

1. **(100 Points)** Explicit versus implicit integration: Consider the following 1st order ordinary differential equation (ODE):

$$\frac{\partial y}{\partial t} = f(y, t) \quad (1)$$

In order to solve this 1st order equation we need one initial condition of the form:

$$y(t = 0) = y_0 \quad (2)$$

This ODE can be solved by using a finite difference (FD) approximation for  $\frac{\partial y}{\partial t}$ . For discrete time values  $t_n = n\Delta t$  we denote  $y(t_n)$  by  $y_n$ . The ODE for  $t_n$  can be written as:

$$y'_n \approx \frac{y_{n+1} - y_n}{\Delta t} = f(y_n, t_n) \Rightarrow \boxed{y_{n+1} = \Delta t f(y_n, t) + y_n} \quad (3)$$

This is an *explicit* scheme as all the equations are written for  $t_n$  where all quantities are known ( $y_n$ ) and the only unknown appears through approximation of  $y'_n$ . Note that whether  $f(y, t)$  is a linear function of  $y$  or not, the equation to obtain  $y_{n+1}$  is linear. This particular FD approximation for 1st derivative is called *forward Euler* as for the derivative of the point  $y_n$  we use its forward value.

If on the other hand we write the equation for  $t_{n+1}$  and use a *backward Euler* approximation for derivative ( $y'_{n+1}$  uses backward  $y_n$  value) we obtain:

$$y'_{n+1} \approx \frac{y_{n+1} - y_n}{\Delta t} = f(y_{n+1}, t_{n+1}) \Rightarrow \boxed{y_{n+1} - \Delta t f(y_{n+1}, t) = y_n} \quad (4)$$

We note that this is an *implicit* equation for  $y_{n+1}$  in general. This equation also becomes nonlinear when  $f(y, n)$  is nonlinear in  $y$ . In general a nonlinear solver is needed for each single step. This is called an *implicit* scheme.

The solution process for (3) or is straightforward; we start with  $y_0$  provided by initial condition and step by step advance  $n$ .

- For  $f(y, t) = y^2$  obtain  $y_{n+1}$  in terms of  $y_n$  for both explicit and implicit methods. Comment on their differences.
  - For  $f(y, t) = -\lambda y$  obtain  $r = y_{n+1}/y_n$  in terms of  $\Delta t$  and  $\lambda$ . The exact solution to this ODE is  $y(t) = y_0 e^{-\lambda t}$  that is a nonincreasing ODE for  $\lambda \geq 0$ . Based on  $|r|$  determine for what ranges of  $\Delta t$  each method becomes *unstable*, that is  $|y|$  start growing time step after time step (note that for this ODE  $y$  should decrease, so increase of  $y$  by numerical integration is only because of numerical error).
  - Based on your observations, compare explicit and implicit methods in terms of their stability and limit on  $\Delta t$  and nonlinearity of their integration schemes.
2. **(300 Points)** Dynamic FEM solution: We use weighted residual to form finite element formulations. For each of  $m$  (number of unknowns) weight functions we obtain one equation in terms of  $a^1, \dots, a^m$  where  $\hat{u} = \Sigma a^j(t) N^j(\mathbf{x})$ . If the partial differential equations (PDEs) and boundary conditions (BCs) that we use are nonlinear the resulting finite element formulation would be nonlinear as well. For dynamic problems the highest order time derivative term is generally linear (e.g.,  $\rho \ddot{\mathbf{u}}$  elastodynamics,  $\rho \dot{v}$  fluid equation,  $CT$  thermal equation, etc.). For example, for a first order dynamic problem such as heat equation the initial boundary value problem can be stated

as,

$$\mathbf{C}\dot{\mathbf{a}} + \mathbf{f}(\mathbf{a}) = \mathbf{0} \quad \text{PDE} \quad (5a)$$

$$\mathbf{u}(x, t = 0) = u_0(\mathbf{x}) \Rightarrow \mathbf{u}(x, t = 0) = \Sigma a^j(t = 0)N^j(x) = u_0(\mathbf{x}) \Rightarrow$$

$$\text{we obtain } a_0^j = a^j(t = 0) \text{ e.g., } a_0^j = u_0(x^j) \text{ for CFEM} \quad \text{IC} \quad (5b)$$

$$u(\mathbf{x}, t) = \bar{u}(\mathbf{x}, t) \text{ for } \mathbf{x} \in \partial\mathcal{D}_u \text{ (Essential BC),}$$

$$\nabla \mathbf{u}(\mathbf{x}, t) = \bar{q}(\mathbf{x}, t) \text{ for } \mathbf{x} \in \partial\mathcal{D}_f \text{ (Natural BC),} \quad \text{BC} \quad (5c)$$

