State observers and recursive filters in classical feedback control theory

State-feedback control example: second-order system

Consider the driven second-order system

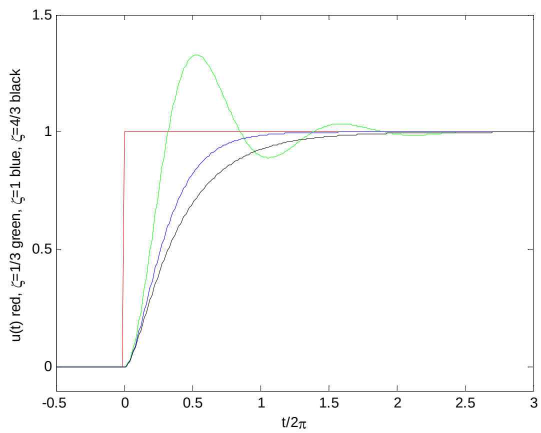
$$\ddot{q} = -2\zeta\omega_0\dot{q} - \omega_0^2 q + u, \quad x_1 \equiv q, \quad x_2 \equiv \dot{q},$$

$$\frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\omega_0^2 & -2\zeta\omega_0 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ u \end{bmatrix}.$$

Here u could represent an external applied force (in a mechanical mass-spring-damper system) or voltage (in the LCR circuit realization). Anticipating conventional control-theoretic notation that we will introduce below, let us also write this as

$$\frac{d}{dt}\vec{x} = A\vec{x} + Bu, \quad A = \begin{bmatrix} 0 & 1 \\ -\omega_0^2 & -2\zeta\omega_0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

Recall that the damping ratio $\zeta \geq 0$ sets important properties of the step (transient) response:



We can write a general solution for the initial-value problem for such a system via the matrix exponential,

$$\begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} = \exp(At)\begin{bmatrix} x_1(0) \\ x_2(0) \end{bmatrix},$$

where methods for computing $\exp(At)$ can be found in elementary linear algebra textbooks (the case $\zeta = 1$ is slightly tricky):

$$\zeta > 1: \exp(At) \to \frac{e^{-\gamma t}}{\sqrt{\zeta^2 - 1}}\begin{bmatrix} \zeta\sinh(\delta t) + \sqrt{\zeta^2-1}\cosh(\delta t) & \frac{1}{\omega_0}\sinh(\delta t) \\ -\omega_0\sinh(\delta t) & -\zeta\sinh(\delta t) + \sqrt{\zeta^2-1}\cosh(\delta t) \end{bmatrix},$$

$$\zeta = 1: \exp(At) \to \begin{bmatrix} e^{-\omega_0 t} + t\omega_0 e^{-\omega_0 t} & te^{-\omega_0 t} \\ -\omega_0 e^{-\omega_0 t} + \omega_0(e^{-\omega_0 t} - t\omega_0 e^{-\omega_0 t}) & e^{-\omega_0 t} - t\omega_0 e^{-\omega_0 t} \end{bmatrix},$$

$$\zeta < 1: \exp(At) \to \frac{e^{-\gamma t}}{\sqrt{1-\zeta^2}}\begin{bmatrix} \zeta\sin(\nu t) + \sqrt{1-\zeta^2}\cos(\nu t) & \frac{1}{\omega_0}\sin(\nu t) \\ -\omega_0\sin(\nu t) & -\zeta\sin(\nu t) + \sqrt{1-\zeta^2}\cos(\nu t) \end{bmatrix}.$$

Let us briefly consider the linear state-feedback control scenario

$$u = -K\vec{x},$$

where (since we have taken u to be scalar)

$$K = \begin{bmatrix} k_1 & k_2 \end{bmatrix}.$$

With this feedback law, we have

$$\frac{d}{dt}\vec{x} = A\vec{x} + Bu = (A - BK)\vec{x},$$

where

$$\begin{aligned} A - BK &\to \begin{bmatrix} 0 & 1 \\ -\omega_0^2 & -2\zeta\omega_0 \end{bmatrix} - \begin{bmatrix} 0 \\ 1 \end{bmatrix}\begin{bmatrix} k_1 & k_2 \end{bmatrix} \\ &= \begin{bmatrix} 0 & 1 \\ -\omega_0^2 & -2\zeta\omega_0 \end{bmatrix} - \begin{bmatrix} 0 & 0 \\ k_1 & k_2 \end{bmatrix} \\ &= \begin{bmatrix} 0 & 1 \\ -\omega_0^2 - k_1 & -2\zeta\omega_0 - k_2 \end{bmatrix}. \end{aligned}$$

Clearly we can view this as a modified second-order system. If we define

$$\omega_0' = \sqrt{\omega_0^2 + k_1},$$

then apparently we have

$$\begin{aligned} \zeta'\omega_0' &= \zeta\omega_0 - k_2, \\ \zeta' &= \frac{\zeta\omega_0 - k_2}{\sqrt{\omega_0^2 + k_1}}, \end{aligned}$$

and it follows that we can actually set ω_0' to any value we like through choice of the feedback parameter k_1, and once this is set (assuming $\omega_0' \neq 0$) we can adjust ζ' as desired through choice of k_2. Hence if we don't like the transient behavior of our original open-loop system (without feedback), we can in principle modify it however we like by use of *state feedback* as described above. We program the desired behavior by setting the feedback gain K.

Note that if accurate, instantaneous measurements of the components of $\vec{x}$ are available as electronic signals, it is straightforward to produce the feedback signal u with an op-amp circuit.

The fact that we have full power to reprogram the oscillation frequency and damping ratio in this state feedback scenario can be made obvious if we look at the dynamics with feedback, transformed back to a single second-order ODE:

$$\begin{aligned} \frac{d}{dt}\begin{bmatrix} q \\ \dot{q} \end{bmatrix} &= \begin{bmatrix} 0 & 1 \\ -\omega_0^2 - k_1 & -2\zeta\omega_0 - k_2 \end{bmatrix}\begin{bmatrix} q \\ \dot{q} \end{bmatrix}, \\ 0 &= \ddot{q} + (2\zeta\omega_0 + k_2)\dot{q} + (\omega_0^2 + k_1)q. \end{aligned}$$

Hence k_1 is associated with a modification of the harmonic restoring force while k_2 directly and independently modifies the velocity-damping.

