数值计算方法：原理、算法和应用
Numerical Methods: Principles, Algorithms and Applications

授课教师：周铁

北京大学数学科学学院

2021 年 11 月 24 日

1. 常微分方程 (ODE) 的数值算法
   - 一阶 ODE 初值问题
   - 一阶 ODE 初值问题的离散化
   - 单步法 (Single-Step Methods)
   - 绝对稳定性
   - 多步方法 (Multi-Step Methods)
   - 预估-校正方法 (predictor-corrector method)
   - 一阶 ODE 组和高阶 ODE 初值问题
   - 刚性问题
   - ODE 的边值问题

一阶常微分方程 (ODE) 初值问题

一阶 ODE 初值问题：求函数 $u(t) \in C^1[t_0, T]$ 满足

$$\begin{cases} u'(t) = f(t, u(t)), \ t \in [t_0, T] \\ u(t_0) = u_0 \end{cases} \tag{1.1}$$

其中 $f(t, u) : [t_0, T] \times \mathbb{R} \longmapsto \mathbb{R}$ 是已知函数.

定理

如果 $f(t, u)$ 满足：

1. $f(t, u) \in C([t_0, T] \times \mathbb{R})$;
2. Lipschitz 条件：存在常数 L 使得

$$|f(t, u_1) - f(t, u_2)| \leq L|u_1 - u_2|, \ \forall\, t \in [t_0, T], \ \forall\, u_1, u_2 \in \mathbb{R}. \tag{1.2}$$

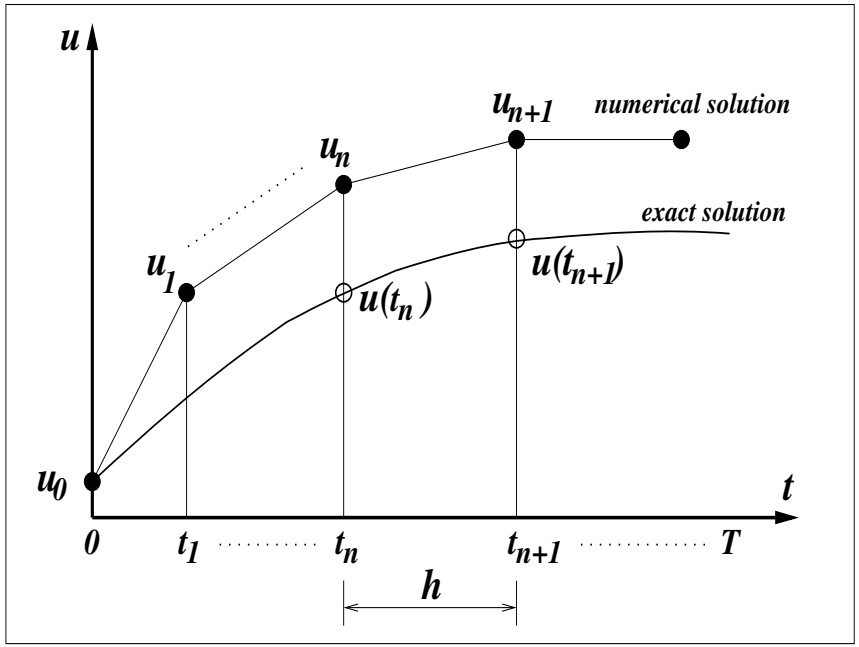
则 ODE 初值问题 (1) 的解 $u(t)$ 存在且惟一，并且 $u(t) \in C^1[t_0, T]$.

ODE 初值问题的离散化

给定区间 $[t_0, T]$ 上的等距节点 (网格点):

$$t_n = t_0 + nh,\ h = (T - t_0)/N,\ n = 0, 1, 2, \cdots, N$$

h 称为网格步长. 要计算每个网格点处的 u_n，使得 $u_n \approx u(t_n)$.



有了 $u_0, u_1, \cdots, u_N$ 后，再插值就得到 ODE 准确解 $u(t)$ 的数值解 $u_h(t)$.

显式 Euler 法

在每个网格点 $t_n (0 \le n \le N-1)$ 处，利用微分中值定理，初值问题 (1) 的解析解 $u(t)$ 都满足：

$$u(t_{n+1}) = u(t_n) + hf(\xi_n, u(\xi_n)), \quad \xi_n \in [t_n, t_{n+1}]. \tag{3.1}$$

右端的 $f(\xi_n, u(\xi_n))$ 可以看成 $u(t)$ 在区间 $[t_n, t_{n+1}]$ 中的平均斜率. 只要给出它的一种近似方法，就可得到问题 (1) 的一个数值方法.

显式 Euler 法：

$$u_{n+1} = u_n + hf(t_n, u_n), \ n = 0, 1, \cdots, N-1. \tag{3.2}$$

这个方法的来源是用 $f(t_n, u_n)$ 近似 $f(\xi_n, u(\xi_n))$.

用方法(3.2)从 u_n 计算 u_{n+1} 只要计算 f 的函数值，不用解方程，所以称为显式方法.

当 u_0 就取为准确的初值时，准确解 $u(t) \in C^2[t_0, T]$ 时，可以证明显式 Euler 方法的误差 $e_n = u(t_n) - u_n$ 满足：

$$|e_n| \leq Ch, \quad n = 0, 1, \ldots N.$$

所以称显式 Euler 法为一阶精度方法.
证明：记：

$$u_n^* = u(t_{n-1}) + hf(t_{n-1}, u(t_{n-1})) = u(t_{n-1}) + hu'(t_{n-1}). \tag{3.3}$$

由微分中值定理，

$$u(t_n) = u(t_{n-1}) + hu'(\xi_{n-1}), \quad \xi_{n-1} \in [t_{n-1}, t_n]. \tag{3.4}$$

关于误差 $e_n = u(t_n) - u_n$，由三角不等式，有

$$|e_n| = |u(t_n) - u_n| \leq |u(t_n) - u_n^*| + |u_n^* - u_n|. \tag{3.5}$$

记 $\max_{t\in[t_0,T]} |u''(t)| = C_1$，(3.3)式和(3.4)式相减，得到(3.5)式右端第一项满足：

$$|u(t_n) - u_n^*| = h|u'(\xi_{n-1}) - u'(t_{n-1})| \leq C_1 h^2$$

此项称为局部截断误差.

利用 Lipschitz 条件(1.2)，(3.5)式右端第二项满足：

$$\begin{aligned}|u_n^* - u_n| &= |u(t_{n-1}) + hf(t_{n-1}, u(t_{n-1})) - u_{n-1} - hf(t_{n-1}, u_{n-1})| \\ &\leq |u(t_{n-1}) - u_{n-1}| + hL|u(t_{n-1}) - u_{n-1}| \\ &= (1 + hL)|e_{n-1}|\end{aligned}$$

其中的 $n = 1, 2, \cdots, N.$
所以有

$$\begin{aligned}|e_1| &\leq |u(t_1) - u_1^*| + |u_1^* - u_1| \leq C_1 h^2 \\ |e_2| &\leq C_1 h^2 + (1 + hL)|e_1| = C_1 h^2[1 + (1 + hL)] \\ &\vdots \\ |e_n| &\leq C_1 h^2[1 + (1 + hL) + \cdots + (1 + hL)^{n-1}] \\ &\leq C_1 h^2 \frac{(1 + hL)^n - 1}{hL} \leq \frac{e^{L(T-t_0)} - 1}{L} C_1 h = Ch\end{aligned}$$

隐式 Euler 法、Crank-Nicolson 方法和 Heun 方法

隐式 Euler 法：

$$u_{n+1} = u_n + hf(t_{n+1}, u_{n+1}), \quad n = 0, 1, \cdots, N-1.$$

这个方法的来源是在(3.1)式中用 $f(t_{n+1}, u_{n+1})$ 近似 $f(\xi_n, u(\xi_n))$. 它被称为是“隐式 (implicit)”的，因为当 f 关于 u 是非线性函数时，从 u_n 计算 u_{n+1} 需要解非线性方程. 可以证明隐式 Euler 法还是一阶精度的方法 ($|e_n| = \mathcal{O}(h)$).

Crank-Nicolson 方法 (二阶，隐式)：

$$u_{n+1} = u_n + \frac{h}{2}[f(t_n, u_n) + f(t_{n+1}, u_{n+1})], \ n = 0, 1, \cdots, N-1.$$

稍微修改一下就得到 Heun 方法 (二阶，显式)：

$$\begin{cases} u^*_{n+1} & = u_n + hf(t_n, u_n) \\ u_{n+1} & = u_n + \frac{1}{2}h[f(t_n, u_n) + f(t_{n+1}, u^*_{n+1})], \ n = 0, 1, \cdots, N-1.. \end{cases}$$

绝对稳定性

先看看 ODE 解的稳定性. 就看测试方程初值问题：

$$\begin{cases} u'(t) = \lambda u(t), \quad t > 0, \quad \text{常数}\, \lambda \in \mathbb{C}, \\ u(0) = u_0, \end{cases} \tag{4.1}$$

其准确解为：$u(t) = u_0\, e^{\lambda t}$.
显然

- 当 $\mathrm{Re}(\lambda) > 0$ 时，有 $\lim\limits_{t \to +\infty} u(t) = \infty$，问题(4.1)的解不稳定.
- 当 $\mathrm{Re}(\lambda) < 0$ 时，有 $\lim\limits_{t \to +\infty} u(t) = 0$，问题(4.1)的解稳定.

对固定网格步长 $h > 0$，将显式 Euler 法应用于测试方程(4.1)，数值解为

$$u_n = (1 + h\lambda) u_{n-1} = (1 + h\lambda)^n u_0, \quad n = 1, 2, 3, \cdots,$$

从初值开始，计算到第 n 个网格点时的放大因子为 $|1 + h\lambda|^n$.

当 $\mathrm{Re}(\lambda) < 0$ 时，问题(4.1)的解是稳定的，但只有

$$|1 + h\lambda| < 1 \tag{4.2}$$

成立时，显式 Euler 法的解 u_n 才能稳定，即 $\lim_{n\to+\infty} u_n = 0$.
为什么用测试方程(4.1)推导稳定性条件?
对于一般的 ODE

$$u' = f(t, u),$$

如果解 u 有小扰动 $u \to u + v$，其中 $|v|$ 很小，利用 Taylor 展开就有

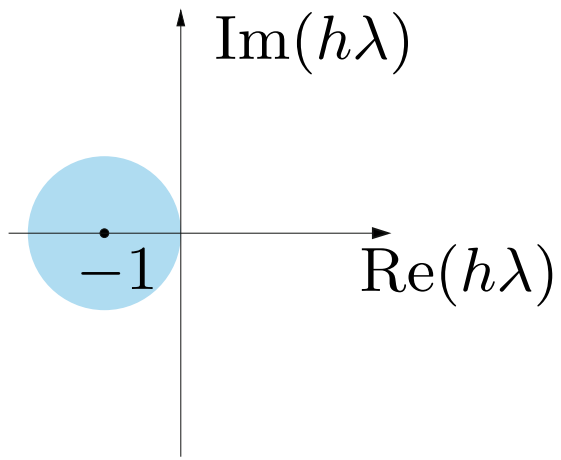
$$u' + v' = f(t, u + v) = f(t, u) + \frac{\partial f}{\partial u}(t, u)v + \mathcal{O}(|v|^2).$$

由于 u 是解，再忽略高阶小量 $\mathcal{O}(|v|^2)$，就有

$$v' = \frac{\partial f}{\partial u}(t, u)v = \lambda v.$$

这里假设 $\lambda = f_u(t, u)$ 为常数.
我们希望小扰动 $v(t)$ 不要随着 $t \to +\infty$ 而扩大，所以只考虑 $\mathrm{Re}(\lambda) \le 0$ 的情形.

不等式(4.2)称为显式 Euler 法的“绝对稳定性条件”，它是对 $h\lambda$ 的限制. 如果记 $z = h\lambda$，则满足 $|1+z| \leq 1$ 的全体复数是复平面 $\mathbb{C}$ 中圆心在 $(-1, 0)$ 的单位圆盘.
当 λ 是实数时，不等式(4.2)等价于 $h\lambda \in (-2, 0)$，$|\lambda|$ 如果很大，网格步长 h 就必须很小.



