Plane Strain Linear FE Theory

By Jorgen Bergstrom, Ph.D.

Principle of Virtual Work:

$$\int_V \sigma_{ij}\delta\varepsilon_{ij}dV = \int_V b_i \delta u_i dV + \int_A t_i \delta u_i dA + \sum \delta u_i f_i$$

where ***b*** is the body force, ***t*** is the surface traction, $\delta\mathbf{u}$ is a virtual displacement field, ***f*** are discrete point forces. The virtual strain is given by:

$$\delta\varepsilon_{ij} = \frac{1}{2}\left[\frac{\partial \delta u_i}{\partial x_j} + \frac{\partial \delta u_j}{\partial x_i}\right] = \frac{1}{2}\left(\partial u_{i,j} + \partial u_{j,i}\right) = \begin{bmatrix}\delta\varepsilon_{11}\\ \delta\varepsilon_{22}\\ \delta\varepsilon_{12}\end{bmatrix}$$

This equation is valid for all materials.

Note: The principle of virtual work can be proven from the divergence theorem:

$$\int_V \frac{\partial f}{\partial x_i} dV = \int_A f n_i dA$$

and the equilibrium equation: $\sigma_{ij,j} + b_i = 0$

Discretize the Body into Finite Elements *m*

Discretize the virtual work equation into finite elements:

$$\sum_m \int_{V^{(m)}} \sigma_{ij}\delta\varepsilon_{ij}\, dV = \sum_m \int_{V^{(m)}} b_i \delta u_i\, dV + \sum_m \int_{A^{(m)}} t_i \delta u_i\, dA + \sum \delta u_i f_i$$

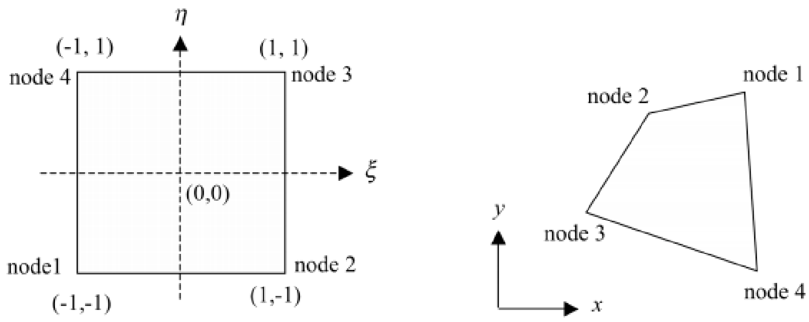
Written in vector notation:

$$\sum_m \int_{V^{(m)}} \delta\boldsymbol{\varepsilon}^\top \boldsymbol{\sigma}\, dV = \sum_m \int_{V^{(m)}} \delta\mathbf{u}^\top \mathbf{b}\, dV + \sum_m \int_{A^{(m)}} \delta\mathbf{u}^\top \mathbf{t}\, dA + \sum \delta\mathbf{u}^\top \mathbf{f}$$

The element integrals can be quickly calculated using isoparametric shape functions. Since we are only concerned with small deformations there is only one coordinate system.

Isoparametric Shape Functions

Consider a plane-strain quadrilateral element. Map the quadrilateral to a square with coordinates (ξ,η):



The position of a point (x,y) inside the quadrilateral can be determined from the corner nodes and (ξ, η)

$$x = N_1(\xi,\eta)x_1 + N_2(\xi,\eta)x_2 + N_3(\xi,\eta)x_3 + N_4(\xi,\eta)x_4$$
$$y = N_1(\xi,\eta)y_1 + N_2(\xi,\eta)y_2 + N_3(\xi,\eta)y_3 + N_4(\xi,\eta)y_4$$

The displacements (*u*,*v*) can be obtained from the same shape function:

$$u = N_1(\xi,\eta)u_1 + N_2(\xi,\eta)u_2 + N_3(\xi,\eta)u_3 + N_4(\xi,\eta)u_4$$
$$v = N_1(\xi,\eta)v_1 + N_2(\xi,\eta)v_2 + N_3(\xi,\eta)v_3 + N_4(\xi,\eta)v_4$$

