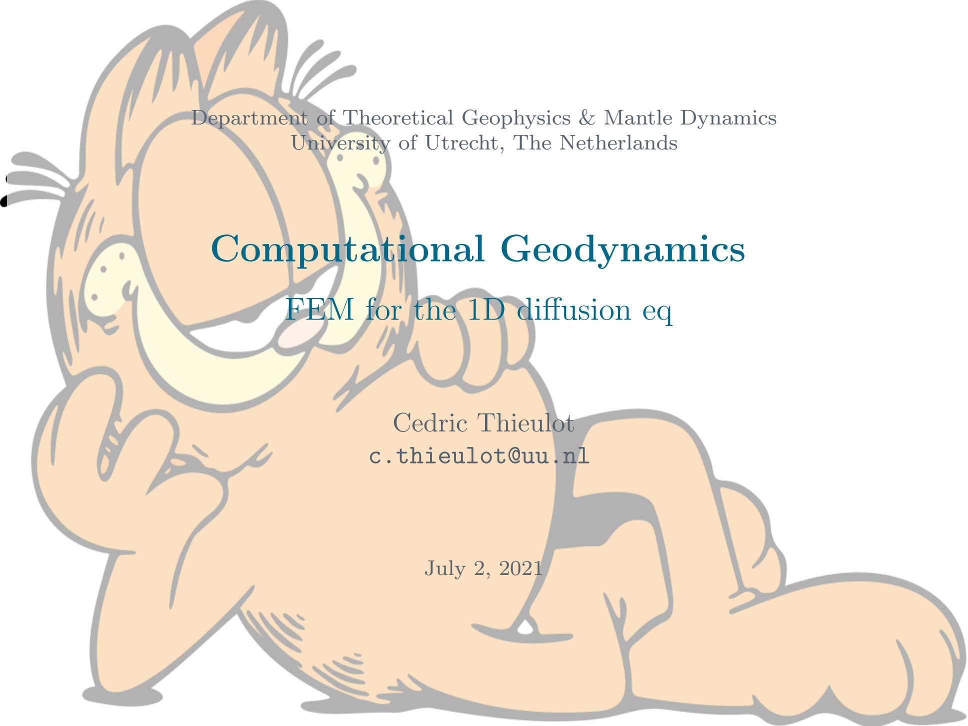


Department of Theoretical Geophysics & Mantle Dynamics
University of Utrecht, The Netherlands

Computational Geodynamics

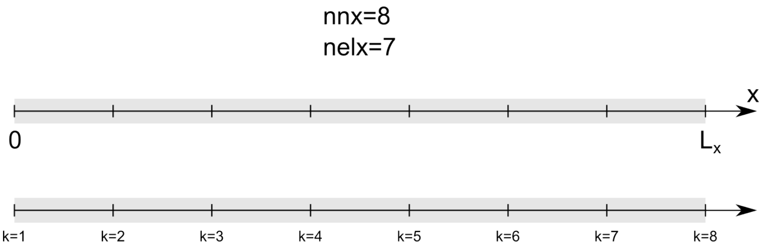
FEM for the 1D diffusion eq

Cedric Thieulot
c.thieulot@uu.nl

July 2, 2021

Content

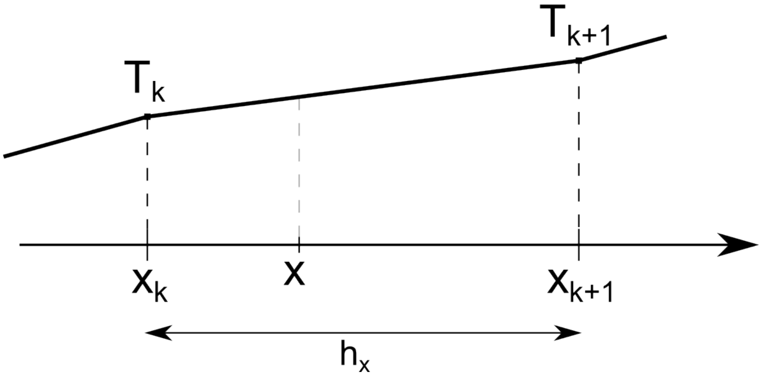
FEM in 1D
A simple 1D grid



- domain Ω of length L_x
- 1D grid, *nnx* nodes, *nelx* elements

FEM in 1D

Zoom on one element



FEM in 1D

From the strong form to the weak form

- We start with the 1D diffusion equation (no advection, no heat sources)

$$\rho C_p \frac{\partial T}{\partial t} = \frac{\partial}{\partial x}\left(k \frac{\partial T}{\partial x}\right)$$

- This is the strong form of the ODE to solve.
- I multiply this equation by a function $f(x)$ and integrate it over Ω:

$$\int_\Omega f(x) \rho C_p \frac{\partial T}{\partial t} dx = \int_\Omega f(x) \frac{\partial}{\partial x}\left(k \frac{\partial T}{\partial x}\right) dx$$

FEM in 1D

From the strong form to the weak form

- I integrate the r.h.s. by parts ($\int uv' = [uv] - \int u'v$):

$$\int_\Omega f(x) \frac{\partial}{\partial x}\left(k \frac{\partial T}{\partial x}\right) dx = \left[f(x) k \frac{\partial T}{\partial x} \right]_{\partial\Omega} - \int_\Omega \frac{\partial f}{\partial x} k \frac{\partial T}{\partial x} dx$$

- Assuming there is no heat flux prescribed on the boundary (i.e. $q_x = -k\partial T/\partial x = 0$), then:

$$\int_\Omega f(x) \frac{\partial}{\partial x}\left(k \frac{\partial T}{\partial x}\right) dx = - \int_\Omega \frac{\partial f}{\partial x} k \frac{\partial T}{\partial x} dx$$

We then obtain the weak form of the diffusion equation in 1D:

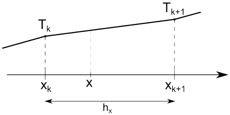
$$\boxed{\int_\Omega f(x)\rho C_p \frac{\partial T}{\partial t} dx + \int_\Omega \frac{\partial f}{\partial x} k \frac{\partial T}{\partial x} dx = 0}$$

We then use the additive property of the integral:

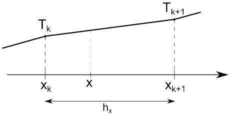
$$\int_\Omega \cdots = \sum_{elts} \int_{\Omega_e} \cdots$$

so that

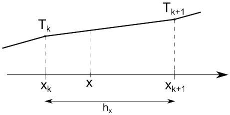
$$\sum_{elts} \left(\underbrace{\int_{\Omega_e} f(x)\rho C_p \frac{\partial T}{\partial t} dx}_{\Lambda_f^e} + \underbrace{\int_{\Omega_e} \frac{\partial f}{\partial x} k \frac{\partial T}{\partial x} dx}_{\Upsilon_f^e} \right) = 0$$



$$T(x) = \alpha T_k + \beta T_{k+1} \qquad x \in [x_k, x_{k+1}]$$



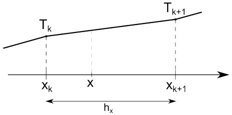
$$
\begin{aligned}
T(x) &= \alpha T_k + \beta T_{k+1} \qquad\qquad x \in [x_k, x_{k+1}] \\
&= N_k(x) T_k + N_{k+1}(x) T_{k+1}
\end{aligned}
$$



$$\begin{aligned} T(x) &= \alpha T_k + \beta T_{k+1} \qquad\qquad x \in [x_k, x_{k+1}] \\ &= N_k(x) T_k + N_{k+1}(x) T_{k+1} \end{aligned}$$

Example:

$$T(x) = \underbrace{\frac{x_{k+1} - x}{h_x}}_{N_k(x)} T_k + \underbrace{\frac{x - x_k}{h_x}}_{N_{k+1}(x)} T_{k+1}$$



$$
\begin{aligned}
T(x) &= \alpha T_k + \beta T_{k+1} \qquad\qquad\qquad x \in [x_k, x_{k+1}] \\
&= N_k(x) T_k + N_{k+1}(x) T_{k+1}
\end{aligned}
$$

Example:

$$
T(x) = \underbrace{\frac{x_{k+1} - x}{h_x}}_{N_k(x)} T_k + \underbrace{\frac{x - x_k}{h_x}}_{N_{k+1}(x)} T_{k+1}
$$

- $x = x_k$ yields $T(x_k) = T_k$
- $x = x_{k+1}$ yields $T(x_{k+1}) = T_{k+1}$
- $x = x_{1/2} = (x_k + x_{k+1})/2$ yields $T(x_{1/2}) = (T_k + T_{k+1})/2$

Let us go back to

$$\sum_{elts} \left(\underbrace{\int_{\Omega_e} f(x) \rho C_p \frac{\partial T}{\partial t} dx}_{\Lambda_f^e} + \underbrace{\int_{\Omega_e} \frac{\partial f}{\partial x} k \frac{\partial T}{\partial x} dx}_{\Upsilon_f^e} \right) = 0$$

and compute Λ_f^e and Υ_f^e separately.

$$
\begin{aligned}
\Lambda_f^e &= \int_{x_k}^{x_{k+1}} f(x)\rho C_p \dot{T}(x) dx \\
&= \int_{x_k}^{x_{k+1}} f(x)\rho C_p \; [N_k(x)\dot{T}_k + N_{k+1}(x)\dot{T}_{k+1}] \; dx \\
&= \int_{x_k}^{x_{k+1}} f(x)\rho C_p N_k(x)\dot{T}_k dx + \int_{x_k}^{x_{k+1}} f(x)\rho C_p N_{k+1}(x)\dot{T}_{k+1} dx \\
&= \left(\int_{x_k}^{x_{k+1}} f(x)\rho C_p N_k(x) dx \right) \dot{T}_k + \left(\int_{x_k}^{x_{k+1}} f(x)\rho C_p N_{k+1}(x) dx \right) \dot{T}_{k+1}
\end{aligned}
$$

FEM in 1D

.

- Taking $f(x) = N_k(x)$ and omitting '(x)' in the rhs:

$$\Lambda_{N_k} = \left(\int_{x_k}^{x_{k+1}} \rho C_p N_k N_k dx \right) \dot{T}_k + \left(\int_{x_k}^{x_{k+1}} \rho C_p N_k N_{k+1} dx \right) \dot{T}_{k+1}$$

FEM in 1D

- Taking $f(x) = N_k(x)$ and omitting '(x)' in the rhs:

$$\Lambda_{N_k} = \left(\int_{x_k}^{x_{k+1}} \rho C_p N_k N_k dx \right) \dot{T}_k + \left(\int_{x_k}^{x_{k+1}} \rho C_p N_k N_{k+1} dx \right) \dot{T}_{k+1}$$

- Taking $f(x) = N_{k+1}(x)$ and omitting '(x)' in the rhs:

$$\Lambda_{N_{k+1}} = \left(\int_{x_k}^{x_{k+1}} \rho C_p N_{k+1} N_k dx \right) \dot{T}_k + \left(\int_{x_k}^{x_{k+1}} \rho C_p N_{k+1} N_{k+1} dx \right) \dot{T}_{k+1}$$

FEM in 1D

.

$$
\left(\begin{array}{c} \Lambda_{N_k} \\ \\ \Lambda_{N_{k+1}} \end{array}\right) = \left(\begin{array}{cc} \int_{x_k}^{x_{k+1}} N_k \rho C_p N_k dx & \int_{x_k}^{x_{k+1}} N_k \rho C_p N_{k+1} dx \\ \\ \int_{x_k}^{x_{k+1}} N_{k+1} \rho C_p N_k dx & \int_{x_k}^{x_{k+1}} N_{k+1} \rho C_p N_{k+1} dx \end{array}\right) \cdot \left(\begin{array}{c} \dot{T}_k \\ \\ \dot{T}_{k+1} \end{array}\right)
$$

or,

$$\left(\begin{array}{c} \Lambda_{N_k} \\ \\ \Lambda_{N_{k+1}} \end{array} \right) = \left(\begin{array}{cc} \int_{x_k}^{x_{k+1}} N_k \rho C_p N_k dx & \int_{x_k}^{x_{k+1}} N_k \rho C_p N_{k+1} dx \\ \\ \int_{x_k}^{x_{k+1}} N_{k+1} \rho C_p N_k dx & \int_{x_k}^{x_{k+1}} N_{k+1} \rho C_p N_{k+1} dx \end{array} \right) \cdot \left(\begin{array}{c} \dot{T}_k \\ \\ \dot{T}_{k+1} \end{array} \right)$$

or,

$$\left(\begin{array}{c} \Lambda_{N_k} \\ \\ \Lambda_{N_{k+1}} \end{array} \right) = \left[\int_{x_k}^{x_{k+1}} \rho C_p \left(\begin{array}{cc} N_k N_k & N_k N_{k+1} \\ \\ N_{k+1} N_k & N_{k+1} N_{k+1} \end{array} \right) dx \right] \cdot \left(\begin{array}{c} \dot{T}_k \\ \\ \dot{T}_{k+1} \end{array} \right)$$

