ODEs - ordinary differential equations

\[ \frac{dy(x)}{dx} = \left[ f(x, y(x)) \right] \]

"RHS" right-hand-side

( first-order, involving first derivative

- two kinds of ODE problems:

1) IVP: initial value problem: \( y(x_0) \) is given at some starting point \( x_0 \) (inner or outer boundary)

This will be the topic of next lecture

2) BVP: boundary value problem: \( y \) is known at two "ends" (the boundaries) of the domain and these "boundary conditions" must be satisfied.

Example of a system of IVP ODEs: simplified stellar structure

- Hydrostatic equilibrium:
  \[ \frac{dP}{dr} = -\frac{GM}{r^2} \gamma \]

- Mass conservation:
  \[ \frac{dM}{dr} = 4\pi r^2 \gamma \]

- Need: equation of state - use polytrope: \( P = K \cdot r^n \gamma \)
  (need this to relate density to pressure)

IVP: specify
  \[ P_c = K \cdot r^n \gamma \]
  \( P_c = 0 \)

\( r_c = 0 \)

Note, we must take special care at \( r_c = 0 \)
Euler's Method

solve

- \( y' = f(x, y) \) with \( y(x_0) = y_0 \)
- introduce fixed step size \( h \)
- estimate of \( y(x) \) at \( x_i = x_0 + ih \) via Taylor expansion:

\[
y(x_i) = y(x_0 + h) = y(x_0) + y'(x_0) \cdot h + O(h^2)
\]

\[
= y(x_0) + h \cdot f(x_0, y(x_0)) + O(h^2)
\]

\( \text{"Forward Euler"} \)

\( y(x_{i+1}) = y_i + h \cdot f(x_i, y_i) + O(h^2) \)

note: local error \( \propto h^2 \), but \( h = \frac{L}{N} \) \( \iff \) # of points

and we are taking \( N \) steps, so

total error \( \propto N \cdot h^2 = \frac{N \cdot L^2}{N^2} = \frac{L^2}{N} = L \cdot h \rightarrow \propto h \)

- FE is an \underline{explicit method} - \( y_{n+1} \) given explicitly in terms of known quantities \( y_n, f(x_n, y_n) \)

+ simple, efficient
- \( \vdash \) can have stability problems
Stability of FE

\[ y' = -ay \quad y(0) = 1, \quad a > 0, \quad \text{and} \quad y' = \frac{2y}{x^2} \]

exact solution: \[ y = \exp(-at) \quad y(0) = 1 \quad y(\infty) = 0 \]

Now apply FE:

\[ y_{n+1} = y_n - ah y_n = (1 - ah)^2 y_{n-1} = \ldots = (1 - ah)^{n+1} y_0 \]

So in order to prevent potential amplification of error we must require \( |1 - ah| < 1 \)

Three cases:
1. \( 0 < 1 - ah < 1 \) \( (1 - ah)^{n+1} \) decays (good!)
2. \( -1 < 1 - ah < 0 \) \( (1 - ah)^{n+1} \) oscillates (not good!)
3. \( 1 - ah < 1 \) \( (1 - ah)^{n+1} \) diverges or oscillates! (no)

Overall stability criterion: \( h < \frac{2}{a} \)

"FE is conditionally stable if \( h < \frac{2}{a} \)

Central Euler:

\[ y_{n+1} = y_n + h f(x_{n+1}, y_{n+1}) \]

"Implicit" because \( y_{n+1} \) depends on unknown quantities (itself!)

For our toy problem \( y' = -ay \)
\[ y_{n+1} = y_n + h \cdot (-ay_{n+1}) \]

\[ y_{n+1} = \frac{1}{1 + ka} \cdot y_n \quad \text{since } k > 0, \quad y_{n+1} < y_n \]

- unconditionally stable

**Higher-Order Methods for ODE integration**

- there are many; just go to higher order in Taylor expansion
- among the most common: Runge-Kutta Methods

2nd-order RK (RK2):

\[ y'(x) = f(x, y(x)) \]

Ansatz:

\[ y_{n+1} = y_n + a k_1 + b k_2 \]

Can show (note):

\[ k_1 = h f(x_n, y_n) \]

\[ k_2 = h f(x_n + \frac{h}{2}, y_n + \frac{1}{2} k_1) \]

\[ a = 0, \quad b = 1 \]

\[ y_{n+1} = y_n + k_2 + O(h^2) \]

See notes for RK3, RK4, Predictor-Corrector
Bed to our example:

2 equations: use vector form with FF

\[
\dot{\mathbf{y}}_{n+1} = \mathbf{y}_n + h \cdot \hat{F}\left(x_n, \dot{\mathbf{y}}_n\right)
\]

RHS:

\[
\begin{align*}
\mathbf{F_1} & \rightarrow -G \cdot \mathbf{N}_n \cdot \mathbf{f}_n \\
\mathbf{F_2} & \rightarrow 4\pi r_n^2 \cdot \mathbf{f}_n
\end{align*}
\]

So if RHS is a function: must pass in \( \mathbf{N}_n, \mathbf{f}_n, r_n \)

must return \( \text{RHSF1} \) and \( \text{RHSF2} \)

Step by step implementation:

1. setup grid with N points from 0 to some large radius \( R_{max} \), which is larger than what we expect the star to be

2. setup initial conditions: \( K, \Pi, f_i, M_e, P_e \),

   set integer \( nsurf = 0 \) (will contain point that represents the surface of the star)

3. loop over all grid points from 0 to \( N \) points - 1:

   (3a) call integrator function with data from point \( n \)

       to obtain data at point \( n+1 \)
(3b) if pressure \([n+1]\) < press_min and \(n_{surf} = 0\):
    \(n_{surf} = n\)

    and if \(n+1 > n_{surf}\) and \(n_{surf} \neq 0\):
    set all vars to their values at \(n = n_{surf}\)

(3c) insert EOS to obtain \(\mathfrak{f}_{n+1}\)

4. Output \(M_{surf}, \mathfrak{g}[0]\) and \(\mathfrak{f}_{x} = \mathfrak{g}[n_{surf}]\)

---

integrator function: this is what is passed in

\[
\text{tov_integrate}(\text{radius}[n], \text{dr}, \text{press}[n], \text{ch}[n], \text{mass}[n]):
\]
new = np.zeros(2)
old = np.zeros(2)
old \[0\] = press
old \[1\] = mass

new = old + dr \cdot \text{tov_RHS}(\text{radius}, \text{press}, \text{ch}, \text{mass})

return (new \[0\], new \[1\])