经典的连续系统仿真建模方法学

参考书：肖田元，范文慧，连续系统建模与仿真，电子工业出版社，2010：130-152.

连续系统离散化原理

- 问题：数字计算机在数值及时间上的离散性

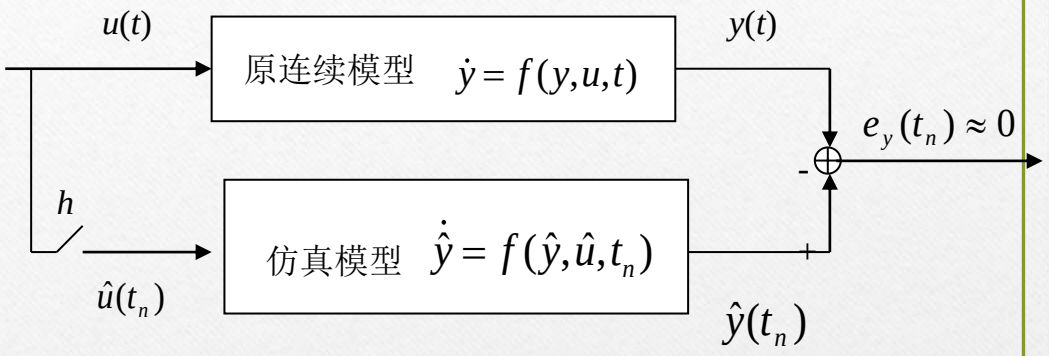
  被仿真系统数值及时间上的连续性

- 连续系统的仿真，从本质上：对原连续系统从时间、数值两个方面对原系统进行离散化并选择合适的数值计算方法来近似积分运算

- 离散模型≈原连续模型？

连续系统离散化原理

1.相似原理



设系统模型为：$\dot{y}=f(y,u,t)$，其中$u(t)$为输入变量，$y(t)$为系统变量；令仿真时间间隔为h，离散化后的输入变量为$\hat{u}(t_n)$，系统变量为$\hat{y}(t_n)$，其中t_n表示$t=nh$。

如果 $\hat{u}(t_n)\approx u(t_n)$, 且 $\hat{y}(t_n)\approx y(t_n)$ ←在$t=t_n$时 离散化前后的输出一致

即$e_u(t_n)=\hat{u}(t_n)-u(t_n)\approx 0$, $e_y(t_n)=\hat{y}(t_n)-y(t_n)\approx 0$

（对所有 $n=0,1,2,\ldots$），则可认为两模型等价。

条件：在$t=t_n=nh$时，离散化前后的输入一致、输出一致。

连续系统离散化原理

2. 对仿真建模方法三个基本要求

（1）稳定性：若原连续系统是稳定的，则离散化后得到的仿真模型也应是稳定的。

（2）准确性：有不同的准确性评价准则，最基本的准则是：

绝对误差准则：$\left|e_y(t_n)\right| = \left|\hat{y}(t_n) - y(t_n)\right| \le \delta$

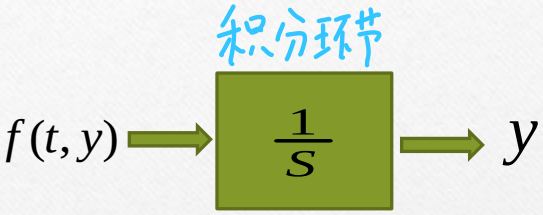
相对误差准则：$\left|e_y(t_n)\right| = \left|\dfrac{\hat{y}(t_n) - y(t_n)}{\hat{y}(t_n)}\right| \le \delta$

其中δ规定精度的误差量。

（3）快速性：若第 n 步计算对应的系统时间间隔为 $h_n = t_{n+1} - t_n$，计算机由计算需要的时间为 T_n，若 $T_n=h_n$ 称为实时仿真，$T_n<h_n$ 称为超实时仿真，$T_n>h_n$ 称为亚实时仿真，对应于离线仿真

连续系统离散化原理

3. 数值积分法



（1） 数值积分法原理

考虑 如下的具有初值的单变量一阶常微分方程

$$\begin{cases} \dot{y} = f(t,y) \\ y(t_0) = y_0 \end{cases} \quad (1)$$

初始条件

对(1)的微分方程两边在时间段$[t_k,t_{k+1}]$求积分，得

$$\int_{t_k}^{t_{k+1}} \frac{dy}{dt} dt = \int_{t_k}^{t_{k+1}} f(t,y)dt$$

$$\Rightarrow \quad y_{k+1} = y_k + \int_{t_k}^{t_{k+1}} f(t,y)dt \quad (2)$$

连续系统离散化原理

$$\begin{cases} \dot{y} = f(t, y) \\ y(t_0) = y_0 \end{cases}$$ 初始条件

通常假设离散时间点 $t_0, t_1, \ldots, t_n$ 是等距离的，从而定义步长h为

$$h = t_{k+1} - t_k \tag{3}$$

(2)式的右端存在一个积分项 Q_k，

$$Q_k = \int_{t_k}^{t_{k+1}} f(t, y)\,dt \tag{4}$$

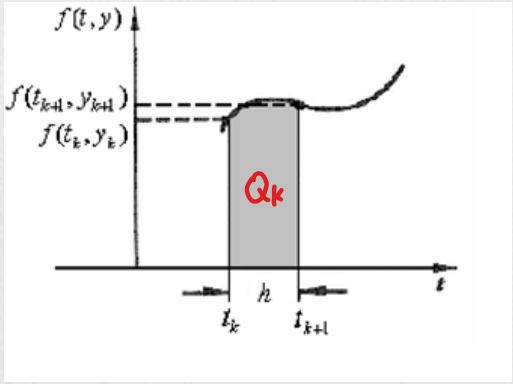
数值积分的目的就是用数值计算的方法计算出该积分项 Q_k 的值，从而构造一个迭代式

