Computational Physics

Shuai Dong

Ordinary differential equations

- Initial-value problems
- The Euler methods
- Predictor-corrector methods
- The Runge-Kutta method
- Chaotic dynamics
- Boundary-value problems
- The shooting method
- Linear equations
- Eigenvalue problems

Most problems in physics and engineering appear in the form of differential equations.

For example

①the motion of a classical particle is described by Newton's equation

$$\vec{f} = m\vec{a} = m\frac{d\vec{v}}{dt} = m\frac{d^2\vec{r}}{dt^2}$$

②The motion of a quantum particle is described by the Schrodinger equation

$$i\hbar\frac{\partial\Psi}{\partial t} = -\frac{\hbar^2}{2m}\nabla^2\Psi + V\Psi$$

③The dynamics and statics of bulk materials such as fluids and solids are all described by differential equations.

In general, we can classify ordinary differential equations into three major categories:

initial-value problems	time-dependent equations with given initial conditions
boundary-value problems	differential equations with specified boundary conditions
eigenvalue problems	solutions for selected parameters (eigenvalues) in the equations

Initial-value problems

- Typically, initial-value problems involve dynamical systems. For example, the motion of the moon, earth, and sun, the dynamics of a rocket, or the propagation of ocean waves.

- A dynamical system can be described by a set of first-order differential equations:

$\frac{d\vec{y}}{dt} = g(\vec{y},t)$	$\vec{y} = (y_1, y_2, \cdots, y_l)$	the generalized position vector

$g(\vec{y},t) = [g_1(\vec{y},t), g_2(\vec{y},t), \cdots g_l(\vec{y},t)]$	the generalized velocity vector

Example

- A particle moving in one dimension under an elastic force

$$\vec{f} = m\vec{a} = m\frac{d\vec{v}}{dt} = -k\vec{x}$$

- Define y_1=x; y_2=v;

- Then we obtain:

$$\frac{dy_1}{dt} = y_2,$$

$$\frac{dy_2}{dt} = -\frac{k}{m}y_1,$$

If the initial position y_1(0)=x(0) and the initial velocity y_2(0) = v(0) are given, we can solve the problem numerically.

The Euler method

$$\frac{dy}{dt} \approx \frac{y_{i+1} - y_i}{t_{i+1} - t_i} \approx g(y_i, t_i)$$

$$y_{i+1} = y_i + \tau g_i + O(\tau^2)$$

$$\tau = t_{i+1} - t_i$$

The accuracy of this algorithm is relatively low. At the end of the calculation after a total of n steps, the error accumulated in the calculation is on the order of $nO(t^2) \sim O(t)$.

We can formally rewrite the above equation as an integral

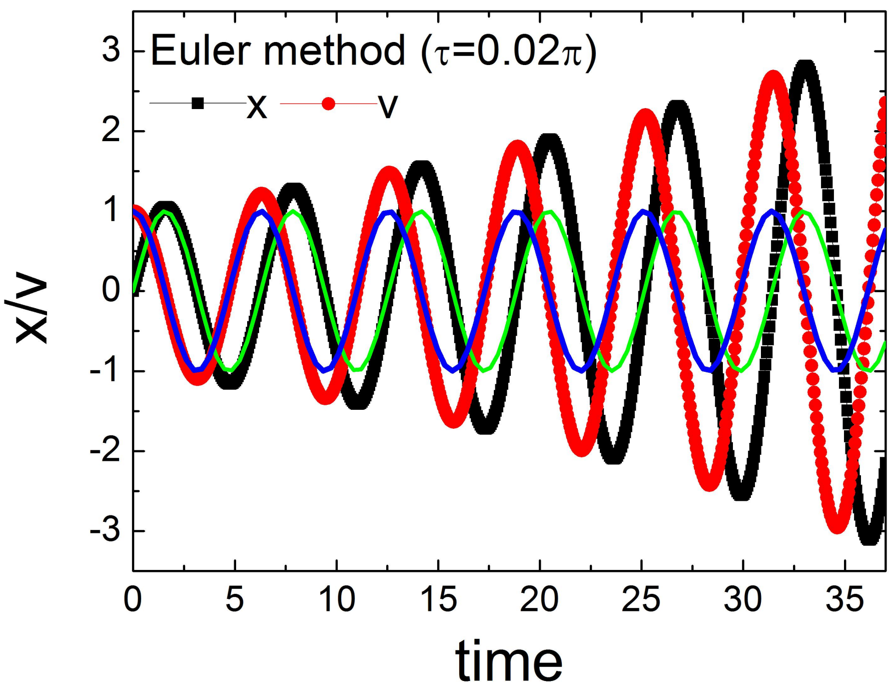
$$y_{i+j} = y_i + \int_{t_i}^{t_{i+j}} g(y,t)dt$$

