Topics: * Implementation of implicit schemes

* Accuracy and consistency

Implementation of Implicit Schemes

Recall our system of ODE's:

$$\frac{d\vec{u}}{dt} = \vec{f}(\vec{u}, t) \qquad \vec{u}(0) = \vec{u}_0$$

An implicit numerical integration is one which requires the calculation of $\vec{f}(\vec{v}^{n+1}, t^{n+1})$ in order to calculate $\vec{v}^{n+1}$. For example, our Backward Euler scheme:

$$\frac{\vec{v}^{n+1} - \vec{v}^{n}}{\Delta t} = \vec{f}(\vec{v}^{n+1}, t^{n+1}) \qquad (1)$$

We use the term "implicit" because to find $\vec{v}^{n+1}$, we must solve the implicit relationship between $\vec{v}^{n+1}$ and $\vec{f}(\vec{v}^{n+1}, t^{n+1})$: $\Rightarrow \boxed{\vec{v}^{n+1} - \Delta t \vec{f}(\vec{v}^{n+1}, t^{n+1}) = \vec{v}^{n}}$

By comparison, an explicit scheme allows a direct solution of $\vec{v}^{n+1}$ without involving $\vec{f}(\vec{v}^{n+1}, t^{n+1})$.

For example, forward Euler:

$$\frac{\vec{v}^{n+1} - \vec{v}^n}{\Delta t} = \vec{f}(\vec{v}^n, t^n)$$

$$\Rightarrow \boxed{\vec{v}^{n+1} = \vec{v}^n + \Delta t\, \vec{f}(\vec{v}^n, t^n)}$$

So, implicit schemes will involve more work per iteration.

Comparison of Implicit & Explicit Schemes for Linear Systems

Suppose our canonical problem were linear:

$$\frac{d\vec{v}}{dt} = A\vec{v} + \vec{f}(t)$$

Forward Euler (FE):

$$\vec{v}^{n+1} = \vec{v}^n + \Delta t \left[A\vec{v}^n + \vec{f}(t^n) \right]$$

Thus, assuming $\vec{f}(t)$ is not too difficult to evaluate, the main cost of FE is the matrix-vector multiply $A\vec{v}^n$. If $\vec{v}$ is an M dimensional vector, this requires:

Worst case FE work : M^2 multiplications & additions

But, in general A is quite sparse.

By sparsity, we mean that the number of non-zeros in A is small compared to the number of zeros:

Define: $M_Z \equiv$ number of zero entries in A

$M_{NZ} \equiv$ number of non-zero entries in A

$(M^2 = M_Z + M_{NZ}!)$

A sparse matrix has $\frac{M_{NZ}}{M_Z + M_{NZ}} << 1$.

* For matrices A arising from finite difference, finite volume or finite element discretization of a PDE, the number of non-zeroes is usually a small number per row of the matrix:

$\Rightarrow M_{NZ} = c_{NZ} M$

where $c_{NZ} \equiv$ average number of non-zeros per row of A. Often $c_{NZ} << M$.

e.g. 2-D finite element for Laplace's equation would have $c_{NZ} \approx 5$

$\Rightarrow$ The work to multiply & add the non-zero entries of A to a vector is:

$c_{NZ} M << M^2$!

Now let's look at our Backward's Euler (BE) scheme:

$$\frac{\vec{v}^{n+1} - \vec{v}^{n}}{\Delta t} = A\vec{v}^{n+1} + \vec{f}(t^{n+1})$$

$$\vec{v}^{n+1} - \Delta t A\vec{v}^{n+1} = \vec{v}^{n} + \Delta t\vec{f}(t^{n+1})$$

$$\Rightarrow \underbrace{(I - \Delta t A)}_{M \times M \text{ matrix}} \vec{v}^{n+1} = \underbrace{\vec{v}^{n} + \Delta t\vec{f}(t^{n+1})}_{\text{RHS}}$$

unknowns

Thus, to find $\vec{v}^{n+1}$, we must solve an $M \times M$ linear system every iteration! This can get quite expensive! Some points:

* In the general case, Gaussian elimination (with partial pivoting) will usually work, but this requires M^3 operations!

* Matrices arising from finite difference, element, or volume methods can often be solved more efficiently
  - structural deformations often $O(M)$
  - heat diffusion often $O(M)$
  - high Reynolds number air flows typically $O(M^2)$ but some $O(M)$ cases

* For small Δt, $I - \Delta t A \approx I$ which is obviously easy to solve. But, we want large Δt's (usually) if we are looking at implicit methods.

Question: suppose a worst case scenario

such that: * Av requires M^2 operations

* Solving $(I-\Delta t A)v^{n+1}=RHS$ requires M^3 operations.

* Assume accuracy requires

$$\Delta t \leq \frac{\tau_{long}}{N_{long}}$$

where $\tau_{long} \equiv$ longest timescale of interest in problem

$N_{long} \equiv$ # of integration steps required for desired accuracy of numerical integration over τ_{long}

Under what conditions is the implicit scheme more efficient to integrate from $0 \to T$?

The next complication is to consider nonlinear $\vec{f}(\vec{v},t)$ with implicit methods.

BE:
$$\frac{\vec{v}^{\,n+1}-\vec{v}^{\,n}}{\Delta t} = \vec{f}(\vec{v}^{\,n+1}, t^{n+1})$$

We re-write this in a residual form:

$$\vec{R}(\vec{v}) = \vec{v} \quad - \vec{v}^{\,n} - \Delta t\,\vec{f}(\vec{v} \quad , t^{n+1})$$

and we are interested in finding $\vec{v}$ such that $\vec{R}(\vec{v}) = 0 \Rightarrow \vec{v} = \vec{v}^{\,n+1}$.