Just for fun, let us calculate the eigenvalues of the feedback-modified A matrix:

$$\begin{aligned} 0 &= \det(A - BK - I\lambda) \to \begin{bmatrix} -\lambda & 1 \\ -\omega_0^2 - k_1 & -2\zeta\omega_0 - k_2 - \lambda \end{bmatrix}, \\ 0 &= \lambda(2\zeta\omega_0 + k_2 + \lambda) + \omega_0^2 + k_1 \\ &= \lambda^2 + (2\zeta\omega_0 + k_2)\lambda + \omega_0^2 + k_1. \end{aligned}$$

Considering the solutions given by the quadratic equation,

$$\lambda = \frac{1}{2}\left(-2\zeta\omega_0 - k_2 \pm \sqrt{(2\zeta\omega_0 + k_2)^2 - 4\omega_0^2 - 4k_1}\right),$$

we see that we can achieve any desired pair of eigenvalues $\{\lambda_+, \lambda_-\}$ by setting

$$\begin{aligned} \lambda_+ + \lambda_- &= -2\zeta\omega_0 - k_2, \\ k_2 &= -2\zeta\omega_0 - \lambda_+ - \lambda_-, \end{aligned}$$

and

$$\begin{aligned} \lambda_+ - \lambda_- &= \sqrt{(2\zeta\omega_0 + k_2)^2 - 4\omega_0^2 - 4k_1}, \\ k_1 &= \frac{1}{4}(2\zeta\omega_0 + k_2)^2 - \omega_0^2 - \frac{1}{4}(\lambda_+ - \lambda_-)^2 \\ &= \frac{1}{4}(\lambda_+ + \lambda_-)^2 - \omega_0^2 - \frac{1}{4}(\lambda_+ - \lambda_-)^2 \\ &= \lambda_+\lambda_- - \omega_0^2. \end{aligned}$$

Hence we find that we can arbitrarily program the eigenvalues of the closed-loop dynamics matrix, a design method called 'pole placement.'

Before turning to some generalities, we note that the second-order system with the control structure we are considering can be pushed into arbitrary states in a similar manner as the double-integrator. Suppose we are starting at the origin at $t = 0$ and want to reach $\begin{bmatrix} q & \dot{q} \end{bmatrix}'$ at a target time $T.$ If we use the input u and allow our input signals to be unbounded, we can in principle take the following approach. First, give the system a momentum impulse at time $t = 0$ that will cause the system to pass through position q at time $T.$ To convince ourselves that this is really possible, we could look at the matrix exponential solution and show that we can always solve for $\dot{q}_0$ such that

$$\begin{bmatrix} q \\ \cdot \end{bmatrix} = \exp(AT)\begin{bmatrix} 0 \\ \dot{q}_0 \end{bmatrix}.$$

In the overdamped case, for example, this leads to

$$\dot{q}_0 = \frac{\omega_0\sqrt{\zeta^2 - 1}}{e^{-\omega_0\zeta T}\sinh\left(\omega_0 T\sqrt{\zeta^2 - 1}\right)}q.$$

Then, at time $T,$ apply a second momentum impulse that "corrects" the final velocity $\dot{q}$ to its desired value. Note that even with this rather singular strategy, in which we rely on our ability to apply controls so strong that they overwhelm most of the system's natural dynamics (harmonic restoring force and velocity-damping), we still rely on the inherent integrator structure

$$\frac{d}{dt}q = \dot{q}$$

that allows us to use an input to $\dot{q}$ only to affect the position $q.$

Reachability rank condition

We say that a linear system

$$\dot{x} = Ax + Bu$$

is *reachable* if for any initial state $x(0) = x_0$, desired final state x_f and 'target time' T it is possible to find a control input $u(t),\ t \in [0, T]$ that steers the system to reach $x(T) = x_f.$ There is a theorem (see A&M pp. 167-170) that says that a system is reachable if its *reachability matrix*

$$W_r \equiv [B \quad AB \quad A^2B \quad \cdots \quad A^{n-1}B]$$

has full rank (here $x \in R^n$). If a system is reachable, it is furthermore possible to solve the *pole placement* problem, in which we want to design a state feedback law

$$u = -Kx$$

such that we can pick any eigenvalues we want for the controlled dynamics

$$\begin{aligned} \dot{x} &= Ax + Bu \\ &= (A - BK)x. \end{aligned}$$

Here A and B are given, and we must find a K to achieve the desired eigenvalues for $(A - BK).$ There is a very convenient Matlab Control Toolbox function, **`place`**, that can be used to accomplish this numerically.

In the case of our second-order system with

$$A = \begin{bmatrix} 0 & 1 \\ -\omega_0^2 & -2\zeta\omega_0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$

we have

$$AB = \begin{bmatrix} 0 & 1 \\ -\omega_0^2 & -2\zeta\omega_0 \end{bmatrix}\begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ -2\zeta\omega_0 \end{bmatrix}, \quad W_r = \begin{bmatrix} 0 & 1 \\ 1 & -2\zeta\omega_0 \end{bmatrix},$$

and clearly W_r has full rank as $\det(W_r) = -1.$

Before moving on, let's look at a simple example (from Åström and Murray, Ex. 5.3) of a system that is *not* reachable:

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

Here we can easily compute

$$W_r = \begin{bmatrix} 1 & -1 \\ 1 & -1 \end{bmatrix},$$

which clearly has determinant zero. This can be understood by noting the complete 'symmetry' of the way that u modifies the evolution of x_1 and x_2. For example, if $x_1(0) = x_2(0)$ there is no way to use u to achieve $x_1(T) \neq x_2(T)$ at any later time.

State feedback versus output feedback

Note that in our discussion of stabilization and pole-placement so far, we have assumed that it makes sense to design a control law of the form

$$u = -Kx.$$

This is called a 'state feedback' law since in order to determine the control input $u(t)$ at time $t,$ we generally need to have full knowledge of the state $x(t).$ In practice this is often not possible, and thus we usually specify the available output signals when defining a control design problem:

$$\dot{x} = Ax + Bu,$$
$$y = Cx.$$

Here the output signal $y(t)$, which can in principle be a vector of any dimension, represents the information about the evolving system state that is made available to the controller via sensors. An 'output feedback' law must take the form

$$u(t) = f[y(\tau \leq t)],$$

where, in general, we can allow $u(t)$ to depend on the entire history of $y(\tau)$ with $\tau \leq t$ (more on this below and later in the course). Output feedback is a natural setting for practical applications. For example, if we are talking about cruise control for an automobile, x may represent a complex set of variables having to do with the internal state of the engine, wheels and chassis while y is only a readout from the speedometer. Hopefully it will seem natural that it is usually prohibitively difficult to install a sensor to monitor every coordinate of the system's state space, and also that it will often be unnecessary to do so (cruise control electronics can function quite well with just the car's speed).

