

Data visualization course

Laboratory work 1

Drawing tensor fields via Matplotlib



Scalar fields visualisation

A scalar field (scalar function) on a finite space V is a function that corresponds to each point in some region of this space (the domain of definition) with a scalar, i.e. a real or complex number. With a fixed basis of space, a scalar field can be defined as a function of several variables on the coordinates of a point.

The difference between a numerical function of several variables and a scalar field is that in a different basis, the scalar field as a function of coordinates changes so that if a new set of arguments represents the same point in space in the new basis, the value of the scalar function does not change.

For example, if in some orthonormalized basis of a two-dimensional vector space a scalar function has the form $f(v) = x^2 + 2y^2$, then in another basis rotated 45 degrees to this one, the same function in the new coordinates will have the form $f(v) = 3x'^2 + 3y'^2 - 2x'y'$.

Most often, we consider scalar functions that are continuous or differentiable (smooth) a sufficient number of times. In practice, they are found to be widely used:

- A function of three variables: $u = u(\mathbf{r}) = u(x, y, z)$ (a scalar field in three-dimensional space).
- A function of two variables: $u = u(\mathbf{r}) = u(x, y)$ (a scalar field in two-dimensional space).

Examples of scalar fields in three dimensions are:

- temperature (if it is different in different points of space);
- electrostatic potential;
- potential in the Newtonian theory of gravitation;
- pressure field in a liquid medium.

Examples of flat (two-dimensional) scalar fields:

- sea depth marked in any way on a flat map;
- charge density on a flat surface of a conductor.

One of the important characteristics of a scalar field is its gradient. The gradient of a scalar field is a vector field with components:

$$\text{grad}(u(x,y)) = \left(\frac{\partial u(x,y)}{\partial x}, \frac{\partial u(x,y)}{\partial y} \right) \quad (1)$$

Example. Plot the visualization of a scalar field $u(x,y) = \sin(x^2+y^2)$, $x \in [-3;3]$; $y \in [-3;3]$. The implementation is presented in Listing 3.1.

```
import numpy as np
from matplotlib import pyplot as plt

n = 256
x = np.linspace(-3., 3., n) # X range
y = np.linspace(-3., 3., n) # Y range
X, Y = np.meshgrid(x, y)

Z = np.sin(X**2+Y**2) # scalar field equation

plt.pcolormesh(X, Y, Z)
plt.show()
```

Listing 3.1 - Example code for visualizing a scalar field

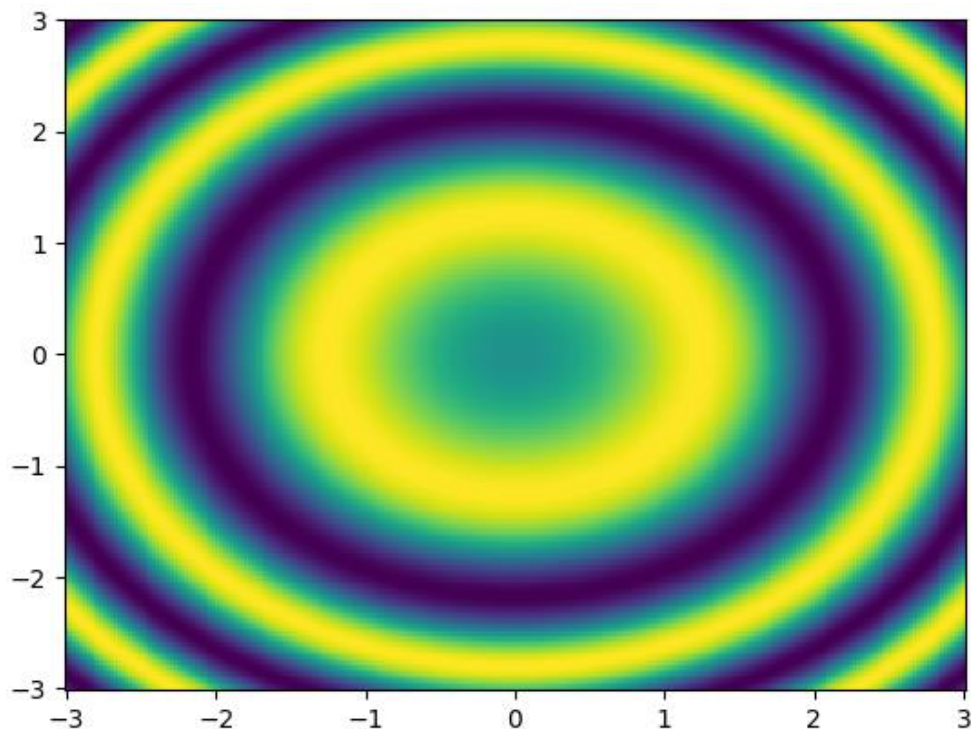


Figure 3.1 - Visualization of a scalar field

Vector field visualization

A vector field is a vector-valued function - a mapping that corresponds to each point of a given space with a vector. In the case of a Euclidean (finite-dimensional vector space with a scalar product) space, the concept of a vector field becomes clear, and the vector field is interpreted as a way to specify the

