第八章　常微分方程数值解

微分方程数值解

- 在科学研究或工程领域中，有许多数学模型都是通过微分方程来描述的，求解微分方程是非常重要的、关键的问题
- 微分方程按自变量的个数可分为：
  - 常微分方程
  - 偏微分方程
- 微分方程按定解条件可分为：
  - 初值问题
  - 边值问题

微分方程数值解

- 研究微分方程解析解的学科：数学物理方程，但是绝大部分的微分方程是没有解析解的
- 数值求解微分方程没有统一的算法，针对不同类型的微分方程，需要设计特定的算法
- 目前，研究数值求解微分方程的方法是热门的课题，正在迅速发展之中，常见的方法：
  - 有限差分法
  - 有限元方法
  - 有限体积法
  - 边界元方法
  - 谱方法
  - …

微分方程数值解

- 常微分方程的初值问题：

$$\begin{cases} \dfrac{dy}{dx} = f(x,y) \quad , \quad x \in [a,b] \\ y(a) = y_0 \end{cases}$$

- 为了使解存在唯一，一般需要对函数 $f(x,y)$ 加限制条件
- （初值问题解的存在唯一性）若函数 $f(x,y)$ 在条带 $a \le x \le b,\ -\infty < y < \infty$ 上连续，且满足Lipschitz条件，即 $|f(x,y_1) - f(x,y_2)| \le L|y_1 - y_2|$，则初值问题的解在区间 $[a,b]$ 上存在且唯一

微分方程数值解

- 常微分方程的解是一个函数，但是，计算机没有办法直接对函数进行运算
- 常微分方程的数值解采用数值离散的方法，即在一系列离散点列上，求未知函数在这些点上函数值的近似
- 基本步骤如下：

（1）对区间进行分割：$\Delta_I : a = x_0 < x_1 < \cdots < x_m = b$（即对函数定义域进行离散），目标是求解$\{y_i = y(x_i)\}$的值

（2）对微分方程进行离散，建立关于$\{y_i\}$的方程，一般要求满足：解的存在唯一性、稳定性、收敛性、相容性

（3）解关于$\{y_i\}$方程，求出$\{y_i\}$的值

微分方程数值解

- 主要问题：
  - 如何对定义域进行离散?
  - 如何对微分方程进行离散?
  - 收敛性问题，即步长充分小时，所得到的数值解能否逼近问题的真解
  - 误差估计
  - 稳定性问题，即舍入误差在以后各步的计算中，是否会无限制扩大
  - 计算效率
  - 并行计算
  - ……

Euler公式

- 对定义域 $[a,b]$ 作等距剖分，即

$$\Delta_I : x_i = a + hi,\ h = \frac{b-a}{m},\ i = 0,1,\ldots,m$$

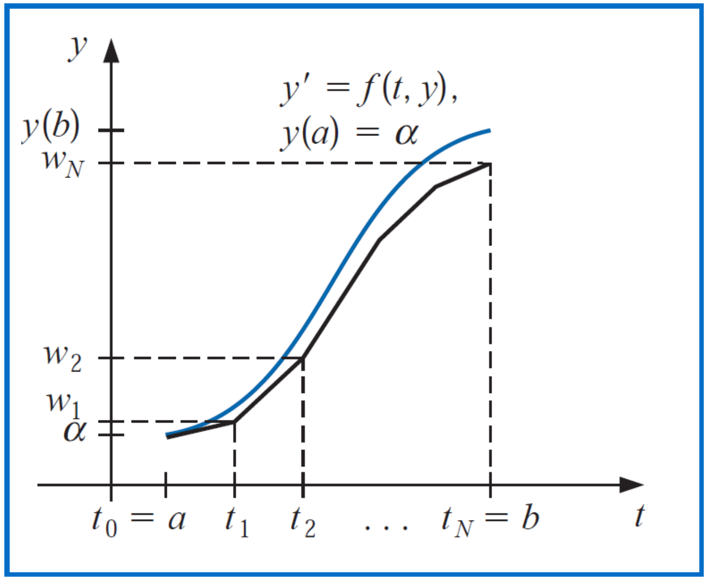
- 向前差商公式

$$\frac{y(x_{i+1}) - y(x_i)}{h} = y'(x_i) + \frac{h}{2} y''(\xi_i)$$

$$\frac{y(x_{i+1}) - y(x_i)}{h} = f(x_i, y(x_i)) + \frac{h}{2} y''(\xi_i)$$

$$y(x_{i+1}) = y(x_i) + hf(x_i, y(x_i)) + \frac{h^2}{2} y''(\xi_i)$$

$$\Rightarrow y_{i+1} = y_i + hf(x_i, y_i)$$



Euler公式

- 定义：在假设第 i 步计算是精确的前提下，考虑截断误差 $T_{i+1}=y(x_{i+1})-y_{i+1}$，称 T_{i+1} 为局部截断误差。若 $T_{i+1}=O(h^{p+1})$，则称方法是 p 阶相容的，简称相容
- 向前差商公式的局部截断误差：

$$
\begin{aligned}
T_{i+1} &= y(x_{i+1})-y_{i+1} \\
&= y(x_i)+hf(x_i,y(x_i))+\frac{h^2}{2}y''(\xi_i)-y_{i+1} \\
&= y(x_i)+hf(x_i,y(x_i))+\frac{h^2}{2}y''(\xi_i)-y_i-hf(x_i,y_i)) \\
&= \frac{h^2}{2}y''(\xi_i) \\
&= O(h^2)
\end{aligned}
$$

