Numerical Analysis Module 6
Solving Ordinary Differential Equations - Initial Value Problems (ODE-IVPs)

Sachin C. Patwardhan
Dept. of Chemical Engineering,
Indian Institute of Technology, Bombay
Powai, Mumbai, 400 076, Inda.
Email: sachinp@iitb.ac.in

Contents

1 Introduction

In this module, we develop solution techniques for numerically solving ordinary differential equations (ODE) of the form

$$\frac{d\mathbf{x}}{d\eta} = F(\mathbf{x}, \eta) \quad (1)$$

$$\mathbf{x}(0) = \mathbf{x}_0 \quad (2)$$

where $\mathbf{x} \in R^n$, $F(.,.) : R^n \rightarrow R^n$ represents function vector, $\mathbf{x}(0)$ denotes the initial condition and η denotes independent variable such as time or space. The problem at hand is to develop numerical approximation for solution over $[0, \eta]$, which can be expressed as the following integral equation

$$x(\eta) = x_0^* + \int_0^\eta F[x(\tau), \tau] d\tau$$

There are two basic approaches to solving ODE-IVPs numerically:

- Taylor series expansion, which forms the basis of Runge - Kutta class of methods
- Polynomial interpolation, which forms the basis of multi-step (or predictor - corrector) methods and orthogonal collocations

In this module, we describe these methods in detail. In the remaining part of this module, we use t as the independent variable. While it is convention to use this variable to denote time, the algorithm developed are general and can be applied even when the independent variable represents spatial dimension.

It may appear that the form given by equations (1-2) is somewhat restrictive or a special class of the set of ODEs as the L.H.S. involves only the first order derivatives. In practice, not all models appear as first order ODEs. In general, one can get an m'th order ODE of the type:

$$\frac{d^m y}{dt^m} = f[y, \frac{dy}{dt}, \frac{d^2 y}{dt^2}, ..., \frac{d^{m-1} y}{dt^{m-1}}, t] \quad (3)$$

$$\text{Given } y(0), ...\frac{d^{m-1} y}{dt^{m-1}}(0) \quad (4)$$

Now, do we develop separate methods for each order? It turns out that such a exercise in unnecessary as a m'th order ODE can be converted to m first order ODEs. Thus, we can

define auxiliary variables

$$\begin{aligned} x_1(t) &= y(t) \\ x_2(t) &= \frac{dy}{dt} \\ &..... \\ &..... \\ x_m(t) &= \frac{d^{m-1}y}{dt^{m-1}} \end{aligned} \tag{5}$$

Using these variables, the original mth order ODE can be converted to m first order ODE's as,

$$\begin{aligned} \frac{dx_1}{dt} &= x_2 \\ \frac{dx_2}{dt} &= x_3 \\ &..... \\ \frac{dx_{m-1}}{dt} &= x_m \\ \frac{dx_m}{dt} &= f[x_1, x_2, x_3, ..., x_m, t] \end{aligned} \tag{6}$$

Defining function vector

$$F(\mathbf{x}) = \begin{bmatrix} x_2 \\ ..... \\ x_m \\ f[x_1, x_2, x_3, .., x_m, t] \end{bmatrix} \tag{7}$$

we can write the above set of

$$\frac{d\mathbf{x}}{dt} = F(\mathbf{x}, t) \tag{8}$$

$$\mathbf{x}(0) = \begin{bmatrix} y(0) & \frac{dy}{dt}(0) ..... \frac{d^{m-1y}}{dt^{m-1}}(0) \end{bmatrix}^T \tag{9}$$

Thus, it is sufficient to study only the solution methods for solving n first order ODE's of the form (1-2). Any set of higher order ODEs can be reduced to a set of first order ODEs. Also, forced systems (non-homogeneous systems) can be looked upon as unforced systems (homogenous systems) with time varying parameters. For example, consider a system of ODEs

$$\frac{d\mathbf{x}}{dt} = F(\mathbf{x}, u(t)) \tag{10}$$

$$\mathbf{x}(0) = \mathbf{x}_0 \tag{11}$$

where $u \in R$ represents system input, such as inlet flow to a reactor or inlet temperature. A typical simulation problem is to investigate system dynamics when the independent input

$u(t)$ is specified, say $u(t) = \sin(\omega t)$ for $t > 0$. With the input $u(t)$ specified, the ODE can be represented as follows

$$\frac{d\mathbf{x}}{dt} = F(\mathbf{x}, \sin(wt)) = F_T(\mathbf{x}, t) \quad (12)$$

$$\mathbf{x}(0) = \mathbf{x}_0 \quad (13)$$

where $F_T(\mathbf{x}, t)$ is a function of states and time. Thus, it is sufficient to study the solution methods for homogenous set of equations of the type (1-2).

2 Existence, Uniqueness and Continuity of Solutions [1]

Before we begin developing numerical solutions to the problem at hand, it is necessary to get some insights into the conditions under which a solution exists for the given set of ODE-IVPs. Given a mathematical model described by ODEs, for it to be useful it must have solution. So, the primary concern, is under what conditions solutions exist? Moreover, a mathematical model typically describes behavior of some real physical system. Our experience with experiments with many real systems indicates that if we repeat an experiment with exactly identical initial and other environmental conditions, then we get exactly identical behavior. For example, a pendulum released from same initial angle will oscillate exactly in same manner if other conditions during repeated experiments are maintained identical. This implies that, given an initial condition, the mathematical model should generate exactly one solution. This aspect is referred to as *uniqueness* of the solution. In practice, it is impossible to carry out two experiments in exactly identical manner. We, however, know from experience that if the experiments are carried out under almost similar conditions, then the outcome of the experiments will be *almost similar*. In mathematical parlance, the solution of the ODE-IVP should depend continuously on the initial conditions. Thus, a mathematical model of a physical process should have the following three properties [1]

- **Existence:** A solution satisfying the given initial condition should exist
- **Uniqueness:** Each set of initial condition should yield a unique solution
- **Continuity:** The solution should depend continuously on the initial condition

Given a mathematical model of the form

$$\frac{d\mathbf{x}}{dt} = F(\mathbf{x}, t) \tag{14}$$
$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{15}$$

let $\mathcal{D}$ define a region in (n+1) dimensional space

$$\mathcal{D} = \{(t, \mathbf{x}) : |t - t_0| < T, \|\mathbf{x} - \mathbf{x}_0\| < \beta\}$$

i.e. interior of a *(n+1) dimensional box* or *cylinder.* The following important results provides sufficient conditions for existence of solution of the ODE-IVP.

Example 1 *Let* $\mathbf{x} \in R^2$, $t_0 = 0$,$T \equiv \infty$, $\mathbf{x}_0 \equiv (0, 0)$ *and* $\beta = 1$.*Then, using 2-norm on* R^2, *we have*

$$\mathcal{D} = \left\{(t, \mathbf{x}) : |t| < \infty, .\sqrt{x_1^2 + x_2^2} < 1\right\}$$

which corresponds interior of cylinder of infinite length in 3 dimensional space consisting of (t, x_1, x_2). *Alternatively, using* $1 - norm$, *we have*

$$\mathcal{D} = \{(t, \mathbf{x}) : |t| < \infty, .. |x_1| + |x_2| < 1\}$$

which represents an infinite channel with square cross section 3 dimensional space (t, x_1, x_2).

Theorem 2 *[1] Let vectors* $F(\mathbf{x}, t)$ *and* $\partial F(\mathbf{x}, t)/\partial \mathbf{x}_k$ $(k = 1, 2, ...n)$ *be continuous on region* $\mathcal{D}$.*Then given, any point* $(\widetilde{\mathbf{x}}, t_0) \in \mathcal{D}$, *there exists a unique solution,*$\boldsymbol{\phi}(t)$, *of the system (14-15) satisfying the initial condition* $\boldsymbol{\phi}(t_0) = \widetilde{\mathbf{x}}$. *The solution exists on any interval containing* t_0, *for which the point (t,*$\boldsymbol{\phi}(t)$*)* $\in \mathcal{D}$. *Furthermore, the solution is a continuous function of the triple* $(t, t_0, \widetilde{\mathbf{x}})$.

It may be noted that the theorem does not require us to compute the solution explicitly. If we can assert continuity of the vectors $F(\mathbf{x}, t)$ and $\partial F(\mathbf{x}, t)/\partial \mathbf{x}_k$ in desired region of the state space, then we are assured of the existence of a solution, the uniqueness of the solution and continuity of the solution with respect to $(t, t_0, \widetilde{\mathbf{x}})$. If region $\mathcal{D}$ is the entire $(t, \mathbf{x})$ space, then every solution exists as long as its norm remains finite.

Example 3 *[1]Consider ODE-IVP given by the following set of coupled equations .*

$$\frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = F(\mathbf{x}, t) = \begin{bmatrix} tx_2 + x_3 \\ \cos(t)x_1 + t^2 x_3 \\ x_1 - x_2 \end{bmatrix}$$

$$\frac{\partial F(\mathbf{x},t)}{\partial x_1} = \begin{bmatrix} 0 \\ \cos(t) \\ 1 \end{bmatrix} \quad ; \quad \frac{\partial F(\mathbf{x},t)}{\partial x_2} = \begin{bmatrix} t \\ 0 \\ -1 \end{bmatrix} \quad ; \quad \frac{\partial F(\mathbf{x},t)}{\partial x_3} = \begin{bmatrix} 1 \\ t^2 \\ 0 \end{bmatrix}$$

It is easy to see that vectors $F(\mathbf{x},t)$ *and* $\partial F(\mathbf{x},t)/\partial \mathbf{x}_k \;\; (k=1,2,3)$ *are continuos functions for* $|t| < \infty$ *and* $\|\mathbf{x}\| < \infty$. *Thus,* $\mathcal{D}$ *is entire* $R \times R^3$ *and using Theorem 1, through any point* $(t, \widetilde{\mathbf{x}}) \in R \times R^3$, *there passes a unique solution on some interval containing* t_0.

3 Analytical Solutions of Linear ODE-IVPs

Before developing numerical schemes for solving ODE IVPs, we consider a special sub-class of ODE IVPs, i.e. linear multi-variable ODE-IVPs, which can be solved analytically. The reason for considering this sub-class is two fold:

- A set of nonlinear ODE-IVPs can often be approximated locally as a set of linear ODE-IVPs using Taylor series approximation. Thus, it provides insights into how solutions of a nonlinear ODE-IVP evolve for small perturbations
- Since the solution of a linear ODE-IVP can be constructed analytically, it proves to be quite useful while understanding stability behavior of numerical schemes for solving ODE-IVPs.

Consider the problem of solving simultaneous linear ODE-IVP

$$\frac{d\mathbf{x}}{dt} = \mathbf{A}\mathbf{x}; \tag{16}$$

$$\mathbf{x} = \mathbf{x}(0) \quad at \quad t = 0 \tag{17}$$

$$\mathbf{x} \in R^m, \quad \mathbf{A} \; is \; a \; (m \times m) \; matrix$$

To begin with, we develop solution for the scalar case and generalize it to the multivariable case.

3.1 Scalar Case

Consider the scalar equation

$$\frac{dx}{dt} = ax; \tag{18}$$

$$x\,(t=0) = x(0) \tag{19}$$

Let the guess solution to this IVP be

$$x(t) = e^{\lambda t} v \;\; ; \;\; v \in R \tag{20}$$

Now,

$$\begin{aligned} x &= x(0)\ at\ t=0 \Rightarrow v = x(0) && (21)\\ \text{or}\quad x(t) &= e^{\lambda t}x(0) && (22)\end{aligned}$$

This solution also satisfies the ODE, i.e.

$$\begin{aligned} \frac{dx}{dt} &= \lambda\left[e^{\lambda t}x(0)\right] = \lambda x(t) = ax(t) && (23)\\ &\Rightarrow \lambda = a \ \text{ and } \ x(t) = e^{at}x(0) && (24)\end{aligned}$$

Asymptotic behavior of solution can be predicted using the value of parameter a as follows

- Unstable behavior: $a > 0 \Rightarrow x(t) = e^{at}x(0) \rightarrow \infty\ as\ t \rightarrow \infty$
- Stable behavior:$a < 0 \Rightarrow x(t) = e^{at}x(0) \rightarrow 0\ as\ t \rightarrow \infty$

3.2 Vector case

Now consider system of equations given by equation (16). Taking clues from the scalar case, let us investigate a candidate solution of the form

$$\mathbf{x}(t) = e^{\lambda t}\mathbf{v}; \quad \mathbf{v} \in R^m \tag{25}$$

where $\mathbf{v}$ is a constant vector. The above candidate solution must satisfy the ODE, i.e.,

$$\begin{aligned} &\frac{d}{dt}(e^{\lambda t}\mathbf{v}) = \mathbf{A}(e^{\lambda t}\mathbf{v})\\ &\Rightarrow \lambda\mathbf{v}e^{\lambda t} = \mathbf{A}\mathbf{v}e^{\lambda t}\end{aligned} \tag{26}$$

Cancelling $e^{\lambda t}$ from both the sides, as it is a non-zero scalar, we get an equation that vector $\mathbf{v}$ must satisfy,

$$\lambda\mathbf{v} = \mathbf{A}\mathbf{v} \tag{27}$$

This fundamental equation has two unknowns λ and $\mathbf{v}$ and the resulting problem is the well known eigenvalue problem in linear algebra. The number λ is called the eigenvalue of the matrix $\mathbf{A}$ and $\mathbf{v}$ is called the eigenvector. Now, $\lambda\mathbf{v} = \mathbf{A}\mathbf{v}$ is a non-linear equation as λ multiplies $\mathbf{v}$. if we discover λ then the equation for $\mathbf{v}$ would be linear. This fundamental equation can be rewritten as

$$(\mathbf{A} - \lambda I)\mathbf{v} = 0 \tag{28}$$

This implies that vector $\mathbf{v}$ should be $\perp$ to the row space of $(\mathbf{A} - \lambda I)$. This is possible only when rows of $(\mathbf{A} - \lambda I)$ are linearly dependent. In other words, λ should be selected in such

a way that rows of $(\mathbf{A} - \lambda I)$ become linearly dependent, i.e., $(\mathbf{A} - \lambda I)$ is singular. This implies that λ is an eigenvalue of A if and only if

$$det(\mathbf{A} - \lambda I) = 0 \tag{29}$$

This is the characteristic equation of $\mathbf{A}$ and it has m possible solutions $\lambda_1, ..., \lambda_m$. Thus, corresponding to each eigenvalue λ_i, there is a vector $\mathbf{v}^{(i)}$ that satisfies $(\mathbf{A} - \lambda_i I)\mathbf{v}^{(i)} = 0$. This implies that each vector $e^{\lambda_i t}\mathbf{v}^{(i)}$ is a candidate solution to equation (16). Now, suppose we construct a vector as lineal combination of these fundamental solutions, i.e.

$$\mathbf{x}(t) = c_1 e^{\lambda_1 t}\mathbf{v}^{(1)} + c_2 e^{\lambda_2 t}\mathbf{v}^{(2)} + ... + c_m e^{\lambda_m t}\mathbf{v}^{(m)} \tag{30}$$

Then, it can be shown that $\mathbf{x}(t)$ also satisfies equation (16). Thus, a general solution to the linear ODE-IVP can be constructed as a linear combination of the fundamental solutions $e^{\lambda_i t}\mathbf{v}^{(i)}$.