movements of a certain dynamic system: the vector at a given point describes the direction and speed of the point along the phase curve. For the Cartesian coordinate system, the field can be represented as $\mathbf{F}(\mathbf{r}) = \{u(x, y, z), v(x, y, z), w(x, y, z)\}$.

Example. Create a visualization of a flat vector field

$$F(\mathbf{r}) = \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} -y / \sqrt{x^2 + y^2} \\ x / \sqrt{x^2 + y^2} \end{pmatrix}, x, y \in [-5; 5]; \quad (2)$$

```
import numpy as np
import matplotlib.pyplot as plt

def u(x, y):
    return -y / np.sqrt(x ** 2 + y ** 2)

def v(x, y):
    return x / np.sqrt(x ** 2 + y ** 2)

xx, yy = np.meshgrid(np.linspace(-5, 5, 10),
                    np.linspace(-5, 5, 10))
u_val = u(xx, yy)
v_val = v(xx, yy)
plt.quiver(xx, yy, u_val, v_val)
plt.show()

plt.streamplot(xx, yy, u_val, v_val)
plt.show()
```

Listing 3.2 - Example code for visualizing a vector field

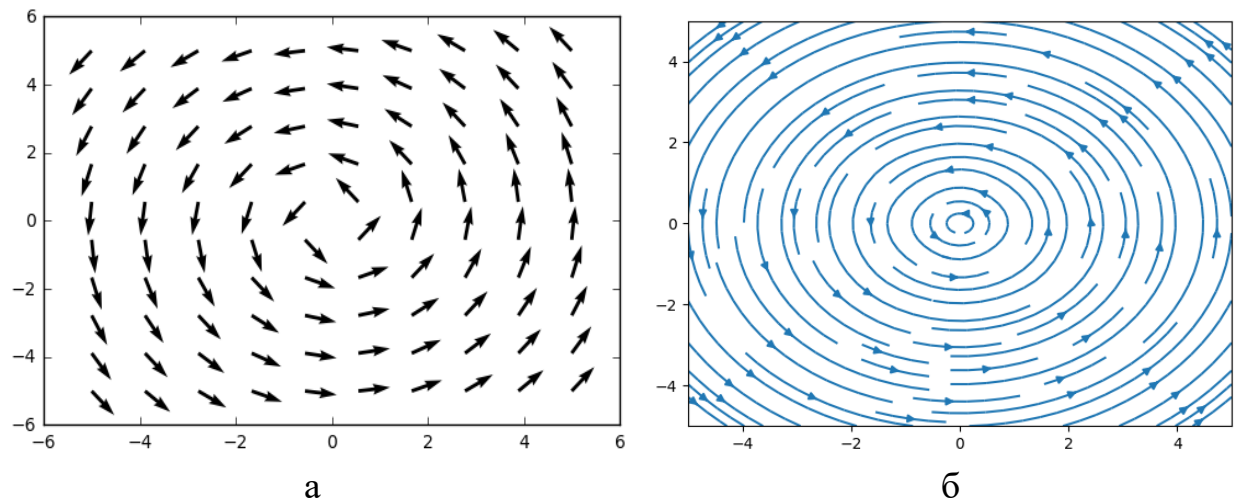


Figure 3.2 - Visualization of a vector field: a - vectors; b - flow lines

It is also quite easy to construct current lines by finding the trajectory of a massless point by performing integration according to the following principle (see Listing 3.3):

$$\begin{pmatrix} x_{i+1} \\ y_{i+1} \end{pmatrix} = \begin{pmatrix} x_i \\ y_i \end{pmatrix} + \begin{pmatrix} u(t_i) \\ v(t_i) \end{pmatrix} \Delta t \quad (3)$$

