

Name: Jennifer Ruliffson

UFID: 8650 2446

Q1: (-4.97192947, 8.5042713); (-5.83969695, 3.29766643)

Q2: 8; 1.1290

Q3: 0.1533

Q4: NOT ANSWERED

Q5: 116.7479 GPa

QUIZ 2: QUESTION 1

Givens

- UFID = 8650 2446
- X1 = the eighth digit of your Gator ID number = 6
- Y1 = the seventh digit of your Gator ID number = 4
- X2 = the fourth digit of your Gator ID number = 0
- Y2 = the third digit of your Gator ID number = 5
- point P in 2D space of coordinates $(X_1, Y_1) = (6, 4)$
- Rotated by a 7-fold rotational axis perpendicular to the 2D space
- Rotation axis located in the coordinate $(X_2, Y_2) = (0, 5)$

References used throughout this problem:

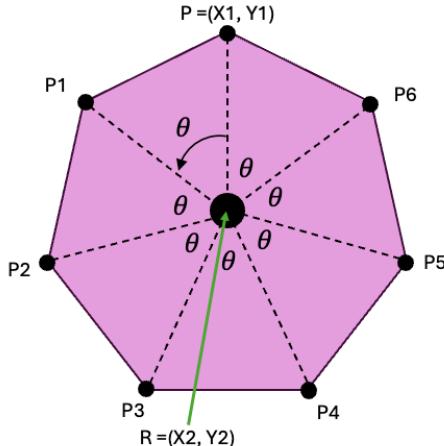
- *Ruliffson work from Quiz 1, Question 3*
- *Dr. Nino feedback from Quiz 1*

Solutions

Link to python notebook of all work for this question: <https://github.com/j3rulif/SC-Quiz/blob/main/SC-Quiz-Q1-JenniferRuliffson.ipynb>

Part A - How many points does the rotational axis generate?

- **6 new points generated**



(Image created in PowerPoint by Jennifer)

Part B - Which are the coordinates of those generated points?

- P1: [4.52277029 9.06749909]
- P2: [-0.36019769 11.07208841]
- P3: [-4.97192947 8.5042713]
- P4: [-5.83969695 3.29766643]
- P5: [-2.31005352 -0.62704654]
- P6: [2.95910733 -0.3144787]

Determine theta in radians:

$$\theta = \frac{360^\circ}{7} = 51.4286^\circ \rightarrow 51.4286^\circ \frac{\pi}{180^\circ} \text{ radians} = 0.8976 \text{ radians}$$

Rotation matrix:

- $$\mathbf{R} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

Rotate the point:

- $$P_1 = \begin{pmatrix} X_3 \\ Y_3 \end{pmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{pmatrix} X_1 - X_2 \\ Y_1 - Y_2 \end{pmatrix} + \begin{pmatrix} X_2 \\ Y_2 \end{pmatrix}$$
- $$P_1 = \mathbf{R}(\theta)(P_0 - P_{\text{rotate}}) + (P_{\text{rotate}})$$
- GENERAL:
$$P_n = \mathbf{R}(\theta)(P_{n-1} - P_{\text{rotate}}) + (P_{\text{rotate}})$$

Python Code used to compute:

```
[1]: # imports
import numpy as np
import matplotlib.pyplot as plt

[2]: # inputs for first rotation
p = [6, 4] #(X1, Y1)
rotate_point = [0, 5] #(X2, Y2)
theta = 360/7 * np.pi/180 # 7-fold rotation, so 360/7

# 2D rotation matrix
R = np.array([
    [np.cos(theta), -np.sin(theta)],
    [np.sin(theta), np.cos(theta)]
])

[3]: # Convert to numpy arrays
p_float = np.array(p, dtype=float)
rot_float = np.array(rotate_point, dtype=float)

# Rotate all the points
p1 = R @ (p_float - rot_float) + rot_float
p2 = R @ (p1 - rot_float) + rot_float
p3 = R @ (p2 - rot_float) + rot_float
p4 = R @ (p3 - rot_float) + rot_float
p5 = R @ (p4 - rot_float) + rot_float
p6 = R @ (p5 - rot_float) + rot_float
```

```
]: #list them all
print("P1:", p1)
print("P2:", p2)
print("P3:", p3)
print("P4:", p4)
print("P5:", p5)
print("P6:", p6)
```

```
P1: [4.52277029 9.06749909]
P2: [-0.36019769 11.07208841]
P3: [-4.97192947 8.5042713 ]
P4: [-5.83969695 3.29766643]
P5: [-2.31005352 -0.62704654]
P6: [ 2.95910733 -0.3144787 ]
```

