Handout # 3: Stability and Accuracy*

ME575/NSE526
Sourabh Apte

1 Stability

Stability of a system is associated with the solution of the governing equations for specified initial or boundary conditions. Here we are interested in systems where the *exact* solution (if known) is 'well-behaved' in the sense that it does not grow unbounded. Such systems are called *stable* systems. There are several cases where the exact solution itself grows in time or space e.g resonance effects, non-linear dynamics and chaos etc. However, we will focus on the former (i.e. we will concentrate on stability of numerical schemes applied to stable systems. However, we may explore problems involving growth in exact solutions using our numerical scheme to evaluate its performance under such conditions).

Numerical approximations to governing equations which have bounded exact solutions, however, may exhibit unphysical behavior, such as the numerically predicted solution grows in time. A critical analysis of the numerical scheme to ensure that the solution remains bounded is necessary. This is obtained by looking at stability of the numerical scheme. Consider a system:

$$\frac{dy}{dt} = f(y,t) \tag{1}$$

In stability analysis, we seek conditions in terms of the step size (time-step, grid spacing etc..) which will allow us to identify the limits of the numerical approximation when applied to the given differential equation. The numerical scheme can then be regarded as:

1. **Unstable Scheme**: Here the numerical solution always blows-up for any choice of parameters.
2. **Conditionally Stable**: The numerical solution is stable only when the given conditions in terms of step size are met.
3. **Unconditionally Stable**: The numerical solution is always stable for any choice of parameters.

It should be noted that when we talk about stability of a numerical scheme, it is related to a class of governing equations (differential equations) to which we apply the scheme. A scheme that is stable for certain systems may not be stable for others and vice-versa. The stability analysis not only provides the bounds on the parameters of the numerical scheme, it also allows us to choose a particular scheme for a given problem. In addition, stability of a scheme indicates its *robustness* in achieving a numerical solution.

This robustness may come with a price, typically increased complexity of the underlying numerical formulation and/or increased computational cost. A numerical analyst always has to make a judicious choice between various schemes based on his/her interests: e.g. many times one is only interested in

*This is just a review of some important concepts in numerical analysis.

the final *steady state* solution ($t \to \infty$). The manner in which the final solution is achieved (or in other words its temporal evolution) is not important. In such cases, a scheme which will allow using large time-steps to achieve faster solution is required. However, in other applications one is interested in the temporal behavior of the solution and one has to use appropriate schemes. The choice of a numerical scheme basically involves a trade-off in terms of robustness and complexity of the scheme. A simple formulation that yields a stable solution is desirable.

2 Accuracy

Accuracy of a numerical scheme is an indicator of *how well* the continuous differential equation is approximated. Majority of the numerical approximations originate from the Taylor series expansion of a function around a given initial point. This expansion involves infinite terms and it is not practical to keep all of them when we approximate. Typically, we keep a few terms in the Taylor series and drop (or truncate the series) further. The order of accuracy of a numerical approximation is then related to the order of the leading term that is truncated. It is a useful measure of accuracy because, it gives an *indication* of how rapidly the accuracy of the numerical solution can be improved with refinement of the step size (time-step, grid spacing etc.). For example, the following approximation to the first derivative of a function y is first order accurate. If we reduce the time-step by a factor of 2, the error (called the truncation error) is reduced by a factor 2.

$$\frac{dy}{dt} = \frac{y^{n+1} - y^n}{\Delta t} = f(y^n, t^n) + \emptyset(\Delta t) \tag{2}$$

Another way of assessing accuracy of a numerical scheme is by performing the modified wave number analysis (presented in Handout 1). Here we ask, how well does our numerical scheme approximate periodic functions, e.g functions used in Fourier expansions. An ideal scheme approximates these functions exactly. Any deviations from the numerical approximations are referred to as numerical errors and define the accuracy limits of a numerical scheme.

