16-642 Fall 2017: Reference Notes for State Space Systems Introduction

George Kantor

9 and 11 October 2017

1 Quick Review

Equations of motion show up in a number of different forms.

For an n link manipulator, we start with Euler Lagrange equations:

$$\boxed{\frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{q}_k} - \frac{\partial \mathcal{L}}{\partial q_k} = \tau_k}$$

for $k = 1, 2, \ldots, n$.

We do a bunch of math, and put them into standard form:

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) = \tau.$$

We define state vector

$$x = \begin{bmatrix} q \\ \dot{q} \end{bmatrix}$$

and input vector $u = \tau$, which allows us to put the equations into state space form:

$$\dot{x} = f(x, u)$$

Now we're going to learn how to approximately solve state space ODEs.

2 Numerical Integration

Finding an explicit solution to ODEs is hard. Fortunately for the most part we can generate approximate (arbitrarily accurate) solutions numerically. That's what this section is all about.

There is a great description of this stuff in (everything you will ever want to know):
Press, W.H., Teukolsky, S.A., Vetterling, W.T, and Flannery, B.P., Numerical Recipes in C: The Art of Scientific Computing, Second Edition, Cambridge University Press, 1992.

2.1 Example Problem

Consider the mass-spring-damper system:

$$m\ddot{y} + \mu\dot{y} + k_s y = u.$$

Define

$$x = \begin{bmatrix} y \\ \dot{y} \end{bmatrix}$$

Then

$$\dot{x} = \begin{bmatrix} x_2 \\ \frac{1}{m}(u - kx_1 - \mu x_2) \end{bmatrix}$$

If we consider a"lossless" mass-spring-damper system (i.e., $\mu = 0$) with $m = 1$ and $k_s = 1$:

$$\ddot{y} + y = 0.$$

We can easily see the exact solution to this ODE is

$$y = \cos(t).$$

2.2 Euler's Method

Euler's method assumes that the derivative is constant between the time t and $t + T$.

$$x(t + T) = x(t) + T\dot{x}(t) + O(T^2).$$

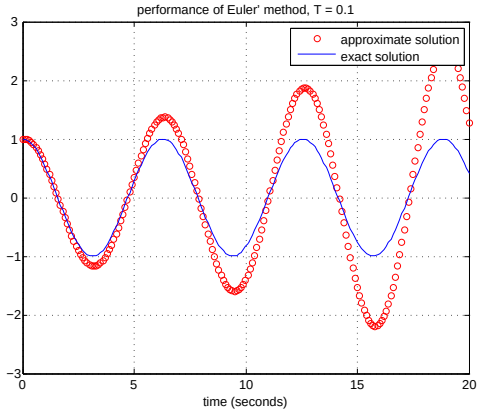
Here the $O(T^2)$ indicates that the approximation

$$x(t + T) \approx x(t) + T\dot{x}(t)$$

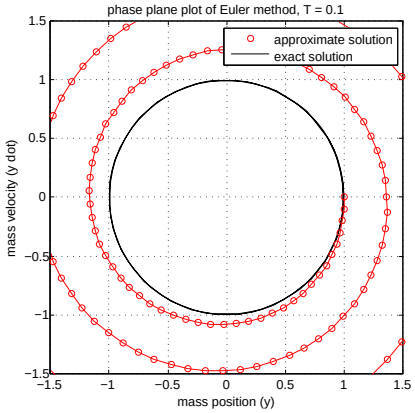
will introduce an error on the order of T^2 at each timestep. Substituting $\dot{x} = f(x, u)$ yields

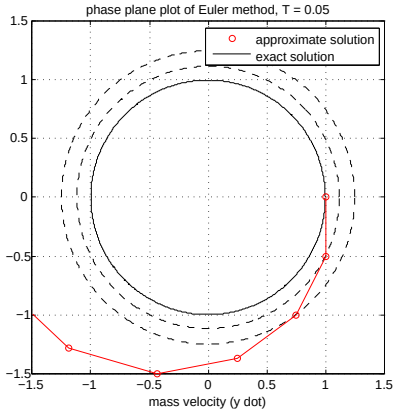
$$x(t + T) \approx x(t) + Tf(x(t), u(t)).$$

The approximation used in Euler's method is similar to the approximation used to convert the ODE to a difference equation, and the result is similar (run `ex2`):



In order to see why this method fails, its easiest to look at the "phase-plane" plot for the system, i.e., the plot of $\dot{y}$ vs y. For a lossless mass-spring system (no damping) this plot should be a circular (or elliptical) trajectory, i.e., it should be periodic. So over each little timestep, the actual path is a curve that follows the arc of the circle. But when we use Euler's method, that arc is approximated by a straight line tangent to the circle at the starting point, causing the approximate solution to go outside the circle. As the integration is integrated, this error is compounded at each step, causing the approximate solution to spiral away from the actual solution.