Part C (Summary Page) - Which are the coordinates of the points farthest away from the original point P?

- P3: [-4.97192947 8.5042713]
- P4: [-5.83969695 3.29766643]

Rationale:

- Farthest point will be one(s) closest to 180° away from original point.
- For 7-fold rotation, will be two points. Based on the diagram labels:
 - P3: [-4.97192947 8.5042713]
 - P4: [-5.83969695 3.29766643]

QUIZ 2: QUESTION 2

Givens:

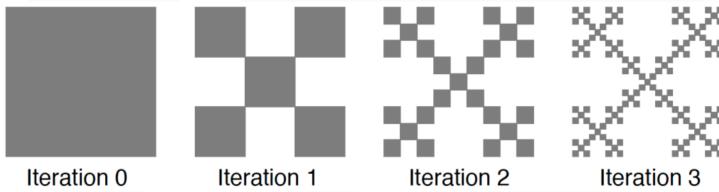
- Polygon flake (Polyflake) fractal
- UFID = 8650 2446
- UFID Sorted = 8665 4420
- First digit of UFID sorted is the number of sides of your polygon = 8
- the polygon flake to work on is the one that includes the central polygon; for clarification, see the fractal on the right hand side on S5-SL64, or the one the left hand side of S5-SL69).



Exercise



The first generations (iterations) of a self-similar pattern is presented below. Calculate its fractal dimension



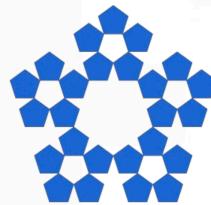
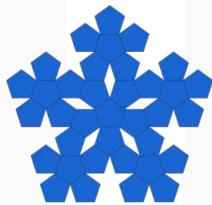
UNIVERSITY of FLORIDA Professor Juan Claudio Nino - EMA 6313: Structure and Mechanical Properties of Materials - Fall 2025

64

Generalizations



An ***n*-flake, polyflake, or Sierpinski *n*-gon**, is a fractal constructed starting from an *n*-gon. This *n*-gon is replaced by a flake of smaller *n*-gons, such that the scaled polygons are placed at the vertices, and sometimes in the center. This process is repeated recursively to result in the fractal. There is also the restriction that the *n*-gons must touch yet not overlap.



What is the dimension of each of the fractals?



UNIVERSITY of FLORIDA Professor Juan Claudio Nino - EMA 6313: Structure and Mechanical Properties of Materials - Fall 2025

69

Additional References used throughout:

- Rulifson Quiz 1, Question 4 Solution plus Dr. Nino Feedback

Solutions:

Part A: (Summary Page) How many sides does your base polygon have (generation zero).