$$y_{k+1} = y_k + Q_k \tag{5}$$

因为已知 y_0，那么就可以迭代计算出 $y_1, y_2, \ldots$

根据积分的定义，右图所示曲线下的灰色区域的面积就是 Q_k

数值积分方法的不同就在于Q_k的计算方法不同。

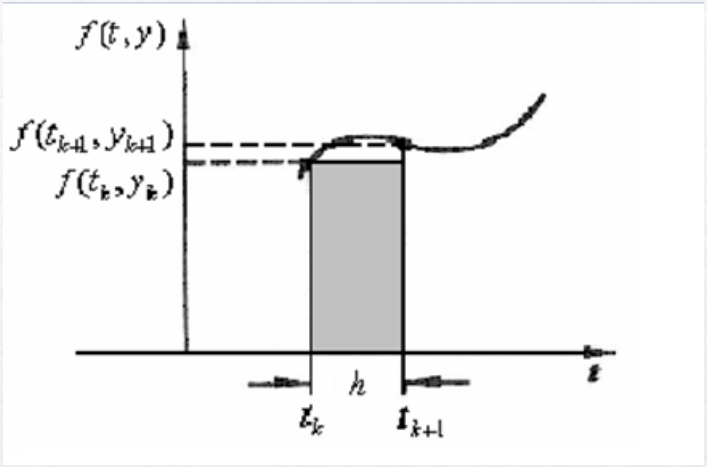


连续系统离散化原理

（2）欧拉法(Euler)

欧拉法是用矩形面积近似表示一个步长区间内的积分面积，即

$$Q_k = h \cdot f(t_k, y_k) \tag{6}$$



带入到(5)式，得到

$$y_{k+1} = y_k + h \cdot f(t_k, y_k) \tag{7}$$

这相当于将整个曲线 *f(t,y)* 所夹的面积分为多个宽度为h的矩形

显然，h越小，计算的面积越准确，但计算量越大。

连续系统离散化原理

已知初始时刻t_0时的初值y_0，

就可以迭代计算出任意时刻的y

y为积分环节的输出

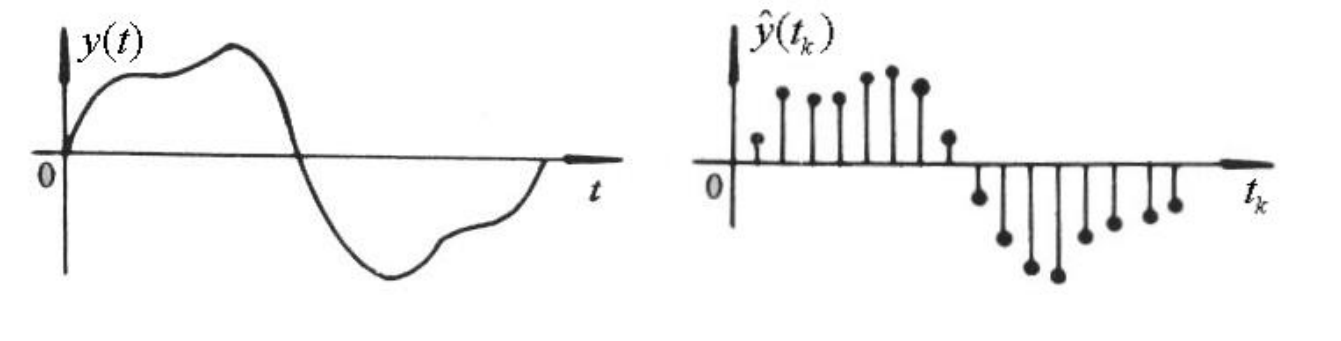
h为时间间隔(步长)

$$k=0,\quad y_1=y_0+h\cdot f(t_0,y_0)$$

$$k=1,\quad y_2=y_1+h\cdot f(t_1,y_1)$$

$$\vdots$$

$$k=n-1,\quad y_n=y_{n-1}+h\cdot f(t_{n-1},y_{n-1})$$



如此进行的计算，得到一系列离散时间点的y的值y_k，而系统(1)的解析解为连续函数y(t)，因而数值积分实际上是相当于用离散序列y_k来表示表示连续函数y(t)，相当于连续信号的采样。由采样定理可知，只要步长h足够小，则数值积分解可近似表示连续解。

连续系统离散化原理

欧拉法比较简单，但计算误差也是比较大的，只有当步长比较小时，计算误差才会比较小。实际上，欧拉法相当于对解析解 y(t) 在 t_k 处近似展开，然后取 t_{k+1} 处的值，即 y=t_k处展开式 $y(t)=y(t_k)+y'(t_k)(t-t_k)+\frac{y''(t_k)}{2}(t-t_k)+$

