Numerical Methods for Partial Differential Equations

Hamza Farooq, Muhammad Awais

2016-CS-122, 2016-CS-109

INTRODUCTION

Differential conditions are a standout amongst the most essential portrayals of the laws that utilize marvels including the development of bodies in a gravitational field or the development of populaces. Here we are going to explain numerical methods for parabolic partial differential equations.

When we turn in finding the solution of Partial differential equations in numerical analysis, we often mean approximation. Because computational machines (Rather mechanical or electrical) don't know the analytic method to solve a mathematical problem. So we have to utilize those numerical techniques in computers to get the closest approximation.

Here we are going to discuss numerical methods that involve partitioning the original problem into smaller parts of that problem. An initial condition is given and approximations are made consequently. After that rough formulation is made. We repeat the process until we get required accuracy.

***Index Terms* – PDE:** Partial Differential Equation, **ODE:** Ordinary Differential Equation, **FDM:** Finite Difference Method, **FEM:** Finite Element Method, **CFL:** Courant–Friedrichs–Lewy, **DG:** Discontinuous Galerkin Method, **CG:** Continuous Galerkin Method

ORDINARY DIFFERENTIAL EQUATION

If all functions appearing in the equation depend only on one variable, we speak of an ordinary differential equation. Ordinary differential equations frequently describe the behavior of a system over time, e.g., the movement of an object depends on its velocity, and the velocity depends on the acceleration.

PARTIAL DIFFERENTIAL EQUATION

On the off chance that the capacities in the condition rely upon in excess of one variable and if the condition in this manner relies upon fractional subsidiaries, we discuss an incomplete differential condition.

A typical example is the potential equation of electrostatics. Given a domain R3, we consider

$$\frac{\partial^2 u}{\partial x_1^2}(x) + \frac{\partial^2 u}{\partial x_2^2}(x) + \frac{\partial^2 u}{\partial x_3^2}(x) = f(x) \ \ for\ all\ x \in \ \Omega$$

Computers have only a finite amount of storage at their disposal, they are generally unable to represent the solution u as an element of the in finite-dimensional space exactly.

- Processing Limitation
- Memory Limitation
- Limited Clock Cycles
- Can't learn analytical methods
- Can only approximate problems

A genuinely basic discretization strategy is the technique for finite differences: we supplant the subsidiaries by difference remainders and supplant by a framework h with the end goal that the difference remainders in the framework focuses can be assessed utilizing just qualities in network focuses. For the situation of the potential condition, this prompts an arrangement of straight conditions that can be fathomed with a specific end goal to get a guess uh of u.

METHOD PROCEDURES

Partial differential equations is the branch of numerical analysis that studies the numerical solution of partial differential equations (PDEs). It contains unknown functions containing more than one variable and their partial derivatives. PDEs are used to solve problems involving functions of more variables, and are either solved by hand, or computation is performed by computer.

A partial differential equation (PDE) for the function

$$u(x_1, \dots x_n)$$

is an equation of the form

$$f\left(x_1, \dots x_n, u, \frac{\partial u}{\partial x_1}, \dots, \frac{\partial u}{\partial x_n}, \frac{\partial^2 u}{\partial x_1 \partial x_1}, \dots, \frac{\partial^2 u}{\partial x_1 \partial x_n}, \dots\right) = 0$$

FINITE DIFFERENCE METHODS FOR PARABOLIC EQUATIONS

In Numerical Analysis, Finite Difference methods are concerned with approximating the derivatives of the difference equations derived from the given function problem are called finite difference methods.

Finite Difference methods are also called **discretization methods.**

DERIVATION FROM TAYLOR'S POLYNOMIAL

First we find the function whose derivatives are properly calculated.

$$f(x_0+h)=f(x_0)+\frac{f'(x_0)}{1!}h+\frac{f^{(2)}(x_0)}{2!}h^2 + \cdots \frac{f^{(n)}(x_0)}{n!}h^n+R_n(x)$$

