AE731

Theory of Elasticity

Dr. Nicholas Smith
Wichita State University, Department of Aerospace Engineering August 28, 2019

upcoming schedule

- Aug 28 - Tensor Calculus
- Sep 2 - Labor Day
- Sep 4 - Displacement and Strain, Homework 1 Due
- Sep 9 - Strain Transformation
- Sep 11 - Exam 1 Review

outline

- group problems
- review
- tensor algebra
- tensor calculus
- other coordinate systems
- chapter summary

group problems

group 1

- Rotate the following matrix into the principal coordinate system

$$\begin{bmatrix} -1 & 1 & 0 \\ 1 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

group 2

- The x' coordinate system is described by a rotation of 53.13° about the x_2 axis
- If $u_i = \langle 10, 15, 5 \rangle$, find u_i'

group 3

- Compare the invariants of the A_ij and B_{ij}

$$A_{ij} = \begin{bmatrix} -1 & 1 & 0 \\ 1 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$B_{ij} = \begin{bmatrix} 0.28 & 0.60 & -0.96 \\ 0.60 & -1 & 0.80 \\ -0.96 & 0.80 & -0.28 \end{bmatrix}$$

review

tensor transformations

- We can use the direction cosines ($\cos(x_i{}', x_j)$) to express any-order tensor in a transformed coordinate system

$$
\begin{aligned}
a' &= a & \text{zero order, scalar} \\
a'_i &= Q_{ip} a_p & \text{first order, vector} \\
a'_{ij} &= Q_{ip} Q_{jq} a_{pq} & \text{second order, matrix} \\
a'_{ijk} &= Q_{ip} Q_{jq} Q_{kr} a_{pqr} & \text{third order} \\
a'_{ijkl} &= Q_{ip} Q_{jq} Q_{kr} Q_{lo} a_{pqro} & \text{fourth order}
\end{aligned}
$$

- Any tensor will follow these transformation rules

programming with index notation

- Some expressions in index notation can be simply translated to matrix expressions
- Others are either confusing, or use higher-order tensors
- For example, if we rotate the fourth-order stiffness tensor ${C_{ijkl}}' = Q_{ip}Q_{jq}Q_{kr}Q_{lo}C_{pqro}$

programming with index notation

```
for i = 1:3
for j = 1:3
for k = 1:3
for l = 1:3
 C(i,j,k,l) = 0;
 for p = 1:3
 for q = 1:3
 for r = 1:3
 for o = 1:3
  C(i,j,k,l) = C(i,j,k,l) +
Q(i,p)*Q(j,q)*Q(k,r)*Q(l,o)*C(p,q,r,o);
 end; end; end; end;
end; end; end; end;
```

programming

- In general, when programming an expression in index notation there are a few things to be careful about
    1. Your programming language's start index (C and Python start at 0, MATLAB and Fortran start at 1)
    2. Make sure your free indexes are on the outside of the loop, and the dummy indexes are on the inside
    3. Don't forget to sum over the dummy indexes

tensor algebra

dot products

- The dot product (inner product) can be used with any-ordered tensor
- Will reduce the order of the tensor by one
- $a_i b_i = c$
- $A_{ij} B_{jk} = C_{ik}$
- $A_{ij} b_j = c_i$
- $A_{ijk} b_k = C_{ij}$

dot products

- We can have higher-order “dot” products when multiple indexes are repeated
- Double dot product will reduce the order of the tensor by two
- $A_{ij}B_{ij} = c$
- $A_{ijk}B_{jkl} = C_{il}$
- $A_{ijkl}B_{kl} = C_{ij}$

dyadic notation

- There is an antiquated notation that you may encounter reading older papers and texts
- Now known as “dyadic notation” (or sometimes “tensor product notation”)
- Dyadic product: $C_{ij} = a_i b_j$ is written as $C = a \otimes b$
- Double dot product: $A_{ij} B_{ji} = c$ is written as $A : B = c$

kronecker delta

- For convenience we define two symbols in index notation
- *Kronecker delta* is a general tensor form of the Identity Matrix

$$\delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

kronecker delta

- Is also used for higher order tensors
- $\delta_{ij} = \delta_{ji}$
- $\delta_{ii} = 3$
- $\delta_{ij} a_j = a_i$
- $\delta_{ij} a_{ij} = a_{ii}$