$$y(t_{k+1}) = y(t_k) + \dot{y}(t_k)h + \frac{\ddot{y}(t_k)h^2}{2} + \cdots$$

而 $\dot{y}(t_k) = f(t_k, y(t_k))$

舍去二次项，即得到欧拉法的公式

$$y_{k+1} = y_k + h \cdot f(t_k, y_k)$$

所以，欧拉法的截断误差是 $O(h^2)$

在时间范围 $[t_0, t_f]$ 内总共需要计算的步数是 $m = \frac{t_f - t_0}{h}$

因而累计截断误差是 $m \cdot O(h^2) = O(h)$

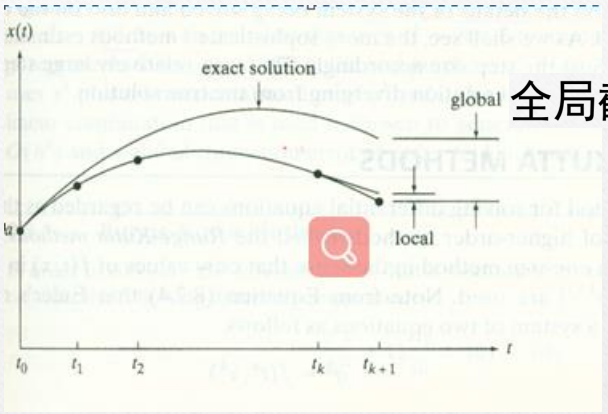
仿真误差包括：

1) 截断误差：由算法精度引起的误差，通常步长越小，截断误差越小；

2) 舍入误差：由计算机精度（有限位数）产生的误差，通常步长越小，计算次数越多，舍入误差越大。

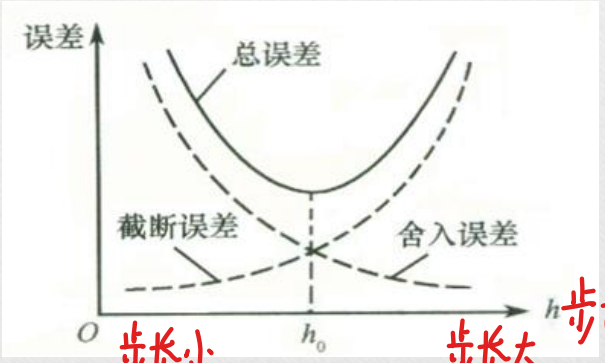
连续系统离散化原理

因而，欧拉法是一种一阶方法，精度比较低。右图表示了局部截断误差和全局截断误差的关系



全局截断误差

计算的总误差是截断误差和舍入误差的和。为了减少截断误差，需要将步长取得比较小。但如果步长取得比较小，那么在既定的仿真时间内需要计算的步数就会增多，计算步数的增多会导致舍入误差累积增大。需要在保证精度的情况下选择合适的步长。因为舍入误差一般较小，所以主要考虑的是截断误差的问题



连续系统离散化原理

欧拉法具备以下特点：

1）欧拉法实际上是采用折线代替了实际曲线，也称之为折线法。

2）欧拉法计算简单，容易实现。由前一点值仅一步递推就可以求出后一点值，所以称为单步法。

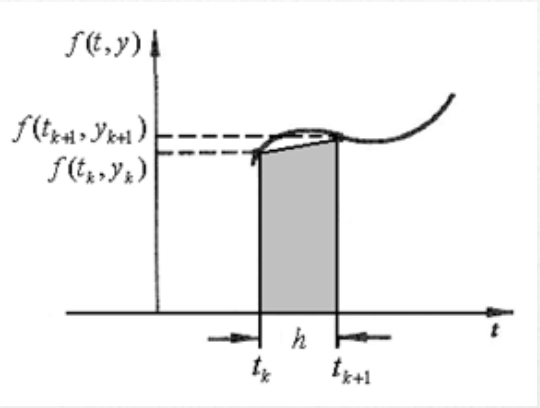
3）欧拉法计算只要给定初始值，即可开始进行递推运算，不需要其它信息，因此它属于自启动模式。

4）欧拉法是一种近似的处理，存在计算误差，所以系统的计算精度较低。欧拉法的截断误差是 $O(h^2)$

连续系统离散化原理

（3）梯形法

梯形法的思想是用梯形面积近似表示一个步长区间内的积分面积，即



$$Q_K = \frac{1}{2}h\left[f(t_k, y_k) + f(t_{k+1}, y_{k+1})\right] \qquad (8)$$

高　下底　上底(未知)

$y_{k+1} = y_k + Q_K$

(8)式在计算Q_k时用到y_{k+1}，但是y_{k+1}是未知的，因而(8)式需要进行迭代计算。一种简单的迭代是用欧拉法迭代一次，构成预估－校正法。

预估值

$$\begin{cases} y_{k+1}^{0} = y_k + hf(t_k, y_k) & (9) \\ y_{k+1} = y_k + \frac{1}{2}h\left[f(t_k, y_k) + f(t_{k+1}, y_{k+1}^{0})\right] & (10) \end{cases}$$

预估值

Q_K

(9)是预报公式，(10)是校正公式

连续系统离散化原理

梯形法具备以下特点：

1）采用梯形代替欧拉法的矩形来计算积分面积，其计算精度要高于欧拉法。

2）采用预报—校正公式，每求一个 y_k，计算量要比欧拉法多一倍。因此计算速度较慢。

3）梯形公式中的右端函数含有未知数，不能直接计算左端的变量值，这是一种隐式处理，要利用迭代法求解。即梯形法不能自启动，要靠多步法来实现计算。

截断误差正比于 h^3，记为 $O(h^3)$

龙格-库塔法

二阶龙格—库塔法

对于系统 $\begin{cases} \dot{y} = f(t, y) \\ y(t_0) = y_0 \end{cases}$ 假设其解是 $y(t)$

设 $t_{k+1} = t_k + h$ ，在 t_{k+1} 附近泰勒展开，保留 h^2 项

$$y_{k+1} = y_k + f(t_k, y_k)h + \frac{1}{2}\left(\frac{\partial f}{\partial t} + \frac{\partial f}{\partial y} \cdot \frac{dy}{dt}\right)h^2 \Big|_{t=t_k} \qquad (11)$$

$y'(t_k)$

上式截断误差是 $O(h^3)$

假设解的形式为：

$$\begin{cases} y_{k+1} = y_k + (a_1 K_1 + a_2 K_2)h \\ K_1 = f(t_k, y_k) = \frac{dy}{dt}\Big|_{t_k} \\ K_2 = f(t_k + b_1 h, y_k + b_2 K_1 h) \end{cases} \qquad (12)$$