The next task is to see to it that the above equation reduces to the initial conditions at $t = 0$. Defining vectors C and matrix Ψ as

$$C = \begin{bmatrix} c_1 & c_2 & ... & c_m \end{bmatrix}^T \quad ; \quad \Psi = \begin{bmatrix} \mathbf{v}^{(1)} & \mathbf{v}^{(2)} & ..... & \mathbf{v}^{(m)} \end{bmatrix} \tag{31}$$

we can write

$$\mathbf{x}(0) = \Psi C \tag{32}$$

If the eigenvectors are linearly independent,

$$C = \Psi^{-1}\mathbf{x}(0) \tag{33}$$

Thus the solution can be written as

$$\begin{aligned} &\mathbf{x}(t) = [e^{\lambda_1 t}\mathbf{v}^{(1)} \quad e^{\lambda_2 t}\mathbf{v}^{(2)}.....e^{\lambda_m t}\mathbf{v}^{(m)}]\Psi^{-1}\mathbf{x}(0) \\ &\Rightarrow \mathbf{x}(t) = [\mathbf{v}^{(1)}\ \mathbf{v}^{(2)}.....\mathbf{v}^{(m)}] \begin{bmatrix} e^{\lambda_1 t} & 0 & ... & 0 \\ 0 & e^{\lambda_2 t} & ... & 0 \\ ... & ... & ... & ... \\ 0 & 0 & 0 & e^{\lambda_m t} \end{bmatrix} \Psi^{-1}\mathbf{x}(0) \end{aligned} \tag{34}$$

Now let us define the matrix $e\mathbf{x}p(At)$ as follows

$$e^{At} = I + At + \frac{1}{2!}(At)^2 + ... \tag{35}$$

Using the fact that matrix A can be diagonalized as

$$\mathbf{A} = \Psi\Lambda\Psi^{-1} \tag{36}$$

where matrix Λ is

$$\Lambda = diag \begin{bmatrix} \lambda_1 & \lambda_2 & .... & \lambda_m \end{bmatrix}$$

we can write

$$\begin{aligned} e^{At} &= \Psi\Psi^{-1} + \Psi\Lambda\Psi^{-1}t + \tfrac{1}{2!}\Psi\Lambda^2\Psi^{-1}t^2 + ... \\ &= \Psi\Psi^{-1} + \Psi\Lambda\Psi^{-1}t + \tfrac{1}{2!}\Psi\Lambda^2\Psi^{-1}t^2 + ... \\ &= \Psi e^{\Lambda t}\Psi^{-1} \end{aligned} \tag{37}$$

Here, the matrix $e^{\Lambda t}$ is limit of infinite sum

$$\begin{aligned} e^{\Lambda t} &= I + t\Lambda + \tfrac{1}{2!}t^2\Lambda^2 + ... \\ &= \begin{bmatrix} e^{\lambda_1 t} & 0 & ... & 0 \\ 0 & e^{\lambda_2 t} & ... & 0 \\ ... & ... & ... & ... \\ 0 & 0 & 0 & e^{\lambda_m t} \end{bmatrix} \end{aligned} \tag{38}$$

Thus, equation (34) reduces to

$$\mathbf{x}(t) = \Psi e^{\Lambda t}\Psi^{-1}\mathbf{x}(0) \tag{39}$$

With this definition, the solution to the ODE-IVP can be written as

$$\mathbf{x}(t) = \Psi e^{\Lambda t}\Psi^{-1}\mathbf{x}(0) = e^{At}\mathbf{x}(0) \tag{40}$$

3.3 Asymptotic behavior of solutions

In the case of linear multivariable ODE-IVP problems, it is possible to analyze asymptotic behavior of the solution by observing eigenvalues of matrix **A**.

$$\begin{aligned} \mathbf{x}(t) = c_1 e^{\lambda_1 t}\mathbf{v}^{(1)} + c_2 e^{\lambda_2 t}\mathbf{v}^{(2)} + ... + c_m e^{\lambda_m t}\mathbf{v}^{(m)} \\ C = \Psi^{-1}\mathbf{x}(0) \end{aligned} \tag{41}$$

Let $\lambda_j = \alpha_j + i\beta_j$ represent j'th eigenvalue of matrix **A**. Then, we can write

$$e^{\lambda_j t} = e^{\alpha_j t}.e^{i\beta_j t} = e^{\alpha_j t}[\cos\beta_j t + i\sin\beta_j t] \tag{42}$$

As

$$\left|[\cos\beta_j t + i\sin\beta_j t]\right| \leq 1 \text{ for all } t \text{ and all } j \tag{43}$$

the asymptotic behavior of the solution $\mathbf{x}(t)$ as $t \to \infty$ is governed by the terms $e^{\alpha_j t}$. We have following possibilities here

- If $\alpha_j < 0$ then $e^{\alpha_j t} \to 0$ as $t \to \infty$
- If $\alpha_j > 0$ then $e^{\alpha_j t} \to \infty$ as $t \to \infty$

- If $\alpha_j = 0$ then $e^{\alpha_j t} \to 1$ as $t \to \infty$

Thus, we can deduce following three cases

- Case A: $||\mathbf{x}(t)|| \to 0$ as $t \to \infty$ if and only if $Re(\lambda_i) < 0$ for $i = 1, 2, ...m$ (Asymptotically stable solution)
- Case B: $||\mathbf{x}(t)|| \leq M < \infty$ as $t \to \infty$ if and only if $Re(\lambda_i) \leq 0$ for $i = 1, 2, ...m$ (Stable solution)
- Case C: $||\mathbf{x}(t)|| \to \infty$ *at* $t \to \infty$ if for any λ_i,Re(λ_i) > 0 for i = 1,2,.....n (Unstable solution)

Note that asymptotic dynamics of linear ODE-IVP is governed by only eigenvalues of matrix $\mathbf{A}$ and is independent of the initial state $\mathbf{x}(t)$. Thus, based on the sign of real part of eignvalues of matrix $\mathbf{A}$, the ODE-IVP is classified as asymptotically stable, stable or unstable.

3.4 Local Analysis of Nonlinear Systems

The above approach can be extended to obtain local or perturbation solutions of nonlinear ODE-IVP systems

$$\frac{d\mathbf{x}}{dt} = F\left[\mathbf{x}(t), u(t)\right] \;\; ; \;\; \mathbf{x} = \;\mathbf{x}(0) \;\text{ at } t = 0 \tag{44}$$

in the neighborhood of a steady state point $\overline{\mathbf{x}}$ such that

$$F(\overline{\mathbf{x}}) = 0 \tag{45}$$

Using Taylor expansion in the neighborhood of $\overline{\mathbf{x}}$ and neglecting terms higher that first order, equation (44) can be approximated as

$$\begin{aligned} \frac{d(\mathbf{x} - \overline{\mathbf{x}})}{dt} &= \left[\frac{\partial F}{\partial \mathbf{x}}\right]_{\mathbf{x}=\overline{\mathbf{x}}} (\mathbf{x} - \overline{\mathbf{x}}) \\ \frac{d\delta\mathbf{x}}{dt} &= \mathbf{A}\;\delta\mathbf{x} \;\; ; \;\; \delta\mathbf{x}(0) = \mathbf{x}(0) - \overline{\mathbf{x}} \\ \mathbf{A} &= \left[\frac{\partial F}{\partial \mathbf{x}}\right]_{\mathbf{x}=\overline{\mathbf{x}}} \end{aligned} \tag{46}$$

Note that the resulting equation is a linear multivariable system of type (16) and the perturbation solution can be computed as

$$\begin{aligned} \delta\mathbf{x}(t) &= e^{At}\delta\mathbf{x}(0) \\ \mathbf{x}(t) &= \overline{\mathbf{x}} + \delta\mathbf{x}(t) \end{aligned}$$

Example 4 ***Stirred Tank Reactor***

The system under consideration is a Continuous Stirred Tank Reactor (CSTR) in which a non-isothermal, irreversible first order reaction

$$A \longrightarrow B$$

is taking place. The dynamic model for a non-isothermal CSTR is given as follows :

$$\frac{dC_A}{dt} = \frac{F}{V}(C_{A0} - C_A) - k_0 \exp(-\frac{E}{RT})C_A \quad \text{(B.1)}$$

$$\frac{dT}{dt} = \frac{F}{V}(T_0 - T) + \frac{(-\Delta H_r)\, k_0}{\rho C_p} \exp(-\frac{E}{RT})C_A - \frac{Q}{V\rho C_p} \quad \text{(B.2)}$$

$$Q = \frac{aF_c^{b+1}}{F_c + \left(\frac{aF_c^b}{2\rho_c C_{pc}}\right)}(T - T_{cin}) \quad \text{(B.3)}$$

This system exhibits entirely different dynamic characteristics for different set of parameter values (Marlin, 1995). The nominal parameter values and nominal steady state operating conditions of the CSTR for the *stable and unstable operating* points are given in Table

Table 1: Model Parameters and Nominal Operating Conditions of CSTR

Parameter (↓) Operating Point (→)		Stable	Unstable
Reaction rate constant (k_0)	min^{-1}	10^{10}	10^{10}
Inlet concentration of A (C_{A0})	$kmol/m^3$	2.0	2.0
Steady state flow rate of A (F)	m^3/min	1.0	1.0
Density of the reagent A (ρ)	g/m^3	10^6	10^6
Specific heat capacity of A(C_p)	cal/g^0C	1.0	1.0
Heat of reaction (ΔH$_r$)	$cal/kmol$	$-130 * 10^6$	$-130 * 10^6$
Density of the coolant (ρ_c)	g/m^3	10^6	10^6
Specific heat capacity of coolant (C_{pc})	cal/g^0C	1.0	1.0
Volume of the CSTR (V)	m^3	1.0	1.0
Coolant flow rate (F_c)	m^3/min	15	15
Inlet temperature of the coolant (T_{cin})	^0K	365	365
Inlet temperature of A (T_0)	^0K	323	343
Reactor temperature (T)	K	393.954	349.88
Reactor concentration of A (C_A)	$kmol/m^3$	0.265	1.372
a		$1.678X10^6$	$0.516X10^6$
Reaction Rate Parameter (E/R)	$(^0K)^{-1}$	8330	8330
b		0.5	0.5

1.

- Perturbation model at stable operating point

$$\frac{d}{dt}\begin{bmatrix} \delta C_A \\ \delta T \end{bmatrix} = \begin{bmatrix} -7.559 & -0.09315 \\ 852.7 & 5.767 \end{bmatrix}\begin{bmatrix} \delta C_A \\ \delta T \end{bmatrix}$$

Eigenvalues of $\left[\frac{\partial F}{\partial \mathbf{x}}\right]_{\mathbf{x}=\overline{\mathbf{x}}}$ are

$$\lambda_1 = -0.8960 + 5.9184i \;\; ; \;\; \lambda_2 = -0.8960 - 5.9184i$$

and all the trajectories for the unforced system (i.e. when all the inputs are held constant at their nominal values) starting in the neighborhood of the steady state operating point converge to the steady state.

- Perturbation model at unstable operating point

$$\frac{d}{dt}\begin{bmatrix} \delta C_A \\ \delta T \end{bmatrix} = \begin{bmatrix} -1.889 & -0.06053 \\ 115.6 & 2.583 \end{bmatrix}\begin{bmatrix} \delta C_A \\ \delta T \end{bmatrix}$$

Eigenvalues of $\left[\frac{\partial F}{\partial \mathbf{x}}\right]_{\mathbf{x}=\overline{\mathbf{x}}}$ are

$$\lambda_1 = 0.3468 + 1.4131i \, ; \;\; \lambda_2 = 0.3468 - 1.4131i$$

and all the trajectories for the unforced system starting in any small neighborhood of the steady state operating point diverge from the steady state.

4 Numerical Solution Schemes: Basic Concepts

4.1 Marching in Time

Let $\{\mathbf{x}^*(t) : 0 \leq t \leq t_f\}$ denote the true / actual solution of the above ODE-IVP. In general, for a nonlinear ODE, it is seldom possible to obtain the true solution analytically. The aim of numerical methods is to find an approximate solution numerically. Let $t_1, t_2, ..., t_n$ be a sequence of numbers such that

$$0 < t_1 < t_2 < ..... < t_n < .... < t_f$$

Instead of attempting to approximate the function $\mathbf{x}^*(t)$, which is defined for all values of t such that $0 \leq t \leq t_f$, we attempt to approximate the sequence of vectors $\{\mathbf{x}^*(t_n) : n =$

$1, ...f\}$. Thus, in order to integrate over a large interval $0 \leq t \leq t_f$,we solve a sequence of ODE-IVPs subproblems

$$\begin{array}{rcl} \dfrac{d\mathbf{x}}{dt} & = & F(\mathbf{x},t) \ ; \quad \mathbf{x}(t_n) = \mathbf{x}(n) \ ; \\ t_n & \leq & t \leq t_{n+1} \ ; \ (n = 1, 2, ...f) \end{array}$$

each defined over a smaller interval $[t_n, t_{n+1}]$. This generates a sequence of approximate solution vectors $\{\mathbf{x}(t_n) : n = 1, ...f\}$. The difference $h_n = t_n - t_{n-1}$ is referred to as the integration step size or the integration interval. Two possibilities can be considered regarding the choice of the sequence $\{t_n\}$

- Fixed integration interval: The numbers t_n are equispaced, i.e., $t_n = nh$ for some $h > 0$
- Variable size integration intervals

For the sake of convenience, we introduce the following notation

$$F(n) \equiv F[\mathbf{x}(t_n), t_n] \tag{47}$$

$$\mathbf{x}(n) \equiv \mathbf{x}(t_n) \tag{48}$$

$$\left(\frac{\partial F}{\partial \mathbf{x}}\right)_n \equiv \left(\frac{\partial F}{\partial \mathbf{x}}\right)_{(\mathbf{x}(t_n), t_n)} \tag{49}$$

and use it throughout in the rest of the text.

4.2 Two Solution Approaches : Implicit and Explicit

There are two basic approaches to numerical integrations. To understand these approaches, consider the integration of the equation (1) over the interval $[t_n, t_{n+1}]$ using Euler's method. Let us also assume that the numbers t_n are equi-spaced and h is the integration stepsize.

- **Explicit Euler method:**If the integration interval is *small*,

$$\begin{array}{l} \dfrac{d\mathbf{x}}{dt} \cong \dfrac{\mathbf{x}(n+1) - \mathbf{x}(n)}{h} = F[\mathbf{x}(n), t_n] \\ \mathbf{x}(n+1) = \mathbf{x}(n) + hF(n) \\ for \quad n = 0, 1, ..., n-1 \end{array} \tag{50}$$

The new value $\mathbf{x}(n+1)$ is a function of only the past value of $\mathbf{x}$ i.e., $\mathbf{x}(n)$. This type of numerical scheme is called explicit as it does not involve iterative calculations while moving forward in time.

