Numerical modeling of rock deformation:

11 FEM 2D numerical integration & isoparametric elements

Stefan Schmalholz
schmalholz@erdw.ethz.ch
NO E61

AS2009, Thursday 10-12, NO D 11

Repetition

- In the last lectures we learned the basic principles how to transform a system of partial differential equations (PDE's) into a system of linear equations using the finite element method (FEM).
- In the FEM, the PDE's are approximated as linear equations on the element level, i.e. a local stiffness matrix, **K**, and a local right hand side vector, **F**, are generated.
- The local matrix **K** and vector **F** are assembled into a global matrix **Kg** and global vector **Fg** which form the linear system of equations describing the physical process in the model domain, i.e. **Kg u** = **Fg**. The vector **u** is unknown and can be determined with **Kg** and **Fg**.
- Boundary and initial conditions are implemented separately.
- The calculation of the components of **K** and **F** requires performing an integral over the element and performing derivatives of the element shape functions. These integrals can be done analytically for simple and constant element geometries, what we have done so far. If we deform the elements we have to apply numerical integration and we have to correct the derivatives due to the deformation. How to do this is the topic of this lecture.

4 node element

$$\frac{\partial T(x,y,t)}{\partial t} - \kappa\left(\frac{\partial^2 T(x,y,t)}{\partial x^2} + \frac{\partial^2 T(x,y,t)}{\partial y^2}\right) + s = 0$$

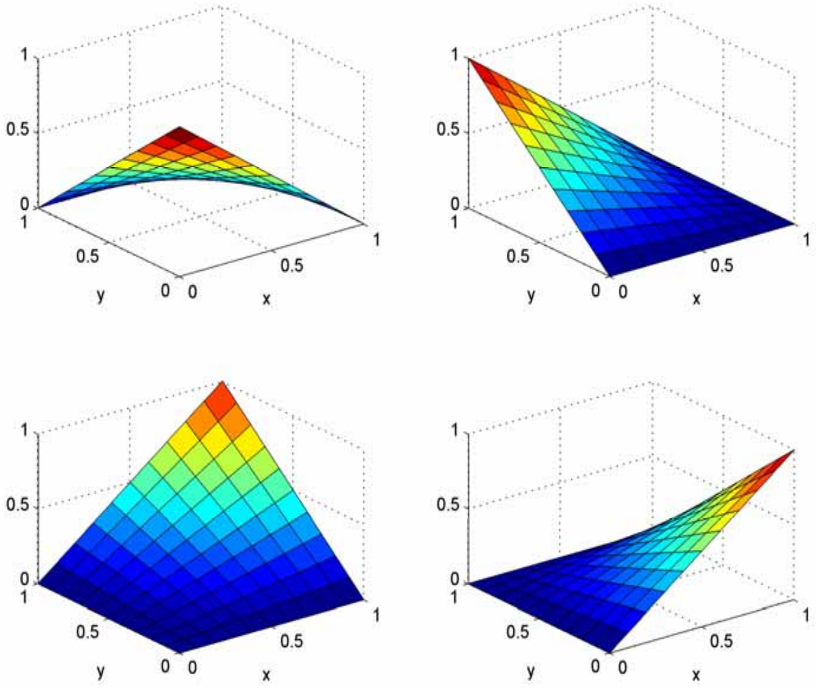
$$T(x,y) \approx \left\{ N(x,y)_1 \quad N(x,y)_2 \quad N(x,y)_3 \quad N(x,y)_4 \right\} \begin{Bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \end{Bmatrix} = \mathbf{N}^{\mathbf{T}}\mathbf{T}$$

$$N(x,y)_1 = \frac{(dx-x)(dy-y)}{dxdy}$$

$$N(x,y)_2 = \frac{x(dy-y)}{dxdy}$$

$$N(x,y)_3 = \frac{xy}{dxdy}$$

$$N(x,y)_4 = \frac{(dx-x)y}{dxdy}$$



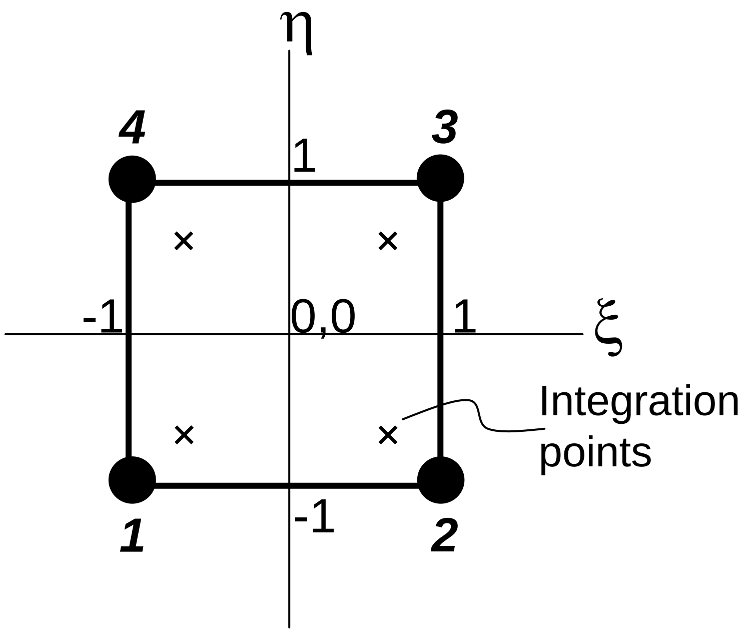
4 node element

$$N(x,y)_1 = \frac{(dx-x)(dy-y)}{dxdy}$$

$$N(x,y)_2 = \frac{x(dy-y)}{dxdy}$$

$$N(x,y)_3 = \frac{xy}{dxdy}$$

$$N(x,y)_4 = \frac{(dx-x)y}{dxdy}$$



$$N(\xi,\eta)_1 = \frac{1}{4}(1-\xi)(1-\eta)$$

$$N(\xi,\eta)_2 = \frac{1}{4}(1+\xi)(1-\eta)$$

$$N(\xi,\eta)_3 = \frac{1}{4}(1+\xi)(1+\eta)$$

$$N(\xi,\eta)_4 = \frac{1}{4}(1-\xi)(1+\eta)$$

Repetition

The 2D temperature diffusion equation is:

$$\frac{\partial T(x,y,t)}{\partial t} - \kappa\left(\frac{\partial^2 T(x,y,t)}{\partial x^2} + \frac{\partial^2 T(x,y,t)}{\partial y^2}\right) - s = 0$$

The corresponding weak form is:

$$\iint \mathbf{N}\mathbf{N}^T dxdy \frac{\partial \mathbf{T}}{\partial t} + \iint \kappa\left(\frac{\partial \mathbf{N}}{\partial x}\frac{\partial \mathbf{N}^T}{\partial x} + \frac{\partial \mathbf{N}}{\partial y}\frac{\partial \mathbf{N}^T}{\partial y}\right) dxdy\mathbf{T} - \iint \mathbf{N} s dxdy = 0$$

The general matrix form is:

$$\mathbf{M}\frac{\partial}{\partial t}\mathbf{T} + \mathbf{K}\mathbf{T} - \mathbf{F} = 0$$

$$\mathbf{M}\left(\frac{\mathbf{T} - \mathbf{T}^{old}}{dt}\right) + \mathbf{K}\mathbf{T} - \mathbf{F} = 0$$

$$\left(\mathbf{M} + dt\mathbf{K}\right)\mathbf{T} = dt\mathbf{F} + \mathbf{T}^{old}$$

$$\mathbf{K}_{tot}\mathbf{T} = \mathbf{F}_{tot}$$

Repetition

Until now we did the integrations analytically with Maple and copy-pasted the result matrices into our Matlab script.