2.3 The Midpoint Method

The Euler method assumes that the derivative over the entire time step is the same as at the beginning of the time step. The Midpoint method improves this approach by estimating the derivative in the middle of the timestep and using the result in an Euler-like step.

Notation: define $x_k = x(t)$ and $x_{k+1} = x(t+T)$.

The idea:

1. Compute that the Euler step would be. Call the Euler step k_1.

$$k_1 = Tf(x_k, u(t)).$$

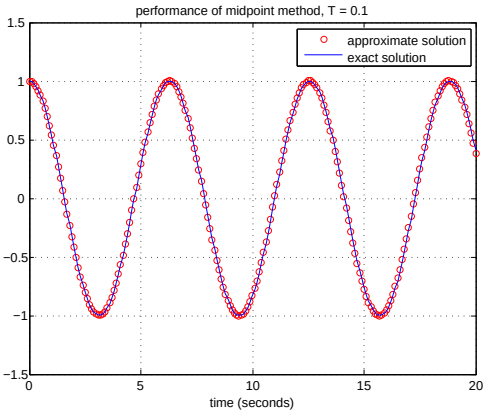
2. compute the step that results when you approximate the derivative at the midpoint between x_k and $x_k + k_1$. This will be a more accurate estimate of the derivative than Euler alone. Call this step k_2.

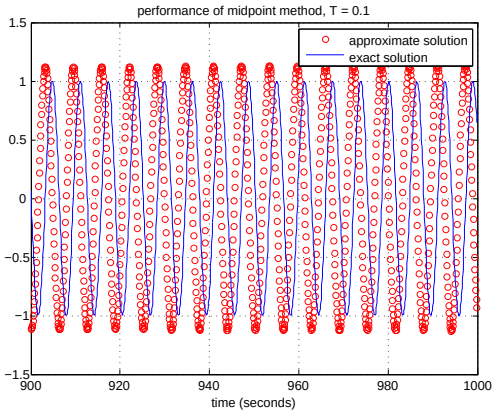
$$k_2 = Tf(x_k + \frac{1}{2}k_1, u(t + 0.5T)).$$

3. use this step to compute x_{k+1}

$$x_{k+1} = x_k + k_2 + O(T^3).$$

The midpoint method is also called *second order Runge-Kutta* because is it accurate to the second order. Here's a figure showing the performance of the midpoint method on the mass-spring example (run `ex3.m` and `ex4.m`):





2.4 Fourth Order Runge-Kutta

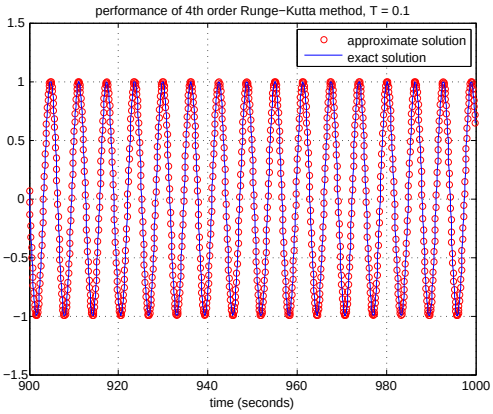
This is the most common because it has the best tradeoff between computational efficiency and accuracy:

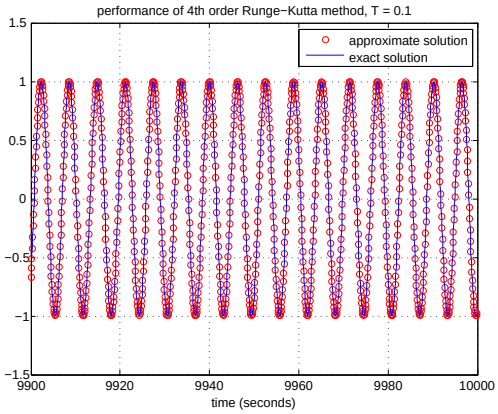
$$k_1 = Tf(x_k, u(t))$$
$$k_2 = Tf(x_k + 0.5k_1, u(t + 0.5T))$$
$$k_3 = Tf(x_k + 0.5k_2, u(t + 0.5T))$$
$$k_4 = Tf(x_k + k_3, u(t + T))$$

Then

$$x_{k+1} = x_k + \frac{k_1}{6} + \frac{k_2}{3} + \frac{k_3}{3} + \frac{k_4}{6} + O(T^5).$$

Here are the plots (run `ex5.m` and `ex6.m`.)





2.5 Other Stuff You Can Do

variable stepsize Runge-Kutta methods.
MATLAB solvers (*e.g.*, `ode45`).

