OBSERVERS

👤 Sébastien Boisgérault

CONTROL ENGINEERING WITH PYTHON

- 📖 Documents (GitHub)
- © License CC BY 4.0
- 🏦 Mines ParisTech, PSL University

SYMBOLS

🐍	Code	🔍	Worked Example
📈	Graph	🧩	Exercise
🏷️	Definition	💻	Numerical Method
💎	Theorem	🧮	Analytical Method
📝	Remark	🧠	Theory
ℹ️	Information	🗝️	Hint
⚠️	Warning	🔓	Solution

🐍 IMPORTS

```python
from numpy import *
from numpy.linalg import *
from scipy.integrate import solve_ivp
from scipy.linalg import solve_continuous_are
from matplotlib.pyplot import *
from numpy.testing import *
```

🐍 STREAM PLOT HELPER

```python
def Q(f, xs, ys):
    X, Y = meshgrid(xs, ys)
    fx = vectorize(lambda x, y: f([x, y])[0])
    fy = vectorize(lambda x, y: f([x, y])[1])
    return X, Y, fx(X, Y), fy(X, Y)
```

🧭 OBSERVABILITY

MOTIVATION

Controling a system generally requires the knowledge of the state $x(t)$, but measuring every state variable may be impossible (or too expensive).

Can we reduce the amount of physical sensors and still be able to compute the state with “virtual” or “software” sensors ?

🏷️ OBSERVERS

Control engineers call these software devices **observers**.

First we adress their mathematical feasibility.

🏷️ OBSERVABILITY

The system

$$\left| \begin{array}{rcl} \dot{x} & = & f(x) \\ y & = & g(x) \end{array} \right.$$

is **observable** if the knowledge of $y(t) = g(x(t))$ on some finite time span $[0, \tau]$ determines uniquely the initial condition $x(0)$.

📝 REMARKS

- The knowledge of $x(0)$ determines uniquely $x(t)$ via the system dynamics.
- Later, observers will provide merely **asymptotically exact** estimates $\hat{x}(t)$ of $x(t)$, that satisfy $\hat{x}(t) - x(t) \to 0$ when $t \to +\infty$.

EXTENSION

The definition of observability may be extended to systems with (known) inputs u:

$$\left| \begin{array}{lcl} \dot{x} & = & f(x, u) \\ y & = & g(x, u) \end{array} \right.$$

In general, the input u may then be selected specifically to generate the appropriate $y(t)$ that allows us to compute $x(0)$.

But for linear systems, the choice of u is irrelevant.

Indeed, if

$$\left| \begin{array}{rcl} \dot{x} & = & Ax + Bu \\ y & = & Cx + Du \end{array} \right.$$

and we can deduce $x(0)$ from $y(t)$ when $u = 0$:

$$y_0(t) = Ce^{At}x(0) \; \to \; x(0)$$

then in the general case, when we measure

$$y_u(t) = Ce^{At}x(0) + (H * u)(t)$$

we can compute

$$y_0(t) = y_u(t) - (H * u)(t)$$

and deduce $x(0)$ at this stage.

🔍 OBSERVABILITY / CAR

The position x (in meters) of a car of mass m (in kg) on a straight road is governed by

$$m\ddot{x} = u$$

where u the force (in Newtons) generated by its motor.

- we don’t know where the car is at $t = 0$,
- we don’t know what its initial speed is,
- we do know that the car doesn’t accelerate ($u = 0$).

If we measure the position $y(t) = x(t)$:

- $x(0) = y(0)$ is known,
- $\dot{x}(0) = \dot{y}(0)$ is also computable.

Thus the system is observable.

🤔 WHAT IF?

What if we measure the speed instead of the location ?

$$y(t) = \dot{x}(t)$$

The system dynamics $m\ddot{x}(t) = u(t) = 0$ yields $x(t) = x(0) + \dot{x}(0)t$ thus

$$\dot{x}(t) = \dot{x}(0)$$

and any $x(0)$ is consistent with a measure of a constant speed.

We can't deduce the position of the car from the measure of its speed; the system is not observable.

💎 KALMAN CRITERION

The system $\dot{x} = Ax,\ y = Cx$ is observable iff:

$$\operatorname{rank} \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix} = n$$

$[C; \ldots; CA^{n-1}]$ is the **Kalman observability matrix**.

🏷️ NOTATION

- "," row concatenation of matrices.
- ";" column concatenation of matrices.

We have

$$[C; \cdots; CA^{n-1}]^t = [C^t, \cdots, (A^t)^{n-1}C^t].$$

💎 DUALITY

The system $\dot{x} = Ax,\ y = Cx$ is observable

$$\iff$$

The system $\dot{x} = A^t x + C^t u$ is controllable.

🧩 FULLY MEASURED SYSTEM

Consider

$$\dot{x} = Ax, \; y = Cx$$

with $x \in \mathbb{R}^n, y \in \mathbb{R}^p$ and $\operatorname{rank} C = n$.

1. 🧠 🧮

Is the system observable ?

🔓 FULLY MEASURED SYSTEM

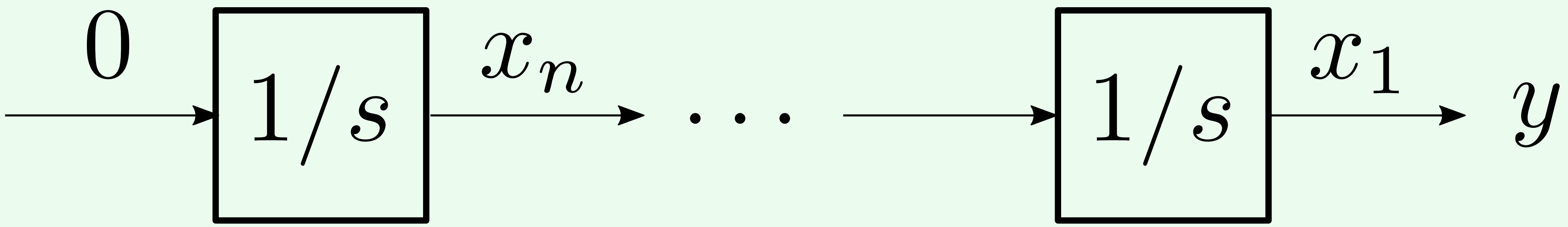
1. 🔓

Yes! The rank of its observability matrix

$$\begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix}$$

is at most n and at least the rank of C, which is also n. Thus by the 💎 Kalman Criterion, the system is observable.

INTEGRATOR CHAIN



$$\dot{x}_n = 0,\ \dot{x}_{n-1} = x_n,\ \cdots,\dot{x}_1 = x_2,\ y = x_1$$

1. 🧠 🧮

Show that the system is observable.