3 A General Comment

It should be noted that stability and accuracy are two different concepts. Stability ensures that the numerical solution will not grow unbounded provided the stability conditions (constraints) are met. Accuracy, on the other hand, is related to the 'estimation' of the solution. A stable scheme is not always accurate and vice-versa. Just as increased stability means increased complexity of the numerical scheme, increased accuracy may lead to increased computational costs. An appropriate trade-off is necessary based on the application.

When a new numerical scheme is presented to a true analyst, he (she) first applies the basic tools of stability analysis and accuracy estimation by solving 'model' problems with this scheme and compares it with exact solutions and/or other schemes. In many applications, the exact solution is unknown, difficult to obtain and/or measure using experimental techniques. This is the reason why we approach numerical methods in the first place. Blind use of a numerical method is dangerous and may lead to inaccurate conclusions. A sound numerical practitioner will employ standard ways of analyzing the 'predictive' capability of the method. For any given numerical scheme, decreasing the step size increases the accuracy and typically should increase its stability (the cost or time to obtain the solution may also increase). However, 'predictive' capability of a numerical scheme is tested by performing such refinement studies, obtaining the solutions, and comparing the solutions. In many cases, with step-size refinement,

the predicted solution converges to a specific value, which is regarded as the *true* solution predicted by that method. This solution may differ from the exact solution (if known), and the error is due to the inherent errors in the numerical approximations.

4 Stability Analysis

The formal procedure of stability analysis is related to the application of a particular numerical scheme to the differential equation of interest. In general, one can obtain stability regions of the numerical scheme by looking at a *model problem* representative of the actual differential equation. For example, the stability of a numerical scheme applied to a class of generalized first-order ODE (1) can be investigated at once, by looking at a linearized differential equation. This concept is similar to the analysis procedure used in 'perturbation methods' and stability analysis introduced in fluid mechanics (the Orr-Sommerfeld equations). Let us expand the function $f(y,t)$ using two-dimensional Taylor series:

$$\begin{aligned} f(y,t) = f(y_0,t_0) + (t-t_0)\frac{\partial f}{\partial t}(y_0,t_0) + (y-y_0)\frac{\partial f}{\partial y}(y_0,t_0) + \\ \frac{1}{2!}\left[(t-t_0)^2\frac{\partial^2 f}{\partial t^2}\Big|_{(y_0,t_0)} + 2(t-t_0)(y-y_0)\frac{\partial^2 f}{\partial t \partial y}\Big|_{(y_0,t_0)} + (y-y_0)\frac{\partial^2 f}{\partial y^2}\Big|_{(y_0,t_0)}\right] + ... \end{aligned} \tag{3}$$

Substituting this equation into the ODE (1) and discarding non-linear terms (terms involving higher powers of y) we get:

$$y' = \lambda y \tag{4}$$

where $\lambda = \frac{\partial f}{\partial y}\big|_{(y_0,t_0)}$. In arriving at this equation, we also discarded terms involving powers of t. The stability analysis is performed on this **model problem**. It turns out that this linearized equation captures the essential behavior of the original non-linear system as far as stability is concerned. The exact solution of this linearized equation with $y(0) = 1$ as the initial condition is, $y = y(0)e^{\lambda t} = e^{\lambda t}$, where $\lambda = \lambda_R + i\lambda_I$ is a complex number with $i = \sqrt{-1}$. We allow complex numbers for λ because many higher order differential equations when converted to a set of coupled first order ODEs, exhibit complex coefficients for λ. Such a generalized analysis will also allow us to investigate several other equations involving *partial differential equations* (PDEs), mainly because the numerical descritization schemes for PDEs are basically derived from ODEs.

4.1 Second Order RK (RK2)

As we include more terms from a Taylor series expansion, the order of accuracy of the scheme is increased. The increase in accuracy is a direct result of utilizing more information at a given point in terms of higher order derivatives. There are different ways in providing additional information about f. Runge-Kutta schemes introduced points between t_n and t_{n+1}, and evaluate f at these intermediate points. These additional function evaluations, increase the cost of the numerical method, however, also provided added accuracy. This class of schemes are called *multi-stage schemes*, mainly because of the added fictitious points and evaluations in between the time-step.