Euler公式

- 整体截断误差和收敛性：考虑局部截断误差的积累和传播

$$\left|e_{i+1}\right| = \left|y(x_{i+1}) - y_{i+1}\right|$$

$$= \left| y(x_i) + hf(x_i, y(x_i)) + \frac{h^2}{2} y''(\xi_i) - y_i - hf(x_i, y_i) \right|$$

$$\le \left|y(x_i) - y_i\right| + h\left|f(x_i, y(x_i)) - f(x_i, y_i)\right| + \left|T_{i+1}\right|$$

$$\le \left|e_i\right| + hL\left|y(x_i) - y_i\right| + \left|T_{i+1}\right|$$

$$\le (1+hL)\left|e_i\right| + T,\ T = \max_j \left|T_j\right|$$

$$\le (1+hL)\left((1+hL)\left|e_{i-1}\right| + T\right) + T$$

$$= (1+hL)^2\left|e_{i-1}\right| + \left((1+hL)+1\right)T$$

$$\le (1+hL)^2\left((1+hL)\left|e_{i-2}\right| + T\right) + \left((1+hL)+1\right)T$$

$$= (1+hL)^3\left|e_{i-2}\right| + \left((1+hL)^2 + (1+hL) + 1\right)T$$

Euler公式

- **整体截断误差和收敛性：**

$$
\begin{aligned}
&\le \cdots \\
&\le (1+hL)^{i+1}|e_0| + \left((1+hL)^i + \cdots + (1+hL) + 1\right)T \\
&= (1+hL)^{i+1}|e_0| + \frac{1-(1+hL)^{i+1}}{1-(1+hL)}T \\
&\le (1+hL)^{i+1}|e_0| + \frac{(1+hL)^{i+1}}{hL}T \\
&\le (1+hL)^{i+1}\left(|e_0| + \frac{T}{hL}\right) \\
&\le e^{(i+1)hL}\left(|e_0| + \frac{T}{hL}\right) \le e^{(b-a)L}\left(|e_0| + \frac{T}{hL}\right)
\end{aligned}
$$

是1阶方法

- **当** $e_0 = 0, h \to 0$ **时，**$T = O(h^2) \Rightarrow e^{(b-a)L}\left(|e_0| + \frac{T}{hL}\right) \to 0$， **即Euler公式是收敛的**

Euler公式

- 稳定性：误差在以后各步的计算中不会无限制扩大
- 下面考虑简单情况：仅初值有误差，而其他计算步骤无误差
- 设$\{z_i\}$是初值有误差后的计算值，则

$$y_{i+1} = y_i + hf(x_i, y_i)$$

$$z_{i+1} = z_i + hf(x_i, z_i)$$

$$\Rightarrow |e_{i+1}| \equiv |y_{i+1} - z_{i+1}| \le |e_i| + h|f(x_i, y_i) - f(x_i, z_i)|$$

$$\le |e_i| + hL|y_i - z_i| = |e_i|(1 + hL)$$

$$\le \cdots \le |e_0|(1 + hL)^{i+1} \le |e_0| e^{(i+1)hL} \le C|e_0|$$

- 可以看出，向前差商公式关于初值是稳定的。当初始误差充分小，以后各步的误差也充分小

Euler公式

- 向后差商公式

$$\frac{y(x_{i+1})-y(x_i)}{h}=y'(x_{i+1})+\frac{h}{2}y''(\xi_i)$$

$$\frac{y(x_{i+1})-y(x_i)}{h}=f(x_{i+1},y(x_{i+1}))+\frac{h}{2}y''(\xi_i)$$

$$y(x_{i+1})=y(x_i)+hf(x_{i+1},y(x_{i+1}))+\frac{h^2}{2}y''(\xi_i)$$

$$\Rightarrow y_{i+1}=y_i+hf(x_{i+1},y_{i+1})$$

- 隐式格式，需要迭代求解

Euler公式

- Picard迭代格式：

$$\begin{cases} y_{i+1}^{(0)} = y_i + hf(x_i, y_i), \\ y_{i+1}^{(k+1)} = y_i + hf(x_{i+1}, y_{i+1}^{(k)}), \end{cases} \quad k = 0,1,2,\ldots$$

- 记 $\phi(y) = y_i + hf(x_{i+1}, y)$，则当 h 充分小时，

$$|\phi^{'}(y)| = |hf_y(x_i, y)| \le hL < 1$$

从而迭代收敛

Euler公式

- 中心差商公式

$$\frac{y(x_{i+1})-y(x_{i-1})}{2h}=y'(x_i)+\frac{h^2}{3!}y''(\xi_i)$$

$$\frac{y(x_{i+1})-y(x_{i-1})}{2h}=f(x_i,y(x_i))+\frac{h^2}{3!}y''(\xi_i)$$

$$\Rightarrow y_{i+1}=y_{i-1}+2hf(x_i,y_i)$$

- 多步格式，二阶格式，数值不稳定

Euler公式

- **基于数值积分的近似公式**

$$\frac{dy}{dx}=f(x,y),\ x\in[a,b]\Rightarrow y(x_{i+1})-y(x_i)=\int_{x_i}^{x_{i+1}}f(x,y)dx=\int_{x_i}^{x_{i+1}}y'(t)dx$$

- **若取** $y'(t)\approx y'(x_i)=f(x_i,y(x_i))$，**则有**

$$\int_{x_i}^{x_{i+1}}y'(t)dx\approx(x_{i+1}-x_i)y'(x_i)=hf(x_i,y(x_i))\Rightarrow y_{i+1}=y_i+hf(x_i,y_i)$$

