Math 128A, Summer 2019

Lecture 18, Tuesday 7/23/2019

1 Review, ODEs

$$
\begin{aligned}
y'(t) &= f(t, y(t)) \\
+y(0) &= y_0
\end{aligned}
$$

Adding these gives an initial value problem:

$$
\begin{aligned}
y(t) &= y_0 + \int_0^t f(s, y(s))\ ds \\
&= F[y](t) \\
F[u](t) &= y_0 + \int_0^t f(s, u(s))\ ds
\end{aligned}
$$

2 Fixed Point Iteration (of Functions)

$$
\begin{aligned}
y_{n+1}(t) &= y_0 + \int_0^t f(s, y_n(s))\ ds \\
y_0(t) &= y_0
\end{aligned}
$$

Convergent if : (1) Invariant set of y, (2) contractive. Suppose we are looking for a solution on the interval $0 \le t \le T$.
To See Contractive:

$$
\sup_{0 \le t \le T} |F[u] - F[v]| \le \frac{1}{2} \sup_{0 \le s \le T} |u - v|.
$$

To See Invariance If our function f is nice and bounded; that is, $|f(t, u)| \le M$, then:

$$
|y_{n+1}(t) - y_0| \le Mt \le MT
$$

Invariance is a lot harder to see in general, but luckily it is not as important to check, according to Strain. So when is F contractive?

$$
\begin{aligned}
F[u](t) - F[v](t) &= \int_0^t f(s, u(s))\ ds - y_0 - \int_0^t f(s, v(s))\ ds \\
&= \int_0^t [f(s, u(s)) - f(s, v(s))\ ]\, ds
\end{aligned}
$$

Contractive is to show that if two points in the domain are close, then their values at those points should be even closer. We want:

$$
\left| \int_0^t f(s, u(s)) - f(s, v(s))\ ds \right| \le L|u(s) - v(s)|
$$

Definition: Lipschitz -

We say a functin $f(t, u)$ is **Lipschitz** if and only if there is a constant L such that

$$|f(t, u) - f(t, v)| \leq L|u - v|,$$

for all u, v.

This is stronger than being continuous, but a tiny bit weaker than differentiable.
As an aside, if we have for $0 < \alpha < 1$,

$$|f(t, u) - f(t, v)| \leq L|u - v|^{\alpha},$$

then we call this **Hölder** α-continuity (pronounced 'Hurlder').
We see that this is the restriction we wanted to impose in our previous differential equations.

Theorem 2.1. If $|f(t, u)| \leq M$ and $|f(t, u) - f(t, v)| \leq L|u - v|$, then there is a **unique solution** to

$$y' = f(t, y) \qquad y(0) = y_0$$

on the interval $[0, \tau]$ where τ is a function $\tau(T, R, M, L) > 0$ on those variables, which can be found via dimensional analysis. These are $0 \leq t \leq T$, $u - y_0| \leq R$, and M, L as above.

3 Systems of ODEs and Higher-Order Equations

For example, $F = ma$ means:

$$y'' = \frac{1}{m} F(y, y')$$

If we stick to a single variable $y' = f(t, y)$, we are very much stuck, but if we are willing to go to a vector $y_i' = f_i(t, y_1, \ldots, y_n)$, where $i = 1 : n$, say

$$\begin{aligned} y'' &= \frac{1}{m} F(\underbrace{y}_{u}, \underbrace{y'}_{v}) \\ v' := y'' &= \frac{1}{m} F(u, v) \\ u' := y' &= v, \end{aligned}$$

this gives a system of equations that unstucks us:

$$\begin{bmatrix} u \\ v \end{bmatrix}' = \begin{bmatrix} u' \\ v' \end{bmatrix} = \begin{bmatrix} v \\ \frac{1}{m} F(u, v) \end{bmatrix} = f(Y).$$

We don't need higher order differentiation coefficients or discretization.