BCs are written for a typical 2nd order equation in terms of spatial derivative (*e.g.*, thermal equation, elastodynamics, *etc.*) as it's commonly the case. The Natural boundary condition can be a more general function of  $\nabla \mathbf{u}$  (*e.g.*, stress in elastodynamics).

The explicit and implicit time discretization of (5a) are,

$$\mathbf{C} \left\{ \frac{\mathbf{a}_{n+1} - \mathbf{a}_n}{\Delta t} \right\} + \mathbf{f}(\mathbf{a}_n) = 0 \Rightarrow \mathbf{C}\mathbf{a}_{n+1} = \mathbf{C}\mathbf{a}_n - \Delta t \mathbf{f}(\mathbf{a}_n) \quad \text{Explicit} \quad (6a)$$

$$\mathbf{C} \left\{ \frac{\mathbf{a}_{n+1} - \mathbf{a}_n}{\Delta t} \right\} + \mathbf{f}(\mathbf{a}_{n+1}) = 0 \Rightarrow \mathbf{C}\mathbf{a}_{n+1} + \Delta t \mathbf{f}(\mathbf{a}_{n+1}) = \mathbf{C}\mathbf{a}_n \quad \text{Implicit} \quad (6b)$$

For the problems that  $\mathbf{f}(\mathbf{a})$  is linear, we can express it as  $\mathbf{f}(\mathbf{a}) = \mathbf{K}\mathbf{a} - \mathbf{F}$ . Matrices  $\mathbf{C}$  and  $\mathbf{K}$  are often called “mass” and “stiffness” matrices motivated by elastodynamics (even though in elastodynamics mass matrix multiplies  $\ddot{\mathbf{u}}$ . For the thermal problem  $\mathbf{K}$  is called conductivity matrix. Anyhow, for linear systems (6) can be simplified as,

$$\mathbf{C}\mathbf{a}_{n+1} = \mathbf{C}\mathbf{a}_n - \Delta t (\mathbf{K}\mathbf{a}_n - \mathbf{F}) \quad \text{Explicit} \quad (7a)$$

$$\{\mathbf{C} + \Delta t \mathbf{K}\} \mathbf{a}_{n+1} = \mathbf{C}\mathbf{a}_n + \Delta t \mathbf{F} \quad \text{Implicit} \quad (7b)$$

The solution for (6) or (7) starts from initial conditions  $\mathbf{a}_0$  and advances in time one time step at a time similar to the scalar equations (3) and (1). For each of the following questions do not provide more than 1-3 sentences in response.

- For a general nonlinear  $\mathbf{f}(\mathbf{a})$  which of time marching equations in (6) is nonlinear (that is a nonlinear equation is required to solve  $\mathbf{a}_{n+1}$  in terms of  $\mathbf{a}_n$ )?
- Clearly for linear case (7) both implicit and explicit methods only involve a simple linear update equation for  $\mathbf{a}_{n+1}$  in terms of  $\mathbf{a}_n$ . Which of the two implicit/explicit methods always requires a linear update regardless of the linearity of  $\mathbf{f}(\mathbf{a})$ ?
- For linear case (7) we can conceptually write the equation as  $\mathcal{K}\mathbf{a} = \mathcal{F}$ . Compare the choices of  $\mathcal{K}$  for implicit and explicit methods.
- Since  $\mathbf{K}$  is typically a fully coupled matrix implicit methods result in nontrivial matrix equation even for the linear equations (7b) regardless of the form of  $\mathbf{C}$ . Compare how the choice of DG and CFEM influences  $\mathcal{K}$  (*e.g.*, LHS) and the solution complexity of (7a).
- For CFEM as mentioned in class  $\mathbf{C}$  couples all dofs making the solution to (7a) nontrivial. An approach generally adopted in CFEM is using a “mass lumped”  $\mathbf{C}$  instead of the actual “consistent”  $\mathbf{C}$  matrix. What is a “mass lumped” mass matrix and how does it simplify the equations?
- In the class we formulated and solved the 1D heat equation for the geometry shown in figure

