Chapter 8

Finite Difference Methods

Outline of Section

- Numerical differentiation
- Introduction to solving ODEs
- Consistency
- Accuracy
- Stability
- Convergence
- Efficiency

8.1 Numerical Differentiation

We are used to defining a derivative, e.g., dy/dx or $y'(x)$, as

$$\frac{dy(x)}{dx} \equiv \lim_{h\to 0} \frac{y(x+h)-y(x)}{h} \equiv \lim_{\Delta x\to 0} \frac{\Delta y}{\Delta x}\,. \tag{8.1}$$

We approach this limit on the computer by defining a **forward difference scheme (FDS)**

$$\boxed{\tilde{y}'_f(x) \equiv \frac{y(x+h)-y(x)}{h}\,.} \tag{8.2}$$

The FDS is an example of a **finite difference approximation**.

Notation: in the remainder of this course a tilde ($\sim$) over a quantity signifies that it is an approximated version, which is subject to truncation error.

By considering the Taylor expansion of $y(x+h)$ around the point x we can understand how the error in the above scheme is first order;

$$y(x+h) = y(x) + y'(x)\,h + \frac{y''(\xi)}{2}\,h^2\,,$$

where we have defined the truncation error as

$$\epsilon = \frac{y''(\xi)}{2}\,h^2,$$

where ξ is an unknown value which lies in the interval $x < \xi < x+h$. Thus we can write the first derivative as

$$y'(x) = \frac{y(x+h)-y(x)}{h} - \frac{y''(\xi)}{2}\,h = \tilde{y}'_f(x) - \mathcal{O}(h)\,. \tag{8.3}$$

So the simple forward difference scheme has error $\mathcal{O}(h)$. Notice that even if we reduce this truncation error by making h really small, in practice, the accuracy of the derivative will have a limit imposed by the round-off error of the computer.

We can also use a **backward difference scheme (BDS)**

$$\boxed{\tilde{y}'_b(x) \equiv \frac{y(x) - y(x-h)}{h}\,,} \tag{8.4}$$

but as you can easily show (expand $y(x-h)$ around x) this will have a similar error to the forward difference estimate. However since they are estimates of the same quantity there must be a way to combine the two to get a more accurate estimate. This is achieved by the **central difference scheme (CDS)** which is the average of the two estimates and is 2nd order accurate,

$$\boxed{\tilde{y}'_c(x) \equiv \frac{\tilde{y}'_f(x) + \tilde{y}'_b(x)}{2} = \frac{y(x+h) - y(x-h)}{2h}\,.} \tag{8.5}$$

Consider the Taylor series (to 3rd order) for $y(x+h)$ and $y(x-h)$ giving

$$y(x+h) = y(x) + y'(x)\,h + \frac{1}{2}y''(x)\,h^2 + \frac{1}{3!}y'''(\xi)\,h^3 \tag{8.6}$$

and

$$y(x-h) = y(x) - y'(x)\,h + \frac{1}{2}y''(x)\,h^2 - \frac{1}{3!}y'''(\zeta)\,h^3 \tag{8.7}$$

then we can use this to define

$$y(x+h) - y(x-h) = 2\,y'(x)\,h + \mathcal{O}(h^3),$$

which gives

$$y'(x) = \frac{y(x+h) - y(x-h)}{2h} - \mathcal{O}(h^2) \equiv \tilde{y}'_c(x) - \mathcal{O}(h^2), \tag{8.8}$$

so the central difference scheme gives the derivative up to an $\mathcal{O}(h^2)$ error. Remember that h is a small step i.e. in suitable units (see later) it will satisfy the condition $h \ll 1$ so the error *decreases* with *increasing* order of magnitude in h.

You can understand why the CDS is more accurate than either FDS or BDS by looking at figure 8.1; the central difference gives a better estimate of the gradient (tangent line) at the point x than the forward or backward difference.