The shape functions are given by:

$$N_1 = 0.25(1-\xi)(1-\eta)$$
$$N_2 = 0.25(1+\xi)(1-\eta)$$
$$N_3 = 0.25(1+\xi)(1+\eta)$$
$$N_4 = 0.25(1-\xi)(1+\eta)$$

Define:

$$\mathbf{N}(\xi,\eta) \equiv \begin{bmatrix} N_1(\xi,\eta) & N_2(\xi,\eta) & N_3(\xi,\eta) & N_4(\xi,\eta) \end{bmatrix} \qquad [1x4]$$

$$d\mathbf{N}(\xi,\eta) \equiv \begin{bmatrix} dN_1/d\xi & dN_2/d\xi & dN_3/d\xi & dN_4/d\xi \\ dN_1/d\eta & dN_2/d\eta & dN_3/d\eta & dN_4/d\eta \end{bmatrix} \qquad [2x4]$$

In Python (with (xi,eta) as natural coordinates):

```
N = 0.25*array([(1-xi)*(1-eta), (1+xi)*(1-eta),
                (1+xi)*(1+eta), (1-xi)*(1+eta)])
dN = 0.25*array([[eta-1, 1-eta, 1+eta, -1-eta],
                 [xi-1, -1-xi, 1+xi, 1-xi]])
```

Let **U** be the displacements of the four corner nodes:

$$\mathbf{U} = [u_1, v_1, u_2, v_2, u_3, v_3, u_4, v_4]^\top \qquad [8x1]$$

Define **H** as:

$$\mathbf{H}(\xi,\eta) = \begin{bmatrix} N_1 & 0 & N_2 & 0 & N_3 & 0 & N_4 & 0 \\ 0 & N_1 & 0 & N_2 & 0 & N_3 & 0 & N_4 \end{bmatrix} \qquad [2x8]$$

Then the displacements of any point inside the element is given by:

$$\mathbf{u} = \begin{bmatrix} u \\ v \end{bmatrix} = \mathbf{HU}$$

[u: 2x1] = [H: 2x8] * [U: 8x1]

Strains from Nodal Displacements

To calculate the strains we need to know the partial derivatives of the displacements with respect to the original coordinates (*x*,*y*).

The gradient of the u-displacement in the natural plane is:

$$\begin{bmatrix} \partial u/\partial \xi \\ \partial u/\partial \eta \end{bmatrix} = d\mathbf{N} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} \dfrac{dN_i}{d\xi} u_i \\ \dfrac{dN_i}{d\eta} u_i \end{bmatrix}$$

[2x1] = [2x4]*[4x1]

To determine the displacement gradient in the real coordinate system consider:

$$\frac{\partial u}{\partial \xi} = \frac{\partial u}{\partial x}\frac{\partial x}{\partial \xi} + \frac{\partial u}{\partial y}\frac{\partial y}{\partial \xi}$$

$$\frac{\partial u}{\partial \eta} = \frac{\partial u}{\partial x}\frac{\partial x}{\partial \eta} + \frac{\partial u}{\partial y}\frac{\partial y}{\partial \eta}$$

which can be written:

$$\begin{bmatrix} \partial u/\partial \xi \\ \partial u/\partial \eta \end{bmatrix} = \begin{bmatrix} \partial x/\partial \xi & \partial y/\partial \xi \\ \partial x/\partial \eta & \partial y/\partial \eta \end{bmatrix} \begin{bmatrix} \partial u/\partial x \\ \partial u/\partial y \end{bmatrix} \equiv \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix} \begin{bmatrix} \partial u/\partial x \\ \partial u/\partial y \end{bmatrix}$$

The Jacobian terms are given by:

$$J_{11} = \frac{\partial x}{\partial \xi} \qquad J_{12} = \frac{\partial y}{\partial \xi} \qquad J_{21} = \frac{\partial x}{\partial \eta} \qquad J_{22} = \frac{\partial y}{\partial \eta}$$

The Jacobian matrix is determined by the nodal coordinates and the point (ξ, η)

If we let **x** be the nodal coordinates:

$$\mathbf{x} = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ x_3 & y_3 \\ x_4 & y_4 \end{bmatrix}$$