龙格-库塔法

二元函数

将 K_2 在 t_k, y_k 处展开，保留到 h 项，得

$$K_2 \approx f(t_k, y_k) + \left(\frac{\partial f}{\partial t} b_1 + \frac{\partial f}{\partial y} b_2 K_1 \right) \Big|_{t_k} h$$

$$\begin{cases} y_{k+1} = y_k + (a_1 K_1 + a_2 K_2) h \\ K_1 = f(t_k, y_k) = \frac{dy}{dt}\Big|_{t_k} \\ K_2 = f(t_k + b_1 h, y_k + b_2 K_1 h) \end{cases}$$

将 K_1, K_2 表达式带入 y_{k+1} 表达式，得到

$$\begin{aligned} y_{k+1} &= y_k + a_1 h f(t_k, y_k) + \\ &\quad a_2 h \left[f(t_k, y_k) + \left(b_1 \frac{\partial f}{\partial t} + b_2 K_1 \frac{\partial f}{\partial y} \right) h \Big|_{t_k} \right] \\ &= y_k + (a_1 + a_2) h f(t_k, y_k) + \left(a_2 b_1 \frac{\partial f}{\partial t} + a_2 b_2 K_1 \frac{\partial f}{\partial y} \right) h^2 \Big|_{t_k} \end{aligned} \tag{14}$$

$$y_{k+1} = y_k + f(t_k, y_k) h + \frac{1}{2} \left(\frac{\partial f}{\partial t} + \frac{\partial f}{\partial y} \cdot \frac{dy}{dt} \right) h^2 \Big|_{t=t_k} \quad (11)$$

将(14)与(11)式比较，可得

注意，在(11)式中，$\left. \frac{dy}{dt} \right|_{t_0} = K_1$

龙格-库塔法

将(14)与(11)式比较，可得

$$a_1 + a_2 = 1, \quad a_2 b_1 = \frac{1}{2}, \quad a_2 b_2 = \frac{1}{2}$$

假定$a_1=a_2$, 得到一组解

$$a_1 = a_2 = \frac{1}{2}, \quad b_1 = b_2 = 1$$

$$\begin{cases} y_{k+1} = y_k + (a_1 K_1 + a_2 K_2)h \\ K_1 = f(t_k, y_k) = \left.\frac{dy}{dt}\right|_{t_k} \\ K_2 = f(t_k + b_1 h, y_k + b_2 K_1 h) \end{cases}$$

得到二阶龙格—库塔公式（RK2）

$$\text{RK2} \quad \begin{cases} y_{k+1} = y_k + \frac{h}{2}(K_1 + K_2) \\ K_1 = f(t_k, y_k) \\ K_2 = f(t_k + h, y_k + K_1 h) \end{cases}$$

RK2法与预估—校正法计算量相当

$a_1=a_2$时RK2法就是梯形法

$$\begin{cases} y_{k+1}^0 = y_k + hf(t_k, y_k) \\ y_{k+1} = y_k + \frac{1}{2}h\left[f(t_k, y_k) + f(t_{k+1}, y_{k+1}^0)\right] \end{cases}$$

龙格-库塔法

四阶龙格－库塔法（RK4） Taylor级数展开保留h^4，可得

公式会给,整计算

RK4

截断误差是$O(h^5)$

$$\begin{cases} y_{k+1} = y_k + \frac{h}{6}(K_1 + 2K_2 + 2K_3 + K_4) \\ K_1 = f(t_k, y_k) \\ K_2 = f(t_k + \frac{h}{2}, y_k + \frac{h}{2}K_1) \\ K_3 = f(t_k + \frac{h}{2}, y_k + \frac{h}{2}K_2) \\ K_4 = f(t_k + h, y_k + hK_3) \end{cases}$$

龙格－库塔法特点：

(1) 计算y_{k+1}时只用到y_k，所以是单步法。

(2) 不要求h固定，可以改变步长。但在一步计算时，h需保持不变。

(3) 精度取决于步长和方法阶次。为达到相同精度，4阶龙格库塔法比2阶龙格库塔法步长大10倍，而计算量仅大1倍。因而多采用4阶龙格库塔法。

龙格-库塔法

几种方法都是进行泰勒级数展开取近似项得到的，欧拉法实际就是一阶龙格库塔法。

h为时间间隔(步长)

算法	展开项保留阶次	截断误差级别	计算量
欧拉法 RK1	h	$O(h^2)$	1
预估一校正法 RK2	h^2	$O(h^3)$	2
RK2法	h^2	$O(h^3)$	2
RK4法	h^4	$O(h^5)$	4

龙格库塔法的一般形式

$$y_{k+1} = y_k + h\sum_{i=1}^{s} C_i K_i$$

其中

$$K_i = f\left(t_k + a_i h, y_k + h\sum_{j=1}^{i-1} b_{ij} K_j\right)$$

$$a_1 = 0$$

$$a_i = \sum_{j=1}^{i-1} b_{ij} \quad a_i < 1$$

$$\sum_{i=1}^{s} C_i = 1$$

s 称为级数
（RK4-5）

龙格-库塔法

关于数值积分法的几点讨论：

(1) 单步法和多步法：数值积分的特点是渐进式迭代。当从t_k推进到t_{k+1}时，只需要用t_k时刻的数据，就称为单步法。如欧拉法和龙格库塔法。反之，需要用到t_k以及过去时刻 t_{k-1},t_{k-2}, …的数据，就称为多步法。