One simple example of a system in which full state knowledge is clearly not necessary is (asymptotic) stabilization of a simple harmonic oscillator. If the natural dynamics of the plant is

$$m\ddot{x} = -kx,$$

and our actuation mechanism is to apply forces directly on the mass, then the control system looks like

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

(where x_1 is now the position and x_2 the velocity). We can clearly make the equilibrium point at the origin asymptotically stable via the linear state-feedback control law

$$u = -bx_2 = \begin{bmatrix} 0 & -b \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix},$$

which makes the overall equation of motion

$$\ddot{x}_1 = -\frac{k}{m}x_1 - b\dot{x}_1,$$

which we recognize as a damped harmonic oscillator. Thus it is clear that the controller only needs to know the velocity of the oscillator in order to implement a successful feedback strategy. So even if we go to a SISO (single-input single-output) formulation of this problem,

$$\dot{x} = Ax + Bu,$$
$$y = Cx,$$

we are obviously fine for any C of the form $(\alpha \neq 0)$

$$C = \begin{bmatrix} 0 & \alpha \end{bmatrix},$$

since $x_2 = y/\alpha$ and we can implement an output-feedback law of the form

$$u = -bx_2 = -\frac{b}{\alpha}y.$$

Clearly, if C is a square matrix and y has the same dimension as $x,$ output feedback is essentially equivalent to state feedback if C is invertible. As a generalization of what we did for the simple harmonic oscillator above, we could just design a state feedback controller $K,$ set

$$\hat{x} = C^{-1}y,$$

and apply feedback

$$u = -K\hat{x} = -KC^{-1}y.$$

However this is a special case and not the sort of convenience we want to count on!

State estimation

At this point it might seem like we might need completely new theorems about reachability and pole-placement for output-feedback laws, when $u(t)$ is only allowed to depend on $y(\tau < t)$. However, it turns out that we can build naturally on our previous results by appealing to a *separation method*. The basic idea is that we will try to construct a procedure for processing the data $y(\tau < t)$ to obtain an estimate $\hat{x}(t)$ of the true system state $x(t)$, and then apply a feedback law $u = -K\hat{x}$ based on this estimate. This can be possible even when C is not invertible (not even square). The controller thus assumes the structure of a dynamical system itself, with $y(t)$ as its input, $u(t)$ as its output and $\hat{x}(t)$ as its internal state. There are various ways of designing 'state estimators' to extract $\hat{x}(t)$ from $y(\tau < t)$, of which we will discuss two, and there is also a convenient procedure for determining whether or not y contains enough information to make full state reconstruction possible in principle. The latter test looks a lot like the test for reachability, for not accidental reasons.

Let's start by thinking about the simple harmonic oscillator again. We noted that in order to asymptotically stabilize the equilibrium point at the origin, it would be most convenient to have an output signal that told us directly about its velocity x_2. However, you may have already realized that in a scenario with $(\alpha \neq 0)$

$$C = \begin{bmatrix} \alpha & 0 \end{bmatrix},$$
$$y = \alpha x_1,$$

it should be simple to obtain a good estimate of x_2 via

$$\hat{x}_2 = \frac{d}{dt}\alpha^{-1}y.$$

This is certainly a valid procedure for estimating x_2, although in practice one should be wary of taking derivatives of measured data since that tends to accentuate high-frequency noise.

In a similar spirit, we note that for any dynamical system

$$\dot{x} = Ax + Bu,$$
$$y = Cx,$$

if we hold u at zero we can make use of the general relations

$$\dot{y} = C\dot{x} = CAx,$$
$$\ddot{y} = C\ddot{x} = C\frac{d}{dt}\dot{x} = C\frac{d}{dt}Ax = CA\dot{x} = CA^2x,$$
$$\vdots$$
$$\frac{d^n}{dt^n}y = CA^n x.$$

If we look at how this applies to our modified simple harmonic oscillator example with

$$C = \begin{bmatrix} \alpha & 0 \end{bmatrix}, \quad A = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & 0 \end{bmatrix},$$
$$y = C\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \alpha x_1,$$

we have

$$CA = \begin{bmatrix} \alpha & 0 \end{bmatrix}\begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & 0 \end{bmatrix} = \begin{bmatrix} 0 & \alpha \end{bmatrix},$$
$$\dot{y} = CA\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \alpha x_2,$$

and we start to get a sense for how the natural dynamics A can move information about state space variables into the 'support' of C. Hopefully it should thus seem reasonable that in order for a system to be *observable*, we require that the observability matrix

$$W_o = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{n-1} \end{bmatrix}$$

have full rank. Informally, if a system is observable then we are guaranteed that we can design a procedure (such as the derivatives scheme above) to extract a faithful estimate $\hat{x}$ from y. However, it will generally be necessary to monitor y for some time (and with good accuracy) before the estimation error

$$\tilde{x}(t) \equiv x(t) - \hat{x}(t)$$

can be made small. In the derivatives scheme, for instance, we can't estimate high derivatives of $y(t)$ until we see enough of it to get an accurate determination of its slope, curvature, etc.

State observer with innovations

A more common (and more robust) method for estimating x from y is to construct a state observer that applies corrections to an initial guess $\hat{x}$ until $C\hat{x}$ becomes an accurate predictor of y.

Suppose that at some arbitrary point in time t we have an estimate $\hat{x}(t)$. How should we update this estimate to generate estimates of the state $x(t')$ with $t' > t$? Most simply, we could integrate

$$\frac{d}{dt}\hat{x} = A\hat{x} + Bu,$$

assuming we know A and B for the plant. It is generally assumed that we know u since this signal is under our control! Then we notice that the estimation error $\tilde{x}$ evolves as

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

Hence, this strategy has the nice feature that if A is stable,

$$\lim_{t \to \infty} \tilde{x} = 0,$$

meaning that our estimate will eventually converge to the true system state. Note that this works even if B and u are non-zero.

What if we are not so lucky as to have sufficiently stable natural dynamics A? As mentioned above, a good strategy is to try to apply corrections to $\hat{x}$ at every time step, in proportion to the so-called innovation,

$$w \equiv y - C\hat{x}.$$

Here $y - C\hat{x}$ is the error we make in predicting $y(t)$ on the basis of $\hat{x}(t)$. Clearly when $\tilde{x}$ is small, so is w. A 'Luenberger state observer' can thus be constructed as

$$\frac{d}{dt}\hat{x} = A\hat{x} + Bu + L(y - C\hat{x}),$$

where L is a 'gain' matrix that is left to our design. This observer equation results in

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

Hence we see that our design task should be to choose L, given A and C, such that $(A - LC)$ has nice stable eigenvalues.