FEM in 1D

Finally, we can define the vectors

$$\vec{N}^T = \left(\begin{array}{c} N_k(x) \\ \\ N_{k+1}(x) \end{array} \right)$$

and

$$\vec{T}^e = \left(\begin{array}{c} T_k \\ \\ T_{k+1} \end{array} \right) \qquad \dot{\vec{T}}^e = \left(\begin{array}{c} \dot{T}_k \\ \\ \dot{T}_{k+1} \end{array} \right)$$

so that

$$\left(\begin{array}{c} \Lambda_{N_k} \\ \\ \Lambda_{N_{k+1}} \end{array} \right) = \left(\int_{x_k}^{x_{k+1}} \vec{N}^T \rho C_p \vec{N} dx \right) \cdot \dot{\vec{T}}^e$$

FEM in 1D

Back to the diffusion term:

$$
\begin{aligned}
\Upsilon_f^e &= \int_{x_k}^{x^{k+1}} \frac{\partial f}{\partial x} k \frac{\partial T}{\partial x} dx \\
&= \int_{x_k}^{x^{k+1}} \frac{\partial f}{\partial x} k \frac{\partial (N_k(x) T_k + N_{k+1}(x) T_{k+1})}{\partial x} dx \\
&= \left(\int_{x_k}^{x^{k+1}} \frac{\partial f}{\partial x} k \frac{\partial N_k}{\partial x} dx \right) T_k + \left(\int_{x_k}^{x^{k+1}} \frac{\partial f}{\partial x} k \frac{\partial N_{k+1}}{\partial x} dx \right) T_{k+1}
\end{aligned}
$$

FEM in 1D

- Taking $f(x) = N_k(x)$

$$\Upsilon_{N_k} = \left(\int_{x_k}^{x^{k+1}} k \frac{\partial N_k}{\partial x} \frac{\partial N_k}{\partial x} dx \right) T_k + \left(\int_{x_k}^{x^{k+1}} k \frac{\partial N_k}{\partial x} \frac{\partial N_{k+1}}{\partial x} dx \right) T_{k+1}$$

FEM in 1D

- Taking $f(x) = N_k(x)$

$$\Upsilon_{N_k} = \left(\int_{x_k}^{x^{k+1}} k \frac{\partial N_k}{\partial x} \frac{\partial N_k}{\partial x} dx \right) T_k + \left(\int_{x_k}^{x^{k+1}} k \frac{\partial N_k}{\partial x} \frac{\partial N_{k+1}}{\partial x} dx \right) T_{k+1}$$

- Taking $f(x) = N_{k+1}(x)$

$$\Upsilon_{N_{k+1}} = \left(\int_{x_k}^{x^{k+1}} k \frac{\partial N_{k+1}}{\partial x} \frac{\partial N_k}{\partial x} dx \right) T_k + \left(\int_{x_k}^{x^{k+1}} k \frac{\partial N_{k+1}}{\partial x} \frac{\partial N_{k+1}}{\partial x} dx \right) T_{k+1}$$

FEM in 1D

$$
\left(\begin{array}{c} \Upsilon_{N_k} \\ \\ \Upsilon_{N_{k+1}} \end{array}\right) = \left(\begin{array}{cc} \int_{x_k}^{x^{k+1}} \frac{\partial N_k}{\partial x} k \frac{\partial N_k}{\partial x} dx & \int_{x_k}^{x^{k+1}} \frac{\partial N_k}{\partial x} k \frac{\partial N_{k+1}}{\partial x} dx \\ \\ \int_{x_k}^{x^{k+1}} \frac{\partial N_{k+1}}{\partial x} k \frac{\partial N_k}{\partial x} dx & \int_{x_k}^{x^{k+1}} \frac{\partial N_{k+1}}{\partial x} k \frac{\partial N_{k+1}}{\partial x} dx \end{array}\right) \cdot \left(\begin{array}{c} T_k \\ \\ T_{k+1} \end{array}\right)
$$

FEM in 1D

$$\left(\begin{array}{c} \Upsilon_{N_k} \\ \\ \Upsilon_{N_{k+1}} \end{array}\right) = \left(\begin{array}{cc} \int_{x_k}^{x^{k+1}} \frac{\partial N_k}{\partial x} k \frac{\partial N_k}{\partial x} dx & \int_{x_k}^{x^{k+1}} \frac{\partial N_k}{\partial x} k \frac{\partial N_{k+1}}{\partial x} dx \\ \\ \int_{x_k}^{x^{k+1}} \frac{\partial N_{k+1}}{\partial x} k \frac{\partial N_k}{\partial x} dx & \int_{x_k}^{x^{k+1}} \frac{\partial N_{k+1}}{\partial x} k \frac{\partial N_{k+1}}{\partial x} dx \end{array}\right) \cdot \left(\begin{array}{c} T_k \\ \\ T_{k+1} \end{array}\right)$$

or,

$$\left(\begin{array}{c} \Upsilon_{N_k} \\ \\ \Upsilon_{N_{k+1}} \end{array}\right) = \left[\int_{x_k}^{x^{k+1}} k \left(\begin{array}{cc} \frac{\partial N_k}{\partial x} \frac{\partial N_k}{\partial x} & \frac{\partial N_k}{\partial x} \frac{\partial N_{k+1}}{\partial x} \\ \\ \frac{\partial N_{k+1}}{\partial x} \frac{\partial N_k}{\partial x} & \frac{\partial N_{k+1}}{\partial x} \frac{\partial N_{k+1}}{\partial x} \end{array}\right) dx\right] \cdot \left(\begin{array}{c} T_k \\ \\ T_{k+1} \end{array}\right)$$

Finally, we can define the vector

$$\vec{B}^T = \left(\begin{array}{c} \frac{\partial N_k}{\partial x} \\ \\ \frac{\partial N_{k+1}}{\partial x} \end{array} \right)$$

so that

$$\left(\begin{array}{c} \Upsilon_{N_k} \\ \\ \Upsilon_{N_{k+1}} \end{array} \right) = \left(\int_{x_k}^{x_{k+1}} \vec{B}^T k \vec{B} dx \right) \cdot \vec{T}^e$$