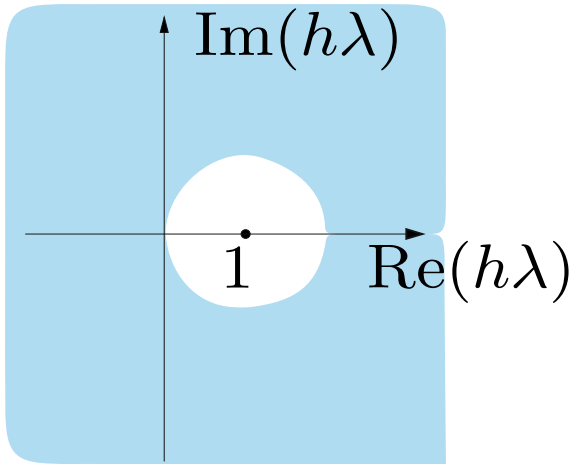
如果用隐式 Euler 法解测试问题(4.1)，容易得到：$u_n = u_{n-1} + h\lambda u_n$.
进而有：

$$u_n = \left(\frac{1}{1-h\lambda}\right)^n u_0, \quad n = 1, 2, 3, \cdots$$

显然，要使隐式 Euler 方法的放大因子小于 1，只要

$$|1 - h\lambda| > 1,$$

即 $z = h\lambda$ 要落在复平面 $\mathbb{C}$ 中圆心在 $(1, 0)$ 的单位圆盘外部.

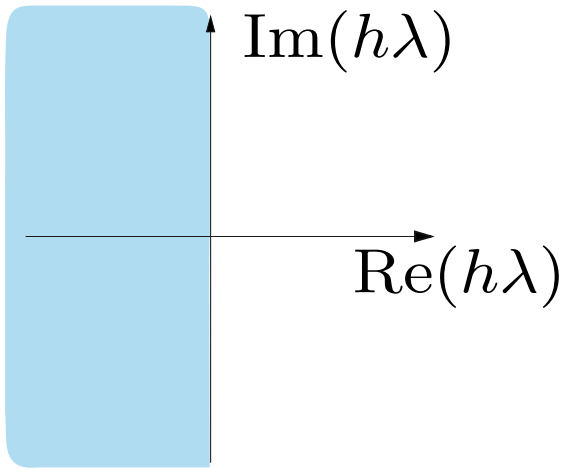


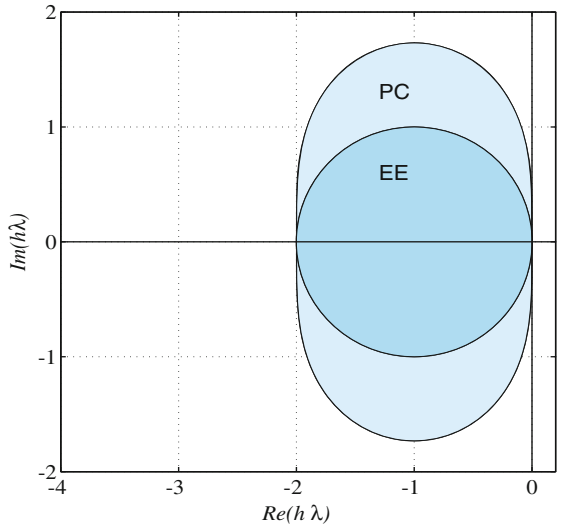
从上图可见，当 $\mathrm{Re}(\lambda) < 0$ 时，对任意大的网格步长 $h > 0$，隐式 Euler 方法都是稳定的，所以我们称隐式 Euler 方法是无条件稳定的.

将 Crank-Nicolson 方法 (隐式方法) 应用到测试问题(4.1)得到:

$$u_n = \left(\frac{1 + \frac{1}{2}h\lambda}{1 - \frac{1}{2}h\lambda} \right)^n u_0, \quad n = 1, 2, \cdots,$$

由此可以看到 Crank-Nicolson 方法也是无条件稳定的.





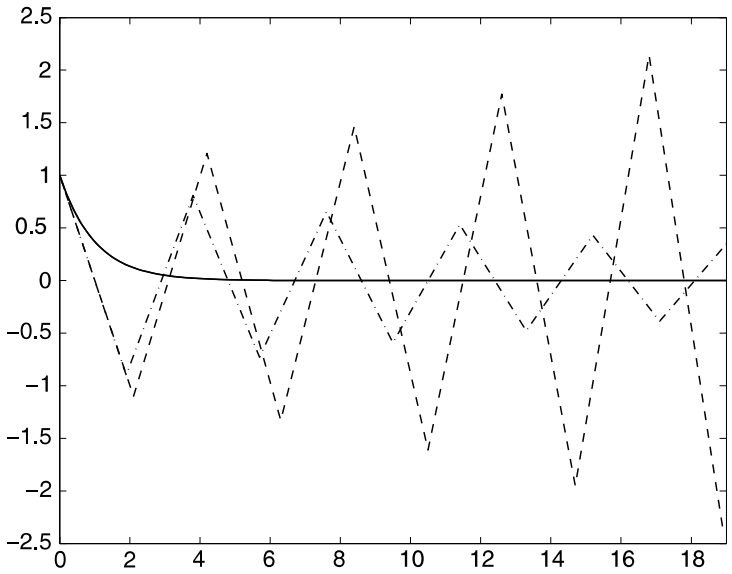
Heun 方法 (显式方法)，也称为预估矫正法 (PC)，它的稳定性条件为:

$$\left| 1 + h\lambda + \frac{1}{2}(h\lambda)^2 \right| < 1$$

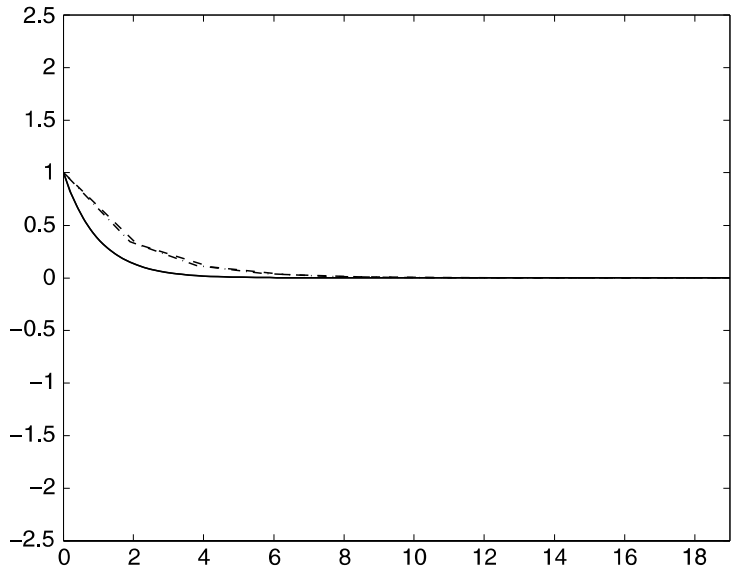
一个算例:

$$\begin{cases} u'(t) = -u(t),\ t > 0, \\ u(0) = 1, \end{cases}$$

用两种网格 $h = 2.1(--),\ h = 1.9(-.)$，分别采用两种方法计算.



显式 Euler 法的计算结果.



隐式 Euler 法的计算结果.

显式 Runge-Kutta 方法

将常微分方程 $u'(t) = f(t, u(t))$ 在 $[t_n, t_{n+1}]$ 上积分可得

$$u(t_{n+1}) - u(t_n) = \int_{t_n}^{t_n+h} f(t, u(t))dt \tag{4.3}$$

用中点公式近似右端积分

$$\int_{t_n}^{t_n+h} f(t, u(t))dt = hf\left(t_n + \frac{h}{2}, u\left(t_n + \frac{h}{2}\right)\right) + \mathcal{O}(h^3)$$

利用 Taylor 展开以及 $u' = f(t, u)$，有

$$u\left(t_n + \frac{h}{2}\right) = u(t_n) + \frac{h}{2}f(t_n, u(t_n)) + \mathcal{O}(h^2) \approx u_n + \frac{h}{2}f(t_n, u_n)$$

于是可以得到一个二阶精度的方法

$$u_{n+1} = u_n + hf\left(t_n + \frac{h}{2}, u_n + \frac{h}{2}f(t_n, u_n)\right)$$

这个方法称为 Runge 方法，可以写成下列的形式

$$\begin{cases} K_1 &= f(t_n, u_n), \\ K_2 &= f\left(t_n + \frac{1}{2}h, u_n + \frac{1}{2}hK_1\right), \quad n = 0, 1, 2, \ldots, \\ u_{n+1} &= u_n + hK_2. \end{cases}$$

如果用梯形公式近似(4.3)式右端积分

$$\int_{t_n}^{t_n+h} f(t, u(t))dt = \frac{h}{2}\left[f(t_n, u(t_n)) + f(t_{n+1}, u(t_{n+1}))\right] + \mathcal{O}(h^3)$$

利用 Taylor 展开以及 $u' = f(t, u)$，有

$$u(t_{n+1}) = u(t_n) + hf(t_n, u(t_n)) + \mathcal{O}(h^2) \approx u_n + hf(t_n, u_n)$$

又可以得到另一个二阶方法

$$u_{n+1} = u_n + \frac{h}{2}\left[f(t_n, u_n) + f(t_{n+1}, u_n + hf(t_n, u_n))\right]$$

它称为 Heun 方法，它也可以写成下列形式

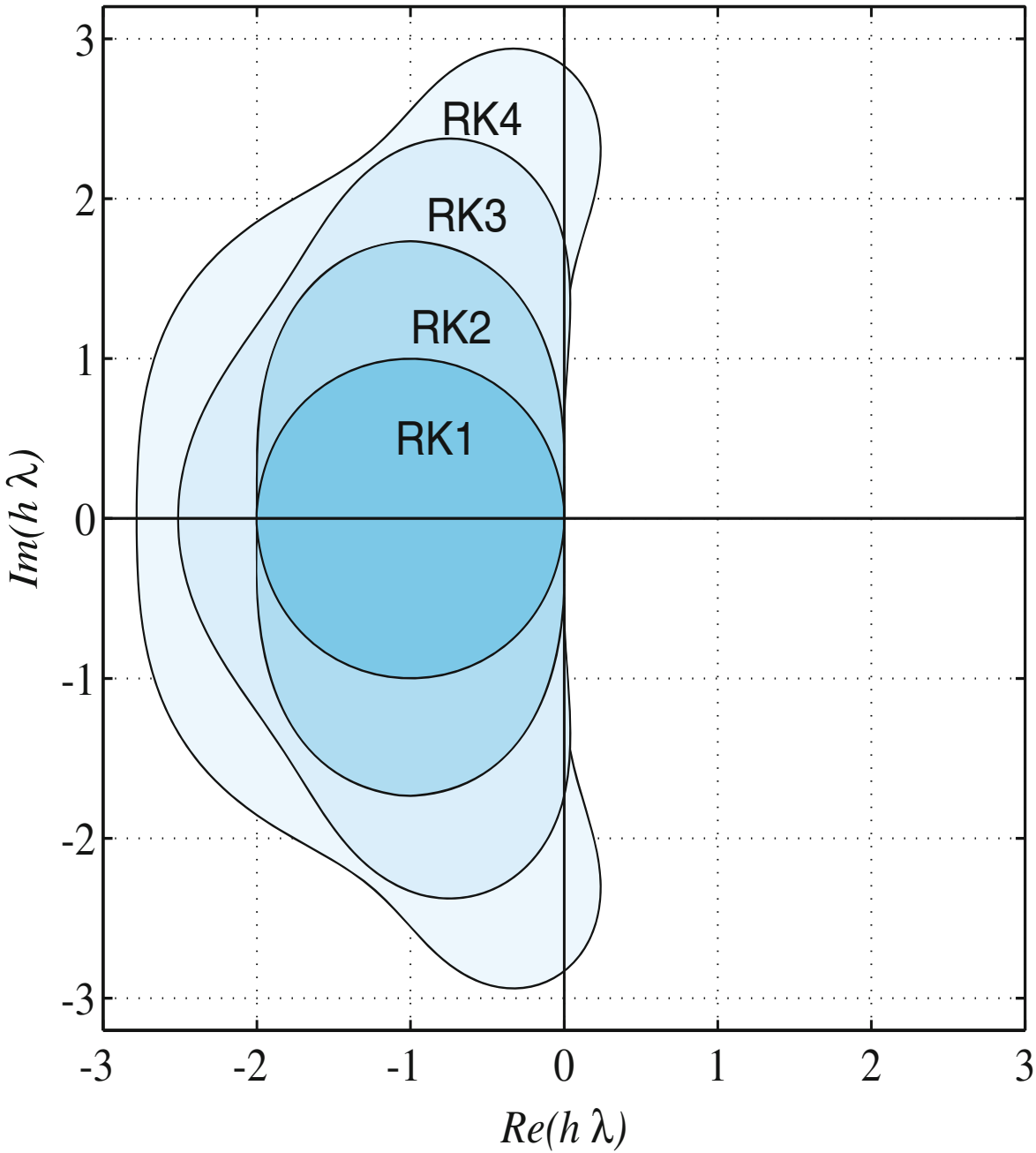
$$\begin{cases} K_1 &= f(t_n, u_n), \\ K_2 &= f\left(t_{n+1}, u_n + hK_1\right), \\ u_{n+1} &= u_n + \frac{1}{2}h(K_1 + K_2). \end{cases}$$

给定一个自然数 s，一般的显式 Runge-Kutta 方法为：

$$\begin{cases} u_{n+1} &= u_n + h\sum_{i=1}^{s} b_i K_i, \\ K_1 &= f(t_n, u_n), \\ K_i &= f\left(t_n + c_i h, u_n + h\sum_{j=1}^{i-1} a_{ij} K_j\right), \ i = 2, \cdots, s. \end{cases} \tag{4.4}$$

1. 可以得到各阶精度的单步显式方法.
2. 参数 b_i, c_i, a_{ij} 的选择要使得截断误差的阶尽量高.
3. 自然数 s 称为 Runge-Kutta 方法的“级 (stage)”.
4. 理想的情况是 s 级方法达到 s 阶精度.
5. $s = 1$ 时就是显式 Euler 法.
6.