[4x2]

then the J matrix can be calculated from:

$$\mathbf{J} = d\mathbf{N}\,\mathbf{x}$$

[2x2] = [2x4]*[4x2]

The determinant of the **J** matrix is:

$$J \equiv \det \mathbf{J} = J_{11}J_{22} - J_{12}J_{21}$$

The inverse of the Jacobian matrix is:

$$\mathbf{J}^{-1} = \frac{1}{J}\begin{bmatrix} J_{22} & -J_{12} \\ -J_{21} & J_{11} \end{bmatrix}$$

Hence, the partial derivatives with respect to the original coordinates (*x*,*y*) can be written:

$$\begin{bmatrix} \partial u/\partial x \\ \partial u/\partial y \end{bmatrix} = \mathbf{J}^{-1} \begin{bmatrix} \partial u/\partial \xi \\ \partial u/\partial \eta \end{bmatrix}, \qquad \begin{bmatrix} \partial v/\partial x \\ \partial v/\partial y \end{bmatrix} = \mathbf{J}^{-1} \begin{bmatrix} \partial v/\partial \xi \\ \partial v/\partial \eta \end{bmatrix}$$

The strain can now be obtained from:

$$\varepsilon_{ij} = \begin{bmatrix} \varepsilon_{11} \\ \varepsilon_{22} \\ \varepsilon_{12} \end{bmatrix} = \begin{bmatrix} \partial u/\partial x \\ \partial v/\partial y \\ \frac{1}{2}\left[\partial u/\partial y + \partial v/\partial x\right] \end{bmatrix}$$

which can also be written:

$$\boldsymbol{\varepsilon} = \mathbf{B}\mathbf{q} = \mathbf{B}^{(m)}\mathbf{U}$$

[3x1]=[3x8]*[8x1]

or expanded:

$$\begin{bmatrix} \varepsilon_{11} \\ \varepsilon_{22} \\ \varepsilon_{12} \end{bmatrix} = \begin{bmatrix} dN_{11} & 0 & dN_{12} & 0 & dN_{13} & 0 & dN_{14} & 0 \\ 0 & dN_{21} & 0 & dN_{22} & 0 & dN_{23} & 0 & dN_{24} \\ dN_{11} & dN_{21} & dN_{12} & dN_{22} & dN_{13} & dN_{23} & dN_{13} & dN_{24} \end{bmatrix} \begin{bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ u_3 \\ v_3 \\ u_4 \\ v_4 \end{bmatrix}$$

The virtual strains can be written: $\delta\boldsymbol{\varepsilon} = \mathbf{B}\delta\mathbf{q}$

Numerical Integration in 2D

Assume 2x2 Gauss points:

$$\int_A f(x,y)dxdy = \int_A f(\xi,\eta)Jd\xi d\eta = \sum_{i=1}^{2}\sum_{j=1}^{2} w_i w_j f(\xi_i,\eta_j)J(\xi_i,\eta_j)$$

where

$$\xi_i = \frac{1}{\sqrt{3}}\begin{bmatrix} -1 & 1 \end{bmatrix} \qquad \eta_i = \frac{1}{\sqrt{3}}\begin{bmatrix} -1 & 1 \end{bmatrix} \qquad w_i = 1$$

Stresses for Element *m* based on Linear Elasticity

$$\boldsymbol{\sigma}^{(m)} = \mathbf{C}^{(m)}\boldsymbol{\varepsilon}^{(m)} = \mathbf{C}^{(m)}\mathbf{B}^{(m)}\mathbf{U}$$

[3x3] * [3x8] * [8x1]

For plane strain, the symmetrical **C** matrix is given by:

$$\sigma_{ij}^{(m)} = \begin{pmatrix} \sigma_{11} \\ \sigma_{22} \\ \sigma_{12} \end{pmatrix} = C_{ij}\begin{pmatrix} \varepsilon_{11} \\ \varepsilon_{22} \\ \varepsilon_{12} \end{pmatrix}$$

$$C_{ij} = \frac{E}{(1+\nu)(1-2\nu)}\begin{pmatrix} 1-\nu & \nu & 0 \\ \nu & 1-\nu & 0 \\ 0 & 0 & 0.5-\nu \end{pmatrix}$$