The weak form discretised over 1 element becomes

$$\underbrace{\left(\int_{x_k}^{x_{k+1}} \vec{N}^T \rho C_p \vec{N} dx\right)}_{\boldsymbol{M}^e} \cdot \dot{\vec{T}}^e + \underbrace{\left(\int_{x_k}^{x_{k+1}} \vec{B}^T k \vec{B} dx\right)}_{\boldsymbol{K}_d^e} \cdot \vec{T}^e = \vec{0}$$

or,

$$\boxed{\boldsymbol{M}^e \cdot \dot{\vec{T}}^e + \boldsymbol{K}_d^e \cdot \vec{T}^e = \vec{0}}$$

or,

$$\boxed{\boldsymbol{M}^e \cdot \frac{\partial \vec{T}^e}{\partial t} + \boldsymbol{K}_d^e \cdot \vec{T}^e = \vec{0}}$$

FEM in 1D

The author wishes to thank R.S. Huismans and P. Steer for stimulating discussions and early reviews of the manuscript. Technical support by Anshul Gupta throughout the code development is appreciated. This work was funded through Norwegian Research Council Grant 177489/V30 and E.U. International Reintegration Grant MIRC-CT-2006-046437 to R.S. Huismans. The calculations presented in here were conducted at the Bergen Center for Computational Science.[13] Prof. Taras Gerya is thanked for his rapid and constructive review of the manuscript.

Appendix A. FEM formulation of equations

The Galerkin finite element equation corresponding to Eq. (12) is

$$(\boldsymbol{K}_\mu + \boldsymbol{K}_\lambda) \cdot \boldsymbol{v} = \boldsymbol{B}$$

with

$$\boldsymbol{K}_\mu = \int_\Omega \boldsymbol{B}^T \cdot \boldsymbol{D}_\mu \cdot \boldsymbol{B} d\Omega$$

$$\boldsymbol{K}_\lambda = \int_\Omega \boldsymbol{B}^T \cdot \boldsymbol{D}_\lambda \cdot \boldsymbol{B} d\Omega$$

$$\boldsymbol{B} = \int_\Omega \boldsymbol{N}^T \rho \boldsymbol{g} d\Omega$$

$$\boldsymbol{D}_\mu^{2D} = \mu \begin{pmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \boldsymbol{D}_\lambda^{2D} = \lambda \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

and where $\boldsymbol{N}$ is the vector of shape functions, and $\boldsymbol{B}$ is the matrix of spatial derivatives of the shape functions.

The finite element equation corresponding to the heat transfer equation is

$$\boxed{\boldsymbol{M}_c \cdot \frac{\partial \boldsymbol{T}}{\partial t} + (\boldsymbol{K}_a + \boldsymbol{K}_d) \cdot \boldsymbol{T} = \boldsymbol{F}}$$

where

$$\boxed{\boldsymbol{M}_c = \int_\Omega \boldsymbol{N}^T \rho c_P \boldsymbol{N} d\Omega}$$

$$\boldsymbol{K}_a = \int_\Omega (\boldsymbol{N}^*)^T \rho c_P \boldsymbol{v} \cdot \boldsymbol{B} d\Omega \qquad \text{(A.1)}$$

$$\boxed{\boldsymbol{K}_d = \int_\Omega \boldsymbol{B}^T k \boldsymbol{B} d\Omega}$$

$$\boldsymbol{F} = \int_\Omega \boldsymbol{N}^T H d\Omega$$

where $\boldsymbol{T}$ is the vector of the nodal temperatures.

In the case where advection dominates over diffusion, the standard Galerkin approach of the advection term leads to problematic oscillations, and a stabilisation scheme is needed. A streamline-upwind Petrov–Galerkin (SUPG) method is therefore implemented, which translates in the modified $\boldsymbol{N}^*$ term in Eq. (A.1):

$$(\boldsymbol{N}^*)^T = \boldsymbol{N}^T + \tau \boldsymbol{v} \cdot \boldsymbol{B}$$

where τ a dimensionless parameter. The case $\tau = 0$ is equivalent to the Bubnov–Galerkin method. The choice of the parameter τ in the context of FEM stabilisation schemes is discussed in Tezduyar and Osawa (2000) and is calculated as follows:

$$\tau = \left(\frac{1}{(\tau_1)^r} + \frac{1}{(\tau_2)^r} + \frac{1}{(\tau_3)^r} \right)^{-1/r}$$

with often $r = 1$ or $r = 1/2$ and

$$\tau_1 = \frac{h}{2|\boldsymbol{v}|}, \quad \tau_2 = \theta dt, \quad \tau_3 = \frac{h^2 \rho c_P}{k} \qquad \text{(A.2)}$$

where h is a measure of the element size and θ is related to the time discretisation scheme ($\theta = 1/2$ in this case as it corresponds to the implemented mid-point implicit scheme, see for instance Braun (2003)). In order to illustrate the beneficial aspect of the SUPG scheme, let us look at the standard problem of the one-dimensional advection of a scalar field containing a steep front (diffusion and source terms are null). In Fig. 19 is shown the analytical initial scalar field. It is a challenging benchmark as the numerical treatment of the advection of such a discontinuity often leads to non-negligible oscillations. The unit segment is discretised by means of 50 elements, over which a unit velocity field is prescribed. The time step is chosen so that $dt = 0.1h/|v| = 0.002$. The discontinuity is initially placed at $x = 1/4$ and after 250 time steps, it is expected to have reached the position $x = 3/4$.

