Numerical Analysis – Lecture 11[1]

Method 3.18 For reasons that will be made clear in the sequel, we wish to consider s-step, s-order methods s.t. $\sigma(w) = \sigma_s w^s$ for some $\sigma_s \in \mathbb{R} \setminus \{0\}$. In other words,

$$\sum_{l=0}^{s} \rho_l \boldsymbol{y}_{n+l} = h\sigma_s \boldsymbol{f}(t_{n+s}, \boldsymbol{y}_{n+s}), \qquad n = 0, 1, \ldots.$$

Such methods are called *backward differentiation formulae (BDF)*.

Lemma 3.19 *The explicit form of the s-step BDF method is*

$$\rho(w) = \sigma_s \sum_{l=1}^{s} \frac{1}{l} w^{s-l}(w-1)^l, \qquad \text{where} \qquad \sigma_s = \left(\sum_{l=1}^{s} \frac{1}{l}\right)^{-1}. \tag{3.9}$$

Proof Set $v = w^{-1}$, therefore the order condition $\rho(w) = \sigma_s w^s \log w + \mathcal{O}\big(|w-1|^{s+1}\big)$ becomes

$$\sum_{l=0}^{s} \rho_l v^{s-l} = -\sigma_s \log v + \mathcal{O}\big(|v-1|^{s+1}\big), \qquad v \to 1.$$

But $\log v = \log(1 + (v-1)) = \sum_{l=1}^{\infty} (-1)^{l-1}(v-1)^l/l$, consequently

$$\sum_{l=0}^{s} \rho_{s-l} v^l = \sigma_s \sum_{l=1}^{s} \frac{(-1)^l}{l}(v-1)^l.$$

Brief manipulation and a restoration of $w = v^{-1}$ yield

$$\sum_{l=0}^{s} \rho_l w^l = \sigma_s \sum_{l=1}^{s} \frac{(-1)^l}{l} w^{s-l}(1-w)^l$$

and we pick σ_s so that $\rho_s = 1$, collecting powers of w^s on the right of the last displayed equation. $\square$

Example 3.20 Let $s = 2$. Substitution in (3.9) yields $\sigma_2 = \frac{2}{3}$ and simple algebra results in $\rho(w) = w^2 - \frac{4}{3}w + \frac{1}{3}$. Hence the 2-step BDF is

$$\boldsymbol{y}_{n+2} - \tfrac{4}{3}\boldsymbol{y}_{n+1} + \tfrac{1}{3}\boldsymbol{y}_n = \tfrac{2}{3}h\boldsymbol{f}(t_{n+2}, \boldsymbol{y}_{n+2}).$$

Remark 3.21 We cannot take it for granted that BDF methods are convergent. It is possible to prove that they are convergent iff $s \le 6$. They *must not* be used outside this range!

Revision 3.22 We may approximate

$$\int_0^h f(t)\mathrm{d}t \approx h \sum_{l=1}^{\nu} b_l f(c_l h),$$

where the weights b_l are chosen in accordance with an explicit formula from Part IB. This *quadrature formula* is exact for all polynomials of degree $\nu - 1$ and, provided that $\prod_{k=1}^{\nu}(x - c_k)$ is orthogonal w.r.t. the weight function $w(x) \equiv 1$, $0 \le x \le 1$, the formula is exact for all polynomials of degree $2\nu - 1$.

[1] Please email all corrections and suggestions to these notes to `A.Iserles@damtp.cam.ac.uk`. All handouts are available on the WWW at the URL `http://www.damtp.cam.ac.uk/user/na/PartII/Handouts.html`.

Methods 3.23 Suppose that we wish to solve the 'ODE' $y' = f(t)$, $y(0) = y_0$. The exact solution is $y(t_{n+1}) = y(t_n) + \int_{t_n}^{t_{n+1}} f(t)\mathrm{d}t$ and we can approximate it by quadrature. In general, we obtain the time-stepping scheme

$$y_{n+1} = y_n + h\sum_{l=1}^{\nu} b_l f(t_n + c_l h) \qquad n = 0, 1, \ldots.$$

