Steady-State Natural Convection

$$\frac{\partial u}{\partial x}+\frac{\partial w}{\partial z}=0$$

$$u\frac{\partial u}{\partial x}+w\frac{\partial u}{\partial z}=-\frac{\partial p}{\partial x}+\Pr\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial z^2}\right)$$

$$u\frac{\partial w}{\partial x}+w\frac{\partial w}{\partial z}=-\frac{\partial p}{\partial z}+\Pr\left(\frac{\partial^2 w}{\partial x^2}+\frac{\partial^2 w}{\partial z^2}\right)+\Pr RaT$$

$$u\frac{\partial T}{\partial x}+w\frac{\partial T}{\partial z}=\frac{\partial^2 T}{\partial x^2}+\frac{\partial^2 T}{\partial z^2}$$

Boussinesq approximation

$$\rho=\rho_0[1+\beta(T-T_0)]$$

1

Non-Linear Operators

$$\mathbf{D}_1(u,w)\equiv\frac{\partial u}{\partial x}+\frac{\partial w}{\partial z}=0$$

$$\mathbf{D}_2(u,w,p)\equiv u\frac{\partial u}{\partial x}+w\frac{\partial u}{\partial z}=-\frac{\partial p}{\partial x}+\Pr\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial z^2}\right)$$

$$\mathbf{D}_3(u,w,p)\equiv u\frac{\partial w}{\partial x}+w\frac{\partial w}{\partial z}=-\frac{\partial p}{\partial z}+\Pr\left(\frac{\partial^2 w}{\partial x^2}+\frac{\partial^2 w}{\partial z^2}\right)+\Pr RaT$$

Linearized operators

$$\left[L_T\left(u^k,w^k\right)\right][T]=\left[u^k\frac{\partial}{\partial x}+w^k\frac{\partial}{\partial z}-\frac{\partial^2}{\partial x^2}-\frac{\partial^2}{\partial z^2}\right]T=0$$

2

$$\left[L_1\left(u^k,\mathrm{v}^k\right)\right]\begin{bmatrix}\Delta u^k\\ \Delta \mathrm{v}^k\end{bmatrix}\equiv\left[\frac{\partial}{\partial x}\right]\Delta u+\left[\frac{\partial}{\partial y}\right]\Delta \mathrm{v}=0$$

$$\left[L_2\left(u^k,\mathrm{v}^k,p^k\right)\right]\begin{bmatrix}\Delta u^k\\ \Delta \mathrm{v}^k\\ \Delta p^k\end{bmatrix}\equiv\left[u^k\frac{\partial}{\partial x}+\frac{\partial u^k}{\partial x}+w^k\frac{\partial}{\partial z}-\Pr\left(\frac{\partial^2}{\partial x^2}+\frac{\partial^2}{\partial y^2}\right)\right]\Delta u$$
$$+\left(\frac{\partial u^k}{\partial y}\right)\Delta w+\left[\frac{\partial}{\partial \mathrm{x}}\right]\Delta p=0$$

$$\left[L_3\left(u^k,\mathrm{v}^k,p^k\right)\right]\begin{bmatrix}\Delta u^k\\ \Delta \mathrm{v}^k\\ \Delta p^k\end{bmatrix}\equiv\left[u^k\frac{\partial}{\partial x}+w^k\frac{\partial}{\partial z}+\frac{\partial z^k}{\partial z}-\Pr\left(\frac{\partial^2}{\partial x^2}-\frac{\partial^2}{\partial z^2}\right)\right]\Delta \mathrm{v}$$
$$+\left(\frac{\partial w^k}{\partial x}\right)\Delta u+\left[\frac{\partial}{\partial y}\right]\Delta p$$

3

Algorithm

1) The dependent variables are know at step k, u^k, w^k, p^k, T^k

2) Perform a Newton-Raphson iteration $\mathbf{L}^k\mathbf{a}=-\mathbf{D}^k$ on the fluid flow equations keeping T^k fixed, to find u^{k+1}, w^{k+1} and p^{k+1}

3) Solve for T^{k+1} from $\left[L_T\left(u^{k+1},w^{k+1}\right)\right]\mathbf{T}^{\mathbf{k+1}}=0$ using the values of u^{k+1} and w^{k+1} from the previous step.