- **若取** $y'(t)\approx y'(x_{i+1})=f(x_{i+1},y(x_{i+1}))$，**则有**

$$\int_{x_i}^{x_{i+1}}y'(t)dx\approx(x_{i+1}-x_i)y'(x_{i+1})=hf(x_{i+1},y(x_{i+1}))\Rightarrow y_{i+1}=y_i+hf(x_{i+1},y_{i+1})$$

- **若取** $\int_{x_i}^{x_{i+1}}y'(t)dx\approx\frac{h}{2}(f(x_i,y(x_i))+f(x_{i+1},y(x_{i+1})))$，**则有**

$$y_{i+1}=y_i+\frac{h}{2}(f(x_i,y_i)+f(x_{i+1},y_{i+1}))$$

$$\Rightarrow\begin{cases}\bar{y}_{i+1}=y_i+hf(x_i,y_i)\\ y_{i+1}=y_i+\frac{h}{2}(f(x_i,y_i)+f(x_{i+1},\bar{y}_{i+1}))\end{cases}$$

预估－校正公式

Euler公式

- 向前差商的Euler算法

Algorithm 21 Euler's Algorithm

Input:

$f(x, y)$, a, b, m, y_0;

$h \leftarrow (b-a)/m$;

for $i = 1$ to m **do**

  $x_i = x_{i-1} + h$;

  $y_i = y_{i-1} + h * f(x_{i-1}, y_{i-1})$;

end for

Output:

(x_i), (y_i)

Runge-Kutta方法

- Taylor级数法（高阶单步方法）：

$$\begin{cases} y(x_{i+1}) = y(x_i) + hy'(x_i) + \cdots + \dfrac{h^k}{k!} y^{(k)}(x_i) + \dfrac{h^{k+1}}{(k+1)!} y^{(k+1)}(\xi_i) \\ y'(x) = f(x,y) \\ y''(x) = f_x(x,y) + f_y(x,y) \cdot y' \\ y'''(x) = \cdots \end{cases}$$

$$\Rightarrow y_{i+1} = y_i + hf(x_i, y_i) + \frac{h^2}{2!}\Big(f_x(x_i, y_i) + f_y(x_i, y_i) \cdot f(x_i, y_i) \Big) + \cdots$$

- 取 $k=1$，向前差商的Euler公式
- 取 $k=2$，可得

$$y_{i+1} = y_i + h\left[f(x_i, y_i) + \frac{h}{2}\big(f_x(x_i, y_i) + f_y(x_i, y_i) f(x_i, y_i) \big) \right]$$

Runge-Kutta方法

- Taylor级数法：要大量计算复合函数的全导数
- Runge-Kutta方法：一个点上的导数值可以用它邻近的一些点上的函数值来近似表示（数值微分的思想）+ Taylor级数法
- 基本思想：

$$y'(x_i)+\cdots+\frac{h^{k-1}}{k!}y^{(k)}(x_i)=\sum_{j=1}^{k}\frac{h^{j-1}}{j!}\frac{d^{j-1}}{dx^{j-1}}f(x_i,y(x_i))\approx\sum_{j=1}^{k}c_j k_j$$

$$k_1=f(x_i,y_i)$$

$$k_j=f(x_i+a_j h,y_i+h\sum_{l=1}^{j-1}b_{jl}k_l)$$

$$a_j=\sum_{l=1}^{j-1}b_{jl}$$

Runge-Kutta方法

- 基本做法：利用二元函数的Taylor展开形式

$$f(x+h,y+l)=\sum_{k=0}^{\infty}\frac{1}{k!}\left(h\frac{\partial}{\partial x}+l\frac{\partial}{\partial y}\right)^k f(x,y)$$

比较方程两边幂次相同的h项系数，得到关于a_j,b_{jl},c_j的方程组

- 取$k=2$，作Taylor展开

$$c_1 f(x_i,y(x_i))+c_2 f(x_i+a_2h,y(x_i)+b_{21}hf(x_i,y(x_i)))$$
$$\approx c_1 f(x_i,y(x_i))+c_2 f(x_i,y(x_i))+c_2a_2hf_x(x_i,y(x_i))+$$
$$c_2b_{21}hf(x_i,y(x_i))\cdot f_y(x_i,y(x_i))+O(h^2)$$

与$f(x_i,y_i)+\frac{h}{2}\left(f_x(x_i,y_i)+f_y(x_i,y_i)f(x_i,y_i)\right)$比较得：

$$\Rightarrow\begin{cases}c_1+c_2=1\\ c_2a_2=1/2\\ c_2b_{21}=1/2\end{cases}$$

4个未知数，3个方程，有无穷多组解

Runge-Kutta方法

- 修正的Euler法（中点法）：取 $c_1=0,\ c_2=1,\ a_2=b_{21}=1/2$

$$\begin{cases} y_{i+1}=y_i+hk_2 \\ k_1=f(x_i,y_i) \\ k_2=f(x_i+h/2,y_i+hk_1/2) \end{cases}$$

- 二阶Runge-Kutta公式：取 $c_1=1/2,\ c_2=1/2,\ a_2=b_{21}=1$

$$\begin{cases} y_{i+1}=y_i+h(k_1+k_2)/2 \\ k_1=f(x_i,y_i) \\ k_2=f(x_i+h,y_i+hk_1) \end{cases}$$

