E3S Web of Conferences **533**, 03013 (2024) https://doi.org/10.1051/e3sconf/202453303013
FORM-2024

Analysis of Plates and Shells with Py-PDE

Alexander Suvorov*, *Irina* Makarova

Moscow State University of Civil Engineering, 26, Yaroslavskoye shosse, Moscow, 129337, Russia

Abstract. In this article we solve a few problems of shell and plate theory with py-pde package. The shell has a shape of elliptic paraboloid. Only membrane shell theory is considered. Both shell and plate have a rectangular boundary and are subjected to the uniform vertical load. The paper includes distributions of internal forces in the shell and bending moments in the plate obtained with the help of the py-pde program.

Keywords. Plate theory, membrane shell theory, python, internal forces, bending moments, elliptic paraboloid, differential equation

1 Introduction

In this paper we solve a few problems of shell and plate theory using the Python py-pde package [1]. This package is suited for solving certain parabolic or elliptic partial differential equations. The unknown function can depend on one space variable or two space variables and time. In case of two space variables, the problem can be solved only for a rectangular region. Systems of differential equations with more than one unknown functions can be analyzed as well and this capability will be illustrated in this paper.

This package solves the differential equation in this form

$$\frac{\partial u}{\partial t} = rhs$$

where rhs is the right-hand side of the equation that includes derivatives of the unknown function with respect to the space variables. Elliptic problems do not have explicit time dependence but the solution to the elliptic problem can be obtained by using the same general framework and assuming that the time is sufficiently large and therefore, the solution ceases to depend on time [2]. The fictitious initial condition must be prescribed in this case and the final time value must be properly selected to ensure the time-invariance of the solution.

This package uses two major classes. The class CartesianGrid is used for creating the Cartesian grid. The dimensions of the region and the number of points along the sides of the region must be specified. The second class is ScalarField. This is the class using which we can define an initial condition, for example, and the final solution obtained at the specified

*Corresponding author: suvorovap@mgsu.ru

value of time is also stored as the scalar field. It is important to note that for creation of the scalar field, one must specify the grid on which this field is defined.

The py-pde package contains many examples in its documentation [1] but authors thought that providing a few examples directly related to structural mechanics, mechanics of plates and shells is useful.

2 Membrane Analysis of a Shell

Consider a shell in the form of an elliptical paraboloid that covers a rectangular area with dimensions $2a$ along the x axis and $2b$ along the y axis. Equation for the shape of this shell is given by

$$z = f\left(\frac{x^2-a^2}{2a^2} + \frac{y^2-b^2}{2b^2}\right)$$

It is clear that f measures the rise of the shell in the center.

The shell is subjected to a vertical load with intensity p, which is defined per unit area of the projection of the shell surface onto the horizontal plane xy.

Membrane forces N_1, N_2, and S act in the cross-section of the shell. Consider the section of the shell by the plane x=*const*. In this section, there is a force N_1, which is parallel to the xz plane, and a force S, which is perpendicular to this plane. If we consider the section of the shell by the plane y=*const*, then the force N_2, parallel to the plane yz, and the force S, which is perpendicular to this plane, will act in this section. We note that these forces are defined per unit length of the sides of the shell piece, cut off by the planes x=*const*, y=*const*.

Next, we will consider the projections of these forces on the horizontal plane xy and, at the same time, we will assume that these projected forces act not on the sides of the shell piece cut off by the x=*const*, y=*const* planes, but on the sides of the projection of the shell piece on the xy plane. Let us denote these new forces as $\underline{N}_1$, $\underline{N}_2$, $\underline{S}$.

True forces are related to these new forces by

$$N_1 = \underline{N}_1\sqrt{\frac{1+\left(\frac{dz}{dx}\right)^2}{1+\left(\frac{dz}{dy}\right)^2}}$$

$$N_2 = \underline{N}_2\sqrt{\frac{1+\left(\frac{dz}{dy}\right)^2}{1+\left(\frac{dz}{dx}\right)^2}}$$

$$S = \underline{S}$$

Let us define the force function F

$$\underline{N}_1 = \frac{\partial^2 F}{\partial y^2}, \qquad \underline{N}_2 = \frac{\partial^2 F}{\partial x^2}, \qquad \underline{S} = -\frac{\partial^2 F}{\partial x\,\partial y}$$