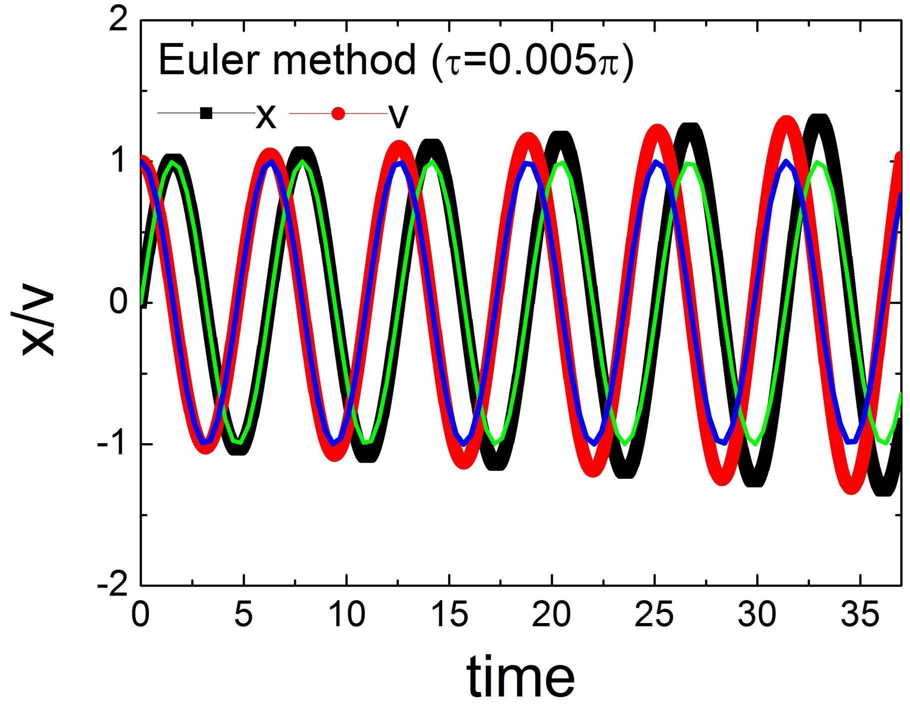
which is the exact solution if the integral can be obtained exactly.

- Because we can not obtain the integral exactly in general, we have to approximate it.
- The accuracy in the approximation of the integral determines the accuracy of the solution.
- If we take the simplest case of j = 1 and approximate $g(y, t) = g_i$ in the integral, we recover the Euler algorithm.

Code example

- 4.1.Euler.cpp (1.3.Intro.cpp)





Predictor-corrector method

- Use the solution from the Euler method as the starting point.
- Use a numerical quadrature to carry out the integration.
- For example, if we choose j = 1 and use the trapezoid rule for the integral.