- Henu公式：取 $c_1=1/4,\ c_2=3/4,\ a_2=b_{21}=2/3$

$$\begin{cases} y_{i+1}=y_i+h(k_1+3k_2)/4 \\ k_1=f(x_i,y_i) \\ k_2=f(x_i+2h/3,y_i+2hk_1/3) \end{cases}$$

Runge-Kutta方法

- 三阶Kutta公式：

$$\begin{cases} y_{i+1} = y_i + h(k_1 + 4k_2 + k_3)/6 \\ k_1 = f(x_i, y_i) \\ k_2 = f(x_i + h/2, y_i + hk_1/2) \\ k_3 = f(x_i + h, y_i - hk_1 + 2hk_2) \end{cases}$$

- 三阶Henu公式：

$$\begin{cases} y_{i+1} = y_i + h(k_1 + 3k_3)/4 \\ k_1 = f(x_i, y_i) \\ k_2 = f(x_i + h/3, y_i + hk_1/3) \\ k_3 = f(x_i + 2h/3, y_i + 2hk_2/3) \end{cases}$$

Runge-Kutta方法

- 四阶Runge-Kutta公式：

$$\begin{cases} y_{i+1} = y_i + h(k_1 + 2k_2 + 2k_3 + k_4)/6 \\ k_1 = f(x_i, y_i) \\ k_2 = f(x_i + h/2, y_i + hk_1/2) \\ k_3 = f(x_i + h/2, y_i + hk_2/2) \\ k_4 = f(x_i + h, y_i + hk_3) \end{cases}$$

- 四阶Kutta公式：

$$\begin{cases} y_{i+1} = y_i + h(k_1 + 3k_2 + 3k_3 + k_4)/8 \\ k_1 = f(x_i, y_i) \\ k_2 = f(x_i + h/3, y_i + hk_1/3) \\ k_3 = f(x_i + 2h/3, y_i - hk_1/3 + hk_2) \\ k_4 = f(x_i + h, y_i + hk_1 - hk_2 + hk_3) \end{cases}$$

Runge-Kutta方法

- 步长的自适应：设置一个误差容限，（1）假如超出误差容限，则否决这一步并减小步长；（2）假如符合误差容限，则接受这一步并选取适合下一步的步长
- 改变步长的策略：
  - 步长加倍或减半
  - 根据阶的信息选取适当的步长

$$\begin{cases} e_i \approx c h_i^{p+1} \\ \dfrac{e_i}{|y_i|} < T \end{cases} \Rightarrow h_{i+1} = 0.8 * \left(\frac{T\,|y_i|}{e_i} \right)^{\frac{1}{p+1}} h_i$$

- 嵌入Runge-Kutta对：一个p阶和另一个$p+1$阶，共享必要的计算

Runge-Kutta方法

- 嵌入Runge-Kutta 4/5对公式

$$k_1 = f(x_i, y_i)$$

$$k_2 = f(x_i + \frac{1}{4}h, y_i + \frac{1}{4}hk_1)$$

$$k_3 = f(x_i + \frac{3}{8}h, y_i + \frac{3}{32}hk_1 + \frac{9}{32}hk_2)$$

$$k_4 = f(x_i + \frac{12}{13}h, y_i + \frac{1932}{2197}hk_1 - \frac{7200}{2197}hk_2 + \frac{7296}{2197}hk_3)$$

$$k_5 = f(x_i + h, y_i + \frac{439}{216}hk_1 - 8hk_2 + \frac{3680}{513}hk_3 - \frac{845}{4104}hk_4)$$

$$k_6 = f(x_i + \frac{1}{2}h, y_i - \frac{8}{27}hk_1 + 2hk_2 - \frac{3544}{2565}hk_3 + \frac{1859}{4104}hk_4 - \frac{11}{40}hk_5)$$

$$y_{i+1} = y_i + h\left(\frac{25}{216}k_1 + \frac{1408}{2565}k_3 + \frac{2197}{4104}k_4 - \frac{1}{5}k_5\right)$$

$$z_{i+1} = y_i + h\left(\frac{16}{135}k_1 + \frac{6656}{12825}k_3 + \frac{28561}{56430}k_4 - \frac{9}{50}k_5 + \frac{2}{55}k_6\right)$$

Runge-Kutta方法

- 步长控制所需的误差估计为：

$$e_i = |z_{i+1} - y_{i+1}| = h\left|\frac{1}{360}k_1 - \frac{128}{4275}k_3 - \frac{2197}{75240}k_4 + \frac{1}{50}k_5 + \frac{2}{55}k_6\right|$$

- 步长自适应策略：

  (1) 若误差测试 $\frac{e_i}{|y_i|} < T$ 成功，则用 z_{i+1} 取代 y_{i+1}，程序进入下一步；

  (2) 否则，用 $h_{i+1} = 0.8 * \left(\frac{T|y_i|}{e_i}\right)^{\frac{1}{p+1}} h_i$ 再尝试一遍（如果重复失败，则步长减半直至成功）

- 在Matlab中，就使用这种方法，譬如ode23，ode45等命令（分别使用嵌入Bogacki-Shampine 2/3对、嵌入Dormand-Prince 4/5对）

Runge-Kutta方法

- Runge-Kutta算法

Algorithm 22 Runge-Kutta Method

Input:

$f(x, y)$, a, b, m, y_0;

