kalman predictor*** equation for the position $\hat{x}_\infty(t|t-1)$, and **(ii)** compute the ***steady-state prediction error covariance*** $\bar{P}$.

A worked example

Pick $m = 1$, $\mu = 1$, $k = 2$, $\sigma_n^2 = 1$, $\sigma_R^2 = 1$

$$\begin{cases} \begin{bmatrix} \dot{x}(t) \\ \ddot{x}(t) \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 \\ -2 & -1 \end{bmatrix}}^{\bar{A}} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} + \overbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}^{B} n(t) \\ y(t) = \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{C} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} + \underbrace{w(t)}_{D=1} \end{cases}$$

A worked example

Pick $m = 1$, $\mu = 1$, $k = 2$, $\sigma_n^2 = 1$, $\sigma_R^2 = 1$

$$\begin{cases} \begin{bmatrix} \dot{x}(t) \\ \ddot{x}(t) \end{bmatrix} = \overbrace{\begin{bmatrix} 0 & 1 \\ -2 & -1 \end{bmatrix}}^{\bar{A}} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} + \overbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}^{B} n(t) \\ y(t) = \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{C} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} + \underbrace{w(t)}_{D=1} \end{cases}$$

sampling

$\exp(\bar{A} T_s)$

$v(t) = \int_0^{T_s} e^{\bar{A}\tau} B n(t + T_s - \tau) \, d\tau$

$$\begin{cases} \begin{bmatrix} x(t+1) \\ \dot{x}(t+1) \end{bmatrix} = \begin{bmatrix} 0.3711 & 0.4445 \\ -0.8890 & -0.0734 \end{bmatrix} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} + v(t) \\ y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} + w(t) \end{cases}$$

A worked example

$$
\begin{cases}
\begin{bmatrix} x(t+1) \\ \dot{x}(t+1) \end{bmatrix} = \overbrace{\begin{bmatrix} 0.3711 & 0.4445 \\ -0.8890 & -0.0734 \end{bmatrix}}^{A} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} + \overbrace{v(t)}^{B=1} \\
y(t) = \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{C} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} + \underbrace{w(t)}_{D=1}
\end{cases}
$$

$$
Q = \int_0^{T_s} \exp(\bar{A}\tau) B B^\top \exp(\bar{A}^\top \tau)\, \mathrm{d}\tau = \begin{bmatrix} 0.1168 & 0.0988 \\ 0.0988 & 0.2997 \end{bmatrix}, \quad R = 1
$$

N.B. $v(t) \perp w(s),\ \forall t, s \quad \Rightarrow \quad F = A$ and $\tilde{Q} = Q$!

A worked example

the model

$$\left\{\begin{array}{l} \begin{bmatrix} x(t+1) \\ \dot{x}(t+1) \end{bmatrix} = \overbrace{\begin{bmatrix} 0.3711 & 0.4445 \\ -0.8890 & -0.0734 \end{bmatrix}}^{F} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} + v(t) \qquad B = 1 \\ y(t) = \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{C} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} + w(t) \qquad D = 1 \end{array}\right.$$

noises cov's

$$\tilde{Q} = \begin{bmatrix} 0.1168 & 0.0988 \\ 0.0988 & 0.2997 \end{bmatrix}$$

$$R = 1$$

Is (F, C) detectable?

```
>> rank(obsv(mF,mC))
ans =
      2
```

Yes!

A worked example

the model

$$\begin{cases} \begin{bmatrix} x(t+1) \\ \dot{x}(t+1) \end{bmatrix} = \overbrace{\begin{bmatrix} 0.3711 & 0.4445 \\ -0.8890 & -0.0734 \end{bmatrix}}^{F} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} + v(t) \qquad B = 1 \\ y(t) = \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{C} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} + w(t) \qquad D = 1 \end{cases}$$

noises cov's

$$\tilde{Q} = \begin{bmatrix} 0.1168 & 0.0988 \\ 0.0988 & 0.2997 \end{bmatrix}$$

$$R = 1$$

Is $(F, \tilde{Q}^{\frac{1}{2}})$ stabilizable?

```
>> rank(ctrb(mF,sqrtm(mQtil)))
ans =
      2
```

Yes!