$$y_{i+1} = y_i + \frac{\tau}{2}(g_i + g_{i+1}) + O(\tau^3)$$

Code example

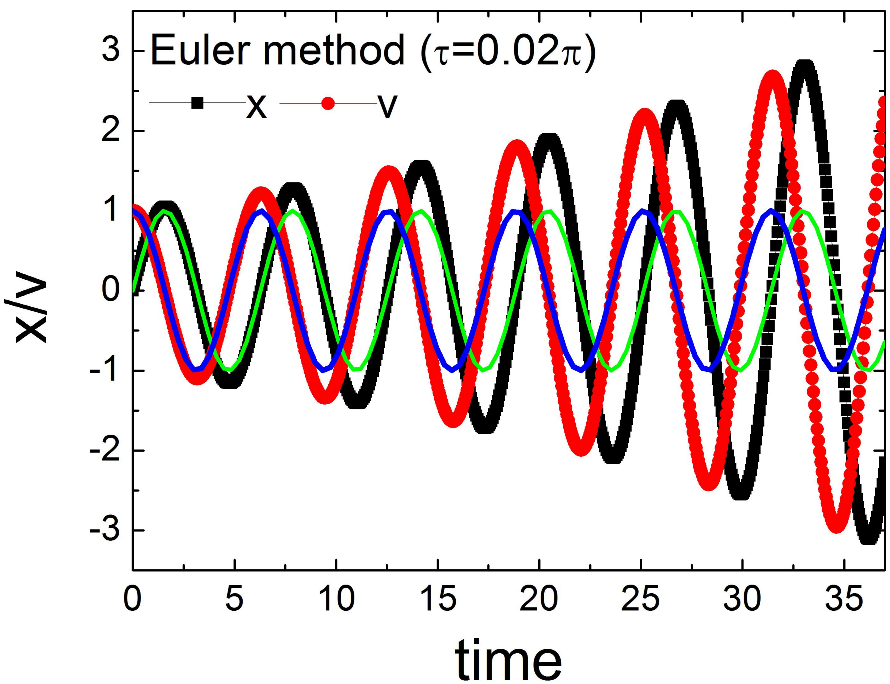
- The harmonic oscillation.
- Euler method: poor accuracy with t = 0.02π.
- Predictor-corrector method: much better?

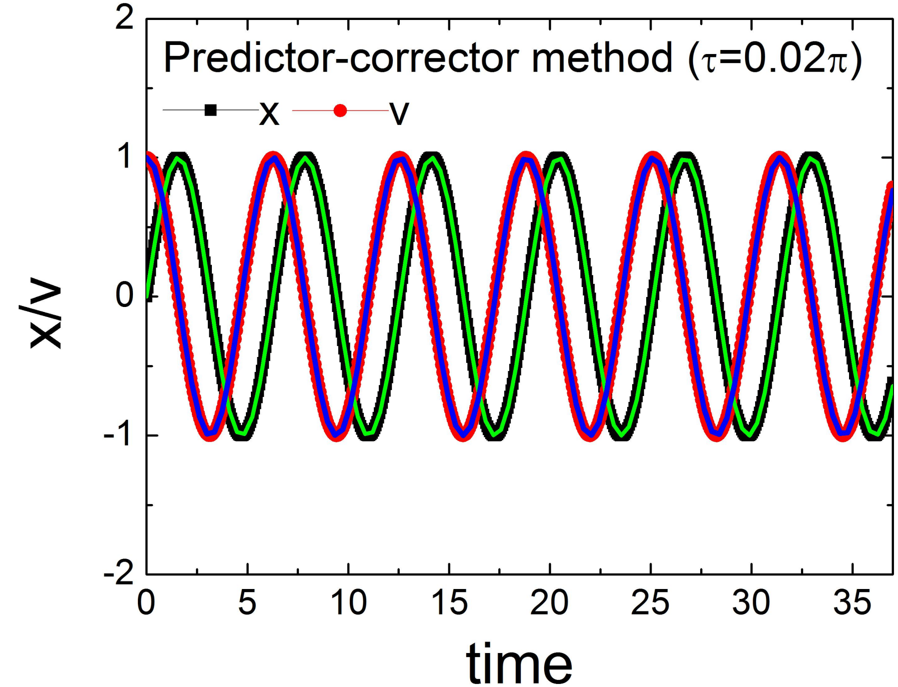
- // Predict the next position and velocity
- x[i+1] = x[i]+v[i]*dt;
- v[i+1] = v[i]-x[i]*dt;