This should remind you immediately of the pole-placement problem in state feedback, in which we wanted to choose K, given A and B, such that $(A - BK)$ had desired eigenvalues. Indeed, one can map between the two problems by noting that the transpose of a matrix M^T has the same eigenvalues as M. Thus we can view our observer design problem as being the choice of L^T such that

$$(A - LC)^T = A^T - C^T L^T$$

has nice stable eigenvalues, and this now has precisely the same structure as before. Indeed, there is a complete 'duality' between state feedback and observer design, with correspondences

$$A \leftrightarrow A^T, \quad B \leftrightarrow C^T, \quad K \leftrightarrow L^T, \quad W_r \leftrightarrow W_o^T.$$

Hence it should be clear, for example, how the Matlab function `place` that we mentioned above can be used also for observer design. And as long as the observability matrix has full rank, we are guarenteed to be able to find an L such that $(A - LC)$ has arbitrary desired eigenvalues.

Pole-placement with output feedback

As discussed in section 7.3 of Åström and Murray (http://www.cds.caltech.edu/~murray/amwiki/index.php/Main_Page), the following theorem holds (here we simplify to the $r = 0$ case):

For a system

$$\dot{x} = Ax + Bu,$$
$$y = Cx,$$

the controller described by

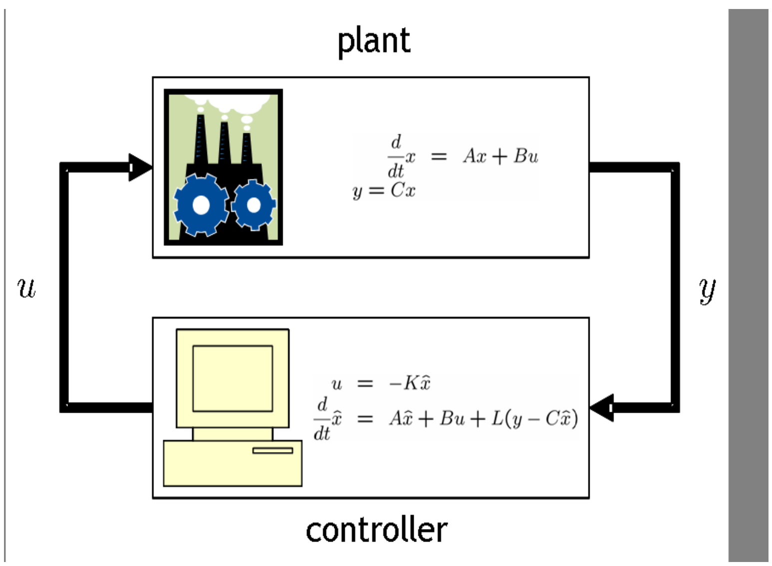
$$u = -K\hat{x},$$
$$\frac{d}{dt}\hat{x} = A\hat{x} + Bu + L(y - C\hat{x})$$

gives a closed-loop system with the characteristic polynomial

$$\det(sI - A + BK)\det(sI - A + LC).$$

This polynomial can be assigned arbitrary roots if the system is observable and reachable.

The overall setup is summarized in the following cartoon:



Next we need to examine how this sort of strategy performs in the presence of noise.

State observers in the presence of noise: some examples

We begin by illustrating some of the concepts we introduced above, on state observers for linear systems.

Consider, as our plant, a simple harmonic oscillator:

$$\frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\omega_0^2 & 0 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix}u,$$

$$\frac{d}{dt}\vec{x} = A\vec{x} + Bu,$$

where as usual $x_1 \leftrightarrow q,\ x_2 \leftrightarrow \dot{q}.$ The dynamics as given fix A and $B,$ and let us consider a general output signal related linearly to the state:

$$y = C\vec{x} = \begin{bmatrix} C_1 & C_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}.$$

We have seen that the observability criterion is that we have full rank for the matrix

$$W_o = \begin{bmatrix} C \\ CA \end{bmatrix},$$

and since

$$CA = \begin{bmatrix} C_1 & C_2 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ -\omega_0^2 & 0 \end{bmatrix} = \begin{bmatrix} -\omega_0^2 C_2 & C_1 \end{bmatrix},$$

we have

$$\det W_0 = \det \begin{bmatrix} C_1 & C_2 \\ -\omega_0^2 C_2 & C_1 \end{bmatrix} = C_1^2 + \omega_0^2 C_2^2.$$

Hence as long as $\omega_0 \neq 0,$ the system is observable as long as C is nonzero.

For general $C,$ our linear (Luenberger) observer structure is

$$\frac{d}{dt}\hat{x} = A\hat{x} + Bu + L(y - C\hat{x}),$$

which induces the following dynamics for the estimation error:

$$\frac{d}{dt}\tilde{x} = (A - LC)\tilde{x}.$$

We thus want to design L to make the eigenvalues of $A - LC$ have negative real part. We noted that we could do this by using Matlab's pole-placement routine, **`place`**, with

$$A \leftrightarrow A^T, \quad B \leftrightarrow C^T, \quad K \leftrightarrow L^T.$$

We try the following examples, setting $\omega_0 = 1$ and designing for eigenvalues $\{-1, -2\}$:

$$C = \begin{bmatrix} 1 & 0 \end{bmatrix} : L = \begin{bmatrix} 3 \\ 1 \end{bmatrix}, \quad L(y - C\hat{x}) = \begin{bmatrix} 3 \\ 1 \end{bmatrix}(x_1 - \hat{x}_1),$$

$$C = \begin{bmatrix} 0 & 1 \end{bmatrix} : L = \begin{bmatrix} -1 \\ 3 \end{bmatrix}, \quad L(y - C\hat{x}) = \begin{bmatrix} -1 \\ 3 \end{bmatrix}(x_2 - \hat{x}_2),$$

$$C = \begin{bmatrix} 1 & 1 \end{bmatrix} : L = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \quad L(y - C\hat{x}) = \begin{bmatrix} 1 \\ 2 \end{bmatrix}(x_1 - \hat{x}_1 + x_2 - \hat{x}_2).$$

To examine how the observers work, we perform some numerical integrations. Setting $u = 0$ and

$$\vec{x}(0) = \begin{bmatrix} 1 \\ 1 \end{bmatrix},$$

we have for the plant evolution

$$\vec{x}(t) = \exp(At)\vec{x}(0) \mapsto \begin{bmatrix} \cos t & \sin t \\ -\sin t & \cos t \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} \cos t + \sin t \\ \cos t - \sin t \end{bmatrix}.$$

