Math 6620: Analysis of Numerical Methods, II
Solvers for initial value problems, Part I

See Ascher and Petzold 1998, Chapters 1-5

Akil Narayan[1]

[1] **Department of Mathematics, and Scientific Computing and Imaging (SCI) Institute**
University of Utah

Initial value problems

We've discussed how to solve simple stationary problems.

Before delving into time-dependent problems, we'll spend some time discussing numerical methods for solving initial value problems.

Initial value problems

We've discussed how to solve simple stationary problems.

Before delving into time-dependent problems, we'll spend some time discussing numerical methods for solving initial value problems.

Our focal problem will be the ordinary differential equation:

$$\boldsymbol{u}'(t) = \boldsymbol{f}(t, \boldsymbol{u}), \qquad \boldsymbol{u}(0) = \boldsymbol{u}_0,$$

where the initial condition $\boldsymbol{u}_0$ is given.

The assumption is that we seek to compute $\boldsymbol{u}(t)$ for $t \in [0, T]$.
(Or possibly only at the endpoint, or at some discrete values in the interval)

The function $\boldsymbol{f}$ can be nonlinear and/or the ODE can be autonomous, and higher-order ODE's can be written as first-order ODE's with an expanded state vector.

Method of Lines

Initial value problems appear ubiquitously in numerical solutions of PDE's, typically through method of lines (MOL) discretizations.

In MOL discretizations, one discretizes *all but one* variable, and the one remaining variable is typically time:

$$u_t = u_{xx} + u\, u_x \quad \longrightarrow \quad \boldsymbol{u}'(t) = \boldsymbol{D}^2 \boldsymbol{u} + \boldsymbol{u} \circ (\boldsymbol{D}\boldsymbol{u}) =: \boldsymbol{f}(\boldsymbol{u}).$$

Above, $\boldsymbol{D}$ is some matrix discretization of the differentiation operation, and $\circ$ is the Hadamard (elementwise) product.

I.e., first we have discretized in x (physical) space, but left t (time) alone.

Initial value problems appear ubiquitously in numerical solutions of PDE's, typically through method of lines (MOL) discretizations.

In MOL discretizations, one discretizes *all but one* variable, and the one remaining variable is typically time:

$$u_t = u_{xx} + u\, u_x \quad \longrightarrow \quad \boldsymbol{u}'(t) = \boldsymbol{D}^2 \boldsymbol{u} + \boldsymbol{u} \circ (\boldsymbol{D}\boldsymbol{u}) =: \boldsymbol{f}(\boldsymbol{u}).$$

Above, $\boldsymbol{D}$ is some matrix discretization of the differentiation operation, and $\circ$ is the Hadamard (elementwise) product.

I.e., first we have discretized in x (physical) space, but left t (time) alone.

MOL discretizations are not the only way to develop PDE solvers, but they appear so frequently that understanding how to solve the resulting IVP's is quite important.

In these slides we'll focus exclusively on numerically solving the IVP, assuming $\boldsymbol{f}$ is given.

Well-posedness for IVP's is a quite well-studied topic.

Theorem (Picard-Lindelöf)

Suppose that $\boldsymbol{f} : \mathbb{R} \times \mathbb{R}^M \to \mathbb{R}^M$ *is continuous in some open ball around* $(0, \boldsymbol{u}_0) \in \mathbb{R} \times \mathbb{R}^M$*, and further is Lipschitz continuous in the* $\boldsymbol{u}$ *variable in this ball. Consider the initial value problem,*

$$\boldsymbol{u}'(t) = \boldsymbol{f}(t; \boldsymbol{u}), \qquad \boldsymbol{u}(0) = \boldsymbol{u}_0.$$

Then there exists some $\epsilon > 0$ *such that there is a unique solution* $\boldsymbol{u}(t)$ *to the above problem for* $t \in [-\epsilon, \epsilon]$.

Well-posed IVP's

Well-posedness for IVP's is a quite well-studied topic.

Theorem (Picard-Lindelöf)

Suppose that $\boldsymbol{f} : \mathbb{R} \times \mathbb{R}^M \to \mathbb{R}^M$ *is continuous in some open ball around* $(0, \boldsymbol{u}_0) \in \mathbb{R} \times \mathbb{R}^M$*, and further is Lipschitz continuous in the* $\boldsymbol{u}$ *variable in this ball. Consider the initial value problem,*

$$\boldsymbol{u}'(t) = \boldsymbol{f}(t; \boldsymbol{u}), \qquad \boldsymbol{u}(0) = \boldsymbol{u}_0.$$

Then there exists some $\epsilon > 0$ *such that there is a unique solution* $\boldsymbol{u}(t)$ *to the above problem for* $t \in [-\epsilon, \epsilon]$.