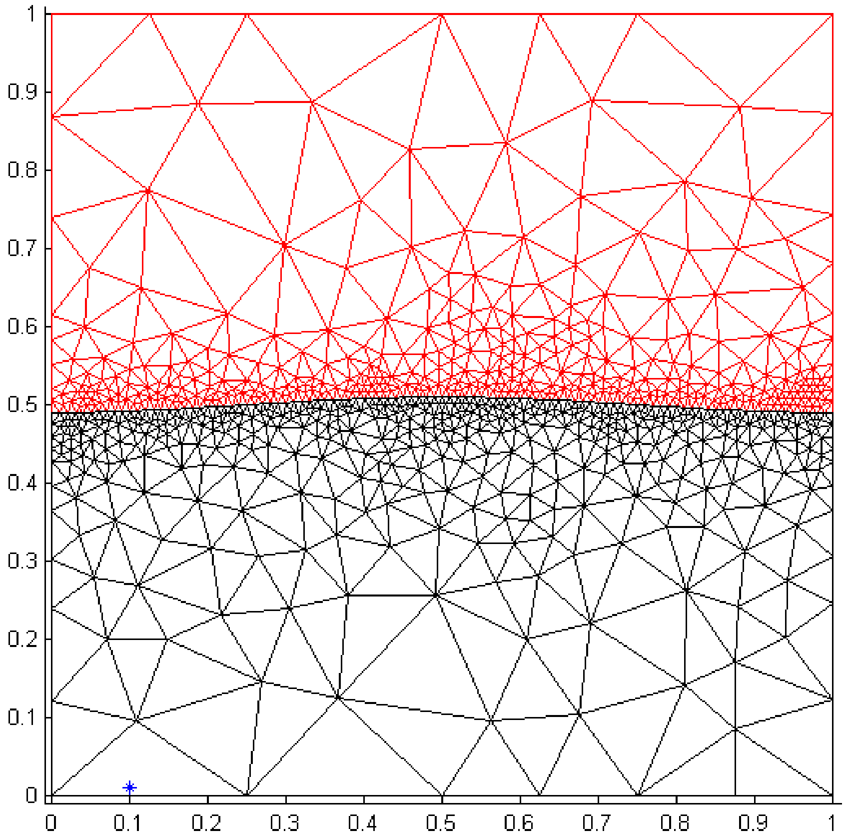
```
for i from 1 to nonel do
Eq_weak[i]  :=int(int(kappa*(diff(T,x)*diff(N[i],x)+diff(T,y)*diff(N[i],y))+Q*N[i],x=0..dx),y=0..dy):
Eq_weak_t[i]:=int(int( (Tdot)*N[i] ,x=0..dx),y=0..dy):
od:
```

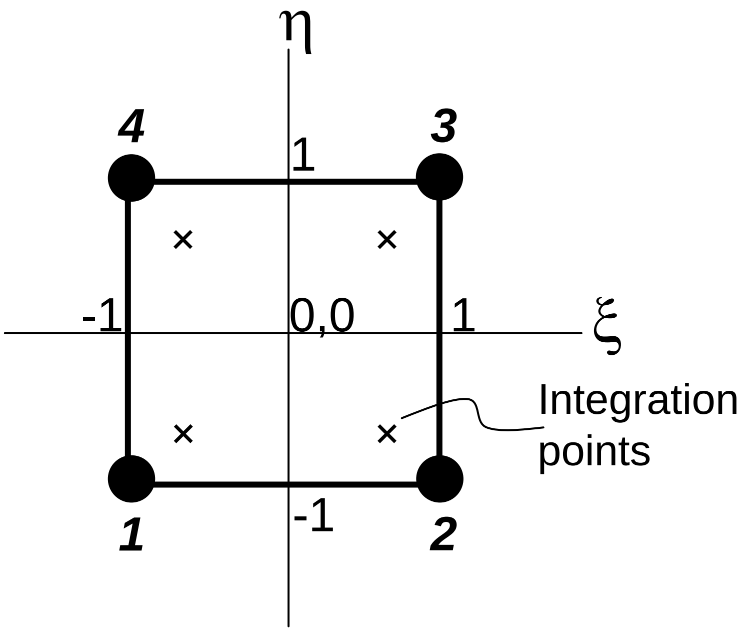
$$\begin{bmatrix} -\frac{2\,\kappa\,dy}{3\,dx}-\frac{2\,\kappa\,dx}{3\,dy}+\kappa\left(\frac{1}{dx^2}+\frac{1}{dy^2}\right)dx\,dy & -\frac{\kappa\,dy}{3\,dx}+\frac{\kappa\,dx}{6\,dy} & -\frac{\kappa\,dy}{6\,dx}-\frac{\kappa\,dx}{6\,dy} & \frac{\kappa\,dy}{6\,dx}-\frac{\kappa\,dx}{3\,dy} \\ -\frac{\kappa\,dy}{3\,dx}+\frac{\kappa\,dx}{6\,dy} & \frac{\kappa\,dy}{3\,dx}+\frac{\kappa\,dx}{3\,dy} & \frac{\kappa\,dy}{6\,dx}-\frac{\kappa\,dx}{3\,dy} & -\frac{\kappa\,dy}{6\,dx}-\frac{\kappa\,dx}{6\,dy} \\ -\frac{\kappa\,dy}{6\,dx}-\frac{\kappa\,dx}{6\,dy} & \frac{\kappa\,dy}{6\,dx}-\frac{\kappa\,dx}{3\,dy} & \frac{\kappa\,dy}{3\,dx}+\frac{\kappa\,dx}{3\,dy} & -\frac{\kappa\,dy}{3\,dx}+\frac{\kappa\,dx}{6\,dy} \\ \frac{\kappa\,dy}{6\,dx}-\frac{\kappa\,dx}{3\,dy} & -\frac{\kappa\,dy}{6\,dx}-\frac{\kappa\,dx}{6\,dy} & -\frac{\kappa\,dy}{3\,dx}+\frac{\kappa\,dx}{6\,dy} & \frac{\kappa\,dy}{3\,dx}+\frac{\kappa\,dx}{3\,dy} \end{bmatrix}$$

```
A =[-2/3*kappa/dx*dy-2/3*kappa*dx/dy+kappa*(1/(dx^2)+1/(dy^2))*dx*dy -1/3*kappa/dx*dy+1/6*kappa*dx/dy -1/6*kappa/dx*dy-1/6*kappa*dx/dy 1/6*kappa/dx*dy-1/3*kappa*dx/dy;...
    -1/3*kappa/dx*dy+1/6*kappa*dx/dy 1/3*kappa/dx*dy+1/3*kappa*dx/dy 1/6*kappa/dx*dy-1/3*kappa*dx/dy -1/6*kappa/dx*dy-1/6*kappa*dx/dy;...
    -1/6*kappa/dx*dy-1/6*kappa*dx/dy 1/6*kappa/dx*dy-1/3*kappa*dx/dy 1/3*kappa/dx*dy+1/3*kappa*dx/dy -1/3*kappa/dx*dy+1/6*kappa*dx/dy;...
    1/6*kappa/dx*dy-1/3*kappa*dx/dy -1/6*kappa/dx*dy-1/6*kappa*dx/dy -1/3*kappa/dx*dy+1/6*kappa*dx/dy 1/3*kappa/dx*dy+1/3*kappa*dx/dy];
```