Where n! represents the factorial of n, and Rn(x) the term that involve approximation errors and also called remainder term. It represents the difference between the taylor polynomial and the remainder term of degree n. We will derive an approximation for the first derivative of the function "f" by first truncating the Taylor polynomial:

$$f(x_0+h)=f(x_0)+\ f'(x_0)h+R_1(x)$$

Put $x_0 = a$ we have

$$f(a+h)=f(a)+\ f'(a)h+R_1(x)$$

Dividing by h we have

$$\frac{f(a+h)}{h}=\frac{f(a)}{h}+f'(a)+\frac{R_1(x)}{h}$$

Solve for $f'(a)$

$$f'(a)=\frac{f(a+h)-f(a)}{h}-\frac{R_1(x)}{h}$$

Assuming $R_1(x)$ is sufficiently small, the approximation of the first derivative of "f" is given as below

$$f'(a)=\frac{f(a+h)-f(a)}{h}$$

HEAT EQUATION

Heat Equation is an example of parabolic Equation. For two dimensional Omega= (0; 1)2, it takes the form

$$\frac{\partial y}{\partial x}(t,x)=g(t,x)+\Delta_z u(t,x)\ \ for\ all\ t\ \in\ \mathbb{R}\ \geq 0, x\ \in \Omega$$

Where delta x is Laplace Operator and is applied on the variable x. To ensure uniqueness of the solution, we have to add boundary conditions.

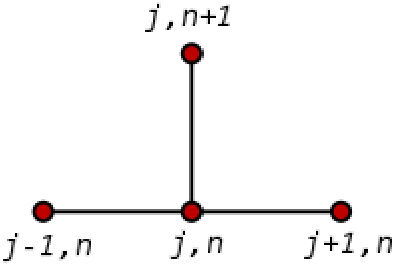
$$u(t,x)=0\quad for\ all\ t\ \in\ \mathbb{R} \geq 0, x\ \in\ \partial\Omega$$

The value u(t, x) is temperature at time belongs to R in that point where x belongs to Omega. The function g describes where and when heat is created
If g is constant with respect to time such that

$$g(t,x)=g_\infty\quad for\ all\ t\ \in\ \mathbb{R},$$

We can prove that the solution ***u*** will converge to a function

(BERLAND, 2008)



$$g_\infty : for\ all\ \Omega\ \rightarrow \mathbb{R}$$

That is the solution of Poisson equation

$$-\Delta u_\infty(x)=g_\infty(x)\quad for\ all\ \ x\ \in \Omega$$

$$u_\infty(x)=0\quad for\ all\ \ x\ \in \partial\Omega$$

This limit is known as Equilibrium solution, and can be calculated by the techniques we have explained.
To handle the Time dependent solution, u and g are interpreted as functions in time mapping to functions in space. Let

$$\hat{u}(t)(x) \coloneqq u(t,x), \qquad \hat{g}(t)(x) \coloneqq g(t,x)$$

$$for\ all\ t\ \in\ \mathbb{R}\ \geq 0, x\ \in \Omega$$

By including the space

$$C^\infty_{\partial\Omega}(\Omega\ ) \coloneqq \{\ u\ \in C(\Omega\ ): u|\Omega\ \ \in C^\infty(\Omega\ ), u|\partial\Omega\ = 0$$

Then enlarging and expanding the Laplace Operator

$$\Delta u(x) \coloneqq \{^{\frac{\partial^2 u}{\partial x_1^2}(x)+\frac{\partial^2 u}{\partial x_2^2}(x)\ if\ x\in\Omega}_{0\qquad\qquad\qquad otherwise} for\ all\ u \in C^\infty_{\partial\Omega}(\Omega\ ), x \in\ \Omega$$

Equation (3.1) can be written as the ordinary differential Equation

$$\hat{u}(0) = \hat{u}_0 \qquad \hat{u}(t) = \hat{g}(t) + \Delta\,\hat{u}(t) \quad for\, all\, t\, \in\, \mathbb{R} \geq 0$$

EXPLICIT METHOD

Using a forward difference at time t_n and a second-order central difference x_j for the first order space derivative at position we get the **recurrence equation**:

$$\frac{u_j^{n+1} - u_j^n}{k} = \frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{h^2}$$

The Forward Euler method can be derived from the above equation in one dimension.
We can obtain u_j^{n+1} from the other values this way:

$$u_j^{n+1} = (1 - 2r)u_j^n + ru_{j-1}^n + \; ru_{j+1}^n$$

Where $r = \frac{k}{h^2}$

So, with this recurrence equation, and knowing the other values at time n, we can obtain the corresponding values at time n+1. u_0^n and u_j^n are replaced by the boundary conditions, in this example they are both 0.