🔓 INTEGRATOR CHAIN

1. 🔓

The standard form of the dynamics associated to the state $x = (x_1, \ldots, x_n)$ is characterized by

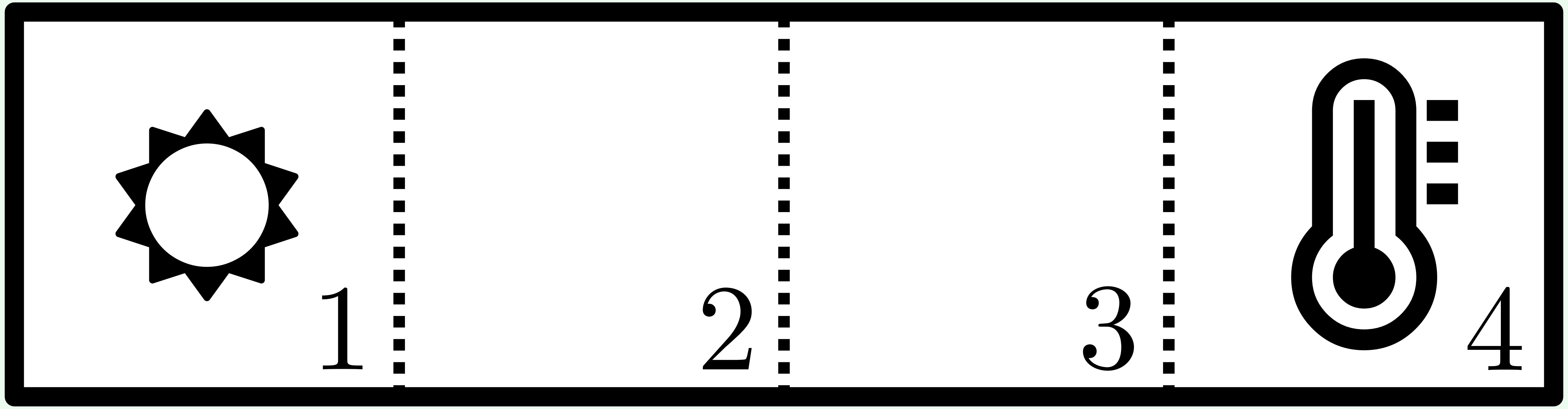
$$A = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & 0 & 1 \\ 0 & \cdots & \cdots & 0 & 0 \end{bmatrix}, \; C = [1, 0, 0, \ldots, 0]$$

Thus,

$$
\begin{aligned}
C &= [1, 0, 0, \ldots, 0] \\
CA &= [0, 1, 0, \ldots, 0] \\
\vdots &= \vdots \\
CA^{n-1} &= [0, 0, 0, \ldots, 1]
\end{aligned}
$$

The observability matrix has rank n and hence the system of observable.

🧩 HEAT EQUATION

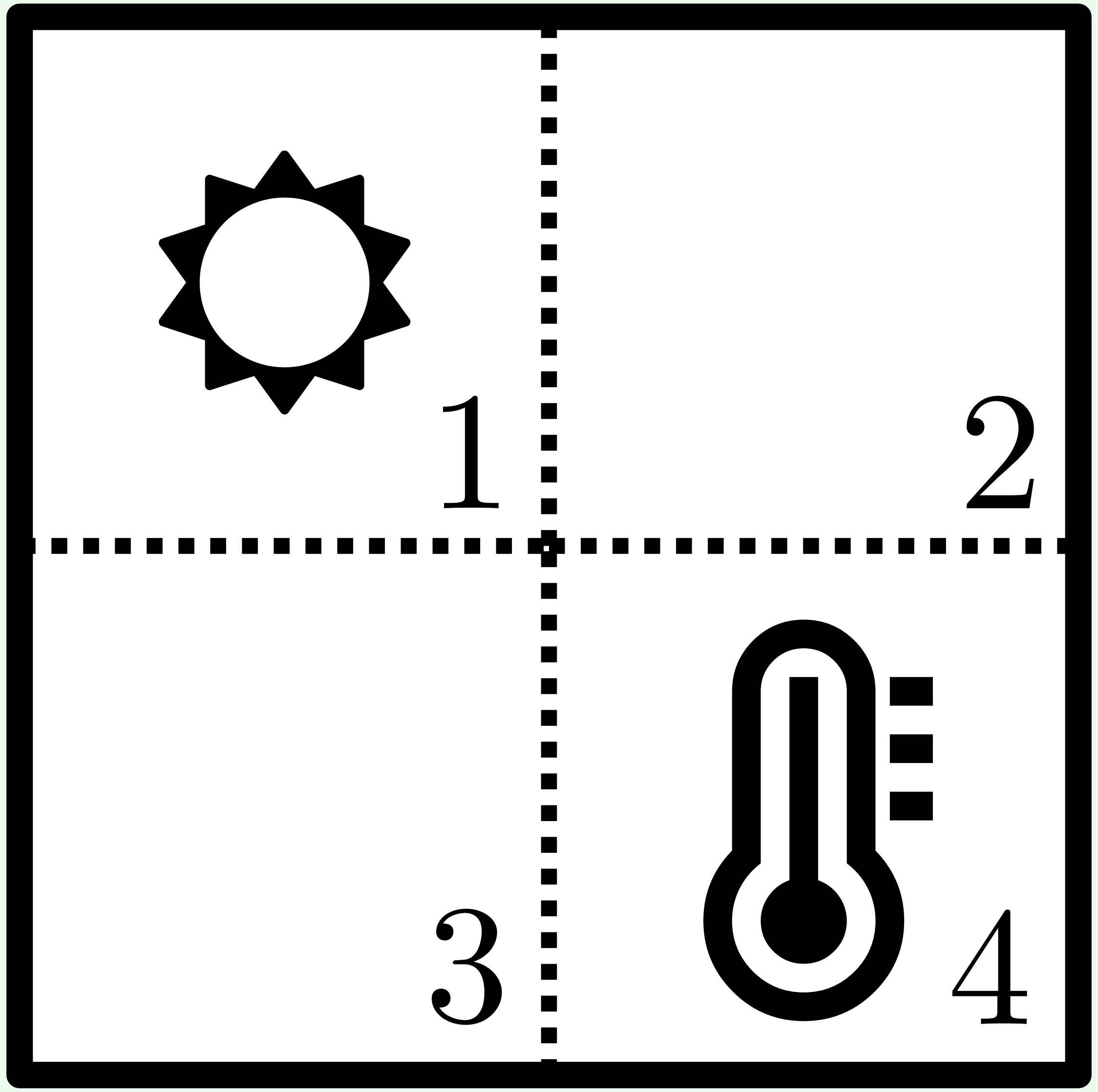


- $dT_1/dt = 0 + (T_2 - T_1)$
- $dT_2/dt = (T_1 - T_2) + (T_3 - T_2)$
- $dT_3/dt = (T_2 - T_3) + (T_4 - T_3)$
- $dT_4/dt = (T_3 - T_4)$
- $y = T_4$

1. 🧠 🧮

Show that the system is observable.