In the proof of this theorem, one constructs $\boldsymbol{u}$ such that,

$$\boldsymbol{u}(t) = \boldsymbol{u}(0) + \int_0^t \boldsymbol{f}(t; \boldsymbol{u}(t)) \mathrm{d}t.$$

This formula is the starting point for many numerical schemes.

Temporal discretization

$$\boldsymbol{u}'(t) = \boldsymbol{f}(t; \boldsymbol{u}), \qquad \boldsymbol{u}(0) = \boldsymbol{u}_0.$$

An easy strategy is to discretize time with equispaced values up to a terminal time T:

$$t_0 := 0, \qquad t_n := nk = n\Delta t, \qquad T = Nk.$$

I.e., we either choose $\Delta t = k$, or N.

$$\boldsymbol{u}'(t) = \boldsymbol{f}(t;\boldsymbol{u}), \qquad \boldsymbol{u}(0) = \boldsymbol{u}_0.$$

An easy strategy is to discretize time with equispaced values up to a terminal time T:

$$t_0 := 0, \qquad t_n := nk = n\Delta t, \qquad T = Nk.$$

I.e., we either choose $\Delta t = k$, or N.

The exact solution satisfies

$$\boldsymbol{u}(t_{n+1}) = \boldsymbol{u}(t_n) + \int_{t_n}^{t_{n+1}} \boldsymbol{f}(t, \boldsymbol{u}(t))\mathrm{d}t,$$

and therefore we seek to compute numerical approximations $u_n \approx u(t_n)$ that would satisfy,

$$\boldsymbol{u}_{n+1} \approx \boldsymbol{u}_n + \int_{t_n}^{t_{n+1}} \boldsymbol{f}(t, \boldsymbol{u}(t))\mathrm{d}t$$

Some basic schemes

$$\begin{aligned} \boldsymbol{u}'(t) &= \boldsymbol{f}(t; \boldsymbol{u}), & \boldsymbol{u}(0) = \boldsymbol{u}_0. \\ \boldsymbol{u}_n &\approx \boldsymbol{u}(t_n) \\ \boldsymbol{u}_{n+1} &\approx \boldsymbol{u}_n + \int_{t_n}^{t_{n+1}} \boldsymbol{f}(t, \boldsymbol{u}(t)) \mathrm{d}t \end{aligned}$$

A great many schemes result from discretization of the integral via quadrature.

$$\boldsymbol{u}'(t) = \boldsymbol{f}(t;\boldsymbol{u}), \qquad \boldsymbol{u}(0) = \boldsymbol{u}_0.$$

$$\boldsymbol{u}_n \approx \boldsymbol{u}(t_n)$$

$$\boldsymbol{u}_{n+1} \approx \boldsymbol{u}_n + \int_{t_n}^{t_{n+1}} \boldsymbol{f}(t,\boldsymbol{u}(t))\mathrm{d}t$$

A great many schemes result from discretization of the integral via quadrature.

The Forward Euler method uses the quadrature rule,

$$\int_{t_n}^{t_{n+1}} \boldsymbol{f}(t,\boldsymbol{u}(t))\mathrm{d}t \approx (t_{n+1}-t_n)\boldsymbol{f}(t,\boldsymbol{u}(t))\big|_{t=t_n} \approx k\boldsymbol{f}(t_n,\boldsymbol{u}_n) =: k\boldsymbol{f}_n,$$

leading to the scheme,

$$\boldsymbol{u}_{n+1} = \boldsymbol{u}_n + k\boldsymbol{f}_n.$$

This scheme is explicit.

$$\boldsymbol{u}'(t) = \boldsymbol{f}(t; \boldsymbol{u}), \qquad \boldsymbol{u}(0) = \boldsymbol{u}_0.$$

$$\boldsymbol{u}_n \approx \boldsymbol{u}(t_n)$$

$$\boldsymbol{u}_{n+1} \approx \boldsymbol{u}_n + \int_{t_n}^{t_{n+1}} \boldsymbol{f}(t, \boldsymbol{u}(t))\mathrm{d}t$$

A great many schemes result from discretization of the integral via quadrature.