Higher order derivatives

It is possible to find numerical approximations to 2nd and higher order derivatives too. A 2nd order accurate version of d^2y/dx^2 is

$$\boxed{\tilde{y}''(x) \equiv \frac{y(x+h) - 2y(x) + y(x-h)}{h^2} = y''(x) + \mathcal{O}(h^2).} \tag{8.9}$$

This can be obtained by adding together the 4th order Taylor expansions for $y(x+h)$ and $y(x-h)$, i.e., equations (8.6) and (8.7) with one more term. Another way to get (8.9) is to take the central difference versions of 1st order derivative at $x+h/2$ and $x-h/2$, and then find the central difference of these;

$$\tilde{y}''(x) = \frac{\tilde{y}'_c(x + h/2) - \tilde{y}'_c(x - h/2)}{h}\,.$$

Three points, $y(x-h)$, $y(x)$ and $y(x+h)$, are needed to get $\tilde{y}''$, the finite difference approximation of y''. In general, to get the finite difference approximation $\tilde{y}^n$ requires at least $n+1$ points and going further away from x, e.g., $y(x+2h)$, $y(x-2h)$, $y(x+3h)$, $y(x-3h)$, etc.

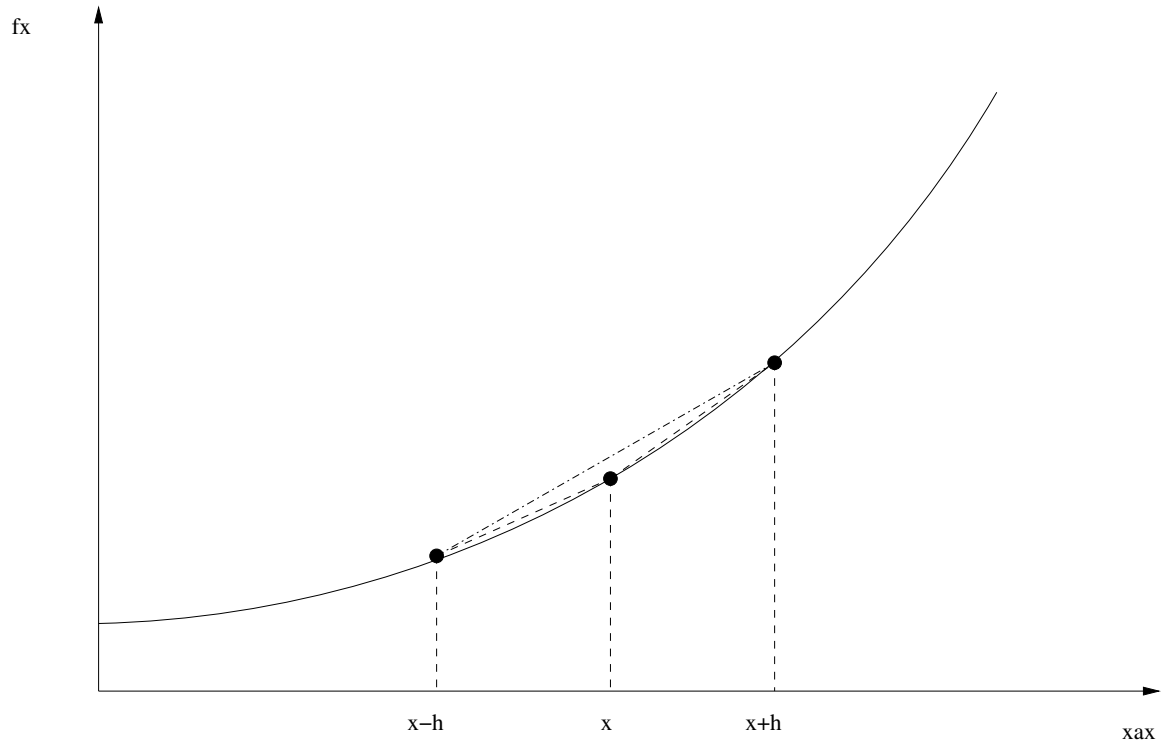

Figure 8.1: Forward, backward, and central difference schemes for approximating the derivative of the function $y(x)$ at the point x