- // Correct the new position and velocity
- x[i+1] = x[i]+(v[i]+v[i+1])*dt/2;
- v[i+1] = v[i]-(x[i]+x[i+1])*dt/2;

Code example

4.2. Predictor-Corrector.cpp





- Another way to improve an algorithm is by increasing the number of mesh points j. Thus we can apply a better quadrature to the integral.

$$y_{i+j} = y_i + \int_{t_i}^{t_{i+j}} g(y,t)dt$$

- For example, take j = 2 and then use the linear interpolation scheme to approximate g(y, t) in the integral from g_i and g_{i+1}:

$$g(y,t) = \frac{t - t_i}{\tau} g_{i+1} - \frac{(t - t_{i+1})}{\tau} g_i + O(\tau^2)$$

Now if we carry out the integration with g(y, t) given from this equation, we obtain a new algorithm

$$y_{i+2} = y_i + 2\tau g_{i+1} + O(\tau^3)$$

which has an accuracy one order higher than that of the Euler algorithm.
However, we need the values of the first two points in order to start this algorithm, because $g_{i+1} = g(y_{i+1}, t_{i+1})$.

We can make the accuracy even higher by using a better quadrature.

For example, we can take j = 2 in above equation and apply the Simpson rule to the integral. Then we have

$$y_{i+2} = y_i + \frac{\tau}{3}(g_{i+2} + 4g_{i+1} + g_i) + O(\tau^5)$$

This implicit algorithm can be used as the corrector if the previous algorithm is used as the predictor.

A car jump over the yellow river

- 1997年，香港回归前夕，柯受良驾驶跑车成功飞越了黄河天堑壶口瀑布，长度达55米。飞越当天刮着大风，第一次飞越没有成功，但第二次成功了，其中有过很多危险的动作，但他都安全度过了，因此获得了“亚洲第一飞人”的称号。

1953-2003

- Let us take a simple model of a car jump over a gap as an example.
- The air resistance on a moving object is roughly given by $f_r = -\kappa \upsilon v = -cA\rho \upsilon v$, where A is cross section of the moving object, ρ is the density of the air, and c is a coefficient that accounts for all the other factors.
- So the motion of the system is described by the equation set

$$\frac{d\vec{r}}{dt} = \vec{v}, \frac{d\vec{v}}{dt} = \vec{a} = \frac{\vec{f}}{m},$$

$$\vec{f} = -mg\hat{y} - \kappa \upsilon \vec{v}.$$

Code example

- f is the total force on the car of a total mass m. Here y is the unit vector pointing upward.
- Assuming that we have the first point given, that is, r_0 and v_0 at $t = 0$.

4.3.FlyingCar.cpp

The Runge–Kutta method

Formally,we can expand y(t+τ) in terms of the quantities at t with the Taylor expansion:

$$y(t+\tau) = y + \tau y' + \frac{\tau^2}{2} y'' + \frac{\tau^3}{3!} y^{(3)} + \cdots$$

A particle moving in one dimension under an elastic force $\vec{f} = m\vec{a} = m\frac{d\vec{v}}{dt} = -k\vec{x}.$

We know the initial condition x(0),v(0).

- $x(t)=x(0)+x'(0)t+x''(0)t^2/2+\ldots\ldots$
- $v(t)=v(0)+v'(0)t+v''(0)t^2/2+\ldots\ldots$
- $x'=v; \qquad x''=v'=-kx/m$
- $v''=-kx'/m=-kv/m$
- The same process for higher orders x^n and v^n:
- $x'''=v'';$
- $v'''=-kv'/m=k^2x/m^2;$
- $x''''=v''';$
- $v''''=k^2v/m^2$