The result of building the stream lines can be seen in Fig. 3.3.

```
import numpy as np
import matplotlib.pyplot as plt

def u(x, y):
    return -y / np.sqrt(x ** 2 + y ** 2)

def v(x, y):
    return x / np.sqrt(x ** 2 + y ** 2)

def create_stream_line(x0, y0, u, v, t0=0, t1=10, dt=0.001):
    t = np.arange(t0, t1, dt)
    xx_new = np.zeros_like(t)
    yy_new = np.zeros_like(t)
    xx_new[0] = x0
    yy_new[0] = y0

    for i in range(1, t.size):
        xx_new[i] = x0 + u(x0, y0) * dt
        yy_new[i] = y0 + v(x0, y0) * dt
```

```

    x0, y0 = xx_new[i], yy_new[i]

    return xx_new, yy_new

for i in range(1, 10):
    x1, y1 = create_stream_line(i, 0, u, v)
    plt.plot(x1, y1)

plt.show()

```

Listing 3.3 - Example code for calculating and visualizing stream lines

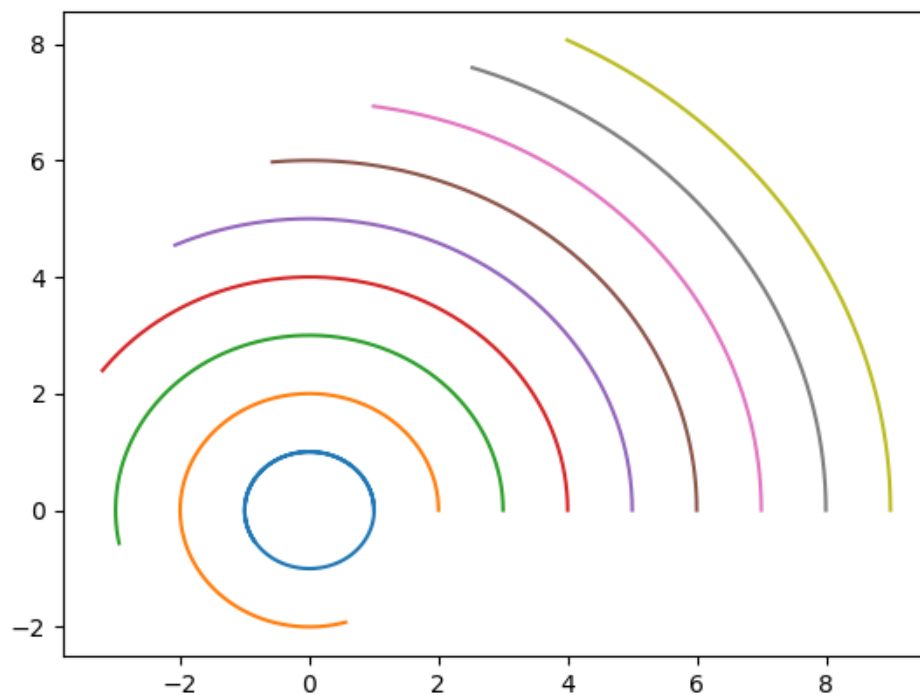


Figure 3.3 - Visualization of current lines

Example. Build a visualization of a vector field

$$F(\mathbf{r}) = \begin{pmatrix} u \\ v \\ w \end{pmatrix} = \begin{pmatrix} \sin(\pi x) \cos(\pi y) \cos(\pi z) \\ -\cos(\pi x) \sin(\pi y) \cos(\pi z) \\ \frac{\sqrt{2}}{3} \cos(\pi x) \cos(\pi y) \sin(\pi z) \end{pmatrix}, \quad x, y, z \in [-0.8; 0.8]; \quad (4)$$

```

from mpl_toolkits.mplot3d import axes3d
import matplotlib.pyplot as plt
import numpy as np

ax = plt.figure().add_subplot(projection='3d')

x, y, z = np.meshgrid(np.arange(-0.8, 1, 0.2),
                      np.arange(-0.8, 1, 0.2),
                      np.arange(-0.8, 1, 0.8))

u = np.sin(np.pi * x) * np.cos(np.pi * y) * np.cos(np.pi * z)

```

```

v = -np.cos(np.pi * x) * np.sin(np.pi * y) * np.cos(np.pi * z)
w = (np.sqrt(2.0 / 3.0) * np.cos(np.pi * x) * np.cos(np.pi * y)
* np.sin(np.pi * z))

ax.quiver(x, y, z, u, v, w, length=0.2, color = 'black')

plt.show()