FIGURE 1
THE STENCIL FOR THE MOST COMMON EXPLICIT METHOD FOR THE HEAT EQUATION

EULER IMPLICIT METHOD

Consider the ordinary differential equation

$$\frac{dy}{dx} = f(t, y)$$

with initial value $y(t_0) = y_0$. Here the function f and the initial data t_0 and y_n are known; the function $y$depends on the real variable t and is unknown. A numerical method produces a sequence $y_0, y_1, y_2,$ ...such that $y_k$approximates $y(t_0 + k)$, where h h is called the step size.
The backward Euler method computes the approximations using

$$y_{k+1} = \; y_k + hf(t_{k+1}, y_{k+1})$$

This differs from the (forward) Euler method in that the latter uses

$$f(t_{k+1}, y_{k+1})$$

Instead of

$$f(t_k, y_k)$$

The backward Euler method is an implicit method: the new approximation y_{k+1} appears on both sides of the equation, and thus the method needs to solve an algebraic equation for the unknown y_{k+1}. For non-stiff problems, this can be done with fixed-point iteration:

$$y_{k+1}^{[0]} = y_k, \quad y_{k+1}^{[i+1]} = y_k + hf(t_{k+1}, y_{k+1}^{[i]})$$

If this sequence converges (within a given tolerance), then the method takes its limit as the new approximation y_{k+1}. Alternatively, one can use (some modification of) the Newton–Raphson method to solve the algebraic equation.

(BERLAND, 2008)

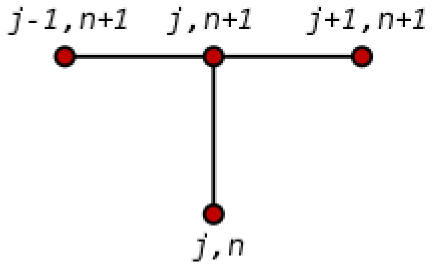


FIGURE 2

THE IMPLICIT METHOD STENCIL.

DERIVATION

If we use the backward difference at time $\boldsymbol{t_{n+1}}$ and a second-order central difference for the space derivative at position $\boldsymbol{x_j}$.

$$\frac{u_j^{n+1} - u_j^n}{k} = \frac{u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}}{h^2}$$

Above equation is an implicit method for solving one dimensional heat equations:

$$(1 + 2r)u_j^{n+1} - ru_{j-1}^{n+1} - ru_{j+1}^{n+1} = u_j^n$$

The errors are linear over the time step and quadratic over the space step:

$$\Delta\boldsymbol{u} = \boldsymbol{O}(\boldsymbol{k^2}) + \boldsymbol{O}(\boldsymbol{h^2})$$

ANALYSIS

The backward Euler method has order one. This means that the local truncation error (defined as the error made in one step) is $\boldsymbol{O}(\boldsymbol{h^2})$, using the big O notation. The error at a specific time t is $\boldsymbol{O}(\boldsymbol{h})$.

The region of absolute stability for the backward Euler method is the complement in the complex plane of the disk with radius 1 centered at 1, depicted in the figure. This includes the whole left half of the complex plane, making it suitable for the solution of stiff equations In fact, the backward Euler method is even L-stable.

CRANK–NICOLSON METHOD

In numerical analysis, Crank-Nicolson method is used for solving heat equation and similar partial differential equations. It is implicit in time and can be written as an implicit Runge–Kutta method, and it is numerically stable (both with respect to time and space step). The method was developed by John Crank and Phyllis Nicolson in the mid 20th century.

The Crank–Nicolson method is based on the trapezoidal rule, giving second-order convergence in time. For example, for the given partial differential equation.

$$\boldsymbol{\frac{\partial u}{\partial t} = F\left(u, x, t, \frac{\partial u}{\partial x}, \frac{\partial^2 u}{\partial x^2}\right)}$$

The equation for Crank–Nicolson method is a combination of the forward Euler method at $\boldsymbol{n}$ and the backward Euler method at n + 1

$\frac{u_i^{n+1}-u_i^n}{\Delta t} = F_j^n\left(u, x, t, \frac{\partial u}{\partial x}, \frac{\partial^2 u}{\partial x^2}\right)$ (forward Eular)

