# SNS COLLEGE OF TECHNOLOGY

**Coimbatore-35**
**An Autonomous Institution**

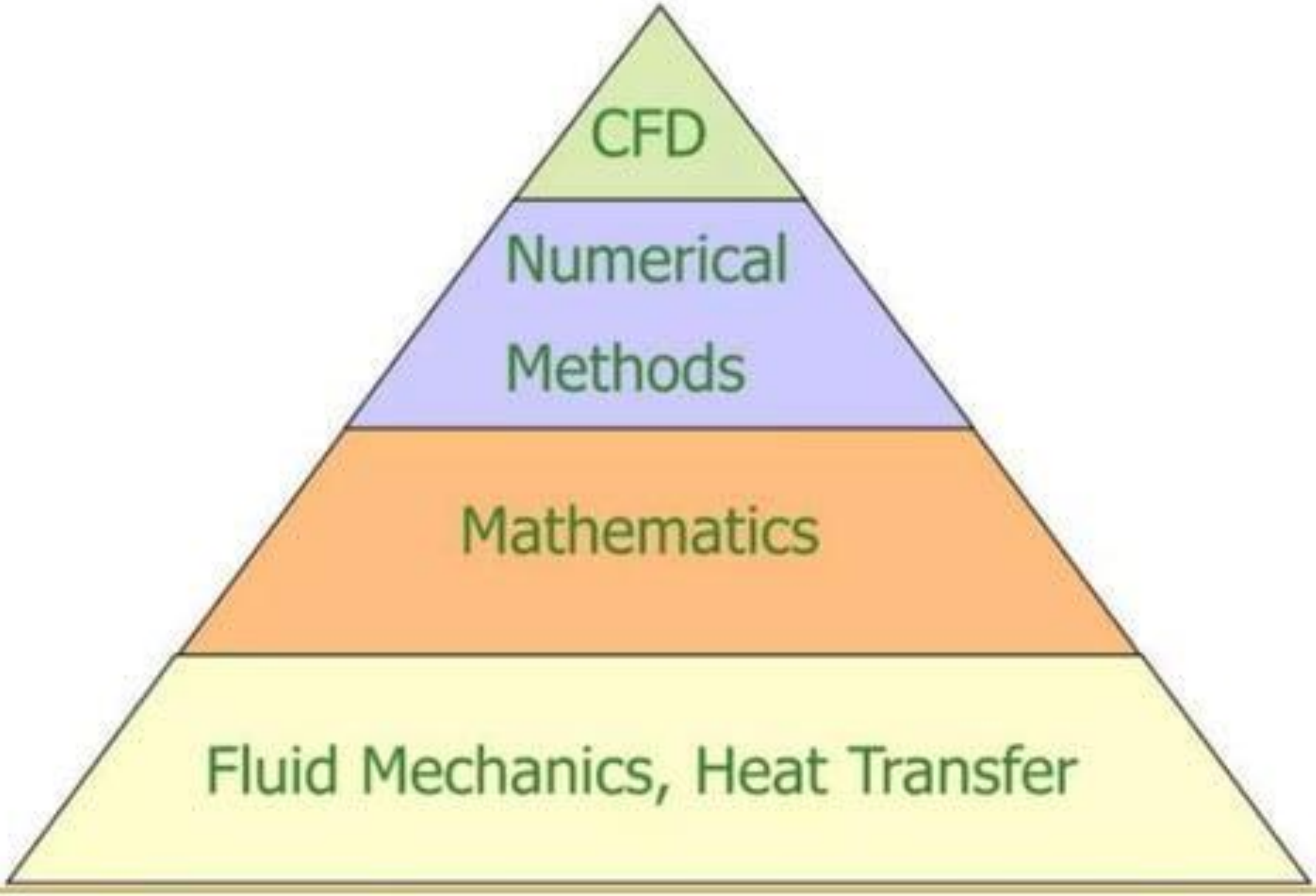
Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A++' Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

**DEPARTMENT OF AEROSPACE ENGINEERING**

**19ASB304 – COMPUTATIONAL FLUID DYNAMICS FOR AEROSPACE APPLICATIONS**
**III YEAR VI SEM**
**UNIT-I FUNDAMENTAL CONCEPTS**
**TOPIC: Explicit finite difference methods of subsonic, supersonic and viscous flows**

NAME: Mr.N.Venkatesh., M.Tech
Assistant Professor
Aerospace Engineering
SNS College of Technology