```

Listing 3.4 - Visualization of a three-dimensional vector field

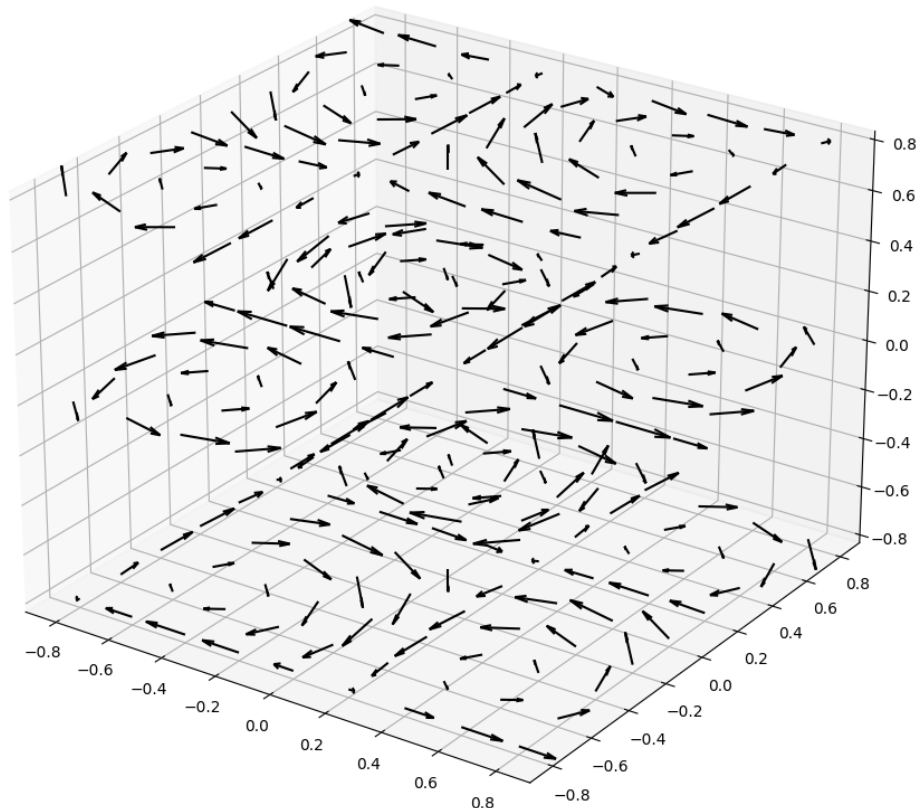


Figure 3.4 - Visualization of a three-dimensional vector field

Visualization of tensor fields

A tensor field, by analogy with a vector field, is a mapping that corresponds to a tensor at each point in space. The simplest tensor is a tensor of rank 2 and it is a "combination" of two vectors into one object. Since a vector in 3-dimensional space has 3 components, a tensor of rank 2 contains 3×3 elements and is represented by a corresponding matrix. If the tensor is symmetric, then only 6 of its

9 components are independent. One of the most common tensors is the mechanical stress tensor (Fig. 3.5):

$$\sigma = \begin{pmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\ \sigma_{xy} & \sigma_{yy} & \sigma_{yz} \\ \sigma_{xz} & \sigma_{yz} & \sigma_{zz} \end{pmatrix} \quad (5)$$

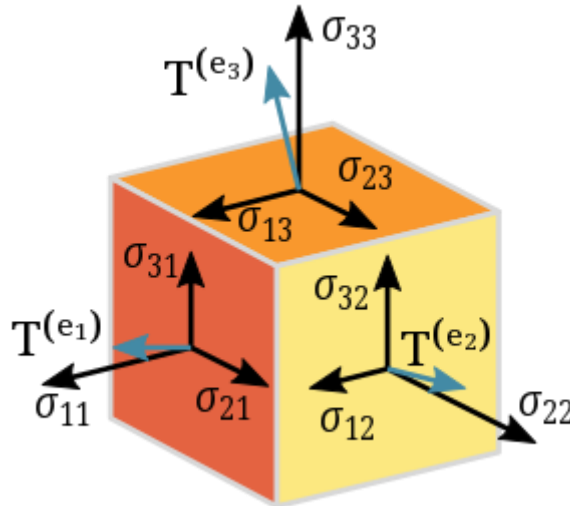



Figure 3.5 - Stress tensor components σ
 (Original image from <https://uk.wikipedia.org/wiki/Тензор>)