2. 🧮



Is it still true if the four cells are organized as a square and the temperature sensor is in any of the corners ?

3. 🧠 🧮

Can you make the system observable with two (adequatly located) sensors?

🔓 HEAT EQUATION

1. 🔓

The standard form of the dynamics associated to the state $T = (T_1, T_2, T_3, T4)$ is characterized by

$$A = \begin{bmatrix} -1 & 1 & 0 & 0 \\ 1 & -2 & 1 & 0 \\ 0 & 1 & -2 & 1 \\ 0 & 0 & 1 & -1 \end{bmatrix}, \; C = [0, 0, 0, 1]$$

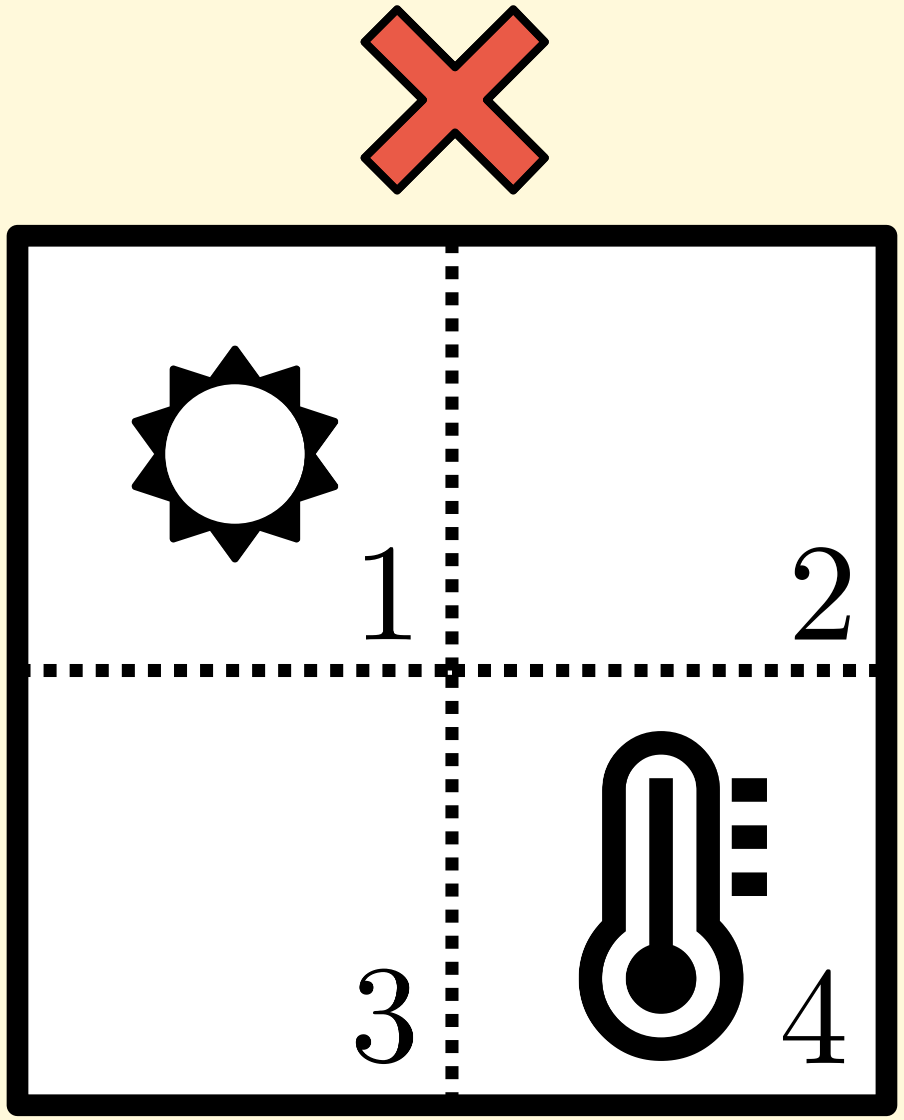
Therefore the observability matrix is

$$\begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & -1 \\ 0 & 1 & -3 & 2 \\ 1 & -5 & 9 & 5 \end{bmatrix}$$

whose rank is 4: the system is observable.

2. 🔓

In the square configuration, there is no way to get observability with a single thermometer.



Indeed the new A matrix would be

$$A = \begin{bmatrix} -2 & 1 & 1 & 0 \\ 1 & -2 & 0 & 1 \\ -1 & 0 & -2 & 1 \\ 0 & 1 & 1 & -2 \end{bmatrix}$$

