CSE 599: Interplay between Convex Optimization and Geometry **Winter 2018**

Lecture 18: How to solve ODE?

Lecturer: Yin Tat Lee

Disclaimer: *Please tell me any mistake you noticed.*

In this lecture, we discuss solving ordinary differential equations (ODE). This lecture is meant for someone with no background on ODE and numerical ODE. So, it might be too basic for some.

Many iterative methods can naturally viewed as an ordinary differential equation. For example, the gradient descent corresponds to the ODE

$$\frac{dx}{dt} = -\nabla f(x),$$

Newton method corresponds to the ODE $\frac{dx}{dt} = -\left(\nabla^2 f(x)\right)^{-1} \nabla f(x)$ and the interior point method corresponds to

$$\begin{aligned} X\frac{ds}{d\ln t} + S\frac{dx}{d\ln t} &= -t \\ A\frac{dx}{dt} &= 0, \\ A^\top \frac{dy}{dt} + \frac{ds}{dt} &= 0. \end{aligned}$$

Therefore, I believe having some basic understand of ODE will be useful for optimization research. Since this lecture is aimed for TCS audience, we will only show how to solve ODE in polynomial time. Note that this is certainly not how ODE is solved in practice.

18.1 Numerical ODE

Consider the explicit ordinary differential equation of order n:

$$x^{(n)}(t) = F(t, x(t), x'(t), x''(t), \cdots, x^{(n-1)}(t)).$$

We can rewrite it into a system of ODE of order 1 with n variables

$$\begin{aligned} x_i'(t) &= x_{i+1}(t) \text{ for } i = 0, \cdots, n-2, \\ x_{n-1}'(t) &= F(t, x_0(t), x_1(t), x_2(t), \cdots, x_{n-1}(t)). \end{aligned}$$

Therefore, from now on, we focus on first order ODE:

$$x'(t) = F(t, x(t))$$

where $x(t) \in \mathbb{R}^n$. Equivalently, you can write it as the integral form:

$$x(t) = x(0) + \int_0^t F(s, x(s))ds.$$

Often, taking integration is more stable than taking derivative. For example, a small change in the function only makes small change into the integration, but it can arbitrarily change the derivative. This is why we mainly work on the integral form.

First, we prove that ODE uniquely exists under Lipschitz assumption.

Theorem 18.1.1 (Picard–Lindelöf theorem)**.** *Suppose that F has Lipschitz constant L on the x variables, namely,*

$$\|F(s,x) - F(s,y)\| \le L\|x-y\|$$

for any norm in $\mathbb{R}^n$. If $TL < 1$, then, for any initial point $x(0)$, there exists an unique solution x^* such that

$$x'(t) = F(t, x(t)) \text{ for all } 0 \le t \le T.$$

Proof. Let $C([0,T], \mathbb{R}^n)$ be the set of continuous function from $[0,T]$ to $\mathbb{R}^n$. Consider the map $\mathcal{T} : C([0,T], \mathbb{R}^n) \to C([0,T], \mathbb{R}^n)$ defined by

$$\mathcal{T}(x)(t) = x(0) + \int_0^t F(s, x(s))ds.$$

Note that the solution of the ODE is exactly the fix point of $\mathcal{T}$. We will prove it by showing that $\mathcal{T}$ has Lipschitz constant less than 1.