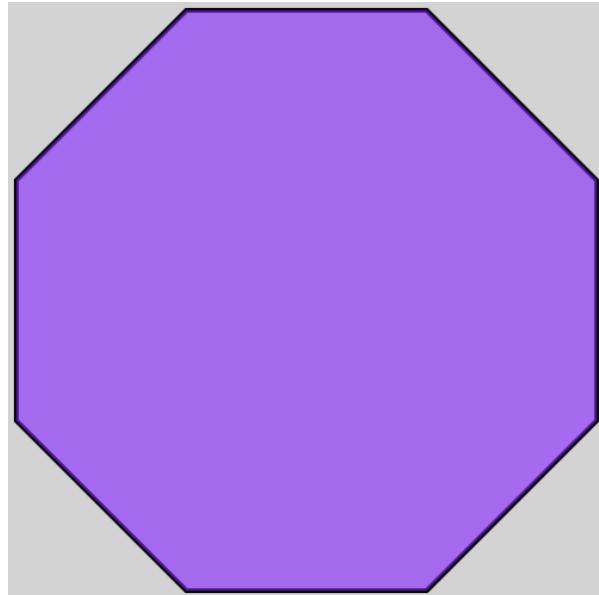
- $n=8$

Part B: Generate geometrically accurate drawings for the first two generations

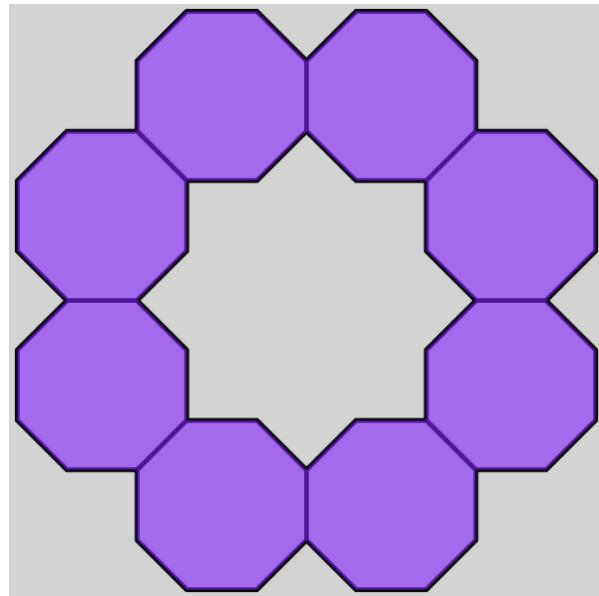
- (zero and 1) of the fractal you are working with.

Generated using <https://onlinetools.com/math/generate-sierpinski-polyflake>

ZERO:



ONE:



Part C: (Summary Page) Determine the Hausdorff dimension for your fractal.

- 1.1290
 - Report the number with 5 significant figures.

Step 1: Determine the Scaling Factor

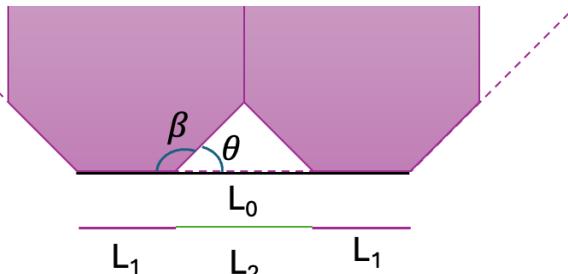
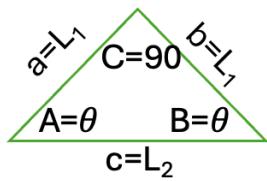
- Interior angle of an octagon (135°) from Wikipedia:
<https://en.wikipedia.org/wiki/Octagon>

$$\frac{L_0}{L_1} = \frac{1}{r} = 3.41421356$$

Work to determine $1/r$:

$$\theta = 180 - \beta$$

$$\theta = 180 - 135 = 45$$



$$L_0 = 2L_1 + L_2$$

Law of Sines

$$\frac{a}{\sin(A)} = \frac{b}{\sin(B)} = \frac{c}{\sin(C)}$$

$$\frac{L_1}{\sin(45)} = \frac{L_2}{\sin(90)}$$

$$L_2 = \frac{2L_1}{\sqrt{2}}$$

$$L_0 = 2L_1 + \frac{2L_1}{\sqrt{2}}$$

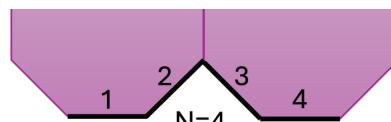
$$L_0 = L_1 \left(2 + \frac{2}{\sqrt{2}} \right)$$

