Other Krylov subspace methods

Idea: Find "best" updates inside Krylov subspaces for $A$ (or sometimes $AT$). Some methods work even with
unsymmetric problems. Here are some examples:

**GMRES** (Generalized minimal residual)

- Works for general systems, residual guaranteed
to decrease but iterations become more costly
  (in computations, storage)

**Restarted GMRES**: Run $k$ steps of GMRES, throw
away "memory" and start $k$ steps of GMRES again
from where we left over.

**CGN**: Conjugate Gradient for normal eq. specially designed
to solve $ATAx = ATb$.

**BCG**: Bi-Conjugate gradient (for general problems)
(no optimality and unstable)

**Bi-CGSTAB**: Stabilized version

**QMR**: quasi-minimal residual (another stabilization
of BCG)

etc...

- Krylov subspace methods are also used for finding
eigenvalues (we shall discuss this later)

- Matlab has a good collection of methods for solving
linear systems:

  help sparfum
Chap 5  Initial value problems for Ordinary Diff Eq (ODE)

Objective: develop methods to solve numerically problems of the kind

\[
\begin{align*}
\frac{dy}{dt} &= f(t, y), & \text{for } t \in [a, b] \quad \text{often an exact solution to (1)} \\text{ is too complicated or even cannot be found.}
\end{align*}
\]

\[y(a) = \alpha\]

Extension to systems:

\[
\begin{align*}
\frac{dy_1}{dt} &= f_1(t, y_1, y_2, \ldots, y_n) \\
\frac{dy_2}{dt} &= f_2(t, y_1, y_2, \ldots, y_n) \\
\vdots \\
\frac{dy_n}{dt} &= f_n(t, y_1, y_2, \ldots, y_n)
\end{align*}
\]

\[\begin{align*}
\begin{cases}
\frac{dy}{dt} = f(t, y) \\
\text{and to } m\text{-th order IVP:}
\end{cases}
\end{align*}
\]

\[y^{(n)}(a) = \alpha_1, \quad y^{(n-1)}(a) = \alpha_2, \quad \ldots, \quad y^{(1)}(a) = \alpha_n, \quad y(a) = \alpha_0
\]

Fundamental questions to answer about the IVP (1):

- \text{existence: does (1) admit a sol?}
- \text{uniqueness: is the sol to (1) unique?}
- \text{"stability": Do small changes in the statement of the problem introduce small changes in the solution?}

A problem satisfying all 3 properties is said to be \textbf{well posed} (in the sense of Hadamard)
The IVP (1) is well posed under mild assumptions on $f$. We need some terminology (maybe you've seen this many times before!)

**Def (Lipchitz condition)**

A function $f(t, y)$ satisfies a Lipchitz condition in the variable $y$ on a set $D \subseteq \mathbb{R}^2$ if

$$|f(t, y_1) - f(t, y_2)| \leq L |y_1 - y_2| \quad \forall (t, y_1) \in D, \quad (t, y_2) \in D,$$

$L = $ Lipchitz constant for $f$.

(means $f(t, \cdot)$ is Lipchitz continuous for all $t$).

**Example:**

$$D = \{(t, y) : 0 < t < 1, -1 < y < 1\}$$

$$f(t, y) = ty.$$ 

$$|f(t, y) - f(b, y_2)| = |t(y) - t(y_2)| = |t| |y_1 - y_2| \leq |t| |y_1 - y_2| \leq |y_1 - y_2|$$

$L = 1$

**Def (Convex set)**

A set $D$ is a convex set iff:

$$\lambda(x, y) \in D \Rightarrow \lambda x + (1-\lambda)y \in D \quad \forall \lambda \in [0, 1]$$

In our case:

$$(t, y_1) \in D, (t, y_2) \in D \Rightarrow (\lambda t, +(1-\lambda)ty_1 + (1-\lambda)ty_2)$$

$\forall \lambda \in [0, 1].$

**For IVP** we usually have:

$$D = \{f(t, y) | a \leq t \leq b, \ y \in \mathbb{R} \}.$$
Here is a sufficient (but not necessary) condition for Lipechity at $t$.