A worked example

the model

$$\begin{cases} \begin{bmatrix} x(t+1) \\ \dot{x}(t+1) \end{bmatrix} = \overbrace{\begin{bmatrix} 0.3711 & 0.4445 \\ -0.8890 & -0.0734 \end{bmatrix}}^{F} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} + \overbrace{v(t)}^{B=1} \\ y(t) = \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{C} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} + \underbrace{w(t)}_{D=1} \end{cases}$$

noises cov's

$$\tilde{Q} = \begin{bmatrix} 0.1168 & 0.0988 \\ 0.0988 & 0.2997 \end{bmatrix}$$

$$R = 1$$

Compute the prediction error state covariance

```
>> mP = dare(mF',mC',mQtil,mR)
mP =
      0.2372    0.0154
      0.0154    0.4553
```

$$\bar{P} = 0.2372$$

A worked example

the model

$$\left\{\begin{array}{ll}\begin{bmatrix}x(t+1)\\ \dot{x}(t+1)\end{bmatrix} &= \overbrace{\begin{bmatrix}0.3711 & 0.4445\\ -0.8890 & -0.0734\end{bmatrix}}^{F}\begin{bmatrix}x(t)\\ \dot{x}(t)\end{bmatrix} + \overbrace{}^{B=1} v(t)\\ y(t) &= \underbrace{\begin{bmatrix}1 & 0\end{bmatrix}}_{C}\begin{bmatrix}x(t)\\ \dot{x}(t)\end{bmatrix} + \underbrace{w(t)}_{D=1}\end{array}\right.$$

noises cov's

$$\tilde{Q} = \begin{bmatrix}0.1168 & 0.0988\\ 0.0988 & 0.2997\end{bmatrix}$$

$$R = 1$$

And the steady-state Kalman predictor is...

$$\begin{bmatrix}\hat{x}_\infty(t+1|t)\\ \hat{\dot{x}}_\infty(t+1|t)\end{bmatrix} = F\begin{bmatrix}\hat{x}_\infty(t+1|t)\\ \hat{\dot{x}}_\infty(t+1|t)\end{bmatrix} + \bar{K}\left(y(t) - C\begin{bmatrix}\hat{x}_\infty(t+1|t)\\ \hat{\dot{x}}_\infty(t+1|t)\end{bmatrix}\right)$$

A worked example

the model

$$
\begin{cases}
\begin{bmatrix} x(t+1) \\ \dot{x}(t+1) \end{bmatrix} = \overbrace{\begin{bmatrix} 0.3711 & 0.4445 \\ -0.8890 & -0.0734 \end{bmatrix}}^{F} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} + \overbrace{v(t)}^{B=1} \\
y(t) = \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{C} \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix} + \underbrace{w(t)}_{D=1}
\end{cases}
$$

noises cov's

$$\tilde{Q} = \begin{bmatrix} 0.1168 & 0.0988 \\ 0.0988 & 0.2997 \end{bmatrix}$$

$$R = 1$$

And the steady-state Kalman predictor is...

$$\hat{x}_\infty(t+1|t) = 0.4445\hat{\dot{x}}_\infty(t+1|t) - 0.0767y(t) + 0.2944\hat{x}_\infty(t+1|t)$$

✎ Practice time 2!

Ex 2.1. Create a function

```
[cvXhat,mP] = predKalman(sys,cvY0,cvX0,mP0)
```

that has as inputs a discrete-time state space system `sys`

$$\begin{cases} x(t+1) & = Ax(t) + Bv(t) \\ y(t) & = Cx(t) + Dw(t) \end{cases}$$

with $v(t)$, $w(t)$ unit variance uncorrelated white noises ($v(t) \perp w(s), \forall t, s$), a measurement vector `cvY0`, a state vector `cvX0`, and an initial prediction error covariance matrix `mP0`.
The function returns

- the one-step Kalman prediction `cvXhat`,
- the prediction error covariance matrix `mP`.