The Backward Euler method uses the quadrature rule,

$$\int_{t_n}^{t_{n+1}} \boldsymbol{f}(t, \boldsymbol{u}(t))\mathrm{d}t \approx (t_{n+1} - t_n)\boldsymbol{f}(t, \boldsymbol{u}(t))\big|_{t=t_{n+1}} \approx k\boldsymbol{f}(t_{n+1}, \boldsymbol{u}_{n+1}) =: k\boldsymbol{f}_{n+1},$$

leading to the scheme,

$$\boldsymbol{u}_{n+1} = \boldsymbol{u}_n + k\boldsymbol{f}_{n+1}.$$

This scheme is implicit.

$$\boldsymbol{u}'(t) = \boldsymbol{f}(t;\boldsymbol{u}), \qquad \boldsymbol{u}(0) = \boldsymbol{u}_0.$$

$$\boldsymbol{u}_n \approx \boldsymbol{u}(t_n)$$

$$\boldsymbol{u}_{n+1} \approx \boldsymbol{u}_n + \int_{t_n}^{t_{n+1}} \boldsymbol{f}(t,\boldsymbol{u}(t))\mathrm{d}t$$

A great many schemes result from discretization of the integral via quadrature.

The Crank-Nicolson method uses the Trapezoid rule approximation,

$$\int_{t_n}^{t_{n+1}} \boldsymbol{f}(t,\boldsymbol{u}(t))\mathrm{d}t \approx \frac{(t_{n+1}-t_n)}{2}\left(\boldsymbol{f}(t,\boldsymbol{u}(t))\big|_{t=t_n} + \boldsymbol{f}(t,\boldsymbol{u}(t))\big|_{t=t_{n+1}}\right)$$

$$= \frac{k}{2}(\boldsymbol{f}_n + \boldsymbol{f}_{n+1}).$$

leading to the scheme,

$$\boldsymbol{u}_{n+1} = \boldsymbol{u}_n + \frac{k}{2}\left(\boldsymbol{f}_n + \boldsymbol{f}_{n+1}\right).$$

This scheme is also implict.

We might reasonbly define convergence as the following:

Definition

A scheme for an ODE is convergent to order p if

$$\max_{n \in [N]} \|\boldsymbol{e}_n\| = \mathcal{O}(k^p), \qquad \boldsymbol{e}_n := \boldsymbol{u}(t_n) - \boldsymbol{u}_n.$$

for some choice of norm $\|\cdot\|$ on M-dimensional vectors.

We might reasonably define convergence as the following:

Definition

A scheme for an ODE is convergent to order p if

$$\max_{n \in [N]} \|\boldsymbol{e}_n\| = \mathcal{O}(k^p), \qquad \boldsymbol{e}_n := \boldsymbol{u}(t_n) - \boldsymbol{u}_n.$$

for some choice of norm $\|\cdot\|$ on M-dimensional vectors.

As you might expect, we cannot tackle convergence directly without some setup involving consistency and stability.

Local truncation error

For ODE IVP's, our local truncation error is again defined as the residual of the scheme when the *exact* solution is inserted, and is consistent if $k \downarrow 0$ causes the LTE to vanish.

To fix details, let's consider forward Euler:

$$\boldsymbol{u}_{n+1} = \boldsymbol{u}_n + k\boldsymbol{f}_n.$$

First, we rewrite the scheme to match the units of the original ODE:

$$\frac{\boldsymbol{u}_{n+1} - \boldsymbol{u}_n}{k} = \boldsymbol{f}_n \quad \longrightarrow \quad D^+\boldsymbol{u}_n - \boldsymbol{f}_n = \boldsymbol{0},$$

where we have introduced the difference operator $D^+\boldsymbol{u}_n := 1/k\,(\boldsymbol{u}_{n+1} - \boldsymbol{u}_n)$.
Note the *superscript* for $+$.

For ODE IVP's, our local truncation error is again defined as the residual of the scheme when the *exact* solution is inserted, and is consistent if $k \downarrow 0$ causes the LTE to vanish.

To fix details, let's consider forward Euler:

$$\boldsymbol{u}_{n+1} = \boldsymbol{u}_n + k\boldsymbol{f}_n.$$

First, we rewrite the scheme to match the units of the original ODE:

$$\frac{\boldsymbol{u}_{n+1} - \boldsymbol{u}_n}{k} = \boldsymbol{f}_n \quad \longrightarrow \quad D^+\boldsymbol{u}_n - \boldsymbol{f}_n = \boldsymbol{0},$$

where we have introduced the difference operator $D^+\boldsymbol{u}_n := 1/k\,(\boldsymbol{u}_{n+1} - \boldsymbol{u}_n)$.
Note the *superscript* for $+$.

After a short computation, we conclude:

$$\mathrm{LTE}_n := \left\| D^+\boldsymbol{u}(t_n) - \boldsymbol{f}(t_n, \boldsymbol{u}(t_n)) \right\| \simeq k\|\boldsymbol{u}''(t_n)\| = Ck,$$

i.e., our scheme is consistent to first order.
This *suggests* what convergence we should expect.

