Backstepping

In complicated systems, like aircraft, there are often parameters which are not fully known, or which might even occasionally change. To cope with this, you need a controller that adapts to these parameters: an adaptive controller. In this summary we'll be investigating how such controllers work. In particular, we'll be investigating the backstepping controller.

1 Basic definitions and theories

1.1 Function definitions

Before we're discussing systems, we start with some formal function definitions. Let's examine a function $V(\mathbf{x})$. We say that $V(\mathbf{x})$ is...

- **positive definite** if $V(\mathbf{0}) = 0$ and $V(\mathbf{x}) > 0$ with $\mathbf{x} \neq \mathbf{0}$.
- **positive semidefinite** if $V(\mathbf{0}) = 0$ and $V(\mathbf{x}) \geq 0$ with $\mathbf{x} \neq \mathbf{0}$.
- **negative (semi)definite** if $-V(\mathbf{x})$ is positive (semi)definite.
- **radially unbounded** if $V(x) \to \infty$ as $|\mathbf{x}| \to \infty$.

1.2 System definitions

Let's examine a system with state $\mathbf{x}$ and dynamics $\dot{\mathbf{x}} = f(\mathbf{x})$. A function $\mathbf{x}(t)$ with initial state $\mathbf{x}(0) = \mathbf{x_0}$ that satisfies the system dynamics is called a **solution** of the system. A system is called...

- **stable** if, for given $\epsilon > 0$, there exists a $\delta(\epsilon) > 0$ such that all solutions with initial conditions $|\mathbf{x}(0)| < \delta$ satisfy $|\mathbf{x}(t)| < \epsilon$ for all $t \geq 0$. More intuitively speaking, all solutions starting near $\mathbf{x} = \mathbf{0}$ remain bounded.
- **asymptotically stable** (AS) if it is stable and a δ can be found such that all solutions with $|\mathbf{x}(0)| < \delta$ satisfy $|\mathbf{x}(t)| \to \mathbf{0}$ as $t \to \infty$. More intuitively speaking, all solutions starting near $\mathbf{x} = \mathbf{0}$ are bounded and converge to zero.
- **globally asymptotically stable** (GAS) if it is asymptotically stable for any initial state $\mathbf{x}(0)$.

1.3 The Lyapunov theory

Let's say we have a time-invariant system with state $\mathbf{x}$ and dynamics $\dot{\mathbf{x}} = f(\mathbf{x})$. How do we prove that the system is stable? For that, we can use the Lyapunov theory. The first thing which we need is a Lyapunov function $V(\mathbf{x})$. This function has to be positive definite in a region Γ near $\mathbf{x} = \mathbf{0}$. (It often helps to think of V as some kind of energy. It is never negative, and can only be zero in the zero state.) Second, we will examine $\dot{V}$. We can rewrite this as

$$\dot{V}(\mathbf{x}) = \frac{\partial V(\mathbf{x})}{\partial t} = \frac{\partial V(\mathbf{x})}{\partial \mathbf{x}} \frac{\partial \mathbf{x}}{\partial t} = \frac{\partial V(\mathbf{x})}{\partial \mathbf{x}} f(\mathbf{x}). \tag{1.1}$$

The Lyapunov theory now states that...

- if $\dot{V}(\mathbf{x})$ is negative semidefinite in the region Γ, then the solution is stable.

- if $\dot{V}(\mathbf{x})$ is negative definite in the region Γ, then the solution is asymptotically stable.
- if $V(\mathbf{x})$ positive definite and radially unbounded for all $\mathbf{x}$, and if $\dot{V}(\mathbf{x})$ is negative definite for all $\mathbf{x}$, then the solution is globally asymptotically stable.

The Lyapunov theory is actually quite logical. If you have some function that is always decreasing, then it must reach zero eventually. So there is no way that the system diverges: it has to be stable.