8.2 Ordinary Differential Equations

In countless problems in physics we encounter the description of a physical system through a set of differential equations. This is because it is simpler to model the behaviour of a system (e.g. variables $\vec{y} \equiv \{u, v, w, \ldots, \text{etc.}\}$) in terms of infinitesimal responses to infinitesimal changes of independent variable(s) (e.g. η). The system will be of the form

$$\begin{aligned} \frac{du}{d\eta} &= f_u(\eta, \vec{y}) \\ \frac{dv}{d\eta} &= f_v(\eta, \vec{y}) \\ \frac{dw}{d\eta} &= f_w(\eta, \vec{y}) \\ &\vdots \end{aligned} \tag{8.10}$$

A compact notation for these coupled 1st order ODEs is

$$\frac{d\vec{y}}{d\eta} = \vec{f}(\eta, \vec{y}), \tag{8.11}$$

where $\vec{f}(\eta, \vec{y}) = \{ f_u(\eta, \vec{y}) , f_v(\eta, \vec{y}) , f_w(\eta, \vec{y}) , \ldots \}$, i.e., the 'components' of $\vec{f}$ are the functions in each ODE.

Often the independent variable η is time t, or denoted by the standard symbol for an independent variable: x. Often, we will talk about 'timesteps' and 'time-stepping' in solving systems of differential equations. This simply refers to choosing values of the independent variable (whether that independent variable actually physically corresponds to time or not).

Note that the functions defining the derivatives of the system variables are in general a function of the independent variable *and* all system variables, i.e., the system can be **coupled**. Only if

the derivatives of each system variable are solely a function of that system variable is the system **uncoupled**.

The aim is to find a solution for the system variables at any value of the independent variable given a known initial condition for the system, e.g., solve for $u(\eta)$ given an initial condition $u(\eta_0)$ for all $\eta > \eta_0$.

To do this we need to integrate the differential equations. This can be done analytically for the simplest cases, e.g.,

$$\frac{dy}{dt} = \frac{t}{y} \qquad \text{with} \quad y(t=0) = 1\,,$$

then

$$\int_{y(0)}^{y(t_f)} y\,dy = \int_0^{t_f} t\,dt\,,$$

giving

$$y(t_f) = \sqrt{t_f^2 + 1}\,.$$

Most often however, if the function describing the derivative is not separable or if the system of many variables is coupled, the system cannot be integrated analytically and we have to take a numerical approach.

We can solve systems numerically if the equations have a continuous solution and we know the initial (or boundary value) conditions for the system variables.

Euler method

The simplest approach to numerical integration of an ODE is obtained by looking again at the Taylor expansion for a forward difference. Using time t as the independent variable,

$$\begin{aligned} y(t+h) &= y(t) + y'(t,y)\,h + \mathcal{O}(h^2) \\ &\approx y(t) + f(t,y)\,h\,. \end{aligned} \tag{8.12}$$

So if we know the value of the variable y at t we can use a finite step h to calculate the next value of y at $t+h$ up to $\mathcal{O}(h^2)$ accuracy.

This Euler step can be iterated, stepping the solution forward h each time. In terms of the discretised limit of the function, and remembering that y is now an approximate solution (due to the truncation error incurred in dropping the $\mathcal{O}(h^2)$ terms), we can write this as

$$\tilde{y}_{n+1} = \tilde{y}_n + f(t_n, \tilde{y}_n)\,h, \tag{8.13}$$

where we use the subscript n to denote values of the function y at **time step**

$$t_n = t_0 + (n-1)\,h. \tag{8.14}$$

Notation: For brevity and convenience, the tilde ($\sim$) will normally be dropped when writing out such finite difference methods. (On occasion, when it is necessary to distinguish between actual/exact solutions and approximate solutions, $\sim$ symbols will be reinserted as appropriate.) Hence we chose to write the Euler method as