4.2 A Two-Stage Runge-Kutta (RK2) Scheme

Consider the first-order differential equation:

$$\frac{dy}{dt} = f(y,t) \tag{5}$$

In RK2, the solution at t_{n+1} is obtained by writing:

$$\begin{aligned} k_1 &= \Delta f(y_n, t_n) && (6)\\ k_2 &= \Delta f(y_n + \beta k_1, t_n + \alpha\Delta) && (7)\\ y_{n+1} &= y_n + \gamma_1 k_1 + \gamma_2 k_2 && (8) \end{aligned}$$

where γ_1, γ_2, β and α are some constants (to be determined). Notice that the RK2 involves evaluations of k_1 and k_2 which can be thought of as intermediate points between the step $\Delta = (t_{n+1} - t_n)$. In other words, going from $t = t_n$ to $t = t_{n+1}$ involves use of the function $f(y_n, t_n)$ and an intermediate point $f(y_n + \beta k_1, t_n + \alpha\Delta)$. The evaluation k_2 can be expanded in terms of a Taylor series as:

$$k_2 = \Delta f(y_n + \beta k_1, t_n + \alpha\Delta) = \Delta[f(y_n, t_n) + \beta k_1 f_{y_n} + \alpha\Delta f_{t_n} + \emptyset(\Delta^2)] \quad (9)$$

Substituting this expansion and equation 6 into equation 8 we get:

$$\begin{aligned} y_{n+1} &= y_n + \gamma_1\Delta f(y_n, t_n) + \gamma_2\Delta[f(y_n, t_n) + \beta\Delta f(y_n, t_n) f_{y_n} + \alpha\Delta f_{t_n}] && (10)\\ &= y_n + \Delta[\gamma_1 + \gamma_2] f(y_n, t_n) + \Delta^2\gamma_2[\alpha f_{t_n} + \beta f_n f_{y_n}] && (11) \end{aligned}$$

In order to evaluate the constants, let us write the Taylor series expansion of $y(t_{n+1})$ and keep terms up to the second derivative

$$y_{n+1} = y_n + \Delta y'_n + \frac{\Delta^2}{2} y''_n + ... \quad (12)$$

From the differential equation we know that $y' = f(y, t)$. Differentiating this by chain rule:

$$y'' = \frac{d}{dt}(y') = \frac{d}{dt}[f(y, t)] = \frac{\partial f}{\partial t} + \frac{\partial f}{\partial y}.\frac{dy}{dt} = f_t + f f_y \quad (13)$$

Substituting in equation 12 we get

$$y_{n+1} = y_n + \Delta f(y_n, t_n) + \frac{\Delta^2}{2}[f_{t_n} + f_n f_{y_n}] + ... \quad (14)$$

To obtain the coefficients in our RK2 scheme (equation 6-8), we compare equation 11 to the above to get:

$$\gamma_1 + \gamma_2 = 1; \quad \gamma_2\alpha = \frac{1}{2}; \quad \gamma_2\beta = \frac{1}{2} \quad (15)$$

These are three equations in four unknowns, and using α as the free parameter we get:

$$\gamma_2 = \frac{1}{2\alpha}; \quad \beta = \alpha; \quad \gamma_1 = 1 - \frac{1}{2\alpha} \quad (16)$$

This gives a one-parameter family of RK2 schemes with α as the only parameter:

$$\begin{aligned} k_1 &= \Delta f(y_n, t_n) && (17)\\ k_2 &= \Delta f(y_n + \alpha k_1, t_n + \alpha\Delta) && (18)\\ y_{n+1} &= y_n + \left(1 - \frac{1}{2\alpha}\right) k_1 + \frac{1}{2\alpha} k_2 && (19) \end{aligned}$$