Define a norm $\|x\| = \max_{t\in[0,T]} \|x(t)\|$. Now, we compute the Lipschitz constant of $\mathcal{T}$ under this norm:

$$\begin{aligned}
\|\mathcal{T}(x) - \mathcal{T}(y)\| &= \max_{t\in[0,T]} \|\mathcal{T}(x)(t) - \mathcal{T}(y)(t)\| \\
&= \max_{t\in[0,T]} \left\| \int_0^t F(s,x(s))ds - \int_0^t F(s,y(s))ds \right\| \\
&\le \max_{t\in[0,T]} \int_0^t \|F(s,x(s)) - F(s,y(s))\| \, ds \\
&\le \max_{t\in[0,T]} \int_0^t L\|x(s) - y(s)\| \, ds \\
&\le TL\|x-y\|.
\end{aligned}$$

Let x_0 be the constant function given by $x(t) = x(0)$. Note that

$$\left\|T^{\circ n}(x_0) - T^{\circ n-1}(x_0)\right\| \le (TL)^{n-1}\|T(x_0) - x_0\| \to 0$$

as $n \to \infty$ because $TL < 1$. Therefore, the limit $x^* = \lim_{n\to\infty} T^{\circ n}(x_0)$ exists. Clearly, we have $T(x^*) = x^*$ and hence this is the solution of the ODE. Furthermore, if there are two solution x_1^* and x_2^* (namely $\mathcal{T}(x_1^*) = x_1^*$ and $\mathcal{T}(x_2^*) = x_2^*$), then we have that

$$\|x_1^* - x_2^*\| = \|\mathcal{T}(x_1^*) - \mathcal{T}(x_2^*)\| \le TL\|x_1^* - x_2^*\|$$

which shows that $x_1^* = x_2^*$, namely the solution of the ODE is unique. □

Remark 18.1.2. Note that the proof is constructive and we only need the Lipschitz condition around some suitable neighborhood of $x(0)$. Furthermore, the mapping converges on the whole domain $0 \le t < \infty$. See Wikipedia for the proof.

18.2 Numerical Integration

As we see from the existence proof, if we can do integration, then we can solve differential equation. Hence, we will focus on numerical integration. The nice thing about the integration problem is that it is a one dimension question, namely, we simply can integrate

$$(\int F_1(s,x(s))ds, \int F_2(s,x(s))ds, \cdots, \int F_n(s,x(s))ds)$$

separately dimension by dimension. To simplify the notation, let consider this 1 variable problem

$$\int_0^1 f(s)ds.$$

One can approximate it via $\frac{1}{N}\sum_{i=0}^{N-1} f(\frac{i+\frac{1}{2}}{N})$. This is called the midpoint rule. In general, this has $O(\frac{1}{N^2})$ error unless f belongs to some special class of functions we will cover next lecture.

Theorem 18.2.1 (midpoint rule)**.** *Suppose that $|f''(x)| \le M_2$. Then, we have that*

$$\left|\int_0^1 f(s)ds - \frac{1}{N}\sum_{i=0}^{N-1} f(\frac{i+\frac{1}{2}}{N})\right| \le \frac{M_2}{24N^2}.$$

Proof. Let us bound the first interval $[0, \frac{1}{N}]$ first. Note that

$$f(s) = f(\frac{1}{2N}) + f'(\frac{1}{2N})(s-\frac{1}{2N}) \pm \frac{M_2}{2}(s-\frac{1}{2N})^2.$$

Hence, we have

$$\begin{aligned}\int_0^{1/N} f(s)ds &= \frac{1}{N}f(\frac{1}{2N}) \pm \frac{M_2}{2}\int_0^{1/N}(s-\frac{1}{2N})^2 ds \\ &= \frac{1}{N}f(\frac{1}{2N}) \pm \frac{M_2}{24}\frac{1}{N^3}.\end{aligned}$$

Summing it over N intervals gives the result. □

It turns out that if f is smooth, we can do much better by using $\sum w_i f(s_i)$ for some well chosen weights $\{w_i\}_{i=0}^N$ and $\{s_i\}_{i=0}^N$. Note that given $f(s_i)$, there is an unique degree N polynomial p such that $p(s_i) = f(s_i)$ for all $i = 0, 1, \cdots, N$. This polynomial p is given by

$$p(x) = \sum_{i=0}^N f(s_i)\ell_i(x)$$

where $\ell_i(x) = \prod_{0\le j\le N, i\ne j} \frac{x-x_j}{x_i-x_j}$. ($\ell_i$ are called Lagrange polynomials.)