To build images (glyphs), it is assumed that the tensor can be represented in the form:

$$\sigma = R\Lambda R^{-1}, \quad (6)$$

where R – is the matrix of rotation of coordinates to the basis that coincides with the main directions; $\Lambda = (\lambda_1, \lambda_2, \lambda_3)$ – tensor’s eigenvalues. More details about the construction of glyphs can be found in the article by G. Kindlmann (Kindlmann G. Superquadric tensor glyphs // Proceedings of the Sixth Joint Eurographics-IEEE TCVG conference on Visualization. – 2004. – C. 147-154. <https://people.cs.uchicago.edu/~glk/pubs/pdf/Kindlmann-Superquad1-VisSym-2004.pdf>).



$$c_l = \frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2 + \lambda_3}$$

$$c_p = \frac{2(\lambda_2 - \lambda_3)}{\lambda_1 + \lambda_2 + \lambda_3}$$

$$c_s = \frac{3\lambda_3}{\lambda_1 + \lambda_2 + \lambda_3}$$
(7)

The sum of three metrics c_l , c_p , c_s equals one and determines the barycentric parameterization of a triangular region with extreme values of linear, planar, and spherical shapes in the three corners.

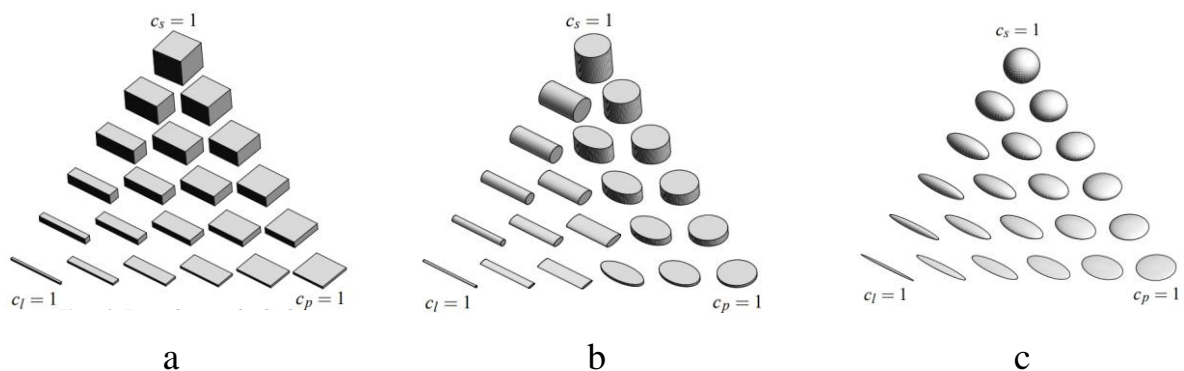


Figure 3.6 - Visualization of the stress tensor at different values c_l , c_p , c_s :

a - cubes, b - cylinders, c - ellipsoids

To implement this idea, the Department of Dynamics and Strength of Machines of NTU "KhPI" developed the `glyph_visualization_lib` library (authors Ruslan Babudzhan, Oleksiy Vodka). This library supports two options for visualization:

- Using `matplotlib` (slower, but does not require additional dependencies);
- Using `mayavi` (faster option).

To install `mayavi`, you need to install the `mayavi` and `pyqt5` packages.

```
pip install mayavi pyqt5
```


Example. Create a visualization of the tensor field:

$$\sigma = \begin{pmatrix} -\sin x & 5\sin(x+y) & 10\sin(xz) \\ & -\sin y & 3\sin(y+z) \\ \text{symm} & & \cos(z) \end{pmatrix} \quad (8)$$