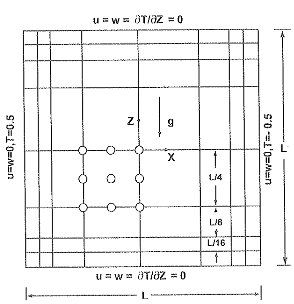
4) Check for convergence in all variables. If the convergence criterion is satisfied STOP. If not set $k=k+1$ and go back to step 2)

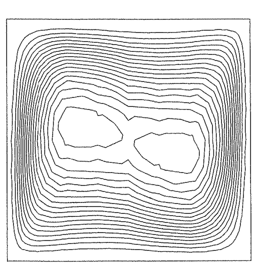
Clearly there are several ways in which this solution can be Obtained, and decisions have to be made depending on the size Of the problem. To solve all equations coupled requires 80% more Storage than this way. ***NO BEST ALGORITHM***

SOLUTION DONE IN STEPS

4

EXAMPLE





8×8 mesh of quadratic 9-node elements

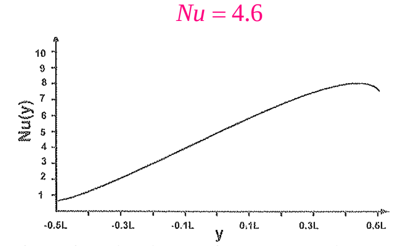
Solution for $Ra=10^5$, $\Pr=1$

5

The heat transfer accross the enclosure is calculated from

$$\overline{Nu}=\int_{-L/2}^{L/2}\frac{\partial T}{\partial x}\left(L/2,y\right)dy$$

$$\overline{Nu}=4.6$$



Local Nusselt number along vertical wall x = L/2 for Ra = 10^5 and Pr = 1.

6

TIME DEPENDENCE

Here we must distinguish between PARABOLIC equations represented by the Diffusion equation

$$\frac{\partial \phi}{\partial t} = \frac{\partial}{\partial x}\left(D\frac{\partial \phi}{\partial x}\right) + \frac{\partial}{\partial y}\left(D\frac{\partial \phi}{\partial y}\right) + S$$

and HYPERBOLIC equations such as the general Wave equation

$$m\frac{\partial^2 \phi}{\partial t^2} + \alpha\frac{\partial \phi}{\partial t} = \frac{\partial}{\partial x}\left(k\frac{\partial \phi}{\partial x}\right) + \frac{\partial}{\partial y}\left(k\frac{\partial \phi}{\partial y}\right) + F(\phi, \dot{\phi})$$

THE SEMI-DISCRETE GALERKIN METHOD

As model equation we will use the one-dimensional diffusion equation

$$\rho c_v \frac{\partial T}{\partial t} = k\frac{\partial^2 T}{\partial x^2} + Q \qquad 0 < x < L \,, t > 0$$

with appropriate boundary conditions. Now $T = T(x,t)$, so we also need an initial condition of the form $\quad T(x,0) = T_0(x)$

7

The weighted residual form is

$$\int_0^L \left(\rho c_v w\frac{\partial T}{\partial t} + k\frac{\partial w}{\partial x}\frac{\partial T}{\partial x} + wQ\right)dx + \left[w\left(-k\frac{\partial T}{\partial x}\right)\right]_0^L = 0$$

The function T(x,t) is written separating the variables as

$$T(x,t) = \sum_j N_j(x)T_j(t)$$

The Galerkin formulation at time t becomes

$$\sum_j \left[\int_0^L \rho c_v N_i N_j \, dx\right]\dot{T}_j + \sum_j \left[\int_0^L k\frac{\partial N_i}{\partial x}\frac{\partial N_j}{\partial x}dx\right]T_j = \int_0^L N_i Q dx - \left[N_i\left(-k\frac{\partial T}{\partial x}\right)\right]_0^L$$

Where $\dot{T} \equiv \dfrac{\partial T}{\partial t}$ and leads to a system of equations of the form

$$\mathbf{C}\dot{\mathbf{T}} + \mathbf{K}\mathbf{T} = \mathbf{Q}$$

8

This is called the *SEMIDESCRETE GALERKIN FORM*

The matrix $\mathbf{C} = \left[c_{ij}\right]$, $c_{ij} = \int_0^L \rho c_v N_i N_j \, dx$ is called the *CONSISTENT MASS MATRIX*

The system $\mathbf{C}\dot{\mathbf{T}} + \mathbf{K}\mathbf{T} = \mathbf{Q}$ is a system of ordinary differential equations in time. We now address their solution.