Example: Last time we looked at

$$\begin{aligned} y' &= a(t)y + b(t) \\ y'' + \omega^2 y &= g(t), \end{aligned}$$

where we want to write the second equation (second order equation) as a first order problem.
Then we have, for $u := y, v := y'$,

$$\begin{aligned} u' &= v \\ v' &= -\omega^2 y + g(t) \\ &= -\omega^2 u + g(t). \end{aligned}$$

Hence:

$$\begin{bmatrix} u \\ v \end{bmatrix}' = \begin{bmatrix} 0 & 1 \\ -\omega^2 & 0 \end{bmatrix} \begin{bmatrix} u & v \end{bmatrix} + \begin{bmatrix} 0 & g(t) \end{bmatrix}$$

gives $y' = Ay + g(t)$, where

$$\begin{aligned} A &= \begin{bmatrix} 0 & 1 \\ -\omega^2 & 0 \end{bmatrix} \\ A^2 &= \begin{bmatrix} -\omega^2 & 0 \\ 0 & -\omega^2 \end{bmatrix} = -\omega^2 I \end{aligned}$$

And further,

$$\begin{aligned} y' &= Ay + g(t) \\ y' - Ay &= g \\ \left(e^{-tA} y\right)' &= e^{-tA} g \\ e^{-tA} y(t) - e^{-0A} y(0) &= \int_0^t e^{-sA} g(s) \, ds \end{aligned}$$

This is almost a solution, but we don't have $y(t)$ isolated yet. That gives:

$$y(t) = e^{tA}y_0 + \int_0^t e^{(t-s)A}g(s)\ ds$$

This tells us how to deal with constant coefficient differential equations, with a forcing constant ω. We have many ways to express e^{tA}, but we noticed a nice pattern with A^2 earlier, so we take the exponential series (Taylor expansion):
Recall:

$$\begin{aligned} e^{\lambda} &= \sum_{n=0}^{\infty} \frac{\lambda^n}{n!} \\ &= \sum_{n=0}^{\infty} \frac{\lambda^{2n}}{(2n)!} + \sum_{n=0}^{\infty} \frac{\lambda^{2n+1}}{(2n+1)!}, \end{aligned}$$

where the left expression gives $\cosh \lambda$ and the right gives $\sinh \lambda$. Inserting i to get an oscillating term within each summation, we get:

$$\begin{aligned} e^{i\lambda} &= \sum_{n=0}^{\infty} \frac{\lambda^{2n}}{(2n)!} + \sum_{n=0}^{\infty} \frac{\lambda^{2n+1}}{(2n+1)!} \\ &= \cos \lambda + i \sin \lambda. \end{aligned}$$

So for our problem, we write (with ω under sin via dimensional analysis):

$$\begin{aligned} e^{tA} &= \cos(\omega t)I + \sin(\omega t)A \\ \frac{d}{dt}Ae^{tA} &= -\omega \sin(\omega t) + \sin(\omega t)A \\ Ae^{tA} &= \cos(\omega t)A + \sin(\omega t)A^2 \\ &= \cos(\omega t)A - \omega^2 \sin(\omega t)I \\ &= \begin{bmatrix} -\omega^2 \sin(\omega t) & \cos(\omega t) \\ -\omega^2 \cos(\omega t) & -\omega^2 \sin(\omega t) \end{bmatrix} \end{aligned}$$

We don't have an i preceding the sin terms because the eigenvalues of A^2 are already complex, in ω.
So, this gives:

$$\frac{d}{dt}e^{tA} = \underbrace{\omega}\, Ae^{tA}$$

However, we should not have ω here, so to fix this, we realize that our definitions for u, v had inconsistent dimensions with respect to ω. Instead, we should have:

$$u := y; \qquad v := \frac{1}{\omega}y'.$$

Then, we have (fixed):

$$\begin{aligned} e^{tA} &= \cos(\omega t)I + \frac{\sin(\omega t)}{\omega}A \\ \frac{d}{dt}Ae^{tA} &= -\omega \sin(\omega t)I + \cos(\omega t)A \\ Ae^{tA} &= \cos(\omega t)A + \frac{\sin(\omega t)}{\omega}A^2 \end{aligned}$$

4 Euler's Method

Strain says this is like the magic wand, where if we know how to use it, we can solve anything. In the initial value problem, we have some specified data, given at 0. The idea behind the IVPs is that we can forget all the history in the past time and just look at the current state at any given point in time.
Consider the Initial Value Problem:

$$y'(t) = f(t, y(t)) \qquad y(t_n) = y_n$$

The integral equation of this is:

$$\begin{aligned} y(t_{n+1}) &= y(t_n) + \int_{t_n}^{t_{n+1}} y'(s)\ ds \\ &= y(t_n) + \int_{t_n}^{t_{n+1}} f(s, y(s))\ ds \end{aligned}$$