最高的阶	1	2	3	4	5	6	7	8
级	1	2	3	4	6	7	9	11


RK4
RK3
RK2
RK1
3
2
1
0
-1
-2
-3
-3
-2
-1
0
1
2
3
Im(h λ)
Re(h λ)

2 级 2 阶显式 Runge-Kutta 方法

2 级显式 Runge-Kutta 方法的一般形式为

$$\begin{cases} u_{n+1} &= u_n + h(b_1K_1 + b_2K_2) \\ K_1 &= f(t_n, u_n) \\ K_2 &= f(t_n + c_2h, u_n + ha_{21}K_1) \end{cases}$$

其中含有 4 个参数 b_1, b_2, c_2, a_{21}.

下面证明：如果这四个参数满足

$$\begin{cases} b_1 + b_2 &= 1 \\ c_2b_2 &= \frac{1}{2} \\ a_{21}b_2 &= \frac{1}{2} \end{cases} \tag{4.5}$$

就可以得到 2 阶精度的单步显式方法.
注意(4.5)式中包含有 4 个未知量，3 个方程，所以应该有很多组解.

设 ODE 的准确解 $u = u(t) \in C^3$, 利用 Taylor 展开, 有

$$u(t_{n+1}) = u(t_n + h) = u(t_n) + hu'(t_n) + \frac{h^2}{2!}u''(t_n) + \frac{h^3}{3!}u'''(\xi) \tag{4.6}$$

其中 $t_n \le \xi \le t_{n+1}$.

利用 $u'(t) = f(t, u(t))$, 有

$$\begin{aligned}
u'(t) &= f(t, u(t)) \\
u''(t) &= f_t(t, u(t)) + f_u(t, u(t))\, u'(t) \\
u'''(t) &= f_{tt}(t, u(t)) + 2f_{tu}(t, u(t))\, u'(t) + f_{uu}(t, u(t))(u'(t))^2 + f_u(t, u(t))\, u''(t)
\end{aligned}$$

带入(4.6)，就得到

$$\begin{aligned}
u(t_{n+1}) = u(t_n) &+ hf(t_n, u(t_n)) + \frac{1}{2}h^2 f_t(t_n, u(t_n)) + \\
&+ \frac{1}{2}h^2 f_u(t_n, u(t_n)) f(t_n, u(t_n)) + \mathcal{O}(h^3)
\end{aligned} \tag{4.7}$$

此式说明：数值格式只要跟右端的前四项形式相同，就一定达到二阶精度.

待定参数的 2 级显式 Runge-Kutta 方法为

$$u_{n+1} = u_n + b_1 h f(t_n, u_n) + b_2 h f(t_n + c_2 h, u_n + h\, a_{21} f(t_n, u_n)) \tag{4.8}$$

利用二元 Taylor 展开，其中右端的

$$\begin{aligned} f(t_n + c_2 h, u_n + h a_{21} f(t_n, u_n)) = & f(t_n, u_n) + c_2 h\, f_t(t_n, u_n) + \\ & + a_{21}\, h f(t_n, u_n) f_u(t_n, u_n) + \mathcal{O}(h^2) \end{aligned}$$

所以有

$$\begin{aligned} u_{n+1} = u_n + & b_1 h f(t_n, u_n) + b_2 h f(t_n, u_n) + b_2 c_2 h^2 f_t(t_n, u_n) + \\ & + b_2 a_{21}\, h^2 f(t_n, u_n) f_u(t_n, u_n) + \mathcal{O}(h^3) \end{aligned}$$

可以整理为：

$$\begin{aligned} u_{n+1} = u_n & + h(b_1 + b_2) f(t_n, u_n) + \\ & + b_2 c_2 h^2 f_t(t_n, u_n) + b_2 a_{21} h^2 f(t_n, u_n) f_u(t_n, u_n) + \mathcal{O}(h^3) \end{aligned} \tag{4.9}$$

比较(4.9)式与(4.7)式

$$u(t_{n+1}) = u(t_n) + hf(t_n, u(t_n)) + \frac{1}{2}h^2 f_t(t_n, u(t_n) + \frac{1}{2}h^2 f_u(t_n, u(t_n))f(t_n, u(t_n))$$

$$u_{n+1} = u_n + h(b_1 + b_2)f(t_n, u_n) + b_2 c_2 h^2 f_t(t_n, u_n) + b_2 a_{21} h^2 f(t_n, u_n) f_u(t_n, u_n)$$

的右端各项可知，如果待定的 4 个参数 b_1, b_2, c_2, a_{21} 满足：

$$\begin{cases} b_1 + b_2 & = 1 \\ b_2 c_2 & = \frac{1}{2} \\ b_2 a_{21} & = \frac{1}{2} \end{cases}$$

就可得到截断误差为二阶的格式，这就是方程组(4.5)的来源.

当然，还要说明(4.9)式中的 $\mathcal{O}(h^3)$ 的系数不会为零.

如果取 $b_1 = b_2 = 1/2, c_2 = 1, a_{21} = 1$ 就得到 Heun 方法：

$$\begin{cases} u_{n+1} &= u_n + \frac{h}{2}(K_1 + K_2), \\ K_1 &= f(t_n, u_n), \\ K_2 &= f(t_n + h, u_n + hK_1). \end{cases}$$

如果取 $b_1 = 0,\ b_2 = 1,\ c_2 = 1/2,\ a_{21} = 1/2$ 就得到 Runge 方法：

$$\begin{cases} u_{n+1} &= u_n + hK_2, \\ K_1 &= f(t_n, u_n), \\ K_2 &= f\left(t_n + \frac{h}{2}, u_n + \frac{h}{2}K_1\right). \end{cases}$$

如果取 $b_1 = 1/4,\ b_2 = 3/4,\ c_2 = 2/3,\ a_{21} = 2/3$ 就得到：

$$\begin{cases} u_{n+1} &= u_n + \frac{h}{4}K_1 + \frac{3h}{4}K_2, \\ K_1 &= f(t_n, u_n), \\ K_2 &= f\left(t_n + \frac{2h}{3}, u_n + \frac{2h}{3}K_1\right). \end{cases}$$

4 级 4 阶显式 Runge-Kutta 方法

$$\begin{cases} u_{n+1} &= u_n + h(b_1K_1 + b_2K_2 + b_3K_3 + b_4K_4) \\ K_1 &= f(t_n, u_n) \\ K_2 &= f(t_n + c_2h, u_n + ha_{21}K_1) \\ K_3 &= f(t_n + c_3h, u_n + h(a_{31}K_1 + a_{32}K_2)) \\ K_4 &= f(t_n + c_4h, u_n + h(a_{41}K_1 + a_{42}K_2 + a_{43}K_3)) \end{cases}$$

有 13 个系数，可以达到 4 阶精度，其中最著名的方法是：

$$\begin{cases} u_{n+1} &= u_n + \frac{h}{6}(K_1 + 2K_2 + 2K_3 + K_4) \\ K_1 &= f(t_n, u_n) \\ K_2 &= f\left(t_n + \frac{h}{2}, u_n + \frac{1}{2}hK_1\right) \\ K_3 &= f\left(t_n + \frac{h}{2}, u_n + \frac{1}{2}hK_2\right) \\ K_4 &= f(t_n + h, u_n + hK_3) \end{cases}$$

称为经典 Runge-Kutta 方法.

自适应显式 Runge-Kutta 方法

- 步长 h 越小，同一种方法的误差就越小；同一个步长 h，方法精度越高，误差就越小.
- 每个网格点处的截断误差大小与当地准确解的性质有关，是否可以在计算时，根据当地的误差确定网格点的间距?
- 从 u_n 出发，可以用两个格式分别计算 $\tilde{u}_{n+1}$ 和 u_{n+1}，但 $\tilde{u}_{n+1}$ 的精度抵，u_{n+1} 的精度高.
- 把 u_{n+1} 看成精确解，得到误差的估计

$$|e_{n+1}| = |u(t_{n+1}) - u_{n+1}| \approx |u_{n+1} - \tilde{u}_{n+1}|$$

- 给定一个步长 h，先从 u_n 出发利用两种格式计算 u_{n+1} 和 $\tilde{u}_{n+1}$，如果

$$|u_{n+1} - \tilde{u}_{n+1}| \leq \frac{1}{4} \times \text{tolerance}$$

就接受计算结果 u_{n+1}；否则，就把步长 h 减半，重新计算 $\tilde{u}_{n+1}$ 和 u_{n+1}.

Runge-Kutta-Fehlberg Method (RKF45):

$$
\begin{aligned}
K_1 &= f(t_n, u_n) \\
K_2 &= f\left(t_n + \frac{h}{4}, u_n + \frac{1}{4}hK_1\right) \\
K_3 &= f\left(t_n + \frac{3h}{8}, u_n + \frac{3h}{32}K_1 + \frac{9h}{32}hK_2\right) \\
K_4 &= f\left(t_n + \frac{12h}{13}, u_n + \frac{1932h}{2197}K_1 - \frac{7200h}{2197}K_2 + \frac{7296h}{2197}K_3\right) \\
K_5 &= f\left(t_n + h, u_n + \frac{439h}{216}K_1 - 8hK_2 + \frac{3680h}{513}K_3 - \frac{845h}{4104}K_4\right) \\
K_6 &= f\left(t_n + \frac{h}{2}, u_n - \frac{8h}{27}K_1 + 2hK_2 - \frac{3544h}{2565}K_3 + \frac{1859h}{4104}K_4 - \frac{11h}{40}K_5\right)
\end{aligned}
$$

$$
u_{n+1} = u_n + h\left(\frac{16}{135}K_1 + \frac{6656}{12825}K_3 + \frac{28561}{56430}K_4 - \frac{9}{50}K_5 + \frac{2}{55}K_6\right)
$$

$$
\tilde{u}_{n+1} = u_n + h\left(\frac{25}{216}K_1 + \frac{1408}{2565}K_3 + \frac{2197}{4101}K_4 - \frac{1}{5}K_5\right)
$$

多步方法 (Multi-Step Methods)

假设在网格点 $t_0, t_1, \cdots, t_n$ 处已经计算出 $u_0, u_1, \cdots, u_n$，再计算 u_{n+1} 时就可以用前面已有的若干点的值.