THE θ-METHOD

The θ-method is defined in two steps:

1) $\dot{\mathbf{T}} \cong \dfrac{1}{\Delta t}\left(\mathbf{T}^{n+1} - \mathbf{T}^n\right)$ where $\mathbf{T}^n = \mathbf{T}(x, t_n)$ and $t_{n+1} = t_n + \Delta t$

2) $\mathbf{T} = \theta\mathbf{T}^{n+1} + (1-\theta)\mathbf{T}^n$

Substituting into $\mathbf{C}\dot{\mathbf{T}} + \mathbf{K}\mathbf{T} = \mathbf{Q}$ we have

$$\left(\frac{1}{\Delta t}\mathbf{C} + \theta\mathbf{K}\right)\mathbf{T}^{n+1} = \left(\frac{1}{\Delta t}\mathbf{C} - (1-\theta)\mathbf{K}\right)\mathbf{T}^n + \theta\mathbf{Q}^{n+1} + (1-\theta)\mathbf{Q}^n$$

9

Assuming constant properties, for Linear elements we have

$$c_{ij} = \int_0^h N_i(x)N_j(x)\,dx \,, \begin{bmatrix} N_1 \\ N_2 \end{bmatrix} = \begin{bmatrix} 1-(x/h) \\ x/h \end{bmatrix} \quad \Rightarrow \quad \mathbf{C} = \frac{h}{6}\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

$$k_{ij} = \int_0^h \frac{dN_i}{dx}\frac{dN_j}{dx}dx \quad \Rightarrow \quad \mathbf{K} = \frac{k}{h}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$$

For constant Q, $\quad f_i = \int_0^h N_i Q dx \quad \Rightarrow \quad \mathbf{f} = \dfrac{hQ}{2}\begin{bmatrix} 1 \\ 1 \end{bmatrix}$

So we have

$$\left\{\frac{\rho c_v h}{6\Delta t}\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} + \frac{\theta k}{h}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}\right\}\begin{bmatrix} T_1^{n+1} \\ T_2^{n+1} \end{bmatrix} = \left\{\frac{\rho c_v h}{6\Delta t}\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} - \frac{(1-\theta)k}{h}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}\right\}\begin{bmatrix} T_1^n \\ T_2^n \end{bmatrix}$$
$$+ \frac{\theta hQ^{n+1}}{2}\begin{bmatrix} 1 \\ 1 \end{bmatrix} + + \frac{(1-\theta)hQ^n}{2}\begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

10

Example

$$\frac{\partial T}{\partial t} = D_T\frac{\partial^2 T}{\partial x^2} \qquad T(0,t) = 0 \,, T(1,t) = 1 \,, T(x,0) = 0$$

Using two Linear elements, D = 0.1 and θ=1, the element equations are

$$\left\{\frac{1}{12\Delta t}\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} + 0.2\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}\right\}\begin{bmatrix} T_1^{n+1} \\ T_2^{n+1} \end{bmatrix} = \left\{\frac{1}{12\Delta t}\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}\right\}\begin{bmatrix} T_1^n \\ T_2^n \end{bmatrix}$$

And after assembly

$$\left\{\frac{1}{12\Delta t}\begin{bmatrix} 2 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 2 \end{bmatrix} + 0.2\begin{bmatrix} 1 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 1 \end{bmatrix}\right\}\begin{bmatrix} T_1^{n+1} \\ T_2^{n+1} \\ T_3^{n+1} \end{bmatrix} = \frac{1}{12\Delta t}\begin{bmatrix} 2 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 2 \end{bmatrix}\begin{bmatrix} T_1^n \\ T_2^n \\ T_3^n \end{bmatrix}$$

11

After applying the boundary conditions we get

$$\frac{1}{12\Delta t}\left(4T_2^{n+1} + 1\right) + 0.2\left(2T_2^n - 1\right) = \frac{1}{12\Delta t}\left(4T_2^n + 1\right)$$

which reduces to $\quad \left(\dfrac{1}{3\Delta t} + 0.4\right)T_2^{n+1} = \dfrac{1}{3\Delta t}T_2^n + 0.2$

Notice that as $\Delta t \to \infty$, $T_2 \to \dfrac{1}{2}$, the correct steady state result.