Motivation

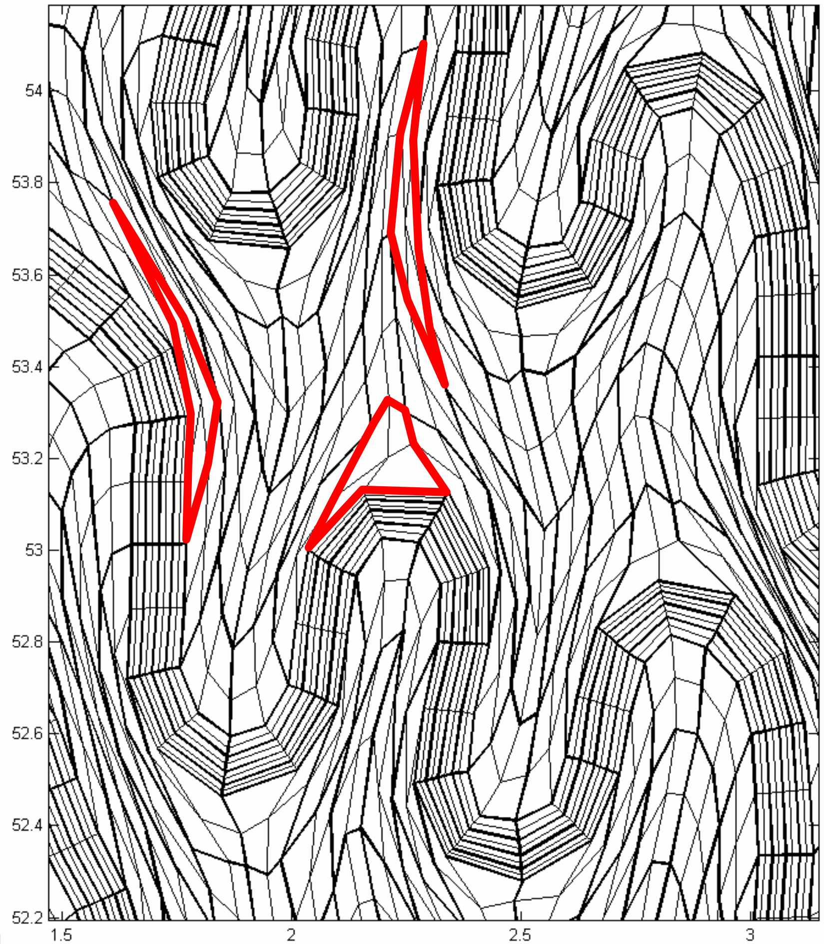
→ Deformable Grid

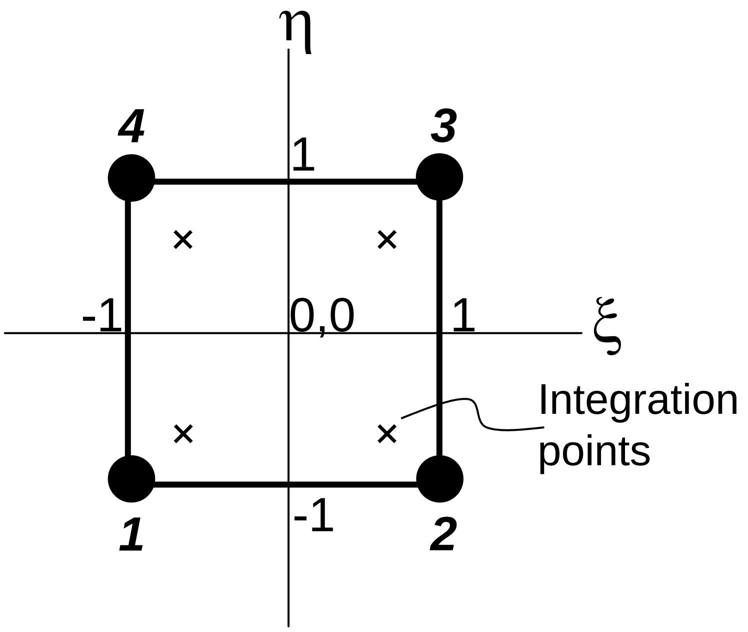




Motivation

→ Deformable Grid





Motivation

- Integral form of system of equations

$$\underbrace{\int_0^{dx}\begin{bmatrix}\frac{\partial N_i(x)}{\partial x}\frac{\partial N_i(x)}{\partial x} & \frac{\partial N_i(x)}{\partial x}\frac{\partial N_{i+1}(x)}{\partial x}\\ \frac{\partial N_{i+1}(x)}{\partial x}\frac{\partial N_i(x)}{\partial x} & \frac{\partial N_{i+1}(x)}{\partial x}\frac{\partial N_{i+1}(x)}{\partial x}\end{bmatrix}Adx}_{\mathbf{K}}\underbrace{\begin{Bmatrix}u_i\\u_{i+1}\end{Bmatrix}}_{\mathbf{u}}-\underbrace{\int_0^{dx}\begin{Bmatrix}N_i(x)\\N_{i+1}(x)\end{Bmatrix}Bdx}_{\mathbf{F}}=0$$

$$\mathbf{K}\quad\mathbf{u}-\mathbf{F}=\mathbf{0}$$

- How do we solve these integrals on a distorted element?
- NUMERICALLY !!!

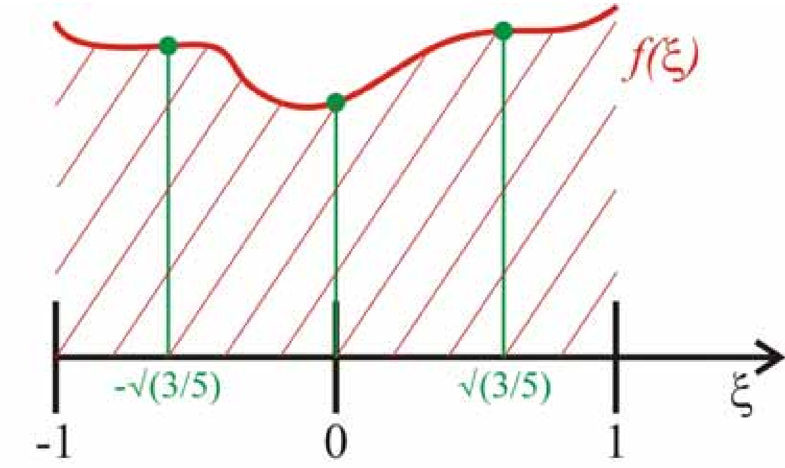
Gauss-Legendre-Quadrature

$\longrightarrow$ General comments

- Numerical integration with Gauss-Legendre-Quadrature only works on an idealized Element
  - For x = -1 to 1 in 1D
  - For x = -1 to 1 and y = -1 to 1 in 2D

Gauss-Legendre-Quadrature

→ For 1D

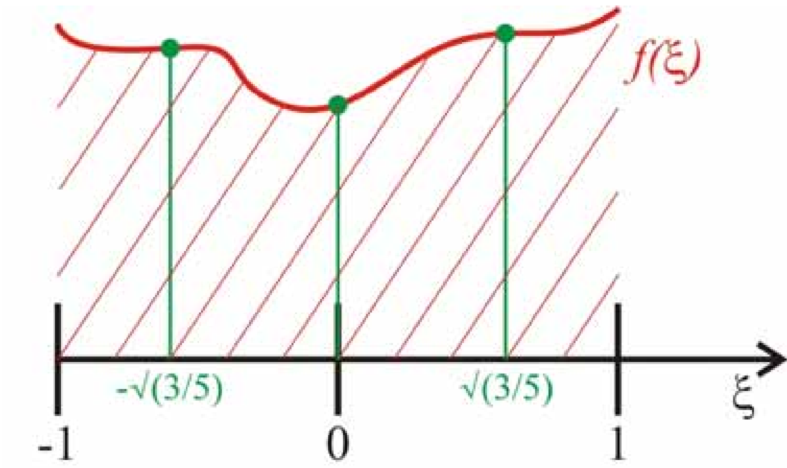


- The function f only needs to be known at the integration points.

Gauss-Legendre-Quadrature

$\longrightarrow$ For 1D

- Formula $$\boxed{\int_{-1}^{1} f(\xi)d\xi = \sum_{n=1}^{nip} f(\xi_n)w_n}$$



- The function f only needs to be known at the integration points.

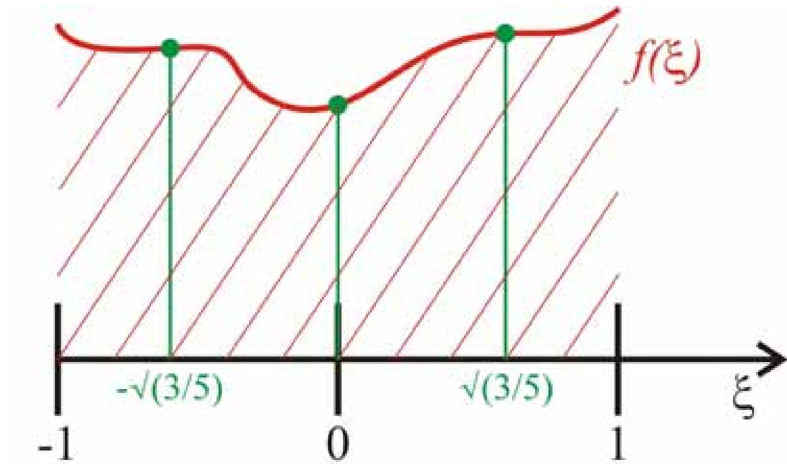
Gauss-Legendre-Quadrature

$\longrightarrow$ For 1D

- Formula $$\boxed{\int_{-1}^{1} f(\xi)d\xi = \sum_{n=1}^{nip} f(\xi_n)w_n}$$

- Integration points and weights

n	ξ_n	w_n
1	0	2
2	$\pm\sqrt{1/3}$	1
3	$-\sqrt{3/5}$, 0 , $\sqrt{3/5}$	$5/9$, $8/9$, $5/9$



- The function f only needs to be known at the integration points.