Then it is possible to show that this force function F satisfies the following equation [3]

$$\underline{N}_1 \frac{d^2 z}{dx^2} + 2\underline{S} \frac{d^2 z}{dx\, dy} + \underline{N}_2 \frac{d^2 z}{dy^2} =- p$$

Substituting the equation for the surface of an elliptic paraboloid *z=f(x,y)* given above and expressing the forces in terms of the force function, we obtain the following equation

$$\frac{\partial^2 F}{\partial y^2}\frac{f}{a^2} + \frac{\partial^2 F}{\partial x^2}\frac{f}{b^2} =- p$$

or

$$\frac{\partial^2 F}{\partial y^2} b^2 + \frac{\partial^2 F}{\partial x^2} a^2 =- \frac{p}{f}\, a^2 b^2$$

Consider boundary conditions. We assume that the shell rests on sides $x=\pm a$, $y=\pm b$ on flat diaphragms, which are stiff only for deformations in the plane of the diaphragm, and compliant for deformations in the perpendicular direction.

Then the boundary conditions can be written as

$$x = \pm a, \quad \underline{N}_1 = \frac{\partial^2 F}{\partial y^2} = 0$$

$$y = \pm b, \quad \underline{N}_2 = \frac{\partial^2 F}{\partial x^2} = 0$$

Note that these boundary conditions are not perfect. In particular, if $\underline{N}_2 = 0$ on the edge $y = b$, then $N_2 = 0$. This means that at the edge $y = b$ there will be only one non-zero component of the internal forces acting along the *x*-axis. This component is equal to S. Therefore, at all edges, the vertical components of the support reactions will be equal to zero, and only the horizontal forces $\underline{S} = S$ will be non-zero. If all vertical components of the support forces are zero, it will be impossible to satisfy the equilibrium equation in the direction of the *z*-axis. This is a disadvantage of the membrane theory of shells, since in this case we obtain a solution with an error and it is not possible to determine the support reactions.

The mentioned boundary conditions mean that the force function can only be linear along the shell sides $x = \pm a$, $y = \pm b$. Therefore, this force function can be modified by adding a bilinear function such that the new force function is equal to zero on the sides of the shell. This new function will also satisfy the governing equation

$$\frac{\partial^2 F}{\partial y^2} b^2 + \frac{\partial^2 F}{\partial x^2} a^2 =- \frac{p}{f}\, a^2 b^2,$$

but will be equal to zero on the boundary. Therefore, to solve the problem, it is sufficient to consider zero boundary conditions

$$F = 0, \quad x = \pm a, \quad y = \pm b.$$

After this function has been found we can find projected forces $\underline{N}_1$, $\underline{N}_2$, $\underline{S}$, and then true forces N_1, N_2, S.

As an example, consider a shell covering rectangular area with sides $2a = 2$, $2b = 1$ and with the rise $f = 1$. Let the load intensity be $p = 2$.

The program code can be written as follows:

```
from pde import PDE, ScalarField, CartesianGrid
grid = CartesianGrid([[-1., 1.], [-0.5, 0.5]], [64,32])  # generate the grid
bc_x = [{"value_expression": f"0" }, {"value_expression": f"0"}]
bc_y = [{"value_expression": f"0" } ,  {"value_expression": f"0"}]
eq = PDE ({"u": f" 1.**2*d2_dx2(u) + 0.25*d2_dy2(u) + 2*1.**2*0.25"}, bc = [bc_x, bc_y ])
state = ScalarField(grid, 0.0)
res = eq.solve(state, t_range=500000e-6, method="scipy")
res.plot(cmap="magma")
```

For finding the internal forces we can use the second derivative operators defined for each field

```
N2_ = res.apply_operator('d2_dx2', bc = [bc_x, bc_y ])
N1_ = res.apply_operator('d2_dy2', bc = [bc_x, bc_y ])
N1_.plot(cmap="magma")
N2_.plot(cmap="magma")
```

Also we can construct equations for the surface of the shell and its derivatives along the axes *x* and *y* considering them as a scalar field

```
z = ScalarField.from_expression(grid, '(x-1)*(x+1)/2 + (y-0.5)*(y+0.5)/(2*0.5**2)')
dzdx = ScalarField.from_expression(grid, 'x')
dzdy = ScalarField.from_expression(grid, 'y/(0.5**2)')
```