Look at the results for the example in pages 263-264 in the text book

The values θ = 0, 0.5, and 1.0 produce the Galerkin equivalent of the Euler, Crank-Nicolson and Backward Implicit methods respectively and are known as

Euler-Galerkin
Crank-Nicolson-Galerkin
Backward-Implicit-Galerkin

12

Accuracy and Stability

1) Accuracy:

It can be shown through truncation error analysis that when $\theta = 0$ and $\theta = 1$ the methods are first order $O(\Delta t)$. When $\theta = 0.5$ the Crank-Nicolson-Galerkin method is second order $O\left((\Delta t)^2\right)$. For any other value $0 < \theta < 1$ the method converges at rates in between 0 and 1.

2) Stability

i) If $1/2 \le \theta \le 1$ the method is *UNCONDITIONALLY STABLE*.

ii) If $0 \le \theta < 1/2$ the method is *CONDITIONALLYSTABLE*. The time step limitation is given by $\Delta t < \dfrac{2}{\lambda(1-2\theta)}$ where λ is the largest eigenvalue of the generalized eigenvalue problem $(\mathbf{K} - \lambda \mathbf{C})\mathbf{X} = \mathbf{0}$

13

Lets go back to our example but keep D_T as a parameter and still use two linear elements and $\theta = 0$.

$$\frac{\partial T}{\partial t} = D_T \frac{\partial^2 T}{\partial x^2} \qquad T(0,t) = 0 \,,\, T(1,t) = 1 \,,\, T(x,0) = 0$$

The final equation becomes $\dfrac{1}{3\Delta t} T_2^{n+1} = \left(\dfrac{1}{3\Delta t} - 4D_T\right) T_2^n + 2D_T$.

That is $\mathbf{C} = \left[\dfrac{1}{3}\right]$ and $\mathbf{K} = [4D_T]$. Therefore the only eigenvalue is λ=$12D_T$, and the stability limit for Euler-Galerkin ($\theta = 0$) is $\Delta t < \dfrac{1}{6D_T}$. Rewrite the equation as $T_2^{n+1} = (1 - 12\Delta t D_T) T_2^n + 6\Delta t D_T$

Now let us try several values for Δt:

1) Set $\Delta t > \dfrac{1}{6D_T} = \dfrac{1}{3D_T}$. The recursive relation becomes $T_2^{n+1} = -3T_2^n + 2$

Then starting from $T_2^0 = 0$ we get $T_2^1 = 2, T_2^2 = -4, T_2^3 = 14, T_2^5 = -40,$ etc that diverges very fast.

14

2) Let $\Delta t = \dfrac{1}{6D_T}$ the stability limit. The recursive relation becomes $T_2^{n+1} = -T_2^n + 1$ and starting with $T_2^0 = 0$ we get $T_2^2 = 1, T_2^3 = 0, T_2^4 = 1,$ etc. which also diverges as expected.

3) Now we choose $\Delta t = \dfrac{1}{8D_T} < \dfrac{1}{6D_T}$ the algorithm becomes $T_2^{n+1} = -\dfrac{1}{2} T_2^n + \dfrac{3}{4}$ and the time history is $T_2^0 = 0, T_2^1 = 0.75,$ $T_2^2 = 0.375,\ T_2^3 = 0.5625,\ T_2^4 = 0.46875,$ etc. Clearly as $t \to \infty$ the solution converges to the correct steady state, but the time evolution is very poorly approximated and oscillates because the time step is too large

4) Choosing a smaller $\Delta t = \dfrac{1}{24D_T}$ gives $T_2^0 = 0,\ T_2^1 = 0.375,$ $T_2^2 = 0.4375, \ldots, T_2^{12} = 0.49988$ there is a great improvement in accuracy and the solution is monotonic.