$$\frac{1}{r} = \frac{L_0}{L_1} = 2 + \frac{2}{\sqrt{2}} = 3.41421356$$

Step 2: Calculate the Hausdorff Dimension

- Using the equation on slide 34 of the lecture notes titled “S5 - Fractals F25 Canvas” and page 9 of the digital document by titled “Chapter 4_ Calculating Fractal Dimensions” to solve for the Hausdorff Dimension to 4 significant figures:

$$D = \frac{\log(N)}{\log\left(\frac{1}{r}\right)} = \frac{\log(4)}{\log(3.41421356)} \approx 1.12895277$$



QUIZ 2: QUESTION 3

Givens:

- Second rank tensor transformation
- Cesium sulfide CsS_3
- Space group: p1
 - Translates, but no other symmetry
- Lattice parameters: $a = 4.75 \text{ \AA}$, $b = 9.32 \text{ \AA}$, $c = 11.65 \text{ \AA}$
- Angles: $\alpha = 95.07^\circ$, $\beta = 95.31^\circ$, $\gamma = 89.60^\circ$
- Relative dielectric permittivity tensor: $k_{11} = 6.692$, $k_{12} = 0.893$, $k_{13} = 0.1519$, $k_{22} = 11.4$, $k_{23} = 0.1739$, and $k_{33} = 6.723$

References used throughout:

- Quiz 2 Q6/Q7, Quiz 3 Q1, and Dr. Nino quiz feedback
- SL15, SL16, De Graef dot/cross product reading

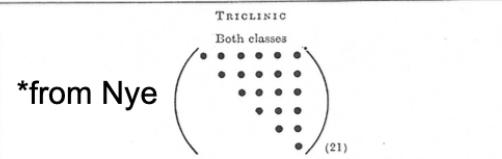
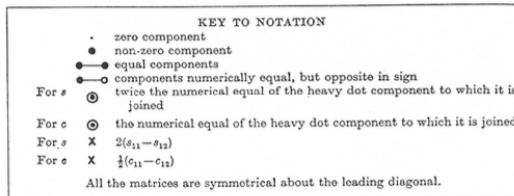
Solutions:

Link to python notebook of all work for this question: <https://github.com/j3rulif/SC-Quiz/blob/main/SC-Quiz-Q3-JenniferRuliffson.ipynb>

Part A: Write the k tensor that represents the dielectric permittivity for this material.

$$k_{\text{crystal basis}} = \begin{matrix} k_{11} & k_{12} & k_{13} & 6.692 & 0.893 & 0.1519 \\ k_{12} & k_{22} & k_{23} & 0.893 & 11.4 & 0.1739 \\ k_{13} & k_{23} & k_{33} & 0.1519 & 0.1739 & 6.723 \end{matrix}$$

- Triclinic system:



- (SL15, p97)

Triclinic	A centre of symmetry or no symmetry	diad axis (y) General quadric. No fixed relation to crystallographic axes	6	$\begin{bmatrix} S_{11} & S_{12} & S_{13} \\ S_{21} & S_{22} & S_{23} \\ S_{31} & S_{32} & S_{33} \end{bmatrix}$
-----------	-------------------------------------	---	---	--

- (SI15, p31)

- So k-tensor will have all non-zero values, reflected across the diagonal

Python work:

```

: # GIVENS

#lattice parameters
a = 4.75 #Å
b = 9.32 #Å
c = 11.65 #Å

#angles
alpha_deg = 95.07#°
alpha = alpha_deg * np.pi / 180
beta_deg = 95.31#°
beta = beta_deg * np.pi / 180
gamma_deg = 89.60#°
gamma = gamma_deg * np.pi / 180

# Relative dielectric permittivity tensor
k11 = 6.692
k12 = 0.893
k13 = 0.1519
k22 = 11.4
k23 = 0.1739
k33 = 6.723

import numpy as np

k_crystal_basis = np.array([[k11,k12,k13],[k12,k22,k23],[k13,k23,k33]])
print("k = ", k_crystal_basis)

k = [[ 6.692   0.893   0.1519]
     [ 0.893   11.4     0.1739]
     [ 0.1519  0.1739  6.723 ]]

```

Part B: (Summary Page) Determine the value of the dielectric permittivity for this material along the [3 D7 2] direction.