2 Applying adaptive control

2.1 The problem statement

Let's examine a time-invariant system with an unknown parameter. For example, consider

$$\dot{x} = \theta x + u, \tag{2.1}$$

where θ is unknown. We could use as a controller $u = -kx$. If $\theta < k$, this will ensure stability. However, if $\theta > k$, we have an instable system. A possible solution would be to use a nonlinear controller, like $u = -k_1 x - k_2 x^3$. Although this ensures that the system state is bounded, it does not ensure an asymptotically stable system. Instead, it will be better to make our system adapt to θ.

2.2 Estimation-based adaptive control

Let's suppose that we can measure the state x of the system, but not the state derivative $\dot{x}$. Just deriving $\dot{x}$ from x is in practice unwise, due to measurement noise. Instead, we filter the state x and the input u first. The filtered signals are denoted as

$$x_f = \frac{x}{s+1} \qquad \text{and} \qquad u_f = \frac{u}{s+1}. \tag{2.2}$$

Using $\dot{x} = sx$, we can now rewrite the system to

$$x = (\theta + 1)x_f + u_f. \tag{2.3}$$

We will also have an **estimated value** of θ, denoted as $\hat{\theta}$. (Initially, we can assume that $\hat{\theta} = 0$.) We can use $\hat{\theta}$ to predict the state x of the system. We will find

$$\hat{x} = (\hat{\theta} + 1)x_f + u_f. \tag{2.4}$$

If our estimate $\hat{\theta}$ is inaccurate, we will of course get an error in our **prediction** $\hat{x}$. This error is

$$e = x - \hat{x} = (\theta - \hat{\theta})x_f. \tag{2.5}$$

To reduce this **prediction error** e, we should adjust our estimate $\hat{\theta}$. One way to do this is by using

$$\dot{\hat{\theta}} = -\frac{\gamma}{2}\frac{\partial(e^2)}{\partial\hat{\theta}} = \gamma e \frac{\partial e}{\partial\hat{\theta}} = -\gamma e x_f. \tag{2.6}$$

By doing this, the prediction error e will decrease. If we then also use the controller $u = -(k + \hat{\theta})x$, then our problem is solved. The error will decrease.

2.3 Lyapunov-based adaptive control

Another way in which to stabilize the system is by using the Lyapunov theory. To do this, we first need a Lyapunov function. We could try the function

$$V(x,\hat{\theta}) = \frac{1}{2}x^2 + \frac{1}{2}\left(\theta - \hat{\theta}\right)^2. \tag{2.7}$$

$\dot{V}$ can be found according to

$$\dot{V} = x\dot{x} - \left(\theta - \hat{\theta}\right)\dot{\hat{\theta}} = x\left(u + \theta x\right) - \left(\theta - \hat{\theta}\right)\dot{\hat{\theta}} = xu + \hat{\theta}\dot{\hat{\theta}} + \theta\left(x^2 - \dot{\hat{\theta}}\right). \tag{2.8}$$

We want to choose our input u and our update law $\dot{\hat{\theta}}$ such that $\dot{V}$ is negative definite. However, neither u nor $\dot{\hat{\theta}}$ can depend on the unknown θ. To solve this problem, we can first set $\dot{\hat{\theta}} = x^2$. This eliminates the term on the right. For the input, we then set $u = -(k + \hat{\theta})x$. This results in $\dot{V} = -kx^2$. If $k > 0$, this is evidently negative definite. The system has thus been stabilized.

2.4 A more complicated problem

We have seen two techniques now: estimation-based and Lyapunov-based adaptive control. But do these techniques always work? We could consider the nonlinear system

$$\dot{x} = \theta x^2 + u. \tag{2.9}$$

Both techniques will use the control law

$$u = -kx - \hat{\theta}x^2. \tag{2.10}$$

But what about the parameter update law? For estimation-based adaptive control it is hard to choose a good parameter update law. If we choose the same law as before, we will get an unstable system. Lyapunov-based adaptive control works better here. Using the same Lyapunov function as before, we find the update law

$$\dot{\hat{\theta}} = x^3 \tag{2.11}$$

which results in a stable system. This shows the advantage of Lyapunov-based adaptive control. So from now on, we will not consider estimation-based adaptive control anymore. Instead, we focus on Lyapunov-based adaptive control. But plenty of challenges await us still. How do we find a Lyapunov function from which we can derive a control law? (Such a function is called a **control Laypunov function** (CLF).) And how do we choose the best control law $u = \alpha(x)$?