and the new C matrix one of

$$[1, 0, 0, 0], \;\; [0, 1, 0, 0], \;\; [0, 0, 1, 0], \;\; [0, 0, 0, 1]$$

The corresponding observability matrices are

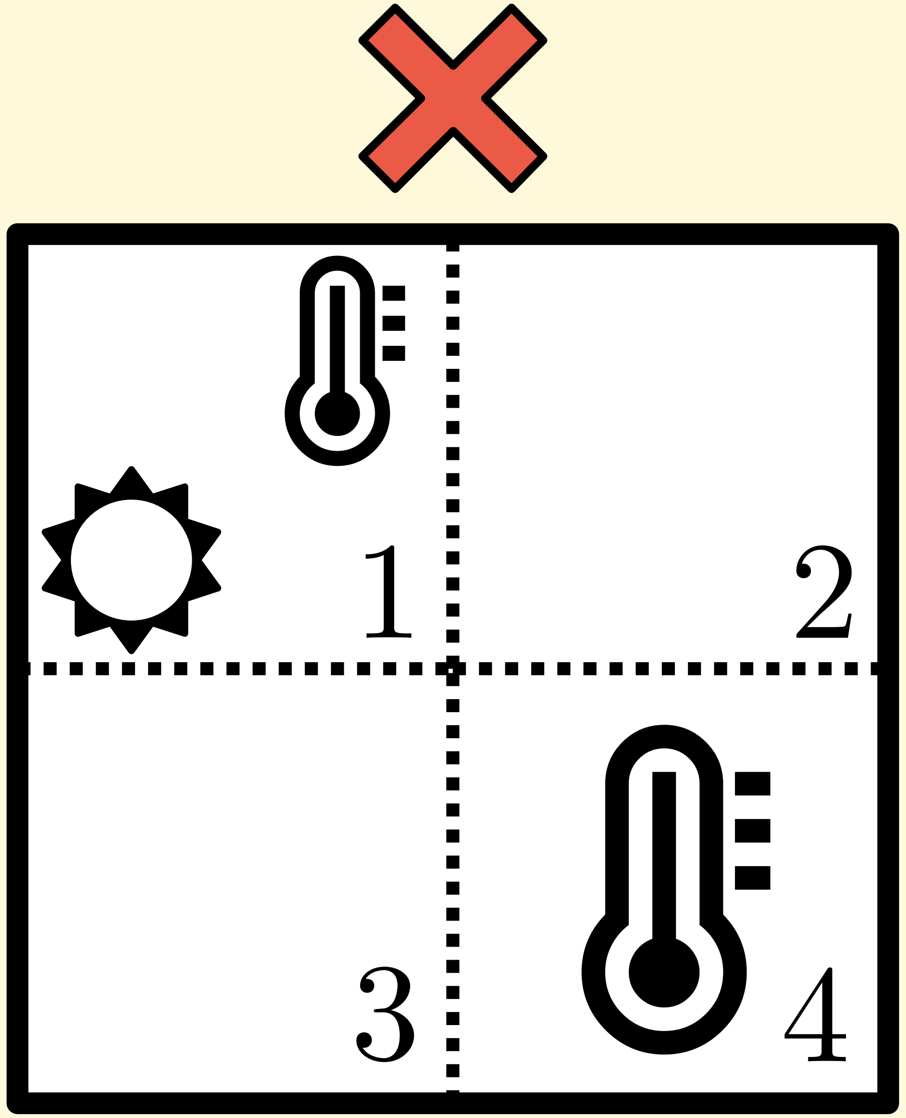
$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ -2 & 1 & 1 & 0 \\ 4 & -4 & -4 & 2 \\ 4 & -4 & -4 & 2 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & -2 & 0 & 1 \\ -4 & 6 & 2 & -4 \\ -4 & 6 & 2 & -4 \end{bmatrix},$$

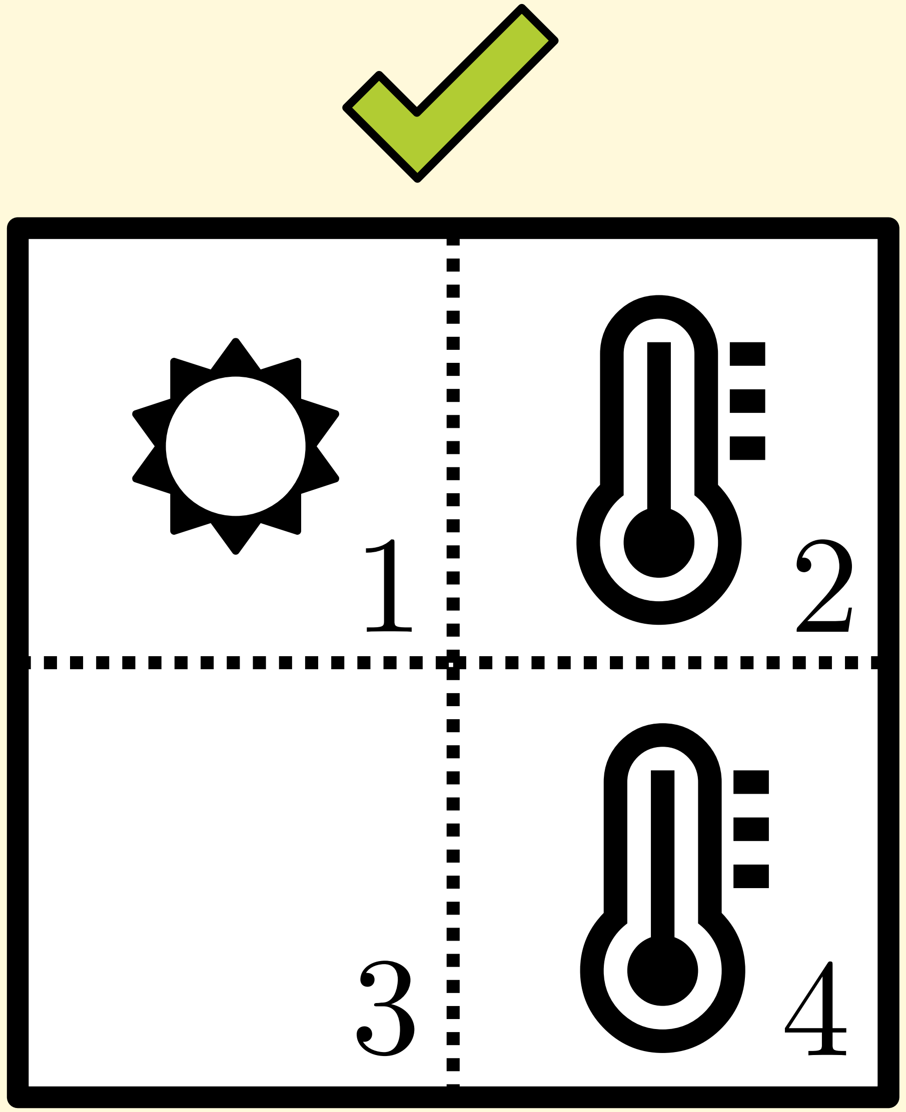
$$\begin{bmatrix} 0 & 0 & 1 & 0 \\ -1 & 0 & -2 & 1 \\ 4 & 0 & 4 & -4 \\ 4 & 0 & 4 & -4 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & -2 \\ 0 & -4 & -4 & 6 \\ 0 & -4 & -4 & 6 \end{bmatrix}$$

All the possible observability matrices in this case have rank 2 or 3 < 4.

3. 🔓

With 2 sensors, “it depends” (on the location of the sensors). For example:





The first case corresponds to

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

the second one to

$$C = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The observability matrices are

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -2 & 1 & 1 & 0 \\ 0 & 1 & 1 & -2 \\ 4 & -4 & -4 & 2 \\ 0 & -4 & -4 & 6 \\ 4 & -4 & -4 & 2 \\ 0 & -4 & -4 & 6 \end{bmatrix}, \quad \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -1 & -2 & 0 & 1 \\ 0 & 1 & 1 & -2 \\ -4 & 6 & 2 & -4 \\ 0 & -4 & -4 & 6 \\ -4 & 6 & 2 & -4 \\ 0 & -4 & -4 & 6 \end{bmatrix}$$

The first has rank 3, the second rank 4.