（2）显示和隐式：计算y_{k+1}时，若公式右端数据均已知，就称为显示法， 否则就称为隐式法。显示法如欧拉法，龙格库塔法，隐式法如梯形法，预估－校正法。

需要预估

龙格-库塔法

$$\begin{cases} \dot{y} = f(t, y) \\ y(t_0) = y_0 \end{cases}$$

【例3.2.1】对微分方程 $\dot{y} = -2y$，用欧拉法，RK4法计算t=0.4时的y，设t=0时，y(0)=1，取步长h=0.1

解:（1）解析解

$$\frac{dy}{dt} = -2y \Rightarrow \frac{dy}{y} = -2dt \Rightarrow \int_{y_0}^{y} \frac{1}{y} dy = \int_0^t -2dt$$

$$\Rightarrow \quad \ln y - \ln y_0 = -2(t-0) \Rightarrow \ln \frac{y}{y_0} = -2t$$

$$\Rightarrow \quad y = y_0 e^{-2t} = e^{-2t}$$

$$\dot{y} = f(t, y) = -2y$$

（2）欧拉法

$$y_{k+1} = y_k + hf(t_k, y_k) = y_k + 0.1*(-2y_k) = 0.8y_k$$

龙格-库塔法

(3)RK4法

$$\dot{y} = f(t, y) = -2y$$

取f的第二个变量 x(-2)

$$K_1 = f(t_k, y_k) = -2y_k$$

$$K_2 = f(t_k + \tfrac{h}{2}, y_k + \tfrac{h}{2}K_1) = -2[y_k + \tfrac{h}{2}(-2y_k)] = -1.8y_k$$

$$K_3 = f(t_k + \tfrac{h}{2}, y_k + \tfrac{h}{2}K_2) = -2[y_k + \tfrac{h}{2}(-1.8y_k)] = -1.82y_k$$

$$K_4 = f(t_k + h, y_k + hK_3) = -2[y_k + h(-1.82y_k)] = -1.636y_k$$

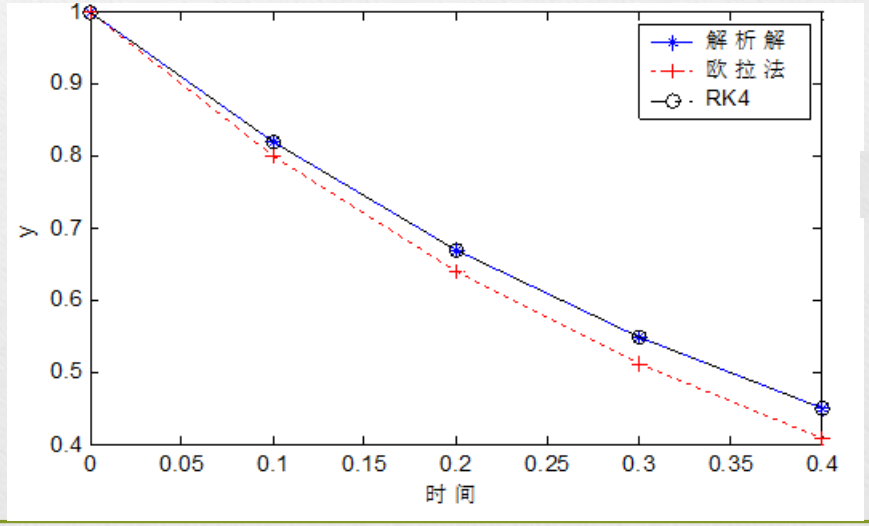
$$\therefore y_{k+1} = y_k + \tfrac{h}{6}(K_1 + 2K_2 + 2K_3 + K_4)$$

$$= y_k + \tfrac{h}{6}(-2y_1 - 3.6y_k - 3.64y_k - 1.636y_k)$$

$$= 0.8187y_k$$

龙格-库塔法

时间t	解析解	欧拉法	RK4
t=0	1	1	1
t=0.1	0.8187	0.8	0.8187
t=0.2	0.6703	0.64	0.6703
t=0.3	0.5488	0.512	0.5488
t=0.4	0.4493	0.4096	0.4493



欧拉法与解析解存在较大误差

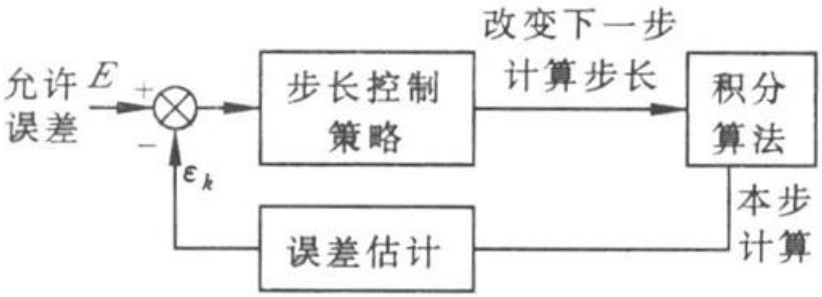
RK4与解析解的结果重合

3.2.3 龙格－库塔法的误差估计与步长控制

在保证精度的情况下，选取尽量大的步长，减少计算量。

步长控制可改变算法稳定性、精度和速度。

步长控制需要局部误差估计和步长控制策略。



龙格-库塔法步长控制示意

类似于反馈控制

每积分一步，都设法估计出本步的计算误差ε_k，然后判断是否满足允许误差E，据此选择相应的步长控制策略，调整步长。

1. 龙格－库塔法的误差估计

方法：找另一个低阶的龙格－库塔公式，两个公式计算结果之差作为估计的误差。

（1）RKM4法

高阶方法：龙格－库塔－默森(Merson)法【4阶5级公式】

$$y_{k+1} = y_k + \frac{h}{6}(K_1 + 4K_4 + K_5) \tag{17}$$

其中：

$$\begin{aligned}
K_1 &= f(t_k, y_k) \\
K_2 &= f\left(t_k + \frac{h}{3}, y_k + \frac{h}{3}K_1\right) \\
K_3 &= f\left(t_k + \frac{h}{3}, y_k + \frac{h}{6}(K_1 + K_2)\right) \\
K_4 &= f\left(t_k + \frac{h}{2}, y_k + \frac{h}{8}(K_1 + 3K_3)\right) \\
K_5 &= f\left(t_k + h, y_k + \frac{h}{2}(K_1 + 4K_4 - 3K_3)\right)
\end{aligned} \tag{18}$$