$\frac{u_i^{n+1}-u_i^n}{\Delta t} = F_i^{n+1}\left(u, x, t, \frac{\partial u}{\partial x}, \frac{\partial^2 u}{\partial x^2}\right)$ (backward Eular)

$$\frac{u_i^{n+1}-u_i^n}{\Delta t} = \frac{1}{2}\left[F_i^{n+1}\left(u, x, t, \frac{\partial u}{\partial x}, \frac{\partial^2 u}{\partial x^2}\right) + F_i^n\left(u, x, t, \frac{\partial u}{\partial x}, \frac{\partial^2 u}{\partial x^2}\right)\right]$$

To get the "next" approximation of u at time, a system of algebraic equations must be solved. The discretization will also be nonlinear if the partial differential equation is nonlinear, so that advancing in time will involve the solution of a system of nonlinear algebraic equations, though linearizations are possible. In many problems, especially linear diffusion, the algebraic problem is tridiagonal and may be efficiently solved with the tridiagonal matrix algorithm, which gives a fast $\boldsymbol{O(n)}$ direct solution as opposed to the usual $\boldsymbol{O(n)^3}$ for a full matrix.

FIGURE 1

THE CRANK–NICOLSON STENCIL.

EXAMPLE: 1D DIFFUSION

The Crank–Nicolson method is applied to single dimensional diffusion problems.

$$\boldsymbol{\frac{\partial u}{\partial t} = a\frac{\partial^2 u}{\partial x^2}}$$

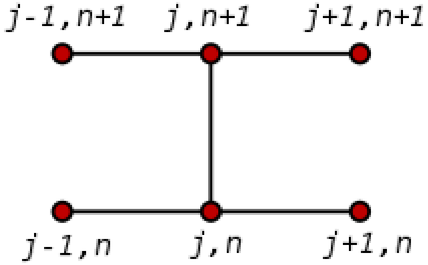
applying a finite difference spatial discretization for the right hand side, the Crank–Nicolson discretization is then :

$$\frac{u_i^{n+1}-u_i^n}{\Delta t} = \frac{a}{2(\Delta x)^2}((u_{i+1}^{n+1} - 2u_i^{n+1} + u_{i-1}^{n+1}) + (u_{i+1}^n - 2u_i^n + u_{i-1}^n))$$

or, letting

$$r = \frac{a}{2(\Delta x)^2}$$

(Berland, 2008)



$$-ru_{i+1}^{n+1} + (1+2r)u_i^{n+1} - ru_{i-1}^{n+1} = ru_{i+1}^n + (1-2r) - ru_i^n + ru_{i-1}^n$$

given that the terms on the right-hand side of the equation are known, this is a tridiagonal problem, so that $\boldsymbol{u_i^{n+1}}$ may be efficiently solved by using the tridiagonal matrix algorithm in favor of a much more costly matrix inversion.

Minimalistic and least general example is given below:

$$\frac{\partial u}{\partial t} = a(u)\frac{\partial^2 u}{\partial x^2}$$

would lead to a nonlinear system of algebraic equations which could not be easily solved as above; however, it is possible in some cases to linearize the problem by using the old value for a, that is $\boldsymbol{a_i^n(u)}$. Other times, it may be possible to estimate $\boldsymbol{a_i^{n+1}(u)}$, using an explicit method and maintain stability.

EXAMPLE: 2D DIFFUSION

When extending into two dimensions on a uniform Cartesian grid, the derivation is similar and the results may lead to a system of band-diagonal equations rather than tridiagonal ones. The two-dimensional heat equation

$$\frac{\partial u}{\partial t} = a\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right)$$

can be solved with the Crank–Nicolson discretization of

$$u_{i,j}^{n+1} =$$

$$u_{i,j}^{n} + \frac{1}{2}\,\frac{a\Delta \mathrm{t}}{(\Delta \mathrm{x})^{2}}\Big[\big(u_{i+1,j}^{n+1} + u_{i-1,j}^{n+1} + u_{i,j+1}^{n+1} + u_{i,j-1}^{n+1} - 4u_{i,j}^{n+1}\big) + \big(u_{i+1,j}^{n} + u_{i-1,j}^{n} + u_{i,j+1}^{n} + u_{i,j-1}^{n} - 4u_{i,j}^{n}\big)\Big]$$