$$\boxed{y_{n+1} = y_n + f(t_n, y_n)\,h.} \tag{8.15}$$

Higher order systems

A system described by an m^{th}-order ordinary differential equation can always be reduced to a set of m 1st-order ODEs. For example, consider the 3rd-order, linear system

$$\frac{d^3y}{dt^3} + \alpha\frac{d^2y}{dt^2} + \beta\frac{dy}{dt} + \gamma\,y = 0.$$

We can introduce three new variables

$$u \equiv y, \quad v \equiv \frac{dy}{dt}, \quad \text{and} \quad w \equiv \frac{d^2y}{dt^2},$$

such that the system can be written as three 1st-order equations

$$\begin{aligned} \frac{du}{dt} &= v \\ \frac{dv}{dt} &= w \\ \frac{dw}{dt} &= -(\gamma\, u + \beta\, v + \alpha\, w), \end{aligned}$$

or in matrix notation

$$\frac{d}{dt}\begin{pmatrix} u \\ v \\ w \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -\gamma & -\beta & -\alpha \end{pmatrix}\begin{pmatrix} u \\ v \\ w \end{pmatrix}.$$

Generalising further, a pair of coupled m^{th}-order and n^{th}-order ODEs can be reduced to a set of $m+n$ 1st-order coupled ODEs. (This can be extended to sets of coupled arbitrary order ODEs.)

In general a **linear** ODE system can be written in terms of a matrix operator $\mathbf{L}$ as

$$\frac{d\vec{y}}{d\eta} = \mathbf{L}\,\vec{y}, \tag{8.16}$$

with initial conditions $\vec{y}(\eta_0) = \vec{y}_0$. The Euler method is then $\vec{y}_{n+1} = \vec{y}_n + h\,\mathbf{L}\,\vec{y}_n$ so that

$$\boxed{\vec{y}_{n+1} = (\mathbf{I} + h\mathbf{L})\,\vec{y}_n \equiv \mathbf{T}\,\vec{y}_n\ ,} \tag{8.17}$$

where $\mathbf{T}$ is the update matrix.

A general, non-linear system is written as

$$\frac{d\vec{y}}{d\eta} = \vec{f}(\eta, \vec{y}) \qquad \text{or alternatively} \qquad \frac{d\vec{y}}{d\eta} = \mathcal{L}(\vec{y}), \tag{8.18}$$

where $\mathcal{L}$ is a **non-linear operator**. Non-linear systems cannot be expressed as linear matrix equations such as equation (8.16).

The Euler method is an example of a **finite difference method** (FDM). We will cover a number of more advanced methods for integrating ODEs numerically, but before we do that we will look at how to evaluate the properties of any particular method.

8.3 Consistency

This is a check that the finite difference method (e.g. Euler method) reduces to the correct differential equation (e.g. $dy/dt = f(t, y)$) in the limit of vanishingly small step size $h \to 0$. Moreover, a consistency analysis of a finite difference equation reveals the actual differential equation that the finite difference method is effectively solving; the '*modified differential equation*'. The finite difference method actually samples the exact solution of the modified differential equation at the discretised time t_n (or the relevant discretised independent variable x_n). The consistency analysis also tells us the order of the method. This is useful if one is given a finite difference equation, but is not told what its order of accuracy is.