**Theorem.** Let $f(t,y)$ be defined on some convex set $D \subset \mathbb{R}^2$.

If $\exists L > 0$ s.t.

$$\left| \frac{\partial f}{\partial y}(t,y) \right| \leq L \quad \forall (t,y) \in D$$

then $f$ satisfies a Lipschitz condition on $D$ in $y$ with Lips. cont. $L$.

Note: $f(t,y) = t |y|$ satisfies the Lipschitz condition and yet $\frac{\partial f}{\partial y}$ does not exist at $y=0$.

Existence and uniqueness for IVP (1) are taken care of by:

**Theorem.** Let $D = \{(t,y) \mid a \leq t \leq b \land y \in \mathbb{R}\}$ and that

1. $f(t,y)$ is continuous on $D$
2. $f$ satisfies Lips. cont$\to$ on $D$ in Variable $y$, then the IVP (1) admits a unique solution.

"Stability": can be formulated as follows:

Let $y(t)$ solve

$$\begin{cases}
\frac{dy}{dt} = f(t,y), \quad a \leq t \leq b \\
y(a) = \alpha
\end{cases}$$

If $\epsilon_0 > 0, \delta_0 > 0 \\forall \\exists t \\delta > \epsilon > 0$

1. $\delta(t)$ continuous s.t. $|\delta(t)| \leq \epsilon$
2. $\delta_0 \in \mathbb{R}$ s.t. $|\delta_0| \leq \epsilon$

$$\begin{cases}
\frac{d\delta}{dt} = f(t,y) + \delta(t), \quad a \leq t \leq b, \\
y(a) = \alpha + \delta_0
\end{cases}$$

(perturbed problem)
has a unique solution \( y(t) \) and:
\[
|y(t) - y(t)| < k \epsilon
\]

(essentially the mapping data \( (\alpha, f(t, y)) \) to solution \( y \) is continuous).

This "stability" property is crucial to trust solutions given by a numerical method (sources of error can be from the method itself or from numerical roundoff).

\( \Rightarrow \) All numerical methods assume IVPs well-posed.

**Example:** \( D = \{(y, \xi) \mid t \in [0, 2], \xi \in \mathbb{R}\} \)

\[
\begin{cases}
\frac{dy}{dt} = (y - t^2 + 1), & 0 \leq t \leq 2 \\
y(0) = \frac{1}{2}
\end{cases} \quad (IVP)
\]

\[
|\frac{\partial f}{\partial y}| = |1| = 1 \Rightarrow f(t, y) \text{ is Lipschitz on } D \text{ with variable } y,
\]

\& if constant \( \Rightarrow \) problem is stable to perturbations in init data.

We can verify this directly:

\[
\begin{cases}
\frac{dy}{dt} = 3 - t^2 + 1 + \delta \\
y(0) = \frac{1}{2} + \delta_0
\end{cases} \quad \text{(perturbed problem)}
\]

IVP has sol: \( y(t) = \frac{1}{2} e^t + (t + 1)^2 \quad \text{and } y(t) = \frac{(1 + \delta_0) e^t + (t + 1)^2 + (e^t - 1) \delta}{2} \)

\[
|y(t) - \hat{y}(t)| = |(\delta_0 + \delta) e^t - \delta| \leq |\delta_0 + \delta| e^t + |\delta| \leq 2e^2 \epsilon + \epsilon \leq (2e^2 + 1) \epsilon
\]
§5.2 Euler’s method

A simple numerical method to solve IVP:

\[ \begin{align*}
\frac{dy}{dt} &= f(t, y), \quad a \leq t \leq b \quad \text{assuming well-posedness} \\
y(a) &= y_0
\end{align*} \]

Euler’s method gives approximations at mesh points:

\[ t_j = a + jh, \quad \text{where} \quad h = \frac{b-a}{N}, \quad j = 0, 1, \ldots, N \]

\[ t_0 = a, \quad t_1, \ldots, t_{N-1}, \quad t_N = b \]