## Computational Fluid Dynamics (CFD)



## Computational Fluid Dynamics (CFD)

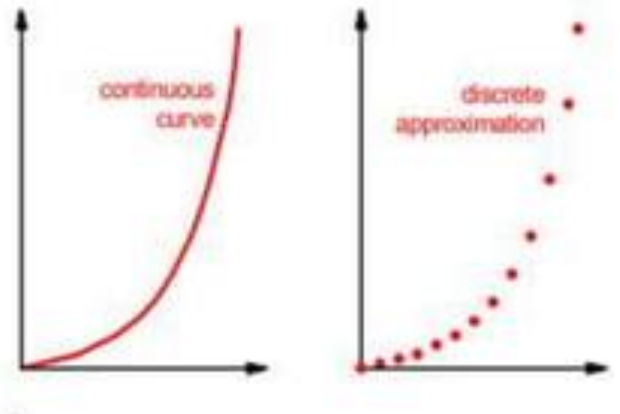
- Computational Fluid Dynamics (CFD) is the analysis of fluid flows using numerical solution methods.
- Simply put, CFD is the calculation of properties of a flowing fluid.
- Fluid dynamics is involved with physical laws in the form of partial differential equations.
- CFD calculations require simultaneous solution of many sets of differential equations

## Computational Fluid Dynamics (CFD)

Computational Fluid Dynamics (CFD) is the approximation of a continuously-varying quantity in terms of values at a finite number of points is called discretisation.

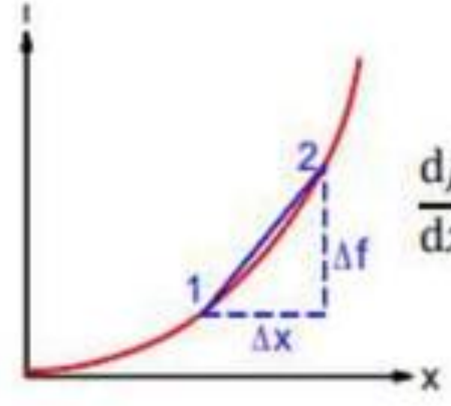
The following are common to any CFD simulation

(1) The flow field is discretised i.e. field variables $(\rho, u, v, w, p, \ldots)$ are approximated by their values at a finite number of node



## Computational Fluid Dynamics (CFD)

(2) The equations of motion are discretised:
derivatives → algebraic approximations
(continuous) (discrete)



$$\frac{df}{dx} \approx \frac{\Delta f}{\Delta x} = \frac{f_2 - f_1}{x_2 - x_1}$$

(3) The resulting system of algebraic equations is solved to give values at the nodes.

## Computational Fluid Dynamics (CFD)

- Fluid-Flow Equations
- The equations of fluid flow are based on fundamental physical conservation principles:
- • mass: change of mass = 0
- • momentum: change of momentum = force × time
- • energy: change of energy = work + heat In fluid flow
- These conservation principles may be expressed mathematically as either: • integral (control-volume) equations; • differential equations.

## Computational Fluid Dynamics (CFD)

- Integral (Control-Volume) Approach

  This describes how the total amount of a physical quantity (mass, momentum, energy, …) changes within a finite region of space (control volume). over an interval of time:
- CHANGE = (AMOUNT ENTERING – AMOUNT LEAVING) + AMOUNT CREATED

## Computational Fluid Dynamics (CFD)

In fluid mechanics this is usually expressed in rate form by dividing by the time interval (and transferring net transfer through the boundary to the LHS):

( TIME DERIVATIVE of amount in V ) + ( NET FLUX through boundary of V ) = ( SOURCE inside V )

## Computational Fluid Dynamics (CFD)

The flux (rate of transport through a surface) is further subdivided into:

advection1 – movement with the flow;

diffusion – net transport by random molecular or turbulent motion

( TIME DERIVATIVE of amount in V ) + ( ADVECTION+DIFFUSION through boundary of V ) = ( SOURCE inside V )

This is a canonical equation, independent of whether the physical quantity is mass, momentum, chemical content, etc

## Differential Equations

- In regions without shocks, interfaces or other discontinuities, fluid-flow equations can also be written in differential forms (Section 2). These describe what is going on at a point rather than over a whole control volume. Mathematically, they can be derived by making the control volumes infinitesimally small. There are many ways of writing these differential equations. Finite-difference methods approximate some differential form of the governing equations

## Scalar

- A scalar is any physical property which can be represented by a single real number in some chosen unit system.
- *e.g.* pressure (P), temperature (T) and density (ρ).
- Scalars are denoted by single letters in italics, *e.g.* P ,T , ρ
- The standard scalar operations must be performed using consistent units of measurement; in particular, addition, subtraction and equality are only physically meaningful for scalars of the same dimensional units.