[3x3]

Back to Principle of Virtual Work

From above:

$$\sum_m \int_{V^{(m)}} \delta\boldsymbol{\varepsilon}^\top \boldsymbol{\sigma}\, dV = \sum_m \int_{V^{(m)}} \delta\mathbf{u}^\top \mathbf{b}\, dV + \sum_m \int_{A^{(m)}} \delta\mathbf{u}^\top \mathbf{t}\, dA + \sum \delta\mathbf{U}^\top \mathbf{f}$$

Integrand 1 (Internal virtual work):

Only consider one element, and recall:

$\delta\boldsymbol{\varepsilon} = \mathbf{B}\,\delta\mathbf{U}$ [strain: 3x1] = [B: 3x8] * [U: 8x1]

$\boldsymbol{\sigma} = \mathbf{CBU}$ [stress: 3x1] = [C: 3x3] * [B: 3x8] * [U: 8x1]

$$\int_V \delta\mathbf{U}^\top\mathbf{B}^\top\mathbf{CB}\,\mathbf{U}\,dV = \delta\mathbf{U}^\top\left[\int_V \mathbf{B}^\top\mathbf{CB}\,dV\right]\mathbf{U} = \delta\mathbf{U}^\top\mathbf{k}^{(m)}\mathbf{U}$$

where the element stiffness matrix is:

$$\mathbf{k}^{(m)} \equiv \int_{V^{(m)}} \mathbf{B}^\top\mathbf{CB}\,dV = \sum_p^{np} \mathbf{B}^\top\mathbf{CB}Jw_p$$

[8x3]*[3x3]*[3x8]

Integrand 2 (External virtual work from body forces):

Only consider one element, and recall: $\delta\mathbf{u} = \mathbf{H}\,\delta\mathbf{U}$ [2x8] * [8x1]

$$\int_V \delta\mathbf{u}^\top\mathbf{b}\,dV = \delta\mathbf{U}^\top\left[\int_V \mathbf{H}^\top\mathbf{b}\,dV\right]$$

[1x8] * [8x2] * [2x1]

Integrand 3 (External virtual work from surface tractions):

Only consider one element, and recall: $\delta\mathbf{u} = \mathbf{H}\,\delta\mathbf{U}$ [2x8] * [8x1]

$$\int_A \delta\mathbf{u}^\top\mathbf{t}\,dA = \delta\mathbf{U}^\top\left[\int_A \mathbf{H}^\top\mathbf{t}\,dA\right]$$

[1x8] * [8x2] * [2x1]

Integrand 4: (External virtual work from point forces):

Only consider one element: $\delta\mathbf{U}^\top\mathbf{f}$ [1x8] * [8x1]

In summary, the principle of virtual work can be written:

$$\left[\sum_m \mathbf{k}^{(m)}\right]\mathbf{U} = \sum_m \int_{V^{(m)}} \mathbf{H}^\top\mathbf{b}\,dV + \sum_m \int_{A^{(m)}} \mathbf{H}^\top\mathbf{t}\,dA + \mathbf{f}$$

Giving: $\mathbf{K}\,\mathbf{U} = \mathbf{R}$, where:

$$\mathbf{K} = \sum_m \mathbf{k}^{(m)} = \sum_m \sum_{p=1}^{np} \mathbf{B}^\top\mathbf{CB}Jw_p$$

$$\mathbf{R} = \sum_m \sum_{p=1}^{np} \mathbf{H}^\top\mathbf{b}Jw_p + \sum_m \sum_{p=1}^{np} \mathbf{H}^\top\mathbf{t}Jw_p + \mathbf{f}$$