\( (N+1) \) equally spaced points

Idea for Euler’s method: Taylor’s theorem:

\[ y(t_{i+1}) = y(t_i) + y'(t_i) (t_{i+1} - t_i) + \frac{y''(\xi_i)}{2} (t_{i+1} - t_i)^2 \]

for some \( \xi_i \in (t_i, t_{i+1}) \).

\[ = y(t_i) + h y'(t_i) + \frac{h^2}{2} y''(\xi_i) \]

\[ = y(t_i) + h f(t_i, y(t_i)) + \frac{h^2}{2} y''(\xi_i) \]

\( y \) satisfies DE: IVP

Neglect for

Euler’s method

\[ y_0 = y(a), \quad \text{for} \quad i = 0, \ldots, N-1 \]

\[ y_{i+1} = y_i + h f(t_i, y_i) \]

Geometrical interpretation

Assume \( y_i = y(t_i) \)

\[ y(t_i) = y_i \]

\[ y'(t_i) = f(t_i, y_i) \]

Next approx.

Systematic error is introduced at every step.
Several types of errors in numerical methods for DE:

- **Local truncation error**: error made in one step.
  - For example in Euler's method: \( O(h^2) \) since:
    \[
    y(t + h) = y(t) + hf(t, y(t)) + O(h^2)
    \]
  - Backdef: this is \( \Delta \)

- **Local roundoff error**: precision used for computation. \( 10^{-16} \) double \( 10^{-8} \) single

- **Global truncation error**: accumulation of all the local truncation errors. If local truncation error is \( O(h^n) \) then the global truncation error must be \( O(h^n) \) because the number of steps is \( O(h) \).

- **Global roundoff error**: accumulation of local roundoff errors of previous steps.

- **Total error** = global truncation error + global roundoff error

  If the global truncation error is \( O(h^n) \) then the method is of order \( n \)

  **Example**: Euler's method is of order 1, and it is relatively simple to derive more precise bounds on the global truncation error.

  If \( f \) is continuous and satisfy the Lipschitz condition with \( L \)-constant \( L \) on \( \mathbb{D} = \{ (t, y) \mid t \in [a, b] \}, y \in \mathbb{R} \} \) and that

  \[ |y''(t)| \leq M \quad \forall t \in [a, b] \]

  for some \( M > 0 \).

  Let \( y(t) \) be the solution to IVP

  \[
  \begin{align*}
  \frac{dy}{dt} &= f(t, y) \\
  y(a) &= \alpha
  \end{align*}
  \]

  and \( y_0, y_1, \ldots, y_n \) the approximations given by Euler's method.
then:
\[ |y(t_i) - y(t_i+1)| < \frac{M}{2L} \left[ e^{L(t_i-a)} - 1 \right], \quad i = 0, 1, \ldots, N. \]

**Proof (sketch)**

\[ y(t_{i+1}) = y(t_i) + hf(t_i, y(t_i)) + \frac{h^2}{2} y''(\xi_i) \]
\[ y_{i+1} = y_i + hf(t_i, y_i) \]

\[ \Rightarrow |y(t_{i+1}) - y_{i+1}| \leq |y(t_i) - y_i| + h |f(t_i, y(t_i)) - f(t_i, y_i)| \]
\[ + \frac{h^2}{2} |y''(\xi_i)| \]
\[ \leq (1+hl) |y(t_i) - y_i| + \frac{h^2 M}{2} \]

Using Lemma 5.8 in book: (geometric series bounding exp)

\[ |y_{i+1} - w_{i+1}| \leq e^{(i+1)hL} \left( \frac{|y_0 - w_0| + \frac{h^2 M}{2hL}}{2} \right) - \frac{h^2 M}{2hL} \]
\[ \leq \frac{h^2 M}{2hL} \left( e^{(i+1)hL} - 1 \right) \]
\[ t_{i+1} - t_i = t_{i+1} - a_i \]