The code that performs the visualization is shown in Listing 3.5. For the code to work, you need to download the `glyph_visualization_lib.py` file and add it to the project directory. The result of visualization is shown in Fig. 3.7.

```
import numpy as np
import matplotlib.pyplot as plt

from mayavi import mlab
import glyph_visualization_lib as gvl

def main():
    x = np.linspace(0, np.pi, 8, dtype=float, endpoint=True)
    y = np.linspace(-np.pi, -1, 8, dtype=float, endpoint=True)
    z = np.linspace(0, 2 * np.pi, 8, dtype=float, endpoint=True)

    X, Y, Z = np.meshgrid(x, y, z)

    stress_tensor = np.array([
        [-np.sin(X), 5 * np.sin(X + Y), 10 * np.sin(X * Z)],
        [5 * np.sin(X + Y), -np.sin(Y), 3 * np.sin(Y + Z)],
        [10 * np.sin(X * Z), 3 * np.sin(Y + Z), np.cos(Z)]
    ])

    print(stress_tensor.shape)

    vm_stress = gvl.get_von_Mises_stress(stress_tensor)

    glyph_radius = 0.25
    limits = [np.min(vm_stress), np.max(vm_stress)]
    colormap = plt.get_cmap('rainbow', 120)
    fig = mlab.figure(bgcolor=(1, 1, 1))

    fig2 = plt.figure()
    ax = fig2.add_subplot(111, projection='3d')

    for i in range(x.size):
        for j in range(y.size):
            for k in range(z.size):
                center = [x[i], y[j], z[k]]
                data = stress_tensor[:, :, i, j, k]
```

```

        color =
colormap(gvl.get_colormap_ratio_on_stress(vm_stress[i, j, k],
limits))[:3]

        """
        glyph_type = {0: 'cuboid', 1: 'cylinder', 2:
'ellipsoid', 3: 'superquadric'}

        if glyph_type == 3 (superquadric)
        there are glyph shape type
        0 - superquadrics,
        1 - Kindlmann_glyph,
        2 - Kindlmann_modified_glyph

        """

        x_g, y_g, z_g = gvl.get_glyph_data(center, data,
limits, glyph_points=12, glyph_radius=glyph_radius,
                                                glyph_type=3,
superquadrics_option=2)

        # surf = ax.plot_surface(x_g, y_g, z_g,
linewidth=0, antialiased=True, color=color)
        # surf = ax.plot_wireframe(x_g, y_g, z_g,
linewidth=1, antialiased=True, color=color)

        mlab.mesh(x_g, y_g, z_g, color=color)

        mlab.move(forward=1.8)
        mlab.savefig("superquadric-Kindlmann_modified-viz.png",
size=(100, 100))
        mlab.show()

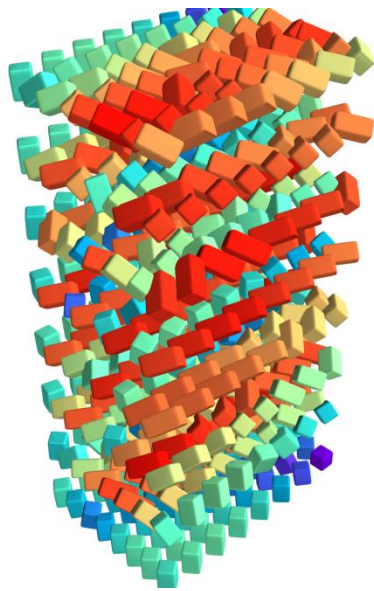
        # plt.show()

pass

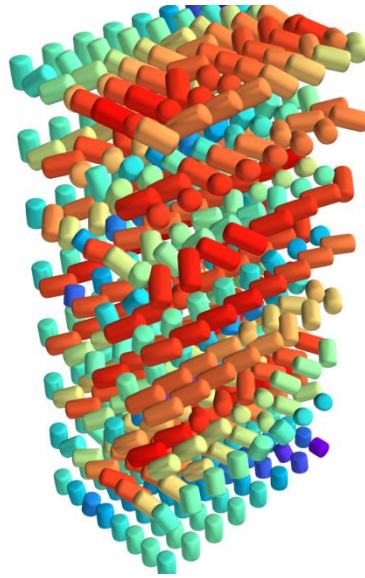
if __name__ == '__main__':
    main()

```

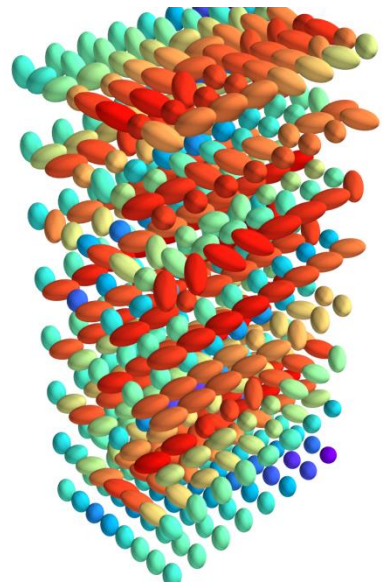
Listing 3.5 - Visualization of a tensor field