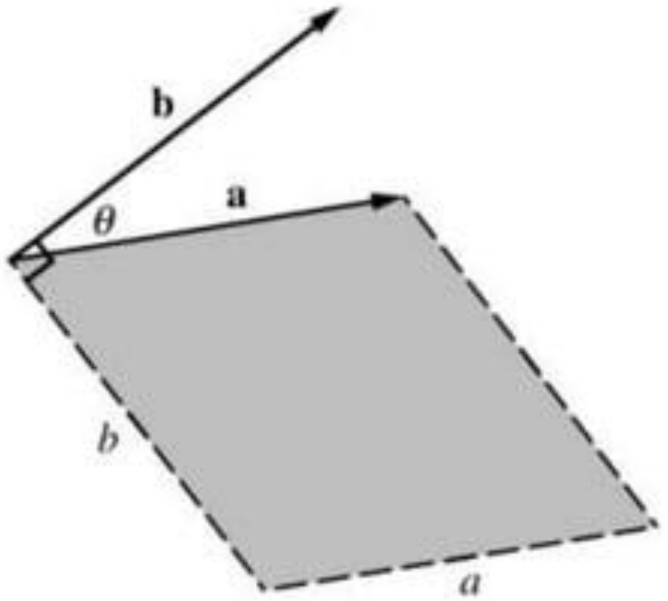
## Vector

- A vector is an entity which is represented by both magnitude and direction. In its most general form an n-dimensional vector a can be denoted by n scalar components $(a_1, a_2, \ldots a_n)$ corresponding to coordinate axes $(x_1, x_2 \ldots x_n)$ .
- Generally we deal with 3 dimensional space, the vector $a=(a_1, a_2, a_3)$ relates to a general set of axes $x_1, x_2, x_3$; representing x,y,z in a rectangular Cartesian system or r, θ,z and r, θ,φ in cylindrical and spherical polar coordinates.
- The *index notation* presents the same vector as ai (i=1,2,3..n) in which corresponds to each of the coordinate axes.

## The scalar product of two vectors

- The scalar product of two vectors $a(a_1,a_2,a_3)$ and $b(b_1,b_2,b_3)$ is defined as $a.b=a_1b_1+a_2b_2+a_3b_3=a_ib_i$
- $a.b=b.a$
- $a.(b+c)=a.b+a.c$
- The geometrical representation of the scalar product is $a.b=ab\cos\theta$ as depicted by the shaded area in figure next

## The scalar product of two vectors



## The vector product of two vectors

The vector product of a
vector $a=(a_1,a_2,a_3)$ with $b=(b_1,b_2,b_3)$ is defined as
$a*b=(a_2b_3-a_3b_2,a_3b_1-a_1b_3,a_1b_2-a_2b_1)=e_{ijk}a_jb_k$
where the *permutation symbol*
$e_{ijk}=0$ when any two indices are equal
$e_{ijk}=+1$ when i,j,k are an even permutation of 1,2,3
$e_{ijk}=-1$ when i,j,k are an odd permutation of 1,2,3

## The vector product of two vectors

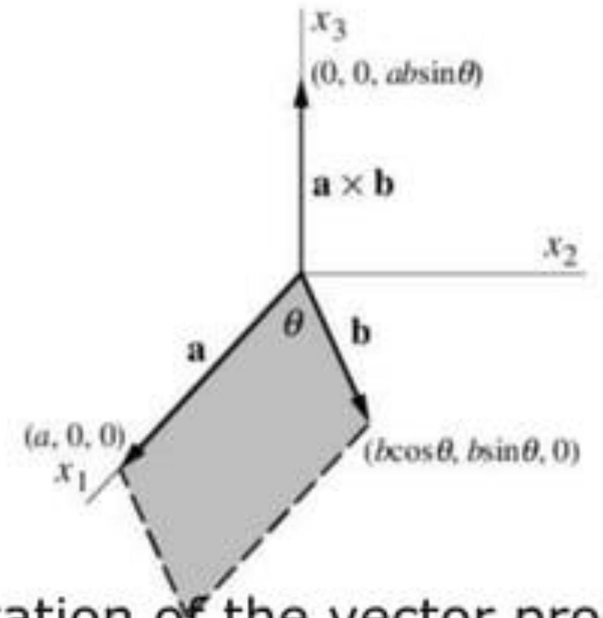
- where the even permutations are 123, 231 and 312 and the odd permutations are 132, 213 and 321.
- The following behaviour is observed:

a*a=0

a*b=-b*a

a*(b+c)=a*b+a*c

## The vector product of two vectors



The geometrical representation of the vector product can be illustrated by defining a and b to lie in the $x_1x_2$ plane of a rectangular coordinate system $ox_1x_2x_3$,a=(a,0,0) and b=(bcos θ,bsin θ,0)

The vector product is then a*b=(0,0,absin θ) to follow the direction of z axis

Therefore, the vector product represents a normal vector of magnitude equal to the area of a parallelogram described by vectors a and b

## Second Rank Tensors

- A vector is itself a first rank tensor and a scalar is a tensor of rank zero.