Stability

There are several notions of stability for numerical methods for IVP's.

Our previous definition of stability concerned the map from scheme inputs to outputs.
Here, that is the map from initial data to, say, $\boldsymbol{u}_n$ for any/all $n \in [N]$.

Stability

There are several notions of stability for numerical methods for IVP's.

Our previous definition of stability concerned the map from scheme inputs to outputs.
Here, that is the map from initial data to, say, $\boldsymbol{u}_n$ for any/all $n \in [N]$.

Unfortunately, this map is *much* more complicated than our previous example. (FD methods for $u'' = f$)

To see why, note that

$$\frac{\boldsymbol{u}_{n+1} - \boldsymbol{u}_n}{k} = \boldsymbol{f}_n = \boldsymbol{f}\left(t_n, \boldsymbol{u}_n\right) \neq \boldsymbol{f}\left(t_n, \boldsymbol{u}(t_n)\right),$$

i.e., at every step, we are actually solving a *perturbed* ODE system that accumulates errors from previous steps.

It turns out that one way to successfully control these perturbations is to assume that the scheme residual operating on the exact solution does not grow out of control.

For Forward Euler, this is:

$$R_n \boldsymbol{u}(t_n) := D^+ \boldsymbol{u}(t_n) - \boldsymbol{f}(t_n, u(t_n)).$$

Definition

A numerical scheme is 0-stable if there is some constant C such that for all h sufficiently small,

$$\max_{n \in [N]} \|\boldsymbol{e}_n\| \leqslant C \left(\|\boldsymbol{e}_0\| + \max_{n \in [N]} \|R_n \boldsymbol{u}(t_n)\| \right).$$

(Above, R_n is the residual of the corresponding time integration scheme.)

$$\max_{n \in [N]} \|\boldsymbol{e}_n\| \leqslant C \left(\|\boldsymbol{e}_0\| + \max_{n \in [N]} \|R_n \boldsymbol{u}(t_n)\| \right).$$

It's not too difficult to see how 0-stability and consistency yield convergence:

- In most practical cases, $\boldsymbol{e}_0 = \mathbf{0}$.
- We have $\|R_n \boldsymbol{u}(t_n)\| = \mathrm{LTE}_n$.

$$\max_{n \in [N]} \|\boldsymbol{e}_n\| \leqslant C \left(\|\boldsymbol{e}_0\| + \max_{n \in [N]} \|R_n \boldsymbol{u}(t_n)\| \right).$$

It's not too difficult to see how 0-stability and consistency yield convergence:

- In most practical cases, $\boldsymbol{e}_0 = \boldsymbol{0}$.
- We have $\|R_n \boldsymbol{u}(t_n)\| = \mathrm{LTE}_n$.

I.e.,

$$\text{Consistency} + 0\text{-stability} \implies \text{Convergence}$$

It remains to establish that schemes of interest are consistent and 0-stable.

0-stability for Forward Euler, I

A technical but somewhat straightforward computation shows that Forward Euler is 0-stable. With,

$$\omega := \max_{n \in [N]} \|R_n \boldsymbol{u}(t_n)\|,$$

$R_n = \frac{u(t_n) - u(t_{n-1})}{k} - k f(t_{n-1}, u(t_{n-1}))$

$u_n = u_{n-1} + k f_{n-1}$

then

$$\begin{aligned} \boldsymbol{e}_n = \boldsymbol{u}(t_n) - \boldsymbol{u}_n &= \boldsymbol{u}(t_n) - \boldsymbol{u}(t_{n-1}) \quad + \quad \boldsymbol{u}(t_{n-1}) - \boldsymbol{u}_{n-1} \quad + \quad \boldsymbol{u}_{n-1} - \boldsymbol{u}_n \\ &= \boldsymbol{e}_{n-1} + kR_n \boldsymbol{u}(t_n) + k\boldsymbol{f}(t_{n-1}, \boldsymbol{u}(t_{n-1})) - k\boldsymbol{f}_{n-1} \\ &= \boldsymbol{e}_{n-1} + k\left[\boldsymbol{f}(t_{n-1}, \boldsymbol{u}(t_{n-1})) - \boldsymbol{f}(t_{n-1}, \boldsymbol{u}_{n-1})\right] + kR_n \boldsymbol{u}(t_n) \end{aligned}$$