Therefore, one way to choose w_i is using

$$\int_0^1 f(s)ds \sim \int_0^1 p(s)ds = \int_0^1 \sum_{i=0}^N f(s_i)\ell_i(x)dx = \sum_{i=0}^n f(s_i)\underbrace{\int_0^1 \ell_i(x)dx}_{w_i}. \tag{18.1}$$

Note that this way of choosing weights guarantees that it calculates the integral of any degree N polynomial exactly. Therefore, the error term involves the $N+1$-th derivative of f.

Suppose now we have a method with the guarantee

$$\left|\int_0^1 f(s)ds - \sum_{i=0}^N w_i f(s_i)\right| \le C_N M_{N+1} \text{ where } M_{N+1} = \max_{x\in[0,1]}\left|f^{(N+1)}(x)\right|.$$

Them, we can cut the integral into pieces and get the following

$$\left|\int_0^1 f(s)ds - \frac{1}{k}\sum_{j=0}^{k-1}\sum_{i=0}^N w_i f\left(\frac{s_i+j}{k}\right)\right| \le k\cdot \underbrace{C_N\frac{M_{N+1}}{k^{N+2}}}_{\text{error in each piece}} = C_N\frac{M_{N+1}}{k^{N+1}}.$$

In some sense, the above method corresponds to approximate the function by a piecewise polynomial. It is not ideal because our function probably does not look like that. Yet this method has much better convergence rate than the midpoint rule. When we double the number of points, the error decreases by 2^{N+1} instead of 4. This basically gives a polynomial time algorithm to compute integral up to machine accuracy as long as the function is smooth. In next section, we discuss how to choose s_i.

18.3 Numerical Interpolation

From (18.1), we see that it suffices to understand the error between a function and its polynomial approximation. This relies on the following Lemma:

Lemma 18.3.1. *Given any $N+1$ times differentiable function f. Let $P_N(x)$ be a N degree polynomial such that $P_N(s_i) = f(s_i)$ for $i = 0, 1, \cdots, N$. Then, we have that*

$$f(x) - P_N(x) = \frac{f^{(N+1)}(\zeta)}{(N+1)!} \prod_{i=0}^{N} (x - s_i)$$

for some $0 \le \zeta \le 1$.

Proof. Let $\omega(x) = \prod_{i=0}^{N}(x - s_i)$. Consider the function

$$F(t) = f(t) - P_N(t) - \frac{f(x) - P_N(x)}{\omega(x)}\omega(t).$$

Note that s_i and x are the roots of $F(t)$. Since there are $N+2$ roots on $[0,1]$, there is $0 \le \zeta \le 1$ such that

$$0 = F^{(N+1)}(\zeta) = f^{(n+1)}(t) - \frac{f(x) - P_N(x)}{\omega(x)}(N+1)!.$$

Hence, we have the result. □

18.3.1 Chebyshev nodes

Using this and (18.1), we have that

$$\left| \int_0^1 f(s)ds - \sum_{i=0}^{N} w_i f(s_i) \right| \le \frac{M_{N+1}}{(N+1)!} \max_{0 \le x \le 1} \left| \prod_{i=0}^{N} (x - s_i) \right| \text{ where } M_{N+1} = \max_{0 \le \zeta \le 1} \left| f^{(N+1)}(\zeta) \right|.$$

So, the error depends on the quantity $\max_{0 \le x \le 1} \left| \prod_{i=0}^{N}(x - s_i) \right|$. Note that for $x = \frac{1}{2}$, we have $|x - s_i| \le \frac{1}{2}$ and hence no matter how we choose s_i, we would get $\frac{1}{2^{N+1}}$. But for the $x = 0$, we could have $\prod_{i=0}^{N}(x - s_i) = 1$ (if all of s_i is 1, which certainly a really bad choice of s_i). But at least this seems to suggest that we should put more node near both boundary. It turns out this quantity is minimized if we use Chebyshev nodes

$$s_i = \frac{1}{2} + \frac{1}{2}\cos\left(\frac{2k+1}{2N+2}\pi\right)$$

and we have

$$\left| \int_0^1 f(s)ds - \sum w_i f(s_i) \right| \le \frac{M_{N+1}}{(N+1)! 2^{2N+1}}.$$

