Lecture 2

Zed

February 23, 2017

1 Lagrange Interpolation

Motivation: for the problem

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

we want to construct a method of *any* order p. When using the theta method with whatever linear combination of only two points (one-step method), we can only achieve order 2. Therefore it is necessary to use more than two points.

Def. **Lagrange Interpolation**: We want to appoximate $g(t)$ which we know $n+1$ values at $t_0 < t_1 < t_2 < ... < t_n$. Define

$$p(t) := \sum_{k=0}^{n} g(t_k)L_k(t), \quad \text{where} \quad L_k(t) := \prod_{j=0, j\neq k}^{n} \frac{t_j - t}{t_j - t_k}$$

L_k is called the Lagrange interp polynomial. $p(t)$ is called the Lagrange interp *of order* n, using $\{L_k\}_0^n$ and the $n+1$ values of g.

Thm. If $g \in C^\infty$, $p(t)$ is Lagrange interp of order n then

$$g(t) - p(t) = \frac{1}{(n+1)!} g^{(n+1)}(\xi) \prod_{k=0}^{n} (t - t_k), \quad t_0 \leq \xi \leq t_n$$

Further, if assuming $t_i - t_{i-1} = h$, then $\prod_{k=0}^{n} \leq (n+1)! h^{n+1}$, hence

$$g(t) - p(t) \leq \max_{t_0 \leq \xi \leq t_n} |g^{(n+1)}(\xi)| \cdot h^{n+1} \sim O(h^{n+1})$$

2 Multistep Methods

Consider the same IVP, $y(t)$ is its solution. Suppose we know the value of $y(t)$ at s points: $t_n, t_{n+1}, ..., t_{n+s-1}$ (so we also know the value of $y'(t)$ at these points), and we want to approximate $y(t_{n+s})$. We begin with the integral formula from t_{n+s-1} to t_{n+s}:

$$\begin{aligned}
y(t_{n+s}) &= y(t_{n+s-1}) + \int_{t_{n+s-1}}^{t_{n+s}} y'(t)dt \\
&= y(t_{n+s-1}) + \int_{t_{n+s-1}}^{t_{n+s}} p(t)dt + O(h^{s+1}) \\
&= y(t_{n+s-1}) + \int_{t_{n+s-1}}^{t_{n+s}} \sum_{k=n}^{n+s-1} y'(t_k)L_k(t)dt + O(h^{s+1}) \\
&= y(t_{n+s-1}) + \sum_{k=n}^{n+s-1} y'(t_k) \int_{t_{n+s-1}}^{t_{n+s}} L_k(t)dt + O(h^{s+1}) \quad (\dagger)
\end{aligned} \tag{1}$$

We hope that $\int_{t_{n+s-1}}^{t_{n+s}} L_k(t)dt$ is a constant times h. Actually we have

$$\int_{t_{n+s-1}}^{t_{n+s}} L_k(t)dt = \int_{t_{s-1}}^{t_s} L_{k'}(t)dt$$

where $k = n, n+1, ..., n+s-1; k' = 0, 1, ..., s-1$. We find this by translating the lattice $\{t_k\}$ to the left by nh. So we just use k in the following text with $k = 0, 1, ..., s-1$. And we denote

$$h\left(\frac{1}{h}\int_{t_{s-1}}^{t_s} L_k(t)dt\right) = hc_k$$

where c_k is indeed a constant. Therefore

$$(\dagger) : y(t_{n+s}) = y(t_{n+s-1}) + h\sum_{k=0}^{s-1} c_k f(t_{k+n}, y(t_{k+n})) + O(h^{s+1})$$

Def. **Multistep Method**: we define the numerical method using $(\dagger)$, by substituting $y(t_k)$ with discrete approx y_k, i.e.

$$y_{n+s} = y_{n+s-1} + h\sum_{k=0}^{s-1} c_k \cdot f(t_{k+n}, y_{k+n}) \tag{2}$$

$$c_k (= c_k^{[s]}) = \frac{1}{h}\int_{t_{s-1}}^{t_s} L_k^{[s]}(t)dt = \frac{1}{h}\int_{t_{s-1}}^{t_s} \prod_{j=0, j\neq k}^{s-1} \frac{t_j - t}{t_j - t_k} dt \tag{3}$$

The iterative formula (2) is called the multistep method with order s. By definition, $(\dagger)$ gives the truncation error of this method, which is $R_k \sim O(h^s)$ when there are s known values of $y(t)$, and the second term is an $(s-1)^{th}$ ordered Lagrange interpolation.
Note that parameter $\{c_k\}$ is a function of k and s (the order of the method). For the methods of different order, c_k's have different values. We say $\{c_k^{[s]}\}_{k=0}^{s-1}$ in the following text to avoid ambiguity.

Ex. 1^{st} order multistep:

$$c_0^{[1]} = \int_0^1 L_0^{[1]}(t)dt = \int_0^1 1dt = 1$$
$$y_{n+1} = y_n + hf(t_n, y_n)$$

Degenerates to the Euler method.