A consistency analysis is carried out by taking the equation for a finite difference method and Taylor expanding all terms about the base point (e.g. x_n). (This is essentially the reverse of the procedure that we used to derive the Euler method in the first place.) As an example, we take the Euler method. Its finite difference equation is

$$y_{n+1} = y_n + f_n\, h, \tag{8.19}$$

where $f_n = f(x_n, y_n)$. Next Taylor expand y_{n+1} about the base point y_n. This yields

$$y_n + y'_n\, h + \frac{y''_n}{2}\, h^2 + \frac{y'''_n}{3!}\, h^3 + \ldots = y_n + f_n\, h,$$

(where $y' = dy/dx$ etc.) which can be rearranged into

$$y'_n = f_n - \frac{y''_n}{2}\, h - \frac{y'''_n}{3!} h^2 - \ldots. \tag{8.20}$$

Remember that the subscript n notation means evaluation of quantities at x_n. Allowing x_n to become continuous, i.e. $x_n \to x$ reveals that this is a differential equation

$$\boxed{y' = f(x,y) - \frac{y''}{2}\, h - \frac{y'''}{3!} h^2 - \ldots}. \tag{8.21}$$

This is the **modified differential equation** (MDE). The discrete points solved by the Euler method (x_n, y_n) lie on the continuous curve that solves the MDE (given the same initial condition of course!). Letting $h \to 0$, equation (8.21) becomes

$$y' = f(x,y), \tag{8.22}$$

which is the intended ODE the Euler method set out to approximate. Thus we say that (8.19) ***is consistent*** with the ODE $y' = f(x,y)$. The ***lowest order in h of the correction terms*** in the MDE, e.g. $-y''\, h/2 \sim \mathcal{O}(h)$ in this case, ***is the order of accuracy of the method***. This is the same as the order of the *global error*, discussed in the following section.

Consistency analysis can be carried out with finite difference methods for partial differential equations too (see Section 12).

8.4 Accuracy

To determine the accuracy of a method we want to take the **local error** (truncation error) – the error incurred each step – and use it to estimate the global error (the error at the end of the calculation).

Using the Euler method as an example we know that the local truncation error is $\mathcal{O}(h^2)$. [See equation (8.12).] y_{n+1}, the **true** value of the function at x_{n+1}, is related to the numerical approximation of the function $\tilde{y}_{n+1}$ by

$$\tilde{y}_{n+1} \equiv y_n + f_n\, h = y_{n+1} + \mathcal{O}(h^2), \tag{8.23}$$

where we have assumed here that y_n is known exactly. Equation (8.23) shows us the error incurred in performing one step of the Euler method. However to integrate from x_0 to x_{end} would take $\mathcal{O}(1/h)$ steps. If we assume the local error at each of the steps simply adds up, in general, we will have

$$\boxed{\textbf{global error} \approx \text{local error} \times \text{number of steps}\,.} \tag{8.24}$$

In this case

$$\epsilon_{\text{end}} \approx \mathcal{O}(h^2) \times \mathcal{O}(1/h) \sim \mathcal{O}(h)\,. \tag{8.25}$$

In general **a method is called n^{th}-order** if its local error is of $\mathcal{O}(h^{n+1})$, i.e., **its global error is of $\mathcal{O}(h^n)$**.

You might think that all we need to do is reduce the step size h (increase the number of steps) arbitrarily to increase the accuracy arbitrarily. Unfortunately the round–off error places a limit on how accurate any method can be. To see this consider a round-off error $\mathcal{O}(\mu)$ added to every step in the integration, then the global error after $\mathcal{O}(1/h)$ steps will be

$$\epsilon_{\text{end}} \sim \frac{\mu}{h} + h\,. \tag{8.26}$$

Thus there is a minimum error (maximum accuracy) achievable for the Euler method when

$$h \sim \mu^{1/2} \,. \tag{8.27}$$

For double precision on most machines $\mu \sim 10^{-16}$ so the smallest useful step size is $h \sim 10^{-8}$ which will give a global accuracy of $\epsilon_{\rm end} \sim 10^{-8}$.