Gauss-Legendre-Quadrature

$\longrightarrow$ Exercise

- Integrate by hand and “numerically” with 3 integration points: $\int_{-1}^{1} \xi^2 d\xi$

Gauss-Legendre-Quadrature

$\longrightarrow$ Exercise

- Integrate by hand and "numerically" with 3 integration points: $$\int_{-1}^{1} \xi^2 d\xi$$

- By hand: $$\int_{-1}^{1} \xi^2 d\xi = \frac{1}{3}\xi^3\Big|_{-1}^{1} = \frac{1}{3}\left(1^3 - \left(-1\right)^3\right) = \frac{1}{3}(1+1) = \frac{2}{3}$$

Gauss-Legendre-Quadrature

$\longrightarrow$ Exercise

- Integrate by hand and "numerically" with 3 integration points: $$\int_{-1}^{1} \xi^2 d\xi$$

- By hand: $$\int_{-1}^{1} \xi^2 d\xi = \frac{1}{3}\xi^3\Big|_{-1}^{1} = \frac{1}{3}\left(1^3 - \left(-1\right)^3\right) = \frac{1}{3}\left(1+1\right) = \frac{2}{3}$$

- Numerically: $$\int_{-1}^{1} \xi^2 d\xi = \sum_{i=1}^{3} \xi_i^2 w_i = \left(-\sqrt{\frac{3}{5}}\right)^2 \frac{5}{9} + 0^2 \frac{8}{9} + \left(\sqrt{\frac{3}{5}}\right)^2 \frac{5}{9} = \frac{3}{5}\frac{5}{9} + \frac{3}{5}\frac{5}{9} = \frac{6}{9} = \frac{2}{3}$$

Gauss-Legendre-Quadrature

$\longrightarrow$ For 2D

- Formula
$$\boxed{\int_{-1}^{1}\int_{-1}^{1} f(\xi,\eta)\,d\xi\,d\eta = \sum_{i=1}^{n\xi}\sum_{j=1}^{n\eta} f\left(\xi_i,\eta_j\right)w_i w_j = \sum_{n=1}^{nip} f\left(\xi_n,\eta_n\right)w_n}$$

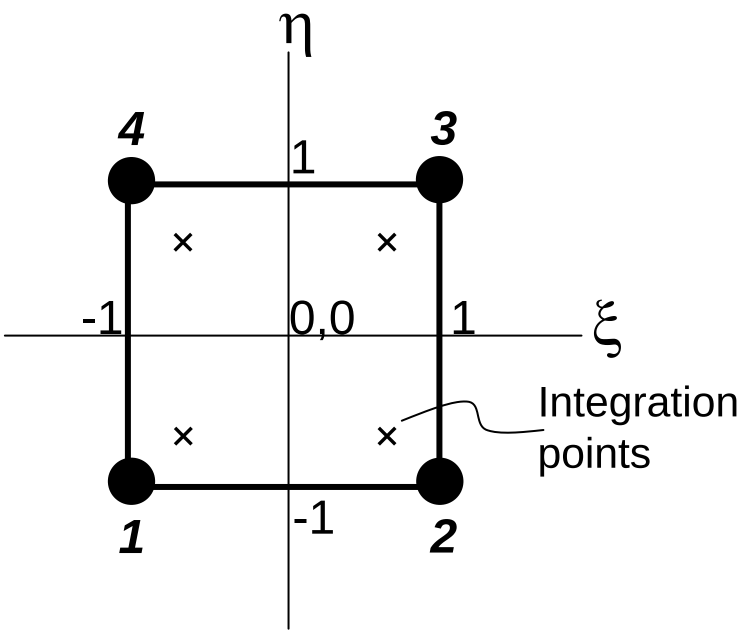
Gauss-Legendre-Quadrature

$\longrightarrow$ For 2D

- Formula $$\boxed{\int_{-1}^{1}\int_{-1}^{1} f(\xi,\eta)d\xi d\eta = \sum_{i=1}^{n\xi}\sum_{j=1}^{n\eta} f\left(\xi_i,\eta_j\right)w_i w_j = \sum_{n=1}^{nip} f\left(\xi_n,\eta_n\right)w_n}$$

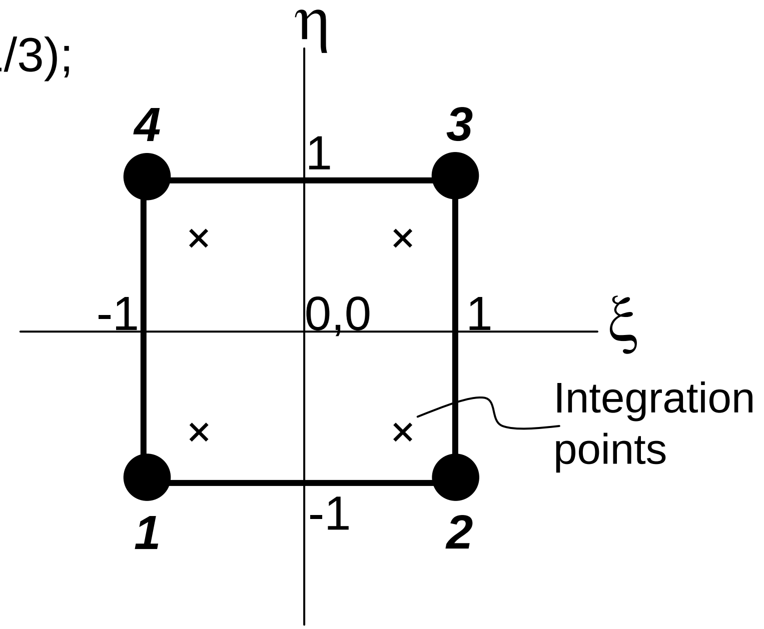
- Integration points are similar to the 1D case. Weights can be defined as a multiplicative combination of the 1D case
  - eg.

$$\eta = \pm\sqrt{1/3}$$
$$\xi = \pm\sqrt{1/3}$$
$$w = 1$$



Define coordinates of integration points

```
% local coordinates
points(1,1)   = -sqrt(1/3);     points(1,2)   = -sqrt(1/3);
points(2,1)   = sqrt(1/3);      points(2,2)   = -sqrt(1/3);
points(3,1)   = sqrt(1/3);      points(3,2)   = sqrt(1/3);
points(4,1)   = -sqrt(1/3);     points(4,2)   = sqrt(1/3);

weight        = [1*1 1*1 1*1 1*1];
```



Numerical integration

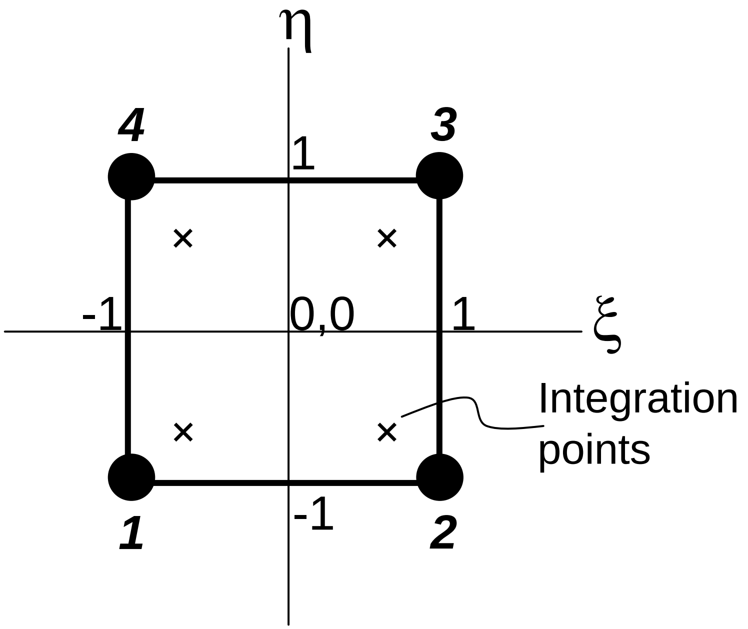
$$\iint \mathbf{N}\mathbf{N}^T dxdy \frac{\partial \mathbf{T}}{\partial t} + \iint \kappa \left(\frac{\partial \mathbf{N}}{\partial x} \frac{\partial \mathbf{N}^T}{\partial x} + \frac{\partial \mathbf{N}}{\partial y} \frac{\partial \mathbf{N}^T}{\partial y} \right) dxdy\mathbf{T} - \iint \mathbf{N} s dxdy = 0$$

Numerical integration requires shape functions and their derivatives at the integration points.