✎ Practice time 2!

Ex 2.2. Create a function

```
[cvXhatSS,mPSS] = predKalmanSS(sys,cvY0,cvX0)
```

that has as inputs a discrete-time state space system `sys`

$$\begin{cases} x(t+1) & = Ax(t) + Bv(t) \\ y(t) & = Cx(t) + Dw(t) \end{cases}$$

with $v(t)$, $w(t)$ unit variance uncorrelated white noises ($v(t) \perp w(s)$, $\forall t, s$), a measurement vector `cvY0`, a state vector `cvX0`. The function returns

- the steady-state one-step Kalman prediction `cvXhatSS`,
- the steady-state prediction error covariance matrix `mPSS`,

whenever these quantities exist. If this is not the case `cvXhatSS` and `mPSS` are left empty.

Practice time 2!

Ex 2.3. Test the functions in Ex 2.1-2.2 with the system described by

$$A = \begin{bmatrix} -1 & 0.5 \\ 0 & 0.5 \end{bmatrix},\ B = \begin{bmatrix} 1 \\ 0 \end{bmatrix},\ C = \begin{bmatrix} 1 & 0.5 \end{bmatrix},\ D = 1.$$

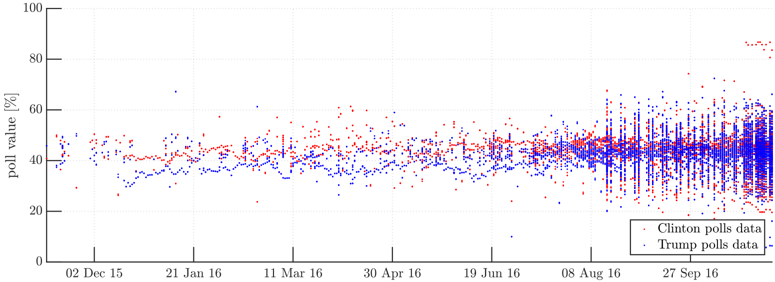
In particular:

- Generate a set of measurement vectors $\{y(t)\}$, $t = 0, 1, \ldots, 30$ using the previous system with $x(0) = [1, 0]^\top$, and uncorrelated noises $v(t) \sim \mathcal{N}(0, 0.1)$, $w(t) \sim \mathcal{N}(0, 0.1)$.
- Use as initial state prediction $\hat{x}(0|-1) \sim \mathcal{N}(\mathbf{0}, I)$ and initial prediction error covariance $P(0|-1) = I$.
- Plot the real trajectory $y(t)$ together with the predicted trajectory $\hat{y}(t+1|t)$ and the steady-state predicted trajectory $\hat{y}_\infty(t+1|t)$ in the interval $t \in [0, 30]$.

Addendum: Kalman filtering on real data
Predicting 2016 US election results

Procedure

1. Getting Clinton/Trump polls data*

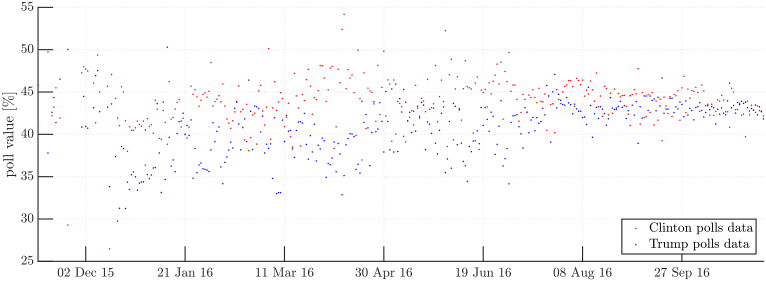


*https://projects.fivethirtyeight.com/2016-election-forecast/national-polls

Addendum: Kalman filtering on real data
Predicting 2016 US election results

Procedure

2. Massaging data (averaging/removing outliers)



Addendum: Kalman filtering on real data
Predicting 2016 US election results

Procedure

3. Modelling polls dynamics

(simplest possible model)

$$\begin{cases} x(t+1) & = x(t) + v(t) \\ y(t) & = x(t) + w(t) \end{cases}$$

$$v(t) \sim \mathcal{N}(0, \sigma_Q),\ w(t) \sim \mathcal{N}(0, \sigma_R),\ v(t) \perp w(s),\ \forall t, s$$