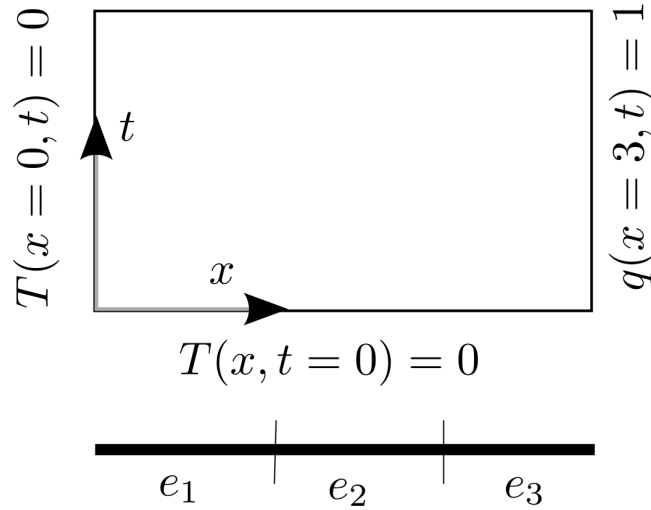


Figure 1: A 1D initial value problem for heat conduction: Problem set up and a three element discretization.

1 but for the static case with  $Q = 0, \kappa = 1, c = 1, A = 1$ . For CFEM by noting that,

$$\mathbf{c}^e = \int_e \mathbf{N}^T c \mathbf{N} \, dv \quad \Rightarrow \quad \mathbf{C} = \frac{1}{6} \begin{bmatrix} 4 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 2 \end{bmatrix} \quad (8a)$$

$$\mathbf{k}^e = \int_e \nabla \mathbf{N}^T \kappa \nabla \mathbf{N} \, dv \quad \Rightarrow \quad \mathbf{K} = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 1 \end{bmatrix} \quad (8b)$$

$$\mathbf{f}^e = \int_e \mathbf{N}^T Q \, dv - \int_{e_f} \mathbf{N}^T q \cdot \mathbf{n} \, ds - \mathbf{f}_d^e \text{ (essential BC force)} \quad \Rightarrow \quad \mathbf{F} = \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix} \quad (8c)$$

and the  $3 \times 3$  system of equations was

$$\mathbf{C} \dot{\mathbf{a}} + \mathbf{K} \mathbf{a} - \mathbf{F} = 0 \quad (9)$$

For DG method the weighted residual for one element is,

$$\int_e \hat{T} (\dot{T} + \nabla \cdot \mathbf{q} - Q) \, dv + \int_{\partial e} \hat{T} (\mathbf{q}^* - \mathbf{q} \cdot \mathbf{n}) \, ds + \int_{\partial e} \hat{\mathbf{q}} \cdot \mathbf{n} (T^* - T) \, ds = 0 \quad (10)$$

where  $\hat{\mathbf{q}} = -\kappa \nabla \hat{T}$ . To derive the final DG  $\mathbf{C}$  and  $\mathbf{K}$  follow these steps:

- i. For a 1D DG element with shape functions  $N_1 = 1$  and  $N_2 = x$  and element unknowns  $T_h = b^1(t)N_1 + b^2(t)N_2$  show that (10) reduces to

$$\begin{bmatrix} 1 & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{3} \end{bmatrix} \begin{bmatrix} \dot{b}^1(t) \\ \dot{b}^2(t) \end{bmatrix} + \begin{bmatrix} q_1^* + q_2^* \\ q_2^* - T_2^* + T_1^* + 2b^2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad (11)$$

where  $T_1^*, T_2^*$  are target temperatures on the left and right sides of the element and the same applies to the outward heat fluxes  $\mathbf{q}_1^*$  and  $\mathbf{q}_2^*$ .