Table 1: Implicit Euler Method: Iterative Computations

Initialize: $\mathbf{x}(0), t_f, h, \varepsilon, N = t_f/h$
FOR n = 0 TO N-1
  $\mathbf{x}^{(0)}(n+1) = \mathbf{x}(n) + hF[\mathbf{x}(n), t_n]$
  WHILE ($\delta > \varepsilon$)
    $\mathbf{x}^{(k+1)}(n+1) = \mathbf{x}(n) + hF[\mathbf{x}^{(k)}(n+1), t_{n+1}]$
    $\delta = \dfrac{||\mathbf{x}^{(k+1)}(n+1) - \mathbf{x}^{(k)}(n+1)||}{||\mathbf{x}^{(k)}(n+1)||}$
  END WHILE
  $\mathbf{x}(n+1) = \mathbf{x}^{(k)}(n+1)$
END FOR

- **Implicit Euler method:**

$$\begin{aligned} &\frac{d\mathbf{x}}{dt} \cong \frac{\mathbf{x}(n+1) - \mathbf{x}(n)}{h} = F[\mathbf{x}(n+1), t_{n+1}] \\ &\mathbf{x}(n+1) = \mathbf{x}(n) + hF(n+1), \\ &for \quad n = 0, 1, ..., n-1 \end{aligned} \tag{51}$$

Each of the above equation has to be solved by iterative method. For example if we use successive substitution method for solving the resulting nonlinear equation(s), the algorithm is summarized in Table 1. This type of numerical scheme is called implicit as it involves iterative calculations while moving forward in time.

5 Numerical Methods Based On Taylor Series Expansion[2]

Consider a simple scalar case

$$\frac{dx}{dt} = f(x,t) \; ; \quad x \in R \tag{52}$$

Suppose we know the exact solution $x^*(t_n) = x^*(n)$ and the integration step size h is selected sufficiently small, then we can compute $x^*(n+1)$ using Taylor series expansion with respect to independent variable t as follows:

$$x^*(n+1) = x^*(t_n + h) \tag{53}$$

$$= x^*(n) + h\frac{dx^*(t_n)}{dt} + \frac{1}{2!}h^2\frac{d^2x^*(t_n)}{dt^2} + ... \tag{54}$$

The various derivatives in the above series can be calculated using the differential equation, as follows:

$$\frac{dx^*(t_n)}{dt} = f\left[t_n, x^*(n)\right] \tag{55}$$

$$\frac{d^2x^*(t_n)}{dt^2} = \frac{df\left[t_n, x^*(n)\right]}{dt} \tag{56}$$

and so on. Now, the exact differential of $f(x,t)$ i.e.

$$df = \left[\frac{\partial f}{\partial x}\right] dx + \frac{\partial f}{\partial t} dt$$

we can write

$$\frac{d^2x^*(t_n)}{dt^2} = \left[\frac{\partial f}{\partial x}\right]_{(x^*(n),t_n)} \frac{dx^*(t_n)}{dt} + \frac{\partial f\left[x^*(n),t_n\right]}{\partial t} \tag{57}$$

$$= \left[\frac{\partial f}{\partial x}\right]_{(x^*(n),t_n)} f\left[x^*(n),t_n\right] + \frac{\partial f\left[x^*(n),t_n\right]}{\partial t} \tag{58}$$

Let us now suppose that, instead of actual solution $x^*(n)$, we have available an approximation to $x^*(n)$, denoted as $x(n)$. With this information, we can construct $x(n+1)$, as follows

$$x(n+1) = x(n) + hf(n) + \frac{h^2}{2}\left[\left(\frac{\partial f}{\partial x}\right)_n f(n) + \left(\frac{\partial f}{\partial t}\right)_n\right] + ... \tag{59}$$

We can now make a further approximation by truncating the infinite series. If the Taylor series is truncated after the term involving h^k, then the Taylor's series method is said to be of order k.

- **Order 1(Euler explicit formula)**

$$x(n+1) = x(n) + hf(n) \tag{60}$$

- **Order 2**

$$x(n+1) = x(n) + hf(n) + \frac{h^2}{2}\left[\left(\frac{\partial f}{\partial x}\right)_n f(n) + \left(\frac{\partial f}{\partial t}\right)_n\right] \tag{61}$$

Taylor's series methods are useful starting points for understanding more sophisticated methods, but are not of much computational use. First order method is too inaccurate and the higher order methods require calculation of a lot of partial derivatives.

5.1 Univariate Runge-Kutta (R-K) Methods [2]

Runge-Kutta methods duplicate the accuracy of the Taylor series methods, but do not require the calculation of higher partial derivatives. For example, consider the second order method that uses the formula

$$x(n+1) = x(n) + h(ak_1 + bk_2) \tag{62}$$

$$\begin{aligned} k_1 &= f(t_n, x(n)) = f(n) && (63)\\ k_2 &= f(t_n + \alpha h, x(n) + \beta h k_1) && (64) \end{aligned}$$

The real numbers a, b, α, β, are chosen such that the RHS of (62) approximates the RHS of Taylor series method of order 2 (ref. 61). To see how this is achieved, let k_2 be represented as

$$k_2 = f(t_n + \Delta t, x(n) + \Delta x(n)) \tag{65}$$

where $\Delta t = \alpha h$ and $\Delta x(n) = \beta h k_1$, and consider the Taylor series expansion of the function k_2, about $(t_n, x(n))$

$$\begin{aligned} k_2 &= f(t_n, x(n)) + \left(\frac{\partial f}{\partial t}\right)_n (\Delta t) + \left(\frac{\partial f}{\partial x}\right)_n \Delta x(n) + O(h^2) && (66)\\ &= f(n) + \left(\frac{\partial f}{\partial t}\right)_n (\alpha h) + \left(\frac{\partial f}{\partial x}\right)_n [\beta h f(n)] + O(h^2) && (67) \end{aligned}$$

where subscript n denotes that the corresponding derivatives have been computed at $(t_n, x(n))$. Substituting the Taylor series expansion in equation (62), we have

$$\begin{aligned} x(n+1) &= x(n) + ahf(n) \\ &\quad + bh\left[f(n) + \left(\frac{\partial f}{\partial t}\right)_n (\alpha h) + \left(\frac{\partial f}{\partial x}\right)_n [\beta h f(n)]\right] + O(h^3) && (68) \end{aligned}$$

$$x(n+1) = x(n) + (a+b)hf(n) + \left(\frac{\partial f}{\partial t}\right)_n \left[\alpha b h^2\right] + \left(\frac{\partial f}{\partial x}\right)_n \left[\beta b h^2 f(n)\right] + O(h^3) \tag{69}$$

Comparing 61 and 69, we arrive at the following set of constraints on the parameters

$$\begin{aligned} a + b &= 1 \\ \alpha b = \beta b &= \frac{1}{2} \end{aligned} \tag{70}$$

Thus, there are 4 unknowns and 3 equations and we can choose one variable arbitrarily. Let us select variable b as the one that can be set arbitrarily. With this choice, we have

$$a = 1 - b; \quad \alpha = \frac{1}{2b}; \quad \beta = \frac{1}{2b} \tag{71}$$

together with the condition $b \neq 0$. Thus, the general 2nd order algorithm can be stated as

$$x(n+1) = x(n) + h\left[(1-b)f(n) + bf\left(t_n + \frac{h}{2b}, x(n) + \frac{h}{2b}f(n)\right)\right] \tag{72}$$

- **Heun's modified algorithm:**Set b = 1/2.

$$x(n+1) = x(n) + h\left[(\frac{1}{2}f(n) + \frac{1}{2}f\left(t_n + h, x(n) + hf(n)\right)\right] \tag{73}$$

- **Modified Euler-Cauchy Method:** Set b = 1.

$$x(n+1) = x(n) + hf\left[t_n + \frac{h}{2}, x(n) + \frac{h}{2}f(n)\right] \tag{74}$$

It must be emphasized that 72. and 61 do not give identical results. However, if we start from the same $x(n)$, then $x(n+1)$ given by 61 and 72 would differ only by $O(h^3)$.

The third and higher order methods can be derived in an analogous manner. The general computational form of the third order method can be expressed as follows

$$x(n+1) = x(n) + h(ak_1 + bk_2 + ck_3) \tag{75}$$

$$\begin{aligned} k_1 &= f(t_n, x(n)) = f(n) && (76)\\ k_2 &= f(t_n + \alpha h, x(n) + \beta h k_1) && (77)\\ k_3 &= f(t_n + \gamma h, x(n) + \delta h k_2) && (78) \end{aligned}$$

The parameters $(a, b, c, \alpha, \beta, \gamma, \delta)$ are chosen such that the RHS of (75) approximates the RHS of Taylor series method of order 3.

5.2 Multivariate R-K Methods

Even though the above derivation has been worked for one dependent variable case, the method can be easily extended to multi-variable case. For example, the most commonly used fourth order R-K method for one variable can be stated as

$$x(n+1) = x(n) + \frac{h}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right) \tag{79}$$

$$\begin{aligned} k_1 &= f(t_n, x(n)) = f(n) && (80)\\ k_2 &= f\left(t_n + \frac{h}{2}, x(n) + \frac{h}{2}k_1\right) && (81)\\ k_3 &= f\left(t_n + \frac{h}{2}, x(n) + \frac{h}{2}k_2\right) && (82)\\ k_4 &= f\left(t_n + h, x(n) + hk_3\right) && (83) \end{aligned}$$

Now, suppose we want to use this method for solving m simultaneous ODE-IVPs

$$\frac{d\mathbf{x}}{dt} = \mathbf{F}(\mathbf{x}, t) \tag{84}$$
$$\mathbf{x}(0) = \mathbf{x}_0 \tag{85}$$

where $\mathbf{x} \in R^m$ and $F(\mathbf{x}, t)$ is a $m \times 1$ function vector. Then, the above algorithm can be modified as follows

$$\mathbf{x}(n+1) = \mathbf{x}(n) + \frac{h}{6}\left(\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4\right) \tag{86}$$

$$\mathbf{k}_1 = \mathbf{F}\left(t_n, \mathbf{x}(n)\right) = \mathbf{F}(n) \tag{87}$$
$$\mathbf{k}_2 = \mathbf{F}\left(t_n + \frac{h}{2}, \mathbf{x}(n) + \frac{h}{2}\mathbf{k}_1\right) \tag{88}$$
$$\mathbf{k}_3 = \mathbf{F}\left(t_n + \frac{h}{2}, \mathbf{x}(n) + \frac{h}{2}\mathbf{k}_2\right) \tag{89}$$
$$\mathbf{k}_4 = \mathbf{F}\left(t_n + h, \mathbf{x}(n) + h\mathbf{k}_3\right) \tag{90}$$

Note that $\mathbf{k}_1, \mathbf{k}_2, \mathbf{k}_3$ and $\mathbf{k}_4$ are $n \times 1$ function vectors.

Note that Runge-Kutta methods can be implemented using variable step size with accuracy monitoring. Thus, these methods (with variable step size implementation) are preferred when $\mathbf{x}(t)$ is expected to change rapidly in some regions and slowly in others.

6 Numerical Methods Based on Polynomial Interpolation [2]

6.1 Multi-step Methods

The multi-step methods are based on polynomial interpolation. We approximate the solution of a given differential equation by a polynomial in the independent variable t. To understand how this is achieved, consider the scalar differential equation

$$\frac{dx}{dt} = f(x, t) \; ; \quad x(t_n) = x(n); \quad x \in R \tag{91}$$

with uniformly spaced integration (time) intervals. At time $t = t_n$, we have the state and the derivative information in the 'past' i.e.

$$\{x(n), x(n-1), x(n-2).....x(0)\}$$

and

$$\{f(n), f(n-1), f(n-2).....f(0)\}$$

which can be used to construct the interpolation polynomial. We approximate $x(t)$ in the neighborhood of $t = t_n$ by constructing a local polynomial approximation of type

$$x_n(t) = a_0(n) + [a_1(n)]\, t + [a_2(n)]\, t^2 + ... + [a_m(n)]\, t^m \tag{92}$$

and use it to estimate or extrapolate $x(n+1)$. The coefficients of the polynomial are computed using the state and the derivative information from the past and possibly $f(n+1)$. It may be noted that time index (n) is included in the representation of the polynomial coefficients to highlight the fact that these coefficients are time varying. In other words, at each integration step, we compaute a separate local interpolation polynomial and use it for local extrapolation or estimation.

To begin with, let us consider a simple case where we want to construct a second order interpolation polynomial of the form

$$x_n(t) = a_0(n) + [a_1(n)]\ \ t + [a_2(n)]\ \ t^2 \tag{93}$$

at instant $t = t_n$. This implies the derivative $f(x,t)$ at time t can be computed as

$$f(x,t) = \frac{dx_n(t)}{dt} = a_1(n) + 2a_2(n)\, t \tag{94}$$

For the sake of computational convenience, we choose a shifted time scale as, $\tau = t - t_n$, it follows that

$$t \;=\; t_n \equiv \tau = 0 \;\;;\;\; t = t_{n+1} \equiv \tau = h \tag{95}$$

$$t \;=\; t_{n-1} \equiv \tau = -h \;\;; t = t_{n-2} \equiv \tau = -2h \tag{96}$$

Now, there are several ways we could go about estimating the unknown parameters of the polynomial.

- **Explicit algorithm:** Let us use only the current and the past information of state and derivatives, which will lead to an explicit algorithm.

$$\begin{aligned} f(n-1) &= a_1(n) - 2ha_2(n) \\ f(n) &= a_1(n) \end{aligned} \tag{97}$$

$$x(n) = a_0(n) \tag{98}$$

Solving above equations simultaneously, we get coefficients

$$a_0(n) = x(n) \;\;;\; a_1(n) = f(n) \;\;;\;\; a_2(n) = \frac{f(n) - f(n-1)}{2h}$$

and the interpolation polynomial with time varying coefficients can be expressed as follows

$$x(\tau) = x(n) + [f(n)]\,\tau + \left[\frac{f(n) - f(n-1)}{2h}\right]\tau^2$$

This interpolation polynomial can be used to extrapolate the value of $x(t)$ at $t = t_{n+1}$ i.e. $\tau = h$ as

$$\begin{aligned} x(n+1) &= x(h) \\ &= x(n) + f(n)\,h + \left[\frac{f(n) - f(n-1)}{2h}\right]h^2 \\ &= x(n) + h\left[\frac{3}{2}f(n) - \frac{1}{2}f(n-1)\right] \end{aligned}$$

- **Implicit algorithm:** Alternatively, we can choose to estimate $x(n+1)$ based on derivative at t_{n+1},i.e.

$$f(n+1) = a_1(n) + 2a_2(n)\,h \tag{99}$$

$$f(n) = a_1(n) \tag{100}$$

$$x(n) = a_{0,n} \tag{101}$$

These equations yield following set of coefficients

$$a_0(n) = x(n) \;\;;\;\; a_1(n) = f(n) \;\;;\;\; a_2(n) = \frac{f(n+1) - f(n)}{2h}$$

and the interpolation polynomial solution can be expressed as follows

$$x(\tau) = x(n) + [f(n)]\,\tau + \left[\frac{f(n+1) - f(n)}{2h}\right]\tau^2$$

and $x(n+1)$ can be estimated as

$$\begin{aligned} x(n+1) &= x(h) \\ &= x(n) + [f(n)]\;h + \left[\frac{f(n+1) - f(n)}{2h}\right]h^2 \end{aligned}$$

The above expression can be rearranged as

$$x(n+1) = x(n) + \frac{h}{2}\,[f(n) + f(n+1)]$$

which is popularly known as *trapezoidal rule* or Crank-Nicholson algorithm.