permutation symbol

- *alternating symbol* or *permutation symbol*

$$\epsilon_{ijk} = \begin{cases} 1 & \text{if } ijk \text{ is an even permutation of 1,2,3} \\ -1 & \text{if } ijk \text{ is an odd permutation of 1,2,3} \\ 0 & \text{otherwise} \end{cases}$$

permutation symbol

- This symbol is not used as frequently as the *Kronecker delta*
- For our uses in this course, it is enough to know that 123, 231, and 312 are even permutations
- 321, 132, 213 are odd permutations
- all other indexes are zero
- $\epsilon_{ijk}\epsilon_{imn} = \delta_{jm}\delta_{kn} - \delta_{jn}\delta_{mk}$

cross product

- The cross-product can be written in index notation

$$\hat{a} \times \hat{b} = \epsilon_{ijk} a_j b_k \hat{e}_i$$

- The coordinate system unit vectors ($\hat{e}_i$) are often neglected

$$\hat{a} \times \hat{b} = \epsilon_{ijk} a_j b_k$$

converting to matrix math

- It is often convenient to write expressions in matrix notation to use MATLAB or graphing calculators
- We need to be careful how this is done, in index notation left and right multiplication are identical, but this is not the case for matrices
  $[A][B]=A_{ij}B_{jk}$
  $[B][A]=B_{ij}A_{jk} = A_{jk}B_{ij}$

converting to matrix math

- Some useful relations

$$
\begin{aligned}
[B] &= A_{ij}B_{jk} \\
[A][B]^T &= A_{ij}B_{kj} \\
[A]^T[B] &= A_{ji}B_{jk} \\
tr([A][B]) &= A_{ij}B_{ji} \\
tr([A][B]^T) &= A_{ij}B_{ij}
\end{aligned}
$$

converting to matrix

- Sometimes our expression is more complex (involves more terms)
- e.g. transformation of a matrix $a_{ij}{}' = Q_{ip}Q_{jq}a_{pq}$
    1. Re-arrange so dummy indexes are adjacent $Q_{ip}a_{pq}Q_{jq}$
    2. Identify which (if any) tensors are transposed (dummy indexes should be on the inside of adjacent terms without a transpose)

  $Q_{ip}\, a_{\,pq}\, \mathrm{Q}_{\,jq}$

  $[Q][a][Q]^T$

example

- Convert the expression in index notation to Matrix notation
  $A_{ik}B_{jl}C_{ml}D_{mk}$

  1. Re-arrange to so that dummy indexes are in adjacent terms

     $A_{ik}D_{mk}C_{ml}B_{jl}$

  2. Identify which terms are transposed

     $A_{ik}\,\color{red}{D_{mk}}C\,_{ml}\,\color{red}{B_{jl}}$
     $[A][D]^T[C][B]^T$

tensor calculus

partial derivatives

- We usually omit the (x_i), but most variables we deal with are functions of x_i
- These are referred to as field variables. e.g.

$$
\begin{aligned}
a &= a(x_1, x_2, x_3) &&= a(x_i) \\
a_i &= a_i(x_1, x_2, x_3) &&= a_i(x_i) \\
a_{ij} &= a_{ij}(x_1, x_2, x_3) &&= a_{ij}(x_i)
\end{aligned}
$$

partial derivatives

- We can use comma notation to simplify taking partial derivatives of field variables

$$
\begin{aligned}
a_{,i} &= \frac{\partial}{\partial x_i} a \\
a_{i,j} &= \frac{\partial}{\partial x_j} a_i \\
a_{ij,k} &= \frac{\partial}{\partial x_k} a_{ij}
\end{aligned}
$$

partial derivatives

- Free index and dummy index conventions still apply to the comma notation
- $a_{,i}$ expands to
  $$\left\langle \frac{\partial}{\partial x_1} a, \frac{\partial}{\partial x_2} a, \frac{\partial}{\partial x_3} a \right\rangle$$
- But $b_{i,i}$ becomes
  $$\frac{\partial}{\partial x_1} b_1 + \frac{\partial}{\partial x_2} b_2 + \frac{\partial}{\partial x_3} b_3$$

partial derivatives

- And $b_{i,j}$ is

$$\begin{bmatrix} b_{1,1} & b_{1,2} & b_{1,3} \\ b_{2,1} & b_{2,2} & b_{2,3} \\ b_{3,1} & b_{3,2} & b_{3,3} \end{bmatrix}$$

gradient

- The gradient operator, ∇, is often used to indicate partial differentiation in matrix and vector notation
- We can represent ∇ as a vector
  $$\nabla = \left\langle \frac{\partial}{\partial x_1}, \frac{\partial}{\partial x_2}, \frac{\partial}{\partial x_3} \right\rangle$$
- ∇ is also referred to as the *del operator*

gradient

- We can convert between vector notation and index notation for many common operations using the ∇.