$$u(t_{n+1}) = u(t_n) + \int_{t_n}^{t_{n+1}} f(t, u(t))dt,$$

用已经计算得到的前面 3 个网格点处的值

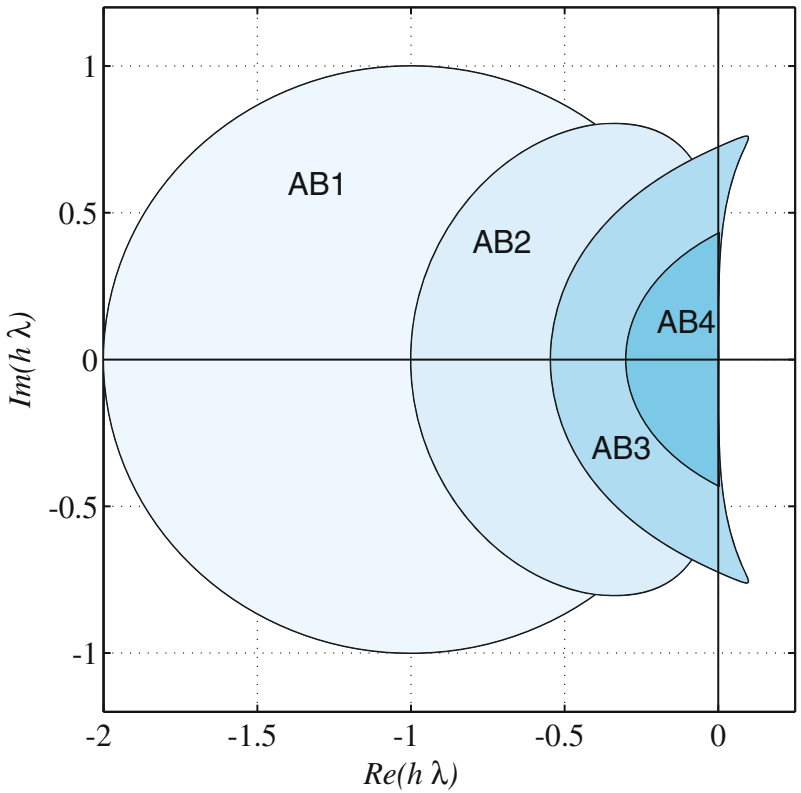
t	t_{n-2}	t_{n-1}	t_n
$f(t,u)$	$f(t_{n-2}, u_{n-2})$	$f(t_{n-1}, u_{n-1})$	$f(t_n, u_n)$

做为插值数据，构造二次插值多项式来代替右端被积函数 $f(t, u(t))$，就得到 Adams-Bashforth 方法 (显式，3 阶精度)：

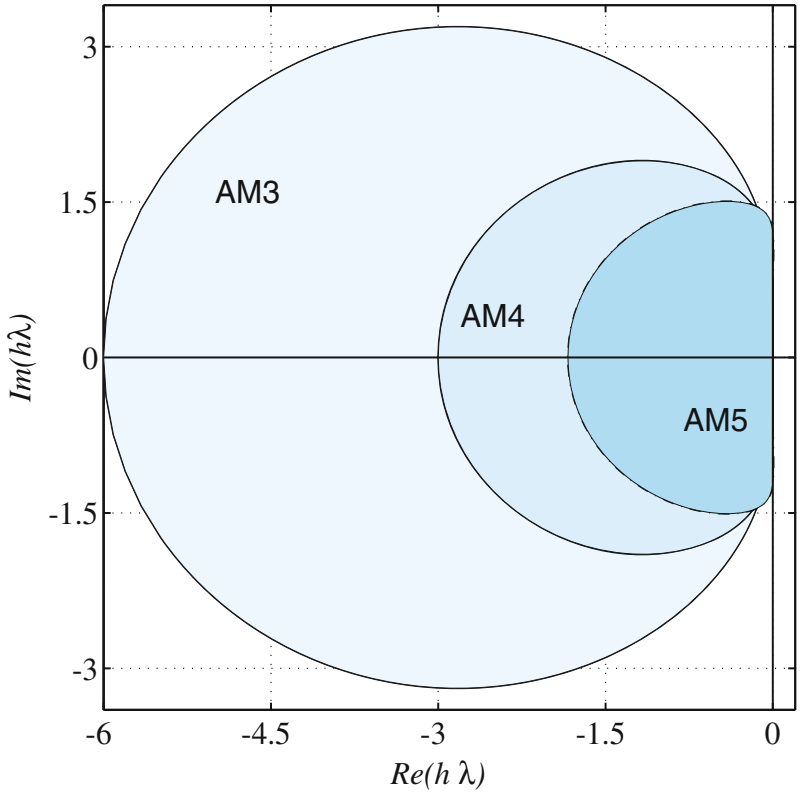
$$u_{n+1} = u_n + \frac{h}{12}[23f(t_n, u_n) - 16f(t_{n-1}, u_{n-1}) + 5f(t_{n-2}, u_{n-2})]$$

用类似做法可以得到 Adams-Moulton 方法 (隐式，4 阶精度)：

$$u_{n+1} = u_n + \frac{h}{24}[9f(t_{n+1}, u_{n+1}) + 19f(t_n, u_n) - 5f(t_{n-1}, u_{n-1}) + f(t_{n-2}, u_{n-2})]$$



AB 多步法的稳定区域.



AM 多步法的稳定区域.

另一种构造隐式多步方法的思路是用

t	t_{n-1}	t_n	t_{n+1}
$u(t)$	u_{n-1}	u_n	u_{n+1}

做为插值数据，构造 Newton 二次插值多项式来代替 $u(t)$，

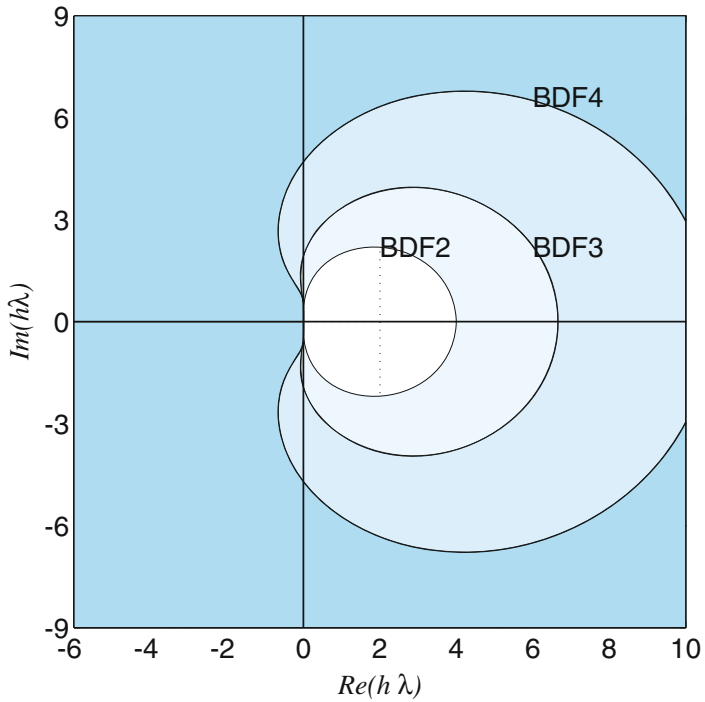
$$u(t) \approx N_2(t) = u_{n+1} + (t - t_{n+1})\frac{u_{n+1} - u_n}{h} + \\ + (t - t_{n+1})(t - t_n)\frac{(u_{n+1} - u_n)/h - (u_n - u_{n-1})/h}{2h}$$

然后求导，并令 $t = t_{n+1}$ 得到：

$$f(t_{n+1}, u(t_{n+1})) \approx N_2'(t_{n+1}) = \frac{u_{n+1} - u_n}{h} + \frac{u_{n+1} - 2u_n + u_{n-1}}{2h}$$

进而得到 Backward Differentiation Formulas (BDF) 方法：

$$u_{n+1} = \frac{4}{3}u_n - \frac{1}{3}u_{n-1} + \frac{2h}{3}f(t_{n+1}, u_{n+1})$$



BDF 多步法的稳定区域.

预估-校正方法 (predictor-corrector method)

隐式方法在绝对稳定性方面有明显的优点，在计算量方面有明显缺点.
以 Crank-Nicolson 方法为例，可以用不动点迭代进行计算

$$u_{n+1}^{(k+1)} = u_n + \frac{h}{2}\left[f(t_n, u_n) + f(t_{n+1}, u_{n+1}^{(k)})\right], \quad k = 0, 1, 2, \ldots$$

可以证明：如果迭代初值 $u_{n+1}^{(0)}$ 选得好，只要迭代一步就可以达到精度要求，即 $u_{n+1}^{(1)}$ 和 u_{n+1} 对 $u(t_{n+1})$ 的逼近阶相同.
自然的想法是用一个足够精度的显式方法计算隐式方法的迭代初值. 这种想法用到 Crank-Nicolson，就是二阶精度的 Heun 方法：

$$\begin{cases} u_{n+1}^* &= u_n + hf(t_n, u_n) \\ u_{n+1} &= u_n + \frac{1}{2}h[f(t_n, u_n) + f(t_{n+1}, u_{n+1}^*)]. \end{cases}$$

计算过程分为两步：第一步称为预估 (prediction)，算出 u_{n+1}^*；第二步称为校正 (correction)，算出 u_{n+1}.
预估-校正方法的绝对稳定性区域比预估步用的显式法大，但比校正步用的隐式法要小.

另一种常用的预估-校正方法为显式多步法与隐式多步法的结合. 例如，用 AB3 与 AM4 结合，就有：

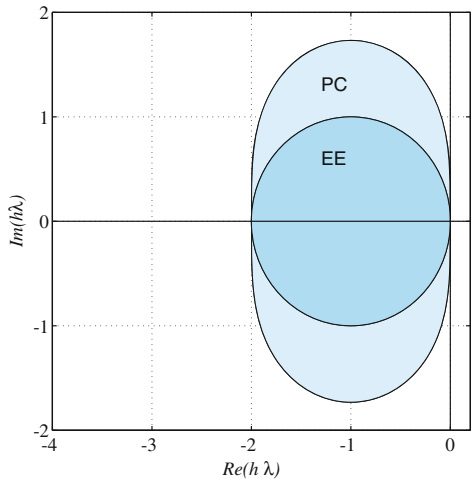
给定 $u_{n-2}, u_{n-1}, u_n, f_{n-2}, f_{n-1}, f_n$

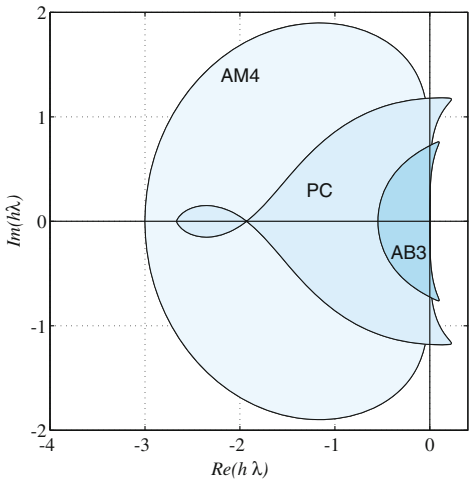
$$u^*_{n+1} = u_n + \frac{h}{12}\left(23f_n - 16f_{n-1} + 5f_{n-2}\right)$$

$$f^*_{n+1} = f(t_{n+1}, u^*_{n+1})$$

$$u_{n+1} = u_n + \frac{h}{24}\left(9f^*_{n+1} + 19f_n - 5f_{n-1} + f_{n-2}\right)$$

$$f_{n+1} = f(t_{n+1}, u_{n+1})$$





一阶 ODE 组的初值问题

$$\begin{cases} \boldsymbol{u}'(t) = \boldsymbol{f}(t, \boldsymbol{u}(t)), \ t \in [t_0, T] \\ \boldsymbol{u}(t_0) = \boldsymbol{a} \end{cases}$$

其中

$$\boldsymbol{u}(t) = \begin{bmatrix} u_1(t) \\ u_2(t) \\ \vdots \\ u_m(t) \end{bmatrix}, \boldsymbol{f}(t, \boldsymbol{u}) = \begin{bmatrix} f_1(t, \boldsymbol{u}) \\ f_2(t, \boldsymbol{u}) \\ \vdots \\ f_m(t, \boldsymbol{u}) \end{bmatrix}, \ \boldsymbol{a} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_m \end{bmatrix}$$

单个一阶 ODE 初值问题的数值方法都可以应用于一阶 ODE 组初值问题，只要按分量实施就行.

考察稳定性的时候，应该用的模型问题为 ($A = [\partial \boldsymbol{f}/\partial \boldsymbol{u}]$)：

$$\begin{cases} \boldsymbol{u}'(t) = A\boldsymbol{u}(t), \ t > t_0, \\ \boldsymbol{u}(t_0) = \boldsymbol{a}, \end{cases}$$

其中 $A \in \mathbb{R}^{m\times m}$ 是常数矩阵，并且有 m 个线性无关的特征向量，记这些特征向量按列排成的矩阵为 T，则有

$$T^{-1}AT = \Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_m), \quad \text{其中 } \lambda_j \text{ 都是 } A \text{ 的特征值.}$$

于是有

$$(T^{-1}\boldsymbol{u})' = (T^{-1}AT)T^{-1}\boldsymbol{u},$$

记 $\boldsymbol{u} = T^{-1}\boldsymbol{u}$，就是

$$\boldsymbol{u}' = \Lambda \boldsymbol{u}.$$

所以稳定性条件只要用单个方程

$$u' = \lambda u,$$

推导，其中 $\lambda = -\max_j\{|\mathrm{Re}(\lambda_j)|\}$.

