Lecture 6

Runge-Kutta Methods

In the previous lectures, we have concentrated on multi-step methods. However, another powerful set of methods are known as multi-stage methods. Perhaps the best known of multi-stage methods are the Runge-Kutta methods. In this lecture, we give some of the most popular Runge-Kutta methods and briefly discuss their properties.

6.1 Two-stage Runge-Kutta Methods

A popular two-stage Runge-Kutta method is known as the modifed Euler method:

$$\begin{aligned} a &= \Delta t f(v^n, t^n) \\ b &= \Delta t f(v^n + a/2, t^n + \Delta t/2) \\ v^{n+1} &= v^n + b \end{aligned}$$

Another popular two-stage Runge-Kutta method is known as the Heun method:

$$\begin{aligned} a &= \Delta t f(v^n, t^n) \\ b &= \Delta t f(v^n + a, t^n + \Delta t) \\ v^{n+1} &= v^n + \frac{1}{2}(a + b) \end{aligned}$$

As can been seen with either of these methods, f is evaluated twice in finding the new value of v^{n+1}: once to determine a and once to determine b. Both of these methods are second-order accurate, $p = 2$.

6.2 Four-stage Runge-Kutta Method

The most popular form of a four-stage Runge-Kutta method is:

$$\begin{aligned} a &= \Delta t f(v^n, t^n) \\ b &= \Delta t f(v^n + a/2, t^n + \Delta t/2) \\ c &= \Delta t f(v^n + b/2, t^n + \Delta t/2) \end{aligned}$$

$$\begin{aligned} d &= \Delta t f(v^n + c, t^n + \Delta t) \\ v^{n+1} &= v^n + \frac{1}{6}(a + 2b + 2c + d) \end{aligned}$$

Note that this method requires four evaluations of f per iteration. This method is fourth-order accurate, $p = 4$.

6.3 Stability Regions

The eigenvalue stability regions for Runge-Kutta methods can be found using essentially the same approach as for multi-step methods. Specifically, we consider a linear problem in which $f = \lambda u$ where λ is a constant. Then, we determine the amplification factor $g = g(\lambda\Delta t)$. For example, let's look at the modified Euler method,

$$\begin{aligned} a &= \Delta t \lambda v^n \\ b &= \Delta t \lambda \left(v^n + \Delta t \lambda v^n/2\right) \\ v^{n+1} &= v^n + \Delta t \lambda \left(v^n + \Delta t \lambda v^n/2\right) \\ v^{n+1} &= \left[1 + \Delta t \lambda + \frac{1}{2}(\Delta t \lambda)^2\right] v^n \\ \Rightarrow g &= 1 + \lambda\Delta t + \frac{1}{2}(\lambda\Delta t)^2 \end{aligned}$$

A similar derivation for the four-stage scheme shows that,

$$g = 1 + \lambda\Delta t + \frac{1}{2}(\lambda\Delta t)^2 + \frac{1}{6}(\lambda\Delta t)^3 + \frac{1}{24}(\lambda\Delta t)^4.$$

When analyzing multi-step methods, the next step would be to determine the locations in the $\lambda\Delta t$-plane of the stability boundary (i.e. where $|g| = 1$). This however is not easy for Runge-Kutta methods and would require the solution of a higher-order polynomial for the roots. Instead, the most common approach is to simply rely on a contour plotter in which the $\lambda\Delta t$-plane is discretized into a finite set of points and $|g|$ is evaluated at these points. Then, the $|g| = 1$ contour can be plotted. The following is the Matlab code which produces the stability region for the second-order Runge-Kutta methods (note: $g(\lambda\Delta t)$ is the same for both second-order methods):

```
% Specify x range and number of points
x0 = -3;
x1 =  3;
Nx = 301;

% Specify y range and number of points
y0 = -3;
y1 =  3;
Ny = 301;
```

```
% Construct mesh
xv    = linspace(x0,x1,Nx);
yv    = linspace(y0,y1,Ny);
[x,y] = meshgrid(xv,yv);

% Calculate z
z = x + i*y;

% 2nd order Runge-Kutta growth factor
g = 1 + z + 0.5*z.^2;

% Calculate magnitude of g
gmag = abs(g);

% Plot contours of gmag
contour(x,y,gmag,[1 1],'k-');
axis([x0,x1,y0,y1]);
axis('square');
xlabel('Real \lambda\Delta t');
ylabel('Imag \lambda\Delta t');
grid on;
```

The plots of the stability regions for the second and fourth-order Runge-Kutta algorithms is shown in Figure 6.1. These stability regions are larger than those of multi-step methods. In particular, the stability regions of the multi-stage schemes grow with increasing accuracy while the stability regions of multi-step methods decrease with increasing accuracy (see Appendix A).

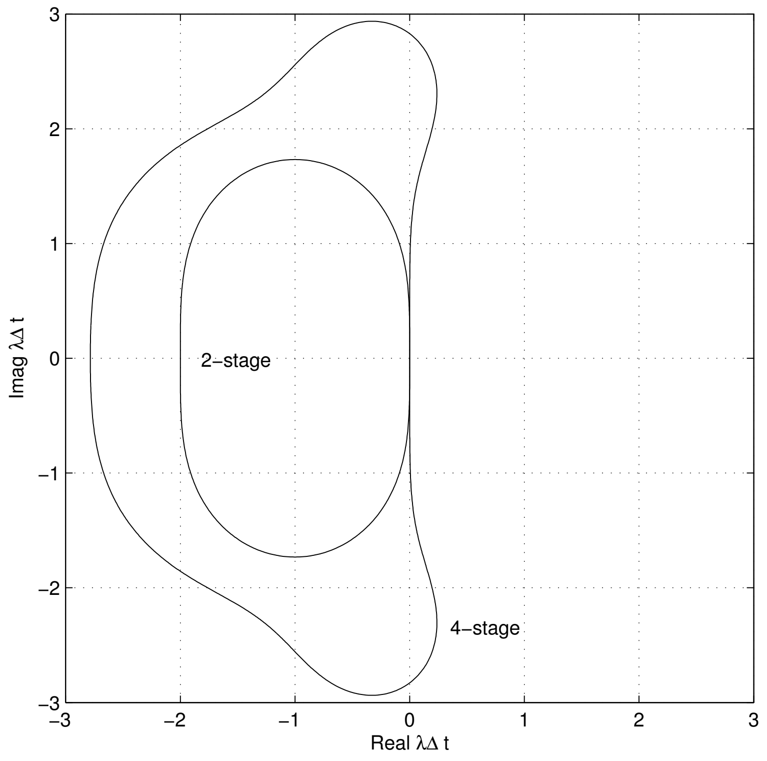

Figure 6.1: Stability boundaries for second-order and fourth-order Runge-Kutta algorithms (stable within the boundaries).