Notes

- Final exam: December 10, 10am-1pm
  X736 (CS Boardroom)
- Another extra class this Friday 1-2pm

Other implicit methods

- Implicit mid-point

$$y_{n+1} = y_n + \Delta t\, f\left(\tfrac{1}{2} y_n + \tfrac{1}{2} y_{n+1}, t_{n+\frac{1}{2}}\right)$$

- Trapezoidal rule

$$y_{n+1} = y_n + \Delta t\left[\tfrac{1}{2} f\left(y_n, t_n\right) + \tfrac{1}{2} f\left(y_{n+1}, t_{n+1}\right)\right]$$

- A-stable, but only conditionally monotone
  - Trapezoidal rule: 1/2 step of FE, 1/2 step of BE
  - Implicit mid-point: very closely related
- Aliasing on imaginary axis

Even more

- Implicit multistep methods:

  Adams(-Bashforth)-Moulton

  Backwards Differentiation Formula (BDF)
- Implicit Runge-Kutta
  - Might need to solve for multiple intermediate values simultaneously…

Solving Nonlinear Equations

- First in 1D: g(x)=0
  - Bisection
  - Secant method
  - Newton's method
- General case: g and x both n-dimensional
  - Newton is the standard
- More can go wrong (than in optimization)
  - E.g. Jacobian can be unsymmetric
- Similar robustifying tricks apply
  - Modifying the Jacobian, line search, …
- Convergence is simpler to identify!

Newton applied to BE

- Initial guess:
  - At least use previous y
  - Can even use an explicit method to predict y
    - For a single step, stability might not be a problem
- Iteration:

$$y^{(k)} + \Delta y = y_n + \Delta t\left(f\left(y^{(k)}, t_{n+1}\right) + \frac{\partial f\left(y^{(k)}, t_{n+1}\right)}{\partial y}\Delta y\right)$$

$$\left(I - \Delta t \frac{\partial f}{\partial y}\right)\Delta y = y_n + \Delta t\, f\left(y^{(k)}\right) - y^{(k)}$$

Extra options

- Semi-implicit methods, "lagging"
  - Run a single step of Newton
  - Equivalently, linearize around current y, solve linear problem for next y
  - Linear stability analysis unchanged, but practice suggests not as robust (e.g. mass-spring problem)
- If Newton fails to converge, try again with smaller time step
  - Equations become easier to solve

F=ma

- Not always a good thing to reduce 2nd order equations to 1st order system
- Example: "symplectic Euler" or "velocity Verlet" (naming is still a bit mixed up)

$$v_{n+\frac{1}{2}} = v_{n-\frac{1}{2}} + \Delta t\, a(x_n)$$

$$x_{n+1} = x_n + \Delta t\, v_{n+\frac{1}{2}}$$

- If acceleration depends on velocity, may need to go implicit to retain second order accuracy

Example problem: gravity

- Take n point masses ("n-body problem")
- Force between two points:

$$\vec{f}_{ij} = -G \frac{m_i m_j}{\left\| \vec{x}_i - \vec{x}_j \right\|^3} \left(\vec{x}_i - \vec{x}_j \right)$$

- Total force on a point: sum of forces from all (n-1) other points
- Big bottleneck: $O(n^2)$ work to simply evaluate acceleration

Fast approximation

- A cluster of points far away can be approximated as a single point at the centre of mass
- How accurate?