Runge-Kutta 方法

$$\begin{cases} \boldsymbol{u}^{n+1} &= \boldsymbol{u}^n + h\sum_{i=1}^{s} b_i \boldsymbol{K}^i \\ \boldsymbol{K}^1 &= \boldsymbol{f}(t_n, \boldsymbol{u}^n) \\ \boldsymbol{K}^i &= \boldsymbol{f}\left(t_n + c_i h, \boldsymbol{u}^n + h\sum_{j=1}^{i-1} a_{ij}\boldsymbol{K}^j\right), \ i = 2, \cdots, s. \end{cases}$$

例如，两个方程组成的 ODE 方程组，其 s 级 RK 方法为：

$$\begin{bmatrix} u_1^{n+1} \\ u_2^{n+1} \end{bmatrix} = \begin{bmatrix} u_1^n \\ u_2^n \end{bmatrix} + h\sum_{i=1}^{s} b_i \begin{bmatrix} K_1^i \\ K_2^i \end{bmatrix}$$

$$\begin{bmatrix} K_1^1 \\ K_2^1 \end{bmatrix} = \begin{bmatrix} f_1(t_n, \boldsymbol{u}^n) \\ f_2(t_n, \boldsymbol{u}^n) \end{bmatrix}$$

$$\begin{bmatrix} K_1^i \\ K_2^i \end{bmatrix} = \begin{bmatrix} f_1\left(t_n + c_i h, \boldsymbol{u}^n + h\sum_{j=1}^{i-1} a_{ij}\boldsymbol{K}^j\right) \\ f_2\left(t_n + c_i h, \boldsymbol{u}^n + h\sum_{j=1}^{i-1} a_{ij}\boldsymbol{K}^j\right) \end{bmatrix}, \ i = 2, \cdots, s.$$

高阶 ODE 初值问题

m 阶 ODE 的初值问题

$$\begin{cases} u^{(m)}(t) = f(t, u(t), u'(t), \ldots, u^{(m-1)}(t)), \\ u(t_0) = a_1, \\ u'(t_0) = a_2, \\ \vdots \\ u^{(m-1)}(t_0) = a_m, \end{cases}$$

可化为等价的一阶 ODE 组的初值问题.

记 $u_1(t) = u(t), u_2(t) = u'(t), u_3(t) = u''(t), \cdots, u_m(t) = u^{(m-1)}(t)$,

$$
\begin{cases}
u_1'(t) = u_2(t), \\
u_2'(t) = u_3(x), \\
\vdots \\
u_{m-1}'(t) = u_m(t), \\
u_m'(t) = f(t, u_1(t), u_2(t), \ldots u_m(t)), \\
u_1(t_0) = a_1, \\
u_2(t_0) = a_2, \\
\vdots \\
u_m(t_0) = a_m.
\end{cases}
$$

从而一阶 ODE 组初值问题的数值方法都可以应用到高阶 ODE 的初值问题的数值计算, 而且还能同时计算出解 $u(t)$ 的各阶导函数的近似值.

ODEs modeling of plant physiology

```
function f = hires(t,y)
%
% f = hires(t,y)
%
% High irradiance response function arising in plant physiology
f = y;
f(1) = -1.71*y(1) + .43*y(2) + 8.32*y(3) + .0007;
f(2) = 1.71*y(1) - 8.75*y(2);
f(3) = -10.03*y(3) + .43*y(4) + .035*y(5);
f(4) = 8.32*y(2) + 1.71*y(3) - 1.12*y(4);
f(5) = -1.745*y(5) + .43*y(6) + .43*y(7);
f(6) = -280*y(6)*y(8)+.69*y(4)+1.71*y(5)-.43*y(6)+.69*y(7);
f(7) = 280*y(6)*y(8) - 1.81*y(7);
f(8) = -280*y(6)*y(8) + 1.81*y(7);

y0 = [1,0,0,0,0,0,0,.0057];
[t,y] = ode45(@hires,[0 322],y0);
plot(t,y,'LineWidth',1)
legend('y_1','y_2','y_3','y_4','y_5','y_6')
```

刚性 (stiff) 问题

下面这个 ODE 初值问题：

$$\begin{cases} u'(t) = -100u(t) + 10, & t > 0, \\ u(0) = 1, \end{cases}$$

准确解为

$$u(t) = \frac{1}{10} + \frac{9}{10}e^{-100t}, \quad t \geq 0.$$

解由两部分组成，一部分变化很快，一部分变化很慢.
如果用显式 Euler 方法求解，网格步长 h 由变化快的部分决定，

$$|1 - 100h| \leq 1 \Rightarrow h \leq 1/50$$

但 e^{-100t} 很快就衰减没了，如果要计算到比较长的时间，比如 $t > 10$ 以后，解的变化已经很缓慢，但是仍然要使用很小的网格步长，造成计算非常慢！这种现象就称为 “stiff”，中文译为” 刚性”.

再看方程组情形.

$$u'(t) = Au(t) + \varphi(t), \quad A \in \mathbb{R}^{m\times m}, \varphi(t) \in \mathbb{R}^m$$

其中 A 有 m 个不同特征值 $\lambda_j, \mathrm{Re}(\lambda_j) < 0, j = 1, \ldots, m$. 准确解为

$$u(t) = \sum_{j=1}^{m} C_j e^{\lambda_j t} v_j + \psi(t),$$

其中 C_j 是常数，$\{v_j\}$ 是特征向量并且构成 $\mathbb{R}^m$ 的基，$\psi(t)$ 是一个特解. 如果特征值的模差别很大，就是刚性问题. 例如，

$$\begin{cases} u'(t) = -20u(t) - 19v(t) \\ v'(t) = -19u(t) - 20v(t) \end{cases} \quad \begin{cases} u(0) = 2 \\ v(0) = 0 \end{cases}$$

准确解为

$$\begin{cases} u(t) = e^{-39t} + e^{-t}, \\ v(t) = e^{-39t} - e^{-t}, \end{cases} \quad t \geq 0$$

用显式 Euler 法

$$\begin{cases} u_{n+1} = u_n + h(-20u_n - 19v_n) \\ v_{n+1} = v_n + h(-19u_n - 20v_n) \end{cases} \quad \begin{cases} u_0 = 2 \\ v_0 = 0 \end{cases}$$

格式的解为

$$\begin{cases} u_n = (1-39h)^n + (1-h)^n, \\ v_n = (1-39h)^n - (1-h)^n, \end{cases} \quad n = 0, 1, 2, \ldots,$$

必须满足

$$\begin{cases} |1-39h| < 1 \\ |1-h| < 1 \end{cases} \quad \text{即} \quad h < \frac{2}{39}$$

才能稳定. 写成矩阵-向量形式:

$$\mathbf{u}'(t) = A\mathbf{u}, \quad A = \begin{pmatrix} -20 & -19 \\ -19 & -20 \end{pmatrix}$$

$$\lambda_1(A) = -1, \quad \lambda_2(A) = -39, \quad \mathrm{cond}(A) = 39$$

用隐式 Euler 法解这个方程组

$$\begin{cases} u_{n+1} = u_n + h(-20u_{n+1} - 19v_{n+1}) \\ v_{n+1} = v_n + h(-19u_{n+1} - 20v_{n+1}) \end{cases}$$

写成向量形式

$$\begin{aligned} & \mathbf{u}_{n+1} = \mathbf{u}_n + hA\mathbf{u}_{n+1} \\ \Rightarrow\ & (I - hA)\mathbf{u}_{n+1} = \mathbf{u}_n \\ \Rightarrow\ & \mathbf{u}_{n+1} = (I - hA)^{-1}\mathbf{u}_n \end{aligned}$$

取 2 范数

$$\|\mathbf{u}_{n+1}\|_2 = \|(I - hA)^{-1}\mathbf{u}_n\|_2 \le \|(I - hA)^{-1}\|_2 \|\mathbf{u}_n\|_2$$

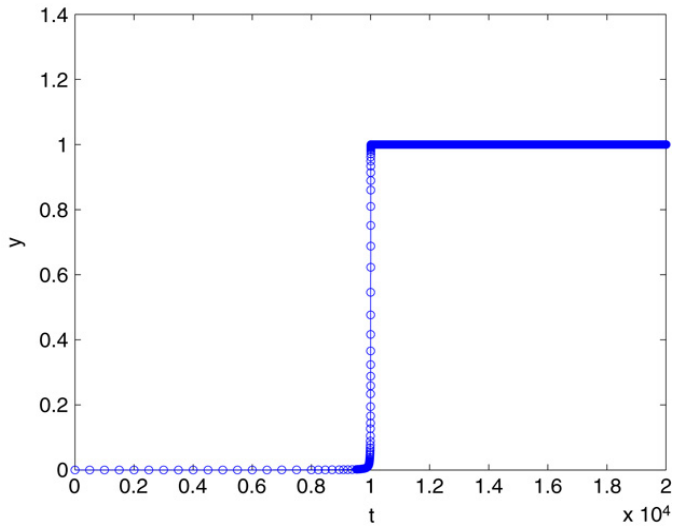
其中

$$\|(I - hA)^{-1}\|_2 = \max_j |\lambda_j((I - hA)^{-1})| = \max_j \frac{1}{|1 - \lambda_j(A)|}$$

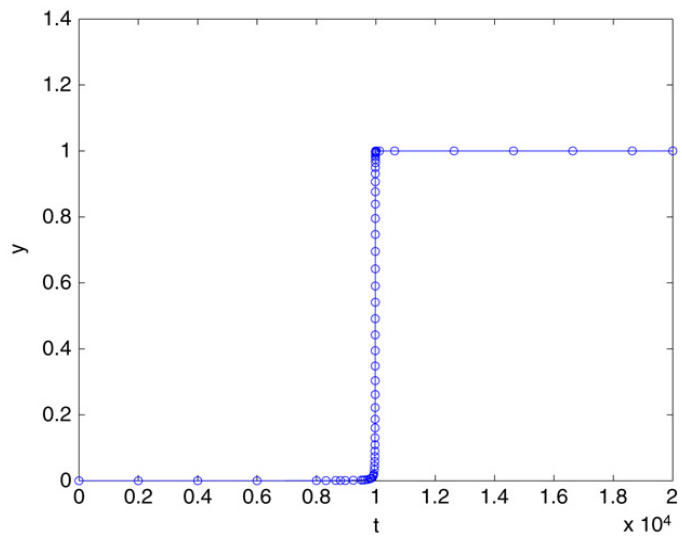
刚性问题应该用绝对稳定的格式计算.

决定网格步长 h 的因素：(1) 方法的精度，(2) 稳定性条件.
如果一个 ODE 初值问题，用显式方法求解时，由稳定性条件决定的网格步长远远小于由方法精度决定的网格步长，那这个问题就是刚性问题.
解刚性问题，应该采用稳定性区域包含整个左半平面的隐式方法. 例如，长时间非线性问题：

$$u' = u^2(1-u), \quad u(0) = 0.0001, \quad t \in [0, 20000]$$



用 ode45 解.



用 ode23s 解.

化学动力学 (chemical kinetics)

$$\begin{cases} u_1' = \dfrac{1}{\varepsilon}\left(-u_1 + \dfrac{u_2}{1+u_2}\right) - \dfrac{u_2}{(1+u_2)^2}, & t \in (0, 10], \\ u_2' = -u_2, & t \in (0, 10], \\ u_1(0) = 1.5, \\ u_2(0) = 1, \end{cases}$$

其中 $\varepsilon > 0$ 是参数，ε 越小问题越 stiff.