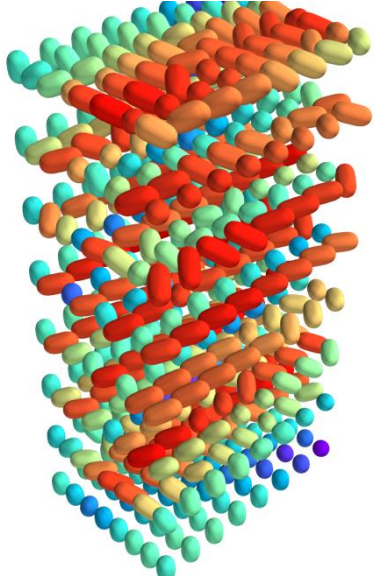
a. cuboid



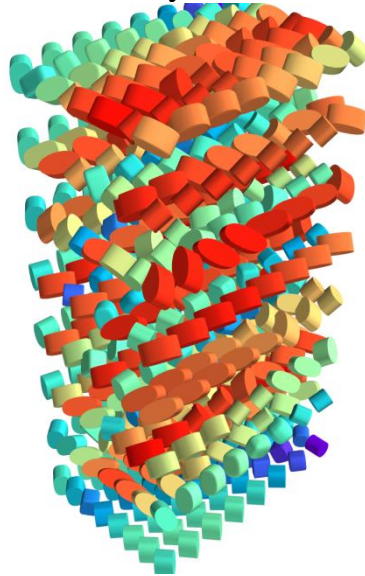
b. cylinder



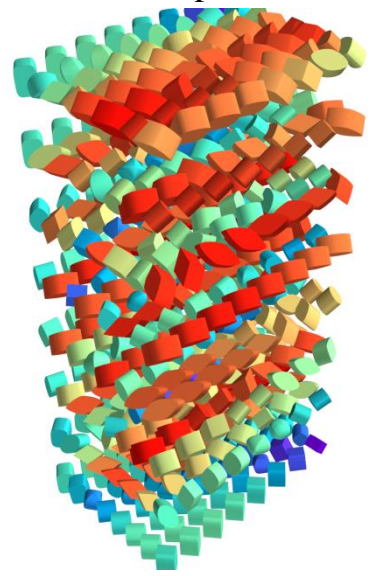
c. ellipsoid



d. superquadrature



e. Kindelman
superquadrature



f. modified Kindelman
superquadrature

Figure 3.7 - Visualization of a tensor field

Task. Build according to your number in group list:

1. Visualize the scalar field. Find its gradient and visualize it as a flat vector field;
2. Visualize a plane vector field both with the help of vectors and current lines from the matplotlib library and with the code from the listing;
3. Construct a three-dimensional visualization of the vector field; For an additional point (optional), modernize the algorithm for constructing current lines in the case of a 3-dimensional field.
4. Construct a visualization of a tensor field using ellipsoids, cuboids, cylinders, and any superquadrant.

Student 1.

1. $u(x, y) = 4 \ln(x^2 + y^2) - 8xy; x, y \in [-5; 3]$

2. $F = (x^2 - 2y; y^2 - 2x); x, y \in [-5; 3]$

3. $F = \left(\frac{y}{z} + 2x; \frac{x}{z}; \frac{xy}{z^2} \right); x, y, z \in [-5; 3]$

4. $T = \begin{pmatrix} x & \ln(xy) & -\ln(xz) \\ & -y & \ln(yz) \\ & & z \end{pmatrix}, x, y, z \in [1..10]$

Student 2.

1. $u(x, y) = x\sqrt{y} + y\sqrt{x}; x, y \in [0; 5]$

2. $F = (x^2 + 2y; y^2 + 2x); x, y \in [-4; 4]$

3. $F = \left(\frac{yz}{x^2 + y^2 + z^2}; \frac{xz}{x^2 + y^2 + z^2}; \frac{xy}{x^2 + y^2 + z^2} \right); x, y, z \in [-3; 4]$

4. $T = \begin{pmatrix} \sin(x) & x + y & x + z \\ & \cos(y) & y + z \\ & & \cos(z) \end{pmatrix}, x, y, z \in [-2\pi..2\pi]$

Student 3.

1. $u(x, y) = -2 \ln(x^2 + 5) - 4xy; x, y \in [-3; 6]$

$$2. F = (x + 2y^2; y + 2x^2); x, y \in [-3; 6]$$

$$3. F = \left(y - \frac{z}{x^2}; x + \frac{1}{z}; \frac{1}{x} - \frac{y}{z^2} \right); x, y, z \in [-3; 6]$$

$$4. T = \begin{pmatrix} \exp(-x^2) & \sin(x+y) & \sin(x+z) \\ & \exp(-y^2) & \sin(y+z) \\ & & \exp(-z^2) \end{pmatrix}, x, y, z \in [-2..2]$$

Student 4.