2.5 Sontag's formula

Let's suppose that we have a control problem of the form

$$\dot{x} = f(x) + g(x)u, \tag{2.12}$$

where $f(0) = 0$. Also, we've already found a suitable CLF. One way to choose a good control law is by using **Sontag's formula**

$$u = \alpha_s(x) = \begin{cases} -\frac{\frac{\partial V}{\partial x}f + \sqrt{\left(\frac{\partial V}{\partial x}f\right)^2 + \left(\frac{\partial V}{\partial x}g\right)^4}}{\left(\frac{\partial V}{\partial x}g\right)} & \text{for } \frac{\partial V}{\partial x}g \neq 0, \\ 0 & \text{for } \frac{\partial V}{\partial x}g = 0. \end{cases} \tag{2.13}$$

The resulting control law often uses parts of the system to help stabilize it. This reduces the magnitude of the control input u. So, the control law is relatively efficient.

3 The backstepping algorithm

3.1 A backstepping example

The control problem becomes more complicated if the input u doesn't directly influence x. As an example, we can consider a system with an integrator, like

$$\dot{x} = x - x^3 + \xi, \tag{3.1}$$
$$\dot{\xi} = u. \tag{3.2}$$

Let's only examine the first equation, and act as if ξ was our control input. A CLF would be $V = \frac{1}{2}x^2$. It follows that $\xi = -(k+1)x$. This would stabilize the system. However, we can't control ξ directly. Instead, we specify the desired value

$$\xi_d = \alpha(x) = -(k+1)x. \tag{3.3}$$

In this case, we call ξ the **virtual control**. The function $\alpha(x)$ which it should follow is the **stabilizing function**. The deviation of ξ from $\alpha(x)$ is called the **error state** z. It is defined as

$$z = \xi - \xi_d = \xi - \alpha(x) = \xi + (k+1)x. \tag{3.4}$$

We can use the error state to rewrite our system to

$$\dot{x} = x - x^3 + z + \alpha(x) = -kx - x^3 + z, \tag{3.5}$$
$$\dot{z} = \dot{\xi} - \dot{\alpha} = u + (k+1)\dot{x} = u + (k+1)(-kx - x^3 + z). \tag{3.6}$$

We now also need to incorporate z in our Lyapunov function. A possible function is

$$V(x,z) = \frac{1}{2}x^2 + \frac{1}{2}z^2 = \frac{1}{2}x^2 + \frac{1}{2}\left(\xi + (k+1)x\right)^2. \tag{3.7}$$

Taking the derivative gives

$$\dot{V}(x,z) = x\dot{x} + z\dot{z} = x(-kx - x^3 + z) + z\left(u + (k+1)(-kx - x^3 + z)\right) \tag{3.8}$$
$$= -kx^2 - x^4 + z\left(x + u + (k+1)(-kx - x^3 + z)\right). \tag{3.9}$$

As a control law, we can now use

$$u = -cz - x - (k+1)(-kx - x^3 + z), \tag{3.10}$$

which turns the Lyapunov derivative into

$$\dot{V}(x,z) = -kx^2 - x^4 - cz^2. \tag{3.11}$$

This is evidently negative definite, so the system has been stabilized.

Now what have we done? We wanted to stabilize x. To do this, we provided a desired value for ξ which would accomplish this. We continued by defining an error signal z, which required to be stabilized as well. To do this, we set the value for u. If the system would have more equations, we could continue stepping back through equations like this. This procedure is thus called **backstepping**.

3.2 The backstepping algorithm

The procedure from the previous paragraph can be generalized. Consider a system of the form

$$\dot{x}_1 = f_1(x_1) + g_1(x_1)x_2, \tag{3.12}$$
$$\dot{x}_2 = f_2(x_1, x_2) + g_1(x_1, x_2)x_3, \tag{3.13}$$
$$\vdots \tag{3.14}$$
$$\dot{x}_n = f_n(x_1, x_2, \ldots, x_n) + g_n(x_1, x_2, \ldots, x_n)u. \tag{3.15}$$

We want the output $y = x_1$ to follow a certain reference signal $y_r(t)$. When applying backstepping, we continuously need to perform the following steps, starting at $i = 1$ and continuing until $i = n$.