8.5 Stability

This is a crucial property of a method and describes how an error 'propagates' as the finite difference equation is iterated. If the error increases catastrophically with iteration then the method is unstable. If it doesn't grow or decreases, the method is stable. Imagine that the numerical solution at step n has an error ϵ_n

$$\tilde{y}_n = y_n + \epsilon_n, \tag{8.28}$$

where y_n is the true solution of the ODE at $x = x_n$ and $\tilde{y}_n$ the numerical approximation. In taking one step of a finite difference method the numerical approximate solution and the error change and satisfy

$$\tilde{y}_{n+1} = y_{n+1} + \epsilon_{n+1}. \tag{8.29}$$

For stability of *any method* we require that the **amplification factor**

$$\boxed{g \equiv \left| \frac{\epsilon_{n+1}}{\epsilon_n} \right| \leq 1,} \tag{8.30}$$

otherwise the error in the finite difference method exponentially 'blows up'.

We consider the Euler method as an example. We can write it as

$$\tilde{y}_{n+1} = \tilde{y}_n + f(x_n, \tilde{y}_n)\, h. \tag{8.31}$$

Substituting in the true values of the function we have

$$y_{n+1} + \epsilon_{n+1} = y_n + \epsilon_n + f(x_n, y_n + \epsilon_n)\, h. \tag{8.32}$$

Stability analysis can only be carried out for linear ODEs. Luckily for non-linear ODEs, i.e., when f is a non-linear function of y, we can still do a stability analysis by assuming that the errors are small $|\epsilon_n|$, $|\epsilon_{n+1}| \ll |y_n|$ and **linearising $f(x, y)$**. To linearise, the function $f(x_n, y_n + \epsilon_n)$ is expanded around the point (x_n, y_n) in the variable y. To 1st-order in the expansion parameter ϵ_n the equation (8.32) becomes

$$\begin{aligned} y_{n+1} + \epsilon_{n+1} &= y_n + \epsilon_n + \left[f(x_n, y_n) + \left.\frac{\partial f}{\partial y}\right|_n \epsilon_n + \mathcal{O}(\epsilon_n^2) \right] h, \\ &= y_n + f(x_n, y_n) h + \epsilon_n \left[1 + \left.\frac{\partial f}{\partial y}\right|_n h \right] + \mathcal{O}(\epsilon_n^2) \,. \end{aligned} \tag{8.33}$$

where $\partial f/\partial y|_n$ means $\partial f/\partial y$ evaluated at the base point $x = x_n$, $y = y_n$. Then since

$$y_{n+1} = y_n + f(x_n, y_n) h + \mathcal{O}(h^2), \tag{8.34}$$

(i.e. Taylor expansion of the true value) and dropping terms of $\mathcal{O}(h^2)$, $\mathcal{O}(\epsilon_n^2)$ and higher we have

$$\boxed{\epsilon_{n+1} \approx \epsilon_n \left(1 + \left.\frac{\partial f}{\partial y}\right|_n h \right) \,,} \tag{8.35}$$

for the Euler method. Inserting $\epsilon_{n+1}/\epsilon_n$ into equation (8.30), and dropping the $|_n$ notation, we can get a condition on the step size h for stability:

$$g = \left| 1 + \frac{\partial f}{\partial y} h \right| \leq 1 \quad \rightarrow \quad -2 \leq \frac{\partial f}{\partial y} h \leq 0,$$

then assuming $h > 0$ the Euler method is only stable if

$$\boxed{\frac{\partial f}{\partial y} < 0} \quad \text{and} \quad \boxed{h \leq \frac{2}{|\partial f/\partial y|}}. \tag{8.36}$$

We say that the Euler method is **conditionally stable** for $\partial f/\partial y < 0$ and **unconditionally unstable** for $\partial f/\partial y > 0$ (i.e. no step size works).

For a non-linear ODE the amplification factor and step size for stability change with y. For a linear ODE, e.g., $dy/dx = f(x,y) = p(x)y + q(x)$, we have $\partial f/\partial y = p(x)$ and so g and the limiting step size are controlled by $p(x)$. (Note that $\partial f/\partial y$ means keeping x fixed.) For constant coefficients, e.g., the decay problem $y' = -\alpha y$ (with α positive), the stability conditions $g = |1-\alpha\, h|$ and $h \leq 2/\alpha$ do not change over the calculation.