低阶方法：另一个3阶4级公式

$$\hat{y}_{k+1} = y_k + \frac{h}{6}(3K_1 - 9K_3 + 12K_4) \tag{19}$$

注：采用y_k计算，对计算结果不存储，只用于本步的误差估计

估计当前步误差为: $E_k = \hat{y}_{k+1} - y_{k+1}$

$$= \frac{h}{6}(2K_1 - 9K_3 + 8K_4 - K_5)$$

（2）RKF1－2法

高阶方法：$y_{k+1} = y_k + \frac{h}{512}(K_1 + 510K_2 + K_3)$

其中

$$K_1 = f(t_k, y_k)$$

$$K_2 = f(t_k + \frac{h}{2}, y_k + \frac{h}{2}K_1)$$

$$K_3 = f(t_k + h, y_k + \frac{h}{256}(K_1 + 255K_2))$$

低阶方法：

$$\hat{y}_{k+1} = y_k + \frac{h}{256}(K_1 + 255K_2)$$

误差为:

$$E_k = \hat{y}_{k+1} - y_{k+1} = \frac{h}{512}(K_1 - K_3)$$

（3）其他方法

RKF4－5法：5阶精度4阶误差估计，比RKM4法精确，但计算量大。

RKS4法：4阶精度3阶误差估计，计算量比RKM4稍微少一点

2. 步长控制:

（1） 加倍一减半法

定义局部误差：

$$e_k = \frac{E_k}{(|y_k|+1)} \tag{21}$$

由此定义可知：y_k较大时为相对误差，y_k较小时为绝对误差。

设定误差上限$\varepsilon_{\max}$和误差下限$\varepsilon_{\min}$

加倍一减半法步长控制策略为：

$$\begin{cases} e_k \geq \varepsilon_{\max} & \Rightarrow h_{k+1} = \frac{1}{2} h_k \\ \varepsilon_{\min} < e_k < \varepsilon_{\max} & \Rightarrow h_{k+1} = h_k \\ e_k \leq \varepsilon_{\min} & \Rightarrow h_{k+1} = 2h_k \end{cases} \tag{22}$$

应用减半步长重算该步

（2） 最优步长法：算法存在一定缺陷，需要求偏导数，易出现振荡

龙格-库塔方法是一类重要算法，但这类算法在每一步都需要先预报几个点上的斜率值，计算量比较大。考虑到计算y_{i+1}之前已得出一系列节点上的斜率值，能否利用这些已知值来减少计算

线性多步法

利用一个多项式去匹配变量的若干值和它们的导数值

已知 $t_n, t_{n+1}, \cdots, t_{n+k-1}$ 时刻的 $y_n, y_{n+1}, \cdots, y_{n+k-1}$ 和 $\dot{y}_n, \dot{y}_{n+1}, \cdots, \dot{y}_{n+k-1}$，而 $y_{n+k}, \dot{y}_{n+k}$ 未知

$$y_m(t) = \sum_{i=0}^{m} d_i \left(\frac{t_{n+k} - t}{h} \right)^i = \sum_{i=0}^{m} d_i \tau^i$$ 为m次多项式，用于拟合y(t)

$$\dot{y}_m(t) = -\frac{1}{h} \sum_{i=0}^{m} i d_i \left(\frac{t_{n+k} - t}{h} \right)^{i-1} = -\frac{1}{h} \sum_{i=0}^{m} i d_i \tau^{i-1}$$

将已知的点代入 共2k个等式

$$y_m(t_{n+k-j}) = y_{n+k-j} = \sum_{i=0}^{m} d_i \left(\frac{t_{n+k} - t_{n+k-j}}{h} \right)^i = \sum_{i=0}^{m} d_i j^i$$

$$\dot{y}_m(t_{n+k-j}) = y_{n+k-j} = -\frac{1}{h} \sum_{i=0}^{m} i d_i \left(\frac{t_{n+k} - t_{n+k-j}}{h} \right)^{i-1} = -\frac{1}{h} \sum_{i=0}^{m} i d_i j^{i-1}$$

$$t_{n+k} - t_{n+k-j} = jh$$

特殊情况：

$$y_{n+k} = d_0$$

$$\dot{y}_{n+k} = -\frac{d_1}{h}$$

预报公式

已知共k个点

由 $y_n, y_{n+1}, \cdots, y_{n+k-1}$ 和 $\dot{y}_n, \dot{y}_{n+1}, \cdots, \dot{y}_{n+k-1}$ 来预报 $y_{n+k}, \dot{y}_{n+k}$

取 $m = 2k-1$

$$
\begin{aligned}
& d_0^p + d_1^p + d_2^p + \cdots + d_m^p = y_{n+k-1} \\
& d_0^p + 2d_1^p + 4d_2^p + \cdots + 2^m d_m^p = y_{n+k-2} \\
& d_0^p + 3d_1^p + 3^2 d_2^p + \cdots + 3^m d_m^p = y_{n+k-3} \\
& \vdots \\
& d_0^p + kd_1^p + k^2 d_2^p + \cdots + k^m d_m^p = y_n
\end{aligned}
$$

$$
\begin{aligned}
& d_1^p + 2d_2^p + \cdots + md_m^p = -h\dot{y}_{n+k-1} \\
& d_1^p + 2\times 2d_2^p + \cdots + m\times 2^{m-1} d_m^p = -h\dot{y}_{n+k-2} \\
& d_1^p + 2\times 3d_2^p + \cdots + m\times 3^{m-1} d_m^p = -h\dot{y}_{n+k-3} \\
& \vdots \\
& d_1^p + 2\times kd_2^p + \cdots + m\times k^{m-1} d_m^p = -h\dot{y}_n
\end{aligned}
$$