- A second rank tensor is defined here as a linear vector function.

- The tensor acts as a linear vector function as follows: $u_i = T_{ij} V_j$

## Second Rank Tensors

An -n dimensional second rank
tensor, T or $T_{ij}$ has components $n^2$ which can be expressed in a array corresponding to axes $x_1, x_2, \ldots, x_n$ as

$$\mathbf{T} = T_{ij} = \begin{pmatrix} T_{11} & T_{12} & \ldots & T_{1n} \\ T_{21} & T_{22} & \ldots & T_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ T_{n1} & T_{n2} & \ldots & T_{nn} \end{pmatrix}$$

the 3-dimensional tensor with 9 components will be used to present tensor algebra in array notation:

$$\mathbf{T} = T_{ij} = \begin{pmatrix} T_{11} & T_{12} & T_{13} \\ T_{21} & T_{22} & T_{23} \\ T_{31} & T_{32} & T_{33} \end{pmatrix}$$

## The single dot product

- Equation  ui=Tijvj can be written in tensor notation as a single dot product operation pairing one geometric vector to another (expanding the vector in a column for convenience)

$$\mathbf{u} = \mathbf{T} \cdot \mathbf{v} = T_{ij} v_j = \begin{pmatrix} T_{11} v_1 + T_{12} v_2 + T_{13} v_3 \\ T_{21} v_1 + T_{22} v_2 + T_{23} v_3 \\ T_{31} v_1 + T_{32} v_2 + T_{33} v_3 \end{pmatrix}$$

## The scalar product of two tensors

- The scalar product of two tensors is denoted by T:S which can be evaluated as the sum of the nine products of the tensor components

$$\begin{aligned}\mathbf{T}:\mathbf{S} = T_{ij}S_{ij} = \; & T_{11}S_{11} + T_{12}S_{12} + T_{13}S_{13} + \\ & T_{21}S_{21} + T_{22}S_{22} + T_{23}S_{23} + \\ & T_{31}S_{31} + T_{32}S_{32} + T_{33}S_{33}\end{aligned}$$

## The tensor product of two vectors

- The tensor product of two vectors, denoted by ab (sometimes denoted a*b), is defined by the requirement that (ab).v=a(b.v) for all and produces a tensor whose components are evaluated as:

$$\mathbf{ab} = a_i b_j = \begin{pmatrix} a_1b_1 & a_1b_2 & a_1b_3 \\ a_2b_1 & a_2b_2 & a_2b_3 \\ a_3b_1 & a_3b_2 & a_3b_3 \end{pmatrix}$$

- The derivation of the principal equations of fluid dynamics is based on the fact that the dynamical behaviour of a fluid is determined by the following conservation laws,

  namely: 1. the conservation of mass

  2. the conservation of momentum

  3. the conservation of energy
- CFD: Attempts to solve the basic (differential) conservation equations.

- The goal of CFD is to replace these differential equations with algebraic approximations, which can be computationally solved using numerical methods, a grid is formed over the domain of interest to calculate variables of discrete points. Variables are both functions of time and space.
- Goal of cfd is to solve a very large set of algebraic equations written for each discrete position verses time within the flow.

## Explicit method and implicit method

- The **explicit method and implicit method** are numerical analysis methods used to solve a time-dependent differential equation.

The explicit method calculates the system status at a future time from the currently known system status. The implicit method calculates the system status at a future time from the system statuses at present and future times.

- For example, when there is a differential equation $y=f(y,t)$ , the **explicit method** expresses it as $y_{n+1}=y_n+hf(y_n,t_n)$
- That is, if you know the state at n, you can calculate the state at n+1.

## Explicit method and implicit method

- **The implicit method**

- On the contrary, the **implicit method** has the state at n+1 on the right-hand side as in $y_{n+1}=y_n+hf(y_n,t_{n+1})$ .
- The explicit method is easier to program and can be calculated within a shorter time.But its stability is so low that you need to use a step size small enough to prevent divergence.
- On the contrary, the implicit method has high stability and converges if you set proper parameters. But, as you need to solve an equation at every step, it takes a long time to calculate.