$$\nabla\phi = \phi_{,i}$$
$$\nabla^2\phi = \phi_{,ii}$$
$$\nabla\hat{u} = u_{i,j}$$
$$\nabla\cdot\hat{u} = u_{i,i}$$
$$\nabla\times\hat{u} = \epsilon_{ijk}u_{k,j}$$
$$\nabla^2\hat{u} = u_{i,kk}$$

divergence theorem

- The Divergence Theorem (or Gauss Theorem) for a vector field, $\hat{u}$,
  $$\iint_S \hat{u} \cdot \hat{n} dS = \iiint \nabla \cdot \hat{u} dV$$
- is also valid for tensors of any order
  $\iint_S a_{ij\ldots k} n_k dS = \iiint_V a_{ij\ldots k,k} dV$

stokes theorem

- Stokes theorem for a vector field, $\hat{u}$,
  $$\oint \hat{u} \cdot d\hat{r} = \iint_S (\nabla \times \hat{u}) \cdot \hat{n} dS$$
- also applies for tensors of any order
  $\oint a_{ij\ldots k} dx_t = \iint_S \epsilon_{rst} a_{ij\ldots k,s} n_r dS$

green's theorem

- Green's theorem is merely a simplification of Stokes theorem in a planar domain.
- If we write the vector field, $\hat{u} = f\hat{e_1} + g\hat{e_2}$, we find $$\iint_S \left(\frac{\partial g}{\partial x_1} - \frac{\partial f}{\partial x_2} \right) dxdy = \int_C (fdx + gdy)$$

zero-value theorem

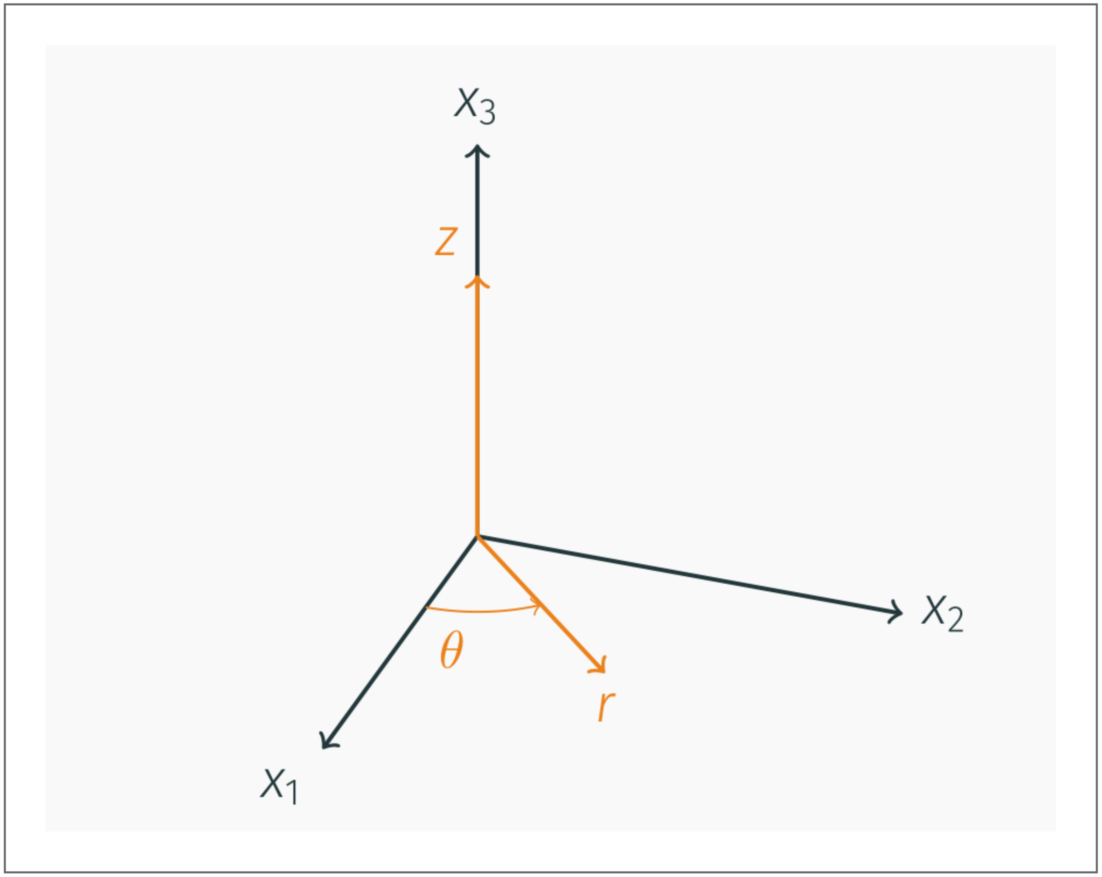
- The zero-value theorem is particularly useful in variational calculus, which we will use later in the course
- If we know that
  $\iiint_V f_{ij...k} dV = 0$
- then
  $f_{ij...k} = 0$