🧭 OBSERVER DESIGN

$$
\left|
\begin{aligned}
\dot{x} &= Ax + Bu \\
y &= Cx + Du
\end{aligned}
\right.
$$

STATE OBSERVER (VERSION 1)

Simulate the system behavior

$$\left| \begin{array}{rl} \dfrac{d\hat{x}}{dt} &= A\hat{x} + Bu \\ \hat{y} &= C\hat{x} + Du \end{array} \right.$$

and since we don't know better,

$$\hat{x}(0) = 0.$$

Is $\hat{x}(t)$ a good asymptotic estimate of $x(t)$?

🏷️ STATE ESTIMATE ERROR

The dynamics of the **state estimate error** $e = \hat{x} - x$ is

$$\begin{aligned}
\dot{e} &= \frac{d}{dt}(\hat{x} - x) \\
&= \frac{d\hat{x}}{dt} - \dot{x} \\
&= (A\hat{x} + Bu) - (Ax + Bu) \\
&= Ae
\end{aligned}$$

💎 STATE ESTIMATOR V1

❌ FAILURE

The state estimator error $e(t)$, solution of

$$\dot{e} = Ae$$

doesn't satisfy in general

$$\lim_{t \to +\infty} e(t) = 0.$$

We have

$$\lim_{t \to +\infty} e(t) = 0$$

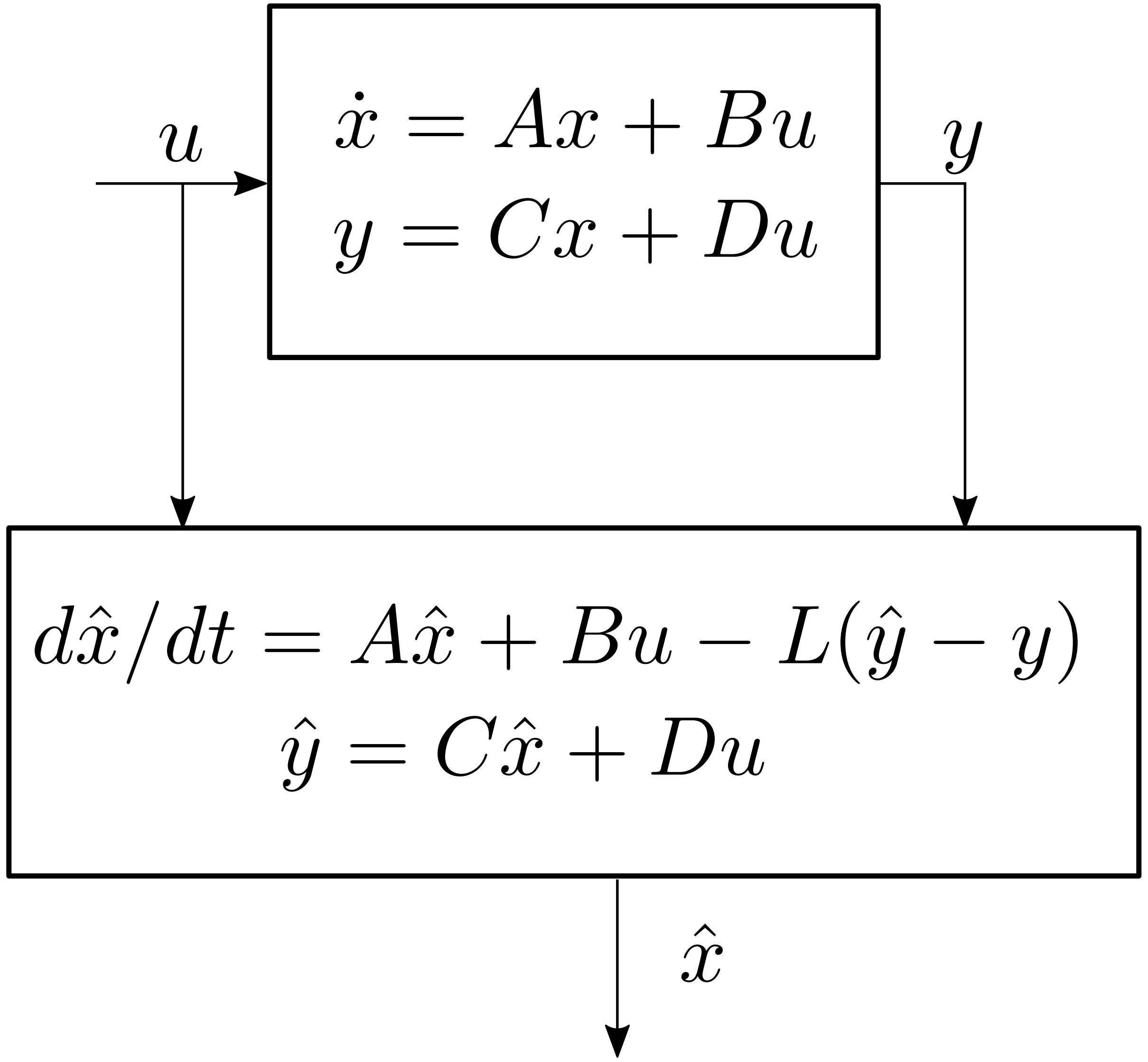
for every value of $e(0) = \hat{x}(0) - x(0)$, if and only if $\dot{x} = Ax$ is asymptotically stable (i.e. the eigenvalues of A are in the open left-hand plane).

STATE OBSERVER (VERSION 2)

Change the observer dynamics to account for differences between $\hat{y}$ and y (both known values):

$$\left| \begin{array}{rl} \dfrac{d\hat{x}}{dt} & = A\hat{x} + Bu - L(\hat{y} - y) \\ \hat{y} & = C\hat{x} + Du \end{array} \right.$$

for some **observer gain** matrix $L \in \mathbb{R}^{n \times p}$ (to be determined).


u
$\dot{x} = Ax + Bu$
$y = Cx + Du$
y
$d\hat{x}/dt = A\hat{x} + Bu - L(\hat{y} - y)$
$\hat{y} = C\hat{x} + Du$
$\hat{x}$

The new dynamics of $e = \hat{x} - x$ is

$$\begin{aligned}
\dot{e} &= \frac{d}{dt}(\hat{x} - x) \\
&= \frac{d\hat{x}}{dt} - \dot{x} \\
&= (A\hat{x} + Bu - L(C\hat{x} - Cx)) - (Ax + Bu) \\
&= (A - LC)e
\end{aligned}$$

📝 REMINDER

The system $\dot{x} = Ax, \; y = Cx$ is observable

$$\iff$$

The system $\dot{x} = A^t x + C^t u$ is commandable.

SO WHAT?