Let us consider using a higher order interpolation polynomial, say a 4'th order polynomial of the form

$$x_n(\tau) = a_0(n) + [a_1(n)]\ \tau + [a_2(n)]\ \tau^2 + [a_3(n)]\ \tau^3 + [a_4(n)]\ \tau^4 \tag{102}$$

There five unknown coefficients, $a_0(n), ...a_4(n)$ of the interpolation polynomial and we need to generate five equations for estimating these coefficients. There are several ways by which we can construct these equations. For example, we can use any one of the following sets to arrive at these equations

$$\{x(n),\ x(n-1),\ x(n-2),\ x(n-3),\ f(n)\}$$
$$\{f(n),\ f(n-1),\ f(n-2),\ f(n-3),\ x(n)\}$$
$$\{x(n), x(n-1), f(n), f(n+1), f(n-1)\}$$

and there are more such possibilities. In particular, let us consider the last set and proceed with derivation of the polynomial coefficients.

$$\tau = 0 : x(n) = a_0(n) \quad \text{and} \quad f(n) = a_1(n) \tag{103}$$

$$\tau = h : f(n+1) = a_1(n) + 2ha_2(n) + 3h^2a_3(n) + 4h^3\ [a_4(n)] \tag{104}$$
$$\tau = -h : f(n-1) = a_1(n) - 2ha_2(n) + 3h^2a_3(n) - 4h^3\ [a_4(n)] \tag{105}$$

Combining equations (103), (104) and (105) yields

$$a_3(n) = \frac{f(n+1) - 2f(n) + f(n-1)}{6h^2} \tag{106}$$

Also, at $\tau = -h$, we have

$$x(n-1) = a_0(n) - [a_1(n)]\ h + [a_2(n)]\ h^2 - [a_3(n)]\ h^3 + [a_4(n)]\ h^4$$

which, are rearranging yields

$$\begin{aligned} [a_2(n)]\ h + [a_4(n)]\ h^3 &= f(n) + [a_3(n)]\ h^2 + \frac{x(n-1) - x(n)}{h} \\ &= \frac{f(n+1)}{6} + \frac{2f(n)}{3} + \frac{f(n-1)}{6} + \frac{x(n-1) - x(n)}{h} \end{aligned} \tag{107}$$

Rearrangement of equation (104) yields

$$\begin{aligned} 2h\,[a_2(n)] + 4h^3\ [a_4(n)] &= f(n+1) - f(n) - 3h^2\,[a_3(n)] \\ &= \frac{f(n+1)}{2} - \frac{f(n-1)}{2} \end{aligned} \tag{108}$$

Using (107) and (108), the remaining two coefficients of the local interpolation polynomial can be estimated as follows

$$a_2(n) = \frac{1}{2h}\left[\frac{f(n+1)}{6} + \frac{8f(n)}{3} + \frac{7f(n-1)}{6} + \frac{4x(n-1)-4x(n)}{h}\right] \quad (109)$$

$$a_4(n) = \frac{1}{2h^3}\left[\frac{f(n+1)}{6} - \frac{5f(n-1)}{2} - \frac{4f(n)}{3} + \frac{2x(n)-2x(n-1)}{h}\right] \quad (110)$$

Thus, the 4'th order interpolation polynomial assumes form

$$\begin{aligned} x_n(\tau) &= x(n) + [f(n)]\ \tau + \frac{1}{2h}\left[\frac{f(n+1)}{6} + \frac{8f(n)}{3} + \frac{7f(n-1)}{6} + \frac{4x(n-1)-4x(n)}{h}\right]\ \tau^2 \\ &+ \left[\frac{f(n+1)-2f(n)+f(n-1)}{6h^2}\right]\ \tau^3 \\ &+ \frac{1}{2h^3}\left[\frac{f(n+1)}{6} - \frac{5f(n-1)}{2} - \frac{4f(n)}{3} + \frac{2x(n)-2x(n-1)}{h}\right]\ \tau^4 \end{aligned} \quad (111)$$

and $x(n+1)$ can be estimated by setting $\tau = h$ i.e.

$$\begin{aligned} x(n+1) &= x(n) + h\,[f(n)]\ + \frac{h}{2}\left[\frac{f(n+1)}{6} + \frac{8f(n)}{3} + \frac{7f(n-1)}{6} + \frac{4x(n-1)-4x(n)}{h}\right] \\ &+ \left[\frac{f(n+1)-2f(n)+f(n-1)}{6}\right]\ h \\ &+ \frac{1}{2}\left[\frac{f(n+1)}{6} - \frac{5f(n-1)}{2} - \frac{4f(n)}{3} + \frac{2x(n)-2x(n-1)}{h}\right]\ h \end{aligned} \quad (112)$$

For the purpose of computations, we can rearrange equation (112) as follows

$$x(n+1) = \alpha_0 x(n) + \alpha_1 x(n-1) + h\left[\beta_{-1} f(n+1) + \beta_0 f(n) + \beta_1 f(n-1)\right] \quad (113)$$

where

$$\alpha_0 = 0 \ ; \ \alpha_1 = 1 \quad ; \ \beta_{-1} = \frac{1}{3}; \ \beta_0 = \frac{1}{3}; \ \beta_1 = -\frac{1}{2}$$

Thus, after eliminating the polynomial coefficients $\{a_0(n), a_1(n), a_2(n)...\}$ the expressions for $x(n+1)$ involves some linear combination of current and past states $\{x(n), x(n-1), ...\}$ and derivatives $\{f(n+1), f(n), f(n-1), ...\}$. A more general expression for *computational form* of $x(n+1)$ can be stated as follows

$$\begin{aligned} x(n+1) &= \alpha_0 x(n) + \alpha_1 x(n-1) + ... + \alpha_p x(n-p) \\ &+ h\left[\beta_{-1} f(n+1) + \beta_0 f(n) + \beta_1 f(n-1) + ... + \beta_p f(n-p)\right] \end{aligned} \quad (114)$$

or

$$x(n+1) = \sum_{i=0}^{p} \alpha_i x(n-i) + h \sum_{i=-1}^{p} \beta_i f(n-i)$$

where p is an integer and α_i, β_i are real numbers to be selected. Note that if $\beta_{-1} \neq 0$, we get an implicit formula for the unknown quantity $x(n+1)$. Otherwise, we get an explicit formula. An algorithm of the type 114 is called $(p+1)$ step algorithm because $x(n+1)$ is given in terms of the values of x at previous $(p+1)$ steps $[x(n), .., x(n-p)]$. Order of the algorithm is the degree of the highest-degree polynomial for which 114 gives an exact expression of $x(n+1)$. To see how this definition of order is used, consider the development of the m'th order algorithm in scalar case. i.e., $x \in R$. (similar arguments can be used in vector case). Suppose polynomial solution of initial value problem is given by

$$x(t) = a_0(n) + a_1(n)t + a_2(n)t^2 + ... + a_m(n)t^m = \sum_{j=0}^{m} a_j(n)(t)^j \tag{115}$$

$$f(x,t) = \frac{dx}{dt} = a_1(n) + 2\left[a_2(n)\right]t + ... + m\ \left[a_m(n)\right]t^{m-1} = \sum_{j=1}^{m} j\left[a_j(n)\right](t)^{j-1} \tag{116}$$

Defining a shifted time scale, $\tau = t - t_n$, it follows that

$$t = t_{n-i} \equiv \tau = -ih \tag{117}$$

and

$$x(n+1) = x(h) = a_0(n) + a_1(n)h + a_2(n)h^2 + ... + a_m(n)h^m \tag{118}$$

$$x(n-i) = a_0(n) + \left[a_1(n)\right](-ih) + \left[a_2(n)\right](-ih)^2 + ... + \left[a_m(n)\right](-ih)^m \tag{119}$$

$$for \quad i = 0, 1, ..., p$$

$$f(n-i) = a_{1,n} + 2a_{2,n}(-ih) + ... + m\,a_{m,n}(-ih)^{m-1} \tag{120}$$

$$for \quad i = -1, 0, ...p$$

Substitution of equations (118),(119) and (120) into (114) gives what is known as the exactness constraints for the algorithm as

$$\begin{aligned}
\sum_{j=0}^{m}\left[a_j(n)\right](h)^j &= \sum_{i=0}^{p}\alpha_i\left[\sum_{j=0}^{m}\left[a_j(n)\right](-ih)^j\right] + h\sum_{i=-1}^{p}\beta_i\left[\sum_{j=1}^{m} j\left[a_j(n)\right](-ih)^{j-1}\right] \\
&= \left(\sum_{i=0}^{p}\alpha_i\right)a_0(n) + \left(\sum_{i=0}^{p}(-i)\alpha_i + \sum_{i=-1}^{p}(-i)^0\beta_i\right)\left[a_1(n)\right]h + ... \\
&\quad ... + \left(\sum_{i=0}^{p}(-i)^m\alpha_i + m\sum_{i=-1}^{p}(-i)^{m-1}\beta_i\right)\left[a_m(n)\right]h^m
\end{aligned} \tag{121}$$

It may be noted that we would like (121) to hold independently of any stepsize. This is achieved by equating like powers of h, which gives rise to the following set of constraints

$$\sum_{i=0}^{p} \alpha_i = 1\,; \quad (j=0) \tag{122}$$

$$\sum_{i=0}^{p} (-i)^j \alpha_i + j \sum_{i=-1}^{p} (-i)^{j-1} \beta_i = 1 \;; \quad (j = 1, 2, ..., m) \tag{123}$$

Note that in the above set of equations it is assumed that $(i)^j = 1$ when $i = j = 0$. Thus, equations (122-123) gives $m+1$ constraints and the number of variables are $2p+3$, namely $\alpha_0, ...\alpha_p, \beta_{-1}, \beta_0, ...\beta_p$. Any choice of these constants makes the corresponding algorithm 114 exact for m'th order polynomial. In order for the algorithm (114) to be exact in case of the m^{th} degree polynomial we must have

$$(m+1) \leq 2p+3 \tag{124}$$

If equality holds, i.e. when

$$m = 2(p+1)$$

then we can solve for $\{\alpha_i\}$ and $\{\beta_i\}$ exactly.

Now, let us re-derive the 2'nd order implicit algorithm again using the above approach. Constraints for this case can be generated by equating coefficients of

$$\begin{aligned} a_0(n) + [a_1(n)]\, h + [a_2(n)]\, h^2 &= \sum_{i=0}^{p} \alpha_i [a_0(n) + [a_1(n)]\,(-ih) + [a_2(n)]\,(-ih)^2] \\ &\quad + h \sum_{i=-1}^{p} \beta_i [a_1(n) + 2\,[a_2(n)]\,(-ih)] \end{aligned} \tag{125}$$

The resulting constraints are

$$\sum_{i=0}^{p} \alpha_i = 1 \tag{126}$$

$$\sum_{i=0}^{p} (-i\alpha_i) + \sum_{i=-1}^{p} \beta_i = 1 \tag{127}$$

$$\sum_{i=0}^{p} i^2 \alpha_i + \sum_{i=-1}^{p} (-2i\beta_i) = 1 \tag{128}$$

Clearly for (126-128) to hold, we must have $2p+3 \geq 3$. The second order algorithm with the smallest number of constants α_i, β_i is obtained by setting $2p+3 = 3$, i.e., $p = 0$. In this

case, we have

$$\begin{aligned} \alpha_0 &= 1 \\ \beta_{-1} + \beta_0 &= 1 \\ 2\beta_{-1} &= 1 \end{aligned} \tag{129}$$

which gives

$$\alpha_0 = 1; \qquad \beta_{-1} = 1/2; \qquad \beta_0 = 1/2 \tag{130}$$

and the second order algorithm becomes

$$x(n+1) = x(n) + \frac{h}{2}\left[f(n) + f(n+1)\right] \tag{131}$$

6.1.1 Examples of Multi-step methods

A number of multi-step algorithms can be obtained by making suitable choices of the parameters $\{\alpha_i\}$ and $\{\beta_i\}$. Some of the popular algorithms are discussed in this sub-section.

Adams-Bashforth Explicit Methods In this class of algorithms, we choose

$$\alpha_1 = \alpha_2 = ... = \alpha_p = 0 \tag{132}$$

$$\beta_{-1} = 0 \tag{133}$$

$$p = m - 1 \tag{134}$$

These are additional $(p+1)$ equations.

$$\begin{aligned} \text{Total number of constraints} &= (m+1) + (p+1). = 2m+1 \\ \text{Total number of variables} &= (2p+3) = 2m+1 \end{aligned}$$

Out of these, $(p+1=m)$ variables are selected to be zero and $(m+1)$ constants namely, $\alpha_0, \beta_0, ...\beta_p$ are to be detected. Using constraints for $j=0$,

$$\sum_{i=0}^{p} \alpha_i = 1; \Rightarrow \alpha_0 = 1 \tag{135}$$

Using the other constraints,

$$\begin{bmatrix} 1 & 1 & ... & 1 \\ 0 & (-1) & ... & (-p) \\ ... & ... & ... & ... \\ 0 & (-1)^{m-1} & ... & (-p)^{m-1} \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ ... \\ \beta_p \end{bmatrix} = \begin{bmatrix} 1 \\ 1/2 \\ ... \\ 1/m \end{bmatrix} \tag{136}$$

Solving for the $\beta' s$, we can write the algorithm as

$$x(n+1) = x(n) + h\left[\beta_0 f(n) + \beta_1 f(n-1) + ... + \beta_p f(n-p)\right] \tag{137}$$

Adam-Moulton Implicit Algorithms In this class of algorithms, we choose

$$\alpha_1 = \alpha_2 = ... = \alpha_p = 0 \tag{138}$$

which gives p constraints and set

$$p = m - 2 \tag{139}$$

For j = 0, we have

$$\sum_{i=0}^{p} \alpha_i = 1; \Rightarrow \alpha_0 = 1 \tag{140}$$

Remaining m variables $\beta_{-1}, ..., \beta_{m-2}$ can be determined by solving

$$\begin{bmatrix} 1 & 1 & ... & 1 \\ 0 & (-1) & ... & (-p) \\ ... & ... & ... & ... \\ 0 & (-1)^{m-1} & ... & (-p)^{m-1} \end{bmatrix} \begin{bmatrix} \beta_{-1} \\ \beta_0 \\ ... \\ \beta_p \end{bmatrix} = \begin{bmatrix} 1 \\ 1/2 \\ ... \\ 1/m \end{bmatrix} \tag{141}$$

The algorithm can be written as

$$\begin{aligned} x(n+1) &= x(n) + h\left[\beta_0 f(n) + \beta_1 f(n-1) + ... + \beta_p f(n-p)\right] + h\beta_{-1} f(n+1) && (142) \\ &= y_n + h\beta_{-1} f\left[x(n+1), t_{n+1}\right] && (143) \end{aligned}$$

where $y(n)$ represents sum of all terms which are known from the past data, i.e.

$$y_n = x(n) + h\left[\beta_0 f(n) + \beta_1 f(n-1) + ... + \beta_p f(n-p)\right] \tag{144}$$

The above implicit equation has to be solved iteratively to obtain $x(n+1)$.