$h \leftarrow (b-a)/m$;
for $i = 1$ to m **do**
  $K_1 = hf(x_{i-1}, y_{i-1})$;
  $K_2 = hf(x_{i-1} + h/2, y_{i-1} + K_1/2)$;
  $K_3 = hf(x_{i-1} + h/2, y_{i-1} + K_2/2)$;
  $K_4 = hf(x_{i-1} + h, y_{i-1} + K_3)$;
  $x_i = x_{i-1} + h$;
  $y_i = y_{i-1} + (K_1 + 2K_2 + 2K_3 + K_4)/6$;
end for

Output:

(x_i), (y_i)

线性多步法

- 线性 k 步方法的一般形式：

$$\sum_{j=0}^{k}\alpha_j y_{m+j} = h\sum_{j=0}^{k}\beta_j f_{m+j},$$

需要给定 k 个初始值 $y_0, y_1, \ldots, y_{k-1}$ 才能启动迭代

- 基本思想：微分方程化为积分方程，用数值积分近似

$$\begin{cases} \dfrac{dy}{dx} = f(x,y) \\ y(a) = y_0 \end{cases}, \; a \le x \le b \quad \Leftrightarrow \quad y(x) = y(x^*) + \int_{x^*}^{x} y'(t)dt$$

线性多步法

- 取 x, x^* 分别为 x_{i+1}, x_{i-p}，有

$$y(x_{i+1}) = y(x_{i-p}) + \int_{x_{i-p}}^{x_{i+1}} y'(x)dx = y(x_{i-p}) + h\sum_{j=0}^{q} a_{i-j} f(x_{i-j}, y(x_{i-j})) + hT_{i+1}$$

- 显式公式：用数值积分节点 $x_i, x_{i-1}, \ldots, x_{i-q}$ 构造插值多项式近似 $y'(x)$，在区间 $[x_{i-p}, x_{i+1}]$ 上计算数值积分 $\int_{x_{i-p}}^{x_{i+1}} y'(x)dx$
- 隐式公式：用数值积分节点 $x_{i+1}, x_i, \ldots, x_{i+1-q}$ 构造插值多项式近似 $y'(x)$，在区间 $[x_{i-p}, x_{i+1}]$ 上计算数值积分 $\int_{x_{i-p}}^{x_{i+1}} y'(x)dx$
- p 控制积分区间，q 控制插值节点

线性多步法

- Adams格式：取 $p=0$
- 例：构造 $p=0,\ q=1$的显式格式

$$y(x_{i+1}) = y(x_i) + \int_{x_i}^{x_{i+1}} \left[l_0(x) y'(x_i) + l_1(x) y'(x_{i-1}) + R(x) \right] dx$$

$$= y(x_i) + a_0 y'(x_i) + a_1 y'(x_{i-1}) + T_{i+1}$$

$$a_0 = \int_{x_i}^{x_{i+1}} l_0(x) dx = \int_{x_i}^{x_{i+1}} \frac{x - x_{i-1}}{x_i - x_{i-1}} dx = \frac{3}{2} h$$

$$a_1 = \int_{x_i}^{x_{i+1}} l_1(x) dx = \int_{x_i}^{x_{i+1}} \frac{x - x_i}{x_{i-1} - x_i} dx = -\frac{1}{2} h$$

$$T_{i+1} = \int_{x_i}^{x_{i+1}} R(x) dx = \int_{x_i}^{x_{i+1}} \frac{y^{(3)}(\eta_x)}{2!} (x - x_i)(x - x_{i-1}) dx = \frac{5}{12} h^3 y^{(3)}(\xi)$$

$$\Rightarrow y_{i+1} = y_i + \frac{h\left[3f(x_i, y_i) - f(x_{i-1}, y_{i-1}) \right]}{2}$$

二阶显式Adams公式

线性多步法

- **三阶显式Adams公式：**

$$y_{i+1} = y_i + \frac{h\left[23f(x_i, y_i) - 16f(x_{i-1}, y_{i-1}) + 5f(x_{i-2}, y_{i-2})\right]}{12}, \quad T_{i+1} = \frac{3}{8}h^4 y^{(4)}(\xi)$$

- **四阶显式Adams公式：**

$$y_{i+1} = y_i + \frac{h\left[55f(x_i, y_i) - 59f(x_{i-1}, y_{i-1}) + 37f(x_{i-2}, y_{i-2}) - 9f(x_{i-3}, y_{i-3})\right]}{24},$$

$$T_{i+1} = \frac{251}{720}h^5 y^{(5)}(\xi)$$

- **三阶隐式Adams公式：**

$$y_{i+1} = y_i + \frac{h\left[5f(x_{i+1}, y_{i+1}) + 8f(x_i, y_i) - f(x_{i-1}, y_{i-1})\right]}{12}, \quad T_{i+1} = -\frac{1}{24}h^4 y^{(4)}(\xi)$$

- **四阶隐式Adams公式：**

$$y_{i+1} = y_i + \frac{h\left[9f(x_{i+1}, y_{i+1}) + 19f(x_i, y_i) - 5f(x_{i-1}, y_{i-1}) + f(x_{i-2}, y_{i-2})\right]}{24},$$

$$T_{i+1} = -\frac{19}{720}h^5 y^{(5)}(\xi)$$

线性多步法

- 例：构造 $p=1,\ q=2$ 的显式格式

$$y(x_{i+1}) = y(x_{i-1}) + \int_{x_{i-1}}^{x_{i+1}} [l_0(x)y'(x_i) + l_1(x)y'(x_{i-1}) + l_2(x)y'(x_{i-2}) + R(x)]dx$$