$$N(\xi,\eta)_1 = \frac{1}{4}(1-\xi)(1-\eta)$$

$$\frac{\partial N(\xi,\eta)_1}{\partial \xi} = \frac{1}{4}(-1+\eta)$$

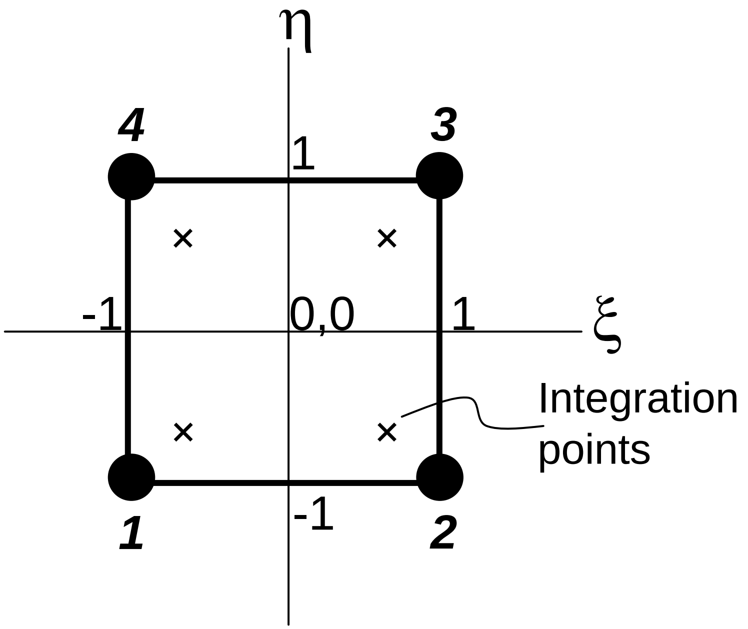
$$\frac{\partial N(\xi,\eta)_1}{\partial \eta} = \frac{1}{4}(-1+\xi)$$



Define shape functions and their derivatives at integration points

$$\iint \mathbf{N}\mathbf{N}^T dxdy \frac{\partial \mathbf{T}}{\partial t} + \iint \kappa \left(\frac{\partial \mathbf{N}}{\partial x}\frac{\partial \mathbf{N}^T}{\partial x} + \frac{\partial \mathbf{N}}{\partial y}\frac{\partial \mathbf{N}^T}{\partial y} \right) dxdy\mathbf{T} - \iint \mathbf{N} s dxdy = 0$$

```
% shape functions
N(:,1)       = 1/4*(1-points(:,1)).*(1-points(:,2));
N(:,2)       = 1/4*(1+points(:,1)).*(1-points(:,2));
N(:,3)       = 1/4*(1+points(:,1)).*(1+points(:,2));
N(:,4)       = 1/4*(1-points(:,1)).*(1+points(:,2));

% derivatives of shape functions
DN_BIG = [];
for i = 1:no_intp
    DN(1,1)     = 1/4*(-1 +points(i,2));
    DN(2,1)     = 1/4*(-1 +points(i,1));
    DN(1,2)     = 1/4*( 1 -points(i,2));
    DN(2,2)     = 1/4*(-1 -points(i,1));
    DN(1,3)     = 1/4*( 1 +points(i,2));
    DN(2,3)     = 1/4*( 1 +points(i,1));
    DN(1,4)     = 1/4*(-1 -points(i,2));
    DN(2,4)     = 1/4*( 1 -points(i,1));

    DN_BIG(:,:,i)      = DN;
end
```



Matlab code

```
for iel = 1:no_el                                    % loop over elements and fill global matrices
    NODES   = g_num(:,iel);                          % extract nodes of element
    COORD   = GLOB_COORD(:,NODES);                   % calculate the coordinates of the nodes

    % initialize local matrices
    MM      = zeros(dof_perel,dof_perel);
    KM      = zeros(dof_perel,dof_perel);
    F       = zeros(dof_perel,1);                    % element vector

    for i = 1:no_intp                                % loop over integration points of element
      jacobi      = DN_BIG(:,:,i)*COORD';            % Jacobi matrix, used for coordinate transformation
      invjacobi   = inv(jacobi);                     % inverse of Jacobi
      detjacobi   = det(jacobi);                     % determinante of Jacobi
      DN_GLOBAL   = invjacobi*DN_BIG(:,:,i);         % derivatives of shape functions wrt global coordinates

      % integrate locally, calculate element matrices
      MM = MM + N(i,:)'*N(i,:)*weight(i)*detjacobi;

      KM = KM + DN_GLOBAL'*KAPPA*DN_GLOBAL*weight(i)*detjacobi;

      KL = MM/dt + KM;                               % element stiffness matrix, left hand side
      KR = MM/dt;                                    % element stiffness matrix, right hand side

      F(:) = F(:) + N(:,i)*s*weight(i)*detjacobi;
    end
    % sum to global matrices
    KL_G(g_num(:,iel),g_num(:,iel))     = KL_G(g_num(:,iel),g_num(:,iel)) + KL;
    KR_G(g_num(:,iel),g_num(:,iel))     = KR_G(g_num(:,iel),g_num(:,iel)) + KR;
    F_G(g_num(:,iel))                   = F_G(g_num(:,iel)) + F;
  end
```

$$\boxed{\int_{-1}^{1}\int_{-1}^{1} f(\xi,\eta)\,d\xi\,d\eta = \sum_{i=1}^{n\xi}\sum_{j=1}^{n\eta} f(\xi_i,\eta_j)\,w_i w_j = \sum_{n=1}^{nip} f(\xi_n,\eta_n)\,w_n}$$

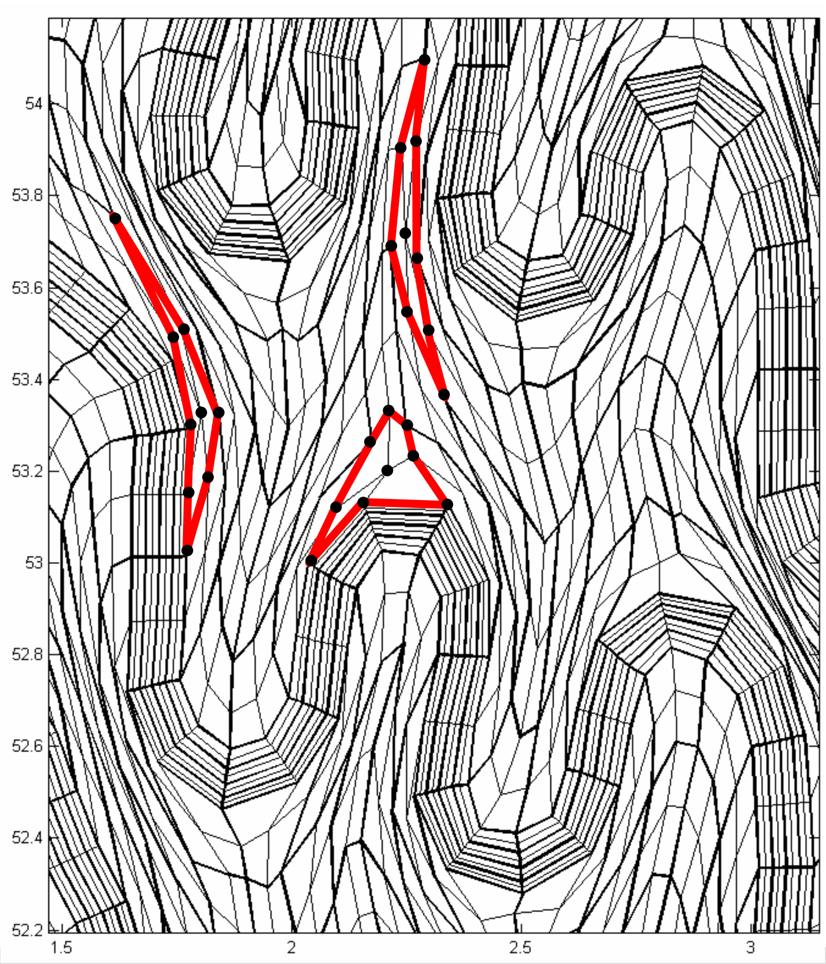
Motivation

$\longrightarrow$ Gauss-Legendre-Quadrature

- Numerical integration with Gauss-Legendre-Quadrature only works on an idealized Element
  - For x = -1 to 1 in 1D
  - For x = -1 to 1 and y = -1 to 1 in 2D
- So, it does not solve the problem of the distorted elements, yet.
- A coordinate transformation from the distorted element to the idealized element is needed in addition.
  See next section.