- ii. Using the definitions of target values given in the class for natural and essential boundaries and inside the domain show that for six unknowns  $a^1$  to  $a^6$   $\mathbf{C}$ ,  $\mathbf{K}$  and  $\mathbf{F}$  are ( $a^1, a^2$  for  $e_1$ ,  $a^3, a^4$  for  $e_2$  and  $a^5, a^6$  for  $e_3$ )

$$\mathbf{C} = \begin{bmatrix} 1 & \frac{1}{2} & 0 & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{3} & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{3} & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \frac{1}{2} \\ 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{3} \end{bmatrix}, \quad \mathbf{K} = \frac{1}{2} \begin{bmatrix} 0 & 1 & 0 & -1 & 0 & 0 \\ -1 & 2 & -1 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -1 \\ 1 & 1 & 0 & 2 & -1 & -1 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & -1 & 2 \end{bmatrix}, \quad \mathbf{F} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ -1 \\ -1 \end{bmatrix} \quad (12)$$

which again the solution satisfied (9).

- iii. Show that for both DG and CFEM the static solution recovers the exact solution  $T(x) = -x$ . Comment on why each method is able to recover the exact static solution.

- (g) Comparison of Explicit vs Implicit integration and DG vs CFEM: Download three Matlab files [www.rezaabedi.com/wp-content/uploads/CourseSDG/Matlab/heat1DDG.m](http://www.rezaabedi.com/wp-content/uploads/CourseSDG/Matlab/heat1DDG.m), [www.rezaabedi.com/wp-content/uploads/CourseSDG/Matlab/heat1DCFEM.m](http://www.rezaabedi.com/wp-content/uploads/CourseSDG/Matlab/heat1DCFEM.m), and [www.rezaabedi.com/wp-content/uploads/CourseSDG/Matlab/heat1D.m](http://www.rezaabedi.com/wp-content/uploads/CourseSDG/Matlab/heat1D.m).

By running `heat1D(deltaT)` where `deltaT` is the time step we compute solution of DG and CFEM for implicit and explicit schemes. Plots of the final solution of each scheme and all time series plots of each simulation will be saved in the “plots” directory. By running `heat1D` for `deltaT = 0.01, 0.1, and 1.0` briefly answer the following,

- i. Comment on the continuity of the solution at element nodes for the DG method.
- ii. Comment on the stability of explicit time integration method for DG and CFEM methods.
- iii. Comment on overall convergence of DG and CFEM methods to the asymptotic solution  $T(x, t \rightarrow \infty) = -x$ .

3. **(50 Points)** Fill out table 3 for the average number of dof per element for DG and CFEM methods for 1D-3D cube and simplex geometries for element polynomial orders 1 to 3 and comment on the ratio of dofs of DG method to dofs of CFEM method based on dimension, polynomial order, and shape (cube vs simplex). To compute the average dof per element consider an infinite domain (*i.e.*, no boundary effects in counting dofs).

4. **(50 Points)** Fill out table 4 with the number of elements coupled to an element in CFEM and DG methods.

shape	dime	topology	poly order	$n_d = \#dof/e$ DG	$n_c = \#dof/e$ CFEM	$n_d/n_c$
Line	1D	Cube/Simplex	1			
			2			
			3			
Square	2D	Cube	1			
			2			
			3			
Cube	3D	Cube	1			
			2			
			3			
Triangle	2D	Simplex	1			
			2			
			3			
Tetrahedron	3D	Simplex	1			
			2			
			3			

Table 1: Comparison of average dof per element for DG and CFEM methods

shape	dime	topology	$n_{ec} = \#e$ connected/e CFEM	$n_{ed} = \#e$ connected/e DG	$n_{ec}/n_{ed}$
Line	1D	Cube/Simplex			
Square	2D	Cube			
Cube	3D	Cube			
Triangle	2D	Simplex			
Tetrahedron	3D	Simplex			

Table 2: Average number of elements coupled to an element in CFEM and DG methods.