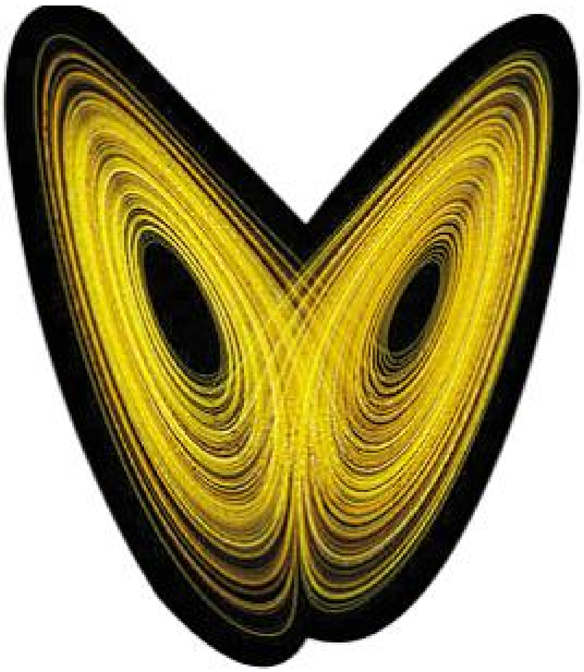
Code example

4th-order Runge–Kutta algorithm for the harmonic oscillator

[4.4.RungeKutta.cpp]

Chaotic dynamics

- nonlinear item
- nonlinear physics
- chaos

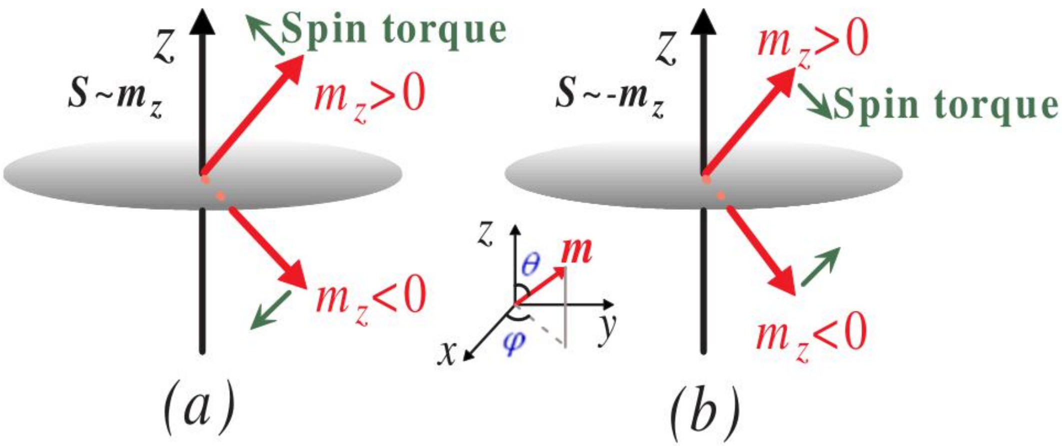


An undergraduate project

PHYSICAL REVIEW B **76**, 054414 (2007)

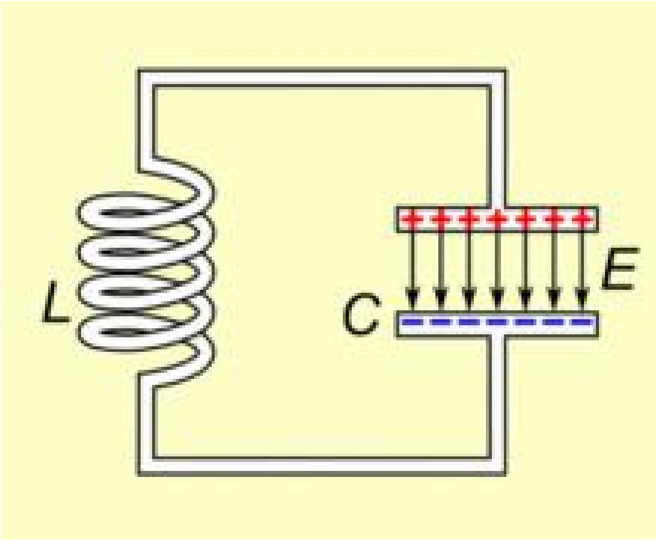
Magnetization oscillation in a nanomagnet driven by a self-controlled spin-polarized current: Nonlinear stability analysis