In this case, we can perform arbitrary pole assignment:

- for any conjugate set Λ of eigenvalues,
- there is a matrix $K \in \mathbb{R}^{p \times n}$ such that

$$\sigma(A^t - C^t K) = \Lambda$$

Since $\sigma(M) = \sigma(M^t)$, for any square matrix M,

$$\begin{aligned}\sigma(A^t - C^t K) &= \sigma((A - K^t C)^t) \\ &= \sigma(A - K^t C)\end{aligned}$$

POLE ASSIGNMENT (OBSERVERS)

✔️ SUCCESS

Thus, if we set

$$L = K^t$$

we have solved the pole assignment problem **for observers:**

$$\sigma(A - LC) = \Lambda$$

🔍 POLE ASSIGNMENT

Consider the double integrator $\ddot{y} = u$

$$\frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$$

$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

(in standard form)

```
from scipy.signal import place_poles
A = array([[0, 1], [0, 0]])
C = array([[1, 0]])
poles = [-1, -2]
K = place_poles(A.T, C.T, poles).gain_matrix
L = K.T
assert_almost_equal(K, [[3.0, 2.0]])
```

$$\frac{d}{dt}\begin{bmatrix}\hat{x}_1 \\ \hat{x}_2\end{bmatrix} = \begin{bmatrix}0 & 1 \\ 0 & 0\end{bmatrix}\begin{bmatrix}\hat{x}_1 \\ \hat{x}_2\end{bmatrix} + \begin{bmatrix}0 \\ 1\end{bmatrix}u - \begin{bmatrix}3 \\ 2\end{bmatrix}(\hat{y} - y)$$

$$\hat{y} = \begin{bmatrix}1 & 0\end{bmatrix}\begin{bmatrix}\hat{x}_1 \\ \hat{x}_2\end{bmatrix}$$

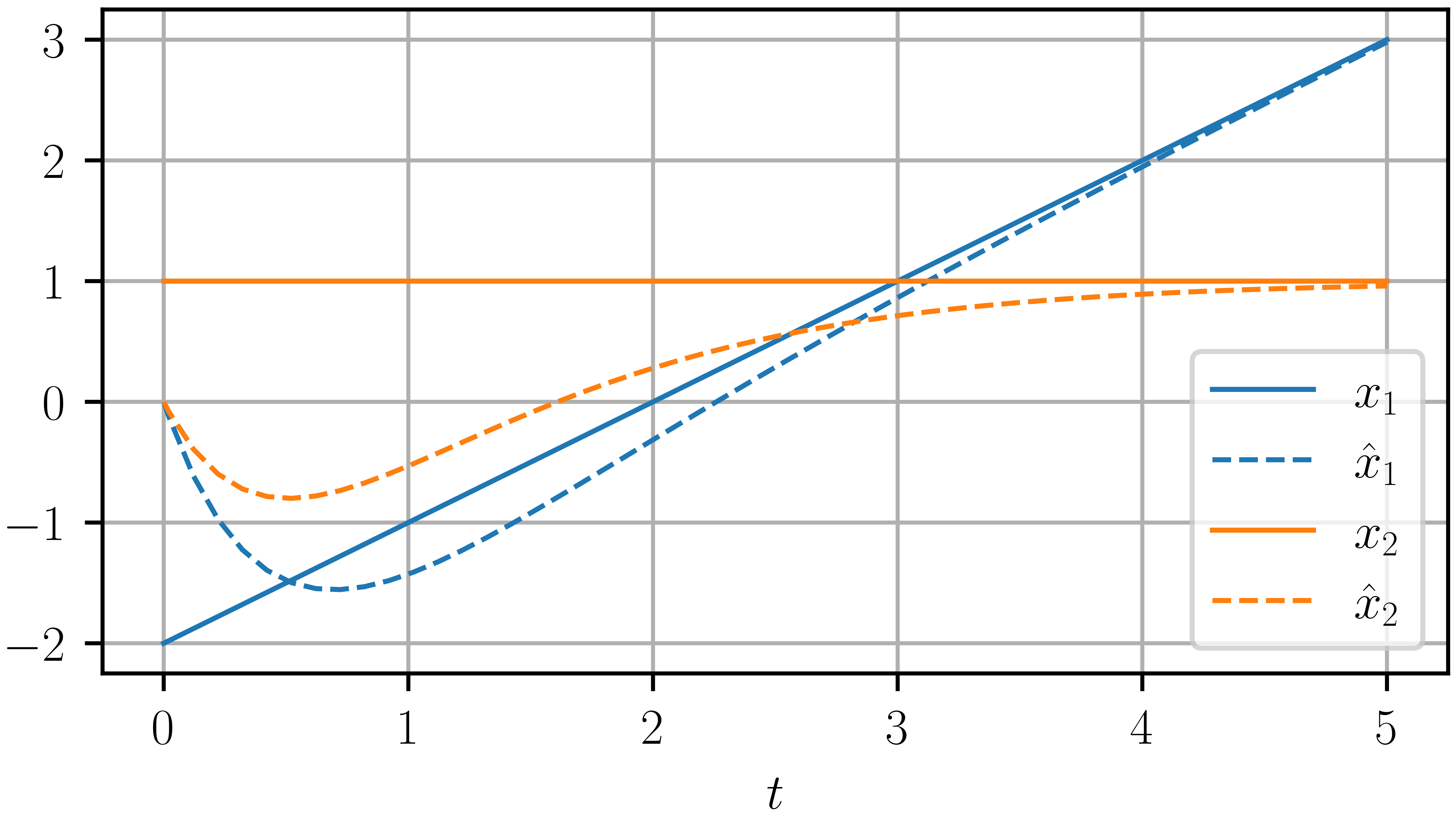
```python
def fun(t, X_Xhat):
    x, x_hat = X_Xhat[0:2], X_Xhat[2:4]
    y, y_hat = C.dot(x), C.dot(x_hat)
    dx = A.dot(x)
    dx_hat = A.dot(x_hat) - L.dot(y_hat - y)
    return r_[dx, dx_hat]
```

🐍 💻

```python
y0 = [-2.0, 1.0, 0.0, 0.0]
result = solve_ivp(
    fun=fun,
    t_span=[0.0, 5.0],
    y0=y0,
    max_step=0.1
)
```

🐍 📈

```
figure()
t = result["t"]
y = result["y"]
plot(t, y[0], "C0", label="$x_1$")
plot(t, y[2], "C0--", label=r"$\hat{x}_1$")
plot(t, y[1], "C1", label="$x_2$")
plot(t, y[3], "C1--", label=r"$\hat{x}_2$")
xlabel("$t$"); grid(); legend()
```


3
2
1
0
−1
−2
0
1
2
3
4
5
t
x1
x̂1
x2
x̂2

KALMAN FILTER

SETTING

Consider $\dot{x} = Ax, \; y = Cx$ where:

- the state $x(t)$ is unknown ($x(0)$ is unknown),
- only (a noisy version of) $y(t)$ is available.