assuming that a square grid is used so that $\Delta \boldsymbol{x} = \Delta \boldsymbol{y}$. This equation can be simplified somewhat by rearranging terms and using the CFL number

$$\mu = \frac{a\Delta \mathrm{t}}{(\Delta \mathrm{x})^{2}}$$

For the Crank–Nicolson numerical scheme, a low CFL number is not required for stability, however it is required for numerical accuracy. We can now write the scheme as:

$$(1 + 2\mu)u_{i,j}^{n} + \tfrac{\mu}{2}\big(u_{i+1,j}^{n+1} + u_{i-1,j}^{n+1} + u_{i,j+1}^{n+1} + u_{i,j-1}^{n+1}\big) = (1 - 2\mu)u_{i,j}^{n} + \tfrac{\mu}{2}\big(u_{i+1,j}^{n} + u_{i-1,j}^{n} + u_{i,j+1}^{n} + u_{i,j-1}^{n}\big)$$

Solving such a linear system is not practical due to extremely high time complexity of solving a linear system by the means of Gaussian elimination or even Strassen algorithm. Hence an Alternating direction implicit method can be implemented to solve the numerical PDE whereby one dimension is treated implicitly and other dimension explicitly for half of the assigned time-step and vice versa for the remainder half of the time-step. The benefit of this strategy is that the implicit solver only requires a Tridiagonal matrix algorithm to be solved. The difference between the true Crank–Nicolson solution and ADI approximated solution has an order of accuracy of $\boldsymbol{O}(\boldsymbol{t}^{4})$ and hence can be ignored with a sufficiently small time-step

Finally if we use the central difference at time $\boldsymbol{t}_{\boldsymbol{n+1/2}}$ and a second-order central difference for the space derivative at position $\boldsymbol{x}_{\boldsymbol{j}}$ ("CTCS") we get the recurrence equation:

$$\frac{u_j^{n+1} - u_j^{n}}{k} = \left(\frac{u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}}{h^2} + \frac{u_{j+1}^{n} - 2u_j^{n} + u_{j-1}^{n}}{h^2}\right)$$

Above equation is known as Crank–Nicolson method

We can obtain u_j^{n+1} from the other values this way:

$$\boldsymbol{(2 + 2r)u_j^{n+1} - ru_{j-1}^{n+1} - ru_{j+1}^{n+1} = (2 - 2r)u_j^{n} + ru_{j-1}^{n} - ru_{j+1}^{n}}$$

The scheme of solution is more stable and accurate but require solving more equations on each step so numerically more intensive. Errors are Quadratic over time and space step:

$$\Delta \boldsymbol{u} = \boldsymbol{O}(\boldsymbol{k}^{2}) + \boldsymbol{O}(\boldsymbol{h}^{2})$$

Scope of Finite Difference Method

Because a number of other phenomena can be modeled with the heat equation (often called the diffusion equation in financial mathematics), the Crank–Nicolson method has been applied to those areas as well. Particularly, the Black–Scholes option pricing model's differential equation can be transformed into the heat equation, and thus numerical solutions for option pricing can be obtained with the Crank–Nicolson method.

The importance of this for finance, is that option pricing problems, when extended beyond the standard assumptions (e.g. incorporating changing dividends), cannot be solved in closed form, but can be solved using this method. Note however, that for non-smooth final conditions (which happen for most financial instruments), the Crank–Nicolson method is not satisfactory as numerical oscillations are not damped. For vanilla options, this results in oscillation in the gamma value around the strike price. Therefore, special damping initialization steps are necessary (e.g., fully implicit finite difference method)

As above, these methods can solve derivative pricing problems that have, in general, the same level of complexity as those problems solved by tree approaches,[1] but, given their relative complexity, are usually employed only when other approaches are inappropriate; an example here, being changing interest rates and / or time linked dividend policy. At the same time, like tree-based methods, this approach is limited in terms of the number of underlying variables, and for problems with multiple dimensions, Monte Carlo methods for option pricing are usually preferred. Note that, when standard assumptions are applied, the explicit technique encompasses the binomial- and trinomial tree methods. Tree based methods, then, suitably parameterized, are a special case of the explicit finite difference method.