Exact solution $T_2(1/2D_T) = 0.49542$

15

Solution using three Linear elements. The element equations are

$$\frac{1}{18\Delta t}\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}\begin{bmatrix} T_1^{n+1} \\ T_2^{n+1} \end{bmatrix} = \left\{\frac{1}{18\Delta t}\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} - 3D_T \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}\right\}\begin{bmatrix} T_1^n \\ T_2^n \end{bmatrix}$$

and the final assembled equations after applying boundary conditions become

$$\frac{1}{18\Delta t}\begin{bmatrix} 4 & 1 \\ 1 & 4 \end{bmatrix}\begin{bmatrix} T_2^{n+1} \\ T_3^{n+1} \end{bmatrix} = \left\{\frac{1}{18\Delta t}\begin{bmatrix} 4 & 1 \\ 1 & 4 \end{bmatrix} - 3D_T \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}\right\}\begin{bmatrix} T_2^n \\ T_3^n \end{bmatrix} + \begin{bmatrix} 0 \\ 3D_T \end{bmatrix}$$

The eigenvalue problem is

$\left| 3D_T \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix} - \dfrac{\lambda}{18}\begin{bmatrix} 4 & 1 \\ 1 & 4 \end{bmatrix} \right| = 0$ which gives $\lambda_1 = 54D_T \,,\, \lambda_2 = 10.8D_T$

Hence the stability limit is $\Delta t < \dfrac{1}{27D_T}$ Note that the smaller te mesh the smaler the critical Δt. A good estimate is given by $\Delta t < h^2/(3D_T)$

16

MASS LUMPING

One of the important cases is the Euler-Galerkin θ = 0. In matrix form this is: $\dfrac{1}{\Delta t}\mathbf{C}\mathbf{T}^{n+1} = \left(\dfrac{1}{\Delta t}\mathbf{C} - \mathbf{K}\right)\mathbf{T}^n + \mathbf{Q}$

This differs from the Euler method only by the presence of the matrix **C,** otherwise it would be a fully explicit method. Because of **C** Galerkin methods can never be fully explicit.

To obtain fully explicit formulations using the Galerkin method we introduce the concept of *MASS LUMPING*. This is of great practical importance and consists in *DIAGONALIZING* **C** in a consistent way.

There are many ways to diagonalize **C**, the only one of interest to us consists in adding the rows of **C** placing the result in the diagonal and setting all off-diagonal elements to zero. That is

$$\bar{\mathbf{C}} = \left[\bar{c}_{ij}\right] = \begin{cases} \sum_k c_{ik} & \text{if } i = j \\ 0 & \text{if } i \ne j \end{cases}$$

17

The matrix $\bar{\mathbf{C}}$ is called the *LUMPED MASS MATRIX*, replacing $\mathbf{C}$ by $\bar{\mathbf{C}}$ in the Euler-Galerkin method gives

$$\frac{1}{\Delta t}\bar{\mathbf{C}}\mathbf{T}^{n+1} = \left(\frac{1}{\Delta t}\mathbf{C} - \mathbf{K}\right)\mathbf{T}^n + \mathbf{Q}$$

Now $\bar{\mathbf{C}}$ can be easily inverted, $\bar{\mathbf{C}}^{-1}$ is diagonal and $c_{ii}^{-1} = 1/c_{ii}$. so

$$\mathbf{T}^{n+1} = \Delta t \bar{\mathbf{C}}^{-1}\left[\left(\frac{1}{\Delta t}\mathbf{C} - \mathbf{K}\right)\mathbf{T}^n + \mathbf{Q}\right]$$

Effect of Mass Lumping

In our example, using 2 Linear elements we obtain $\bar{\mathbf{C}} = [1/2]$ and $\mathbf{K} = [4D_T]$. The eigenvalue is $\lambda = 8D_T$ and the time step limit is given by $\Delta t < \dfrac{1}{4D_T}$ as opposed to $\Delta t < \dfrac{1}{6D_T}$ with the consistent Mass Matrix.

18

With 3 Linear elements the eigenvalue problem is

$$\left| 3D_T \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix} - \lambda \begin{bmatrix} 1/3 & 0 \\ 0 & 1/3 \end{bmatrix} \right| = 0 \text{ with solution } \lambda_1 = 27D_T,\ \lambda_2 = 0D_T$$

So the stability limit is $\lambda < \dfrac{2}{27D_T}$ as opposed to $\lambda < \dfrac{1}{27D_T}$ with the consistent Mass Matrix.

REMARKS:

1) Lumped matrices can also be obtained using cosed Newto-Cotes quadrature rules in which the integration points coincide with the nodes. For example, if the trapezoidat rule is used to integrate Bilinear elements we have

$N_i(x_j, y_j) = \delta_{ij}$

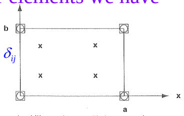


Figure 7.1 Rectangular bilinear element. Circles are nodes; crosses are Gauss integration points; squares are trapezoidal rule integration points.

$$\mathbf{C} = \bar{\mathbf{C}} = \frac{\rho c_v ab}{4} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

2) The stability limit is always larger using lumped masses, than consistent mass matrix. The algorithms are more stable.