We want a sensible estimation $\hat{x}(t)$ of $x(t)$.

We now assume the existence of state and output disturbances $v(t)$ and $w(t)$ (deviations from the exact dynamics)

$$\left| \begin{array}{l} \dot{x} = Ax + v \\ y = Cx + w \end{array} \right.$$

Thes disturbances (or “noises”) are unknown; we are searching for the estimate $\hat{x}(t)$ of $x(t)$ that requires the smallest deviation from the exact dynamics to explain the data.

For a known $y(t)$, among all possible trajectories $x(t)$ of the system, find the one that minimizes

$$J = \int_0^{+\infty} v(t)^t Q v(t) + w(t)^t R w(t) \, dt$$

where:

- $Q \in \mathbb{R}^{n \times n}$ and $R \in \mathbb{R}^{p \times p}$,
- (to be continued …)

- Q and R are **symmetric** ($R^t = R$ and $Q^t = Q$),
- Q and R are **positive definite** (denoted "> 0")

HEURISTICS

If it is known that there is a large state disturbance but small output disturbance, it makes sense to reduce the impact of the state disturbance in the composition of J, hence to select a small Q wrt R.

💎 OPTIMAL SOLUTION

Assume that $\dot{x} = Ax, \; y = Cx$ is observable.

There is a state estimation $\hat{x}(t)$, given for some $L \in \mathbb{R}^{n \times p}$ as the solution of

$$
\left|
\begin{array}{rl}
d\hat{x}/dt &= A\hat{x} - L(\hat{y} - y) \\
\hat{y} &= C\hat{x}
\end{array}
\right.
$$

The dynamics of the corresponding estimation error $e(t) = \hat{x}(t) - x(t)$ is asymptotically stable.

💎 ALGEBRAIC RICCATI EQUATION

The gain matrix L is given by

$$L = \Pi C^t R,$$

where $\Pi \in \mathbb{R}^{n \times n}$ is the unique matrix such that $\Pi^t = \Pi, \Pi > 0$ and

$$\Pi C^t R C \Pi - \Pi A^t - A\Pi - Q^{-1} = 0.$$

OPTIMAL CONTROL $\leftrightarrow$ FILTER

Solve the Ricatti equation for optimal control with

$$(A, B, Q, R) = (A^t, C^t, Q^{-1}, R^{-1})$$

then define

$$L := \Pi C^t R$$

🔍 KALMAN FILTER

Consider the system

$$\dot{x} = v$$
$$y = x + w$$

If we believe that the state and output perturbation are of the same scale, we may try

$$Q = [1.0], R = [1.0]$$

With $\Pi = [\sigma]$, the filtering Ricatti equation becomes

$$\sigma^2 - 2\sigma - 1 = 0$$

whose only positive solution is

$$\sigma = \frac{2 + \sqrt{(-2)^2 - 4 \times 1 \times (-1)}}{2} = 1 + \sqrt{2}.$$

With $L = [\ell]$, we end up with

$$\ell = \sigma = 1 + \sqrt{2}.$$

Thus, the optimal filter is

$$\begin{aligned} d\hat{x}/dt &= -(1+\sqrt{2})(\hat{y} - y) \\ \hat{y} &= \hat{x} \end{aligned}$$

🔍 KALMAN FILTER

Consider the double integrator $\ddot{x} = 0, y = x$.

$$\frac{d}{dt}\begin{bmatrix} x \\ \dot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} x \\ \dot{x} \end{bmatrix} + \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}, y = \begin{bmatrix} 1 & 0 \end{bmatrix}\begin{bmatrix} x \\ \dot{x} \end{bmatrix} + w$$

(in standard form)

```python
A = array([[0, 1], [0, 0]])
B = array([[0], [1]])
Q = array([[1, 0], [0, 1]])
R = array([[1]])
```

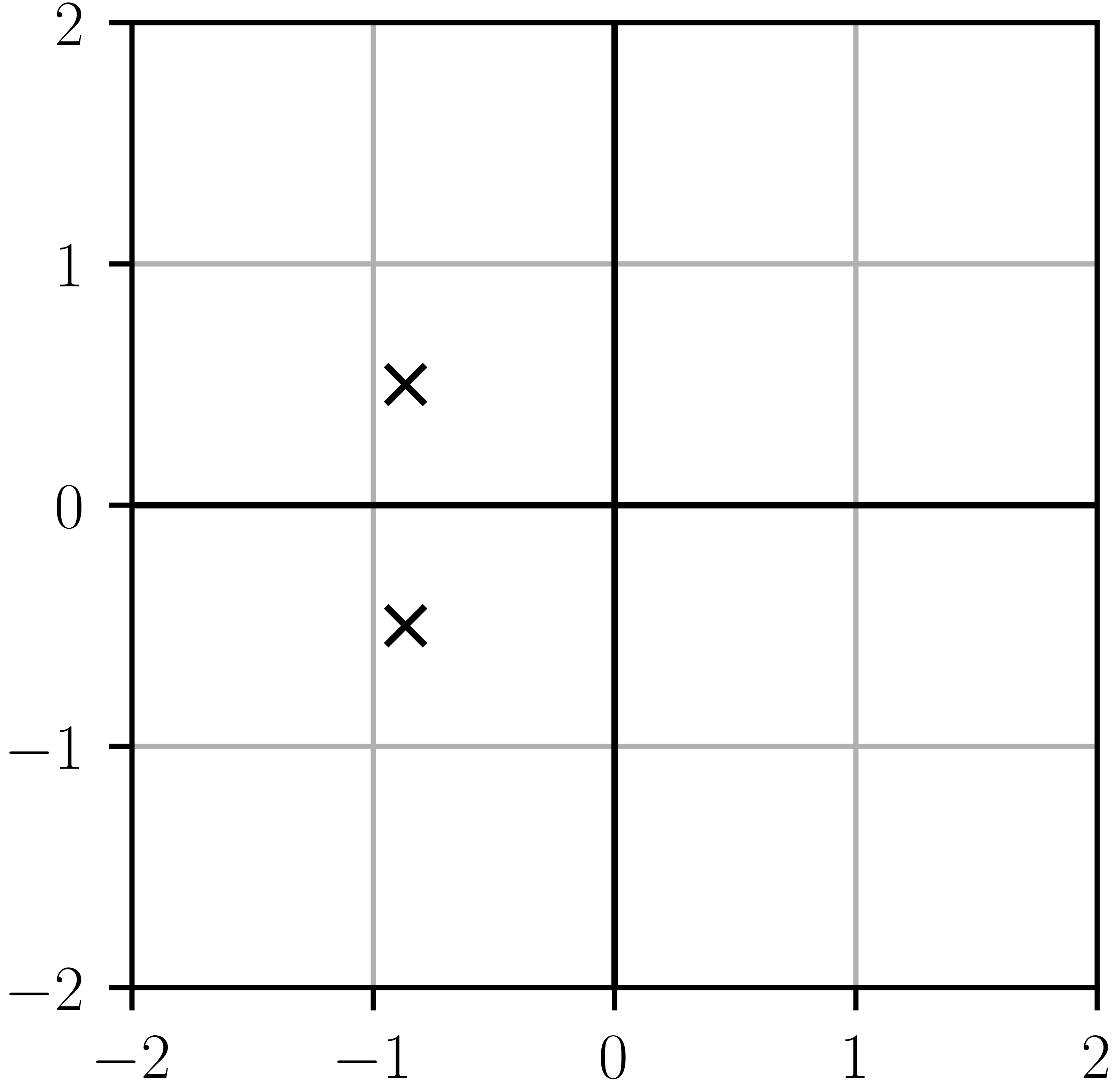
```
sca = solve_continuous_are
Sigma = sca(A.T, C.T, inv(Q), inv(R))
L = Sigma @ C.T @ R

eigenvalues, _ = eig(A - L @ C)
assert all([real(s) < 0 for s in eigenvalues])
```