After finding these expressions it is possible to find true internal forces

```
N2 = N2_*np.sqrt((1+dzdy**2)/(1+dzdx**2))
N1 = N1_*np.sqrt((1+dzdx**2)/(1+dzdy**2))
N1.plot(cmap="magma")
N2.plot(cmap="magma")
```

Note that the true forces may differ significantly from the projected forces if the shell rise is large. Figure 1 shows projected force $\underline{N}_1$ (top) and true force N_1 (bottom). Note that the compressive forces N_1 take the largest value on the sides of the shell $y = -0.5$, $y = 0.5$. The true values turn out to be *smaller* than the projected values, which seems to be counterintuitive. But this is possible to explain if we remember that the force $\underline{N}_1$ is defined per unit length of the *projection* of the shell section by the plane *x=const*, which is shorter than the true length of the shell section side *x=const*. It is also important to note that the forces N_2 are smaller in magnitude than the forces N_1, and take the largest values on the sides of the shell $x = \pm 1$. This means that the shell mainly works in the direction of its longer side, i.e., the *x*-axis in this example.

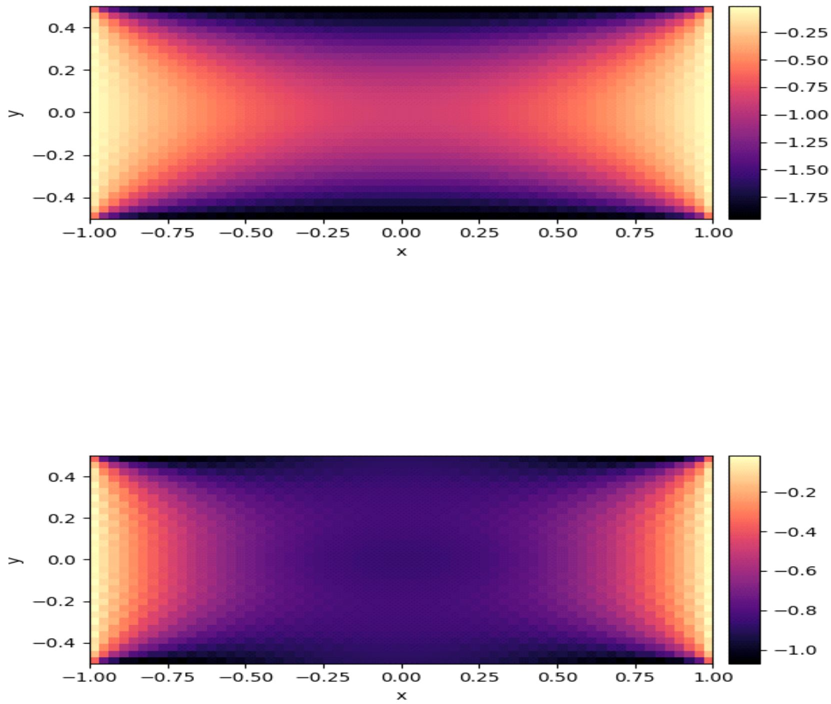

Fig. 1. Membrane forces $\underline{N}_1$ (top) and N_1 (bottom) in the shell with the shape of an elliptical paraboloid subjected to uniform vertical load of intensity $p = 2$

Of course, we can also prescribe the non-uniform load *p(x,y)* acting on a given shell.

3 Plate Analysis

In this section we analyze the rectangular plate with the sides *a* and *b* along the axes *x* and y, respectively. The plate lies in the plane *xy* and covers the area $-\frac{a}{2} \leq x \leq \frac{a}{2}$, $-\frac{b}{2} \leq y \leq \frac{b}{2}$. Assume that the plate is simply-supported on all four sides. A distributed transverse load of intensity q is acting on the plate. The objective is to find the deflection of the plate *w(x,y)*, and also bending moments $M_x(x,y)$, $M_y(x,y)$.

Introduce the stiffness of the plate

$$D = \frac{Eh^3}{12(1-\nu^2)}$$

Here h is the thickness of the plate, E – modulus of elasticity of the plate material, ν – Poisson's ratio.