3) Algorithms using consistent mass are in general more accurate than their lumped mass counterpart. Mass lumping can have an adverse effect in accuracy.

4) The consistent mass can introduce oscillations during the early stages of the solution, especially if the initial data is not smooth. These oscillations disappear quickly as time advances, but are unacceptable in many problems. Lumped masses do not suffer from this problem.

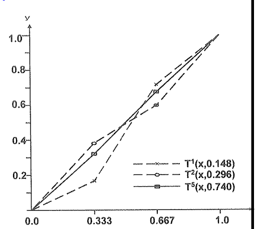
5) The oscillations in consistent formulations are generally known as "*noise*" and occur when the time step exceeds a *critical* time step related to the solution.
Unfortunately this critical value may be unreasonably small and it is not realistic to reduce the time step to that level.

Now we apply the Crank-Nicolson-Galerkin method (θ = ½) to our example problem. Since the method is unconditionally stable, we use 3 Linear elements with D_T= 1 and look at the behavior of the solution. The resulting system of equations is

$$\left\{ \begin{bmatrix} 4 & 1 \\ 1 & 4 \end{bmatrix} + 27\Delta t \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix} \right\} \begin{bmatrix} T_2^{n+1} \\ T_3^{n+1} \end{bmatrix} = \left\{ \begin{bmatrix} 4 & 1 \\ 1 & 4 \end{bmatrix} - 27D_T \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix} \right\} \begin{bmatrix} T_2^n \\ T_3^n \end{bmatrix} + \begin{bmatrix} 0 \\ 54\Delta t \end{bmatrix}$$

1) With Δt = 4/27 the solution for the first 5 time steps is

n	T_2^n	T_3^n
1	0.17778	0.71111
2	0.38716	0.60049
3	0.29665	0.70198
4	0.35486	0.64499
5	0.32036	0.67962



The initial oscillations due to the large Time step are evident as well as their quickly dying out in time

For Δt = 1/27 the oscillations do not occur, the first 5 time steps are

n	T_2^n	T_3^n
1	0.0	0.33333
2	0.11111	0.44444
3	0.18518	0.51852
4	0.23457	0.56790
5	0.26749	0.60082

Using mass lumping and Δt = 4/27 the oscillations practically disappear as shown in the first 5 time steps below

n	T_2^n	T_3^n
1	0.17778	0.62222
2	0.33185	0.62815
3	0.32316	0.66884
4	0.33459	0.66652
5	0.33321	0.66701

Runge-Kutta Methods (Explicit)

Consider the 1-degree of freedom ordinary differential equation $\dfrac{dy}{dt} = f(y,t)$. The second order Runge-Kutta method, also known as the modified Euler method is defined by: If y_n is known at t_n then y_{n+1} at time $t_{n+1} = t_n + \Delta t$ is found from

$$k_1 = f(y_n, t_n),\quad k_2 = f\left(y_n + \frac{\Delta t}{2} k_1,\ t_n + \frac{\Delta t}{2} \right),\quad y_{n+1} = y_n + \Delta t \cdot k_2$$

The method is $O\left((\Delta t)^2\right)$. An algorithm which is $O\left((\Delta t)^4\right)$, called the fourth order or "classical" Runge-Kutta method is given by

$$k_1 = f(y_n, t_n),\ k_2 = f(y_n + \frac{\Delta t}{2} k_1, t_n + \frac{\Delta t}{2}),\ k_3 = f(y_n + \frac{\Delta t}{2} k_2, t_n + \frac{\Delta t}{2})$$

$$k_4 = f(y_n + \Delta t\, k_3, t_n + \Delta t),\ y_{n+1} = y_n + \frac{\Delta t}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right)$$