## Explicit method and implicit method

- Mathematically, if $Y(t)$ is the current system state and $Y(t+\Delta t)$ is the state at the later time ($\Delta t$ is a small time step), then, for an explicit method
  $Y(t+\Delta t) = F(Y(t))$
- while for an implicit method one solves an equation $G(Y(t)\,,\,Y(t+\Delta t)\,)=0$ to find $Y(t+\Delta t)$

## Explicit method and implicit method

- **A Physical Example**
- An elementary physical problem involving the propagation of a pressure wave can be used to illustrate the differences between implicit and explicit methods. Imagine an increase in pressure is applied to one end of an organ pipe that is closed at the opposite end. We know that a pressure wave will move down the pipe and be reflected at the closed end. Given enough time, pressure waves will travel back and forth in the pipe many times before the pressure distribution settles down to the constant value applied at the open end.

## Explicit method and implicit method

- If only steady-state results are wanted, then an implicit solution scheme with lots of damping of the pressure waves should be used so that steady conditions will be reached as quickly as possible. In this case the damping incorporated in the implicit iteration method (i.e., the under-relaxation) is highly desirable.
- If, instead, the transient pressure waves are to be investigated, then we want the least amount of numerical damping so that many wave reflections can be accurately followed. This situation is best treated with an explicit solution method.

## Explicit method and implicit method

- Explicit methods require a time-step size that limits the advance of the pressure step to less than one computational cell per time step. However, this restriction is related to accuracy because most difference equations involve quantities from neighboring cells only. A pressure wave that propagates further than one cell in one time step would then be moving into regions that have no defined influence on the pressure. Not only is this physically unrealistic, it also leads to numerical instability.

## Explicit method and implicit method

- Implicit methods, on the other hand, couple all the cells together through an iterative solution that allows pressure signals to be transmitted through a grid. The price for this communication between distantly located cells is a damping or smoothing of the pressure waves introduced by the under-relaxation needed to solve the coupled equations.
- The choice of whether an implicit versus explicit method should be used ultimately depends on the goal of the computation. When time accuracy is important, explicit methods produce greater accuracy with less computational effort than implicit methods.

## Explicit method and implicit method

- example of initial value problem in ODE

$y(t)=f(t,y)$    $y(t0)=y0$

The forward Euler's method is one such numerical method and is explicit.

Let tk, k=0,1,2,…, be a sequnce in time with

$tk+1=tk+ \Delta t$

Let yk and Yk be the exact and the approximate solution at t = tk , respectively

then

$Yk+1=Yk + f(tk,Yk) \Delta t$

As the step size or Δ decreases then the error between the actual and approximation is reduced

## Explicit method and implicit method

- The backward Euler's method is an implicit one which contrary to explicit methods finds the solution by solving an equation involving the current state of the system and the later one. More precisely

- $Yk+1=Yk + f(tk,Yk+1)\ \Delta t$

- This disadvantage to using this method is the time it takes to solve this equation. However, advantages to this method include that they are usually more numerically stable for solving a stiff equation a larger step size $\Delta$ can be used

$$y' + 2y = 2 - e^{-4t}, \quad y(0) = 1, \qquad 0 \le t \le 0.5,$$

we will use forward and the backward Euler's method to approximate the solution to this problem and these approximations to the exact solution

$$y(t) = 1 + 0.5 * e^{-4t} - 0.5 * e^{-2t}.$$

In both methods we let $\Delta t = 0.1$ and the final time $t_f = 0.5.$

**Table** 1a. Forward and Backward Euler's Method Compared To Exact Solution

| $t_n$ (time) | $Y_n$, forward Euler's approximation | $Y_n$, backward Euler's approximation | $y_n$, exact | $\|e_i\|$ error forward | $\|e_i\|$ error backward |
|---|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 0 | 0 |
| 0.1 | 0.9 | 0.9441 | 0.925795 | 0.025795 | 0.018305 |
| 0.2 | 0.853 | 0.916 | 0.889504 | 0.036504 | 0.026496 |
| 0.3 | 0.8374 | 0.9049 | 0.876191 | 0.03791 | 0.028709 |
| 0.4 | 0.8398 | 0.9039 | 0.876191 | 0.036391 | 0.027709 |
| 0.5 | 0.8517 | 0.9086 | 0.883728 | 0.032028 | 0.024872 |