From the bending theory of plates the differential equation for the deflection is given by [4,5]

$$\frac{\partial^4 w}{\partial x^4} + 2\frac{\partial^4 w}{\partial x^2 \partial y^2} + \frac{\partial^4 w}{\partial y^4} = \frac{q}{D}.$$

The bending moments can be found using the formulas

$$M_x = D\left(\frac{\partial^2 w}{\partial x^2} + \nu\frac{\partial^2 w}{\partial y^2}\right)$$
$$M_y = D\left(\frac{\partial^2 w}{\partial y^2} + \nu\frac{\partial^2 w}{\partial x^2}\right)$$

By introducing the function

$$M = M_x + M_y$$

we can represent one differential equation of the fourth order as a system of two differential equations of the second order

$$D(1 + \nu)\Delta w - M = 0,$$
$$\Delta M - (1 + \nu)q = 0$$

Here Δ is the Laplace operator.

For the simply-supported plate the boundary conditions can be written as follows

$$w = 0, \quad x = \pm\frac{a}{2}, \quad y = \pm\frac{b}{2}$$
$$M_x = 0, \quad x = \pm\frac{a}{2}$$
$$M_y = 0, \quad y = \pm\frac{b}{2}.$$

We thus have two boundary conditions on each edge of the plate. From the definition of the bending moments and due to the fact that the function w itself is zero at the boundary of the region, stronger boundary conditions can be imposed

$$\frac{\partial^2 w}{\partial x^2} = 0, \quad \frac{\partial^2 w}{\partial y^2} = 0, \quad x = \pm\frac{a}{2}, \ y = \pm\frac{b}{2}.$$

Thus the function M itself satisfies the zero boundary conditions

$$M = 0, \quad x = \pm\frac{a}{2}, \ y = \pm\frac{b}{2}.$$

Thus, the problem is reduced to finding two functions w and M, which are equal to zero on the boundary of the plate and satisfy the system of differential equations written above.

Let us present the program code for this problem

```
from pde import FieldCollection, PDE, ScalarField, CartesianGrid
q=1
E=10920
h=0.1
nu=0.3
D=E*h**3/(12*(1-nu**2))
a = 2
b = 1
grid = CartesianGrid([[-1., 1.], [-0.5, 0.5]], [65,33])  # generate grid
bc_x = [{"value_expression": f"0" }, {"value_expression": f"0"}]
bc_y = [{"value_expression": f"0" } ,  {"value_expression": f"0"}]
eq = PDE(    {        "w": f"laplace(w)*({D}*(1+{nu})) - M",
       "M": f"laplace(M) - (1+{nu})*{q}",    },
    bc = [bc_x, bc_y ])
w = ScalarField(grid, 0.0)
M = ScalarField(grid, 0.0)
state = FieldCollection([w, M])
res = eq.solve(state, t_range=500000e-6, method="scipy")
w = res[0]
```

In this example we analyze the plate with the sides $a = 2$, $b = 1$, which is subjected to a uniform load $q = 1$. The action of the non-uniform load *q(x,y)* can also be considered. With the knowledge of the deflection, we can find the bending moments using their definitions

```
wx2 = w.apply_operator('d2_dx2', bc = [bc_x, bc_y ])
wy2 = w.apply_operator('d2_dy2', bc = [bc_x, bc_y ])
Mx = D*(wx2 + nu*wy2)
My = D*(wy2 + nu*wx2)
Mx.plot()
My.plot()
plt.plot(Mx.data[:,16])
plt.show()
```

Figure 2 shows the distribution patterns of bending moments when the Poisson’s ratio is equal to zero. At the top the bending moment M_x is shown, at the bottom M_y.