The extension to vector systems of the form $\mathbf{C}\dot{\mathbf{u}} + \mathbf{F}(\mathbf{u}) = \mathbf{0}$ is obtained after first lumping the mass to get $\dot{\mathbf{u}} = -\bar{\mathbf{C}}^{-1}\mathbf{F}(\mathbf{u}) \equiv \mathbf{G}(\mathbf{u})$. We can now re-write the agorithms replacing y by $\mathbf{u}$ and f by $\mathbf{G}$. That is, the second order Runge-Kutta method becomes

$$\mathbf{K}_1 = \mathbf{G}(\mathbf{u}^n, t_n),\ \mathbf{K}_2 = \mathbf{G}(\mathbf{u}^n + \frac{\Delta t}{2}\mathbf{K}_1, t_n + \frac{\Delta t}{2}),\ \mathbf{u}^{n+1} = \mathbf{u}^n + \Delta t \mathbf{K}_2$$

The fourth order method becomes

$$\mathbf{K}_1 = \mathbf{G}(\mathbf{u}^n, t_n),\ \mathbf{K}_2 = \mathbf{G}(\mathbf{u}^n + \frac{\Delta t}{2}\mathbf{K}_1, t_n + \frac{\Delta t}{2}),\ \mathbf{K}_3 = \mathbf{G}(\mathbf{u}^n + \frac{\Delta t}{2}\mathbf{K}_2, t_n + \frac{\Delta t}{2})$$

$$\mathbf{K}_4 = \mathbf{G}(\mathbf{u}^n + \Delta t\mathbf{K}_3, t_n + \Delta t),\ \mathbf{u}^{n+1} = \mathbf{u}^n + \frac{\Delta t}{6}(\mathbf{K}_1 + 2\mathbf{K}_2 + 2\mathbf{K}_3 + \mathbf{K}_4)$$

Example

Apply the second order Runge-Kutta method to the Burgers equation.

$$\frac{\partial u}{\partial t} u \frac{\partial u}{\partial x} = \varepsilon \frac{\partial^2 u}{\partial x^2},\quad 0 < x < 1,\quad u(0,t) = 1,\ u(1,t) = 0),\quad u(x,0) = \begin{cases} 1 \text{ if } x \le \frac{1}{4} \\ 0 \text{ if } x > \frac{1}{4} \end{cases}$$

Setting ε = 1, the semidescrete Galerkin form using 4 Linear elements is

$$\frac{1}{24}\begin{bmatrix}4&1&0\\1&4&1\\0&1&4\end{bmatrix}\begin{bmatrix}\dot{u}_2\\ \dot{u}_3\\ \dot{u}_4\end{bmatrix}=-\begin{bmatrix}8u_2-4u_3+\frac{1}{6}u_2u_3+\frac{1}{6}u_3^2-\frac{26}{6}\\ -4u_2+8u_3-4u_4-\frac{1}{6}u_2^2-\frac{1}{6}u_2u_3+\frac{1}{6}u_3u_4+\frac{1}{6}u_4^2\\ -4u_3+8u_4-\frac{1}{6}u_3^2-\frac{1}{6}u_3u_4\end{bmatrix}$$

After lumping and inverting the Mass Matrix

$$\begin{bmatrix}\dot{u}_2\\ \dot{u}_3\\ \dot{u}_4\end{bmatrix}=-\begin{bmatrix}\frac{4}{5}\left(-48u_2+24u_3-u_2u_3-u_3^2+25\right)\\ \frac{2}{3}\left(24u_2-48u_3+24u_4+u_2^2+u_2u_3-u_3u_4-u_4^2\right)\\ \frac{4}{5}\left(24u_3-48u_4+u_3^2+u_3u_4\right)\end{bmatrix}$$

Now apply the second order Runge-Kutta method

25

$$\mathbf{K}_1=\begin{bmatrix}k_{12}\\k_{13}\\k_{14}\end{bmatrix}=-\begin{bmatrix}\frac{4}{5}\left(-48u_2^n+24u_3^n-u_2^nu_3^n-\left(u_3^n\right)^2+25\right)\\ \frac{2}{3}\left(24u_2^n-48u_3^n+24u_4^n+\left(u_2^n\right)^2+u_2^nu_3^n-u_3^nu_4^n-\left(u_4^n\right)^2\right)\\ \frac{4}{5}\left(24u_3^n-48u_4^n+\left(u_3^n\right)^2+u_3^nu_4^n\right)\end{bmatrix}$$