In Fig. 19 are shown the advected field for various values of the dimensionless coefficient $\gamma = \tau|\boldsymbol{v}|/h$. The Galerkin scheme ($\gamma = 0$) leads to strong oscillations, as already described by Donea and Huerta (2003). Using Eq. (A.2), one arrives to $\gamma = 0.045$, which leads to a desired removal of the oscillations through a small amount of numerical diffusion. Braun (2003) argues for a constant

[13] http://www.bccs.uni.no/

Thieulot, PEPI 188, 2011

FEM in 1D

As we have seen in the context of the FD method, we use a first order in time discretisation for the time derivative:

$$\dot{\vec{T}} = \frac{\partial \vec{T}}{\partial t} = \frac{\vec{T}^{new} - \vec{T}^{old}}{\delta t}$$

Using an implicit scheme, we get

$$\boldsymbol{M}^e \cdot \frac{\vec{T}^{new} - \vec{T}^{old}}{\delta t} + \boldsymbol{K}_d^e \cdot \vec{T}^{new} = \vec{0}$$

or,

$$\boxed{(\boldsymbol{M}^e + \boldsymbol{K}_d^e \delta t) \cdot \vec{T}^{new} = \boldsymbol{M}^e \cdot \vec{T}^{old}}$$

with

$$\boldsymbol{M}^e = \int_{x_k}^{x_{k+1}} \vec{N}^T \rho C_p \vec{N} dx \qquad \boldsymbol{K}_d^e = \int_{x_k}^{x_{k+1}} \vec{B}^T k \vec{B} dx$$

FEM in 1D

Let us compute $\boldsymbol{M}$ for an element:

$$\boldsymbol{M}^e = \int_{x_k}^{x_{k+1}} \vec{N}^T \rho C_p \vec{N} dx$$

with

$$\vec{N}^T = \left(\begin{array}{c} N_k(x) \\ \\ N_{k+1}(x) \end{array} \right) = \left(\begin{array}{c} \frac{x_{k+1}-x}{h_x} \\ \\ \frac{x-x_k}{h_x} \end{array} \right)$$

Then

$$\boldsymbol{M}^e = \left(\begin{array}{cc} M_{11} & M_{12} \\ M_{21} & M_{22} \end{array} \right) = \left(\begin{array}{cc} \int_{x_k}^{x_{k+1}} \rho C_p N_k N_k dx & \int_{x_k}^{x_{k+1}} \rho C_p N_k N_{k+1} dx \\ \\ \int_{x_k}^{x_{k+1}} \rho C_p N_{k+1} N_k dx & \int_{x_k}^{x_{k+1}} \rho C_p N_{k+1} N_{k+1} dx \end{array} \right)$$

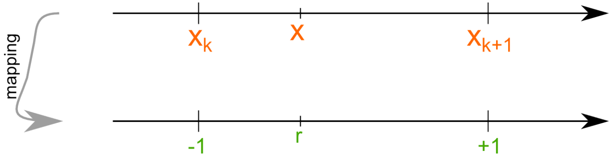
I need to compute 3 integrals ($M_{12} = M_{21}$)

FEM in 1D

Let us look at M_{11}:

$$M_{11} = \int_{x_k}^{x_{k+1}} \rho C_p N_k(x) N_k(x) dx = \int_{x_k}^{x_{k+1}} \rho C_p \frac{x_{k+1}-x}{h_x} \frac{x_{k+1}-x}{h_x} dx$$

It is customary to carry out a change of variables (mapping $x \rightarrow r$):



$$r = \frac{2}{h_x}(x - x_k) - 1 \qquad\qquad x = \frac{h_x}{2}(1+r) + x_k$$

FEM in 1D

In what follows we assume for simplicity that ρ and C_p are constant within each element.

$$M_{11} = \rho C_p \int_{x_k}^{x_{k+1}} \frac{x_{k+1}-x}{h_x} \frac{x_{k+1}-x}{h_x} dx = \frac{\rho C_p h_x}{8} \int_{-1}^{+1} (1-r)(1-r) dr = \frac{h_x}{3} \rho C_p$$

Similarly we arrive at

$$M_{12} = \rho C_p \int_{x_k}^{x_{k+1}} \frac{x_{k+1}-x}{h_x} \frac{x-x_k}{h_x} dx = \frac{\rho C_p h_x}{8} \int_{-1}^{+1} (1-r)(1+r) dr = \frac{h_x}{6} \rho C_p$$

and

$$M_{22} = \rho C_p \int_{x_k}^{x_{k+1}} \frac{x-x_k}{h_x} \frac{x-x_k}{h_x} dx = \frac{\rho C_p h_x}{8} \int_{-1}^{+1} (1+r)(1+r) dr = \frac{h_x}{3} \rho C_p$$

Finally

$$\boxed{\boldsymbol{M}^e = \frac{h_x}{3}\rho C_p \left(\begin{array}{cc} 1 & 1/2 \\ 1/2 & 1 \end{array} \right)}$$

FEM in 1D

In the new coordinate system, the shape functions

$$N_k(x) = \frac{x_{k+1} - x}{h_x} \qquad N_{k+1}(x) = \frac{x - x_k}{h_x}$$

become

$$N_k(r) = \frac{1}{2}(1 - r) \qquad N_{k+1}(r) = \frac{1}{2}(1 + r)$$

FEM in 1D

In the new coordinate system, the shape functions

$$N_k(x) = \frac{x_{k+1} - x}{h_x} \qquad N_{k+1}(x) = \frac{x - x_k}{h_x}$$

become

$$N_k(r) = \frac{1}{2}(1 - r) \qquad N_{k+1}(r) = \frac{1}{2}(1 + r)$$

Also,

$$\frac{\partial N_k}{\partial x} = -\frac{1}{h_x} \qquad \frac{\partial N_{k+1}}{\partial x} = \frac{1}{h_x}$$

so that

$$\vec{B}^T = \left(\begin{array}{c} \frac{\partial N_k}{\partial x} \\ \\ \frac{\partial N_{k+1}}{\partial x} \end{array} \right) = \left(\begin{array}{c} -\frac{1}{h_x} \\ \\ \frac{1}{h_x} \end{array} \right)$$

FEM in 1D

We here also assume that k is constant within the element:

$$\boldsymbol{K}_d = \int_{x_k}^{x_{k+1}} \vec{B}^T k \vec{B} dx = k \int_{x_k}^{x_{k+1}} \vec{B}^T \vec{B} dx$$

simply becomes

$$\boldsymbol{K}_d = k \int_{x_k}^{x_{k+1}} \frac{1}{h_x^2} \left(\begin{array}{cc} 1 & -1 \\ -1 & 1 \end{array} \right) dx$$

FEM in 1D

We here also assume that k is constant within the element:

$$\boldsymbol{K}_d = \int_{x_k}^{x_{k+1}} \vec{B}^T k \vec{B} dx = k \int_{x_k}^{x_{k+1}} \vec{B}^T \vec{B} dx$$