18.3.2 Stability

In practice, we often has numerical error in computing $f(s_i)$. Suppose we have $|f_i - f(s_i)| \le \varepsilon$. Then, we have that

$$\left|\sum_{i=0}^{N} w_i f(s_i) - \sum_{i=0}^{N} w_i f_i\right| = \left|\sum_{i=0}^{N} w_i f(s_i) - \sum_{i=0}^{N} w_i f_i\right| \le \varepsilon \sum_{i=0}^{N} \left|\int_0^1 \ell_i(x)dx\right| \le \varepsilon \max_{x\in[0,1]} \sum_{i=0}^{N} |\ell_i(x)|$$

where we used that $w_i = \int_0^1 \ell_i(x)dx$. Hence, how stable is the numerical integration depends on the quantity $\max_{x\in[0,1]} \sum_{i=0}^{N} |\ell_i(x)|$ which is called Lebesgue constant Λ_N.

According to Wikipedia, the Λ_N for equidistant nodes has

$$\Lambda_N \sim \frac{2^{N+1}}{eN\log N}.$$

On the other hand, the Λ_N for Chebyshev node has

$$\Lambda_N \le \frac{2}{\pi}\log(N+1) + 1.$$

This is doubly exponentially better! Note that Chebyshev node is not the optimal nodes for the Lebesgue constant.

18.4 Going back to numerical ODE

Recall from the existence proof that if the Lipschitz constant $L < \frac{1}{T}$, then the solution of the ODE exists. More importantly, it can be found by computing integration repeatedly using

$$\mathcal{T}(x)(t) = x(0) + \int_0^t F(s, x(s))ds.$$

Since we cannot compute the integral exactly, we can first approximate $s \to F(s, x(s))$ using Lagrange polynomials, and consider the following algorithm instead

$$\mathcal{T}_N(x)(t) = x(0) + \int_0^t \sum_{i=0}^{N} F(s_i, x(s_i))\ell_i(s)ds.$$

Note that $F(s_i, x(s_i))$ is a vector and $\ell_i(s)$ is a polynomial. Since $\ell_i(s)$ is a polynomial, so is $\int_0^t \ell_i(s)ds$. Therefore, $\mathcal{T}_N$ in fact maps a function to a degree $N+1$ polynomial "vector" $\mathcal{P}_{N+1}$. Alternatively, we can think $\mathcal{T}_N$ maps $\mathcal{P}_{N+1}$ to $\mathcal{P}_{N+1}$. Again, we can assign the norm $\|p\| = \max_{0\le s\le T} \|p(x)\|$ on the space $\mathcal{P}_{N+1}$. To show that there is a fix point on $\mathcal{T}_N$, again we need to estimate the Lipschitz constant of $\mathcal{T}_N$. Note that

$$\begin{aligned}
\|\mathcal{T}_N(x) - \mathcal{T}_N(y)\| &= \left\| \int_0^t \sum_{i=0}^{N} F(s_i, x(s_i))\ell_i(s)ds - \int_0^t \sum_{i=0}^{N} F(s_i, y(s_i))\ell_i(s)ds \right\| \\
&\le T \max_{s\in[0,1]} \sum_{i=0}^{N} |\ell_i(s)| \, \|F(s_i, x(s_i)) - F(s_i, y(s_i))\| \\
&\le T \max_{s\in[0,1]} \sum_{i=0}^{N} |\ell_i(s)| \, L \, \|x - y\| = \Lambda_N LT \, \|x - y\| .
\end{aligned}$$

Recall that originally, we have the Lipschitz constant of $\mathcal{T}$ is LT. But now we have the Lipschitz constant of $\mathcal{T}_N$ is $\Lambda_N LT$ instead. If we use the Chebyshev node, we are only paying a $\log(n)$ factor in the Lipschitz constant! (Note that it would be horrible if we use equidistant point instead.)