0-stability for Forward Euler, I

A technical but somewhat straightforward computation shows that Forward Euler is 0-stable.
With,

$$\omega := \max_{n\in[N]} \|R_n \boldsymbol{u}(t_n)\|,$$

then

$$\begin{aligned}\boldsymbol{e}_n = \boldsymbol{u}(t_n) - \boldsymbol{u}_n &= \boldsymbol{u}(t_n) - \boldsymbol{u}(t_{n-1}) \quad + \quad \boldsymbol{u}(t_{n-1}) - \boldsymbol{u}_{n-1} \quad + \quad \boldsymbol{u}_{n-1} - \boldsymbol{u}_n \\ &= \boldsymbol{e}_{n-1} + kR_n\boldsymbol{u}(t_n) + k\boldsymbol{f}(t_{n-1}, \boldsymbol{u}(t_{n-1})) - k\boldsymbol{f}_{n-1} \\ &= \boldsymbol{e}_{n-1} + k\left[\boldsymbol{f}(t_{n-1}, \boldsymbol{u}(t_{n-1})) - \boldsymbol{f}(t_{n-1}, \boldsymbol{u}_{n-1})\right] + kR_n\boldsymbol{u}(t_n)\end{aligned}$$

Therefore, we have,

$$\begin{aligned}\|\boldsymbol{e}_n\| &\leqslant \|\boldsymbol{e}_{n-1}\| + k\|\boldsymbol{f}(t_{n-1}, \boldsymbol{u}(t_{n-1})) - \boldsymbol{f}(t_{n-1}, \boldsymbol{u}_{n-1})\| + k\omega \\ &\leqslant \|\boldsymbol{e}_{n-1}\| + kL\|\boldsymbol{u}(t_{n-1}) - \boldsymbol{u}_{n-1}\| + k\omega \\ &= \|\boldsymbol{e}_{n-1}\|\,(1 + kL) + k\omega,\end{aligned}$$

where the last inequality uses (assumed!) Lipschitz continuity of $\boldsymbol{f}$:

$$\|\boldsymbol{f}(t, \boldsymbol{x}) - \boldsymbol{f}(t, \boldsymbol{y})\| \leqslant L\|\boldsymbol{x} - \boldsymbol{y}\|.$$

0-stability for Forward Euler, II

$$\omega = \max_{n \in [N]} \|R_n \boldsymbol{u}(t_n)\|, \qquad \|\boldsymbol{e}_n\| \leqslant \|\boldsymbol{e}_{n-1}\| (1 + kL) + k\omega,$$

Iterating on the inequality, we have,

$$\begin{aligned}
\|\boldsymbol{e}_n\| &\leqslant \|\boldsymbol{e}_{n-1}\| (1 + kL) + k\omega \\
&\leqslant (1 + kL)^2 \|\boldsymbol{e}_{n-2}\| + k\omega \left(1 + (1 + kL)\right) \\
&\vdots \\
&\leqslant (1 + kL)^n \|\boldsymbol{e}_0\| + k\omega \sum_{j=0}^{n-1} (1 + kL)^j \\
&\overset{\|\boldsymbol{e}_0\| = 0}{=} k\omega \sum_{j=0}^{n-1} (1 + kL)^j \\
&\overset{(1+a/N)^N \leqslant \exp a}{\leqslant} k\omega \sum_{j=0}^{n-1} e^{L t_j} \\
&\overset{x \mapsto e^x \text{ is convex}}{\leqslant} \omega \int_0^T e^{Lt} \mathrm{d}t \leqslant \frac{e^{LT}}{L} \omega = C \max_{n \in [N]} \|R_n \boldsymbol{u}(t_n)\|.
\end{aligned}$$

$$kL = \frac{k_j L}{j} = \frac{t_j L}{j}$$

Forward Euler convergence

Hence, we have that Forward Euler is convergent:

$$\mathrm{LTE}_n = \mathcal{O}(k) \quad \overset{\text{0-stability}}{\Longrightarrow} \quad \max_{n \in [N]} \|\boldsymbol{e}_n\| = \mathcal{O}(k).$$

Hence, we have that Forward Euler is convergent:

$$\mathrm{LTE}_n = \mathcal{O}(k) \quad \stackrel{\text{0-stability}}{\Longrightarrow} \quad \max_{n \in [N]} \|\boldsymbol{e}_n\| = \mathcal{O}(k).$$

This proves strict first-order convergence, but typically the hidden constants are *very* large, e.g., behave like e^{LT}.

Note that the statements above are asymptotic in k as $k \downarrow 0$. More refined (and in some sense useful) analysis can be achieved if we consider finite, nonzero k.

Ascher, Uri M. and Linda R. Petzold (1998). *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. SIAM. ISBN: 978-1-61197-139-2.