6.1.2 Predictor-Corrector Algorithms

We saw that a m step Adams-Bashwforth algorithm is exact for polynomials of order m, while a m-step Adams-Moulton algorithm is exact for the polynomials of order $(m+1)$. However, the Adams-Moulton algorithm is implicit, i.e.,

$$x(n+1) = y(n) + h\beta_{-1} f\left[x(n+1), t_{n+1}\right] \tag{145}$$

where the quantity $y(n)$ depends on $x(n), ..., x(n-p)$ and is known. The above implicit equation can be solved iteratively as

$$x^{(k+1)}(n+1) = y(n) + h\beta_{-1} f\left[x^{(k)}(n+1), t_{n+1}\right] \tag{146}$$

where iterations are terminated when

$$|x^{(k+1)}(n+1) - x^{(k)}(n+1)| < \varepsilon \tag{147}$$

If we choose the initial guess $x^{(0)}(n+1)$ reasonably close to the solution, the convergence of the iterations is accelerated. To achieve this, we choose $x^{(0)}(n+1)$ as the value generated by an explicit $m-$step algorithm and then apply the iterative formula. This is known as the predictor-corrector method. For example, a two-step predictor-corrector algorithm can be given as

$$\text{Predictor:} \quad x^{(0)}(n+1) = x(n) + h\left[\frac{3}{2}f(n) - \frac{1}{2}f(n-1)\right] \tag{148}$$

$$\text{Corrector:} \quad x^{(k+1)}(n+1) = x(n) + h\left[\frac{1}{2}f(x^{(k)}(n+1), t_{n+1}) + \frac{1}{2}f(n)\right] \tag{149}$$

If the stepsize is selected properly, relatively few applications of the correction formula are enough to determine $x(n+1)$, with a high degree of accuracy.

Gear's Predictor-Corrector Algorithms A popular algorithm used for numerical integration is Gear's predictor corrector. The equations for this algorithm are as follows:

- Gear's m-th order predictor algorithm is an explicit algorithm, with

$$\begin{aligned} p &= m-1 && (150)\\ \beta_{-1} &= \beta_1 = ... = \beta_p = 0; \quad \beta_0 \neq 0 && (151)\\ x(n+1) &= \alpha_0 x(n) + \alpha_1 x(n-1) + ... + \alpha_p x(n-p) + h\beta_0 f(n) && (152) \end{aligned}$$

- Gear's m-th order corrector

$$\begin{aligned} p &= m-1 && (153)\\ \beta_0 &= \beta_1 = ... = \beta_p = 0; \quad \beta_{-1} \neq 0 && (154)\\ x(n+1) &= \alpha_0 x(n) + \alpha_1 x(n-1) + ... + \alpha_p x(n-p) + h\beta_{-1} f(n+1) && (155) \end{aligned}$$

The Gear's corrector formulae are also often referred to as dackward difference formulae (BDF). Coefficients of the above algorithm can be computed by setting up appropriate constraint equations as shown previously.

6.1.3 Multivariate Case

Even though the above derivations have been worked for one dependent variable case, these methods can be easily extended to multi-variable case

$$\frac{dx}{dt} = F(\mathbf{x}, t) \; ; \quad \mathbf{x} \in R^n \tag{156}$$

where $F(\mathbf{x}, t)$ is a $n \times 1$ function vector. In the multivariable extension, the scalar function $f(x, t)$ is replaced by the function vector $F(\mathbf{x}, t)$, i.e.

$$\begin{aligned} \mathbf{x}(n+1) &= \alpha_0 \mathbf{x}(n) + \alpha_1 \mathbf{x}(n-1) + ... + \alpha_p \mathbf{x}(n-p) \\ &\quad + h\left[\beta_{-1} F(n+1) + \beta_0 F(n) + \beta_1 F(n-1) + ... + \beta_p F(n-p)\right] \end{aligned}$$

where

$$\begin{aligned} F(n-i) &\equiv F\left[\mathbf{x}(t_n - ih), (t_n - ih)\right] \\ i &= -1, 0, 1, ...p \end{aligned}$$

and the scalar coefficients $\left\{\alpha_0 ....\alpha_p, \beta_{-1}, \beta_0, \beta_1, ...\beta_p\right\}$ are identical with the coefficients derived for the scalar case as described in the above section.

The main advantage of multi-step methods is that there are no extraneous 'inter-interval' calculations as in the case of Runge-Kutta methods. These methods can be used for stiff equations if the integration interval is chosen carefully. However, since the time instances should be uniformly spaced, selection of the integration interval is a critical issue.

6.2 Numerical Solution using Orthogonal Collocations

Other method based on polynomial interpolation is orthogonal collocations. Consider the scalar ODE-IVP given by equations (91), which has to be integrated over interval $[t_n, t_{n+1} = t_n + h]$. Defining scaled time variable τ as

$$\tau = \frac{t - t_n}{h}$$

the ODE-IVP can be transformed as follows

$$\frac{dx}{d\tau} = h f(x, t_n + h\tau) \; ; \quad x(0) = x(n) \tag{157}$$

It may be noted that $x(1)$ now corresponds to $x(t_{n+1})$. We illustrate here how this transformed ODE-IVP can be solved using three internal collocation points between $[0, 1]$. Assuming that we have 3 internal collocation points at roots of the 3'rd order shifted Legandre polynomial, we define five time points

$$\tau_1 = 0, \tau_2 = 0.1127, \tau_3 = 0.5, \tau_4 = 0.8873 \text{ and } \tau_5 = 1$$

Let us define a vector $\mathbf{x}$ such that i^{th} element of $\mathbf{x}$ corresponds to the value of x at $\tau = \tau_i$

$$\mathbf{x}_i = x(\tau_i)$$

Since the initial value is specified, it follows that

$$\mathbf{x}_1 = x(0) = x(n) \tag{158}$$

Now, using $\mathbf{S}$ matrix defined in Module on Problem Discretization using Approximation Theory, we can set up the following algebraic constraints

$$\left[\mathbf{s}^{(i)}\right]^T \mathbf{x} = hf(\mathbf{x}_i, t_n + h\tau_i) \tag{159}$$

for $i = 2, 3, 4, 5$. Equations (158) and (159) can be combined and rearranged as follows

$$\begin{bmatrix} 3.87 & 2.07 & -1.29 & 0.68 \\ -3.23 & 0 & 3.23 & -1.5 \\ 1.29 & -2.07 & -3.87 & 5.32 \\ -1.88 & 2.67 & -14.79 & 13 \end{bmatrix} \begin{bmatrix} \mathbf{x}_2 \\ \mathbf{x}_3 \\ \mathbf{x}_4 \\ \mathbf{x}_5 \end{bmatrix} = \begin{bmatrix} hf(\mathbf{x}_2, t_n + h\tau_2) \\ hf(\mathbf{x}_3, t_n + h\tau_3) \\ hf(\mathbf{x}_4, t_n + h\tau_4) \\ hf(\mathbf{x}_5, t_n + h\tau_5) \end{bmatrix} - \begin{bmatrix} -5.32 \\ 1.5 \\ -0.68 \\ 1 \end{bmatrix} x(n)$$

The resulting set of nonlinear algebraic equations can be solved using any standard approach such as Newton's method or Leverberg-Marquardt method. The solution yields $x(t_{n+1}) = x(\tau = 1) = \mathbf{x}_5$ along with the values of x at intermediate time points. Generalization of this approach to the case with more number of internal collocation points in interval [0,1] is straightforward and not discussed separately.

To see how this method can be extended to deal with a vector differential equation, consider coupled system of ODE IVPs

$$\frac{dx}{d\tau} = hf_1(x, y, t_n + h\tau) \; ; \quad x(0) = x(n) \tag{160}$$

$$\frac{dy}{d\tau} = hf_2(x, y, t_n + h\tau) \; ; \quad y(0) = y(n) \tag{161}$$

Defining a vectors $\mathbf{x}$ and $\mathbf{y}$ such that

$$\mathbf{x}_i = x(\tau_i) \quad \text{and} \quad \mathbf{y}_i = y(\tau_i)$$

we have to solve coupled set of equations

$$\left[\mathbf{s}^{(i)}\right]^T \mathbf{x} = hf(\mathbf{x}_i, \mathbf{y}_i, t_n + h\tau_i) \tag{162}$$

$$\left[\mathbf{s}^{(i)}\right]^T \mathbf{y} = hf(\mathbf{x}_i, \mathbf{y}_i, t_n + h\tau_i) \tag{163}$$

for $i = 2, 3, 4{,}5$ simultaneously. The final rearranged set of coupled equations are as follows

$$\begin{bmatrix} 3.87 & 2.07 & -1.29 & 0.68 \\ -3.23 & 0 & 3.23 & -1.5 \\ 1.29 & -2.07 & -3.87 & 5.32 \\ -1.88 & 2.67 & -14.79 & 13 \end{bmatrix} \begin{bmatrix} \mathbf{x}_2 \\ \mathbf{x}_3 \\ \mathbf{x}_4 \\ \mathbf{x}_5 \end{bmatrix} = \begin{bmatrix} hf(\mathbf{x}_2, \mathbf{y}_2, t_n + h\tau_2) \\ hf(\mathbf{x}_3, \mathbf{y}_3, t_n + h\tau_3) \\ hf(\mathbf{x}_4, \mathbf{y}_4, t_n + h\tau_4) \\ hf(\mathbf{x}_5, \mathbf{y}_5, t_n + h\tau_5) \end{bmatrix} - \begin{bmatrix} -5.32 \\ 1.5 \\ -0.68 \\ 1 \end{bmatrix} x(n)$$

$$\begin{bmatrix} 3.87 & 2.07 & -1.29 & 0.68 \\ -3.23 & 0 & 3.23 & -1.5 \\ 1.29 & -2.07 & -3.87 & 5.32 \\ -1.88 & 2.67 & -14.79 & 13 \end{bmatrix} \begin{bmatrix} \mathbf{y}_2 \\ \mathbf{y}_3 \\ \mathbf{y}_4 \\ \mathbf{y}_5 \end{bmatrix} = \begin{bmatrix} hf(\mathbf{x}_2, \mathbf{y}_2, t_n + h\tau_2) \\ hf(\mathbf{x}_3, \mathbf{y}_3, t_n + h\tau_3) \\ hf(\mathbf{x}_4, \mathbf{y}_4, t_n + h\tau_4) \\ hf(\mathbf{x}_5, \mathbf{y}_5, t_n + h\tau_5) \end{bmatrix} - \begin{bmatrix} -5.32 \\ 1.5 \\ -0.68 \\ 1 \end{bmatrix} y(n)$$

Defining vector

$$\mathbf{z} = \begin{bmatrix} \mathbf{x}_2 & .. & \mathbf{x}_5 & \mathbf{y}_2 & .. & \mathbf{y}_5 \end{bmatrix}^T$$

the above set of coupled nonlinear algebraic equations can be arranged in the standard form

$$\mathbf{A}\mathbf{z} = \mathbf{G}(\mathbf{z})$$

and can be solved using a suitable iteration scheme.

7 Convergence Analysis and Selection of Integration Interval

Selection of integration interval is a crucial parameter while solving ODE-IVPs numerically. In this section, we provide some insights into this aspect. The methods developed in this module are primarily meant for numerically solving nonlinear ODE-IVPs. However, for the purpose of analysis, we apply them on linear ODE-IVPs. This is because the true solution of linear ODE-IVPs can be found and thus can be used as a basis for carrying out the convergence analysis.

7.1 Analysis of Linear ODE-IVPs

To see how choice of integration interval can affect solution behavior, consider a scalar linear ODE-IVP of the form

$$\frac{dx}{dt} = ax; \qquad x(0) = x_0 \tag{164}$$

where $a < 0$. Analytical (true) solution of the above equation is given as

$$x^*(t) = e^{at}x_0 \tag{165}$$

Defining $x^*(t_n) = x^*(n)$, we can write true solution as a difference equation as follows

$$\begin{aligned} x^*(n) &= e^{anh}x_0 && (166)\\ x^*(n+1) &= e^{a(n+1)h}x_0 && (167)\end{aligned}$$

or

$$x^*(n+1) = e^{ah}x^*(n) \tag{168}$$

Now consider the approximate solution of the above ODE-IVP by explicit Euler methods

$$\begin{aligned} x(n+1) &= x(n) + hf(n)\\ &= (1+ah)x(n) \\ \Rightarrow x(n) &= (1+ah)^n x_0 \end{aligned} \tag{169}$$

Defining approximation error introduced due to numerical integration,

$$e(n) = x^*(n) - x(n) \tag{170}$$

we can write

$$e(n+1) = (1+ah)e(n) + \left[e^{ah} - (1+ah)\right] x^*(n) \tag{171}$$

Thus, the combined equation becomes

$$\begin{bmatrix} e(n+1) \\ x^*(n+1) \end{bmatrix} = \begin{bmatrix} (1+ah) & \left[e^{ah} - (1+ah)\right] \\ 0 & e^{ah} \end{bmatrix} \begin{bmatrix} e(n) \\ x^*(n) \end{bmatrix} \tag{172}$$

Now, since $a < 0$, i.e. the ODE (164) is asymptotically stable. As a consequence, for the difference equation (168) we have $e^{ah} < 1$ and $x^*(n) \to 0$ as $n \to \infty$, i.e. the solution of the difference equation is also asymptotically stable. Thus, we can expect that the approximate solution $\{x(n)\}$ should exhibit similar behavior qualitatively and $e(n) \to 0$ as $n \to \infty$. This requires that the difference equation given by (172) should be asymptotically stable, i.e., all eigenvalues of matrix

$$B = \begin{bmatrix} (1+ah) & \left[e^{ah} - (1+ah)\right] \\ 0 & e^{ah} \end{bmatrix}$$

should have magnitude strictly less than one. The eigenvalues of matrix B can be computed by solving the characteristic equation of B, i.e.

$$\det(\lambda I - B) = \left[\lambda - (1+ah)\right]\left[\lambda - e^{ah}\right] = 0$$

Thus, the approximation error $e(n) \to 0$ as $n \to \infty$ provided the following condition holds

$$\begin{aligned} |\lambda_1| &= |1+ah| < 1 && (173)\\ &\Rightarrow -2 < ah < 0 && (174)\end{aligned}$$

This inequality gives constraint on the choice of integration interval h, which will ensure that approximation error will vanish asymptotically.