Hence, if $T < \frac{1}{T\Lambda_N}$, we find an algorithm to solve ODE with the convergent rate is constant!

Finally, we need to show that the fix point of $\mathcal{T}_N$ is indeed close to the solution of the ODE. Let x_N^* be the fix point of $\mathcal{T}_N$ and x^* be the solution of the ODE. Assume that there is a degree N polynomial p_N that is close to x^*. Note that $T_N(x_N^*) = x_N^*$, $T(x^*) = x^*$ and $T_N(p_N) = T(p_N)$. Hence, we have

$$\begin{aligned}\|x_N^* - x^*\| &= \|T_N(x_N^*) - T(x^*)\| \\ &\leq \|T_N(x_N^*) - T(p_N)\| + \|T(p_N) - T(x^*)\| \\ &\leq \|T_N(x_N^*) - T_N(p_N)\| + LT\|p_N - x^*\| \\ &\leq \Lambda_N LT\|x_N^* - p_N\| + LT\|p_N - x^*\| \\ &\leq \Lambda_N LT\|x_N^* - x^*\| + (\Lambda_N + 1)LT\|p_N - x^*\|.\end{aligned}$$

If $\Lambda_N LT \leq \frac{1}{2}$, then we have that

$$\|x_N^* - x^*\| \leq 2(\Lambda_N + 1)LT\|p_N - x^*\| = O(1)\|p_N - x^*\|.$$

Therefore, as long as there is a polynomial approximating the real ODE solution, then our method is able to find it up to multiplicative error!

Theorem 18.4.1. *Let x^* be the solution of the ODE $\frac{dx}{dt} = F(t, x(t))$ with initial condition $x(0)$ for $0 \leq t \leq T$. Suppose that there is a degree N polynomial p_N such that*

$$\max_{0\leq t\leq T} \|p_N(t) - x^*(t)\| \leq \varepsilon$$

for any norm on $\mathbb{R}^n$. Suppose that F is L Lipschitz, namely $\|F(s,x) - F(s,y)\| \leq L\|x - y\|$ with the same norm. If $LT \leq \frac{1}{1+\log(N+1)}$, we can find a polynomial u such that

$$\max_{0\leq t\leq T} \|u - x^*(t)\| \leq O(\varepsilon)$$

and that the algorithm only involves $\tilde{O}(N)$ many evaluations of F and $\tilde{O}(nN)$ extra time.

In some TCS applications [96], we can approximate the solution of an ODE using $\tilde{O}(1)$ degree polynomial. In this case, this algorithm is nearly linear time!

18.5 Solving ODE in practice

We note that the approach I mentioned only make sense if the function globally can be approximated by $\tilde{O}(1)$ degree polynomial. (Or you can think there is only one "data" hidden in the ODE we need to recover.) In practice, the solution of ODE can be much more complicated and hence it makes more sense to solve ODE locally. The simplest method is the Euler method $x_{t+h} = x_t + h \cdot F(t, x_t)$. One can relate Euler method and the method we proved by cutting the domain into h-size pieces and apply the fix point iteration above in each piece. Solving ODE is a huge subject on its own and here is one basic textbook I like [95].

References

[95] Arieh Iserles. *A first course in the numerical analysis of differential equations*. Number 44. Cambridge university press, 2009.

[96] Yin Tat Lee and Santosh S Vempala. Convergence rate of riemannian hamiltonian monte carlo and faster polytope volume computation. *arXiv preprint arXiv:1710.06261*, 2017.