Isoparametric element

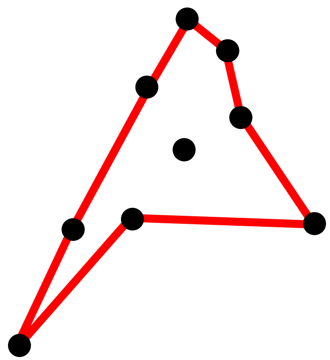
Distorted vs. idealized element



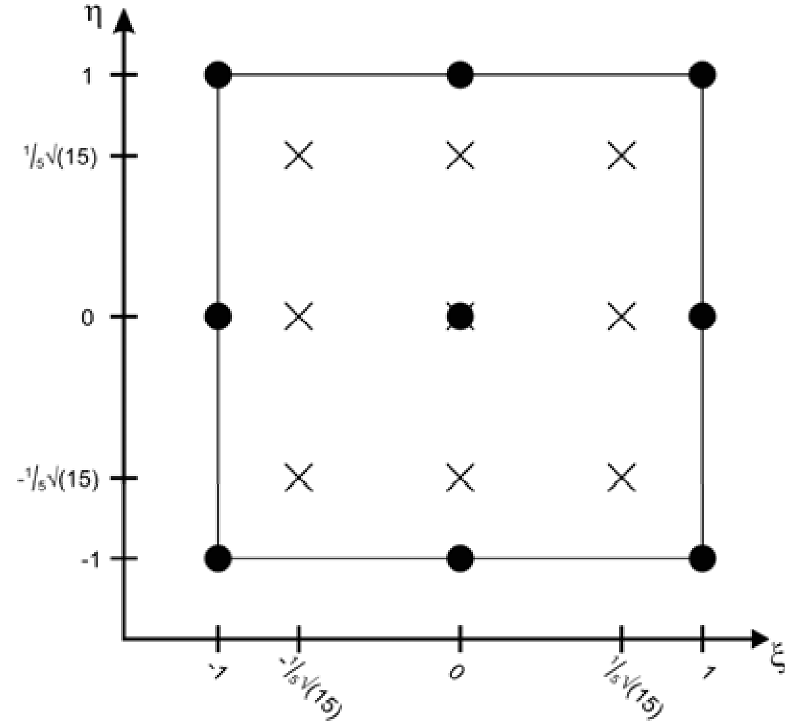
Distorted vs. idealized element

$$\int_0^{dx} \begin{bmatrix} \frac{\partial N_i(x)}{\partial x}\frac{\partial N_i(x)}{\partial x} & \frac{\partial N_i(x)}{\partial x}\frac{\partial N_{i+1}(x)}{\partial x} \\ \frac{\partial N_{i+1}(x)}{\partial x}\frac{\partial N_i(x)}{\partial x} & \frac{\partial N_{i+1}(x)}{\partial x}\frac{\partial N_{i+1}(x)}{\partial x} \end{bmatrix} A dx \begin{Bmatrix} u_i \\ u_{i+1} \end{Bmatrix} - \int_0^{dx} \begin{Bmatrix} N_i(x) \\ N_{i+1}(x) \end{Bmatrix} B dx = 0$$

1D: FEM introduction



2D



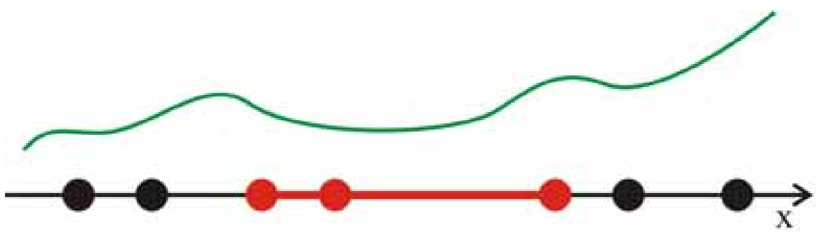
- Derivatives of shape functions with respect to global coordinates
- Integral form written in terms of global coordinates (dx)
- Shape functions given in terms of local coordinates (ξ)
- Numerical integration only possible on a local coordinate system.

Two transformations are necessary

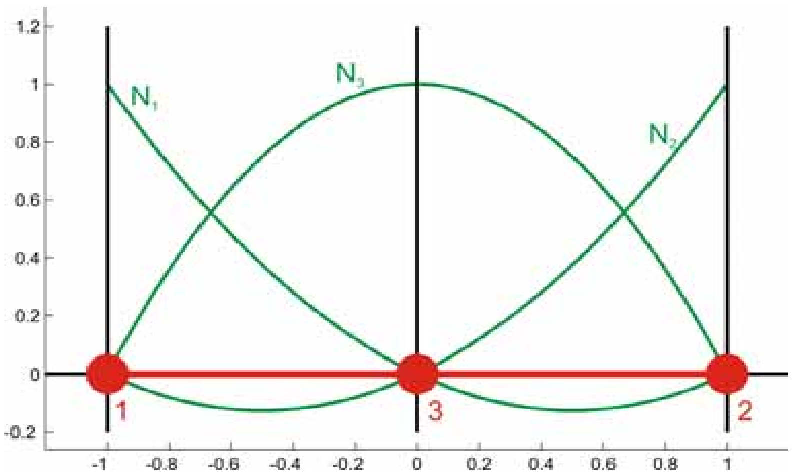
- Transform locally defined derivatives of shape functions to global coordinate system
- Transform locally performed (numerical) integration to global coordinates

First transformation in 1D
$\longrightarrow$ Derivatives of shape fcts. from local to global

- Global distorted element
  - Coordinate x arbitrary
  - Derivatives of shape functions wanted here



- Local isoparametric element
  - Coordinate ξ from -1 to 1
  - Shape functions defined here
  - Derivatives of shape functions determinable here



First transformation in 1D
$\longrightarrow$ Derivatives of shape fcts. from local to global

- Global distorted element
  - Coordinate x arbitrary
  - Derivatives of shape functions wanted here

- Local isoparametric element
  - Coordinate ξ from -1 to 1
  - Shape functions defined here
  - Derivatives of shape functions determinable here

$$\mathbf{N}(\xi) = \begin{Bmatrix} N_1(\xi) \\ N_2(\xi) \\ N_3(\xi) \end{Bmatrix} \qquad \frac{\partial \mathbf{N}(\xi)}{\partial \xi} = \begin{Bmatrix} \partial N_1(\xi) / \partial \xi \\ \partial N_2(\xi) / \partial \xi \\ \partial N_3(\xi) / \partial \xi \end{Bmatrix}$$

First transformation in 1D
$\longrightarrow$ Derivatives of shape fcts. from local to global

- Global distorted element
  - Coordinate x arbitrary
  - Derivatives of shape functions wanted here

- Definition of Jacobian

$$\frac{\partial}{\partial \xi} = \frac{\partial x}{\partial \xi}\frac{\partial}{\partial x} = \mathbf{J}\frac{\partial}{\partial x}$$

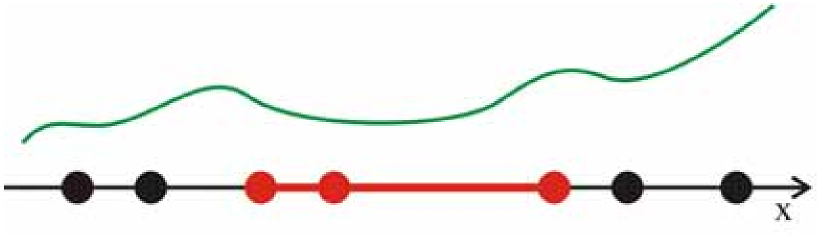
$$\boxed{\frac{\partial}{\partial x} = \mathbf{J}^{-1}\frac{\partial}{\partial \xi}}$$

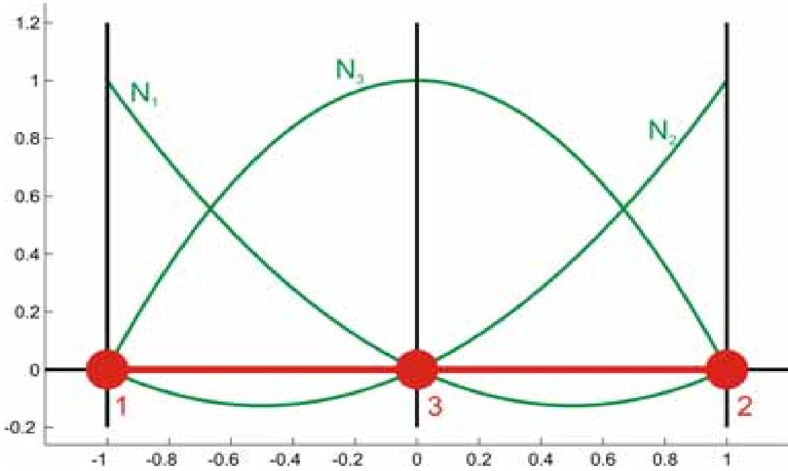
- Local isoparametric element
  - Coordinate ξ from -1 to 1
  - Shape functions defined here
  - Derivatives of shape functions determinable here

$$\mathbf{N}(\xi) = \begin{Bmatrix} N_1(\xi) \\ N_2(\xi) \\ N_3(\xi) \end{Bmatrix} \qquad \frac{\partial \mathbf{N}(\xi)}{\partial \xi} = \begin{Bmatrix} \partial N_1(\xi) / \partial \xi \\ \partial N_2(\xi) / \partial \xi \\ \partial N_3(\xi) / \partial \xi \end{Bmatrix}$$

First transformation in 1D
$\longrightarrow$ Derivatives of shape fcts. from local to global

- How to derive the Jacobian in a FEM manner?
  - Definition $\mathbf{J} = \dfrac{\partial x}{\partial \xi}$
  - With the FEM approximation quantities can be interpolated from nodal points to every point in the element. Why not also the coordinates?