$$1. u(x, y) = xy^2 - \sqrt{x^3 y}; x, y \in [0; 8]$$

$$2. F = (x^3; -y^3); x, y \in [-8; 8]$$

$$3. F = \left(\frac{x^2}{x^3 + y^3 + z^3}; \frac{y^2}{x^3 + y^3 + z^3}; \frac{z^2}{x^3 + y^3 + z^3} \right); x, y, z \in [-7; 7]$$

$$4. T = \begin{pmatrix} \frac{\sin(x+y)}{x^2 + y^2 + z^2} & x & y \\ & \frac{\cos(y+z)}{x^2 + y^2 + z^2} & z \\ & & \frac{\cos(x+z)}{x^2 + y^2 + z^2} \end{pmatrix}, x, y, z \in [-2..2]$$

Student 5.

$$1. u(x, y) = x\sqrt{y} - yx^2; x, y \in [0; 9]$$

$$2. F = (x + y; x - y); x, y \in [-9; 9]$$

$$3. F = \left(\frac{1}{y} - \frac{z}{x^2}; \frac{1}{z} - \frac{x}{y^2}; \frac{1}{x} - \frac{y}{z^2} \right); x, y, z \in [-9; 9]$$

$$4. T = \begin{pmatrix} \frac{\ln(x)}{x^2 + y^2 + z^2} & \sqrt{x} & \sqrt{y} \\ & \frac{\ln(y)}{x^2 + y^2 + z^2} & \sqrt{z} \\ & & \frac{\ln(z)}{x^2 + y^2 + z^2} \end{pmatrix}, x, y, z \in [1..5]$$

Student 6.

$$1. u(x, y) = 7 \ln(x^2 + \frac{1}{13}) - 4 \sin(xy); x, y \in [-10; 10]$$

$$2. F = (x^2 y; -y); x, y \in [-10; 10]$$

$$3. F = \left(\frac{x+z}{x^2}; \frac{1}{y}; \frac{1}{z} \right); x, y, z \in [-10; 10]$$

$$4. T = \begin{pmatrix} \frac{\ln(x)}{\sin(x)} & \sqrt{x}/y & \sqrt{y}/z \\ & \frac{\ln(y)}{\sin(y)} & \sqrt{z}/x \\ & & \frac{\ln(z)}{\sin(z)} \end{pmatrix}, x, y, z \in [1..3]$$

Student 7.

$$1. u(x, y) = \arcsin\left(\frac{2y}{x^2 + y^2}\right); x, y \in [-\pi/2; \pi/2]$$

$$2. F = (2xy - y; x^2 + x); x, y \in [-11; 11]$$

$$3. F = \left(\frac{2x}{y} + 1; \frac{x^2}{y^2}; -6z^2 \right); x, y, z \in [-11; 11]$$

$$4. T = \begin{pmatrix} 1 & \sin(x)/x & \sin(y)/y \\ & 1 & \sin(z)/z \\ & & 1 \end{pmatrix}, x, y, z \in [-3..3]$$

Student 8.

$$1. u(x, y) = \arcsin\left(\frac{2(x+y)}{x^2 + y^2}\right); x, y \in [-\pi/2; \pi/2]$$

$$2. F = (\sin(x); \sin(y); \sin(z)); x, y, z \in [-2\pi; 2\pi]$$

$$3. F = \left(\frac{x+z}{x^2}; \frac{y+x}{y^2}; \frac{z+x}{z^2} \right); x, y, z \in [-11; 11]$$

$$4. T = \begin{pmatrix} \frac{x}{x^2 + y^2 + z^2} & \frac{x+y}{z} & \frac{x+z}{y} \\ & \frac{y}{x^2 + y^2 + z^2} & \frac{y+z}{x} \\ & & \frac{z}{x^2 + y^2 + z^2} \end{pmatrix}, x, y, z \in [1..5]$$

Student 9.

$$1. u(x, y) = \frac{1}{4}x^2y - \sqrt{x^2 + 5y^2}; x, y \in [-2; 7]$$

$$2. F = (x + y; 2x); x, y \in [-2; 7]$$

$$3. F = \left(\frac{x}{(x^2 + y^2 + z^2)^3}; \frac{y}{(x^2 + y^2 + z^2)^3}; \frac{z}{(x^2 + y^2 + z^2)^3} \right); x, y, z \in [-2; 7]$$

$$4. T = \begin{pmatrix} \frac{x}{x^2 + y^2 + z^2} & x & y \\ & \frac{y}{x^2 + y^2 + z^2} & z \\ & & \frac{z}{x^2 + y^2 + z^2} \end{pmatrix}, x, y, z \in [-2..2]$$