写成矩阵形式

共2K个等式

m+1个未知数

$$\begin{bmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & 2 & 2^2 & 2^3 & \cdots & 2^m \\ 1 & 3 & 3^2 & 3^3 & \cdots & 3^m \\ \vdots & & & \vdots & & \vdots \\ 1 & k & k^2 & k^3 & \cdots & k^m \\ 0 & 1 & 2 & 3 & \cdots & m \\ 0 & 1 & 2\times 2 & 3\times 2^2 & \cdots & m\times 2^{m-1} \\ 0 & 1 & 2\times 3 & 3\times 3^2 & \cdots & m\times 3^{m-1} \\ \vdots & & & \vdots & & \vdots \\ 0 & 1 & 2\times k & 3\times k^2 & \cdots & m\times k^{m-1} \end{bmatrix} \begin{bmatrix} d_0^p \\ d_1^p \\ d_2^p \\ \vdots \\ d_{k-1}^p \\ d_k^p \\ d_{k+1}^p \\ d_{k+2}^p \\ \vdots \\ d_m^p \end{bmatrix} = \begin{bmatrix} y_{n+k-1} \\ y_{n+k-2} \\ y_{n+k-3} \\ \vdots \\ y_n \\ -h\dot{y}_{n+k-1} \\ -h\dot{y}_{n+k-2} \\ -h\dot{y}_{n+k-3} \\ \vdots \\ -h\dot{y}_n \end{bmatrix}$$

$$V^p d^p = Z^p \longrightarrow d^p = (V^p)^{-1} Z^p \longrightarrow y_{n+k} = d_0, \dot{y}_{n+k} = -\frac{1}{h} d_1$$

缺点：每次计算得到的d向量中的各元素只需要 d_0, d_1
其他的计算是多余的 $y_{n+k} = d_0, \dot{y}_{n+k} = -\frac{1}{h} d_1$

引入 $e_1^T = (1 \quad 0 \quad \cdots \quad 0)$

$$y_{n+k} = e_1^T d^p = e_1^T (V^p)^{-1} Z^p$$

定义辅助变量 $(\phi^p)^T = e_1^T (V^p)^{-1}, (\phi^p)^T = [a_1^p, a_2^p, \cdots, a_k^p, b_1^p, b_2^p, \cdots, b_k^p]$

$$(\phi^p)^T V^p = e_1^T \Rightarrow (V^p)^T \phi^p = e_1$$

$$y_{n+k} = e_1^T d^p = (\phi^p)^T Z^p$$

得到 y_{n+k} 显示表达式 $y_{n+k} = \sum_{j=1}^{k} a_j^p y_{n+k-j} - h \sum_{j=1}^{k} b_j^p \dot{y}_{n+k-j}$

例：试用 $y_{n+k-1}, \dot{y}_{n+k-1}, \dot{y}_{n+k-2}$ 推导预报 y_{n+k} 公式

解：由于给出了三个条件 $y_{n+k-1}, \dot{y}_{n+k-1}, \dot{y}_{n+k-2}$

故用二阶多项式近似，且知 $(\phi^p)^T = (a_1^p, b_1^p, b_2^p)$

则由 $(V^p)^T \phi^p = e_1$ 可得

$$V^p = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 2 \\ 0 & 1 & 4 \end{bmatrix} \qquad (V^p)^T \phi^p = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 2 & 4 \end{bmatrix} \begin{bmatrix} a_1^p \\ b_1^p \\ b_2^p \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

$$a_1^p = 1, b_1^p = -3/2, b_2^p = 1/2$$

$$y_{n+k} = y_{n+k-1} + \frac{1}{2} h(3\dot{y}_{n+k-1} - y_{n+k-2})$$

校正公式

前面讨论的多步法预报公式是显示公式，未包含 $\dot{y}_{n+k}$

在高精度仿真时，对该预报值进行校正，即先预报得到 y_{n+k}，然后再用此值推出 $\dot{y}_{n+k}$

由 $y_n, y_{n+1}, \cdots, y_{n+k-1}$ 和 y_{n+k} 及 $\dot{y}_n, \dot{y}_{n+1}, \cdots, \dot{y}_{n+k-1}$ 来预报 $\dot{y}_{n+k}$