Note that this analysis does not tell us about the global error when the method is stable. Even when $g < 1$ there is still a global error that accumulates with each step. Global error (for a stable method) comes in through the $\mathcal{O}(h^2)$ term dropped in equation (8.34).

8.6 Stability Analysis of General (coupled) Linear Systems

A more general stability analysis can be carried out for m, coupled, linear 1st order ODEs by looking at the general matrix operator form for the method

$$\tilde{\vec{y}}_{n+1} = \mathbf{T}\,\tilde{\vec{y}}_n, \tag{8.37}$$

where $\mathbf{T}$ is the update matrix (e.g. $\mathbf{T} = (\mathbf{I} + \mathbf{L}\,h)$ for the Euler method). For a set of inhomogeneous ODEs $\tilde{\vec{y}}_{n+1} = \mathbf{T}\,\tilde{\vec{y}}_n + h\,\vec{q}(x_n)$ the term $h\,\vec{q}(x_n)$ does not influence numerical stability. Only the homogeneous part of the equation set, i.e., equation (8.37) need be analysed.

We can relate the true solution vector $\vec{y}_n = \{y_n^1, y_n^2, \ldots, y_n^m\}$ (where y^i are each of the dependent variables in the coupled ODE set) to the numerical approximation $\tilde{\vec{y}}_n$ in an analogous way to before;

$$\tilde{\vec{y}}_n = \vec{y}_n + \vec{\epsilon}_n$$

where now $\vec{\epsilon}_n = \{\epsilon_n^1, \epsilon_n^2, ..., \epsilon_n^m\}$. Inserting this into the finite difference equation (8.37) yields

$$\vec{y}_{n+1} + \vec{\epsilon}_{n+1} = \mathbf{T}\vec{y}_n + \mathbf{T}\vec{\epsilon}_n.$$

We can Taylor expand $\vec{y}_{n+1}$ about $\vec{y}_n$

$$\vec{y}_{n+1} = \vec{y}_n + \vec{f}_n\,h + \mathcal{O}(h^2) = (\mathbf{I} + \mathbf{L}\,h)\,\vec{y}_n + \mathcal{O}(h^2) \approx \mathbf{T}\vec{y}_n,$$

where $\vec{f}_n = \{f^1(\vec{y}_n), f^2(\vec{y}_n), \ldots, f^m(\vec{y}_n)\}$, i.e., its components are the functions in each ODE $dy^i/dx = f^i(x, \vec{y})$. This yields a matrix equation for the error propagation;

$$\vec{\epsilon}_{n+1} = \mathbf{T}\,\vec{\epsilon}_n. \tag{8.38}$$

The terms of $\mathcal{O}(h^2)$, discarded in obtaining at (8.38), are the source of global error. However, they are not relevant to the question of stability. It turns out that condition for stability is that the modulus of all the eigenvalues λ^i of the update matrix $\mathbf{T}$ must be less than or equal to unity

$$\boxed{\left|\lambda^i\right| \leq 1 \quad \text{for} \quad i = 1, \ldots, m} \tag{8.39}$$

For real eigenvalues the condition then translates to

$$\lambda^i \leq 1 \quad \text{and} \quad -\lambda^i \leq 1, \tag{8.40}$$

To show condition (8.39) it is useful to diagonalise the matrix equation for error propagation so that we are left with m *decoupled* equations which we can deal with individually. Any non-singular $m \times m$ matrix that has m linearly independent eigenvectors can be diagonalised, i.e., written as

$$\boxed{\mathbf{T} = \mathbf{R}\,\mathbf{D}\,\mathbf{R}^{-1} \quad \text{with} \quad \mathbf{D} \equiv \text{diag}(\lambda^1, \lambda^2, ..., \lambda^m),} \tag{8.41}$$