- **0.1533**
 - [3 4 2]
 - D7 = 4
 - UFID = 8650 2446

Notes:

- References: S15 p61, S16 pp56-58
- Sources: De Graef Introduction to Conventional TEM, pp 5-9; S16 pp 34-35
- DIRECT metric tensor (p6):

$$G = \begin{bmatrix} a^2 & ab \cdot \cos\gamma & ac \cdot \cos\beta \\ ab \cdot \cos\gamma & b^2 & bc \cdot \cos\alpha \\ ac \cdot \cos\beta & bc \cdot \cos\alpha & c^2 \end{bmatrix}$$

Python Work:

METRIC TENSOR FOR TRICLINIC

```
# from de graef reading
metric_tensor = np.array([
    [a**2, a*b*np.cos(gamma), a*c*np.cos(beta)],
    [a*b*np.cos(gamma), b**2, b*c*np.cos(alpha)],
    [a*c*np.cos(beta), b*c*np.cos(alpha), c**2]
])

#expecting a value in each position since no angles are 90 for triclinic
print("metric tensor = ", metric_tensor)

metric tensor = [[ 22.5625      0.30906039 -5.12117423]
 [ 0.30906039  86.8624     -9.59533741]
 [-5.12117423 -9.59533741 135.7225     ]]
```

APPLY THE DIRECTION

```
: # UFID = 8650 2446; D7 = 4

#direction vector
direction = np.array([3, 4, 2]) # [u,v,w]

# magnitude of direction, |d| = sqrt d^T G d
dir_mag = np.sqrt(direction.T @ metric_tensor @ direction) #angstroms

#normalized direction, n = d / |d|
n = direction / dir_mag #unitless

print("magnitude of direction: ", dir_mag, " angstroms")
print("normalized direction: ", n, "(unitless)")

magnitude of direction: 43.91114733385873 angstroms
normalized direction: [0.06831978 0.09109304 0.04554652] (unitless)
```

CALC EFFECTIVE VALUE IN DIRECTION

```
: # determine effective k, after S15

k_eff = n.T @ k_crystal_basis @ n # effective k along the direction.

print(k_eff, " (unitless)")

0.1532822879046431 (unitless)
```

QUIZ 2: QUESTION 4

Not answered

QUIZ 2: QUESTION 5

Young's Modulus in any direction

Givens:

- Molybdenum dioxide, MoO_2
- crystallizes with space group $P2 / c$
 - MONOCLINIC
 - P = primitive lattice
 - 2 = 2-fold rotation axis
 - c = glide plane (c)
 - symmetry operations include inversion centers and glide reflections
- lattice parameters, $a = 5.54 \text{ \AA}$, $b = 4.87 \text{ \AA}$, $c = 5.64 \text{ \AA}$,
- Angles: α - alpha = 90° , β - beta = 120.18° , and γ - gamma = 90° ,
- Mechanical coefficients (in GPa) $C = 358$, $C = 144$, $C = 67$, $C = -25.9$, $C = 426$, $C = 127$, $C = 38.3$, $C = 240$, $C = -23.3$, $C = 99.1$, $C = -38.8$, $C = 78.7$, $C = 130$

Solutions:

Link to python notebook for all work on this question: <https://github.com/j3rulif/SC-Quiz/blob/main/SC-Quiz-Q5-JenniferRulifson.ipynb>

Part A: calculate and write full compliance matrix

- Compliance matrix (S_{ij}) is the inverse of the stiffness matrix (C_{ij})

From SL15, p97:

KEY TO NOTATION	
•	zero component
•	non-zero component
—	equal components
—	components numerically equal, but opposite in sign
For s	◎ twice the numerical equal of the heavy dot component to which it is joined
For c	◎ the numerical equal of the heavy dot component to which it is joined
For s	✗ $2(s_{11} - s_{12})$
For c	✗ $\frac{1}{2}(c_{11} - c_{12})$
All the matrices are symmetrical about the leading diagonal.	
MONOCLINIC	
All classes	
Diad $\parallel x_2$ (standard orientation)	$\begin{pmatrix} \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \end{pmatrix} \quad (13)$
Diad $\parallel x_2$	$\begin{pmatrix} \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \end{pmatrix} \quad (13)$

Python work:

```

#GIVENS

#lattice parameters,
a = 5.54 #Å,
b = 4.87 #Å,
c = 5.64 #Å,

# angles in degrees
alpha_deg = 90 #°,
beta_deg = 120.18 #°
gamma_deg = 90 #°,

#angles in radians
import numpy as np
alpha = alpha_deg * np.pi /180
beta = beta_deg * np.pi /180
gamma = gamma_deg * np.pi /180

# mechanical coefficients (in GPa)
c11 = 358
c12 = 144
c13 = 67
c15 = -25.9
c22 = 426
c23 = 127
c25 = 38.3
c33 = 240
c35 = -23.3
c44 = 99.1
c46 = -38.8
c55 = 78.7
c66 = 130

```

```

# stiffness matrix

C = np.array([
    [c11, c12, c13, 0, c15, 0],
    [c12, c22, c23, 0, c25, 0],
    [c13, c23, c33, 0, c35, 0],
    [0, 0, 0, c44, 0, c46],
    [c15, c25, c35, 0, c55, 0],
    [0, 0, 0, c46, 0, c66]
])
print(C)

[[358. 144. 67. 0. -25.9 0. ]
 [144. 426. 127. 0. 38.3 0. ]
 [67. 127. 240. 0. -23.3 0. ]
 [0. 0. 0. 99.1 0. -38.8]
 [-25.9 38.3 -23.3 0. 78.7 0. ]
 [0. 0. 0. -38.8 0. 130. ]]

[3]:
# compliance matrix is inverse of stiffness matrix
S = np.linalg.inv(C)
print("S = ", S, "(GPa)^-1")

S = [[ 0.00345939 -0.0012911 -0.0001143 0. 0.00173296 0. ]
 [-0.0012911 0.00355632 -0.00178194 0. -0.00268317 0. ]
 [-0.0001143 -0.00178194 0.0053766 0. 0.00242138 0. ]
 [ 0. 0. 0. 0.011426 0. 0.00341022]
 [ 0.00173296 -0.00268317 0.00242138 0. 0.01529946 0. ]
 [ 0. 0. 0. 0.00341022 0. 0.00871013]] (GPa)^-1

```

Part B: (Summary Page) Calculate the Young's modulus along the [2 D3 1] direction

Young's modulus along [2 5 1]: 275.8053215498748 GPa

- [2 5 1] direction
 - UFID = 8650 2446; D3 = 5
- "All directions and planes are referred to in the crystallographic system."
- The mechanical coefficients are already in crystallographic, SO no metric tensor etc
 - $\frac{1}{E(n)} = S_{ijkl}n_i n_j n_k n_l$; $\frac{1}{E(n)} = S_{ijkl}m_{ij} n_{kl}$
 - For monoclinic, many of the 81 components are zero:

$$E = \frac{1}{S_{1111}} \quad E' = \frac{1}{S'_{1111}} \quad S'_{1111} = a_{1m} a_{1n} a_{1o} a_{1p} S_{mnop}$$