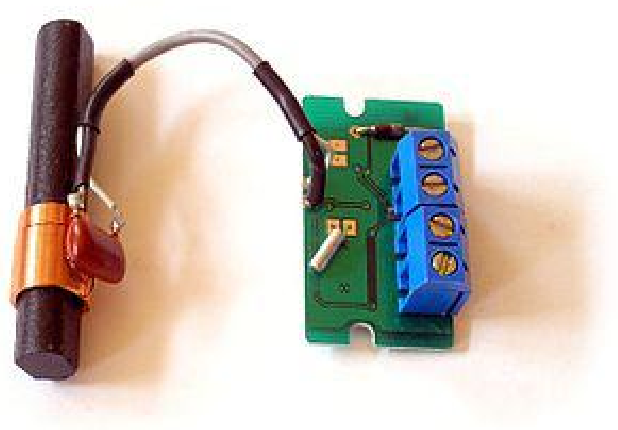
Xiao Chen,[1] Zhizhuai Zhu,[1] Yun Jing,[2] Shuai Dong,[1] and Jun-Ming Liu[1,3,*]
[1]*Laboratory of Solid State Microstructures, Nanjing University, Nanjing 210093, China*
[2]*Department of Electronic Science and Engineering, Nanjing University, Nanjing 210093, China*
[3]*International Center for Materials Physics, Chinese Academy of Sciences, Shenyang 110015, China*



$$\frac{1+\alpha^2}{\gamma}\frac{d\boldsymbol{m}}{dt} = -\boldsymbol{m}\times\boldsymbol{h}_{eff} - \alpha\boldsymbol{m}\times(\boldsymbol{m}\times\boldsymbol{h}_{eff}) - \boldsymbol{m}\times(\boldsymbol{m}\times\boldsymbol{S})$$

An LC circuit





- An LC circuit, also called a resonant circuit, tank circuit, or tuned circuit, is an electric circuit consisting of an inductor, represented by the letter L, and a capacitor, represented by the letter C, connected together. The circuit can act as an electrical resonator, an electrical analogue of a tuning fork, storing energy oscillating at the circuit's resonant frequency.

Equations of LC circuit

$$V_C = V_L.$$

$$i_C = -i_L.$$

$$V_L(t) = L\frac{\mathrm{d}i_L}{\mathrm{d}t}$$

$$i_C(t) = C\frac{\mathrm{d}V_C}{\mathrm{d}t}.$$

$$\frac{\mathrm{d}^2 i_L(t)}{\mathrm{d}t^2} + \frac{1}{LC} i_L(t) = 0.$$

$$\omega_0 = \frac{1}{\sqrt{LC}}.$$

$$\frac{\mathrm{d}^2 i_L(t)}{\mathrm{d}t^2} + \omega_0^2 i_L(t) = 0.$$

- https://en.wikipedia.org/wiki/LC_circuit
- https://baike.baidu.com/item/LC%E6%8C%AF%E8%8D%A1%E7%94%B5%E8%B7%AF/2139277?fr=aladdin

Homework

- Use the 4th order Runge-Kutta method to solve a LC circuit with resistance & excitation.

Boundary-value problems

- The solution of the Poisson equation with a given charge distribution and known boundary values of the electrostatic potential.
- Wave equations with given boundary conditions.
- The stationary Schrodinger equation with a given potential and boundary conditions.

One-dimensional example

$$\boxed{u'' = f(u, u'; x)}$$

- Where u is a function of x, u' and u'' are the 1st and 2nd derivatives of u with respect to x; f(u,u';x) is a function of u, u', and x.

- Either u or u' is given at each boundary point. We can always choose a coordinate system so that the boundaries of the system are at x=0 and x=1 without losing any generality if the system is finite.