ε	ode23	ode45	ode23s	ode15s
10^{-2}	409	1241	73	73
10^{-3}	3991	12081	84	81
10^{-4}	39808	120553	87	85

```
function [f]= funds(t,u)
epsilon =1.e-6; [n,m]= size (u);f=zeros(n,m);
f(1)= -1/ epsilon *u(1)+( (1/ epsilon -1)* u(2)+...
1/ epsilon *u(2)* u(2))/(1+ u(2))^2;
f(2)=-u(2);
end


>> u0 =[1.5 ,1]; tspan =[0 ,10];
>> tic, [t,u]= ode23s(@funds ,tspan ,u0); toc
时间已过 0.129486 秒。

>> tic, [t,u]= ode45(@funds ,tspan ,u0); toc
时间已过 137.316671 秒。
```

Lotka–Volterra predator prey system

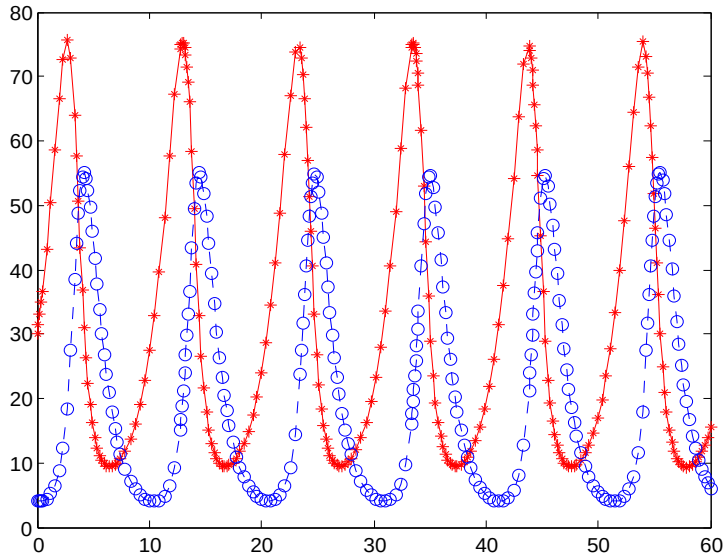
这个 ODE 组描述捕食者 (predator) 与猎物 (prey) 族群规模的此消彼长过程.

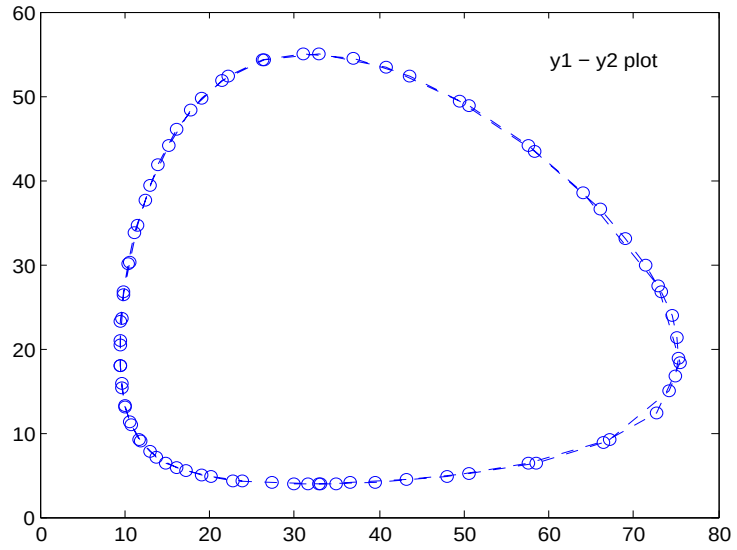
$$\begin{cases} y_1' = y_1(a - by_2) \\ y_2' = y_2(cy_1 - r) \\ y_1(0) = y_{10} \\ y_2(0) = y_{20} \end{cases}$$

方程组中的 y_1 是猎物 (比如兔子) 的数量，y_2 是捕食者（比如狼）的数量. 参数和初值为：

$$\begin{aligned} a &= 0.5471, \quad & b &= 0.0281, \\ r &= 0.8439, \quad & c &= 0.0266, \\ y_{10} &= 30, \quad & y_{20} &= 4. \end{aligned}$$

```
function yprime = lv(t,y)
% LV: Contains Lotka-Volterra equations
a = 0.5471; b = 0.0281; c = 0.0266; r = 0.8439;
yprime = [a*y(1)-b*y(1)*y(2);-r*y(2)+c*y(1)*y(2)];


[t,y]=ode45(@lv,[0  60],[30;4]);
 figure; plot(t,y(:,1),'r*-',t,y(:,2),'bo-')
 figure; plot(y(:,1),y(:,2),'bo - ')
```





The Van der Pol equation

下面二阶 ODE 是自振荡效应的一个数学模型.

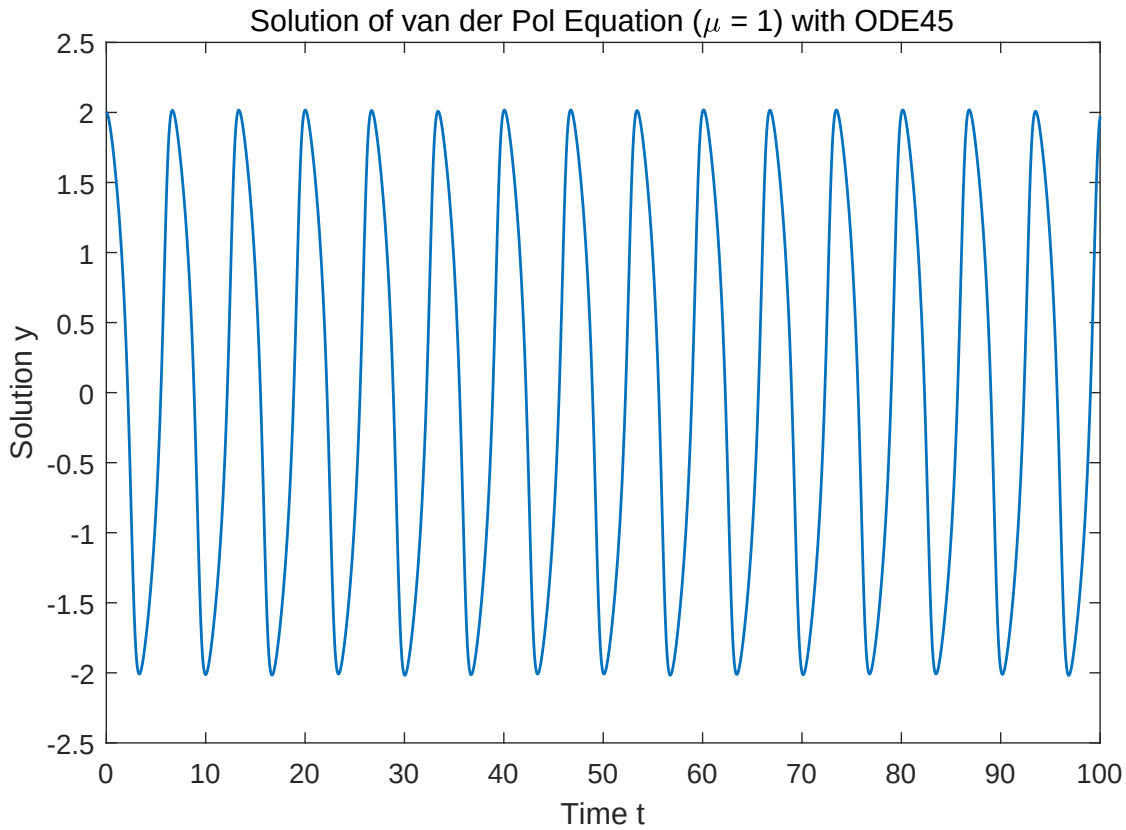
$$y_1'' - \mu(1 - y_1^2)y_1' + y_1 = 0, \quad \mu > 0 \text{是参数.}$$

令 $y_2 = y_1'$，将二阶方程写成方程组

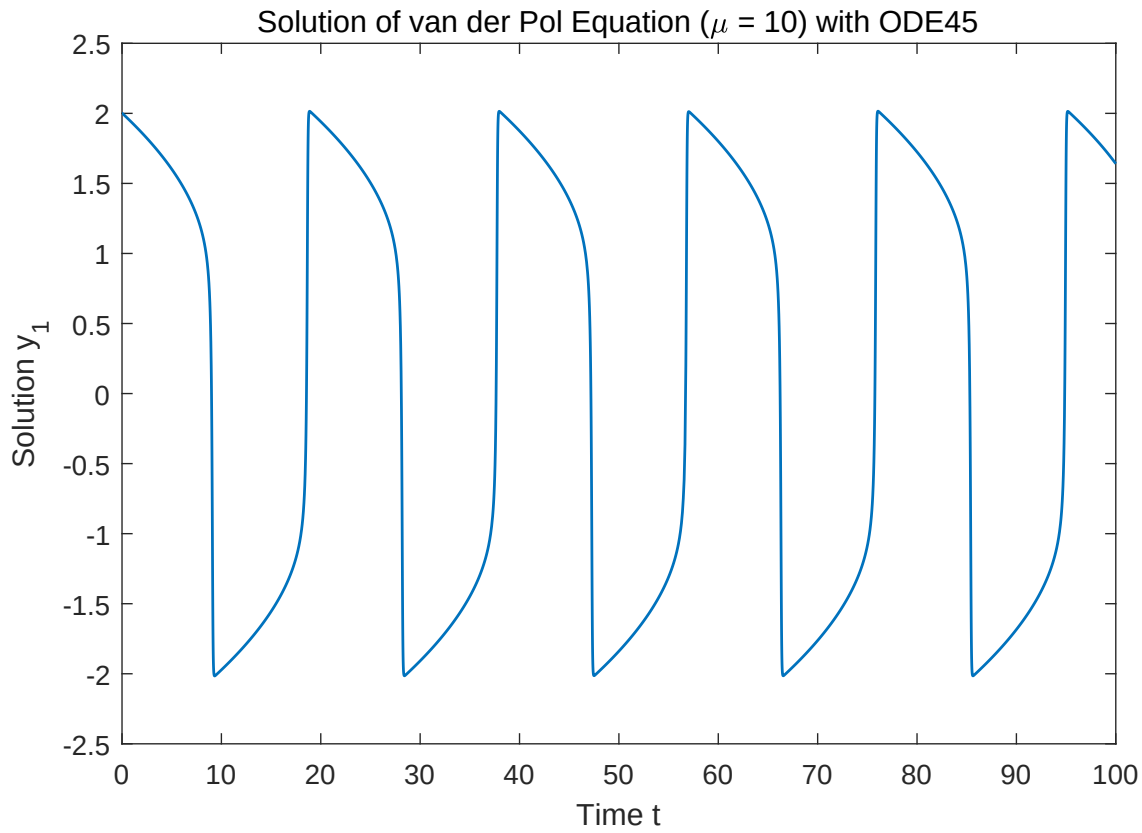
$$\begin{cases} y_1' = y_2 \\ y_2' = \mu(1 - y_1^2)y_2 - y_1 \end{cases}$$

```
function dydt = vdp1(t,y)
% VDP1  Evaluate the van der Pol ODEs for mu = 1
%
%   See also ODE113, ODE23, ODE45.

%   Jacek Kierzenka and Lawrence F. Shampine
%   Copyright 1984-2014 The MathWorks, Inc.

dydt = [y(2); (1-y(1)^2)*y(2)-y(1)];
```

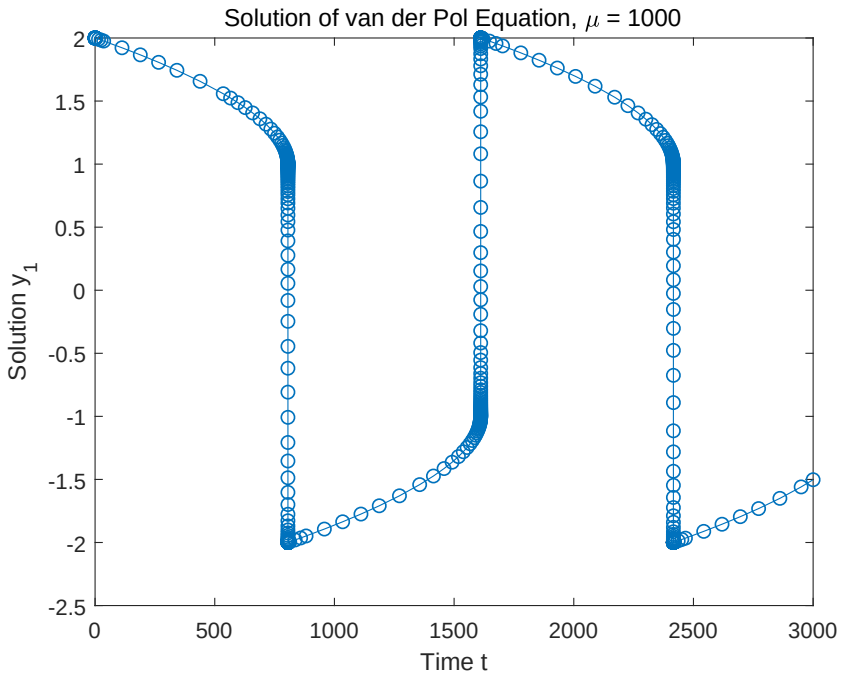
```
[t,y] = ode45(@vdp1,[0  100],[2;  0]);
plot(t,y(:,1))
title('Solution of van der Pol Equation (\mu = 1) with ODE45')
xlabel('Time t');
ylabel('Solution y_1');
```