**Lemma 5.8**

\( \{ a_i \}_{i=0}^{k} \) is a seq with \( a_0 \geq -\frac{b}{4} \) and

\( a_{i+1} \leq a_i (1+b) + t \) for \( i = 0, \ldots, k-1 \)

\[ a_{i+1} \leq e^{(i+1)s} (a_0 + \frac{b}{4}) - \frac{b}{4} \]

**Proof:**

\[ a_{i+1} \leq a_i (1+b) + t \leq ((1+b)a_{i-1} + t) (1+b) + t \]
\[ \leq \left( (1+b)(a_{i-1} + t) + t \right) (1+b) + t \]
\[ \vdots \]
\[ \leq (1+b)^{i+1} a_0 + t \sum_{j=0}^{i} (1+b)^j \]
\[ a_{i+1} \leq (1+a) a_{i} + t \left[ \frac{1 - (1+a)^{i+1}}{1 - (1+a)} \right] \]

\[ = (1+a)^{i+1} (a_{0} + \frac{b_{0}}{a}) - \frac{b_{0}}{a} \]

\[ \leq e \left( a_{0} + \frac{b_{0}}{a} \right) - \frac{b_{0}}{a} \]

since \( (1+x)^{n} = e \alpha \ln (1+x) \alpha x \)

\[ \leq e \]

can be shown using Taylor's

Thus.

For Euler's method it's even possible to incorporate the round-off
errors in the analysis.

Instead of having:

\[ y_{0} = x \]

\[ f_{i} = 0 \ldots N-1 \]

\[ y_{i+1} = y_{i} + h f(t_{i}, y_{i}) \]

we cannot a make take at each step

\[ y_{0} = x + \delta_{0} \]

\[ f_{i} = 0 \ldots N-1 \]

\[ y_{i+1} = y_{i} + h f(t_{i}, y_{i}) + \delta_{i+1} \]

If \( |\delta_{i}| < \delta \) it is possible to show:

\[ |y(t_{i}) - y_{i}| \leq \frac{1}{2} \left( \frac{hM}{2} + \frac{\delta}{h} \right) \left[ e^{L(t_{i} - a)} - 1 \right] \]

\[ + |\delta_{i}| e^{L(t_{i} - a)} \]

performance degrades as \( h \to 0 \)!!

(similar to problem with numerical differentiation)

The \( h \) giving smallest error is

\[ h = \sqrt{\frac{2 \delta}{M}} \] (simple calculation)
High order Taylor methods:

Some derivation as Euler method but any Taylor series further:

\[
y(t_{i+1}) = y(t_i) + h \cdot y'(t_i) + \frac{h^2}{2} y''(t_i) + \cdots + \frac{h^n}{n!} y^{(n)}(t_i) + \frac{h^{n+1}}{(n+1)!} y^{(n+1)}(\xi)
\]

for some \( \xi \in (t_i, t_{i+1}) \).

\[
y'(t) = f(t, y(t))
\]
\[
y''(t) = \frac{d}{dt} \left[ f(t, y(t)) \right]
\]
\[
y^{(k)}(t) = \frac{d^{k-1}}{dt^{k-1}} \left[ f(t, y(t)) \right]
\]

Taylor method of order \( n \):

\[
y_0 = y(0), \quad n = 1, 2, \ldots, N-1,
\]
\[
y_{i+1} = y_i + h \cdot f(t_i, y(t)) + \frac{h^2}{2} \cdot \frac{d}{dt} f(t_i, y(t)) + \cdots + \frac{h^n}{n!} \cdot \frac{d^{n-1}}{dt^{n-1}} f(t_i, y(t))
\]

Local truncation error is \( O(h^{n+1}) \).

In general evaluating \( \frac{d^k}{dt^k} [f(t, y(t))] \) can be quite involved because of the repeated application of the chain rule.