First transformation in 1D
$\longrightarrow$ Derivatives of shape fcts. from local to global

- How to derive the Jacobian in a FEM manner?
  - Definition $\mathbf{J} = \dfrac{\partial x}{\partial \xi}$
  - With the FEM approximation quantities can be interpolated from nodal points to every point in the element. Why not also the coordinates?
  - So $$\boxed{\mathbf{J} = \frac{\partial x}{\partial \xi} = \frac{\partial \mathbf{N}^T}{\partial \xi}\mathbf{x} = \left\{\frac{\partial N_1}{\partial \xi}, \frac{\partial N_2}{\partial \xi}, \frac{\partial N_3}{\partial \xi}\right\}\begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix}}$$

Isoparametric element

Matlab code

```
for iel = 1:no_el                                       % loop over elements and fill global matrices
    NODES   = g_num(:,iel);                             % extract nodes of element
    COORD   = GLOB_COORD(:,NODES);                      % calculate the coordinates of the nodes

    % initialize local matrices
    MM      = zeros(dof_perel,dof_perel);
    KM      = zeros(dof_perel,dof_perel);
    F       = zeros(dof_perel,1);

    for i = 1:no_intp
        jacobi      = DN_BIG(:,:,i)*COORD';
        invjacobi   = inv(jacobi);
        detjacobi   = det(jacobi);
        DN_GLOBAL   = invjacobi*DN_BIG(:,:,i);

        % integrate locally, calculate element matrices
        MM = MM + N(i,:)'*N(i,:)*weight(i)*detjacobi;

        KM = KM + DN_GLOBAL'*KAPPA*DN_GLOB        cobi;

        KL = MM/dt + KM;                                % element stiffness matrix, left hand side
        KR = MM/dt;                                     % element stiffness matrix, right hand side

        F(:) = F(:) + N(:,i)*s*weight(i)*detjacobi;
    end
    % sum to global matrices
    KL_G(g_num(:,iel),g_num(:,iel))     = KL_G(g_num(:,iel),g_num(:,iel)) + KL;
    KR_G(g_num(:,iel),g_num(:,iel))     = KR_G(g_num(:,iel),g_num(:,iel)) + KR;
    F_G(g_num(:,iel))                   = F_G(g_num(:,iel)) + F;
end
```

$$\boxed{\mathbf{J} = \frac{\partial x}{\partial \xi} = \frac{\partial \mathbf{N}^T}{\partial \xi}\mathbf{x} = \left\{\frac{\partial N_1}{\partial \xi}, \frac{\partial N_2}{\partial \xi}, \frac{\partial N_3}{\partial \xi}\right\}\left\{\begin{matrix} x_1 \\ x_2 \\ x_3 \end{matrix}\right\}}$$

$$\boxed{\frac{\partial}{\partial x} = \mathbf{J}^{-1}\frac{\partial}{\partial \xi}}$$

Second transformation in 1D

$\longrightarrow$ Integration form from local to global

- Global distorted element
  - Coordinate x arbitrary
  - Integral form of system of equations given here

- Local isoparametric element
  - Coordinate ξ from -1 to 1
  - Numerical integration performed here

Second transformation in 1D

$\longrightarrow$ Integration form from local to global

- Global distorted element
  - Coordinate x arbitrary
  - Integral form of system of equations given here

- Local isoparametric element
  - Coordinate ξ from -1 to 1
  - Numerical integration performed here

- Transformation of integration boundaries from local to global coordinates

$$\boxed{\int_{x_1}^{x_2} f\left(x\right)dx = \int_{-1}^{1} f\left(\xi\right)\det\left(\mathbf{J}\right)d\xi}$$

$$\mathbf{J} = \frac{\partial x}{\partial \xi}$$

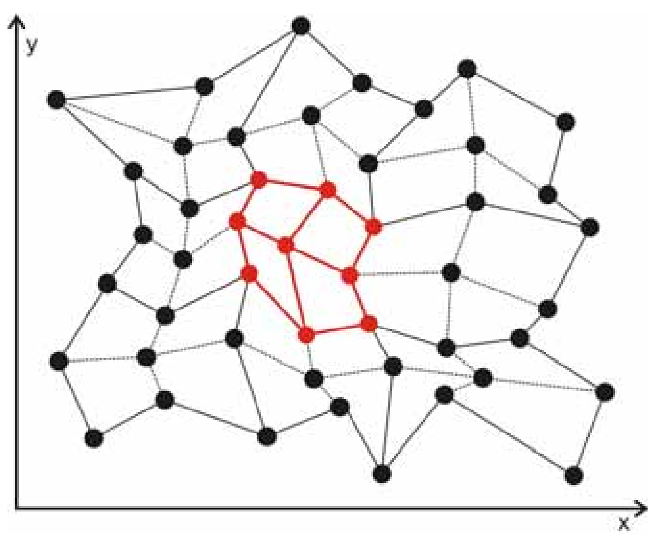
Matlab code

```
for iel = 1:no_el                                          % loop over elements and fill global matrices
      NODES    = g_num(:,iel);                              % extract nodes of element
      COORD    = GLOB_COORD(:,NODES);                       % calculate the coordinates of the nodes

      % initialize local matrices
      MM       = zeros(dof_perel,dof_perel);
      KM       = zeros(dof_perel,dof_perel);
      F        = zeros(dof_perel,1);                        % element vector

      for i = 1:no_intp                                     % loop over integration points of element
         jacobi       = DN_BIG(:,:,i)*COORD';               % Jacobi matrix, used for coordinate transformation
         invjacobi    = inv(jacobi);                        % inverse of Jacobi
         detjacobi    = det(jacobi);                        % determinante of Jacobi
         DN_GLOBAL    = invjacobi*DN_BIG(:,:,i);            % derivatives of shape functions wrt global coordinates

         % integrate locally, calculate element matrices
         MM = MM + N(i,:)'*N(i,:)*weight(i)*detjacobi;

         KM = KM + DN_GLOBAL'*KAPPA*DN_GLOBAL*weight(i)*detjacobi;

         KL = MM/dt + KM;
         KR = MM/dt;

         F(:) = F(:) + N(:,i)*s*weight(i)*detjacobi;
      end
      % sum to global matrices
      KL_G(g_num(:,iel),g_num(:,iel))     = KL_G(g_num(:,iel),g_num(:,iel)) + KL;
      KR_G(g_num(:,iel),g_num(:,iel))     = KR_G(g_num(:,iel),g_num(:,iel)) + KR;
      F_G(g_num(:,iel))                   = F_G(g_num(:,iel)) + F;
   end
```

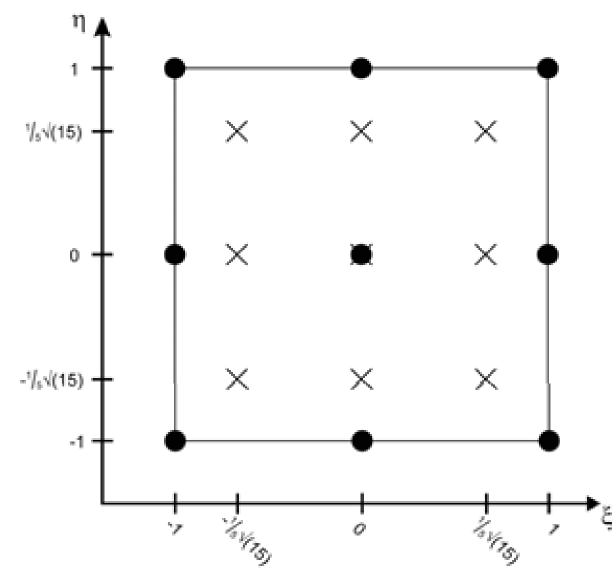
$$\boxed{\int_{x_1}^{x_2} f(x)\,dx = \int_{-1}^{1} f(\xi)\det(\mathbf{J})\,d\xi}$$