Here $h = t_{n+1} - t_n$ (the points t_n need not be equispaced). Can we generalize this to genuine ODEs of the form $\boldsymbol{y}' = \boldsymbol{f}(t, \boldsymbol{y})$? Formally,

$$\boldsymbol{y}(t_{n+1}) = \boldsymbol{y}(t_n) + \int_{t_n}^{t_{n+1}} \boldsymbol{f}(t, \boldsymbol{y}(t))\mathrm{d}t,$$

and this can be 'approximated' by

$$\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + h\sum_{l=1}^{\nu} b_l \boldsymbol{f}(t_n + c_l h, \boldsymbol{y}(t_n + c_l h)). \tag{3.10}$$

except that, of course, the vectors $\boldsymbol{y}(t_n + c_l h)$ are unknown! *Runge–Kutta methods* are a means of implementing (3.10) by replacing unknown values of $\boldsymbol{y}$ by suitable linear combinations. The general form of a *ν-stage explicit Runge–Kutta method (RK)* is

$$\begin{aligned}
\boldsymbol{k}_1 &= \boldsymbol{f}(t_n, \boldsymbol{y}_n), \\
\boldsymbol{k}_2 &= \boldsymbol{f}(t_n + c_2 h, \boldsymbol{y}_n + hc_2\boldsymbol{k}_1), \\
\boldsymbol{k}_3 &= \boldsymbol{f}(t_n + c_3 h, \boldsymbol{y}_n + h(a_{3,1}\boldsymbol{k}_1 + a_{3,2}\boldsymbol{k}_2)), \qquad a_{3,1} + a_{3,2} = c_3, \\
&\ \vdots \\
\boldsymbol{k}_\nu &= \boldsymbol{f}\left(t_n + c_\nu h, \boldsymbol{y}_n + h\sum_{j=1}^{\nu-1} a_{\nu,j}\boldsymbol{k}_j\right), \qquad \sum_{j=1}^{\nu-1} a_{\nu,j} = c_\nu, \\
\boldsymbol{y}_{n+1} &= \boldsymbol{y}_n + h\sum_{l=1}^{\nu} b_l \boldsymbol{k}_l.
\end{aligned}$$

The choice of the *RK coefficients* $a_{l,j}$ is motivated at the first instance by order considerations.

Example 3.24 Set $\nu = 2$. We have $\boldsymbol{k}_1 = \boldsymbol{f}(t_n, \boldsymbol{y}_n)$ and, Taylor-expanding about $(t_n, \boldsymbol{y}_n)$,

$$\begin{aligned}
\boldsymbol{k}_2 &= \boldsymbol{f}(t_n + c_2 h, \boldsymbol{y}_n + c_2 h\boldsymbol{f}(t_n, \boldsymbol{y}_n)) \\
&= \boldsymbol{f}(t_n, \boldsymbol{y}_n) + hc_2\left[\frac{\partial \boldsymbol{f}(t_n, \boldsymbol{y}_n)}{\partial t} + \frac{\partial \boldsymbol{f}(t_n, \boldsymbol{y}_n)}{\partial \boldsymbol{y}}\boldsymbol{f}(t_n, \boldsymbol{y}_n)\right] + \mathcal{O}\big(h^2\big).
\end{aligned}$$

But

$$\boldsymbol{y}' = \boldsymbol{f}(t, \boldsymbol{y}) \qquad \Rightarrow \qquad \boldsymbol{y}'' = \frac{\partial \boldsymbol{f}(t, \boldsymbol{y})}{\partial t} + \frac{\partial \boldsymbol{f}(t, \boldsymbol{y})}{\partial \boldsymbol{y}}\boldsymbol{f}(t, \boldsymbol{y}).$$

Therefore, substituting the exact solution $\boldsymbol{y}_n = \boldsymbol{y}(t_n)$, we obtain $\boldsymbol{k}_1 = \boldsymbol{y}'(t_n)$ and $\boldsymbol{k}_2 = \boldsymbol{y}'(t_n) + hc_2\boldsymbol{y}''(t_n) + \mathcal{O}\big(h^2\big)$. Consequently, the *local* error is

$$\begin{aligned}
\boldsymbol{y}(t_{n+1}) - \boldsymbol{y}_{n+1} = {} & [\boldsymbol{y}(t_n) + h\boldsymbol{y}'(t_n) + \tfrac{1}{2}h^2\boldsymbol{y}''(t_n) + \mathcal{O}\big(h^3\big)] \\
& - [\boldsymbol{y}(t_n) + h(b_1 + b_2)\boldsymbol{y}'(t_n) + h^2 b_2 c_2 \boldsymbol{y}''(t_n) + \mathcal{O}\big(h^3\big)].
\end{aligned}$$

We deduce that the RK method is of order 2 if $b_1 + b_2 = 1$ and $b_2 c_2 = \frac{1}{2}$. It is easy to demonstrate that no such method may be of order ≥ 3 (e.g. by applying it to $y' = \lambda y$).