1. At equation i, define the error signal $z_i = x_i - x_{i,d}$. (Note that $x_{1,d} = y_r$.)
2. Rewrite the equation using the error signal z_i. (So use $\dot{z}_i = \dot{x}_i - \dot{x}_{i,d}$ and substitute x_i by $z_i + x_{i,d}$.)
3. Treat x_{i+1} as if it's the control input. (It is in fact the virtual control.)
4. Find a CLF for the system so far. (That is, for equations 1 to i.) A suggestion would be
$$V_i = \frac{1}{2}z_1^2 + \ldots + \frac{1}{2}z_i^2. \tag{3.16}$$
5. Use the CLF to derive an expression $x_{i+1,d}$ (the stabilizing function) for the virtual control x_{i+1}. (Note that, at equation n, you use the input u instead of x_{n+1}.)

Applying the above steps for $i = 1$ to $i = n$ will stabilize the system.

3.3 The idea of adaptive backstepping – step 1

In the problem of the previous paragraph, we assumed that the model was completely known. This is of course not always the case. This time, we will examine a model of the form

$$\begin{aligned}
\dot{x}_1 &= \theta^T \mathbf{f_1}(x_1) + g_1(x_1)x_2, && (3.17)\\
\dot{x}_2 &= \theta^T \mathbf{f_2}(x_1, x_2) + g_1(x_1, x_2)x_3, && (3.18)\\
&\vdots && (3.19)\\
\dot{x}_n &= \theta^T \mathbf{f_n}(x_1, x_2, \ldots, x_n) + g_n(x_1, x_2, \ldots, x_n)u. && (3.20)
\end{aligned}$$

Note that this time the known functions $\mathbf{f_i}$ return vectors. Also, the unknown parameter vector θ is a vector. The output $y = x_1$ of this system should follow some reference output y_r. To accomplish this, we will need **adaptive backstepping**.

Before we are going to set up an algorithm for adaptive backstepping, we'll first try to apply the normal backstepping algorithm. So we define the error signal z_1

$$z_1 = x_1 - y_r. \tag{3.21}$$

We then rewrite the first of the system equations to

$$\dot{z}_1 = \dot{x}_1 - \dot{y}_r = \theta^T \mathbf{f_1}(z_1 + y_r) + g_1(z_1 + y_r)x_2 - \dot{y}_r. \tag{3.22}$$

But this is where we expand the backstepping algorithm, by introducing an estimate $\hat{\theta}$ of θ. We also denote the **estimate error** as $\tilde{\theta} = \theta - \hat{\theta}$. (So we have $\dot{\tilde{\theta}} = -\dot{\hat{\theta}}$.) We now use, as Lyapunov function for the first equation,

$$V_1(z_1, \tilde{\theta}) = \frac{1}{2}z_1^2 + \frac{1}{2}\tilde{\theta}^T \Gamma^{-1} \tilde{\theta}. \tag{3.23}$$

Here, $\Gamma = \Gamma^T > 0$ is the **adaptation gain matrix**. It determines how fast we adapt the various parameters of $\hat{\theta}$. The Lyapunov derivative then equals

$$\begin{aligned}
\dot{V}_1 &= z_1 \dot{z}_1 + \tilde{\theta}^T \Gamma^{-1} \dot{\tilde{\theta}} && (3.24)\\
&= z_1 \left(\theta^T \mathbf{f_1}(z_1 + y_r) + g_1(z_1 + y_r)x_2 - \dot{y}_r\right) - \left(\theta - \hat{\theta}\right)^T \Gamma^{-1} \dot{\hat{\theta}} && (3.25)\\
&= z_1 \left(g_1(z_1 + y_r)x_2 - \dot{y}_r\right) + \theta^T \left(z_1 \mathbf{f_1}(z_1 + y_r) - \Gamma^{-1}\dot{\hat{\theta}}\right) + \hat{\theta}^T \Gamma^{-1} \dot{\hat{\theta}}. && (3.26)
\end{aligned}$$