Following similar line of arguments, we can derive conditions for choosing integration interval for different methods. For example,

- **Implicit Euler**

$$\begin{bmatrix} e(n+1) \\ x^*(n+1) \end{bmatrix} = \begin{bmatrix} \frac{1}{(1-ah)} & \left[e^{ah} - \frac{1}{(1-ah)}\right] \\ 0 & e^{ah} \end{bmatrix} \begin{bmatrix} e(n) \\ x^*(n) \end{bmatrix} \tag{175}$$

  Convergence conditions can be stated as follows

$$\left|\frac{1}{1-ah}\right| < 1 \tag{176}$$

  Since it is assumed that $a < 0$ and the step size h is always positive, the condition given by equation (176) is satisfied for any positive value of h. As a consequence, the implicit Euler has much better convergence properties when compared with the explicit Euler method.

- **Trapeziodal Rule (Simpson's method):**

$$\begin{bmatrix} e(n+1) \\ x^*(n+1) \end{bmatrix} = \begin{bmatrix} \frac{1+(ah/2)}{1-(ah/2)} & \left[e^{ah} - \frac{1+(ah/2)}{1-(ah/2)}\right] \\ 0 & e^{ah} \end{bmatrix} \begin{bmatrix} e(n) \\ x^*(n) \end{bmatrix} \tag{177}$$

  Convergence conditions can be stated as follows

$$\left|\frac{1+(ah/2)}{1-(ah/2)}\right| < 1 \tag{178}$$

  Since it is assumed that $a < 0$ and the step size h is always positive, the condition given by equation (178) is satisfied for any positive value of h. As a consequence, the Simpson's method has much better convergence properties when compared with the explicit Euler method.

- **2'nd Order Runge Kutta Method**

$$\begin{bmatrix} e(n+1) \\ x^*(n+1) \end{bmatrix} = \begin{bmatrix} \left(1+ah+\frac{(ah)^2}{2}\right) & \left[e^{ah} - \left(1+ah+\frac{(ah)^2}{2}\right)\right] \\ 0 & e^{ah} \end{bmatrix} \begin{bmatrix} e(n) \\ x^*(n) \end{bmatrix} \tag{179}$$

  Convergence conditions can be stated as follows

$$\left|1+ah+\frac{(ah)^2}{2}\right| < 1 \Rightarrow -2 < ah + \frac{(ah)^2}{2} < 0 \tag{180}$$

Thus, the choice of integration interval depends on the parameters of the equation to be solved and the method used for solving ODE IVP. These simple example also shows that the approximation error analysis gives considerable insight into relative merits of different numerical methods for solving ODE-IVPs. For example, in the case of implicit Euler or Simpson's rule, the approximation error asymptotically reduces to zero for any choice of $h > 0$. (Of course, larger the value of h, less accurate is the numerical solution.) This, however, is not true for explicit Euler method or Runge-Kutta 2'nd order methods. This clearly shows that implicit Euler method and Simpson's rule are superior to explicit Euler method.

The above analysis can be easily extended to a coupled system of linear ODE-IVP of the form

$$\begin{aligned} \frac{d\mathbf{x}}{dt} &= \mathbf{A}\mathbf{x} && (181) \\ \mathbf{x}(0) &= \mathbf{x}_0 \text{ at } t = 0 && (182) \end{aligned}$$

where $\mathbf{x} \in R^n$ and $\mathbf{A}$ is a $(n \times n)$ matrix. Let us further assume that all eigenvalues of $\mathbf{A}$ are in the left half of the complex plane i.e.

$$\text{Re}\,[\lambda_i(\mathbf{A})] < 1 \text{ for all i}$$

The true solution is given as follows (see Appendix for details)

$$\mathbf{x}^*(t) = \exp(\mathbf{A}t)\mathbf{x}_0$$

or in discrete settings

$$\mathbf{x}^*(n+1) = \exp(\mathbf{A}h)\mathbf{x}^*(n)$$

When matrix $\mathbf{A}$ is diagonalizable, i.e. $\mathbf{A} = \mathbf{\Psi}\mathbf{\Lambda}\mathbf{\Psi}^{-1}$, we can write

$$\begin{aligned} \exp(\mathbf{A}t) &= \mathbf{\Psi}\exp(\mathbf{\Lambda}t)\mathbf{\Psi}^{-1} \\ \exp(\mathbf{\Lambda}t) &= \begin{bmatrix} \lambda_1 & 0 & .. & 0 \\ 0 & \lambda_2 & .. & 0 \\ 0 & .. & ... & 0 \\ 0 & .. & 0 & \lambda_n \end{bmatrix} \end{aligned}$$

Since all eigenvalues of matrix $\mathbf{A}$ have -ve real part, it follows that

$$\begin{aligned} \exp(\lambda_i h) &= \exp(\alpha_i h + j\beta_i h) \\ &= \exp(\alpha_i h)\left[\cos(\beta_i h) + j\sin(\beta_i h)\right] \\ |\exp(\lambda_i h)| &= \exp(\alpha_i h) < 1 \quad for\ all\ \ i \\ \exp(\mathbf{\Lambda}t) &\rightarrow [\mathbf{0}] \quad \text{as} \quad n \rightarrow \infty \end{aligned}$$

and $\|\mathbf{x}^*(n)\| \rightarrow 0$ as $n \rightarrow \infty$.Then, following similar arguments as in the scalar case, it can be shown that condition for choosing the integration interval are as follows

- **Explicit Euler:** Approximate solution can governed by the following difference equation

$$\mathbf{x}(n+1) = [\mathbf{I} + h\mathbf{A}]\,\mathbf{x}(n)$$

and the dynamics of the approximation error, $\mathbf{e}(n) = \mathbf{x}(n) - \mathbf{x}^*(n)$, is governed by the folling matrix difference equation

$$\begin{bmatrix} \mathbf{e}(n+1) \\ \mathbf{x}^*(n+1) \end{bmatrix} = \begin{bmatrix} [\mathbf{I}+h\mathbf{A}] & [\exp(\mathbf{A}t) - \mathbf{I}+h\mathbf{A}] \\ 0 & \exp(\mathbf{A}t) \end{bmatrix} \begin{bmatrix} \mathbf{e}(n) \\ \mathbf{x}^*(n) \end{bmatrix} \tag{183}$$

$$\rho\,[\mathbf{I} + h\mathbf{A}] < 1 \tag{184}$$

where $\rho(.)$ represents spectral radius of the matrix $[\mathbf{I} + h\mathbf{A}]$. When matrix $\mathbf{A}$ is diagonalizable, we can write

$$\mathbf{I} + h\mathbf{A} = \mathbf{\Psi}\,[\mathbf{I} + h\mathbf{\Lambda}]\,\mathbf{\Psi}^{-1}$$

and eigen values of matrix $\mathbf{I}+h\mathbf{A}$ are $(1+h\lambda_i)$ for $i = 1, 2, ..., n$ where $\{\lambda_i : i = 1, 2, ..., n\}$ represent the eigenvalues of matrix $\mathbf{A}$. Thus, the convergence condition can be stated as

$$|1 + h\lambda_i| < 1 \text{ for } i = 1, 2, ..., n$$

- **Implicit Euler:** Approximate solution can governed by the following difference equation

$$\mathbf{x}(n+1) = (I - h\mathbf{A})^{-1}\mathbf{x}(n) \tag{185}$$

and the dynamics of the approximation error, $\mathbf{e}(n) = \mathbf{x}(n) - \mathbf{x}^*(n)$, is governed by the folling matrix difference equation

$$\begin{bmatrix} \mathbf{e}(n+1) \\ \mathbf{x}^*(n+1) \end{bmatrix} = \begin{bmatrix} (I - h\mathbf{A})^{-1} & [\exp(\mathbf{A}t) - (I - h\mathbf{A})^{-1}] \\ 0 & \exp(\mathbf{A}t) \end{bmatrix} \begin{bmatrix} \mathbf{e}(n) \\ \mathbf{x}^*(n) \end{bmatrix} \tag{186}$$

In order that $\|\mathbf{e}(n)\| \to 0$ as $n \to \infty$, the following condition should hold

$$\rho\left[(I - h\mathbf{A})^{-1}\right] < 1$$

When matrix A is diagonalizable, the convergence condition can be stated as

$$\left|\frac{1}{1 - h\lambda_i}\right| < 1 \text{ for } i = 1, 2, ..., n$$

- **Trapeziodal Rule:** Approximate solution can governed by the following difference equation

$$\mathbf{x}(n+1) = [I - (h/2)\mathbf{A}]^{-1}[I + (h/2)\mathbf{A}]\,\mathbf{x}(n) \tag{187}$$

and the dynamics of the approximation error, $\mathbf{e}(n) = \mathbf{x}(n) - \mathbf{x}^*(n)$, is governed by the folling matrix difference equation

$$\begin{bmatrix} \mathbf{e}(n+1) \\ \mathbf{x}^*(n+1) \end{bmatrix} = \Phi \begin{bmatrix} \mathbf{e}(n) \\ \mathbf{x}^*(n) \end{bmatrix} \tag{188}$$

$$\Phi = \begin{bmatrix} [I - (h/2)\mathbf{A}]^{-1}[I + (h/2)\mathbf{A}] & \left[\exp(\mathbf{A}t) - [I - (h/2)\mathbf{A}]^{-1}[I + (h/2)\mathbf{A}]\right] \\ 0 & \exp(\mathbf{A}t) \end{bmatrix}$$

In order that $\|\mathbf{e}(n)\| \to 0$ as $n \to \infty$, the following condition should hold

$$\rho\left[\left(I - \frac{h}{2}\mathbf{A}\right)^{-1}\left(I + \frac{h}{2}\mathbf{A}\right)\right] < 1 \tag{189}$$

When matrix A is diagonalizable, the convergence condition can be stated as

$$\left|\frac{1 + (\lambda_i h/2)}{1 - (\lambda_i h/2)}\right| < 1 \text{ for } i = 1, 2, ..., n \tag{190}$$

Similar error analysis (or stability analysis) can be performed for other integration methods. For example, consider the scenario when the 3-step algorithm is used for obtaining the numerical solution of equation (164).

$$\begin{aligned} x(n+1) &= \beta_{-1}hf(n+1) + \alpha_0 x(n) + \alpha_1 x(n-1) + \alpha_2 x(n-2) \\ &= a\beta_{-1}hx(n+1) + \alpha_0 x(n) + \alpha_1 x(n-1) + \alpha_2 x(n-2) \\ &= \frac{1}{1 - a\beta_{-1}h}[\alpha_0 x(n) + \alpha_1 x(n-1) + \alpha_2 x(n-2)] \\ &= \eta_0 x(n) + \eta_1 x(n-1) + \eta_2 x(n-2) \end{aligned}$$

The above difference equation can be rearranged in the following form.

$$\begin{bmatrix} x(n-1) \\ x(n) \\ x(n+1) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ \eta_2 & \eta_1 & \eta_0 \end{bmatrix} \begin{bmatrix} x(n-2) \\ x(n-1) \\ x(n) \end{bmatrix} \tag{191}$$

Defining

$$\mathbf{z}(n) = \begin{bmatrix} x(n-2) \\ x(n-1) \\ x(n) \end{bmatrix}; \quad \mathbf{z}(n+1) = \begin{bmatrix} x(n-1) \\ x(n) \\ x(n+1) \end{bmatrix}; \quad \mathbf{B} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ \eta_2 & \eta_1 & \eta_0 \end{bmatrix} \tag{192}$$

we have

$$\begin{aligned} \mathbf{z}(n+1) &= \mathbf{B}\mathbf{z}(n) \quad (193) \\ x(n+1) &= \mathbf{z}_3(n+1) \\ &= \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \mathbf{z}(n+1) = C\mathbf{z}(n+1) \quad (194) \end{aligned}$$

Similarly, the true solution can be expressed as

$$\begin{aligned} \mathbf{z}^*(n+1) &= \mathbf{B}^*\mathbf{z}^*(n) \quad (195) \\ x^*(n+1) &= C\mathbf{z}^*(n+1) \end{aligned}$$

where

$$\mathbf{B}^* = \begin{bmatrix} e^{ah} & 0 & 0 \\ 0 & e^{ah} & 0 \\ 0 & 0 & e^{ah} \end{bmatrix}$$

The evolution of the approximation error is given as

$$\begin{aligned} \mathbf{e}(n+1) &= \mathbf{B}\mathbf{e}(n) + [\mathbf{B}^* - \mathbf{B}]\, \mathbf{z}^*(n) \\ \mathbf{e}(n) &= \mathbf{z}^*(n) - \mathbf{z}(n) \end{aligned}$$

The stability criterion that can be used to choose integration interval h can be derived as

$$\rho(\mathbf{B}) < 1 \quad (196)$$

Note that characteristic equation for matrix B is given as

$$\lambda^3 - \eta_0\lambda^2 - \eta_1\lambda - \eta_2 = 0 \quad (197)$$

Thus, eigenvales of matrix $\mathbf{B}$ can be directly computed using the coefficients η_0, η_1 and $\eta_{2,}$ which are functions of integration interval h.

Equations such as (184), (185), (190) and (197) can be used to generate *stability envelopes* for each method in the complex plane (eigenvalues of a matrix can be complex). Stability envelopes for most of the methods are available in literature. The following general conclusions can be reached by studying these plots [3].

- Even though the first and second order Adams-Moulton methods (implicit Euler and Crank-Nicholson) are Asymptotically-stable, the higher order techniques have restricted regions of stability. These regions are larger than the Adams-Bashforth family of the same order.

- All forms of the R-K algorithms with order ≤ 4 have identical stability envelopes.
- Explicit R-K techniques have better stability characteristics than explicit Euler.
- For predictor-corrector schemes, accuracy of scheme improves with order. However, stability region shrinks with order.

7.2 Extension to Nonlinear ODE IVPs

The conclusions reached from studying linear systems can be extended to general nonlinear systems locally using Taylor expansion.

$$\frac{d\mathbf{x}}{dt} = \mathbf{F}(\mathbf{x}) \tag{198}$$

can be approximated as

$$\begin{aligned} \frac{d\mathbf{x}}{dt} &\cong \mathbf{F}(\mathbf{x}(n)) + \left[\frac{\partial \mathbf{F}}{\partial \mathbf{x}}\right]_{\mathbf{x}=\mathbf{x}(n)} (\mathbf{x} - \mathbf{x}(n)) && (199) \\ &\cong \left[\frac{\partial \mathbf{F}}{\partial \mathbf{x}}\right]_{\mathbf{x}=\mathbf{x}(n)} \mathbf{x} + \left[\mathbf{F}\left[\mathbf{x}(n)\right] - \left[\frac{\partial \mathbf{F}}{\partial \mathbf{x}}\right]_{\mathbf{x}=\mathbf{x}(n)} \mathbf{x}(n)\right] && (200) \\ &\cong (\mathbf{A})_n \mathbf{x} + (\mathbf{d})_n && (201) \end{aligned}$$

Applying some numerical technique to solve this problem will lead to a difference equation of the form

$$\mathbf{x}(n+1) = (\mathbf{B})_n \mathbf{x}(n) + (\mathbf{c})_n \tag{202}$$

and stability will depend on the choice of h such that $\rho[(\mathbf{B})_n] < 1$ for all n. Note that, it is difficult to perform global analysis for general nonlinear system.