We conclude that we should integrate from t_n to t_{n+1}. We know how to perform numerical integration in many ways, so it's simply a matter of choosing one.
How to approximate the integral? We can take, for example, 5 points, with ECTR: $w_0, \ldots, w_4$:

$$\int_{t_n}^{t_{n+1}} f(s, y(s))\ ds = w_0 f(t_n, y_n) + w_1 f(t_{n+\frac{1}{4}}, y_{n+\frac{1}{4}}) + \cdots + w_4 f(t_{n+1}, y_{n+1})$$

Actually, we can't use ECTR because we don't know things like $y_{n+\frac{1}{4}}$. Instead, we want to find some numerical integration rule that only uses the endpoints (namely only the left endpoint), because we want an explicit formula for moving forward.
This is Euler's method:

$$u_{n+1} = u_n + hf(t_n, u_n), \qquad u_0 = y_0$$

$$\int_{t_n}^{t_{n+1}} f(s, y(s))\ ds = hf(t_{n+1}, y_{n+1})$$

This is called "Backwards Euler", and an implicit equation,

$$u_{n+1} = u_n + hf(t_{n+1}, u_{n+1}).$$

This was thought to be 'backwards' but turns out to be incredibly useful.
Trapezoidal Rule: Implicit. Alternatively, we can use trapezoidal rule even without endpoint correction (with just a simple trapezoid, averaging the two values):

$$\sum_{t_n}^{t_{n+1}} f(s, y(s))\ ds = \frac{h}{2}\left[f(t_n, y_n) + f(t_{n+1}, y_{n+1})\right]$$

Midpoint Estimation: Implicit. Also, alternatively, we can take the midpoint in the domain and estimate the function value.

$$\begin{aligned} \int_{t_n}^{t_{n+1}} f(s, y(s))\ ds &= hf(t_{n+\frac{1}{2}}, y_{n+\frac{1}{2}}) \\ &= hf\left(t_n + \frac{h}{2}, \frac{y_n + y_{n+1}}{2}\right) \end{aligned}$$

Gaussian Integration: (w_j, θ_j) **on** $[-1, 1]$

$$\int_{t_n}^{t_{n+1}} f(s, y(s))\ ds = \frac{h}{2}\sum_{p=0}^{w}\left[w_p f\left(t_n + \frac{h}{2} + \frac{h}{2}\theta_p, \underbrace{y\left(t_n + \frac{h}{2} + \frac{h}{2}\theta_p\right)}\right)\right],$$

where we leave these guys for later, because we don't have a way of computing these yet, until Runge-Kutta methods.
Suppose we decide that actually we can use some integration rule with:

$$\int_{t_n}^{t_{n+1}} f(s, y(s))\ ds = h\left[\alpha f(t_n, y_n) + \beta f(t_{n+1}, y_{n+1}) + \gamma f(t_{n-1}, y_{n-1}) + \cdots + \delta f(t_{n-17}, y_{n-17})\right]$$

This is the idea that when we integrate, we can use past values (and not future). We call this "Multistep"
We know the exact solution:

$$\begin{aligned} y_{n+1} &= y_n + \int_{t_n}^{t_{n+1}} y'(s)\ ds \\ &= y_n + hf(t_n, y_n) - \underbrace{hf(t_n, y_n)\int_{t_n}^{t_{n+1}} y'(s)\ ds}_{h\tau_n} \end{aligned}$$

Euler's method says to do something a bit different, to get an approximation to the integral:

$$u_{n+1} = u_n + hf(t_n, u_n).$$

Ideally, we want to make these equations look alike and subtract them.
Hence we have the **local truncation error**, where τ stands for **t**runcation

$$\tau_n := \frac{y_{n+1} - y_n}{h} - f(t_n, y_n)$$

Notice that we take Euler's method and apply it to the **exact** solution, because it will not exactly satisfy the Euler's method (whereas the meshpoints from our computed solution would not do the trick).

Remark: Another way to express local truncation error is "the residual of exact y in the numerical method".

Notice in the above equation, we have $f(t_n, y_n) = y'_n$, so:

$$\begin{aligned} \tau_n &= \frac{y_{n+1} - y_n}{h} - y'_n \\ &= \frac{y_n + hy'_n + \frac{1}{2}h^2 y''(\xi_n) - y_n}{h} - y'_n, \end{aligned}$$

and

$$\tau_n = \frac{1}{2}hy''(\xi_n) = O(h)$$

so $\tau_n \to 0$ as $h \to 0$. We call this 'consistency', in the sense that our numerical method **converges** to the differential equation. However, the goal is to have the **solution** of the numerical method to converge to the solution of the differential equation.