(Note that z_1 is a scalar, so we can move it around in the order of multiplication.) We don't know θ. So a nice update law for our prediction $\hat{\theta}$ will be

$$\dot{\hat{\theta}} = z_1 \Gamma \mathbf{f_1}(z_1 + y_r). \tag{3.27}$$

This turns the expression for $\dot{V}_1$ into

$$\dot{V}_1 = z_1 \left(\hat{\theta}^T \mathbf{f_1}(z_1 + y_r) + g_1(z_1 + y_r)x_2 - \dot{y}_r \right). \tag{3.28}$$

An evident choice for the desired value of x_2, denoted as $x_{2,d}$, now is

$$x_{2,d} = \frac{\dot{y}_r - c_1 z_1 - \hat{\theta}^T \mathbf{f_1}(z_1 + y_r)}{g_1(z_1 + y_r)}. \tag{3.29}$$

3.4 The idea of adaptive backstepping – step 2

Using $x_{2,d}$, we can step back to the second equation of the system. We can then rewrite it using $z_2 = x_2 - x_{2,d}$. The Lyapunov function is

$$V_2 = \frac{1}{2} z_1^2 + \frac{1}{2} z_2^2 + \frac{1}{2} \tilde{\theta}^T \Gamma^{-1} \tilde{\theta}. \tag{3.30}$$

The Lyapunov derivative $\dot{V}_2$ is a bit harder to work out. (For brevity, we will write $\mathbf{f_1}(z_1 + y_r)$ simply as $\mathbf{f_1}$, and similarly for g_1, $\mathbf{f_2}$ and g_2.) We get

$$\begin{aligned} \dot{V}_2 &= z_1 \dot{z}_1 + z_2 \dot{z}_2 + \tilde{\theta}^T \Gamma^{-1} \dot{\tilde{\theta}} && (3.31) \\ &= z_1 \left(\theta^T \mathbf{f_1} + (z_2 + x_{2,d}) g_1 - \dot{y}_r \right) + z_2 \left(\theta^T \mathbf{f_2} + g_2 x_3 - \dot{x}_{2,d} \right) - \left(\theta - \hat{\theta} \right)^T \Gamma^{-1} \dot{\hat{\theta}}. && (3.32) \end{aligned}$$

Now we run into a few problems. The first problem comes with the term $\dot{x}_{2,d}$. To find it, we have to use partial derivatives, like

$$\dot{x}_{2,d} = \frac{\partial x_{2,d}}{\partial t} = \frac{\partial x_{2,d}}{\partial x_1} \dot{x}_1 + \frac{\partial x_{2,d}}{\partial y_r} \dot{y}_r + \frac{\partial x_{2,d}}{\partial \dot{y}_r} \ddot{y}_r + \frac{\partial x_{2,d}}{\partial \hat{\theta}} \dot{\hat{\theta}}, \tag{3.33}$$

where we can substitute $\dot{x}_1$ by $\left(\theta^T f_1 + g_1 x_2 \right)$. Using the relations for $x_{2,d}$ and $\dot{x}_{2,d}$, we can further work out the expression for $\dot{V}_2$. Doing this, and reordering terms, gives

$$\begin{aligned} \dot{V}_2 &= -c_1 z_1^2 + g_1 z_1 z_2 - z_1 \hat{\theta}^T \mathbf{f_1} + \theta^T \left(z_1 \mathbf{f_1} + z_2 \left(\mathbf{f_2} - \frac{\partial x_{2,d}}{\partial x_1} \mathbf{f_1} \right) - \Gamma^{-1} \dot{\hat{\theta}} \right) \\ &\quad + z_2 \left(g_2 x_3 - \frac{\partial x_{2,d}}{\partial x_1} g_1 x_2 - \frac{\partial x_{2,d}}{\partial y_r} \dot{y}_r - \frac{\partial x_{2,d}}{\partial \dot{y}_r} \ddot{y}_r - \frac{\partial x_{2,d}}{\partial \hat{\theta}} \dot{\hat{\theta}} \right) + \left(\hat{\theta}^T \Gamma^{-1} \right) \dot{\hat{\theta}}. \end{aligned} \tag{3.34}$$