- For example, if the actual boundaries are at $x = x_1$ and $x = x_2$ for a given problem, we can always bring them back to $x'=0$ and $x'=1$ by moving and scaling with a transformation: $x' = (x - x_1)/(x_2 - x_1)$

- For problems in one dimension, we can have a total of four possible types of boundary conditions:

  (1) $u(0) = u_0$ and $u(1) = u_1$;

  (2) $u(0) = u_0$ and $u'(1) = v_1$;

  (3) $u'(0) = v_0$ and $u(1) = u_1$;

  (4) $u'(0) = v_0$ and $u'(1) = v_1$.

- (2) is the same as (3) by reversing the direction.

- The boundary-value problem is more difficult to solve than the similar initial-value problem with the differential equation.

- For example, if we want to solve an initial-value problem and the initial conditions $u(0) = u_0$ and $u'(0) = v_0$, the solution will follow the algorithms discussed earlier.

- However, for the boundary-value problem, we know only $u(0)$ or $u'(0)$, which is not sufficient to start an algorithm for the initial-value problem without some further work.

Example:

longitudinal vibrations along an elastic rod

- The equation describing the stationary solution of elastic waves is $u''(x) = -k^2 u(x)$
- If both ends ($x=0$ and $x=1$) of the rod are fixed, the boundary conditions are $u(0)=u(1)=0$.
- If one end ($x=0$) is fixed and the other end ($x=1$) is free, the boundary conditions are $u(0)=0$ and $u'(1)=0$.

- For example, if both ends of the rod are fixed, the eigenfunctions

$$u_l(x) = \sqrt{2}\sin k_l x$$

are the possible solutions of the differential equation.

- Here the eigenvalues are given by

$$k_l^2 = (l\pi)^2 \quad \text{with } l = 1, 2, \ldots, \infty.$$

The shooting method

- The key here is to make the problem look like an initial-value problem by introducing an adjustable parameter; the solution is then obtained by varying the parameter.

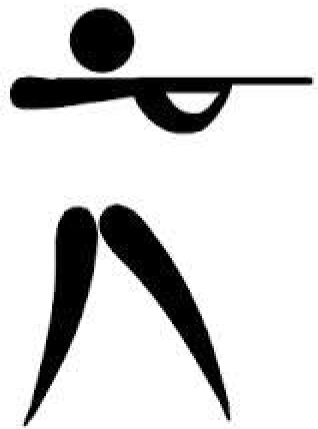
- For example, given u(0) and u(1), we can guess a value of u'(0)=a, where a is the parameter to be adjusted.