First transformation in 2D
$\longrightarrow$ Derivatives of shape fcts. from local to global

- Global distorted element
  - Coordinate x and y arbitrary
  - Derivatives of shape functions wanted here



- Local isoparametric element
  - Coordinate ξ and η from -1 to 1
  - Shape functions and their derivatives defined here



First transformation in 2D
$\longrightarrow$ Derivatives of shape fcts. from local to global

- Global distorted element
  - Coordinate x and y arbitrary
  - Derivatives of shape functions wanted here

- Local isoparametric element
  - Coordinate ξ and η from -1 to 1
  - Shape functions and their derivatives defined here

$$\mathbf{N}(\xi,\eta)=\begin{Bmatrix} N_1(\xi,\eta) \\ N_2(\xi,\eta) \\ \dots \\ N_9(\xi,\eta) \end{Bmatrix}$$

$$\nabla_{\xi,\eta}\mathbf{N}(\xi,\eta)=\begin{bmatrix} \dfrac{\partial N_1(\xi,\eta)}{\partial \xi} & \dfrac{\partial N_1(\xi,\eta)}{\partial \eta} \\ \dfrac{\partial N_2(\xi,\eta)}{\partial \xi} & \dfrac{\partial N_2(\xi,\eta)}{\partial \eta} \\ \dots & \dots \\ \dfrac{\partial N_9(\xi,\eta)}{\partial \xi} & \dfrac{\partial N_9(\xi,\eta)}{\partial \eta} \end{bmatrix}$$

First transformation in 2D $\longrightarrow$ Derivatives of shape fcts. from local to global

- Global distorted element
  - Coordinate x and y arbitrary
  - Derivatives of shape functions wanted here

- Definition of Jacobian

$$\begin{Bmatrix} \partial/\partial\xi \\ \partial/\partial\eta \end{Bmatrix} = \begin{bmatrix} \partial x/\partial\xi & \partial y/\partial\xi \\ \partial y/\partial\eta & \partial x/\partial\eta \end{bmatrix} \begin{Bmatrix} \partial/\partial x \\ \partial/\partial y \end{Bmatrix} = \mathbf{J} \begin{Bmatrix} \partial/\partial x \\ \partial/\partial y \end{Bmatrix}$$

$$\boxed{\begin{Bmatrix} \partial/\partial x \\ \partial/\partial y \end{Bmatrix} = \mathbf{J}^{-1} \begin{Bmatrix} \partial/\partial\xi \\ \partial/\partial\eta \end{Bmatrix}}$$

- Local isoparametric element
  - Coordinate ξ and η from -1 to 1
  - Shape functions and their derivatives defined here

$$\mathbf{N}(\xi,\eta) = \begin{Bmatrix} N_1(\xi,\eta) \\ N_2(\xi,\eta) \\ \dots \\ N_9(\xi,\eta) \end{Bmatrix}$$

$$\nabla_{\xi,\eta}\mathbf{N}(\xi,\eta) = \begin{bmatrix} \dfrac{\partial N_1(\xi,\eta)}{\partial\xi} & \dfrac{\partial N_1(\xi,\eta)}{\partial\eta} \\ \dfrac{\partial N_2(\xi,\eta)}{\partial\xi} & \dfrac{\partial N_2(\xi,\eta)}{\partial\eta} \\ \dots & \dots \\ \dfrac{\partial N_9(\xi,\eta)}{\partial\xi} & \dfrac{\partial N_9(\xi,\eta)}{\partial\eta} \end{bmatrix}$$

First transformation in 2D
$\longrightarrow$ Derivatives of shape fcts. from local to global

- Derivation of the Jacobian in a FEM manner!
  - Definition $\mathbf{J} = \begin{bmatrix} \partial x/\partial \xi & \partial y/\partial \xi \\ \partial y/\partial \eta & \partial x/\partial \eta \end{bmatrix}$
  - So

$$\boxed{\mathbf{J} = \begin{bmatrix} \partial x/\partial \xi & \partial y/\partial \xi \\ \partial y/\partial \eta & \partial x/\partial \eta \end{bmatrix} = \nabla_{\xi,\eta}\mathbf{N}^T(\xi,\eta)\mathbf{x} = \begin{bmatrix} \dfrac{\partial N_1(\xi,\eta)}{\partial \xi} & \dfrac{\partial N_2(\xi,\eta)}{\partial \xi} & \cdots & \dfrac{\partial N_9(\xi,\eta)}{\partial \xi} \\ \dfrac{\partial N_1(\xi,\eta)}{\partial \eta} & \dfrac{\partial N_2(\xi,\eta)}{\partial \eta} & \cdots & \dfrac{\partial N_9(\xi,\eta)}{\partial \eta} \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ \cdots \\ x_9 \end{Bmatrix}}$$

Second transformation in 2D

$\longrightarrow$ Integration form from local to global

- Global distorted element
  - Coordinate x and y arbitrary
  - Integral form of system of equations given here

- Local isoparametric element
  - Coordinate ξ and η from -1 to 1
  - Numerical integration performed here

Second transformation in 2D

$\longrightarrow$ Integration form from local to global

- Global distorted element
  - Coordinate x and y arbitrary
  - Integral form of system of equations given here

- Local isoparametric element
  - Coordinate ξ and η from -1 to 1
  - Numerical integration performed here

- Transformation of integration boundaries from local to global coordinates

$$\boxed{\int_{x_1}^{x_2}\int_{y_1}^{y_2} f(x,y)\,dx\,dy = \int_{-1}^{1}\int_{-1}^{1} f(\xi,\eta)\det(\mathbf{J})\,d\xi\,d\eta}$$

$$\mathbf{J} = \begin{bmatrix} \partial x/\partial\xi & \partial y/\partial\xi \\ \partial y/\partial\eta & \partial x/\partial\eta \end{bmatrix}$$

Matlab code

```
for iel = 1:no_el                                         % loop over elements and fill global matrices
      NODES   = g_num(:,iel);                             % extract nodes of element
      COORD   = GLOB_COORD(:,NODES);                      % calculate the coordinates of the nodes

      % initialize local matrices
      MM      = zeros(dof_perel,dof_perel);
      KM      = zeros(dof_perel,dof_perel);
      F       = zeros(dof_perel,1);                       % element vector

      for i = 1:no_intp                                   % loop over integration points of element
         jacobi      = DN_BIG(:,:,i)*COORD';              % Jacobi matrix, used for coordinate transformation
         invjacobi   = inv(jacobi);                       % inverse of Jacobi
         detjacobi   = det(jacobi);                       % determinante of Jacobi
         DN_GLOBAL   = invjacobi*DN_BIG(:,:,i);           % derivatives of shape functions wrt global coordinates

         % integrate locally, calculate element matrices
         MM = MM + N(i,:)'*N(i,:)*weight(i)*detjacobi;

         KM = KM + DN_GLOBAL'*KAPPA*DN_GLOBAL*weight(i)*detjacobi;

         KL = MM/dt + KM;                                 % element stiffness matrix, left hand side
         KR = MM/dt;                                      % element stiffness matrix, right hand side

         F(:) = F(:) + N(:,i)*s*weight(i)*detjacobi;
      end
      % sum to global matrices
      KL_G(g_num(:,iel),g_num(:,iel))     = KL_G(g_num(:,iel),g_num
      KR_G(g_num(:,iel),g_num(:,iel))     = KR_G(g_num(:,iel),g_num
      F_G(g_num(:,iel))                   = F_G(g_num(:,iel)) + F;
   end
```

$$\int_{x_1}^{x_2}\int_{y_1}^{y_2} f(x,y)\,dx\,dy = \int_{-1}^{1}\int_{-1}^{1} f(\xi,\eta)\det(\mathbf{J})\,d\xi\,d\eta$$

$$\int_{-1}^{1}\int_{-1}^{1} f(\xi,\eta)\,d\xi\,d\eta = \sum_{i=1}^{n\xi}\sum_{j=1}^{n\eta} f(\xi_i,\eta_j)\,w_i w_j = \sum_{n=1}^{nip} f(\xi_n,\eta_n)\,w_n$$