5 Error Analysis

$$\begin{aligned} & y_{n+1} = y_n + hf(t_n, y_n) + h\tau_n \\ -[\quad & u_{n+1} = u_n + hf(t_n u_n) \quad] \end{aligned}$$

Subtracting these 'remarkably similar-looking' equations gives:

$$e_{n+1} := y_{n+1} - u_{n+1} = [y_n - u_n] + h\left(f(t_n, y_n) - f(t_n, u_n)\right) + h\tau_n$$

We cry a bit because our error does not decrease. But recognize that we march along step by step, so there's no reason for our error to decrease. We bound this via triangle inequality:
Assuming f is Lipschitz, not only do we have the existence of a solution within our relevant timespan, but also we get a bound for the error in our numerical method.

$$|e_{n+1}| \leq |e_n| + hL|e_n|$$

For our solution, we needed f differentiable down to 2 derivatives anyway, where

$$\tau_n = \frac{1}{2}hy''(\xi_n) = O(h),$$

from above.
Hence we have a recurrence relation with **inequality** as opposed to equality.

$$|e_{n+1}| \leq (1 + hL)|e_n| + h|\tau_n|$$

By the compound interest argument, we know the $(1 + hL)$ grows at a bounded rate. The difference between compounding yearly versus daily versus every minute is bounded. As $h \to 0$, we have more steps; however, at each step, the difference in function values is smaller.
Let $\tau := \max_n |\tau_n| = O(h)$. Then, we have:

$$\begin{aligned} |e_n| &\leq (1 + hL)\,|e_{n-1}| + h\tau \\ |e_{n+1}| &\leq (1 + hL)^2\,|e_{n-1}| + \left[(1 + hL)^1 + (1 + hL)^0\right] h\tau \\ &\leq (1 + hL)^{n+1}\,|e_0| + \left[(1 + hL)^n + (1 + hL)^{n-1} + \cdots + (1 + hL)^0\right] \underbrace{h\tau} \end{aligned}$$

We call this bit 'stability' as $h \to 0$ (errors don't explode). Later with consistency (the idea that our method converges to the equation), these two will imply convergence of the numerical solution to the exact solution.
We assume our inital error $|e_0| = 0$, otherwise this error is very expensive. Then the remaining terms is a finite geometric series:

$$\begin{aligned} |e_{n+1}| &\leq \frac{(1 + hL)^{n+1} - 1}{(1 + hL) - 1} h\tau \\ &= \frac{1}{L}\left[(1 + hL)^{n+1} - 1\right]\tau \end{aligned}$$

We estimate $(1 + hL)$ by:

$$1 + hL \leq 1 + hL + \frac{1}{2!}(hL)^2 + \frac{1}{3!}(hL)^3 + \frac{1}{4!}(hL)^4 + \cdots = e^{hL}$$

Then, we conclude:

$$(1 + hL)^n \leq [e^{hL}]^n = e^{nhL} = e^{t_n L},$$

where $t_n = nh$, because n varies from 0 to τ. Notice this is a sloppy (inoptimal bound) but looks elegant. In particular, our error may decrease due to the stiffness of the equation.
The idea between convergence proof is to know that theoretically we have a solution; later we'll worry about if these are practical.
We conclude that we proved a theorem:

Theorem 5.1. If y'' is bounded with $|y''| \leq M$, and f is Lipschitz, then for $0 \leq t_n \leq T$,

$$|u_n - y_n| \leq \frac{e^{t_n L - 1}}{L} \frac{1}{2} hM,$$

where $n \to \infty$ and $h \to 0$.

Even though we are sampling at more and more points, our maximum error converges to zero. We call this "convergence" in the sense that the numerical solution converges to the exact solution in a fixed interval $[0, T]$.

Lecture ends here.

6 Office Hour

Review Gaussian Integration:

$$\int_{-1}^{1} f(t)\ dt = \sum_{j=0}^{p} w_j f(t_j),$$

for $p + 1$ points.
This gives

$$\sum_{j=0}^{p} w_j t_j^k = \int_{-1}^{1} t^k\ dt, \qquad 0 \leq k \leq 2p + 1,$$

where we normally have $0 \leq k \leq p$, but the magic in Gaussian Integration is we have $2p + 1$ here.