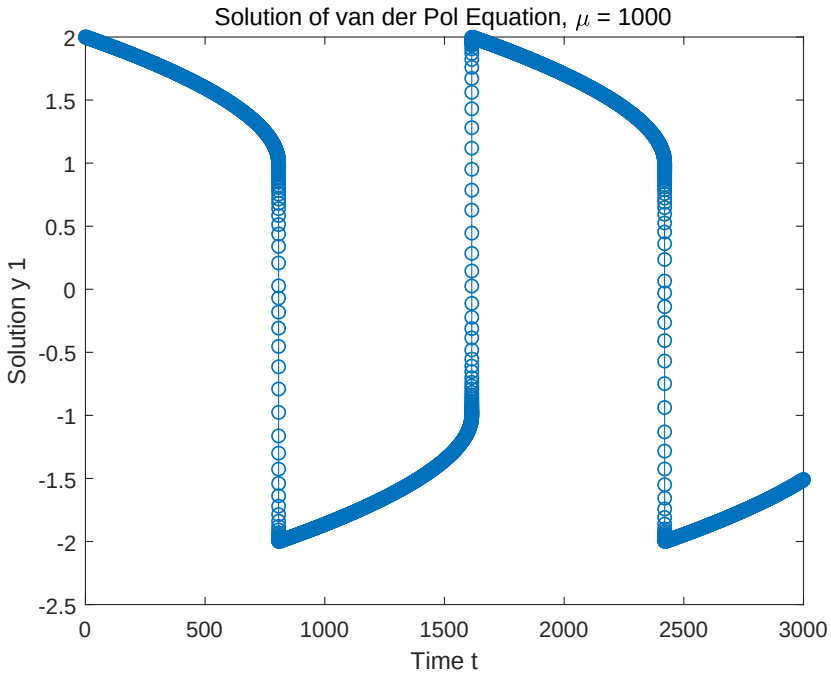
```
[t,y] = ode45(@vdp10,[0  100],[2;  0]);
plot(t,y(:,1))
title('Solution of van der Pol Equation (\mu = 10) with ODE45')
xlabel('Time t');
ylabel('Solution y_1');
```



```
[t,y] = ode15s(@vdp1000,[0 3000],[2; 0]);
plot(t,y(:,1),'-o');
title('Solution of van der Pol Equation, \mu = 1000');
xlabel('Time t');
ylabel('Solution y_1');
```

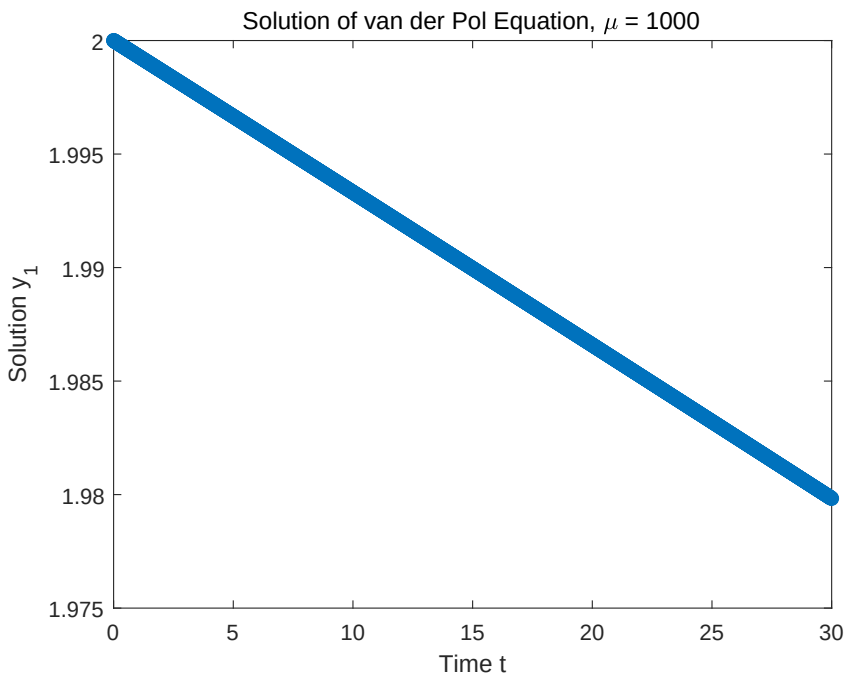


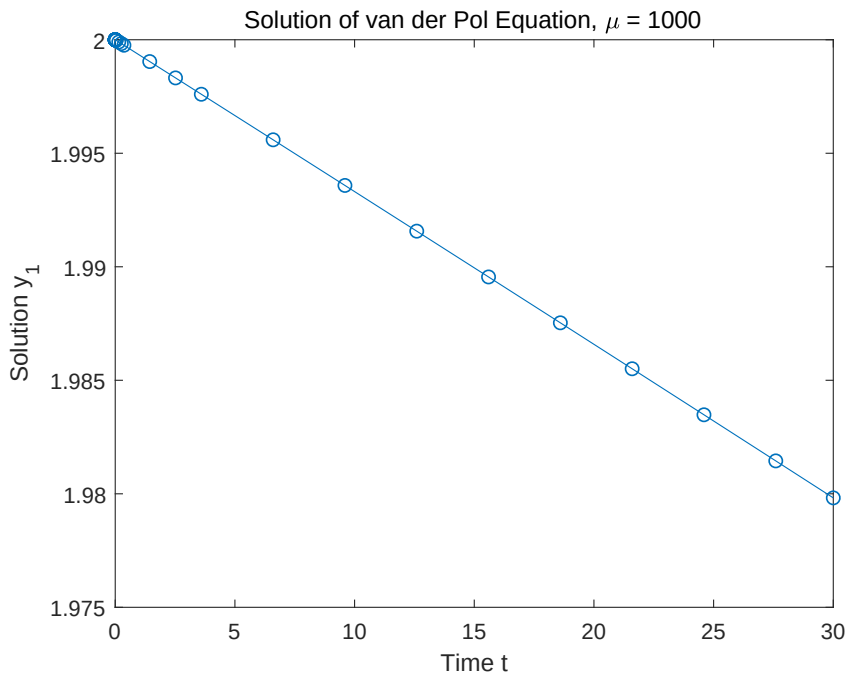
用 ode15s 解，0.106692 秒.



用 ode45 解，55.481266 秒.

```
[t,y] = ode45(@vdp1000,[0  30],[2;  0]);
plot(t,y(:,1),'-o');
title('Solution of van der Pol Equation, \mu = 1000');
xlabel('Time t');
ylabel('Solution y_1');
```





二阶线性 ODE 的边值问题

考虑二阶 ODE 的两点边值问题：

$$\begin{cases} u''(x) = f(x, u(x), u'(x)), \ x \in (a, b) \\ u(a) = \alpha, \quad u(b) = \beta \end{cases} \tag{9.1}$$

其中 $f(x, u, v) : [a, b] \times \mathbb{R} \times \mathbb{R} \longmapsto \mathbb{R}$ 是已知函数.

例如，二阶线性 ODE 的两点边值问题：

$$\begin{cases} u''(x) + p(x)u'(x) + q(x)u(x) = f(x), \ x \in (a, b) \\ u(a) = \alpha, \quad u(b) = \beta \end{cases} \tag{9.2}$$

本节的内容就是讨论如何求上述两点边值问题的数值解.

两点边值问题的差分方法

我们看一个极端简单的例子：

$$\begin{cases} -u''(x) = f(x),\ x \in (a,b) \\ u(a) = \alpha, \quad u(b) = \beta \end{cases} \tag{9.3}$$

第一步：把区间 $[a,b]$ 离散化. 给定 $n \in \mathbb{N}_+$，定义

$$h = \frac{b-a}{n+1}, \quad x_j = a + jh, \quad j = 0, 1, 2, \ldots, n+1$$

这些离散点 $x_j \in [a,b]$ 称为网格点.

第二步：在每个网格点 x_j 处把 ODE 离散化，由于 $u(x) \in C^4[a,b]$ 时有

$$\frac{u(x_{j+1}) - 2u(x_j) + u(x_{j-1})}{h^2} = u''(x_j) + \frac{h^2}{12}u^{(4)}(x_j) + \mathcal{O}(h^3)$$

所以在网格点 x_j 处，两点边值问题(9.3)就可以写成

$$-\frac{u(x_{j+1})-2u(x_j)+u(x_{j-1})}{h^2}=f(x_j)+\mathcal{O}(h^2),\quad j=1,2,\ldots,n$$

忽略右端的误差项，就得到差分格式：

$$-\frac{u_{j+1}-2u_j+u_{j-1}}{h^2}=f(x_j),\quad j=1,2,\ldots,n \tag{9.4}$$

其中 $u_0=\alpha,\ u_{n+1}=\beta$. 通过引进向量：

$$\begin{aligned} u&=(u_1,\ldots,u_n)^{\mathrm{T}}\\ b&=(h^2f(x_1)+\alpha,\ h^2f(x_2),\ \ldots,\ h^2f(x_n)+\beta)^{\mathrm{T}}\end{aligned}$$

两点边值问题(9.3)的差分格(9.4)可以写成线性方程组

$$Au=b \tag{9.5}$$

其中的系数矩阵为

$$A = \begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{bmatrix} \in \mathbb{R}^{n,n} \tag{9.6}$$

这是一个实对称正定矩阵，所以线性方程组(9.5)有惟一解. 而且(9.5)是一个三对角线性方程组，可以用 Thomas 算法快速求解.

定理

如果两点边值问题(9.3)中的函数$f(x) \in C([a,b])$，则差分方法(9.4)的解就有误差估计

$$\max_{j=0,\ldots,n+1} |u_j - u(x_j)| \leq \frac{(b-a)^2 h^2}{96} \max_{x\in[a,b]} |f(x)| \tag{9.7}$$

两点边值问题的 Galerkin 方法

考虑二阶常微分方程两点边值问题：

$$\begin{cases} -u''(x) + q(x)u(x) = f(x), \quad a < x < b, \\ u(a) = 0, \quad u(b) = 0, \end{cases} \tag{9.8}$$

其中 $q(x),\ f(x) \in C[a,b]$ 已知，而且 $q(x) \geqslant 0$.
我们要在由 n 个线性无关的基函数

$$\varphi_1(x),\ \varphi_2(x), \cdots, \varphi_n(x), \quad \varphi_j(a) = \varphi_j(b) = 0,$$

生成的线性空间中近似求解问题(9.8)，近似解 $U(x)$ 可以表示为

$$U(x) = \sum_{j=1}^{n} c_j\, \varphi_j(x) \tag{9.9}$$

显然，只要设法确定上式中的系数 c_j，就可得到近似解 $U(x)$.

基函数 $\varphi_j(x)$ 可以取为三角函数

$$\varphi_j(x) = \sin\left(j\pi\frac{x-a}{b-a}\right), \quad j = 1, 2, \ldots, n,$$

或者多项式

$$\varphi_j(x) = (x-a)(b-x)x^j, \quad j = 1, 2, \ldots, n.$$

将近似解 $U(x)$ 的表达式(9.9)代入常微分方程(9.8)，两边未必相等，而是有一个残量，其表达式为

$$R(x\,;c) = -\sum_{j=1}^{n} c_j\,\varphi_j''(x) + q(x)\sum_{j=1}^{n} c_j\,\varphi_j(x) - f(x). \tag{9.10}$$

如果我们能找到 $c = (c_1, c_2, \ldots, c_n)^T \in \mathbb{R}^n$，使得

$$R(x\,;c) = 0, \quad \forall\, x \in [a,\, b]$$

则 $U(x)$ 就是两点边值问题(9.8)的准确解. 但这一般是做不到的!