other coordinate systems

curvilinear coordinates

- We discussed coordinate transformations earlier
- However, we often desire to use other coordinate systems entirely
- Polar coordinates (in 2D) are an example of this
- In 3D, we can use cylindrical or spherical coordinates

cylindrical coordinates



cylindrical coordinates

- We can convert between Cartesian and cylindrical coordinate systems

$$
\begin{aligned}
x_1 &= r\cos\theta \\
x_2 &= r\sin\theta \\
x_3 &= z
\end{aligned}
$$

cylindrical coordinates

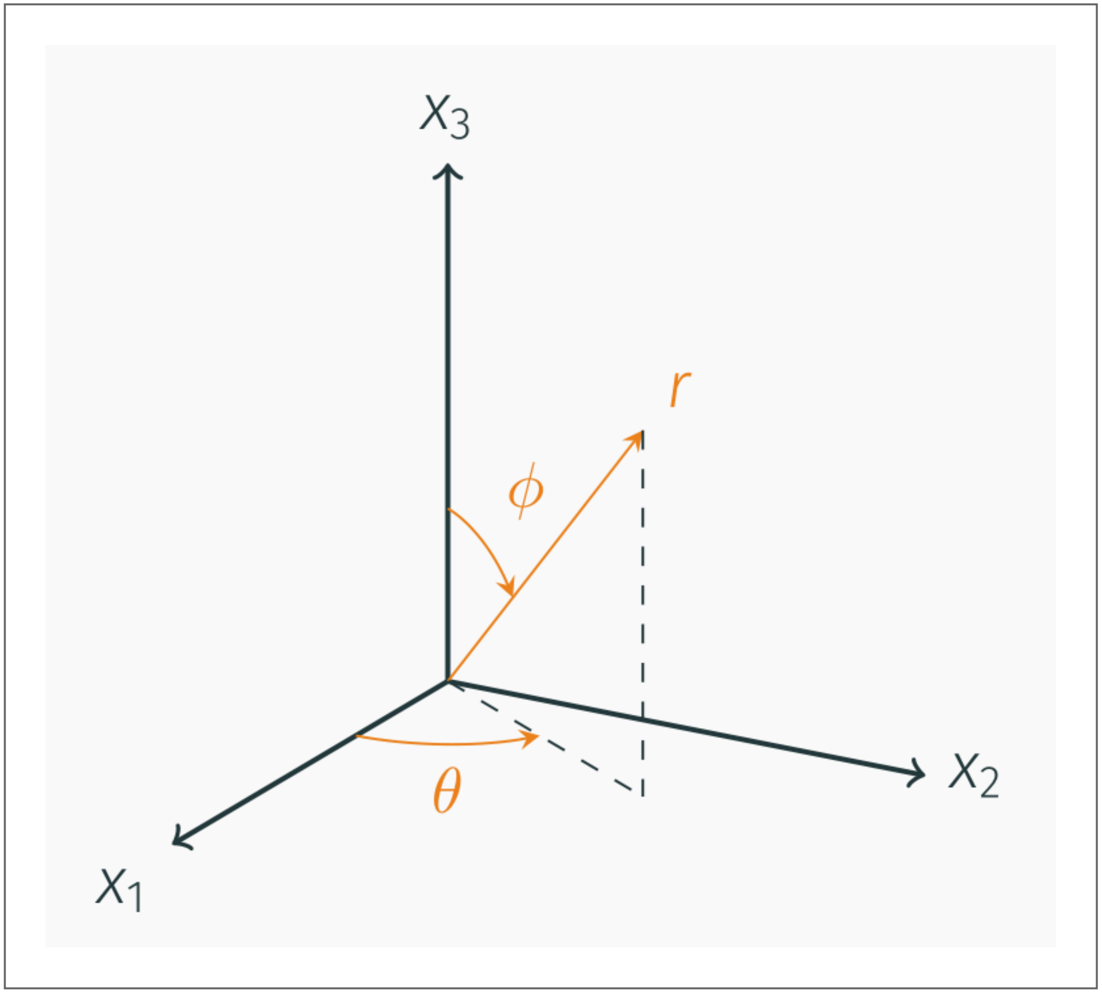
- Or to convert from Cartesian to cylindrical