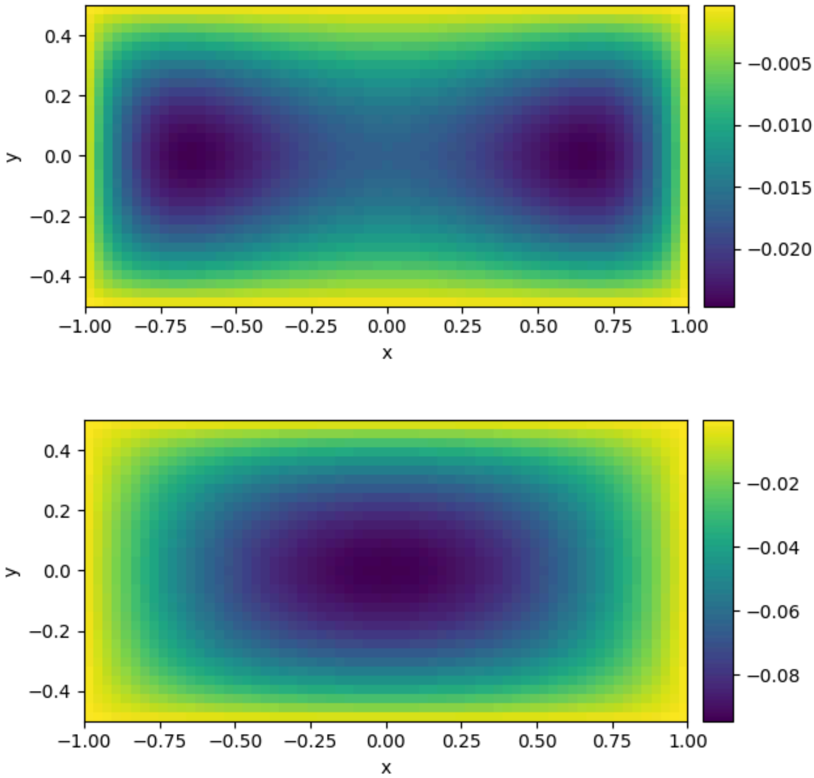

Fig. 2. The distributions of bending moments M_x (top) and M_y (bottom) for rectangular plate with Poisson's ratio equal to zero subjected to a transverse uniformly distributed load of intensity $q = 1$

Note one interesting feature of the solution. It lies in the fact that the bending moment M_x is considerably smaller than the bending moment M_y. This is because the side along the x-axis is larger than the side along the y-axis. In bending, the plate mainly works along the shorter side. Along the shorter side the value of the bending moment M_y (0.01) is not so far from the value obtained from the simple beam theory

$$M_y^b = \frac{qb^2}{8} = 0.125$$

However this formula cannot be used for estimating the bending M_x along the longer side of the plate, but it is evident that M_x is smaller than the moment M_y.

From the top figure it can be observed that the maximum value of the bending moment M_x occurs not in the center of the plate, but approximately at points with the coordinates $x = \pm 0.7$. This is an interesting feature of the solution, which however becomes less significant if the Poisson's ratio is increased to more realistic values.

4 Conclusions

In this work we analyze a few problems of shell and plate theory with the help of py-pde package. This package can be used for solving differential equations on a segment in one-dimensional space or on a rectangular domain in 2D space. It requires specification of the differential equations to be solved, boundary conditions and initial condition. If an elliptic equation must be solved without the time dependence, then the initial condition does not have a physical meaning but must be prescribed anyway due to the program requirements.

In the analysis of the shell we considered only the membrane theory. The shell is in the form of an elliptic paraboloid and located on a rectangular domain. We showed that under the action of the uniform load, the shell mainly works in the direction of its longer side since in this direction the internal forces are higher.

When analyzing the plate, we considered the rectangular plate under the action of the uniform load. The plate is simply-supported on all four sides. We showed that under the action of the uniform load, the plate mainly works in the direction of its shorter side since in this direction the bending moment is higher. Furthermore, the bending moment in the direction of the shorter side can be estimated quite accurately by the simple beam theory.

The program requires the specification of the final time at which solution is calculated. For pure elliptic problems this is not very convenient because the time variable does not enter the problem formulation. But it is important to select this final time value properly (sufficiently high) since the solution must be fully settled (stabilized) and independent on time.

It is important to mention that not all possible boundary conditions that encounter in plate and shell theory can be analyzed using this software. Therefore, this package cannot be considered as a universal solver unlike many finite element programs. In addition, the boundary conditions are not exactly satisfied but become more accurate when the grid size is increased.

The authors are grateful to the Department of Computer Science and Applied Mathematics for providing computer facilities.

References

1. Documentation to the py-pde package, https://py-pde.readthedocs.io/en/latest/
2. V. E. Zaliznyak. Numerical methods. Fundamentals of scientific computing, Moscow, Urait publishing house, 2023, https://urait.ru/bcode/510699
3. E. Ventsel, T. Krauthammer. Thin Plates and Shells. Theory, Analysis, and Applications. Marcel Dekker, Inc. New York, Basel, 2001.
4. V. Monakhov. Theory of plates and shells, Penza, Penza State University of Civil Engineering and Architecture, 2016.
5. S. Timoshenko, S. Woinowsky-Krieger. Theory of plates and shells, McGraw-Hill Book Company, Inc., 1987.