For example:

\[
\begin{align*}
y' &= \cos t - \sin y + t^2 = f(t, y) \\
y(-1) &= 3
\end{align*}
\]

\[
y'' = \frac{d}{dt} \left[ f(t, y(t)) \right] = -\sin t - y' \cos y + 2t
\]

\[
y''' = \frac{d^2}{dt^2} \left[ f(t, y(t)) \right] = -\cos t - y'' \cos y + (y')^2 \sin y + 2
\]

\[
y^{(4)} = \frac{d^3}{dt^3} \left[ f(t, y(t)) \right] = \sin t - y^{(3)} \cos y + 3y' y'' \sin y + (y')^3 \cos y
\]

etc...
§5.4 Runge-Kutta methods

Taylor series method for
\[
\begin{align*}
\frac{dy}{dt} &= f(t, y), \quad t \in [a, b] \\
y(a) &= y_0
\end{align*}
\]
require us to compute formulas for
\[
\begin{align*}
y'' &= \frac{df}{dt} \\
y''' &= \frac{d^2f}{dt^2} \\
\end{align*}
\]

which can be quite involved.

Runge-Kutta methods avoid this difficulty by carefully chosen combinations of values of \( f(t, y) \).

Second order Runge-Kutta methods:

Start with Taylor series:
\[
y(t+h) = y(t) + \frac{h}{2} y'(t) + \frac{h^2}{2} y''(t) + \frac{h^3}{6} y'''(t) + \ldots.
\]

From the DE we get:
\[
\begin{align*}
y'(t) &= f(t, y) = \frac{\partial f}{\partial t} \\
y''(t) &= \frac{\partial f}{\partial t} + \frac{\partial^2 f}{\partial t \partial y} y' = f_t + f_{ty} \\
y'''(t) &= \frac{\partial^2 f}{\partial t^2} + \frac{\partial^2 f}{\partial t \partial y} y'' + \frac{\partial^3 f}{\partial t^3} + \frac{\partial^2 f}{\partial t \partial y^2} y' + \frac{\partial^3 f}{\partial t^2 \partial y} y + \frac{\partial^3 f}{\partial t \partial y^2} y + \frac{\partial^3 f}{\partial y^3} y + O(h^3)
\end{align*}
\]

\[
y(t+h) = y(t) + \frac{hf}{2} + \frac{h^2}{2} (f_t + f_{ty}) + O(h^3)
\]
The idea is to eliminate the partial drift of \( f \) by using the first few terms of the two variable Taylor series for \( f(t, y) \):

\[
f(t+h, y+hf) = f + \frac{\partial f}{\partial x}(t, y) h + O(h^2)
= f + hf_t + hff_y + O(h^2)
\]

Rewriting the Taylor series of \( y \):

\[
y(t+h) = y(t) + \frac{1}{2} hf + \frac{1}{6} h^2 \left[ f(t+h, y+hf) + O(h^3) \right]
= y(t) + \frac{1}{2} hf + \frac{1}{6} h \left[ f(t+h, y+hf) + O(h^3) \right]
+ O(h^3)
\]

Thus it is possible to construct an update which has the same \( O(h^3) \) local truncation error as 2nd order Taylor method:

**Modified Euler method**

\[
y_i = A
\]

for \( i = 0, 1, \ldots, N-1 \)

\[
F_1 = hf(t_i, y_i)
F_2 = hf(t_i+h, y_i+F_1)
\]

\[
y_{i+1} = y_i + \frac{1}{2} F_1 + \frac{1}{2} F_2
\]

The general form for second order Runge-Kutta update is:

\[
y(t+h) = y + w_1 hf + w_2 h \left[ f(t+\alpha h, y+\beta hf) \right]
\]

where \( w_1, w_2, \alpha, \beta \) are parameters that we can adjust. Using two variable Taylor expansion:

\[
y(t+h) = y + w_1 hf + w_2 h \left[ f + h f_t + \beta hff_y \right]
\]
Matching comparable terms with (4) we get:

\[
\begin{align*}
\omega_1 + \omega_2 &= 1 \\
\omega_2 \alpha &= \frac{1}{2} \\
\omega \beta &= \frac{1}{2}
\end{align*}
\]