$$r = \sqrt{x_1^2 + x_2^2}$$

$$\theta = \tan^{-1}\left(\frac{x_2}{x_1}\right)$$

$$z = x_3$$

spherical coordinates



spherical coordinates

- We can convert between Cartesian and spherical coordinate systems

$$x_1 = r\cos\theta\sin\phi$$
$$x_2 = r\sin\theta\sin\phi$$
$$x_3 = r\cos\phi$$

spherical coordinates

- Or to convert from Cartesian to cylindrical

$$r = \sqrt{x_1^2 + x_2^2 + x_3^2}$$

$$\phi = \cos^{-1}\left(\frac{x_3}{\sqrt{x_1^2 + x_2^2 + x_3^2}}\right)$$

$$\theta = \tan^{-1}\left(\frac{x_2}{x_1}\right)$$

calculus in cylindrical coordinates

$$\nabla f = \frac{\partial f}{\partial r}\hat{r} + \frac{1}{r}\frac{\partial f}{\partial \theta}\hat{\theta} + \frac{\partial f}{\partial z}\hat{z}$$

$$\nabla \cdot \mathbf{u} = \frac{1}{r}\frac{\partial (r u_r)}{\partial r} + \frac{1}{r}\frac{\partial u_\theta}{\partial \theta} + \frac{\partial u_z}{\partial z}$$

$$\nabla \times \mathbf{u} = \left(\frac{1}{r}\frac{\partial u_z}{\partial \theta} - \frac{\partial u_\theta}{\partial z}\right)\hat{r} + \left(\frac{\partial u_r}{\partial z} - \frac{\partial u_z}{\partial r}\right)\hat{\theta} + \frac{1}{r}\left(\frac{\partial (r u_\theta)}{\partial r} - \frac{\partial u_r}{\partial \theta}\right)\hat{z}$$

calculus in spherical coordinates

$$\nabla f = \frac{\partial f}{\partial r}\hat{r} + \frac{1}{r}\frac{\partial f}{\partial \phi}\hat{\phi} + \frac{1}{r\sin\phi}\frac{\partial f}{\partial \theta}\hat{\theta}$$

$$\nabla \cdot \mathbf{u} = \frac{1}{r^2}\frac{\partial (r^2 u_r)}{\partial r} + \frac{1}{r\sin\phi}\frac{\partial (u_\phi \sin\phi)}{\partial \phi} + \frac{1}{r\sin\phi}\frac{\partial u_\theta}{\partial \theta}$$

$$\nabla \times \mathbf{u} = \frac{1}{r\sin\phi}\left(\frac{\partial (u_\theta \sin\phi)}{\partial \phi} - \frac{\partial u_\phi}{\partial \theta}\right)\hat{r} + \frac{1}{r}\left(\frac{1}{\sin\phi}\frac{\partial u_r}{\partial \theta} - \frac{\partial (r u_\theta)}{\partial r}\right)\hat{\phi} + \frac{1}{r}\left(\frac{\partial (r u_\phi)}{\partial r} - \frac{\partial u_r}{\partial \phi}\right)\hat{\theta}$$

chapter summary

topics

- Index notation
  - Free index vs. dummy index
  - Solving matrix and vector equations
  - Translation to matrix expressions
  - Programming with index notation

topics

- Coordinate transformation
  - Direction cosines
  - Compound transformations (multiple rotations)
  - Vector, matrix, and general tensor transformation

topics

- Principal values, directions, and invariants
- Partial derivative notation
- Cylindrical and spherical coordinates