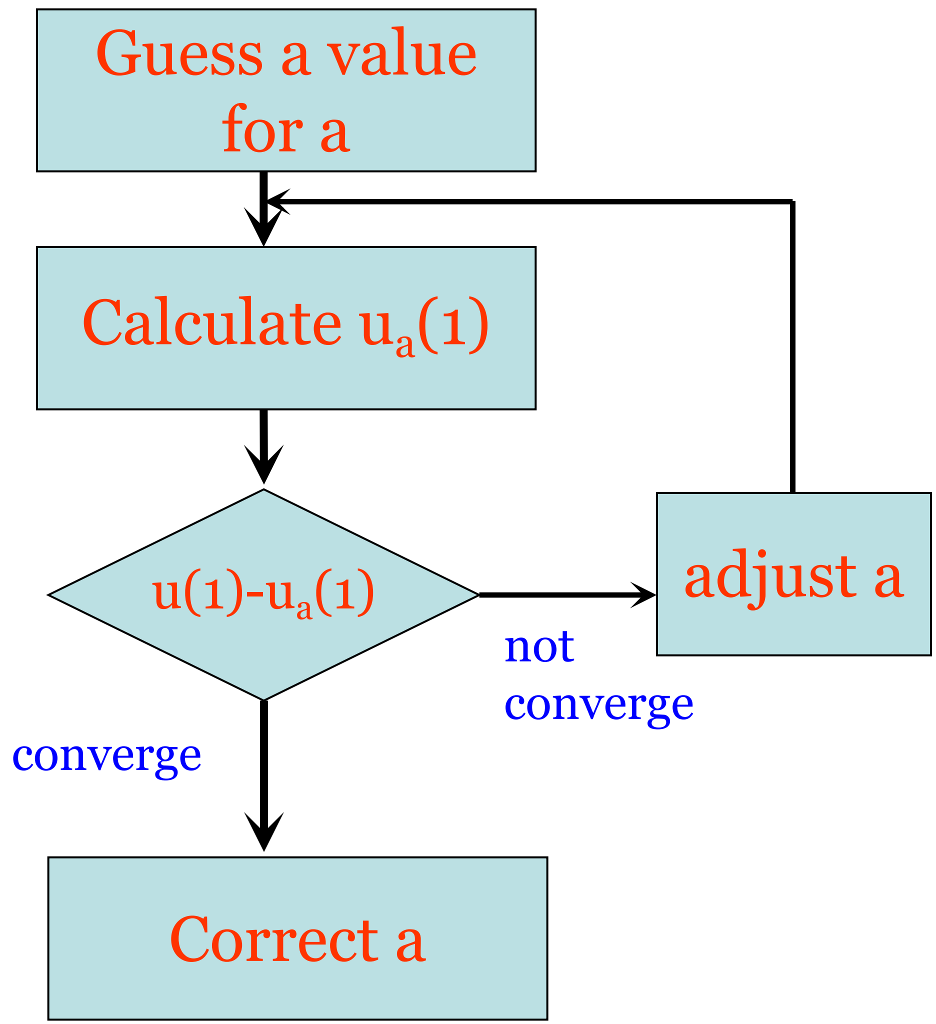
The shooting method

- For a specific a, the value of the function $u_\alpha(1)$, resulting from the integration with $u'(0)=a$ to $x = 1$, would not be the same as u_1.

- The idea of the shooting method is to use one of the root search algorithms to find the appropriate α that ensures $f(\alpha)=u_\alpha(1)-u(1)=0$ within a given tolerance δ.

The shooting method





Example

$$u'' = -\frac{\pi^2}{4}(u+1)$$

- With given boundary conditions u(0) = 0 and u(1) = 1, We can define new variables y_1=u and y_2=u';

$$\frac{dy_1}{dx} = y_2, \frac{dy_2}{dx} = -\frac{\pi^2}{4}(y_1 + 1)$$

- Assume that this equation set has the initial values $y_1(0) = 0$ and $y_2(0) = \alpha$.
- Here α is a parameter to be adjusted in order to have $f(\alpha) = u_\alpha(1)-1 = 0$.
- We can combine the secant method for the root search and the 4th-order Runge–Kutta method for initial-value problems to solve the above equation set.

Linear equations

- Many eigenvalue or boundary-value problems are in the form of linear equations, such as

$$u'' + d(x)u' + q(x)u = s(x)$$

- Assume that the boundary conditions are $u(0) = u_0$ and $u(1) = u_1$. If all d(x), q(x), and s(x) are smooth, we can solve the equation with the shooting method as shown above.

- However, an extensive search for the parameter a from $f(\alpha) = u_\alpha(1) - u_1 = 0$ is unnecessary in this case, because of the principle of superposition of linear equations: any linear combination of the solutions is also a solution of the equation.

- We need only two trial solutions $u_{\alpha 0}$ (x) and $u_{\alpha 1}$ (x), where α_0 and α_1 are two different parameters.
- The correct solution of the equation is given by

$$u(x) = au_{\alpha_0}(x) + bu_{\alpha_1}(x)$$

where a and b are determined from $u(0) = u_0$ and $u(1) = u_1$. Note that $u_{\alpha 0}(0) = u_{\alpha 1}(0) = u(0) = u_0$. So we have

$$a + b = 1,$$

$$u_{\alpha_0}(1)a + u_{\alpha_1}(1)b = u_1,$$

Code example

- 4.5.Boundary.cpp

$$u'' = -\frac{\pi^2}{4}(u+1)$$