There exist several methods for solving ODEs such as the time-symmetric Leapfrog method or the half-point Verlet method. However, the classic algorithm is the fourth-order Runge-Kutta (RK4) method.

**RK4-method:~#** Runge-Kutta methods are slope-extrapolating algorithms via subsequent Taylor series approximations to solve a first-order IVP of form x’(t)=f(t,x), x(0)=x­0. It has an error of O(h5). It can be mathematically proven that if x’(t)=f(t), RK4 is Simpson’s rule.

for t in tlist:

xlist.append(x)

k1 = h\*f(x,t)

# RK4 uses four slope quantities for extrapolation : k1 being at the start (Euler’s method), k2 at the midpoint using k1, k3 at

k2 = h\*f(x+0.5\*k1,t+0.5\*h)

k3 = h\*f(x+0.5\*k2,t+0.5\*h)

k4 = h\*f(x+k3,t+h)

the midpont using k2, and k4 at the end of the interval.

x += (k1+2\*k2+2\*k3+k4)/6

! Pitfall: RK4 is relatively insensitive to erroneous syntax. Always re-check the code and avoid basing red-flags on results alone.

**order-reduction:~#** RK4 can be extended directly to solve multidimensional systems. As a major implication, reduction of order allows RK4 to be usable for higer-order ordinary differential equations. For instance, x’’[t]=f[x,t] can be reduced to y’[t]=f[x,t] and x’[t]=y[x,t] and solved simultaneously.

r = array([xi,yi], float)

for t in tpoints:

xpoints.append(r[0])

ypoints.append(r[1])

k1 = h\*f(r,t)

…

r += (k1+2\*k2+2\*k3+k4)/6

# A multidimensional function f(r,t) uses an array object r to contain both x and y. Since vectors follow similar rules of additon and taylor expansion, RK4 algorithm carries on similarly (denoted by ellipses).

! Pitfall: RK4 has no time-reversal symmetry. Use leapfrog method for energy-sensitive physics problems with higher-order equations of motion.

**boundary-value-problem:~#** To solve a BVP with conditions of form x(0)=0 and x(tf)=0, the shooting method converts a boundary condition into an initial condition .

def f(v):

r = array([0,v], float)

…

# Ellipses denote RK4 algorithm. To find initial condition, the RK4 solution is parametrized in terms of an initial condition v and finding the roots of f(v) via a root-finding algorithm.

! Pitfall: Problems of root-finding algorithm carries over here such as properly bracketing a guess and chaotic dynamics.