$$= y(x_{i-1}) + a_0 y'(x_i) + a_1 y'(x_{i-1}) + a_2 y'(x_{i-2}) + T_{i+1}$$

$$a_0 = \int_{x_{i-1}}^{x_{i+1}} l_0(x)dx = \int_{x_{i-1}}^{x_{i+1}} \frac{(x-x_{i-1})(x-x_{i-2})}{(x_i-x_{i-1})(x_i-x_{i-2})}dx = \frac{7}{3}h$$

$$a_1 = \int_{x_{i-1}}^{x_{i+1}} l_1(x)dx = \int_{x_{i-1}}^{x_{i+1}} \frac{(x-x_i)(x-x_{i-2})}{(x_{i-1}-x_i)(x_{i-1}-x_{i-2})}dx = -\frac{2}{3}h$$

$$a_2 = \int_{x_{i-1}}^{x_{i+1}} l_2(x)dx = \int_{x_{i-1}}^{x_{i+1}} \frac{(x-x_i)(x-x_{i-1})}{(x_{i-2}-x_i)(x_{i-2}-x_{i-1})}dx = \frac{1}{3}h$$

$$T_{i+1} = \int_{x_{i-1}}^{x_{i+1}} R(x)dx = \int_{x_{i-1}}^{x_{i+1}} \frac{y^{(4)}(\eta_x)}{3!}(x-x_i)(x-x_{i-1})(x-x_{i-2})dx$$

$$\Rightarrow y_{i+1} = y_{i-1} + \frac{h[7f(x_i,y_i) - 2f(x_{i-1},y_{i-1}) + f(x_{i-2},y_{i-2})]}{3}$$

线性多步法

- 例：构造 $p=2,\ q=2$ 的隐式格式

$$y(x_{i+1})=y(x_{i-2})+\int_{x_{i-2}}^{x_{i+1}}\left[l_0(x)y'(x_{i+1})+l_1(x)y'(x_i)+l_2(x)y'(x_{i-1})+R(x)\right]dx$$

$$=y(x_{i-2})+a_0y'(x_{i+1})+a_1y'(x_i)+a_2y'(x_{i-1})+T_{i+1}$$

$$a_0=\int_{x_{i-2}}^{x_{i+1}}l_0(x)dx=\int_{x_{i-2}}^{x_{i+1}}\frac{(x-x_i)(x-x_{i-1})}{(x_{i+1}-x_i)(x_{i+1}-x_{i-1})}dx=\frac{3}{4}h$$

$$a_1=\int_{x_{i-2}}^{x_{i+1}}l_1(x)dx=\int_{x_{i-2}}^{x_{i+1}}\frac{(x-x_{i+1})(x-x_{i-1})}{(x_i-x_{i+1})(x_i-x_{i-1})}dx=0$$

$$a_2=\int_{x_{i-2}}^{x_{i+1}}l_2(x)dx=\int_{x_{i-2}}^{x_{i+1}}\frac{(x-x_{i+1})(x-x_i)}{(x_{i-1}-x_{i+1})(x_{i-1}-x_i)}dx=\frac{9}{4}h$$

$$T_{i+1}=\int_{x_{i-2}}^{x_{i+1}}R(x)dx=\int_{x_{i-2}}^{x_{i+1}}\frac{y^{(4)}(\eta_x)}{3!}(x-x_{i+1})(x-x_i)(x-x_{i-1})dx$$

$$\Rightarrow y_{i+1}=y_{i-2}+\frac{h\left[3f(x_{i+1},y_{i+1})+9f(x_{i-1},y_{i-1})\right]}{4}$$

线性多步法

- 为避免迭代，可用预估－校正公式

$$\begin{cases} \bar{y}_{i+1} = y_{i-1} + \dfrac{h\left[7f(x_i, y_i) - 2f(x_{i-1}, y_{i-1}) + f(x_{i-2}, y_{i-2})\right]}{3} \\ y_{i+1} = y_{i-2} + \dfrac{h\left[3f(x_{i+1}, \bar{y}_{i+1}) + 9f(x_{i-1}, y_{i-1})\right]}{4} \end{cases}$$

常微分方程组的数值解法

- 设 m 个一阶方程组成的常微分方程初值问题

$$\begin{cases} \dfrac{dy_1}{dx} = f_1(x, y_1, \cdots, y_m) \\ \quad\vdots \\ \dfrac{dy_m}{dx} = f_m(x, y_1, \cdots, y_m) \ , \ a \le x \le b \\ y_1(a) = \eta_1 \\ \quad\vdots \\ y_m(a) = \eta_m \end{cases}$$

- 写成向量形式

$$\begin{cases} \dfrac{dY}{dx} = F(x, Y(x)) \\ Y(a) = \eta \end{cases}, \ Y(x) = \begin{pmatrix} y_1(x) \\ y_2(x) \\ \vdots \\ y_m(x) \end{pmatrix}, \ F(x, Y(x)) = \begin{pmatrix} f_1(x, y_1(x), \ldots, y_m(x)) \\ f_2(x, y_1(x), \ldots, y_m(x)) \\ \vdots \\ f_m(x, y_1(x), \ldots, y_m(x)) \end{pmatrix}$$

常微分方程组的数值解法

- 各种方法都可以直接运用过来
- 例：两个一阶微分方程组成的微分方程组

$$\begin{cases} \dfrac{dy}{dx} = f(x,y,z) \\ \dfrac{dz}{dx} = g(x,y,z), \quad a \le x \le b \\ y(a) = y_0 \\ z(a) = z_0 \end{cases}$$

