

Київський національний університет імені Тараса Шевченка

ННІ "Інститут геології"

Кафедра геоінформатики

## **Практична робота № 1**

# **Просторовий аналіз та статистичне моделювання в геології**

Виконала студентка 3 курсу, групи «Big Data»

Яковенко Марія Ігорівна

Київ-2023

## Завдання

а) У просторі  $R^2$  заданий вектор  $\vec{i}$ . Визначити, яким буде результат застосування в заданій послідовності наступних перетворень:

- 1)  $R(\theta)$  – обертання у площині  $O_{xy}$  на  $\theta$  градусів;
- 2)  $T(x, y)$  – зсув на площині  $O_{xy}$ ;
- 3)  $S(x, y)$  – масштабування (розтягування та/або стиснення).

Вирішити завдання для двох варіантів вхідних даних.

б) Запрограмувати мовою Python операцію лінійного відображення вхідного вектора (матриці) у вихідних за допомогою перетворень: обертання  $R$ , зсуву  $T$  та масштабування  $S$ .

## Вхідні дані

Таблиця 1. Варіант 2

Послідовність	$\vec{i}(x, y)$		$R(\theta)$	$T(x_T, y_T)$		$S(x_S, y_S)$	
$S-R-T$	3	2	$12.6^\circ$	2.0	2.0	1.0	2.2
$T-R-S$			$36.4^\circ$	1.0	1.0	1.3	1.2

Таблиця 2. Варіант 3

Послідовність	$\vec{i}(x, y)$		$R(\theta)$	$T(x_t, y_t)$		$S(x_s, y_s)$	
$S-R-T$	6	2	$15.0^\circ$	2.0	2.0	1.5	2.7
$T-R-S$			$51.0^\circ$	3.0	2.0	1.0	3.2

Хід роботи (GitHub: <https://github.com/mashaven/sheva-stats/> – task-1)

Для спрощення побудови алгоритму в якості нульової операції для даної задачі будемо розглядати операцію тотожного перетворення  $E$ :

$$E \cdot \vec{x} = \vec{x},$$

де матриця трансформації  $E$  – одинична матриця:

$$E = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Вигляд матриць трансформації для операцій обертання  $R$ , зсуву  $T$  та масштабування  $S$ , відповідно, буде наступним:

$$R(\theta) = \begin{pmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (1)$$

$$T(x_T, y_T) = \begin{pmatrix} 1 & 0 & x_T \\ 0 & 1 & y_T \\ 0 & 0 & 1 \end{pmatrix} \quad (2)$$

$$S(x_S, y_S) = \begin{pmatrix} x_S & 0 & 0 \\ 0 & y_S & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (3)$$

Операція лінійного перетворення матриці  $T(\vec{x}) = \vec{y}$  володіє властивістю асоціативності, тобто:

$$T_1 \cdot (T_2 \cdot \vec{x}) = (T_1 \cdot T_2) \cdot \vec{x}$$

Таким чином, якщо відомі операції трансформацій  $T = \{R, T, S\}$  та послідовність їх застосування  $Seq = \{t_i | t_i \in T, i = \overline{1, n}\}$  – можна розрахувати матрицю лінійного відображення вхідного вектора  $T: \vec{x} \rightarrow \vec{y}$  за наступним алгоритмом:

```
transformMatrix(operations: dict[Name: Matrix], sequence: list[Name]):
    Tr: Matrix = identity(3)
    for each name in sequence:
        Tr = Tr * operations[name]
    return Tr
```

Запрограмуємо розрахунок матриць (1-3) мовою Python:

а) обертання,  $R$

```
def rotate(theta):
    theta = theta * np.pi / 180
    return np.array([
        [np.cos(theta), -np.sin(theta), 0],
        [np.sin(theta), np.cos(theta), 0],
        [0, 0, 1]
    ])
```

б) зсув,  $T$

```
def move(x, y):  
    return np.array([  
        [1, 0, x],  
        [0, 1, y],  
        [0, 0, 1]  
    ])
```

в) масштабування,  $S$

```
def scale(x, y):  
    return np.array([  
        [x, 0, 0],  
        [0, y, 0],  
        [0, 0, 1]  
    ])
```

Та виконаємо реалізацію методу `modelMatrix(inMatrix, sequence)` як функції мовою Python для вирішення поставленого завдання за отриманими варіантами:

```
def modelMatrix(inMatrix, sequence):  
    V = np.array(inMatrix['V'] + (1,)) # V(x, y, 1)  
    # calculate transformation matrix for each operation  
    Tr = {}  
    Tr['R'] = rotate( inMatrix['R'])  
    Tr['T'] = move (*inMatrix['T'])  
    Tr['S'] = scale (*inMatrix['S'])  
    # calculate a transformation matrix for the `sequence`  
    T = np.identity(3)  
    for op in sequence:  
        T = T @ Tr[op] # @ is a binary operator for the matrix mul.  
    # apply the transformation matrix to the vector `V`  
    Vtr = T @ V  
    return Vtr
```

За допомогою отриманої функції обчислимо розв'язок поставленого завдання:

а) варіант 2

```
In [8]: inMatrix = {  
        'R': 12.6,  
        'T': (2, 2),  
        'S': (1, 2.2),  
        'V': (3, 2),  
        }  
        modelMatrix(inMatrix, 'SRT')  
  
Out[8]: array([ 4.00701084, 10.98764316,  1.          ])
```

Рисунок 1. Результати перетворень вектору  $\vec{l}(3, 2)$  в послідовності **S-R-T**

```
In [9]: inMatrix = {  
        'R': 36.4,  
        'T': (1, 1),  
        'S': (1.3, 1.2),  
        'V': (3, 2),  
        }  
        modelMatrix(inMatrix, 'TRS')  
  
Out[9]: array([2.71488048, 5.24607877,  1.          ])
```

Рисунок 2. Результати перетворень вектору  $\vec{l}(3, 2)$  в послідовності **T-R-S**

б) варіант 3

```
In [10]: inMatrix = {  
        'R': 15,  
        'T': (2, 2),  
        'S': (1.5, 2.7),  
        'V': (6, 2),  
        }  
        modelMatrix(inMatrix, 'SRT')  
  
Out[10]: array([10.03819564, 16.0224903 ,  1.          ])
```

Рисунок 3. Результати перетворень вектору  $\vec{l}(6, 2)$  в послідовності **S-R-T**

```
In [11]: inMatrix = {  
        'R': 51,  
        'T': (3, 2),  
        'S': (1, 3.2),  
        'V': (6, 2),  
        }  
        modelMatrix(inMatrix, 'TRS')  
  
Out[11]: array([ 1.80218819, 10.69052627,  1.          ])
```

Рисунок 4. Результати перетворень вектору  $\vec{l}(6, 2)$  в послідовності **T-R-S**