Yes, that's one big equation. Now we have to get rid of the term with θ. To do this, we should define $\dot{\hat{\theta}}$. But this is again a problem. We have already defined it. As a solution, we just redefine it to be

$$\dot{\hat{\theta}} = \Gamma \left(z_1 \mathbf{f_1} + z_2 \left(\mathbf{f_2} - \frac{\partial x_{2,d}}{\partial x_1} \mathbf{f_1} \right) \right). \tag{3.35}$$

By now you're probably wondering, 'Okay, by redefining it, you know that $\dot{V}_2$ is negative definite. But you aren't sure anymore that $\dot{V}_1$ is negative definite!' To see whether that's true, we examine $\dot{V}_1$. If we insert our new value for $\dot{\hat{\theta}}$ and the value for $x_{2,d}$ into equation (3.26), we wind up with

$$\dot{V}_1 = -c_1 z_1^2 - z_2 \theta^T \left(\mathbf{f_2} - \frac{\partial x_{2,d}}{\partial x_1} \mathbf{f_1} \right). \tag{3.36}$$

Normally, this is not necessarily a negative definite function. However, the whole system with z_2 is asymptotically stable. (Or it will be once we define the $x_{3,d}$.) So $z_2 \to 0$. In other words, in time, the rightmost term vanishes, which in turn implies that the function will become negative definite. So this part of the system will be stable as well.

Now that we have a new update law, we can also define the desired value for x_3. We will use

$$x_{3,d} = \frac{1}{g_2}\left(-c_2 z_2 - g_1 z_1 + \frac{\partial x_{2,d}}{\partial x_1} g_1 x_2 + \frac{\partial x_{2,d}}{\partial y_r}\dot{y}_r + \frac{\partial x_{2,d}}{\partial \dot{y}_r}\ddot{y}_r + \frac{\partial x_{2,d}}{\partial \hat{\theta}}\dot{\hat{\theta}} - \hat{\theta}^T\left(\mathbf{f_2} - \frac{\partial x_{2,d}}{\partial x_1}\mathbf{f_1}\right)\right). \quad (3.37)$$

This turns the Lyapunov derivative into $\dot{V}_2 = -c_1 z_1^2 - c_2 z_2^2$, which is evidently negative definite.

3.5 The adaptive backstepping algorithm

By now we see how adaptive backstepping works. In this paragraph we'll present the general algorithm. To solve the general adaptive backstepping problem, we should follow the following steps.