1) Into how many terms does the blue equation expand? 81

$S_{11} = S_{1111}$	1	$S_{23} = S_{32} = S_{2233} = S_{3322}$	33
$S_{22} = S_{2222}$	2	$S_{24} = S_{42} = S_{2223} = S_{2232} = S_{3322} = S_{3222}$	37
$S_{33} = S_{3333}$	3	$S_{25} = S_{52} = S_{2213} = S_{2231} = S_{1322} = S_{3122}$	41
$S_{44} = S_{2323} = S_{3232} = S_{2332} = S_{3223}$	7	$S_{26} = S_{62} = S_{2212} = S_{2221} = S_{1222} = S_{2122}$	45
$S_{55} = S_{1313} = S_{1311} = S_{1331} = S_{3113}$	11	$S_{34} = S_{43} = S_{3323} = S_{3322} = S_{2333} = S_{3233}$	49
$S_{66} = S_{1212} = S_{2121} = S_{1221} = S_{2112}$	15	$S_{35} = S_{53} = S_{3313} = S_{3331} = S_{1333} = S_{3133}$	53
$S_{12} = S_{21} = S_{1122} = S_{2211}$	17	$S_{36} = S_{63} = S_{3312} = S_{3321} = S_{1233} = S_{2133}$	57
$S_{13} = S_{31} = S_{1133} = S_{3311}$	19	$S_{45} = S_{54} = S_{2919} = S_{2994} = S_{9219} = S_{1929} = S_{3129} = S_{4132}$	65
$S_{14} = S_{41} = S_{1123} = S_{1132} = S_{2311} = S_{3211}$	23	$S_{46} = S_{64} = S_{2312} = S_{2321} = S_{3212} = S_{3221} = S_{1223} = S_{2123} = S_{1232} = S_{2132}$	73
$S_{15} = S_{51} = S_{1113} = S_{1131} = S_{1311} = S_{3111}$	27	$S_{58} = S_{65} = S_{1312} = S_{1321} = S_{3112} = S_{3121} = S_{1213} = S_{2113} = S_{1231} = S_{2131}$	81
$S_{16} = S_{61} = S_{1112} = S_{1121} = S_{1211} = S_{2111}$	31		

- Non-zero values: S11, S22, S33, S44, S55, S66, S12, S13, S15, S23, S25, S35, S46
- $S_{ijkl}n_i n_j n_k n_l = \sum_{i,j,k,l=1}^3 S_{ijkl}n_i n_j n_k n_l$

Python work:

```
direction = np.array([2, 5, 1])

##### "All directions and planes are referred to in the crystallographic system."
##### the mechanical coefficients are already in crystallographic, SO no metric tensor etc

# direction cosines = normalized direction vector components
dir_mag = np.linalg.norm(direction)
n = direction / dir_mag      # unit vector

n1, n2, n3 = n # n is the direction cosines
print(n)

[0.36514837 0.91287093 0.18257419]
```

```

# need a 6 component vector to work with the 6x6 matrix.
# voigt vector form; leveraging S17 pp31-34
# voigt notation, create the column vector representing the direction cosines
q = np.array([
    n1**2,# 11
    n2**2,# 22
    n3**2,# 33
    2*n2*n3,# 23
    2*n1*n3,# 13
    2*n1*n2 # 12
])

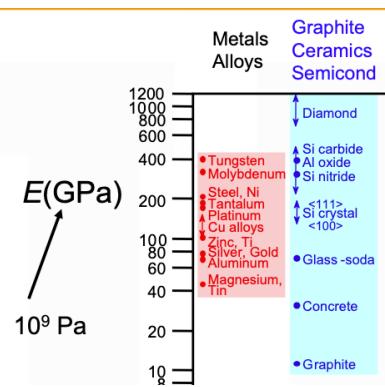
inv_E = q @ S @ q # 1/E(n), in 1/GPa
E_n = 1.0 / inv_E # Young's modulus a in GPa

print("Young's modulus along [2 5 1]:", E_n, "GPa")

```

Young's modulus along [2 5 1]: 116.74794356713704 GPa

Check answer order of magnitude for metal oxides = OK:



HONOR CODE STATEMENT

"On my honor, I assert that have neither given nor received unauthorized aid in doing this exam."

A handwritten signature in black ink, appearing to read "J. Ruliffson".

Jennifer Ruliffson