$\alpha = 1/2$ is a common choice for RK2. The resulting RK2 can be written in the following two-stage predictor-corrector scheme:

$$y^p_{n+1/2} = y_n + \frac{\Delta}{2} f(y_n, t_n) \tag{20}$$

$$y_{n+1} = y_n + \Delta f(y^p_{n+1/2}, t_{n+1/2}) \tag{21}$$

Note that, this is the same as the modified midpoint Euler scheme. To consider the stability of this scheme, let us apply this scheme to our model problem:

$$\frac{dy}{dt} = \lambda y; \qquad y(0) = 1 \tag{22}$$

The resultant expression for the two-stage scheme is:

$$y^p_{n+1/2} = y_n + \frac{\Delta}{2} \lambda y_n \tag{23}$$

$$y_{n+1} = y_n + \Delta \lambda y^p_{n+1/2} \tag{24}$$

which can be written as $y_{n+1} = y_n[1 + \lambda\Delta + \frac{(\lambda\Delta)^2}{2}]$ and the corresponding amplification factor G is given as:

$$G = \frac{y_{n+1}}{y_n} = 1 + \lambda\Delta + \frac{(\lambda\Delta)^2}{2} \tag{25}$$

For stability we need $\|G\| \leq 1$. The stability boundary can be obtained by writing $\|G\| = 1 = e^{i\phi}$ and finding the roots of the complex polynomial:

$$\lambda\Delta + \frac{(\lambda\Delta)^2}{2} + 1 - e^{i\phi} = 0 \tag{26}$$

for $0 \leq \phi \leq 2\pi$. The corresponding stability curve is shown in Figure 1.

MATLAB code for RK2 stability curve

```
clear all;
clf;
format long e;
% loop over phi
for i = 1:100;
   phi = (i-1)/100 * 2. * pi;
   % define the polynomial
   C = [1/2, 1, 1 - complex(cos(phi), sin(phi))];
   % find roots
   R(1:2, i) = ROOTS(C);
end

hold on;
axis([-3*pi/2, pi/2, -3*pi/2, 3*pi/2]);
plot(R(1,:), 'm.')
plot(R(2,:), 'k.')
```

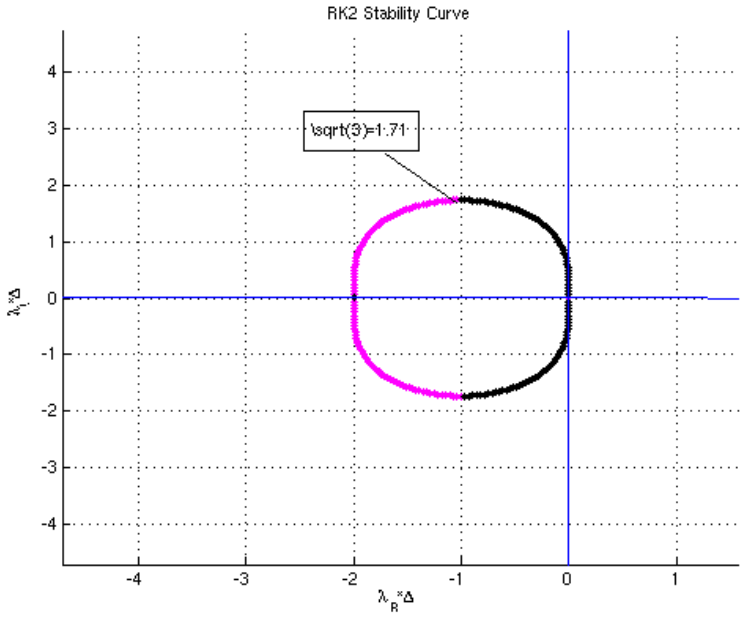

Figure 1: Stability Limit of Two-Stage RK2 Scheme with $\alpha = 1/2$. The two colors correspond to the two roots of this complex second-order polynomial.

grid;
$xlabel('\lambda_R \quad \Delta')$
$ylabel('\lambda_I \quad \Delta')$
$title('RK2 \quad Stability \quad Curve')$