7.3 Stiffness of ODEs [3]

The problem of integrating multi-variable ODE-IVP with some variables changing very fast in time while others changing slowly, is difficult to solve. This is because, the stepsize has to be selected according to the fastest changing variable / mode. For example, consider the equation

$$\begin{aligned} \frac{d}{dt}\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} &= \begin{bmatrix} -100 & 0 \\ 2 & -1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \\ A = \begin{bmatrix} -100 & 0 \\ 2 & -1 \end{bmatrix} \quad &; \quad y(0) = \begin{bmatrix} 2 \\ 1 \end{bmatrix} \end{aligned} \tag{203}$$

It can be shown that the solution for the above system of equations is

$$\begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} = \begin{bmatrix} 2e^{-100t} \\ \frac{103}{99}e^{-t} - \frac{4}{99}e^{-100t} \end{bmatrix} \tag{204}$$

It can be observed that the terms with e^{-100t} lead to a sharp decrease in $y_1(t)$ and to a small maximum in $y_2(t)$ at $t = 0.0137$. The term $y_2(t)$ is dominated by e^{-t} which decreases slowly. Thus,

$$y_1(t) < 0.01 y_1(0) \qquad for \qquad t > 0.03 \tag{205}$$

$$y_2(t) < 0.01 y_1(t) \qquad for \qquad t > 4.65 \tag{206}$$

Now, stepsize should be selected such that the faster dynamics can be captured. The stiffness of a given ODE-IVP is determined by finding the stiffness ratio defined as

$$S.R. = \frac{|Re\lambda_i(A)|_{\max}}{|Re\lambda_i(A)|_{\min}} \tag{207}$$

where matrix A is defined above. Systems with '*large*' stiffness ratio are called as stiff.

This analysis can be extended to a general nonlinear systems only locally. Using Taylor's theorem, we can write

$$\frac{d\mathbf{x}}{dt} = F(\mathbf{x}) = F\left[\mathbf{x}(n) + \mathbf{x}(t) - \mathbf{x}(n)\right] \tag{208}$$

$$\cong F(\mathbf{x}_n) + \left[\frac{\partial F}{\partial \mathbf{x}}\right]_{\mathbf{x}=\mathbf{x}(n)} \left[\mathbf{x}(t) - \mathbf{x}(n)\right] \tag{209}$$

Local stiffness ratio can be calculated using eigenvalues of the Jacobian and the ODE-IVP is locally stiff if the local S.R. is high, i.e., the system has at least one eigenvalue which does not contribute significantly over most of the domain of interest. In general, eigenvalues of the Jacobian are time dependent and S.R. is a function of time. Thus, for stiff systems it is better to use variable step size methods or special algorithms for stiff systems.

7.4 Variable stepsize implementation with accuracy monitoring [2]

One practical difficulty involved in the integration with fixed stepsize is the choice of stepsize such that the approximation errors are kept small. If a system of nonlinear ODEs is stiff only in certain regions of the state space, then selecting a fixed step size is a non-trivial task. In such a situation, a variable stepsize algorithm is implemented with error monitoring as given in Table 2. It may be noted that above algorithm can be implemented with any Runge-Kutta class methods.

Table 2: Variable Size Implememtation of Runge-Kutta Algorithms

Given: $t_n, \mathbf{x}(n) = \mathbf{x}(t_n), \varepsilon$

Step 1: Choose stepsize h_1 and let $t_{n+1}^{(1)} = t_n + h_1$

Step 2: Compute $\mathbf{x}^{(1)}(n+1)$ using the chosen method (e.g. Euler / Runge-Kutta etc.).

Step 3: Define $h_2 = h_1/2; \quad t_{n+1}^{(2)} = t_n + h_2$

$t_{n+2}^{(2)} = t_n + 2h_2 \quad (= t_{n+1}^{(1)})$

Compute $\mathbf{x}_{n+1}^{(2)}$ *and* $\mathbf{x}_{n+2}^{(2)}$ using the chosen method

Step 4:

IF $(\left\|\mathbf{x}^{(1)}(n+1) \; - \; \mathbf{x}^{(2)}(n+2)\right\| < \; \varepsilon)$,

Accept $\mathbf{x}^{(1)}(n+1)$ as the new value

Set $\mathbf{x}(n+1) \; = \; \mathbf{x}^{(1)}(n+1)$, and $n = n \; + 1$ and proceed to Step 1.

ELSE

Set $h_1 = h_2$ and proceed to the step 2.

END IF

8 Solutions of Differential Algebraic System of Equations

Differential algebraic equations (DAEs) is an important class of problems that arise in many engineering applications. Such equations often appear while developing dynamic models of systems, which involve multiple physical phenomenon each occurring at different time scale. For example, while modeling a distillation column, the dynamics associated vapor phase is significantly faster than the dynamics associated with the liquid phase. Another context where such equations arise is while solving ODE-BVPs or PDEs by discretization using orthogonal collocations method.

A general DAE system can be expressed as follows

$$\mathbf{F}\left[\mathbf{X}, \frac{d\mathbf{X}}{dt}, t\right] = \overline{\mathbf{0}} \tag{210}$$

with $\mathbf{X}(0) = \mathbf{X}_0$ and $\mathbf{F}[.] : R^n \rightarrow R^n$ represents a nonlinear function vector. DAEs that cannot be represented by any further simplification are referred to as *fully implicit*. The DAEs that can be further simplified are classified as follows

- **Linear implicit**:

$$\mathbf{A}\frac{d\mathbf{X}}{dt} + \mathbf{F}[\mathbf{X}, t] = \overline{\mathbf{0}} \tag{211}$$

with $\mathbf{X}(0) = \mathbf{X}_0$.

- **Semi-explicit:**

$$\frac{d\mathbf{x}}{dt} = \mathbf{f}[\mathbf{x}, \mathbf{z}, t] \quad (212)$$

$$\overline{\mathbf{0}} = \mathbf{g}[\mathbf{x}, \mathbf{z}] \quad (213)$$

where vector $\mathbf{x}$ represents differential variables, vector $\mathbf{z}$ represents algebraic variables and $\mathbf{X}^T = \left[\mathbf{x}^T\ \mathbf{z}^T\right]$.

Here, we only consider the semi-explicit DAEs. DAEs are solved using extensions of ODE solvers. There are two approaches for solving DAEs numerically:

- **Nested Approach:** This is probably the most commonly used approach**.**
  - Given $\mathbf{x}(n)$, solve for $\mathbf{g}[\mathbf{x}(n), \mathbf{z}(n)] = \overline{\mathbf{0}}$ and compute $\mathbf{z}(n)$
  - Using an ODE method, evolve $\mathbf{x}(n+1) = \mathbf{x}(n+1) + \int_{t_n}^{t_{n+1}} \mathbf{f}[\mathbf{x}(\tau), \mathbf{z}(\mathbf{x}(\tau)), \tau]\, d\tau$

  This approach requires $\mathbf{z} = \mathbf{z}(\mathbf{x})$ (implicit function) and used if only an explicit solver is available (e.g., explicit Euler or Runge-Kutta). The approach can be can be computationally expensive due to inner iterative calculations involved in solving for $\mathbf{g}[\mathbf{x}(\tau), \mathbf{z}(\tau)] = \overline{\mathbf{0}}$ given $\mathbf{x}(\tau)$.

- **Simultaneous Approach:** Solves equations (212-213) simultaneously using an implicit method, such as implicit Euler or BDF, to evolve both $\mathbf{x}(n+1)$ and $\mathbf{z}(n+1)$ in time. This approach is much more efficient and allows more flexible problem specification. Consider a Gear's corrector (or BDF) solver. For a semi-explicit system, we can write

$$\mathbf{x}(n+1) = h\beta_{-1}\mathbf{f}[\mathbf{x}(n+1), \mathbf{z}(n+1), t_{n+1}] + \alpha_0\mathbf{x}(n) + \alpha_1\mathbf{x}(n-1) + \ldots\ldots + \alpha_p\mathbf{x}(n-p)$$

$$\overline{\mathbf{0}} = \mathbf{g}[\mathbf{x}(n+1), \mathbf{z}(n+1)]$$

and the resulting system of nonlinear algebraic equations can be solved simultaneously for computing $\mathbf{x}(n+1)$ and $\mathbf{z}(n+1)$ using, say, Newton's method.

A detailed treatment of solution approaches for solving DAE systems can be found in Ascher and Petzoldt [4].

9 Solution of ODE-BVP using Shooting Method [3]

In this section, we present a technique for solving an ODE-BVP using numerical algorithms for solving ODE-IVPs in combination with a technique for solving nonlinear algebraic equation. By this approach, the missing conditions at one end point of the independent variable (say at $z = 0$) are guessed, which converts the ODE-BVP into an initial value problem. We then integrate or *shoot* to the other boundary point (i.e. $z = 1$) and use the boundary conditions at $z = 1$ as algebraic constraints on the dependent variables that must be satisfied by the solution. The guess of initial value that satisfies the boundary constraints yields the solution to the ODE-BVP. The steps involved in single shooting method can be summarized as follows:

- Step 1: Represent the ODE-BVP into a standard form

$$\frac{d\mathbf{x}}{dz} = F(\mathbf{x})$$

- Step 2: Assume the 'missing' initial conditions at $z = 0$.
- Step 3: Integrate (shoot) the ODE-IVPs from $z = 0$ to $z = 1$ as if it is an ODE-IVP using any standard numerical integration method for solving ODE-IVP
- Step 4: Check whether all the specified boundary conditions are satisfied at $z = 1$. If the BC at $z = 1$ is satisfied, terminate the iterations. Else, use method such as Newton's method / secant method / optimization to generate new guess values at $z = 0$ and go to step 2.

We illustrate this idea using a specific examples.

Example 5 *The ODE-BVP describing tubular reactor with axial mixing (TRAM) in which an irreversible 2nd order reaction is carried out is given as*

$$\frac{1}{Pe}\frac{d^2C}{dz^2} - \frac{dC}{dz} - DaC^2 = 0 \qquad (0 < z < 1) \tag{214}$$

$$z = 0 : \frac{dC}{dz} = Pe(C - 1)$$
$$z = 1 : \frac{dC}{dz} = 0$$

where C is the dimensionless concentration, z is axial position, Pe is the Peclet number for mass transfer and Da is the Damkohler number. Now, defining new state variables

$$x_1 = C \qquad and \qquad x_2 = \frac{dC}{dz} \tag{215}$$

we can transform the above ODE's as

$$\frac{d}{dz}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ (Pe)x_2 + (Da.Pe)x_1^2 \end{bmatrix} \equiv \mathbf{F}(\mathbf{x}) \tag{216}$$

$$z = 0 : x_2(0) = Pe\,[x_1(0) - 1] \tag{217}$$

$$z = 1 : x_2(1) = 0 \tag{218}$$

The single shooting method solves this problem by converting it into a sequence of ODE-IVP problems as follows: Step 1: Guess $x_1(0) = \alpha$*, which determines* $x_2(0) = Pe(\alpha - 1)$*,Step 2: Integrate the two ODE's simultaneously using any standard ODE-IVP solver from* $z = 0$ *to* $z = 1$ *and Step 3: Examine whether the given boundary condition,* $f(\alpha) = x_2(1) = 0$*, is satisfied, i.e. stop iterations if* $|x_2(1)| < \varepsilon$ *else go to Step 1. Defining*

$$f(\alpha) = x_2(1) \tag{219}$$

the value of α *can be changed from iteration to iteration by the secant method as follows*

$$\alpha^{(k+1)} = \alpha^{(k)} - f[\alpha^{(k)}]\left[\frac{\alpha^{(k)} - \alpha^{(k-1)}}{f[\alpha^{(k)}] - f[\alpha^{(k-1)}]}\right] \tag{220}$$

Alternatively, we can use Newton's method for generating search directions

$$\alpha^{(k+1)} = \alpha^{(k)} - \left[\frac{f[\alpha^{(k)}]}{[df/d\alpha]_{\alpha=\alpha^{(k)}}}\right] \tag{221}$$

The derivative $[df/d\alpha]_{\alpha=\alpha^{(k)}}$ *can be computed by simultaneously integrating the sensitivity equations. Given a set of the first order nonlinear equations*

$$\frac{d\mathbf{x}}{dz} = F(\mathbf{x}) \;\; ; \;\; \mathbf{x}(0) = \mathbf{x}_0 \; ; \; \mathbf{x} \in \mathbf{R}^n \tag{222}$$

and $F(\mathbf{x})$ *is a* $n \times 1$ *vector, the associated sensitivity equations are defined as*

$$\frac{d\Phi(z)}{dz} = \left[\frac{\partial F}{\partial \mathbf{x}}\right]\Phi(z) \;\; ; \;\; \Phi(0) = \mathbf{I} \tag{223}$$

where

$$\Phi(z) = \left[\frac{\partial \mathbf{x}(z)}{\partial \mathbf{x}_0}\right]$$

represents the $n \times n$ *sensitivity of solution vector* $\mathbf{x}(z)$ *with respect to the initial conditions and* $\mathbf{I}$ *denotes identity matrix. In the present case, the sensitivity equations are*

$$\frac{d\Phi}{dz} = \begin{bmatrix} 0 & 1 \\ 2D_aP_ex_1 & P_e \end{bmatrix}\Phi(z) \tag{224}$$

$$\Phi(z) = \begin{bmatrix} \dfrac{\partial x_1(z)}{\partial x_1(0)} & \dfrac{\partial x_1(z)}{\partial x_2(0)} \\ \dfrac{\partial x_2(z)}{\partial x_1(0)} & \dfrac{\partial x_2(z)}{\partial x_2(0)} \end{bmatrix}$$

These equations have to be integrated from $z = 0$ to $z = 1$ to evaluate

$$[df/d\alpha]_{\alpha=\alpha^{(k)}} = \Phi_{21}(1) = \frac{\partial x_2(1)}{\partial x_1(0)}$$

Another possibility is to formulate an optimization problem as follows

$$\begin{aligned} &\min_{\alpha} \ [x_2(1)]^2 \\ &\textit{Subject to} \\ \frac{d\mathbf{x}}{dz} &= F(\mathbf{x}) \quad for \quad z \in (0,1] \\ \mathbf{x}(0) &= \begin{bmatrix} \alpha \\ Pe(\alpha - 1) \end{bmatrix} \end{aligned}$$

This optimization problem can be solved using any standard approach, such as conjugate gradient method. Given a guess $\alpha^{(k)}$, each optimization iteration requires an ODE-IVP solver to compute $x_2(1)$.