Ex. 2^{nd} order multistep:

$$c_0^{[2]} = \int_1^2 L_0^{[2]}(t)dt = \int_1^2 \frac{1-t}{1-0}dt = -\frac{1}{2}$$
$$c_1^{[2]} = \int_1^2 L_1^{[2]}(t)dt = \int_1^2 \frac{0-t}{0-1}dt = \frac{3}{2}$$
$$y_{n+2} = y_{n+1} + h\left[-\frac{1}{2}f(t_n, y_n) + \frac{3}{2}f(t_{n+1}, y_{n+1})\right]$$

Ex. 3^{rd} order multistep:

$$c_0^{[3]} = \int_2^3 L_0^{[3]}(t)dt = \int_2^3 \frac{1-t}{1-0}\cdot\frac{2-t}{2-0}dt = \frac{5}{12}$$
$$c_1^{[3]} = \int_2^3 L_1^{[3]}(t)dt = \int_2^3 \frac{0-t}{0-1}\cdot\frac{2-t}{2-1}dt = -\frac{3}{4}$$

$$c_2^{[3]} = \int_2^3 L_2^{[3]}(t)dt = \int_2^3 \frac{0-t}{0-2} \cdot \frac{1-t}{1-2} dt = \frac{23}{12}$$

$$y_{n+3} = y_{n+2} + h\left[\frac{5}{12} f(t_n, y_n) - \frac{4}{3} f(t_{n+1}, y_{n+1}) + \frac{23}{12} f(t_{n+2}, y_{n+2})\right]$$

The complexity of computing parameters $\{c_k^{[s]}\}$ increases with the order, but we can hardcode the values in the program anyway. The running time of the methods will not involve the computation of $\{c_k^{[s]}\}$.

3 General Formulation of Multistep Methods

We propose a general formulation instead of turning to Lagrange interpolation:

$$\sum_{k=0}^{s} a_k y_{n+k} = h \sum_{k=0}^{s} b_k f(t_{n+k}, y_{n+k}) \tag{4}$$

where $\{a_k\}_{k=0}^s$, $\{b_k\}_{k=0}^s$ are unsolved constants (parameters), independent wrt h, n or the ODE. Let $a_s = 1$, we can obtain an explicit method iff $b_s = 0$:

$$y_{n+s} = -\sum_{k=0}^{s-1} a_k y_{n+k} + h \sum_{k=0}^{s-1} b_k f(t_{n+k}, y_{n+k})$$

Define

$$\psi(n, y) := \sum_{k=0}^{s} a_k y(t_{n+k}) - h \sum_{k=0}^{s} b_k f(t_{n+k}, y(t_{n+k})) \tag{5}$$

By defintion we want $\psi(n, y) \sim O(h^{p+1})$, then the method is of order p, $1 \le p \le s$. With Taylor expansion of $y(t_{n+k})$ and $y'(t_{n+k})$ at t_n, we have

$$\begin{aligned}
\psi(n, y) &= \sum_{k=0}^{s} a_k \sum_{m=0}^{\infty} \left(y^{(m)}(t_n) \cdot \frac{(kh)^m}{m!} \right) - h \sum_{k=0}^{s} b_k \sum_{m=0}^{\infty} \left(y^{(m+1)}(t_n) \cdot \frac{(kh)^m}{m!} \right) \\
&= \sum_{m=0}^{\infty} \frac{h^m y^{(m)}(t_n)}{m!} \sum_{k=0}^{s} a_k k^m - \sum_{m=1}^{\infty} \frac{h^m y^{(m)}(t_n)}{(m-1)!} \sum_{k=0}^{s} b_k k^{m-1} \\
&= y(t_n) \sum_{k=0}^{s} a_k + \sum_{m=1}^{\infty} \frac{h^m y^{(m)}(t_n)}{(m-1)!} \sum_{k=0}^{s} (a_k k^m - m b_k k^{m-1})
\end{aligned} \tag{6}$$

It is clear that the method is of order p if the $m = 0, 1, ..., p$ order of h^m shrink, i.e. The method (4) is of order $p \iff$

$$\begin{aligned}
&(a) \quad \sum_{k=0}^{s} a_k = 0 \\
&(b) \quad \sum_{k=0}^{s} (a_k k^m - m b_k k^{m-1}) = 0 \quad \text{for } m = 1, 2, ..., p \\
&(c) \quad \sum_{k=0}^{s} (a_k k^m - m b_k k^{m-1}) \ne 0 \quad \text{for } m = p+1 \text{ (or higher)}
\end{aligned} \tag{7}$$

Now we consider these parameters, define

$$c_0 := \sum_{k=0}^{s} a_k, \quad c_m := \frac{1}{m!} \sum_{k=0}^{s} (a_k k^m - m b_k k^{m-1})$$

The **generating polynomial** of $\{c_m\}$ is:

$$P(z) := \sum_{m=0}^{\infty} c_m z^m$$

$$\begin{aligned}
\sum_{m=0}^{\infty} c_m z^m &= \sum_{k=0}^{s} a_k z^0 + \sum_{m=1}^{\infty} \frac{z^m}{m!} \sum_{k=0}^{s} (a_k k^m - m b_k k^{m-1}) \\
&= \sum_{k=0}^{s} a_k \left(1 + \sum_{m=1}^{\infty} \frac{(kz)^m}{m!}\right) + \sum_{k=0}^{s} b_k z \sum_{m=1}^{\infty} \frac{(kz)^{m-1}}{(m-1)!} \\
&= \sum_{k=0}^{s} a_k \sum_{m=0}^{\infty} \frac{(kz)^m}{m!} + \sum_{k=0}^{s} b_k z \sum_{m=0}^{\infty} \frac{(kz)^m}{m!} \\
&= \sum_{k=0}^{s} a_k e^{kz} + \sum_{k=0}^{s} b_k z e^{kz}
\end{aligned} \tag{8}$$

Which is a powerful result:

$$P(z) = \sum_{m=0}^{\infty} c_m z^m = \sum_{k=0}^{s} a_k e^{kz} + \sum_{k=0}^{s} b_k z e^{kz}$$