3 Moving to Control: Nonlinear State Space Systems

A continuous nonlinear state space system looks like this:

$$\dot{x}(t) = f(x(t), u(t))$$

$$y(t) = h(x(t))$$

where:

- the state $x \in \mathbb{R}^n$
- the input $u \in \mathbb{R}^m$
- the output $y \in \mathbb{R}^p$

And the continuous functions f and h are such that

$$f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$$

and

$$h : \mathbb{R}^n \to \mathbb{R}^p$$

We think of f and h as a vector of functions, each of which are real valued:

$$f(x, u) = \begin{bmatrix} f_1(x, u) \\ f_2(x, u) \\ \vdots \\ f_n(x, u) \end{bmatrix}$$

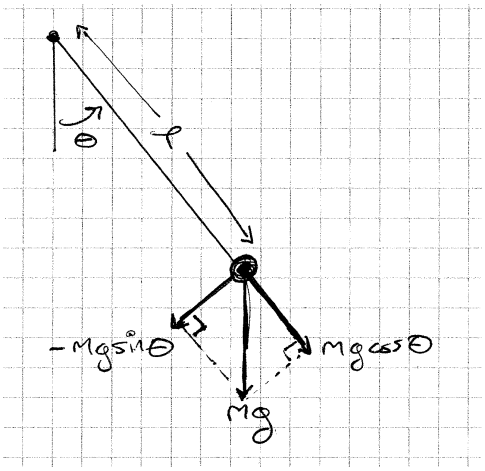
$$h(x) = \begin{bmatrix} h_1(x) \\ h_2(x) \\ \vdots \\ h_p(x) \end{bmatrix}$$

4 Equilibrium Points and Stability

a point x_e is an equilibrium point if $\dot{x}$ is zero at x_e for the unforced system, in other words

$$f(x_e, 0) = 0$$

Consider a pendulum modeled as a point mass at the end of a massless rod with a frictionless pivot:



Let's derive the equations of motion using what we know about physics. The pendulum bob will move on a circle around the pivot point. The velocity tangent to the circle is

$$v_t = \ell\dot{\theta}$$

Newton's law says

$$\begin{aligned} & m\dot{v}_t = F_t \\ \Rightarrow \qquad & m\ell\ddot{\theta} = -mg\sin\theta \\ \Rightarrow \qquad & \ddot{\theta} = -\frac{g}{\ell}\sin\theta \end{aligned}$$

We can put this into state space form as follows. Define

$$x = \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix}$$

Then

$$\dot{x} = \begin{bmatrix} x_2 \\ -\frac{g}{\ell}\sin x_1 \end{bmatrix}.$$

If we extend that slightly to add a control torque $u = \tau$ and viscous friction ($-\gamma\dot{\theta}$) at the pivot, then the system becomes

$$\dot{x} = \begin{bmatrix} x_2 \\ -\frac{g}{\ell}\sin x_1 - \gamma x_2 + u \end{bmatrix}$$

Suppose we had a sensor that could measure position but not velocity, then the output of this system would be given by

$$y(t) = x(t),$$

in other words, $h(x) = x$.

The pendulum has an equilibrium point at

$$x = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

which corresponds to the pendulum hanging straight down.

There is an equilibrium point at

$$x = \begin{bmatrix} \pi \\ 0 \end{bmatrix}$$

which corresponds to the pendulum standing straight up.

There are actually an infinite number of points at

$$x = \begin{bmatrix} n\pi \\ 0 \end{bmatrix}$$

for any n.

stability: From experience, we know that the equilibrium point at the top is unstable and the equilibrium point at the bottom is stable. If I add a little friction the equilibrium point at the bottom becomes asymptotically stable.

Let's make some more formal stability definitions.

First, we need to review the mathematical notion of open sets: A set $D \subset R^n$ is said to be open if every point in D is in the interior of D. More rigorously, D is open if for any $x \in D$ there exists $\epsilon > 0$ so that every point that is within a ball of radius is also in D, i.e., $B_\epsilon(x) \subset D$, where

$$B_\epsilon(x) = \{y \in R^n \mid \|x - y\| < \epsilon\}.$$

The equilibrium point x_e is said to be

- **stable** if for every $\epsilon > 0$ there exists $\delta > 0$ so that if $\|x_0 - x_e\| < \delta$ then $\|x(t) - x_e\| < \epsilon$ for all time $t > 0$. Note that this is inherently a local property.
- **globally stable** if for any $x_0 \in R^n$, there is a bound $K > 0$ such that $\|x(t) - x_e\| < K$ for all $t > 0$.
- **asymptotically stable** if there exists an open D with $x_e \in D$ such that $\|x(t) - x_e\| \to 0$ for any initial condition $x_0 \in D$.
- globally asymptotically stable if the above definition for asymptotic stability holds with $D = R^n$.
- **unstable** if it is not stable, i.e., there exists an $\epsilon > 0$ so that for any $\delta > 0$ there is an initial condition $x_0 \in B_\delta(x_e)$ so that $\|x(t) - x_e\| > \epsilon$ for some $t > 0$.