For the observer we assume no knowledge of the initial state, and thus set

$$\hat{x}(0) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

The dynamics of the state estimate is

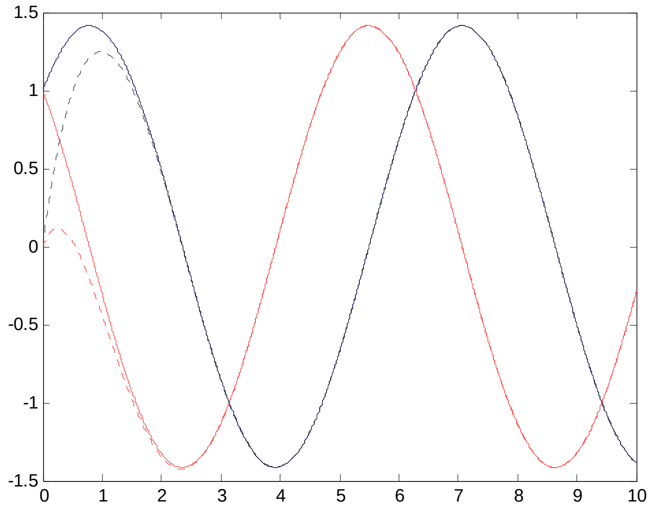
$$\begin{aligned} \frac{d}{dt}\hat{x} &= \begin{bmatrix} 0 & 1 \\ -\omega_0^2 & 0 \end{bmatrix} \begin{bmatrix} \hat{x}_1 \\ \hat{x}_2 \end{bmatrix} + Ly - LC\hat{x} \\ &= \left(\begin{bmatrix} 0 & 1 \\ -\omega_0^2 & 0 \end{bmatrix} - LC \right) \begin{bmatrix} \hat{x}_1 \\ \hat{x}_2 \end{bmatrix} + Ly. \end{aligned}$$

For the purposes of this example we could actually integrate this analytically, treating y as a driving term. However in the spirit of recursive state estimation we instead integrate numerically (Matlab example).

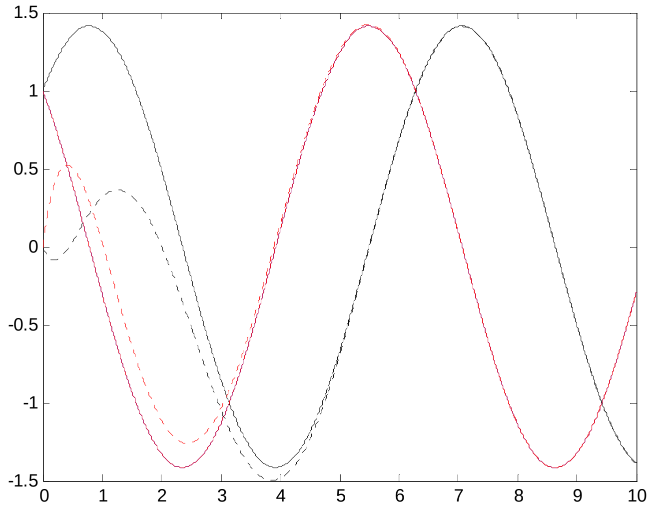
```
Ttot=10; Nsteps=5000;
t=linspace(0,Ttot,Nsteps); dt=t(2)-t(1);
x=[cos(t)+sin(t);cos(t)-sin(t)];
xhat=zeros(2,Nsteps);
for ii=2:Nsteps,
 y=C*x(:,ii);
 xhat(:,ii) = xhat(:,ii-1) + dt*(A*xhat(:,ii-1)+L*(y-C*xhat(:,ii-1)));
end;
```

In the following we plot the results, with x_1 as solid black, x_2 as solid red, $\hat{x}_1$ as dashed black, and $\hat{x}_2$ as dashed red.

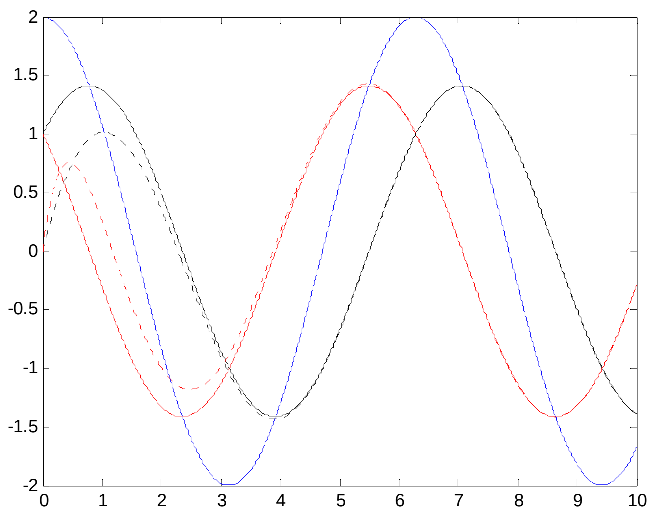
The results for $C = \begin{bmatrix} 1 & 0 \end{bmatrix}$:



The results for $C = \begin{bmatrix} 0 & 1 \end{bmatrix}$:



The results for $C = \begin{bmatrix} 1 & 1 \end{bmatrix}$, with y in blue:



Could we have guessed the forms of these observers?

For $C = \begin{bmatrix} 1 & 0 \end{bmatrix}$, note that we can construct an "intuitive" observer by setting $\hat{x}_1 = y$, $\hat{x}_2 = \dot{y}$, which should work well as long as there is negligible measurement noise.

For $C = \begin{bmatrix} 0 & 1 \end{bmatrix}$ we obviously have $\hat{x}_2 = y$, and we can guess

$$\hat{x}_1 = \int_0^t ds\, y(s),$$

but it is not entirely clear how we should choose $\hat{x}_1(0)$. Note that the Luenberger observer does something more complicated, as it integrates

$$\frac{d}{dt}\hat{x} = \left(\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} - LC\right)\begin{bmatrix} \hat{x}_1 \\ \hat{x}_2 \end{bmatrix} + Ly$$

$$\equiv A'\hat{x} + Ly, \quad A' = \begin{bmatrix} 0 & 2 \\ -1 & -3 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ -1 & -\frac{1}{2} \end{bmatrix}\begin{bmatrix} -2 & 0 \\ 0 & -1 \end{bmatrix}\begin{bmatrix} -1 & -2 \\ 2 & 2 \end{bmatrix},$$

$$\exp(A't) = \begin{bmatrix} 1 & 1 \\ -1 & -\frac{1}{2} \end{bmatrix}\begin{bmatrix} e^{-2t} & 0 \\ 0 & e^{-t} \end{bmatrix}\begin{bmatrix} -1 & -2 \\ 2 & 2 \end{bmatrix} = \begin{bmatrix} 2e^{-t} - e^{-2t} & 2e^{-t} - 2e^{-2t} \\ e^{-2t} - e^{-t} & 2e^{-2t} - e^{-t} \end{bmatrix},$$

$$\hat{x}(t) = \exp(A't)\left\{\hat{x}(0) + \int_0^t ds\, \exp(-A's)Ly\right\}$$

$$= \begin{bmatrix} 2e^{-t} - e^{-2t} & 2e^{-t} - 2e^{-2t} \\ e^{-2t} - e^{-t} & 2e^{-2t} - e^{-t} \end{bmatrix}\left\{\hat{x}(0) + \int_0^t ds \begin{bmatrix} 4e^{-s} - 5e^{-2s} \\ 5e^{-2s} - 2e^{-s} \end{bmatrix} y\right\},$$