- Euler公式

$$\begin{cases} y_{n+1} = y_n + hf(x_n, y_n, z_n) \\ z_{n+1} = z_n + hg(x_n, y_n, z_n) \end{cases}$$

常微分方程组的数值解法

- **Runge-Kutta公式**

$$\begin{pmatrix} y_{n+1} \\ z_{n+1} \end{pmatrix} = \begin{pmatrix} y_n \\ z_n \end{pmatrix} + \frac{h}{6}(K_1 + 2K_2 + 2K_3 + K_4)$$

$$K_1 = \begin{pmatrix} f(x_n, y_n, z_n) \\ g(x_n, y_n, z_n) \end{pmatrix}$$

$$K_2 = \begin{pmatrix} f(x_n + \frac{h}{2}, y_n + \frac{h}{2}K_1^{(1)}, z_n + \frac{h}{2}K_1^{(2)}) \\ g(x_n + \frac{h}{2}, y_n + \frac{h}{2}K_1^{(1)}, z_n + \frac{h}{2}K_1^{(2)}) \end{pmatrix}$$

$$K_3 = \begin{pmatrix} f(x_n + \frac{h}{2}, y_n + \frac{h}{2}K_2^{(1)}, z_n + \frac{h}{2}K_2^{(2)}) \\ g(x_n + \frac{h}{2}, y_n + \frac{h}{2}K_2^{(1)}, z_n + \frac{h}{2}K_2^{(2)}) \end{pmatrix}$$

$$K_4 = \begin{pmatrix} f(x_n + h, y_n + hK_3^{(1)}, z_n + hK_3^{(2)}) \\ g(x_n + h, y_n + hK_3^{(1)}, z_n + hK_3^{(2)}) \end{pmatrix}$$

常微分方程组的数值解法

- 高阶微分方程：引入辅助变量，可化为一阶微分方程组
- 设m阶常微分方程

$$\begin{cases} \dfrac{d^m y}{dx^m} = f(x, y, y', \cdots, y^{(m-1)}) \\ y(a) = \eta_1 \\ y'(a) = \eta_2 \\ \quad \vdots \\ y^{(m-1)}(a) = \eta_m \end{cases}, \quad a \le x \le b$$

- 引入辅助变量

$$\begin{cases} y = y_1 \\ \dfrac{dy_1}{dx} = y_2 \\ \quad \vdots \\ \dfrac{dy_{m-1}}{dx} = y_m \end{cases} \Rightarrow \begin{cases} \dfrac{dy_1}{dx} = y_2 \\ \vdots \\ \dfrac{dy_m}{dx} = f(x, y_1, \cdots, y_m), \quad a \le x \le b \\ y_1(a) = \eta_1 \\ \vdots \\ y_m(a) = \eta_m \end{cases}$$

常微分方程的稳定性

- 定义：若一个离散变量的 k 步方法，在Lipschitz条件下，存在常数 C 和 $h_0>0$，使得以 $(0,h_0]$ 中任意值 h 为步长，通过任意两组初值 $\{u_m\}_{m=0}^{k-1}$ 和 $\{v_m\}_{m=0}^{k-1}$ 得到两组离散值 $\{u_m\}_{m=k}^{N}$ 和 $\{v_m\}_{m=k}^{N}$（$N=\left[\frac{T-t_0}{h}\right]$），都成立

$$\max_{k\le m\le N}|u_m-v_m|\le C\max_{0\le m\le k-1}|u_m-v_m|,$$

则称该方法是稳定的

- 数值方法的稳定性：初值产生误差的传播（或者离散解关于初值的连续依赖性）
- 数值方法的稳定性不仅于数值方法有关，而且与微分方程本身有关。如果微分方程本身是不稳定的，那就没理由要求数值方法稳定。因此，数值方法的稳定性概念是建立在微分方程稳定的基础上的

常微分方程的稳定性

- 为了比较不同方法的性质，取典型的微分方程

$$\begin{cases} \dfrac{dy}{dx} = \lambda y, \ \ \mathrm{Re}(\lambda) < 0 \\ y(a) = y_0 \end{cases}$$

- 将一般的差分方程（数值格式）：

$$\sum_{j=0}^{k} a_j y_{n+j} = h \sum_{j=0}^{k} b_j f(x_{n+j}, y_{n+j})$$

应用到典型的微分方程，可得：

$$\sum_{j=0}^{k} a_j y_{n+j} = \lambda h \sum_{j=0}^{k} b_j y_{n+j}, \ \ \mathrm{Re}(\lambda) < 0$$

对于给定的初始误差$e_0, e_1, \cdots, e_{k-1}$，误差方程具有一样的形式

$$\sum_{j=0}^{k} a_j e_{n+j} = \lambda h \sum_{j=0}^{k} b_j e_{n+j}, \ \ \mathrm{Re}(\lambda) < 0$$