Finite Element Method

The subdivision of a whole domain into simpler parts has several advantages:

- Accurate representation of complex geometry
- Inclusion of dissimilar material properties
- Easy representation of the total solution
- Capture of local effects.

A typical work out of the method involves (1) dividing the domain of the problem into a collection of subdomains, with each subdomain represented by a set of element equations to the original problem, followed by (2) systematically recombining all sets of element equations into a global system of equations for the final calculation.

Finite element methods are numerical methods for approximating the solutions of mathematical problems that are usually formulated so as to precisely state an idea of some aspect of physical reality.

EXAMPLE: ONE-DIMENSIONAL

We will demonstrate the finite element method using two sample problems from which the general method can be extrapolated. It is assumed that the reader is familiar with calculus and linear algebra. P1 is a one-dimensional problem

$$\boldsymbol{P1}: \left\{ \begin{matrix} \boldsymbol{u''(x) = f(x) in\ (0, 1),} \\ \boldsymbol{u(0) = u(1) = 0} \end{matrix} \right\}$$

If $\boldsymbol{u}$ solves P1, then for any smooth function v that satisfies the displacement boundary conditions, i.e. v=0 at x=0 and x=1, we have

$$\int_0^1 \boldsymbol{f(x)\, v(x) dx} = \int_0^1 \boldsymbol{u''\,(x) v(x) dx}$$

We define a new function $\emptyset(\boldsymbol{u}, \boldsymbol{v})$ by using integration by parts on the right-hand-side of (1):

$$\int_0^1 \boldsymbol{f(x)\, v(x) dx} = \int_0^1 \boldsymbol{u''\,(x) v(x) dx}$$

$$= \boldsymbol{u'(x) v(x)} - \int_0^1 \boldsymbol{u''\,(x) v'(x) dx}$$

$$= -\int_0^1 \boldsymbol{u'\,(x) v'(x) dx} = -\emptyset(\boldsymbol{u}, \boldsymbol{v})$$

where we have used the assumption that v(0)=v(1)=0.

EXAMPLE: TWO-DIMENSIONAL

If we integrate by parts using a form of Green's identities, we see that if u solves P2, then we may define $\emptyset(\boldsymbol{u}, \boldsymbol{v}$ for any v by

$$\int_\Omega \boldsymbol{f(v) ds} = -\int_\Omega \boldsymbol{\nabla u . \nabla v ds} \equiv -\emptyset(\boldsymbol{u}, \boldsymbol{v})$$

GALERKIN METHOD

In Numerical Analysis, Glerkin methods are applied to solve the differential equations in discontinuous manner. They combine finite volume and finite element framework to solve differential problems. Galerkin methods are usually of two types. (1) Continuous (2) Discontinuous. In Discontinuous manner, the problem is divided into smaller elemental equations in such a way that a triangular mesh is formed. The distance between each two boundary points of the triangle is proportional to the distance between the boundary points of the circle.

EXAMPLE

As an example, consider the continuity equation for a scalar unknown P in a spatial domain Ω without "sources" or "sinks" :

$$\boldsymbol{\frac{\partial p}{\partial x} + \nabla . J = 0}$$

where J is the flux of P .

Now consider the finite-dimensional space of discontinuous piecewise polynomial functions over the spatial domain Ω restricted to a discrete triangulation Ωh written as

$$\boldsymbol{S_h^p}(\Omega_{\boldsymbol{h}}) = \left\{ \boldsymbol{v}_{|\Omega_{ei}} \in \boldsymbol{P^p}(\Omega_{\boldsymbol{ei}}), \forall \Omega_{\boldsymbol{ei}} \in \Omega_{\boldsymbol{h}} \right\}$$

for $\boldsymbol{P^p}(\Omega_{\boldsymbol{ei}})$ the space of polynomials with degrees less than or equal to p over element $\Omega_{\boldsymbol{ei}}$ indexed by i. Then for finite element shape functions the solution is represented by

$$\boldsymbol{p_h} = \sum_{\boldsymbol{j=1}}^{\boldsymbol{dofs}} \emptyset_{\boldsymbol{j}}^{\boldsymbol{i}} \boldsymbol{N_j^i}\,(\boldsymbol{x}), \forall \boldsymbol{x} \in \Omega_{\boldsymbol{ei}}$$

multiplying the continuity equation by Ph and integrating by parts in space, the semi discrete DG formulation becomes:

$$\boldsymbol{\frac{d}{dt}} \int_{\Omega_{\boldsymbol{ei}}} \boldsymbol{Ph\emptyset hdx} + \int_{\partial\Omega_{\boldsymbol{ei}}} \emptyset_{\boldsymbol{h}} \boldsymbol{J_h . ndx} = \int_{\partial\Omega_{\boldsymbol{ei}}} \boldsymbol{J_h . \nabla \emptyset_h dx}$$

