How can we improve our results? Pick a better $u^*$.
- broaden the class of $K$ a.s.d. we choose from

\[ u^*(r) = \alpha_0 + \sum_{k=1}^{N} \alpha_k r^k \]

then \( V[u^*] = V(\alpha_0, \alpha_1, \ldots, \alpha_N) \)

find the \( \alpha_k \) by solving \( \frac{\partial V}{\partial \alpha_k} = 0 \) \((k = 0, 1, \ldots, N)\)

(\(N+1\) equations, \(N+1\) variables)

\( \uparrow \) can find \( \alpha_0, \alpha_1, \alpha_2 \) \( \uparrow \)

\( \uparrow \) messy, but doable in Mathematica

---

**Numerical Approximations Based on Min. Pot. En. Thm.**

**e.g.) Finite Element Method**

\( \uparrow \) a restatement of MPET is the "virtual work" equation, we will write it here for the case of no body forces.

\[ \int_{\mathcal{R}} \sigma_{ij} \varepsilon_{ij} d\mathcal{R} - \int_{\mathcal{T}} T_i S_{ij} dA = 0 \]

What is this equation? Will discuss in detail later.

For now: for boundary value problems solved on a computer, we use this instead of the equilibrium equations.
note: equilibrium eq's vs. Virtual work (MPET) integral equations

numerically it is easier to integrate than to differentiate accurately

How do we use it? If $\sigma_{ij}$ satisfies the Virtual work equation for a R.A.S.D. $u(x)$ and

$$\delta \varepsilon_{ij} = \frac{1}{2} \left( \frac{\partial}{\partial x_j} Su_i + \frac{\partial}{\partial x_i} Su_j \right)$$

then it automatically satisfies equilibrium

In finite elements we solve this equation using a discretized form of the displacement

---

Introduction to the Method

→ linear, elastic materials

→ plane stress

→ small deformation

→ 3 noded triangular elements

→ no index notation

$x_1, x_2 \rightarrow x, y$

$u_1, u_2 \rightarrow u, v$

$\varepsilon_{11}, \varepsilon_{22}, \varepsilon_{12} \rightarrow \varepsilon_x, \varepsilon_y, \varepsilon_{xy}$

$\sigma_{11}, \sigma_{22}, \sigma_{12} \rightarrow \sigma_x, \sigma_y, \sigma_{xy}$

vector of nodal displacements

$$\Delta \alpha = \{ u_1, v_1, u_2, v_2, \ldots \} \rightarrow u_N, v_N$$

$\Delta = \text{nodes}$

---

$\Delta \rightarrow \text{unknown}$
global coordinate table:

<table>
<thead>
<tr>
<th>node</th>
<th>x-coord</th>
<th>y-coord</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>x₁</td>
<td>y₁</td>
</tr>
<tr>
<td>2</td>
<td>x₂</td>
<td>y₂</td>
</tr>
<tr>
<td>3</td>
<td>x₃</td>
<td>y₃</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>N</td>
<td>xₙ</td>
<td>yₙ</td>
</tr>
</tbody>
</table>

global connectivity table:

<table>
<thead>
<tr>
<th>elem</th>
<th>a-coord</th>
<th>b-coord</th>
<th>c-coord</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1 (P₁)</td>
<td>3 (q₁)</td>
<td>2 (r₁)</td>
</tr>
<tr>
<td>2</td>
<td>4 (P₂)</td>
<td>3 (q₂)</td>
<td>1 (r₂)</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>7 (Pₙₑ)</td>
<td>10 (qₙₑ)</td>
<td>6 (rₙₑ)</td>
</tr>
</tbody>
</table>

Now consider a representative element

\[ \mathbf{S}_\alpha = \{ U_A, V_A, U_B, V_B, U_C, V_C \} \]

→ element vector of displacements

MUST express \( u(x,y) \) within \( ABC \)
in terms of \( U_A, U_B, U_C \)

the simplest form that can be determined
uniquely is linear interpolation
Consider \( N_a(x, y) = \frac{(y-y_b)(x_c-x_b)-(x-x_b)(y_c-y_b)}{(y_a-y_b)(x_c-x_b)-(x_a-x_b)(y_c-y_b)} \)

\( N_a(x, y) \) is (i) linear in \( x, y \)
(ii) \( = 0 \) on \( bc \)
(iii) \( = 1 \) on \( a \)

Similarly, we can define \( N_b, N_c \). Then

\[
\begin{align*}
  \{ & u(x, y) = u_a N_a(x, y) + u_b N_b(x, y) + u_c N_c(x, y) \\
  & \text{"shape functions" or "interpolation functions"}
\}
\end{align*}
\]

\[
\begin{align*}
  \{ & v(x, y) = v_a N_a(x, y) + v_b N_b(x, y) + v_c N_c(x, y) \\
  & \text{express this as} \quad u_\alpha = N_{\alpha\beta} S_\beta \quad \rightarrow \quad \alpha=1,2 \quad \beta=1,\ldots,b
\}
\end{align*}
\]

where \( N_{\alpha\beta} = \begin{bmatrix} N_a & o & N_b & o & N_c & o \\
 o & N_a & o & N_b & o & N_c \end{bmatrix} \)

\( u_\alpha \) contains continuous \( x \) and \( y \) displacements within the triangular element, interpolated from nodes of that element.
Now, we also want a vector of strain components in each element.

\[ e_\alpha = \{e_x, e_y, e_{xy}\} \quad \alpha = 1, 2, 3 \quad \text{(within an element)} \]

then \[ e_\alpha = B_{\alpha\beta} s_\beta \]

where:

\[
B_{\alpha\beta} = \begin{bmatrix}
N_{ax} & 0 & N_{bx} & 0 & N_{cx} & 0 \\
0 & N_{ay} & 0 & N_{by} & 0 & N_{cy} \\
\frac{1}{2} N_{ax} & \frac{1}{2} N_{ay} & \frac{1}{2} N_{bx} & \frac{1}{2} N_{by} & \frac{1}{2} N_{cx} & \frac{1}{2} N_{cy} \\
\end{bmatrix}
\]

\[ e_{q)} N_{ax} = \frac{\partial N_a (x,y)}{\partial x} \]

this formulation \[ e_\alpha = B_{\alpha\beta} s_\beta \] is an expression of compatibility \[ e_{ij} = \frac{1}{2} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \]

gradients are taken on the shape functions, not on nodal values.

\[ e_{q)} \alpha = 1: \]

\[ L_\beta \ e_1 = \sum B_{1\beta} \ s_\beta = N_{ax} u_a + N_{bx} u_b + N_{cx} u_c \]

this is \[ e_1 = e_x \]

\[ L_\beta \ aslo \ u_1 = u_a N_a + u_b N_b + u_c N_c \]

so \[ \frac{\partial u_1}{\partial x} = u_a N_{ax} + u_b N_{bx} + u_c N_{cx} \]

Note: all components of the matrix B are constants
(since they are spatial derivatives of linear functions of position)

i.e. \[ \text{so the elements we are using are} \]
"constant strain triangles"