常微分方程的稳定性

- 定义：差分方程称为绝对稳定的，若差分方程作用到典型的微分方程

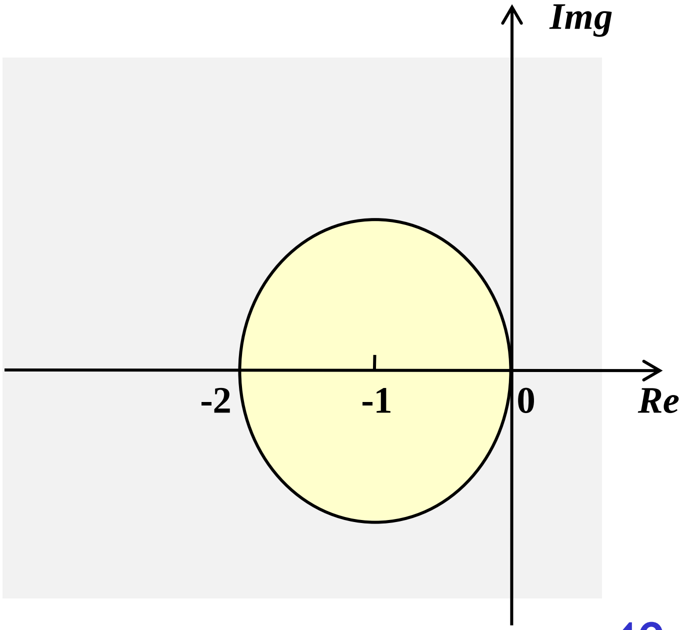
$$\frac{dy}{dx} = \lambda y, \ \ \mathrm{Re}(\lambda) < 0,$$

对任意的初值，总存在左半复平面上的一个区域，当在这个区域时，差分方程的解趋于 $0 \ (x \to \infty)$，这个区域称为稳定区域

- 例：向前Euler公式的稳定性

$$y_{n+1} = y_n + \lambda h y_n \ \ \Rightarrow \ \ e_{n+1} = e_n + \lambda h e_n$$

$$\Rightarrow \frac{|e_{n+1}|}{|e_n|} = |1 + \lambda h| < 1$$



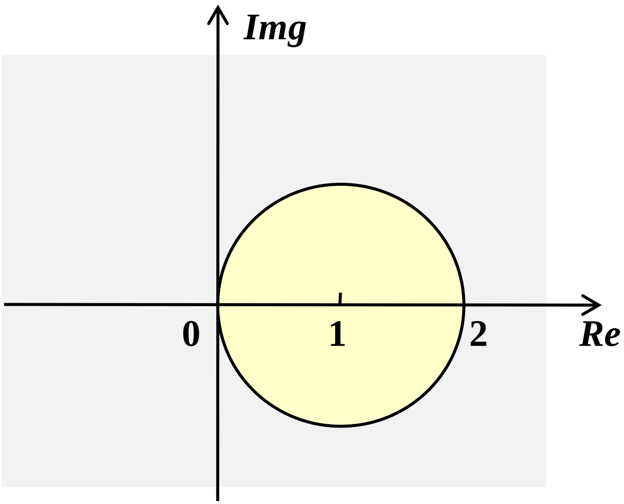
常微分方程的稳定性

- 例：向后Euler公式的稳定性

$$y_{i+1} = y_i + h\lambda\, y_{i+1} \Rightarrow y_{i+1} = \left(\frac{1}{1-\lambda h}\right) y_i$$

$$\Rightarrow e_{i+1} = \left(\frac{1}{1-\lambda h}\right) e_i$$

$$\Rightarrow \frac{|e_{n+1}|}{|e_n|} = \frac{1}{|1-\lambda h|} < 1$$



- 定义：当绝对稳定区域是左半平面时，则称该数值方法是无条件绝对稳定的
- 一般来说，隐式格式的绝对稳定性比同阶的显式法的好

常微分方程的稳定性

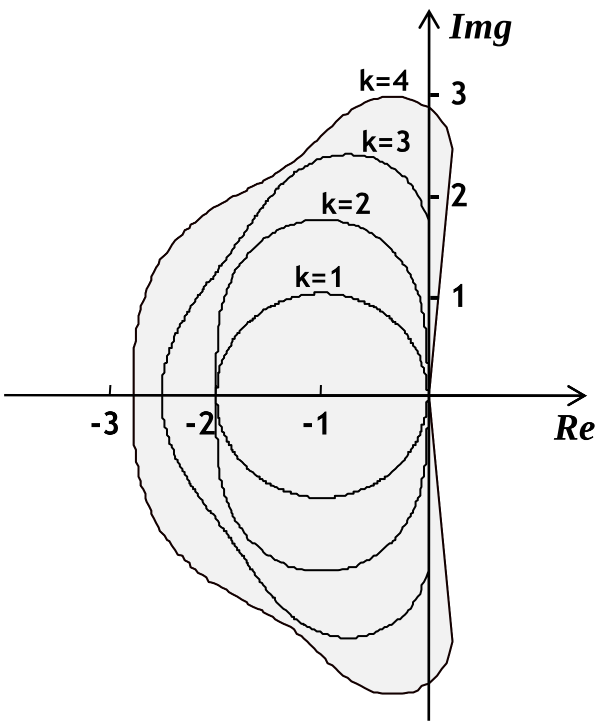
- 例：3阶Runge-Kutta公式的稳定性

$$y_{n+1} = y_n + \frac{h}{6}[K_1 + 4K_2 + K_3]$$

$$K_1 = \lambda y_n$$

$$K_2 = \lambda y_n (1 + \frac{1}{2}\lambda h)$$

$$K_3 = \lambda y_n \left[1 + \lambda h + (\lambda h)^2\right]$$



$$y_{n+1} = y_n \left[1 + \lambda h + \frac{1}{2}(\lambda h)^2 + \frac{1}{6}(\lambda h)^3\right]$$

$$\Rightarrow \frac{|e_{n+1}|}{|e_n|} = \left|1 + \lambda h + \frac{1}{2}(\lambda h)^2 + \frac{1}{6}(\lambda h)^3\right| < 1$$

- 可以证明：Runge-Kutta方法和隐式Adams方法都是绝对稳定的