🐍 📈

```
figure()
x = [real(s) for s in eigenvalues]
y = [imag(s) for s in eigenvalues]
plot(x, y, "kx"); xlim(-2, 2); ylim(-2, 2)
plot([0, 0], [-2, 2], "k");
plot([-2, 2], [0, 0], "k")
grid(True); axis("square")
title("Eigenvalues")
```

Eigenvalues



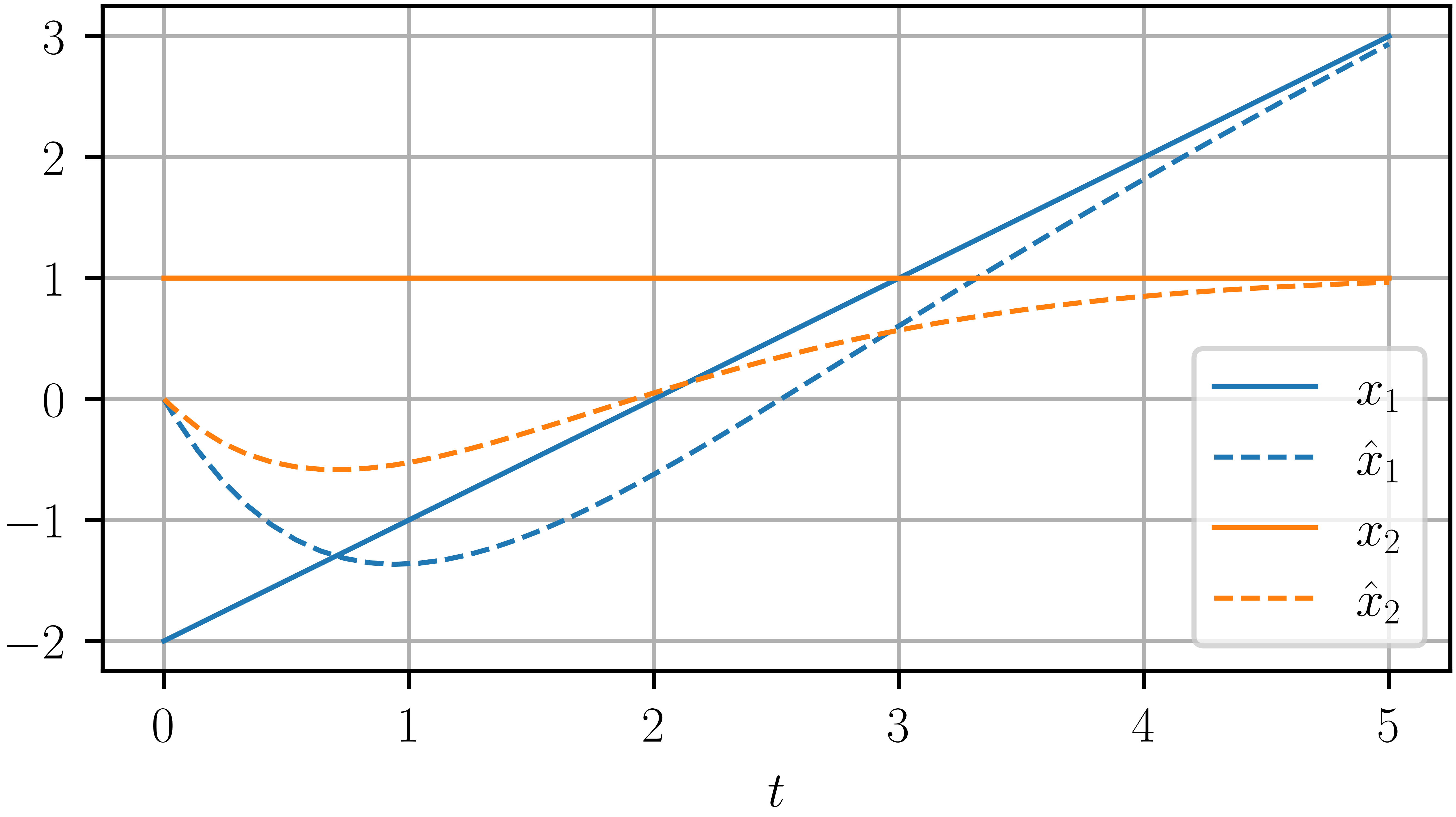
```python
def fun(t, X_Xhat):
    x, x_hat = X_Xhat[0:2], X_Xhat[2:4]
    y, y_hat = C.dot(x), C.dot(x_hat)
    dx = A.dot(x)
    dx_hat = A.dot(x_hat) - L.dot(y_hat - y)
    return r_[dx, dx_hat]
```

🐍 💻

```python
y0 = [-2.0, 1.0, 0.0, 0.0]
result = solve_ivp(
    fun=fun,
    t_span=[0.0, 5.0],
    y0=y0,
    max_step=0.1
)
```

🐍 📈

```python
figure()
t = result["t"]; y = result["y"]
plot(t, y[0], "C0", label="$x_1$")
plot(t, y[2], "C0--", label=r"$\hat{x}_1$")
plot(t, y[1], "C1", label="$x_2$")
plot(t, y[3], "C1--", label=r"$\hat{x}_2$")
xlabel("$t$")
grid(); legend()
```



3
2
1
0
−1
−2
0
1
2
3
4
5
t
x_1
$\hat{x}_1$
x_2
$\hat{x}_2$