$m = 2k\text{-}1$

取 $m=2k-1$ 共2k+1个等式

$$d_0^c = y_{n+k}$$

$$d_0^c + d_1^c + d_2^c + \cdots + d_m^c = y_{n+k-1}$$

$$d_0^c + 2d_1^c + 4d_2^c + \cdots + 2^m d_m^c = y_{n+k-2}$$

$$d_0^c + 3d_1^c + 3^2 d_2^c + \cdots + 3^m d_m^c = y_{n+k-3}$$

$$\vdots$$

$$d_0^c + kd_1^c + k^2 d_2^c + \cdots + k^m d_m^c = y_n$$

$$d_1^c + 2d_2^c + \cdots + md_m^c = -h\dot{y}_{n+k-1}$$

$$d_1^c + 2\times 2d_2^c + \cdots + m\times 2^{m-1} d_m^c = -h\dot{y}_{n+k-2}$$

$$d_1^c + 2\times 3d_2^c + \cdots + m\times 3^{m-1} d_m^c = -h\dot{y}_{n+k-3}$$

$$\vdots$$

$$d_1^c + 2\times kd_2^c + \cdots + m\times k^{m-1} d_m^c = -h\dot{y}_n$$

矩阵形式：

$$
\begin{bmatrix}
 & 1 & 0 & 0 & 0 & \cdots & 0 \\
1 & 1 & 1 & 1 & \cdots & 1 \\
1 & 2 & 2^2 & 2^3 & \cdots & 2^m \\
1 & 3 & 3^2 & 3^3 & \cdots & 3^m \\
\vdots & & & \vdots & & \vdots \\
1 & k & k^2 & k^3 & \cdots & k^m \\
0 & 1 & 2 & 3 & \cdots & m \\
0 & 1 & 2\times 2 & 3\times 2^2 & \cdots & m\times 2^{m-1} \\
0 & 1 & 2\times 3 & 3\times 3^2 & \cdots & m\times 3^{m-1} \\
\vdots & & & \vdots & & \vdots \\
0 & 1 & 2\times k & 3\times k^2 & \cdots & m\times k^{m-1}
\end{bmatrix}
\begin{bmatrix}
d_0^c \\ d_1^c \\ d_2^c \\ \vdots \\ d_{k-1}^c \\ d_k^c \\ d_{k+1}^c \\ d_{k+2}^c \\ \vdots \\ d_m^c
\end{bmatrix}
=
\begin{bmatrix}
y_{n+k} \\ y_{n+k-1} \\ y_{n+k-2} \\ y_{n+k-3} \\ \vdots \\ y_n \\ -h\dot{y}_{n+k-1} \\ -h\dot{y}_{n+k-2} \\ -h\dot{y}_{n+k-3} \\ \vdots \\ -h\dot{y}_n
\end{bmatrix}
$$

引入 $e_2^T = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \end{pmatrix}$

$$-h\dot{y}_{n+k} = e_2^T d^c = e_2^T (V^c)^{-1} Z^c$$

定义辅助变量 $(\phi^c)^T = e_2^T (V^c)^{-1}$ $(\phi^p)^T = [a_0^c, a_1^c, \cdots, a_k^c, b_1^c, b_2^c, \cdots, b_k^c]$

$$(\phi^c)^T V^c = e_2^T \Rightarrow (V^c)^T \phi^c = e_2$$

$$-h\dot{y}_{n+k} = e_2^T d^c = (\phi^c)^T Z^c$$

得到 $\dot{y}_{n+k}$ 显示表达式

$$\dot{y}_{n+k} = -\frac{1}{h}\left(\sum_{j=0}^{k} a_j^c y_{n+k-j} - h\sum_{j=1}^{k} b_j^c \dot{y}_{n+k-j} \right)$$

会考

例：首先用已知值 $y_{n+k-1}, y_{n+k-2}, y_{n+k-3}$ 预估 y_{n+k}

然后用 $y_{n+k}, y_{n+k-1}, y_{n+k-2}$ ， 推导 $\dot{y}_{n+k}$ 的校正公式

解：预估公式的矩阵表达式 矩阵是固定的

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 3 & 9 \end{bmatrix} \begin{bmatrix} d_0^p \\ d_1^p \\ d_2^p \end{bmatrix} = \begin{bmatrix} y_{n+k-1} \\ y_{n+k-2} \\ y_{n+k-3} \end{bmatrix} \qquad \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \\ 1 & 4 & 9 \end{bmatrix} \begin{bmatrix} a_1^p \\ a_2^p \\ a_3^p \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

得到 $(\phi^p)^T = [a_1^p, a_2^p, a_3^p] = [3, -3, 1]$

$$y_{n+k} = 3y_{n+k-1} - 3y_{n+k-2} + y_{n+k-3}$$

校正公式矩阵表达式

$$\begin{bmatrix}1 & 0 & 0\\ 1 & 1 & 1\\ 1 & 2 & 4\end{bmatrix}\begin{bmatrix}d_0^c\\ d_1^c\\ d_2^c\end{bmatrix}=\begin{bmatrix}y_{n+k}\\ y_{n+k-1}\\ y_{n+k-2}\end{bmatrix}\qquad \begin{bmatrix}1 & 1 & 1\\ 0 & 1 & 2\\ 0 & 1 & 4\end{bmatrix}\begin{bmatrix}a_0^c\\ a_1^c\\ a_2^c\end{bmatrix}=\begin{bmatrix}0\\ 1\\ 0\end{bmatrix}$$

$$(\phi^c)^T=[\mathrm{a}_0^c,\mathrm{a}_1^c,\mathrm{a}_2^c]=[-\frac{3}{2},2,\frac{1}{2}]$$

校正公式

$$\dot{y}_{n+k}=-\frac{1}{h}(-\frac{3}{2}\mathrm{y}_{n+k}+2\,\mathrm{y}_{n+k-1}-\frac{1}{2}\mathrm{y}_{n+k-2})$$

预报公式和校正公式可以统一为如下表达式

$$\sum_{i=0}^{k}\alpha_i y_{n+k-i}-h\sum_{j=0}^{k}\beta_j \dot{y}_{n+k-j}=0$$

当 $\alpha_0=0$ 时，显示预报 $\dot{y}_{n+k}$

当 $\beta_0=0$ 时，显示预报 y_{n+k}

当 $\alpha_0,\beta_0\neq 0$ 时，称为隐式校正公式

典型的Adams公式

$$y_{n+k}=y_{n+k-1}+h\sum_{j=0}^{k}b_j\dot{y}_{n+k-j}$$

$b_0=0$ 显示公式

$b_0\neq 0$ 隐式公式

利用插值原理计算数值积分

$$y_{k+1} = y_k + Q_k \qquad Q_k = \int_{t_k}^{t_{k+1}} f(t,y)dt$$

一般地，利用插值原理所建立的一系列数值积分方法也可以导出解微分方程的一系列计算公式。运用插值方法的关键在于选取合适的插值节点。假设已构造出$f(t,y(t))$的插值多项式$P_r(t)$,则

$$Q_k = \int_{t_k}^{t_{k+1}} f(t,y)dt \approx \int_{t_k}^{t_{k+1}} p_r(t)dt$$

$$y_{k+1} = y_k + \int_{t_k}^{t_{k+1}} p_r(t)dt$$