Python Code to Assemble the Global Stiffness Matrix:

```
def gradshape(xi):
    """Gradient of the shape functions for a 4-node, isoparametric element.
       dN_i(x,y)/dx and dN_i(x,y)/dy
       Input: 1x2,  Output: 2x4"""
    x,y = tuple(xi)
    dN = [[-(1.0-y),  (1.0-y), (1.0+y), -(1.0+y)],
          [-(1.0-x), -(1.0+x), (1.0+x),  (1.0-x)]]
    return 0.25*array(dN)

K = zeros((2*num_nodes, 2*num_nodes))
B = zeros((3,8))
q4 = array([[-1,-1],[1,-1],[-1,1],[1,1]]) / sqrt(3.0)
# conn[0] is [0, 1, 62, 61]<= node numbers of the element
for c in conn:   # loop through each element
     # coordinates of each node in the element
     # shape = 4x2
     # for the first element:
     #    nodePts = [[0.0,   0.0],
     #               [0.033, 0.0],
     #               [0.033, 0.066],
     #               [0.0,   0.066]]
     nodePts = nodes[c,:]
     Ke = zeros((8,8))   # element stiffness matrix is 8x8
     for q in q4:        # for each Gauss point
          dN = gradshape(q)
          J  = dot(dN, nodePts) # J is 2x2
          dN = dot(inv(J), dN)  # [2x4]
          B[0,0::2] = dN[0,:]   # set to 1st row in dN [1x4]
          B[1,1::2] = dN[1,:]   # set to 2nd row in dN [1x4]
          B[2,0::2] = dN[1,:]
          B[2,1::2] = dN[0,:]
          Ke += dot(dot(B.T,C),B) * det(J)

     # Scatter operation
     for i,I in enumerate(c):
          for j,J in enumerate(c):
               K[2*I,2*J]     += Ke[2*i,2*j]
               K[2*I+1,2*J]   += Ke[2*i+1,2*j]
               K[2*I+1,2*J+1] += Ke[2*i+1,2*j+1]
               K[2*I,2*J+1]   += Ke[2*i,2*j+1]
```

APPENDIX: Stress Equilibrium

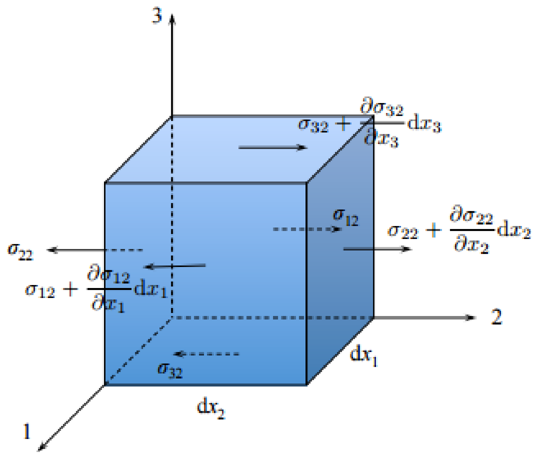


Figure 3.11: All components of the stress tensor contributing to the force equilibrium in x_2 direction must be in equilibrium.

According to Newton's law, the sum of all forces (stress times the surface area) acting along x_2 must be zero

$$\left(\sigma_{22} + \frac{\partial \sigma_{22}}{\partial x_2} \mathrm{d}x_2\right) \mathrm{d}x_1 \mathrm{d}x_3 - \sigma_2 \mathrm{d}x_1 \mathrm{d}x_3 + \left(\sigma_{12} + \frac{\partial \sigma_{12}}{\partial x_1} \mathrm{d}x_1\right) \mathrm{d}x_2 \mathrm{d}x_3 - \sigma_{12} \mathrm{d}x_2 \mathrm{d}x_3 + \left(\sigma_{32} + \frac{\partial \sigma_{32}}{\partial x_3} \mathrm{d}x_3\right) \mathrm{d}x_1 \mathrm{d}x_2 - \sigma_{32} \mathrm{d}x_1 \mathrm{d}x_2 + B_2 \mathrm{d}x_1 \mathrm{d}x_2 \mathrm{d}x_3 = 0 \tag{3.14}$$

For generality, the body force (force per unit volume) was included as well. The body force represent for example gravity force $B = \rho g$ or d'Alambert inertia force $B = \rho \ddot{u}$ so that the derivation is valid both for static and dynamic problems. Summing up the forces one gets the first equilibrium equation

$$\frac{\partial \sigma_{22}}{\partial x_2} + \frac{\partial \sigma_{12}}{\partial x_1} + \frac{\partial \sigma_{32}}{\partial x_3} + B_2 = 0 \tag{3.15}$$