→ so there is a family of RK order 2 methods

**Modified Euler:** \( \omega_1 = \omega_2 = \frac{1}{2}, \alpha = \beta = 1 \)

**Midpoint method:** \( \omega_1 = 0, \omega_2 = 1, \alpha = \beta = \frac{1}{2} \)

\( y_0 = A \)

for \( i = 0, 1, \ldots, N-1 \)

\[
\begin{align*}
F_1 &= h f(t_i, y_i) \\
F_2 &= h f(t_i + \frac{h}{2}, y_i + \frac{1}{2} F_1) \\
y_{i+1} &= y_i + F_2
\end{align*}
\]

(also \( O(h^3) \) LTE)

**Heun's method:** \( \omega_1 = \frac{1}{4}, \omega_2 = \frac{3}{4}, \alpha = \beta = \frac{2}{3} \)

\( y_0 = A \)

for \( i = 0, 1, \ldots, N-1 \)

\[
\begin{align*}
F_1 &= h f(t_i, y_i) \\
F_2 &= h f(t_i + \frac{h}{3}, y_i + \frac{2}{3} F_1) \\
y_{i+1} &= y_i + \frac{1}{4} F_1 + \frac{3}{4} F_2
\end{align*}
\]
Runge Kutta methods of order 3 are obtained by matching terms between Taylor expansion of order 3 of \( y(t) \) and 
\[ f(t + h, y + hf(t, y)) \]

The derivation is quite tedious, but here is one possible RK order 3 method:

\[ y_0 = A \]
for \( i = 0, \ldots, N-1 \)
\[
F_1 = h f(t_i, y_i) \\
F_2 = h f(t_i + \frac{1}{2} h, y_i + \frac{1}{2} F_1) \\
F_3 = h f(t_i + \frac{3}{4} h, y_i + \frac{3}{4} F_2) \\
y_{i+1} = y_i + \frac{1}{3} (2F_1 + 3F_2 + 4F_3)
\]

However, it is not commonly used in practice.

The most popular RK method is that of order 4; again, the derivation is tedious but the implementation straightforward:

Runge Kutta method of order 4 (LTE O(h^5))

\[ y_0 = A \]
for \( i = 0, \ldots, N-1 \)
\[
F_1 = h f(t_i, y_i) \\
F_2 = h f(t_i + \frac{1}{2} h, y_i + \frac{1}{2} F_1) \\
F_3 = h f(t_i + \frac{3}{4} h, y_i + \frac{3}{4} F_2) \\
F_4 = h f(t_i + h, y_i + F_3) \\
y_{i+1} = y_i + \frac{1}{6} (F_1 + 2F_2 + 2F_3 + F_4)
\]
So # of function eval increases more rapidly than the max order of RK method. => higher order RK method are less attractive than the classical RK4. (it makes more sense to use smaller time steps for RK4 than using a higher order method with bigger steps).

§5.5 Adaptive Runge-Kutta Fehlberg method

Idea: Estimate local truncation error and adjust the step length accordingly.

Local truncation error estimation
Assume we are given two update formulas for solving

\[
\begin{cases}
\frac{y'}{y(t)} = f(t, y) , \text{te} [a, b] \\
y(a) = \alpha
\end{cases}
\]
with local truncation errors differing by \( \pm \epsilon \).

1. \( \tilde{y}(t_{i+1}) = y(t_i) + h \tilde{\phi}(t_i, y(t_i), \tilde{y}(t_i)) + O(h^{n+1}) \) (order \( n \))

\( y_0 = \alpha \)
for \( i = 0, \ldots, N-1 \)

\( \tilde{y}_{i+1} = \tilde{y}_i + h \tilde{\phi}(t_i, y(t_i), \tilde{y}(t_i)) \)

2. \( y(t_{i+1}) = y(t_i) + h \phi(t_i, y(t_i), R) + O(h^{n+1}) \) (order \( n+1 \))

\( \tilde{y}_0 = \alpha \)
for \( i = 0, \ldots, N-1 \)

\( \tilde{y}_{i+1} = \tilde{y}_i + h \phi(t_i, y(t_i), R) \)