Example 6 *Consider the problem of axial conduction and diffusion in a tubular reactor*

$$\begin{aligned} \frac{1}{2}\frac{d^2C}{dz^2} - \frac{dC}{dz} - kC\exp(\eta - \frac{\eta}{T}) &= 0 \\ \frac{1}{2}\frac{d^2T}{dz^2} - \frac{dT}{dz} - \beta kC\exp(\eta - \frac{\eta}{T}) &= 0 \end{aligned}$$

defined over $0 < z < 1$ together with boundary conditions

$$\begin{aligned} At\ z &= 0: \ \frac{1}{2}\frac{dC}{dz} = C - 1 \quad and \quad \frac{1}{2}\frac{dT}{dz} = T - 1 \\ At\ z &= 1: \ \frac{dC}{dz} = \frac{dT}{dz} = 0 \end{aligned}$$

Here k,α and β are known constants. It is desired to solve this coupled set of ODE-BVP using single shooting method. Defining new state variables

$$x_1 = C, \ x_2 = \frac{dC}{dz}, \ x_3 = T, \ x_4 = \frac{dT}{dz} \tag{225}$$

the coupled ODEs can be transformed as follows

$$\frac{d}{dz}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} x_2 \\ 2x_2 + 2kx_1\exp(\eta - \frac{\eta}{x_2}) \\ x_4 \\ 2x_4 + 2k\beta x_1\exp(\eta - \frac{\eta}{x_2}) \end{bmatrix} \equiv \mathbf{F}(\mathbf{x})$$

$$z = 0 : x_2(0) = 2\,[x_1(0) - 1] \tag{226}$$

$$z = 0 : x_4(0) = 2\,[x_3(0) - 1] \tag{227}$$

$$z = 1 : x_2(1) = 0 \tag{228}$$

$$z = 1 : x_4(1) = 0 \tag{229}$$

If we guess $x_1(0) = \alpha$ *and* $x_3(0) = \beta$, *then the initial condition becomes*

$$\mathbf{x}(0) = \left[\begin{array}{cccc} \alpha & 2\,[\alpha - 1] & \beta & 2\,[\beta - 1] \end{array} \right]^T$$

We can formulate an optimization problem as follows

$$\begin{aligned} & \min_{\alpha,\,\beta} \ \omega_1 \left[x_2(1)\right]^2 + \omega_2 \left[x_4(1)\right]^2 \\ & \textit{Subject to} \\ \frac{d\mathbf{x}}{dz} &= \mathbf{F}(\mathbf{x}) \quad for \quad z \in (0, 1] \\ \mathbf{x}(0) &= \left[\begin{array}{cccc} \alpha & 2\,[\alpha - 1] & \beta & 2\,[\beta - 1] \end{array} \right]^T \end{aligned}$$

where ω_1,$\omega_2 > 0$ *are suitable weights. The resulting optimization problem can be solved using any standard approach, such as conjugate gradient method or quasi-Newton method, in combination with a suitable ODE-IVP solver.*

10 Summary

In these lecture notes, we have studied numerical methods for solving ODE-IVPs. In particular, we have discussed development of numerical algorithms based on

- Taylor series approximations (Runge-Kutta methods)
- Polynomial interpolation based algorithms (Predictor-corrector type methods and Orthogonal Collocation).

In the end, we provide a brief introduction to the stability analysis of the numerical algorithms for solving ODE-IVPs.

11 Exercise

1. Express the following set of equations in the standard form

$$d\mathbf{x}/dt = \mathbf{A}\mathbf{x} \quad \text{with Initial Condition } \mathbf{x}(0)$$

(a) Set 1

$$d^2y/dt^2 + 4dy/dt + 3y = 0;\ y(0) = 1; dy/dt = 0 \text{ at } t = 0$$

(b) Set 2

$$d^3y/dt^3 + 6d^2y/dt^2 + 11dy/dt + 6y = 0$$

$$y(0) = 1;\ dy/dt = d^2y/dt^2 = 0 \text{ at } t = 0;$$

(c) Set 3

$$dy/dt + 3y + z = 0; y(0) = 1$$

$$d^2z/dt^2 + 3dz/dt + 2z = 0$$

$$z(0) = 1; dz/dt = 0$$

Based on the eigenvalues of $\mathbf{A}$, find conditions on the choice of integration interval if it is desired to use (i) Crank-Nicholson method (trapezoidal rule) (ii) explicit Euler .

2. Consider the PDE given below

$$\partial C/\partial t = \partial^2 C/\partial z^2$$

$$C(0,t) = C(1,t) = 0 \text{ for all } 0 \leq t \leq \infty$$

$$C(z,0) = 1 \text{ for } 0 \leq z \leq 1$$

(a) Use the finite difference technique on the dimensionless diffusion equation obtain a set of ODE-IVPs assuming N internal grid points. Particularly for the case $N = 2$, Based on the eigenvalues of $\mathbf{A}$, find conditions on the choice of integration interval if it is desired to use (i) implicit euler and (ii) explicit Euler method.

(b) Repeat the above exercise using orthogonal collocation to discretize in space with two internal collocation points.

(c) Based on the eigenvalues of $\mathbf{A}$, find conditions on the choice of integration interval if it is desired to use (i) Crank-Nicholson method (trapezoidal rule) (ii) Runge Kutta 2'nd order method.

3. Consider Van der Pol equation given below

$$d^2y/dt^2 - (1 - y^2)dy/dt + 3y = 0$$

$$y(0) = 2; dy/dt = 0 \text{ at } t = 0$$

(a) Express the above ODE-IVP in standard form

$$d\mathbf{x}/dt = F(\mathbf{x});\ \mathbf{x} = \mathbf{x}(0) \text{ at } t = 0$$

(b) Using Taylor series approximation, linearize the resulting equation in the neighborhood of $\overline{\mathbf{x}} = [\ 0 \quad 0\ ]^T$ and obtain the perturbation ODE-IVP of the form

$$\begin{aligned} d(\delta\mathbf{x})/dt &= \mathbf{A}\delta\mathbf{x} \\ \mathbf{A} &= \left[\frac{\partial F}{\partial \mathbf{x}}\right]_{(0,0)} \end{aligned}$$

where $\delta\mathbf{x}(t) = \mathbf{x}(t) - \overline{\mathbf{x}}$.

(c) Based on the decomposition of $\mathbf{A} = \mathbf{\Psi}\Lambda\mathbf{\Psi}^{-1}$ using eigen values and eigen vectors of matrix A, find true solution to the linearized ODE-IVP in the following form

$$\delta\mathbf{x}(t) = \left[\mathbf{\Psi}\exp(\Lambda t)\mathbf{\Psi}^{-1}\right]\delta\mathbf{x}(0)$$

(d) Based on the eigenvalues of $\mathbf{A}$, find conditions on the choice of integration interval if it is desired to use (i) Crank-Nicholson method (trapezoidal rule) (ii) Runge Kutta 2'nd order method.

4. The steady state behavior of an isothermal tubular reactor with axial mixing, in which a first order irreversible reaction is carried out, is represented by following ODE-BVP

$$\frac{d^2C}{dz^2} - \frac{dC}{dz} - 6C = 0$$

$$At\ z = 0 : \frac{dC}{dz} = C(0) - 1\ \ ;\ \ At\ \ z = 1 : \frac{dC}{dz} = 0$$

Represent the above second order equation in the standard form $d\mathbf{x}/dz = \mathbf{A}\mathbf{x}$ by appropriately defining a state vector $\mathbf{x}$. Compute $exp(\mathbf{A}z) = \Psi\exp(\Lambda z)\Psi^{-1}$ using eigen values and eigen vectors of matrix $\mathbf{A}$.Find the missing initial condition at $z = 0$ such that the analytical solution

$$\mathbf{x}(z) = exp(\mathbf{A}z)\mathbf{x}(0)$$

satisfies the boundary condition at $z = 1$.

5. It is desired to develop an implicit multi-step method for the following scalar ODE-IVP

$$\frac{dx}{dt} = f(x,t)\ \ ;\ \ x(t_n) = x(n)$$

for inegrating over the interval $[t_n, t_{n+1}]$ using an interpolation polynomial of the form

$$x(t) = a_{0,n} + a_{1,n}t + a_{2,n}t^2 + a_{3,n}t^3$$

Here, $t_{n+1} = t_n + h$ and h represents fixed integration step size. Find the interpolation polynomial coefficients in terms of $x(n), x(n-1), f(n)$ and $f(n+1)$ and derive expression for $x(n+1)$.

6. It is desired to solve the following scalar ODE-IVP

$$\frac{dx}{dt} = f(x,t) \;\; ; \;\; x(t_n) = x(n) \tag{230}$$

using Milne's multi-step algorithm. The Milne's implicit formulae for solving ODE-IVPs are obtained by imposing following additional constraints

$$\alpha_0 = \alpha_2 = \alpha_3 = ... = \alpha_p = 0 \text{ and } \alpha_1 \neq 0$$

along with the exactness constraints and selecting $p = m - 2$. Find the coefficients of the 3'rd order Milne's implicit algorithm (i.e. $m = 3$, $p = 1$) and state the final form of the integration algorithm.

7. It is desired to derive 3'rd order Gear's implicit integration formula of the form

$$x(n+1) = \alpha_0 x(n) + \alpha_1 x(n-1) + \alpha_2 x(n-2) + h\beta_{-1} f(n+1)$$

for numerically integrating an ODE-IVP of the form

$$dx/dt = f(x,t) \;\; ; \;\; I.c. : x(t_n) = x(n) \tag{231}$$

from $t = t_n$ to $t = t_n + 1$. Setup the necessary constraint equations and obtain coefficients $\{\alpha_i\}$ and β_{-1}.

8. Consider the following set of differential algebraic equations (DAE)

$$\frac{dx}{dt} = az + bx^2$$

$$z^3 + (c+x)z^2 + (dx - e) + f = 0$$

The initial values $x(n)$ and $z(n)$ at $t = t_n$ are known and we wish to integrate the equation to obtain $x(n+1)$ and $z(n+1)$ at $t_{n+1} = t_n + h$, where h represents the integration interval, using the *orthogonal collocation* method with two internal collocation points lying between $[t_n, t_{n+1}]$. Transform the DAE in terms of an independent variable τ such that $\tau = 0$ corresponds to $t = t_n$ and $\tau = 1$ corresponds to $t = t_{n+1} = t_n + h$. For the choice of internal collocation points at $\tau = 0.2$ and $\tau = 0.8$, write down the appropriate nonlinear algebraic equations that need to be solved simultaneously. What is

the degree of freedom (i.e. number of unknowns - number of equations) of the resulting set of nonlinear algebraic equations?

$$S = \begin{bmatrix} -7 & 8.2 & -2.2 & 1 \\ -2.7 & 1.7 & 1.7 & -0.7 \\ 0.7 & -1.7 & -1.7 & 2.7 \\ -1 & 2.2 & -8.2 & 7 \end{bmatrix} \; ; \; T = \begin{bmatrix} 24 & -37.2 & 25.2 & -12 \\ 16.4 & -24 & 12 & -4.4 \\ -4.4 & 12 & -24 & 16.4 \\ -12 & 25.2 & -37.2 & 24 \end{bmatrix}$$

9. It is desired to apply the method of finite difference to solve the following PDE

$$\frac{\partial C}{\partial t} = \frac{\partial^2 C}{\partial t^2}$$

$$\begin{aligned} \text{Boundary Conditions} &: \quad C(0,t) = C(1,t) = 0 \\ \text{Initial Condition} &: \quad C(z,0) = 1 \end{aligned}$$

where t and z represent dimensionless time and dimensionless length, respectively. Assuming 'n' equidistant grid points and defining vector

$$\mathbf{x} = \begin{bmatrix} C_1 & C_2 & ... & C_n \end{bmatrix}^T$$

we obtain the following set of ODE-IVP from the PDE

$$d\mathbf{x}/dt = A\mathbf{x} \; ; \; \mathbf{x}(0) = \begin{bmatrix} 1 & 1 & ... & 1 \end{bmatrix}^T$$

$$A = \frac{1}{(\Delta z)^2} \begin{bmatrix} -2 & 1 & 0 & .... & 0 \\ 1 & -2 & 1 & .... & 0 \\ .... & .... & .... & .... & .... \\ 0 & .... & 1 & -2 & 1 \\ 0 & .... & 0 & 1 & -2 \end{bmatrix}$$

(a) Suppose that it is desired to solve the resulting linear algebraic equations analytically as $\mathbf{x}(t) = [\Psi \exp(\Lambda t)\Psi^{-1}]\,\mathbf{x}(0)$ where $A = \Psi\Lambda\Psi^{-1}$. Show that vector

$$\mathbf{v}^{(k)} = \begin{bmatrix} \sin(k\pi\Delta z) & \sin(2k\pi\Delta z) & ..... & \sin(nk\pi\Delta z) \end{bmatrix}^T$$

is an eigenvector of matrix A with eigenvalue

$$\lambda_k = \frac{2}{(\Delta z)^2}\left[\cos(k\pi\Delta z) - 1\right]$$

where and $k = 1, 2, ...n$ and $\Delta z = 1/(n+1)$. (Show calculations for 1^{st}, $i^{th}$and the last row).

(b) Suppose, instead of solving the problem analytically, the set of ODE-IVP is to be integrated using Crank-Nicholson method (i.e. trapezoidal rule). Find the condition on the integration step size 'h' in terms of eigenvalues of matrix A so that the approximation error will decay exponentially and approximate solution will approach the true solution.

Note: Crank-Nicholson algorithm for the scalar case can be stated as

$$x(n+1) = x(n) + \frac{h}{2}\left[f(n) + f(n+1)\right]$$

10. A chemical reactor is modelled using the following set of ODE-IVP

$$\frac{dC}{dt} = \frac{1-C}{V} - 2C^2 \tag{232}$$

$$\frac{dV}{dt} = 1 - V \tag{233}$$

Linearize the above equations in the neighborhood of steady state $C = 0.5$ and $V = 1$ and develop a linear perturbation model. Obtain the analytical solution for the linearized system starting from initial condition $C = 0.7$ and $V = 0.8$. Also, compute stiffness ratio and comment upon asymptotic stability of the solution.

11. It is desired to integrate the following ODE-IVP using the explicit Euler method

$$\begin{aligned} \frac{d\mathbf{x}}{dt} &= \mathbf{A}\mathbf{x} \\ \mathbf{A} &= \begin{bmatrix} -6 & -11 & -6 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \\ \mathbf{x}(0) &= \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}^T \end{aligned}$$

Find the condition on the choice of integration step size h such that the approximation errors will asymptotically decay to zero.

(a) **Additional Information:** $\lambda = -1$ *and* $\lambda = -2$ *are eigenvalues of* $\mathbf{A}$.

References

References

[1] Brauer, F. and J. A. Nohel, The Qualitative Theory of Ordinary Differential Equations: An Introduction, Dover, 1969.

[2] Vidyasagar, M.; Nonlinear Systems Analysis. Prentice Hall, 1978.

[3] Gupta, S. K.; Numerical Methods for Engineers. Wiley Eastern, New Delhi, 1995.

[4] Ascher, U. M. and L. R. Petzoldt , Computer Methods for Ordinary Differential Equations and Differential Algebraic Equations., 1997.