我们可以在 $\mathbb{R}^n$ 中选择 c 使得 $R(x\,;c)$ 尽量小. 一个自然的想法是用积分意义下的最小二乘法. 设目标函数为

$$F(c) = \int_a^b |R(x\,;c)|^2\,dx$$

问题变成：求 $c^* \in \mathbb{R}^n$，使得

$$F(c^*) = \min_{c\in\mathbb{R}^n} F(c).$$

这个最小二乘问题的法方程组为

$$\begin{cases} \dfrac{\partial F}{\partial c_1} = 0 \\ \dfrac{\partial F}{\partial c_2} = 0 \\ \vdots \\ \dfrac{\partial F}{\partial c_n} = 0 \end{cases} \tag{9.11}$$

其中

$$\begin{aligned}\frac{\partial F}{\partial c_i} &= 2\int_a^b R(x;c)\frac{\partial R}{\partial c_i}\,dx \\ &= 2\int_a^b R(x;c)\big[-\varphi_i''(x)+q(x)\varphi_i(x)\big]\,dx \\ &= 2\sum_{j=1}^n c_j\int_a^b\big(-\varphi_j''+q\varphi_j\big)\big(-\varphi_i''+q\varphi_i\big)dx - 2\int_a^b f\big(-\varphi_i''+q\varphi_i\big)\,dx\end{aligned}$$

由此可以看出法方程组(9.11)是一个关于 $c_1, c_2, \ldots, c_n$ 的线性代数方程组

$$Ac = b$$

其中

$$A = [a_{ij}] \in \mathbb{R}^{n,n}, \quad a_{ij} = \int_a^b \big(-\varphi_j''+q\varphi_j\big)\big(-\varphi_i''+q\varphi_i\big)\,dx,$$

$$b = (b_1, \ldots, b_n)^T \in \mathbb{R}^n, \quad b_i = \int_a^b f\big(-\varphi_i''+q\varphi_i\big)\,dx.$$

可以选择合适的数值积分公式计算 a_{ij} 和 b_i，然后再用适当的方法求解这个线性代数方程组就得到了两点边值问题 (9.8)的近似解.

上述方法是对残量函数进行最小二乘导出的. 最后得到的关于待定展开系数 $c_1, c_2, \ldots, c_n$ 的方程组为

$$\int_a^b R(x\,;c)[-\varphi_i''(x) + q(x)\varphi_i(x)]\,dx = 0, \quad i = 1, 2, \ldots, n$$

更一般地，可以从

$$\int_a^b R(x\,;c)W(x)\,dx = 0,$$

导出 $c_1, c_2, \ldots, c_n$ 满足的方程，其中函数 $W(x)$ 的不同取法可导致不同的近似解法. 显然，取

$$W(x) = -\varphi_i''(x) + q(x)\varphi_i'(x), \quad i = 1, 2, \ldots, n$$

就是上述最小二乘法.

Galerkin 方法

所谓 **Galerkin 方法**，就是让残量 $R(x\,;c)$ 与每个基函数在积分内积的意义下都正交

$$\int_a^b R(x\,;c)\varphi_i(x)\,dx = 0, \quad i = 1, 2, \ldots, n.$$

代入 $R(x\,;c)$ 的表达式(9.10)，得

$$\sum_{j=1}^{n} \Big[\int_a^b \big(-\varphi_j''(x)\varphi_i(x) + q(x)\varphi_j(x)\varphi_i(x) \big)\,dx \Big] c_j = \int_a^b f(x)\varphi_i(x)\,dx,$$

再利用分部积分公式

$$-\int_a^b \varphi_j''(x)\varphi_i(x)\,dt = -\varphi_j'(x)\varphi_i(x)\Big|_a^b + \int_a^b \varphi_j'(x)\varphi_i'(x)\,dx$$

和 $\varphi_i(a) = \varphi_i(b) = 0$，有

$$\sum_{j=1}^{n} c_j \int_a^b \big(\varphi_j'\varphi_i' + q\,\varphi_j\varphi_i\big)dx = \int_a^b f\,\varphi_i\,dx, \quad i = 1, 2, \ldots, n.$$

求解两点边值问题(9.8)的 Galerkin 方法的线性代数方程组为

$$Ac = b$$

其中

$$A = [a_{ij}] \in \mathbb{R}^{n,n}, \quad a_{ij} = \int_a^b \left(\varphi_j' \varphi_i' + q\, \varphi_j \varphi_i\right) dx$$

$$b = (b_1, b_2, \ldots, b_n)^T \in \mathbb{R}^n, \quad b_i = \int_a^b f\, \varphi_i\, dx$$

它们分别称为两点边值问题(9.8)的刚度矩阵与载荷向量.

- Galerkin 方法并没有将定解区域 $[a, b]$ 离散化，但已经把微分方程求解问题化成线性代数方程组求解问题.
- Galerkin 方法得到的线性方程组的系数矩阵 A 是稠密而且病态的.
- 整体的基函数对于多维区域很难满足事先给定的边界条件.

有限元方法 = Galerkin 方法 + 分片低次多项式基函数

在选取基函数之前，先在定解区域 $[a,b]$ 中给定 $n+1$ 个网格点

$$a = x_0 < x_1 < \cdots < x_{n+1} = b$$

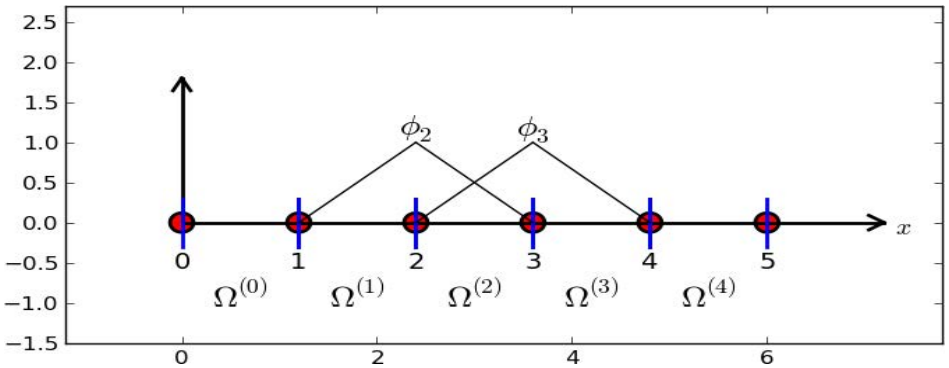
它们把 $[a,b]$ 剖分为 n 个小区间，每个小区间 $(x_i,\, x_{i+1})$ 称为“**单元**”，记

$$h_i = x_{i+1} - x_i, \quad h = \max_{0 \leqslant i \leqslant n} (x_{i+1} - x_i).$$

在此基础上，取基函数 $\varphi_i(x)$ 满足：

- 在每个小区间 $[x_i, x_{i+1}]$ 上为 k 次多项式；
- 在整个 $[a,b]$ 区间上连续；
- 在 $x = a,\ x = b$ 处函数值为零.

最简单的情况是 $k = 1$，此时，基函数 $\varphi_i(x)$ 为分段线性函数.



$$\varphi_i(x)=\begin{cases}0, & x\in[a,x_{i-1}],\\ \dfrac{x-x_{i-1}}{h_{i-1}}, & x\in[x_{i-1},x_i],\\ \dfrac{x_{i+1}-x}{h_i}, & x\in[x_i,x_{i+1}],\\ 0, & x\in[x_{i+1},b],\end{cases}\qquad i=1,2,\ldots,n, \tag{9.12}$$

$$\varphi_i'(x)=\begin{cases}0, & x\in[a,x_{i-1}),\\ \dfrac{1}{h_{i-1}}, & x\in[x_{i-1},x_i),\\ -\dfrac{1}{h_i}, & x\in[x_i,x_{i+1}),\\ 0, & x\in[x_{i+1},b],\end{cases}\qquad i=1,2,\ldots,n. \tag{9.13}$$

由于 φ_i 和 φ_i' 只在区间 (x_{i-1},x_{i+1}) 上非零，所以除 $j=i-1,\ i,\ i+1$ 之外，总有

$$\varphi_i(x)\varphi_j(x)\equiv 0,\quad \varphi_i'(x)\varphi_j'(x)\equiv 0.$$

此时用 Galerkin 方法导出的线性代数方程组的系数矩阵 A 是一个对称正定的三对角矩阵，其非零元素为

$$
\begin{aligned}
a_{ii} &= \int_a^b \left[(\varphi_i')^2 + q(\varphi_i)^2\right] dx = \int_{x_{i-1}}^{x_{i+1}} \left[(\varphi_i')^2 + q(\varphi_i)^2\right] dx \\
&= \frac{1}{h_{i-1}} + \frac{1}{h_i} + \frac{1}{h_{i-1}^2} \int_{x_{i-1}}^{x_i} (x - x_{i-1})^2 q(x)\, dx + \\
&\quad + \frac{1}{h_i^2} \int_{x_i}^{x_{i+1}} (x_{i+1} - x)^2 q(x)\, dx, \\
a_{i,i+1} &= \int_a^b \left(\varphi_i' \varphi_{i+1}' + q\varphi_i \varphi_{i+1}\right) dx \\
&= -\frac{1}{h_i} + \frac{1}{h_i^2} \int_{x_i}^{x_i+1} (x_{i+1} - x)(x - x_i) q(x) dx \\
a_{i,i-1} &= \int_a^b \left(\varphi_i' \varphi_{i-1}' + q\varphi_i \varphi_{i-1}\right) dx \\
&= -\frac{1}{h_{i-1}} + \frac{1}{h_{i-1}^2} \int_{x_{i-1}}^{x_i} (x_i - x)(x - x_{i-1}) q(x)\, dx,
\end{aligned}
$$

右端向量 b 的分量为

$$b_i = \int_a^b f\varphi_i \, dx = \frac{1}{h_{i-1}} \int_{x_{i-1}}^{x_i} (x - x_{i-1}) f(x) dx + \frac{1}{h_i} \int_{x_i}^{x_{i+1}} (x_{i+1} - x) f(x) dx.$$

可以看到，若令

$$Q_{1,i} = \frac{1}{h_i^2} \int_{x_i}^{x_{i+1}} (x_{i+1} - x)(x - x_i) q(x) \, dx, \quad i = 1, 2, \ldots n-1,$$

$$Q_{2,i} = \frac{1}{h_{i-1}^2} \int_{x_{i-1}}^{x_i} (x - x_{i-1})^2 q(x) \, dx, \quad i = 1, 2, \ldots n,$$

$$Q_{3,i} = \frac{1}{h_i^2} \int_{x_i}^{x_{i+1}} (x_{i+1} - x)^2 q(x) \, dx, \quad i = 1, 2, \ldots n,$$

$$Q_{4,i} = \frac{1}{h_{i-1}} \int_{x_{i-1}}^{x_i} (x - x_{i-1}) f(x) dx, \quad i = 1, 2, \ldots n,$$

$$Q_{5,i} = \frac{1}{h_i} \int_{x_i}^{x_{i+1}} (x_{i+1} - x) f(x) dx, \quad i = 1, 2, \ldots n,$$

则有

$$
\begin{aligned}
a_{ii} &= Q_{2,i} + Q_{3,i} + 1/h_{i-1} + 1/h_i, \quad i = 1, 2, \ldots n, \\
a_{i,i+1} &= Q_{1,i} - 1/h_i, \quad i = 1, 2, \ldots n-1, \\
a_{i,i-1} &= Q_{1,i-1} - 1/h_{i-1}, \quad i = 2, 3, \ldots n, \\
b_i &= Q_{4,i} + Q_{5,i}, \quad i = 1, 2, \ldots n.
\end{aligned}
$$

这样，在形成线性代数方程组的系数矩阵 A 和右端向量 b 时，需要计算的积分共有 5 种. 这种先将定解区域剖分成一些小的单元，然后用分段低次多项式构成的基函数与 Galerkin 方法相结合导出的微分方程数值求解方法，就是**有限元方法**.

定理

如果准确解 $u(x) \in C^2([a,b])$，则存在一个常数 $C > 0$ 使得

$$
\int_a^b |U(x) - u(x)|^2 \, dx \le CMh^4, \quad M = \int_a^b |u''(x)|^2 \, dx
$$

与差分方法相比，有限元方法的优点主要体现在多维问题中，它可以很好地离散定解区域不规则的问题.

参考文献

1. 薛定宇, 陈阳泉
   高等应用数学问题的 Matlab 求解 (第三版)
   清华大学出版社, 2013.
2. Michael T. Heath,
   Scientific Computing An Introductory Survey (revised 2nd Edition)
   SIAM, 2018.
3. Alfio Quarteroni, Riccardo Sacco, Fausto Saleri,
   Numerical Mathematics(2nd Edition),
   Springer, 2007.
4. Walter Gander, Martin J. Gander, Felix Kwok
   Scientific Computing: an introduction using Maple and MATLAB
   Texts in Computational Science and Engineering 11
   Springer 2007.