Set $\alpha=\dfrac{\Delta t}{2}$

$$\mathbf{K}_2=\begin{bmatrix}k_{22}\\k_{23}\\k_{24}\end{bmatrix}=-\begin{bmatrix}\frac{4}{5}\begin{pmatrix}-48\left(u_2^n+\alpha k_{12}\right)+24\left(u_3^n+\alpha k_{13}\right)-\left(u_2^n+\alpha k_{12}\right)\left(u_3^n+\alpha k_{13}\right)\\ -\left(u_3^n+\alpha k_{13}\right)^2+25\end{pmatrix}\\ \frac{2}{3}\begin{pmatrix}24\left(u_2^n+\alpha k_{12}\right)-48\left(u_3^n+\alpha k_{13}\right)+24\left(u_2^n+\alpha k_{12}\right)^2\\ +\left(u_2^n+\alpha k_{12}\right)\left(u_3^n+\alpha k_{13}\right)-\left(u_3^n+\alpha k_{13}\right)\left(u_4^n+\alpha k_{14}\right)-\left(u_4^n+\alpha k_{14}\right)^2\end{pmatrix}\\ \frac{4}{5}\begin{pmatrix}24\left(u_3^n+\alpha k_{13}\right)-48\left(u_4^n+\alpha k_{14}\right)+\left(u_3^n+\alpha k_{13}\right)^2\\ +\left(u_3^n+\alpha k_{13}\right)\left(u_4^n+\alpha k_{14}\right)\end{pmatrix}\end{bmatrix}$$

26

$\mathbf{u}^{n+1}$ becomes $\begin{bmatrix}u_2^{n+1}\\u_4^{n+1}\\u_4^{n+1}\end{bmatrix}=\begin{bmatrix}u_2^{n+1}+\Delta t k_{22}\\u_3^{n+1}+\Delta t k_{23}\\u_4^{n+1}+\Delta t k_{24}\end{bmatrix}$. Results shown below

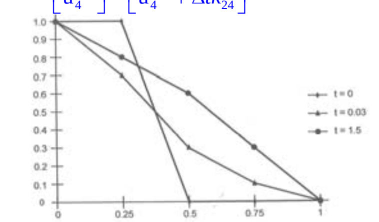


Figure 7.3 Solutions to Eq. (7.29) using the second-order Runge-Kutta method with four linear elements (ε = 1, Δt = 0.001).

t	u_2	u_3	u_4
0.00	1.0000	0.0000	0.0000
0.03	0.7095	0.3277	0.0685
1.50	0.8049	0.5670	0.2937

27

The greatest advantage of Explicit methods is that they do not require the solution of a linear system of equations. This greatly reduces the memory requirements and the CPU time.

The greatest disadvantage is the time step limitation, which can be very severe. When very small time steps must be used the CPU time can again grow excessively large. Vectorization and paralelization must be used for very large problems.

In general, stability analysis can only be done for linear equations. For Runge-Kutta methods we use the linearized equation $\dfrac{dy}{dt}+\alpha y=f(t)$ where $\alpha=\partial f/\partial y$ and depends on the solution.

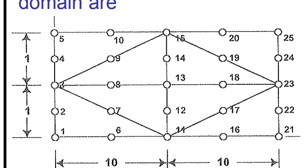
For the second order method the stability limit is $\alpha\Delta t\le 2$.

For the fourth order method we get $\alpha\Delta t\le 2.875$

These expressions give us a good idea of what the time step should be, but ARE NOT EXACT, only approximations. The difficulty to calculate the time step limit is a disadvantage of the Runge-Kutta methods.

28

Oden and Wellford (1972) used 4th order Runge-Kutta to simulate a Couette flow using quadratic triangular elements. The mesh and domain are



The boundary conditions are

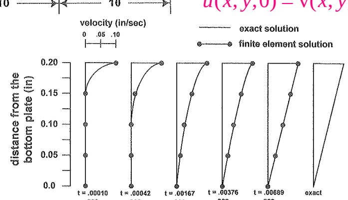
$u(x,0.2,t)=0.1$, $v(x,0.2,t)=0.0$

$u(x,0.0,t)=v(x,0.2,t)=0.0$

$$\left[-p+\mu\frac{\partial u}{\partial x}\right]_{x=0,2}=\left[\frac{\partial v}{\partial x}\right]_{x=0,2}=0$$

$\mu=0.00362$, $\rho=0.00242$

$u(x,y,0)=v(x,y,0)=0$



29