- At equation i, define the error signal $z_i = x_i - x_{i,d}$. (Note that $x_{1,d} = y_r$.)
- Rewrite the equation using the error signal z_i. (So use $\dot{z}_i = \dot{x}_i - \dot{x}_{i,d}$ and substitute x_i by $z_i + x_{i,d}$.)
- Treat x_{i+1} as if it's the control input. (It is in fact the virtual control.)
- Define the CLF
$$V_i = \frac{1}{2}z_1^2 + \ldots + \frac{1}{2}z_i^2 + \frac{1}{2}\tilde{\theta}^T\Gamma^{-1}\tilde{\theta}. \quad (3.38)$$
- Define the parameter $\omega_\mathbf{i}$ as
$$\omega_\mathbf{i} = \mathbf{f_i} - \sum_{k=1}^{i-1}\frac{x_{i,d}}{x_k}\mathbf{f_k}. \quad (3.39)$$
- Define the **tuning function** $\tau_\mathbf{i}$ recursively as
$$\tau_\mathbf{i} = \tau_\mathbf{i-1} + z_i\omega_\mathbf{i}, \quad (3.40)$$
where $\tau_\mathbf{0} = 0$ and thus $\tau_\mathbf{1} = z_1\omega_\mathbf{1} = z_1\mathbf{f_1}$.
- Define the preliminary parameter update law
$$\dot{\hat{\theta}}_\mathbf{i} = \Gamma\tau_\mathbf{i}. \quad (3.41)$$
(By the way, the final parameter update law for the system will be $\dot{\hat{\theta}} = \dot{\hat{\theta}}_\mathbf{n} = \Gamma\tau_\mathbf{n}$.)
- Define the desired value $x_{i+1,d}$ (or, when $i = n$, define the input u) according to
$$x_{i+1,d} = \frac{1}{g_i}\left(-c_i z_i - g_{i-1}z_{i-1} - \hat{\theta}^T\omega_\mathbf{i} + \sum_{k=1}^{i-1}\left(\frac{\partial x_{i,d}}{\partial x_k}g_k x_{k+1}\right) + \sum_{k=1}^{i}\left(\frac{\partial x_{i,d}}{\partial y_r^{(k-1)}}y_r^{(k)}\right)\right. \quad (3.42)$$
$$\left. + \frac{\partial x_{i,d}}{\partial \hat{\theta}}\dot{\hat{\theta}} + \sum_{k=2}^{i-1}\left(z_k\frac{\partial x_{k,d}}{\partial \hat{\theta}}\Gamma\omega_\mathbf{i}\right)\right). \quad (3.43)$$
The last term (with z_k) might surprise you, as it hasn't appeared in all our equations so far. That is because it first appears in the relation for $x_{4,d}$. (That is, when $i = 3$.) However, we won't derive this relation, so as to still somewhat limit the number of oversized equations in this summary.

In the end, as was already mentioned, the parameter update law will be $\dot{\hat{\theta}} = \Gamma\tau_\mathbf{n}$. The input to the system will be given by the relation $u = x_{n+1,d}$. And, if these are the only two relations you would want, then you don't even have to go through the entire algorithm above. Just find $\omega_\mathbf{i}$ and $\tau_\mathbf{i}$ for all i from 1 to n. When you have done that, it is easy to find $\dot{\hat{\theta}}$ and u.

4 Robust backstepping

The backstepping algorithm that was just discussed only works for systems with constant coefficients. But what do we do if there are time-varying parameters?

If these parameters vary slowly, then we can still use adaptive backstepping. The algorithm will adapt in time. However, if the parameters vary too fast, achieving asymptotic stability can be very hard, if not impossible. The best we may do in this case is make sure that the parameters are bounded. This makes the controller robust.

To illustrate how this can be done, we examine a simple system, like

$$\dot{x} = k(t)\varphi(x) + u. \tag{4.1}$$

Here, $k(t)$ is an unknown time-varying constant and $\varphi(x)$ is a known nonlinear function. To stabilize the system, we will try using the Lyapunov function $V = \frac{1}{2}x^2$. This gives the derivative

$$\dot{V} = x\dot{x} = k(t)x\varphi(x) + xu. \tag{4.2}$$

We want this to become negative definite. But if $k(t)$ becomes big enough, then $\dot{V}$ will be positive. This is impossible to prevent. So ensuring asymptotic stability will not be possible. As some sort of intermediate solution, we can set

$$u = -cx - \kappa x\varphi(x)^2. \tag{4.3}$$

This turns the Lyapunov derivative into

$$\dot{V} = -cx^2 - \kappa x^2\varphi(x)^2 + k(t)x\varphi(x) = -cx^2 - \kappa\left(x\varphi(x) - \frac{k(t)}{2\kappa}\right)^2 + \frac{k(t)^2}{4\kappa}. \tag{4.4}$$

Now examine the three terms in the right-hand part of the above relation. The two left terms are negative definite. However, the term $\frac{k(t)^2}{4\kappa}$ is not. But (when ignoring the middle term) we do know that $\dot{V}$ can only be positive if

$$\frac{k(t)^2}{4\kappa} > cx^2, \qquad \text{or} \qquad |x| \leq \frac{|k(t)|}{2\sqrt{\kappa c}}. \tag{4.5}$$

In any other case, $\dot{V}$ is negative. So the above relation effectively is the bound on x imposed by our control law. In other words, although the system isn't asymptotically stable, at least the state is bounded.