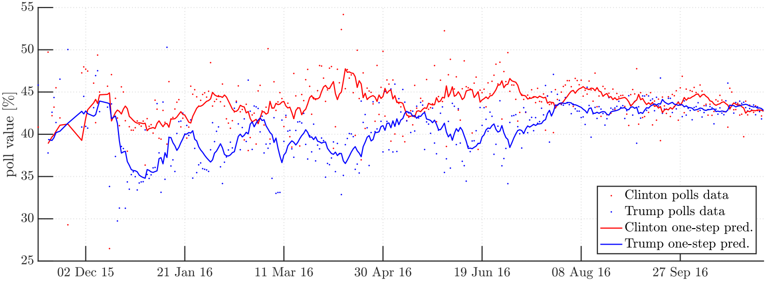
"tuning" parameters $\longrightarrow$ $\sigma_Q = 1,\ \sigma_R = 5$

(my choice)

Addendum: Kalman filtering on real data
Predicting 2016 US election results

Procedure

4. Applying Kalman one-step predictor



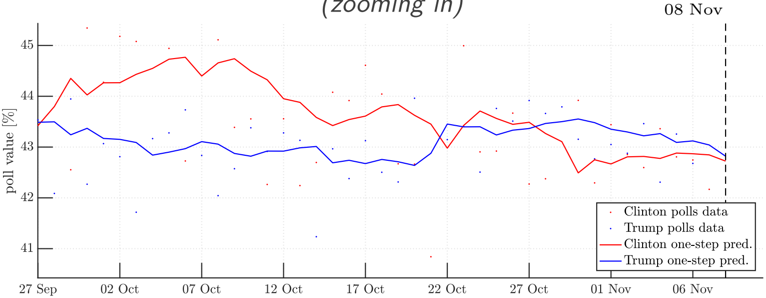
Addendum: Kalman filtering on real data

Predicting 2016 US election results

Procedure

4. Applying Kalman one-step predictor

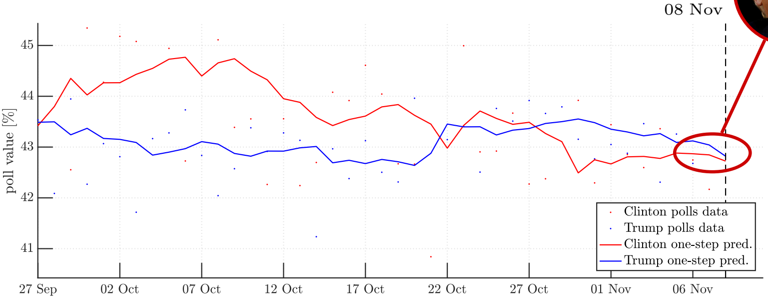
(zooming in)



Addendum: Kalman filtering on real data
Predicting 2016 US election results

Procedure

4. Applying Kalman one-step predictor



Addendum: Kalman filtering on real data
Predicting 2016 US election results

Some caveats

The predicted trajectories strongly depend on:

1. The choice of the model

2. The tuning of σ_Q and σ_R

Addendum: Kalman filtering on real data
Predicting 2016 US election results

Some caveats

The predicted trajectories strongly depend on:

1. The choice of the model

Ex Add.1. Try to use a different state space model*

2. The tuning of σ_Q and σ_R

Ex Add.2. Try to tune differently σ_Q and/or σ_R*

*Massaged polls data and sample code available at `baggio.dei.unipd.it/~teaching`