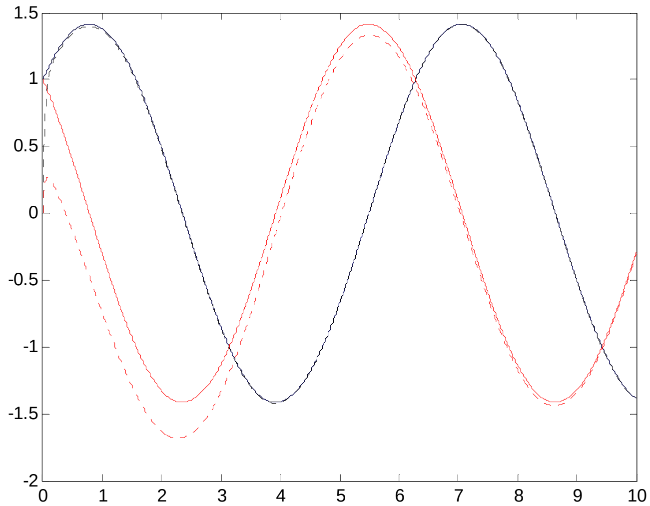
with $\hat{x}(0)$ arbitrary.

There doesn't seem to be an obvious intuitive strategy for $C = \begin{bmatrix} 1 & 1 \end{bmatrix}$.

Before turning to consider noisy observation scenarios, we take a brief look at the behavior of the Luenberger observer for $C = \begin{bmatrix} 1 & 0 \end{bmatrix}$ and varying gain. Recall that when we designed for eigenvalues $\{-1, -2\}$ we obtained

$$L \to \begin{bmatrix} 3 \\ 1 \end{bmatrix}.$$

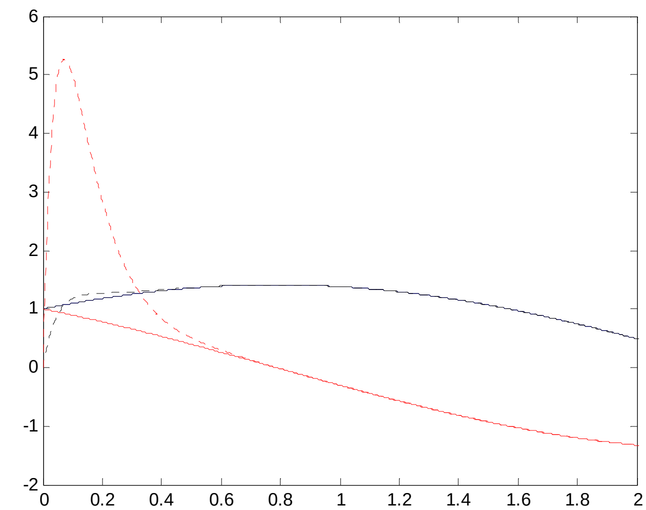
What happens if we try simply multiplying this gain by 10?



If instead we design for eigenvalues $\{-10, -20\}$ we get

$$L \to \begin{bmatrix} 30 \\ 199 \end{bmatrix},$$

and the performance is transiently bad, but does indeed settle quite quickly:



Stochastic models (notation)

We will normally write linear stochastic control models in the form

$$dx_t = Ax_t\,dt + Bu_t\,dt + F dV_t,$$
$$dy_t = Cx_t\,dt + G\,dW_t.$$

Here the subscripts serve to remind us of things that depend on time, and the vector nature of x and/or y is implicit. The stochastic increments dV_t and dW_t satisfy

$$\langle dV_t\rangle = \langle dW_t\rangle = 0,$$
$$dV_t^2 = dW_t^2 = dt,$$
$$dV_t dt = dW_t dt = 0,$$

and for $s \neq t$ we have $\langle dV_s dV_t\rangle = \langle dW_s dW_t\rangle = 0.$ We can informally think of dV_t and dW_t as gaussian white noises with zero mean and variance dt. It is conventional to refer to dV_t as process noise and to dW_t as measurement noise or observation noise.

It is important to be aware of the fact that stochastic differential equations (SDE's) of the type we have written are, rigorously speaking, a sort of shorthand notation for stochastic integrals. There is an important distinction between Itô and Stratonovich stochastic integrals, and therefore between Itô and Stratonovich SDE's. In control theory one normally works with Itô SDE's, and in any case there is a straightforward recipe for converting a model between Itô and Stratonovich forms.

The Stratonovich form is sometimes preferred (especially in physics) because Stratonovich SDE's can be manipulated using standard calculus. For Itô SDE's, however, one must in general be careful to observe the Itô Rule, which says that if x_t obeys the Itô SDE

$$dx_t = A(x_t)dt + B(x_t)dW_t,$$

then a variable y_t related to x_t via

$$y_t = U(x_t)$$

evolves according to

$$dy_t = \left[A(x_t)\frac{\partial U}{\partial x} + \frac{1}{2}B^2(x_t)\frac{\partial^2 U}{\partial x^2}\right]dt + B(x_t)\frac{\partial U}{\partial x}dW_t,$$

where the second-derivative term in the square brackets is known as the Itô correction. Note that if U is a linear function the Itô correction vanishes and we recover the prediction of normal calculus.

An important advantage of working with Itô SDE's is that if x_t obeys the Itô SDE

$$dx_t = A(x_t)dt + B(x_t)dW_t,$$

then x_t is uncorrelated with $dW_t.$ This considerably simplifies the computation of statistical moments.