SCOPE OF FINITE ELEMENT METHOD

- ***Stratified flow*** – This model is applied to stratified flow. It is concerned with the fluids flow that depends upon gravity. Due to which stratified flows are very common in earth’s atmosphere.
- ***Flow over irregular boundaries*** – The flow that is generated by longitudinal and linear transverse vibrations in two dimensional grooves and cylindrical functions , the frequency described in

terms of sinusoidal waves. The related problems are solved using this method and utilized in practical flow of fluids.

- ***Wheels*** – Wheels are common and essential parts of many machines. For good design and machine requirements, the tires are supposed to be designed to bear more weight and are supposed to require less material for build. Their geometry and shape is designed and calculated using this method easily by utilizing computational machines.

- ***Radial load fatigue analysis*** **–** Wheels are usually connected with the devices through rod or shafts of some kind. Fatigue is weakening of the material due to the passage of time and loading more material regularly. The method is utilized with stress-strain law. Many materials break without reaching the yield point. Many machines are designed to calculate stress-strain graph of a material are then plot and analyzed.

- ***Bending fatigue analysis*** – A ductile material such as metal is when applied to stress, the stress produced in the material is proportional to the applied stress but with the passage of time, the ductility of the material is reduced at relatively lower rate but is decreased continuously.

    1. ***Bolt preload*** – To find the bolt hole parameters, we analyses the mesh structure built by computer. A load is applied on a pre-defined structure that is a mesh like three dimensional shapes. We then find the most accurate mesh structure of the bolt according to the mesh data of the material.

    2. ***Rotational centrifugal force*** – Wheels are designed in a way such that the outer boundary and inner boundary is thinner than the thickness of material in between the boundaries. When wheel rotates, the overall thickness is then stabilized.

- ***Body and frame*** - Frames is usually a outer skeleton of a machine such as vehicle and it has an important result for vehicles. It requires stiffness as well as elasticity such that when the car is collided, the outer body should bend so that the staff is not injured. It should qualify the conditions that material fatigue. Structural and dynamic properties are analyzed through FEM method**.**

COMPARISON OF FINITE DIFFERENCE METHOD

PROPERTY	**FINITE DIFFERENCE METHOD**	**FINITE ELEMENT METHOD**
TIME STEPS	Incremental time steps need to be smaller than critical value of stability	Time steps can be large with unconditionally stable schemes
CALCULATIONS	Small amount of calculations in each step	Large amount of calculations in each step
NON-LINEAR CONSTITUTIVE LAW	No iterations necessary to follow nonlinear constitutive law	To follow a nonlinear constitutive law an iterative procedure is required
VALIDITY OF NON-LINEAR LAWS	Following of nonlinear laws in a valid physical way is guaranteed	The following of physically correct path should always be assured
MEMORY REQUIREMENT	No large memory requirement so no large matrices are formed	Large memory requirement for storage of stiffness matrices
ACCURACY	Small amount of calculations but relatively lower accuracy	Requires more calculations and memory but highly accurate
DISCRETIZATION CRITERIA	Discretization is based upon the differential form of the PDE to be solved. Each derivative is replaced with an approximate difference formula	Discretization is based upon a piecewise representation of the solution in terms of specified basis functions.
APPROACH TOWARDS SOLUTION	FEM solves the physical problem by dividing the geometry into small elements and calculates the stresses and strains in those	FDM divides the problem into small time steps and predicts the stresses and strains of the next time step based on the present time step using finite difference

	elements using theory of superposition	formulation such as forward and central differences

Bibliography

Berland. (2008, July 3). *Finite Difference Method*. Retrieved May 13, 2018, from Wikipedia.com: https://en.wikipedia.org/wiki/Finite_difference_method#/media/File:Explicit_method-stencil.svg