4.3 Fourth Order Runge-Kutta (RK4)

The most widely used RK scheme is the fourth-order RK (RK4). This fourth order formula can be written as:

$$k_1 = \Delta f(y_n, t_n) \tag{27}$$
$$k_2 = \Delta f(y_n + \frac{k_1}{2}, t_n + \frac{\Delta}{2}) \tag{28}$$
$$k_3 = \Delta f(y_n + \frac{k_2}{2}, t_n + \frac{\Delta}{2}) \tag{29}$$
$$k_4 = \Delta f(y_n + k_3, t_n + \Delta) \tag{30}$$
$$y_{n+1} = y_n + \frac{1}{6}k_1 + \frac{1}{3}(k_2 + k_3) + \frac{1}{6}k_4 \tag{31}$$

In this scheme, four different function evaluations are required per time-step, increasing the computational cost. Applying this scheme to our model problem $y' = \lambda y$, we obtain:

$$y_{n+1} = \left(1 + \lambda\Delta + \frac{\lambda^2\Delta^2}{2} + \frac{\lambda^3\Delta^3}{6} + \frac{\lambda^4\Delta^4}{24}\right) y_n \tag{32}$$

The stability curve is obtained by requiring the amplification factor $|\sigma| = \left|\frac{y_{n+1}}{y_n}\right| = 1$. Again looking for complex roots, we can write the polynomial with complex coefficients:

$$\lambda\Delta + \frac{\lambda^2\Delta^2}{2} + \frac{\lambda^3\Delta^3}{6} + \frac{\lambda^4\Delta^4}{24} + 1 - e^{i\phi} = 0; \quad 0 \leq \phi \leq 2\pi \tag{33}$$

Following the procedure for RK2, the stability diagram can be obtained and is shown in Figure 2. Also shown are stability curves for RK2 and the explicit Euler schemes. Notice that the stability region of RK4 is considerably larger than the other two schemes, thus relaxing the constraints on the step-size. Also, RK4 shows stability regions for pure imaginary values of λ. In addition, a small stability region is shown even for $\lambda_R > 0$, where the *exact* solution actually grows! The RK4 method is artificially stable for parameters corresponding to these regions.

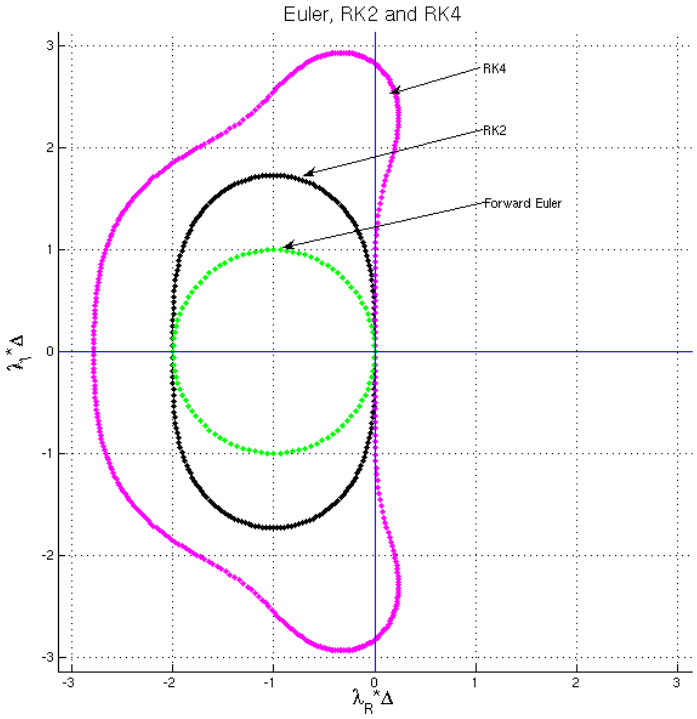

Figure 2: Stability Curves for Euler, RK2, and RK4 schemes