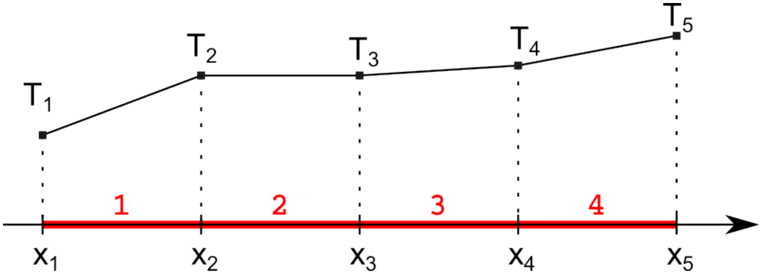
simply becomes

$$\boldsymbol{K}_d = k \int_{x_k}^{x_{k+1}} \frac{1}{h_x^2} \left(\begin{array}{cc} 1 & -1 \\ -1 & 1 \end{array} \right) dx$$

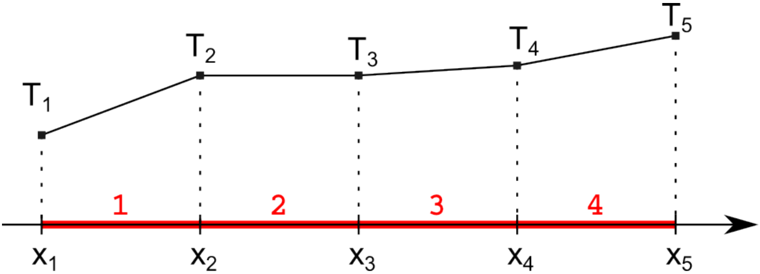
and then

$$\boxed{\boldsymbol{K}_d = \frac{k}{h_x} \left(\begin{array}{cc} 1 & -1 \\ -1 & 1 \end{array} \right)}$$

FEM in 1D



FEM in 1D



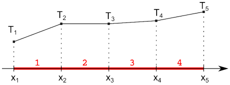
For each element

$$\underbrace{(\boldsymbol{M}^e + \boldsymbol{K}_d^e \, \delta t)}_{\boldsymbol{A}^e} \cdot \vec{T}^{new} = \underbrace{\boldsymbol{M}^e \cdot \vec{T}^{old}}_{\vec{b}^e}$$

or,

$$\boldsymbol{A}^e \cdot \vec{T}^{new} = \vec{b}^e$$

FEM in 1D



- element 1

$$\boldsymbol{A}^1 \cdot \begin{pmatrix} T_1 \\ T_2 \end{pmatrix} = \boldsymbol{b}^1$$

- element 2

$$\boldsymbol{A}^2 \cdot \begin{pmatrix} T_2 \\ T_3 \end{pmatrix} = \boldsymbol{b}^2$$

- element 3

$$\boldsymbol{A}^3 \cdot \begin{pmatrix} T_3 \\ T_4 \end{pmatrix} = \boldsymbol{b}^3$$

- element 4

$$\boldsymbol{A}^4 \cdot \begin{pmatrix} T_4 \\ T_5 \end{pmatrix} = \boldsymbol{b}^4$$

FEM in 1D

- element 1

$$\left\{ \begin{array}{l} A^1_{11} T_1 + A^1_{12} T_2 = b^1_x \\ A^1_{21} T_1 + A^1_{22} T_2 = b^1_y \end{array} \right.$$

- element 2

$$\left\{ \begin{array}{l} A^2_{11} T_2 + A^2_{12} T_3 = b^2_1 \\ A^2_{21} T_2 + A^2_{22} T_3 = b^2_2 \end{array} \right.$$

- element 3

$$\left\{ \begin{array}{l} A^3_{11} T_3 + A^3_{12} T_4 = b^3_1 \\ A^3_{21} T_3 + A^3_{22} T_4 = b^3_2 \end{array} \right.$$

- element 4

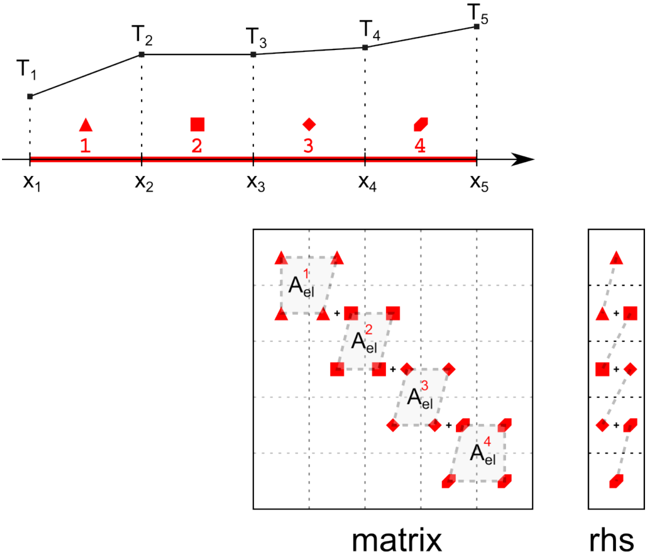
$$\left\{ \begin{array}{l} A^4_{11} T_4 + A^4_{12} T_5 = b^4_1 \\ A^4_{21} T_4 + A^4_{22} T_5 = b^4_2 \end{array} \right.$$

FEM in 1D

All equations can be cast into a single linear system: this is the assembly phase.

$$
\begin{pmatrix}
A^1_{11} & A^1_{12} & & & \\
A^1_{21} & A^1_{22}+A^2_{11} & A^2_{12} & & \\
 & A^2_{21} & A^2_{22}+A^3_{11} & A^3_{12} & \\
 & & A^3_{21} & A^3_{22}+A^4_{11} & A^4_{12} \\
 & & & A^4_{21} & A^4_{22}
\end{pmatrix}
\begin{pmatrix}
T_1 \\ T_2 \\ T_3 \\ T_4 \\ T_5
\end{pmatrix}
=
\begin{pmatrix}
b^1_1 \\ b^1_2 + b^2_1 \\ b^2_2 + b^3_1 \\ b^3_2 + b^4_1 \\ b^4_2
\end{pmatrix}
$$

FEM in 1D



The assembled matrix system also takes the form

$$\begin{pmatrix} A_{11} & A_{12} & & & \\ A_{21} & A_{22} & A_{23} & & \\ & A_{32} & A_{33} & A_{34} & \\ & & A_{43} & A_{44} & A_{45} \\ & & & A_{54} & A_{55} \end{pmatrix} \begin{pmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \\ T_5 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{pmatrix}$$

Let us assume that we wish to fix the temperature at node 2.