For example consider the linear SDE model

$$dx_t = Ax_t dt + FdV_t,$$

with x_t a scalar and $A < 0$ (the Ornstein-Uhlenbeck model). We then have

$$d\langle x_t\rangle = A\langle x_t\rangle dt + F\langle dV_t\rangle$$
$$= A\langle x_t\rangle dt,$$
$$\langle x_t\rangle = \langle x_0\rangle \exp(At),$$

and if $y_t = x_t^2,$ so that $\langle y_t\rangle$ is the variance of $x_t,$

$$dy_t = [2Ax_t^2 + F^2]dt + 2Fx_t dV_t,$$
$$d\langle y_t\rangle = [2A\langle y_t\rangle + F^2]dt + 2F\langle x_t dV_t\rangle$$
$$= [2A\langle y_t\rangle + F^2]dt + 2F\langle x_t\rangle\langle dV_t\rangle$$
$$= [2A\langle y_t\rangle + F^2]dt,$$
$$\langle y_t\rangle = \exp(2At)\left\{\langle y_0\rangle + \int_0^t ds\exp(-2As)F^2\right\}$$
$$= \exp(2At)\left\{\langle y_0\rangle + F^2\int_0^t ds\exp(-2As)\right\}.$$

If we assume that x_t evolves from a known value x_0 at $t = 0,$ then $\langle x_0\rangle = x_0$ and $\langle y_0\rangle = x_0^2,$ and the mean-square uncertainty in x_t is

$$\langle x_t^2\rangle - \langle x_t\rangle^2 = \langle y_t\rangle - \langle x_t\rangle^2$$
$$= \exp(2At)F^2\int_0^t ds\exp(-2As)$$
$$= \exp(2At)F^2\left(-\frac{1}{2A}\right)(\exp(-2At) - 1)$$
$$= -\frac{F^2}{2A}(1 - \exp(2At)).$$

The mean-square uncertainty thus has a steady-state value as $t \to \infty,$

$$\langle x_t^2\rangle - \langle x_t\rangle^2 \to \frac{F^2}{2|A|}.$$

In numerical simulations, we can simply update x_t according to

$$x_{t+dt} = x_t + Ax_t\,dt + Bu_t\,dt + F\,dV_t,$$
$$dy_t = Cx_t\,dt + G\,dW_t,$$

where dV_t and dW_t are independent normal random variables with variance dt. In Matlab, if **dt** is a variable with some assigned numerical value,

```
dVt=sqrt(dt)*randn(1); dWt=sqrt(dt)*randn(1);
```

This simple procedure is known as the Itô-Euler stochastic integration routine, which is easy to implement but has the disadvantage that it only converges to order $(dt)^{1/2}$. Higher-order integrators can be found in various computer packages (including SDE toolboxes for Matlab), and are described in textbooks.

State observers - performance with noise

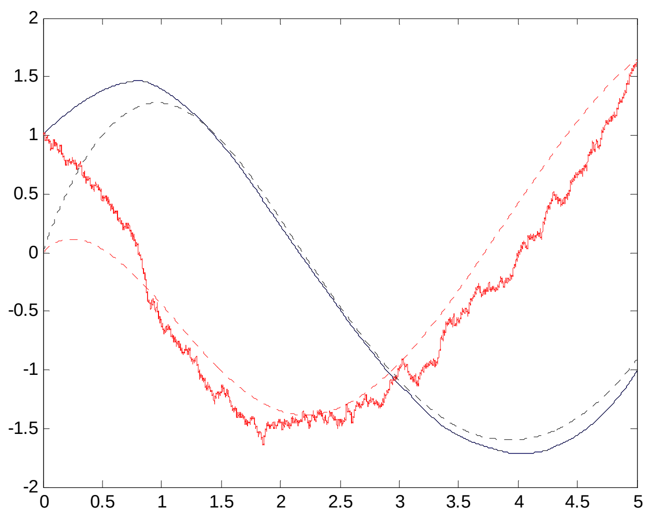
First we consider the case of process noise only. Returning to our simple harmonic oscillator with $C = \begin{bmatrix} 1 & 0 \end{bmatrix}$, we can add a noisy force acting on the oscillator by setting

$$F = \begin{bmatrix} 0 \\ 0.3 \end{bmatrix}.$$

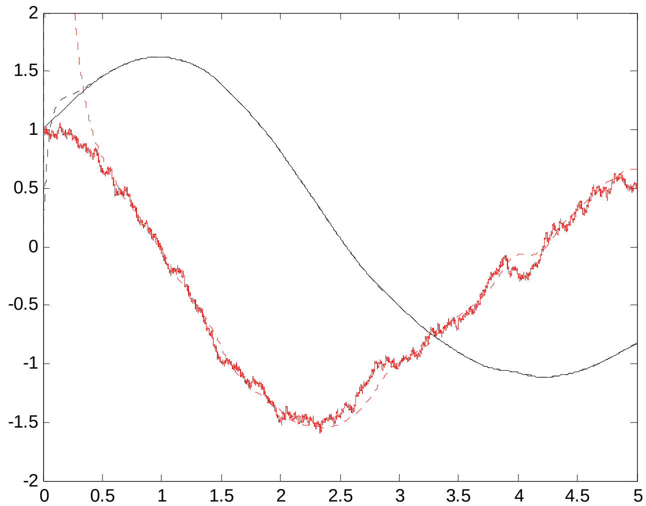
Simulating both the plant and the response of the Luenberger observer, which we now write as

$$\hat{x}_{t+dt} = \hat{x}_t + (A - LC)\hat{x}_t\,dt + Bu_t\,dt + L\,dy_t,$$

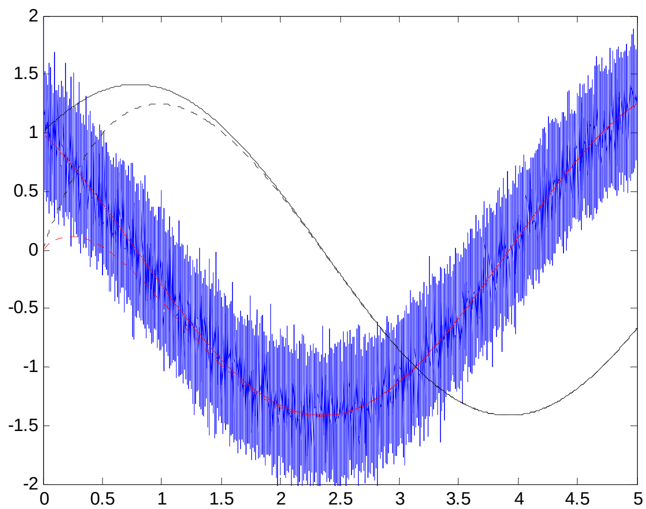
we obtain:



Here we used the L computed for target eigenvalues $\{-1, -2\}$. It is clear that the observer still tracks the state but with degraded performance due to the process noise. If we try turning up the observer gain by using values computed for target eigenvalues $\{-10, -20\}$, we do much better:

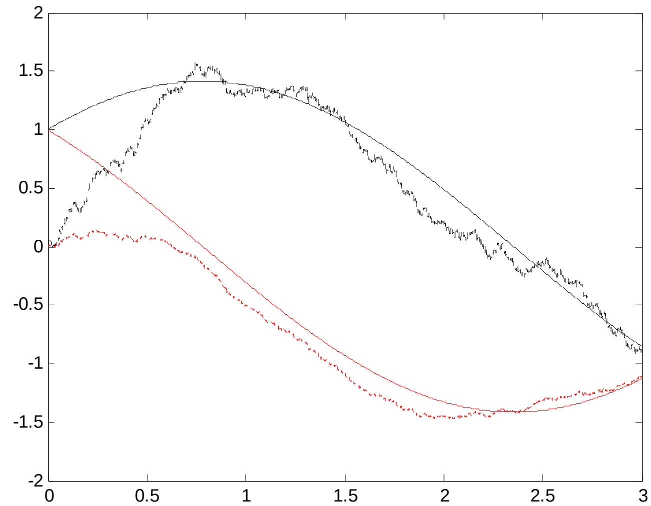


Next we set F to zero but $G = 0.001$. The results are:

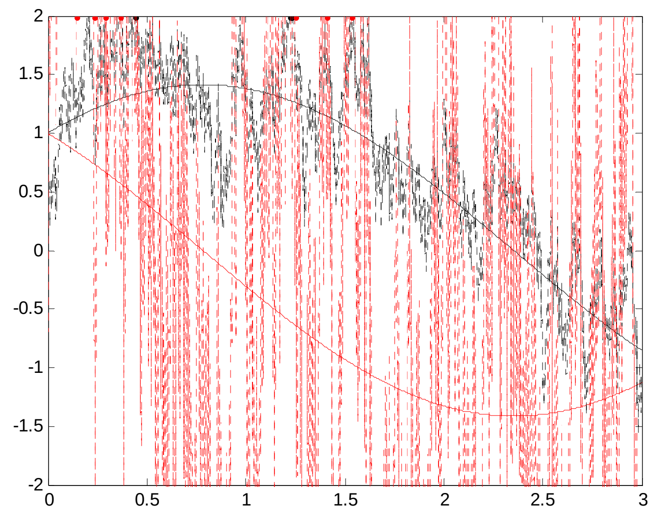


Here the blue shows the performance of a naive velocity estimator, $\hat{x}_2 = \dot{y}$ (note that there is some aliasing). The Luenberger observer does much better, as expected.

In order to bother the Luenberger observer we turn G all the way up to 0.1:



Now if we try to get greedy with this much observation noise, by turning up L to the values that would achieve eigenvalues $\{-10, -20\}$ in the noiseless system,



and we see that the estimation of velocity becomes very poor. So apparently too much observer gain is a bad thing, when there is noise. Is there an optimal value of the Luenberger gain? This would seem to be an especially important question when there is both process noise and measurement noise.

The Kalman-Bucy filter

To answer this sort of question we first have to state how we judge the observer's performance quantitatively. It is most common to adopt a minimum-least-squares framework, in which our objective is to design the estimator (method of generating $\hat{x}_t$ from knowledge of y_s with $s \leq t$) that achieves the lowest possible value of $\langle (x_t - \hat{x}_t)(x_t - \hat{x}_t)^T \rangle$. As discussed in A&M section 7.4, for the plant and observation model

$$dx_t = Ax_t\,dt + Bu_t\,dt + F\,dV_t,$$
$$dy_t = Cx_t\,dt + G\,dW_t,$$

we have the following Theorem:

(Kalman-Bucy, 1961) The optimal estimator has the form of a linear observer

$$d\hat{x}_t = (A\hat{x}_t + Bu_t)dt + L_t(dy_t - C\hat{x}_t dt), \quad \hat{x}_0 = \langle x_0 \rangle,$$

where $L_t = P_t C^T [GG^T]^{-1}$ and $P_t = \langle (x_t - \hat{x}_t)(x_t - \hat{x}_t)^T \rangle$ is the (symmetric and positive-definite) estimation error covariance matrix that satisfies the following matrix Riccati equation:

$$\frac{d}{dt} P_t = FF^T + AP_t + P_t A^T - P_t C^T [GG^T]^{-1} C P_t, \quad P_0 = \langle x_0 x_0^T \rangle.$$

It is important to note that the Kalman filter provides both a point estimate of the evolving system state and a computation of the estimation error covariance matrix - it gives you its best guess *and* a numerical uncertainty. When the system is stationary and if P_t converges, the observer gain settles to a constant:

$$L = PC^T [GG^T]^{-1}, \quad FF^T + AP + PA^T - PC^T [GG^T]^{-1} CP = 0.$$

The second equation is called the *algebraic Riccati equation*, and may be solved using Matlab's **`lqe`** function.

We see that the essence of Kalman filtering is an optimal choice of the observer gain, which may be time-dependent in a way that reflects our evolving degree of confidence in our state estimate. The general structure is to apply high observer gain when we have large uncertainty, and to reduce it when our uncertainty approaches a limiting value set by the process and measurement noises.

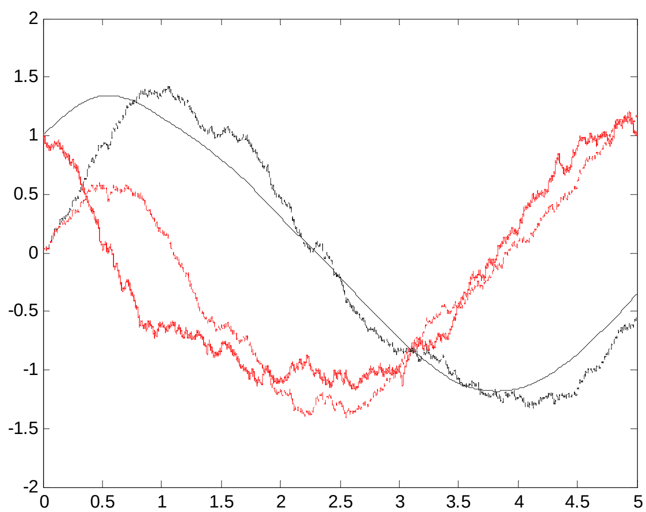
As an example let us compute the Kalman gain for our simple harmonic oscillator example with

$$F = \begin{bmatrix} 0 \\ 0.3 \end{bmatrix}, \quad G = 0.1.$$

This results in

$$L \approx \begin{bmatrix} 2.08 \\ 2.16 \end{bmatrix},$$

and a simulation looks as follows:



It is interesting to note that, as a consequence of its least-squares optimality, the Kalman-Bucy filter achieves what is known as "whitening" of the innovations process $dy_t - C\hat{x}_t dt$. That is, if $\hat{x}_t$ is propagated by the Kalman-Bucy filter then $(dy_t - C\hat{x}_t dt)$ becomes a completely random signal (Gaussian white noise); roughly we can think that $\hat{x}_t$ becomes good enough that subtracting $C\hat{x}_t\, dt$ from dy_t removes all the information from the observed signal. The notions of least-squares optimal state estimation, the innovations process, and whitening all carry over to nonlinear scenarios.