This is finding the root of a nonlinear set of coupled equations. There are several options, but we will only consider the Newton–Raphson method applied to this problem:

To find $\vec{v}^{\,n+1}$:

① Initialize $\vec{w}^{\,0} = \vec{v}^{\,n}$ ← use previous iteration as starting guess for $\vec{v}^{\,n+1}$

Initialize $m = 0$

sub-iteration

Loop → ② Linearize about $\vec{w}^{\,m}$

$$\vec{R}(\vec{w}^{\,m+1}) = \vec{R}(\vec{w}^{\,m} + \Delta\vec{w}) = 0$$

$$\simeq \vec{R}(\vec{w}^{\,m}) + \left.\frac{\partial\vec{R}}{\partial\vec{w}}\right|_{\vec{w}^{\,m}} \Delta\vec{w} = 0$$

$$\Rightarrow \boxed{\left.\frac{\partial\vec{R}}{\partial\vec{w}}\right|_{\vec{w}^{\,m}} \Delta\vec{w} = -\vec{R}(\vec{w}^{\,m})}$$

M×M linear syst of eqns to solve

For our BE, $\frac{\partial \vec{R}}{\partial \vec{w}}$ is:

$$\frac{\partial \vec{R}}{\partial \vec{w}} = \frac{\partial}{\partial \vec{w}}\left\{\vec{w} - \vec{v}^n - \Delta t \vec{f}(\vec{w}, t^{n+1})\right\}$$

$$\frac{\partial \vec{R}}{\partial \vec{w}} = I - \Delta t \frac{\partial \vec{f}}{\partial \vec{w}}(\vec{w}, t^{n+1})$$

$$\Rightarrow \boxed{\left.\frac{\partial \vec{R}}{\partial \vec{w}}\right|_{\vec{w}^m} = I - \Delta t \frac{\partial \vec{f}}{\partial \vec{w}}(\vec{w}^m, t^{n+1})}$$

(3) Solve linear system for $\Delta \vec{w}$ and update:

$$\vec{w}^{m+1} = \vec{w}^m + \Delta \vec{w}$$

(4) Check for convergence:

(*) Calculate $\vec{R}(\vec{w}^{m+1})$. If $\frac{\|\vec{R}(\vec{w}^{m+1})\|}{\|\vec{w}^{m+1}\|} \leq \varepsilon_{tol}$ then converged

$$\Rightarrow \boxed{\vec{v}^{n+1} = \vec{w}^{m+1} \quad \text{DONE}}$$

Note: ε_{tol} should be small, say $10^{-5} \to 10^{-}$

The norm $\|\vec{x}\|$ of $\vec{x}$ can be done several ways:

* $\|\vec{x}\|_2 = \sqrt{\sum_{i=1}^{M} x_i^2}$ perhaps most common (L_2 n

* $\|\vec{x}\|_1 = \sum_{i=1}^{M} |x_i|$ (L_1 norm)

* $\|\vec{x}\|_\infty = \max_{i=1 \to M} |x_i|$ (L_∞ norm)

(*) If not converged sufficiently, $m \to m+1$ and return to (2) starting next sub-iteratio

Comments on Newton sub-iterations:

* In many cases, only a few sub-iterations are required as $\vec{v}^n$ is often a good starting guess for $\vec{v}^{n+1}$.

* To be careful, you probably want to put a limit on the total number of sub-iterations

* For some really tough problems, you may also want to limit the Newton sub-iteration update:

$$\vec{w}^{m+1} = \vec{w}^m + \eta \Delta\vec{w}$$

where $0 < \eta \leq 1$. $\eta = 1$ is full update

η is re-calculated at every iteration usually by comparing the magnitude of some of the $\Delta\vec{w}$ entries to $\vec{w}^m$. For example, suppose:

$$\vec{w} = \begin{bmatrix} x \\ y \\ u \\ v \end{bmatrix}$$

and you had a characteristic length L and velocity V_∞

Then, you might require

$$\Delta\vec{w} = \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta u \\ \Delta v \end{bmatrix}$$

for example

$\eta \Delta x \leq 0.1 L$

or

$\eta \Delta u \leq 0.01 V_\infty$

For work estimates:

$$\text{FE:} \quad \frac{T}{\Delta t_{FE}} M^2$$

$$\text{BE:} \quad \frac{T}{\Delta t_{BE}} M^3 = \frac{T}{\tau_{long}} N_{long} M^3$$

But Δt_{BE} is bound by $\frac{\tau_{long}}{N_{long}}$ for accuracy

$$\text{Work BE} < \text{Work FE}$$

$$\frac{T}{\tau_{long}} N_{long} M^3 < \frac{T}{\Delta t_{FE}} M^2$$

$$\frac{\Delta t_{FE}}{\tau_{long}} N_{long} M < 1$$

But $\Delta t_{FE} = \tau_{short}$ due to instability

$$\Rightarrow \boxed{\frac{\tau_{short}}{\tau_{long}} N_{long} M < 1}$$

need stiff problems (this is 1/condition number)

higher accurate numerical scheme will decrease N_{long}

small # of states

Note: the work ratio per iteration gives rise to the M:

$$\frac{\text{Work per iter BE}}{\text{Work per iter FE}} = \frac{M^3}{M^2} = M$$

So, if $\dfrac{\text{Work/iter BE}}{\text{Work/iter FE}} = C$

(say if both were $O(M)$) then clearly the implicit scheme is a winner for stiff problems:

$$\frac{\text{Work BE}}{\text{Work FE}} = \frac{\tau_{short}}{\tau_{long}} N_{long}\, C$$

Underlying Message: Good linear solvers (for $Ax=b$ type problems) are critical for numerical efficiency!