where λ^i are the eigenvalues of $\mathbf{T}$ and $\mathbf{R}$ is the matrix whose columns are the eigenvectors of $\mathbf{T}$. Substituting (8.41) into (8.38) and operating on both sides with a further $\mathbf{R}^{-1}$ we obtain the diagonal (uncoupled) system

$$\vec{\zeta}_{n+1} = \mathbf{D}\,\vec{\zeta}_n, \tag{8.42}$$

where we have defined the linear transformation $\vec{\zeta}_n = \mathbf{R}^{-1}\,\vec{\epsilon}_n$ which rotates the coupled vectors onto an uncoupled basis. This reduces to m uncoupled equations for the rotated errors ζ_n^i

$$\zeta^i_{n+1} = \lambda^i\,\zeta^i_n\,.$$

Since these are uncoupled we can now impose the constraint on their growth separately for each of them, i.e., for all $i = 1, ..., m$ we have the requirement;

$$g^i = \left|\frac{\zeta^i_{n+1}}{\zeta^i_n}\right| = |\lambda^i_n| \leq 1, \tag{8.43}$$

for a stable method. Since the coupled and uncoupled basis are related by a linear transformation any constraint applied in one basis is equivalent to one applied in the other. That is, the condition (8.43) holds for the original equation (8.38) too.

Non-linear Systems

The above analysis of the eigenvalues of the update matrix can be used for a coupled system of non-linear ODEs, if they are first linearised around the base point $(x_n, \vec{y}_n)$. This is necessary to get the update matrix $\mathbf{T}$ that applies at the current step in the iteration.

Considering the Euler method, we have a set of equations

$$\tilde{y}^i_{n+1} = \tilde{y}^i_n + f^i(x,\, \tilde{y}^1_n,\, \tilde{y}^2_n,\, \ldots\,,\, \tilde{y}^m_n)\,h. \tag{8.44}$$

We substitute $\tilde{y}^i_n = y^i_n + \epsilon^i_n$ into $f^i(x, \tilde{y}^1_n, \ldots)$ and Taylor expand about the true solution to first order in the error, assumed to be small $|\epsilon^i_n| \ll |y^i_n|$;

$$f^i(x, y^1_n + \epsilon^1_n,\, y^2_n + \epsilon^2_n,\, \ldots\,,\, y^m_n + \epsilon^m_n) = f^i(x, \vec{y}_n) + \sum_{k=1}^{m} \left.\frac{\partial f^i}{\partial(y^k)}\right|_n \epsilon^k_n + \mathcal{O}((\epsilon_n)^2). \tag{8.45}$$

This is essentially what was done in section 8.5 with equation (8.33). Following the procedure used before we then get

$$\epsilon^i_{n+1} \approx \epsilon^i_n + h \sum_{k=1}^{m} \left.\frac{\partial f^i}{\partial(y^k)}\right|_n \epsilon^k_n \tag{8.46}$$

so that the elements of the update matrix $\mathbf{T}_n$ are

$$\boxed{T_{jk} = \delta_{jk} + h\,\frac{\partial f^j}{\partial(y^k)},} \tag{8.47}$$

where δ_{jk} is the Kronecker delta function. (Note that the superscripts in y^j, f^i, etc., denote components and not 'raising to a power'.) The eigenvalues of this update matrix can then be analysed to determine the local stability of the method.

8.7 Convergence

If a finite difference method is both consistent with the ODE (system) being solved and stable, then the approximate solution $\tilde{y}_n^i$ converges to the true solution y_n in the limit as $h \to 0$ (in the absence of round-off error).

8.8 Efficiency

Another consideration when selecting a method is the number of computations required for each step. The Euler method requires 1 functional evaluation for each step. As we will see higher order methods will require more evaluations at each step but will in general be more stable and accurate. A compromise must be reached between efficiency, accuracy and stability for any particular system being solved.