Let us assume that we wish to fix the temperature at node 2.
Then

$$T_2 = T^{bc}$$

Let us assume that we wish to fix the temperature at node 2. Then

$$T_2 = T^{bc}$$

This can be cast as

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \\ T_5 \end{pmatrix} = \begin{pmatrix} 0 \\ T^{bc} \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

This replaces the second line in the previous matrix equation:

$$
\begin{pmatrix}
A_{11} & A_{12} & & & \\
0 & 1 & 0 & & \\
& A_{32} & A_{33} & A_{34} & \\
& & A_{43} & A_{44} & A_{45} \\
& & & A_{54} & A_{55}
\end{pmatrix}
\begin{pmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \\ T_5 \end{pmatrix}
=
\begin{pmatrix} b_1 \\ T^{bc} \\ b_3 \\ b_4 \\ b_5 \end{pmatrix}
$$

- Before applying the b.c., the assembled matrix $\boldsymbol{A}$ is symmetric.

- Before applying the b.c., the assembled matrix $\boldsymbol{A}$ is symmetric.
- Now it is not symmetric anymore.

- Before applying the b.c., the assembled matrix $\boldsymbol{A}$ is symmetric.
- Now it is not symmetric anymore.
- Some solvers require a symmetric matrix.

- Before applying the b.c., the assembled matrix $\boldsymbol{A}$ is symmetric.
- Now it is not symmetric anymore.
- Some solvers require a symmetric matrix.
- Symmetric matrices can be stored more economically.

- Before applying the b.c., the assembled matrix $\boldsymbol{A}$ is symmetric.
- Now it is not symmetric anymore.
- Some solvers require a symmetric matrix.
- Symmetric matrices can be stored more economically.

Can we restore symmetry ?

- Before applying the b.c., the assembled matrix $\boldsymbol{A}$ is symmetric.
- Now it is not symmetric anymore.
- Some solvers require a symmetric matrix.
- Symmetric matrices can be stored more economically.

Can we restore symmetry ?
yes.

- Before applying the b.c., the assembled matrix $\boldsymbol{A}$ is symmetric.
- Now it is not symmetric anymore.
- Some solvers require a symmetric matrix.
- Symmetric matrices can be stored more economically.

Can we restore symmetry ?
yes.
duh :)

$$
\begin{pmatrix}
A_{11} & 0 & & & \\
0 & 1 & 0 & & \\
 & 0 & A_{33} & A_{34} & \\
 & & A_{43} & A_{44} & A_{45} \\
 & & & A_{54} & A_{55}
\end{pmatrix}
\begin{pmatrix}
T_1 \\ T_2 \\ T_3 \\ T_4 \\ T_5
\end{pmatrix}
=
\begin{pmatrix}
b_1 - A_{12} T^{bc} \\ T^{bc} \\ b_3 - A_{32} T^{bc} \\ b_4 \\ b_5
\end{pmatrix}
$$

Matrix is symmetric !

FEM in 1D
Applying b.c.

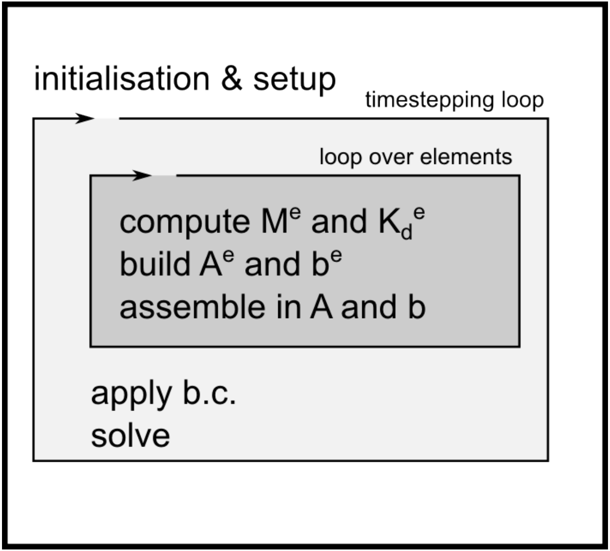
The matrix is now symmetric, but its condition number may have been changed.

The matrix is now symmetric, but its condition number may have been changed.
Fix:

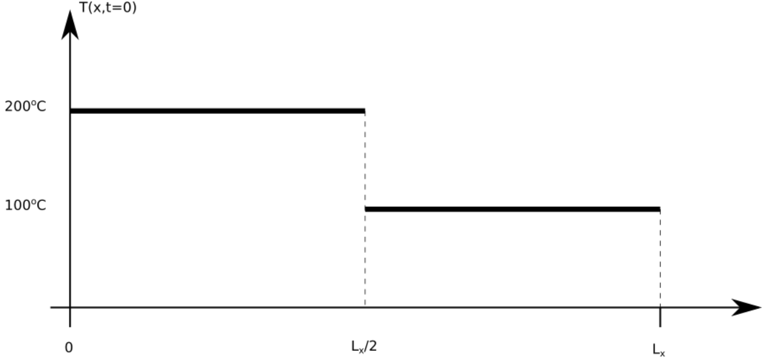
$$
\begin{pmatrix}
A_{11} & 0 & & & \\
0 & A_{22} & 0 & & \\
 & 0 & A_{33} & A_{34} & \\
 & & A_{43} & A_{44} & A_{45} \\
 & & & A_{54} & A_{55}
\end{pmatrix}
\begin{pmatrix}
T_1 \\ T_2 \\ T_3 \\ T_4 \\ T_5
\end{pmatrix}
=
\begin{pmatrix}
b_1 - A_{12}T^{bc} \\
A_{22}T^{bc} \\
b_3 - A_{32}T^{bc} \\
b_4 \\
b_5
\end{pmatrix}
$$

FEM in 1D

Program structure



The initial temperature profile is as follows:



$$T(x, t = 0) = 200 \quad x < L_x/2 \qquad T(x, t = 0) = 100 \quad x \geq L_x/2$$

Exercise

The properties of the material are as follows:

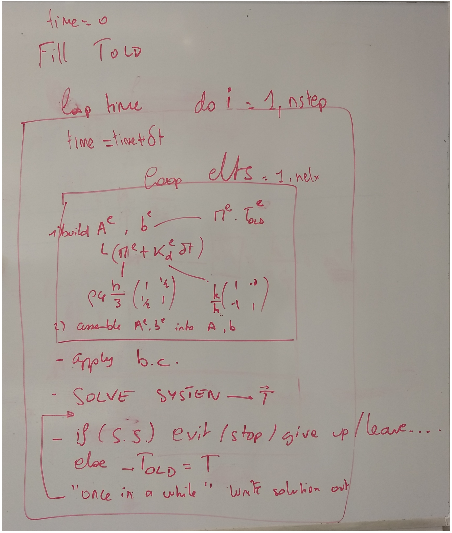
$$\rho = 3000 \qquad k = 3 \qquad C_p = 1000$$

Furthermore, $L_x = 100$km.
Boundary conditions are:

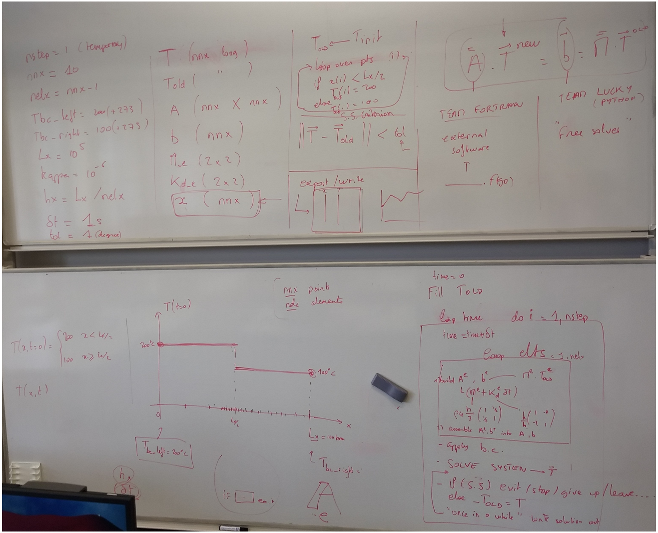
$$T(t, x = 0) = 200^\circ C \qquad T(t, x = L_x) = 100^\circ C$$

There are `nelx` elements and `nnx` nodes. All elements are `hx` long.
The code will carry out `nstep` timesteps of length `dt`.

Exercise


time = 0
Fill TOLD
loop time do i = 1, nstep
time = time + δt
loop elts = 1, nelx
1) build A^e, b^e
M^e . T^e_OLD
L(M^e + K_d^e δt)
ρC h/3 (1 1/2 ; 1/2 1)
k/h (1 -1 ; -1 1)
2) assemble A^e, b^e into A, b
- apply b.c.
- SOLVE SYSTEM → T
- if (S.S.) exit / stop / give up / leave....
else → TOLD = T
"once in a while" write solution out

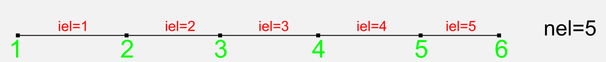
Exercise



nstep = 1 (temporary)
nnx = 10
nelx = nnx-1
Tbc_left = 200(+273)
Tbc_right = 100(+273)
Lx = 10^5
kappa = 10^-6
hx = Lx / nelx
δt = 1 s
tol = 1 (degree)
T (nnx long)
Told (")
A (nnx X nnx)
b (nnx)
M_e (2x2)
Kd_e (2x2)
x (nnx)
Told ← Tinit
loop over pts
S.S. criterion
|| T − Told || < tol
export / write
TEAM FORTRAN
external software
TEAM LUCKY (PYTHON)
"Free solver"
nnx points
nelx elements
T(t=0)
200°C
100°C
Lx = 100 km
Tbc_left = 200°C
Tbc_right =
T(x,t)
time = 0
Fill TOLD
loop time do i = 1, nstep
time = time + δt
loop elts = 1, nelx
- apply b.c.
- SOLVE SYSTEM → T
else TOLD = T
"once in a while" write solution out

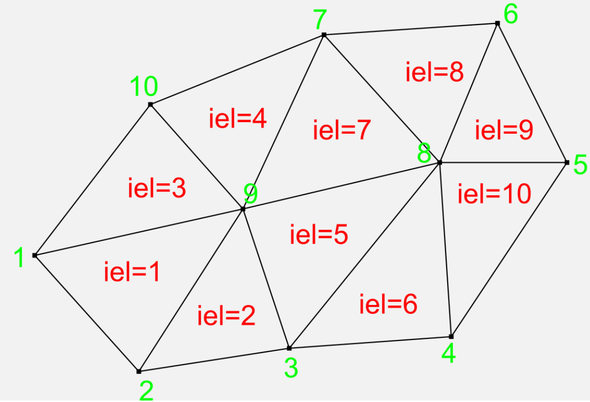
- Typically one uses a connectivity array
- Two-dimensional integer array
- icon (# elements , # vertices per element)

Example



- The above mesh counts 5 elements.
- Each element is composed of 2 nodes

icon(1,1)=1
icon(1,2)=2
icon(2,1)=2
icon(2,2)=3
icon(3,1)=3
icon(3,2)=4
icon(4,1)=4
icon(4,2)=5
icon(5,1)=5
icon(5,2)=6



icon(1,1)=1
icon(1,2)=2
icon(1,3)=9
...
icon(5,1)=3
icon(5,2)=8
icon(5,3)=9
...
icon(10,1)=4
icon(10,2)=5
icon(10,3)=8

- The above mesh counts 10 elements.
- Each element is composed of 3 nodes

initialisation & setup
mesh domain (fill icon array)

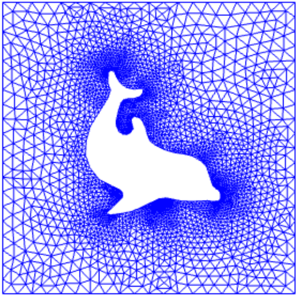
timestepping loop

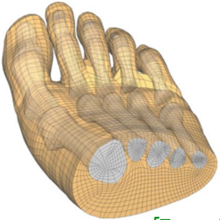
do iel=1,nel

loop over elements

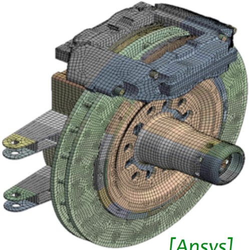
- use icon array to retrieve node # which make up iel
- compute M^e and K_d^e
- build A^e and b^e
- assemble in A and b

apply b.c.
solve





[Truegrid]



[Ansys]

FEM in 1D

FEM in 1D