Checking a concrete floor by the finite difference method

Exercise for course CEGM1000-25 Modelling, uncertainty, and data analysis for engineers (MUDE)

P.C.J. Hoogenboom, 27 August 2025

In this exercise, we model a reinforced concrete floor in Python with the finite difference method. At the end, we conclude that the proposed reinforcement is sufficient and this floor can be built. The exercise takes about 2 hours. After this exercise, you are invited to calculate and check the floor you are standing on.

1. Notation

l = 7 m	plate span in both directions
t = 280 mm	plate thickness
E = 30 000 N/mm^2	Young's modulus of concrete
ν = 0.15	Poisson's ratio of concrete
p = 9.7 kN/m^2 (SLS) or 13.5 kN/m^2 (ULS)	evenly distributed load
x, y, z	coordinate system
w	plate deflection [mm]
m_{xx} , m_{yy} , m_{xy}	plate moments [kNm/m]
v_x , v_y	plate shear forces [kN/m]
f = 500 N/mm^2 (SLS) or 435 N/mm^2 (ULS)	rebar yield strength
m_{tx} , m_{ty} = 68 kNm/m (ULS)	top rebar moment capacity (negative z axis)
m_{bx} , m_{by} = 68 kNm/m (ULS)	bottom rebar moment capacity (positive z axis)
D = E*t**3/(12*(1-ν**2))	plate stiffness
h = l/47	grid line spacing

2. Equations

A reinforced concrete floor can be modelled as a linear elastic plate. The differential equation is

$$p = D\left(\frac{\partial^4 w}{\partial x^4} + 2\frac{\partial^4 w}{\partial x^2 \partial y^2} + \frac{\partial^4 w}{\partial y^4}\right) \qquad (1)$$

This equation will be derived in the course CIEM5110 [1]. The floor is simply supported at 2 edges, fixed at 1 edge and free at 1 edge (fig. 1).

The boundary conditions are

$$\begin{aligned} &y = 0, \quad w = 0, \quad \frac{\partial w}{\partial y} = 0 \\ &x = 0, \quad w = 0, \quad m_{xx} = 0 \\ &x = l, \quad w = 0, \quad m_{xx} = 0 \\ &y = l, \quad m_{yy} = 0, \; v_y + \frac{\partial m_{xy}}{\partial x} = 0 \end{aligned} \qquad (2)$$

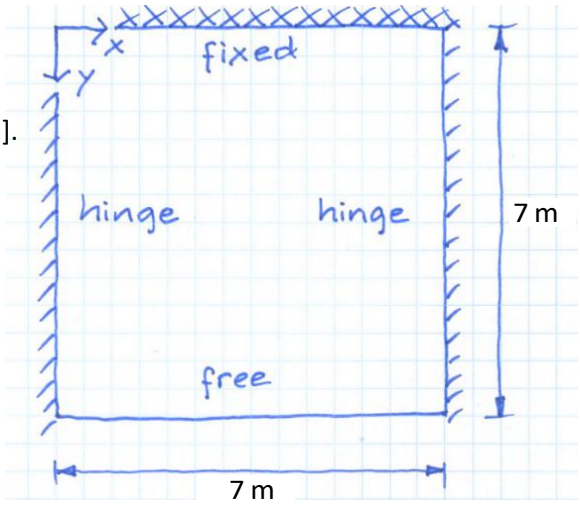

Figure 1. Structural idealisation of the floor

The latter boundary condition, for the free edge, is not obvious. Just accept it and ask the teacher when you take the plate course.[1]

When the differential equation is solved, the moments are determined by

$$
\begin{aligned}
m_{xx} &= -D\left(\frac{\partial^2 w}{\partial x^2} + \nu\frac{\partial^2 w}{\partial y^2}\right) \\
m_{yy} &= -D\left(\frac{\partial^2 w}{\partial y^2} + \nu\frac{\partial^2 w}{\partial x^2}\right) \\
m_{xy} &= -D(1-\nu)\frac{\partial^2 w}{\partial x \partial y}
\end{aligned}
\tag{3}
$$

The unit of a plate moment is kNm/m (fig. 2).

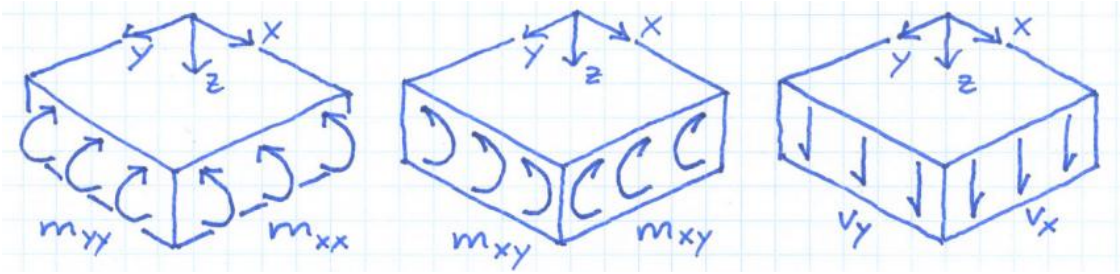

Figure 2. Moments and shear forces in a plate part

The shear forces are determined by

$$
\begin{aligned}
v_x &= \frac{\partial m_{xx}}{\partial x} + \frac{\partial m_{xy}}{\partial y} \\
v_y &= \frac{\partial m_{yy}}{\partial y} + \frac{\partial m_{xy}}{\partial x}
\end{aligned}
\tag{4}
$$

The unit of a plate shear force is kN/m (fig. 2).

[1] Equation 1 was first derived by Sophie Germain and Joseph-Louis Lagrange in 1813 [2 p. 119]. Germain had worked on this for many years and it was a great accomplishment. Subsequently, scientists calculated in what shapes plates vibrate but the results did not agree with experiments. There was a problem and nobody knew what it was. In 1850, Gustav Kirchhoff solved the problem by deriving the correct boundary conditions for free edges [3]. This is why we call it the *Kirchhoff plate theory*.

Wikipedia calls it the Kirchhoff-Love plate theory. However, Augustus Love did not contribute. He was born in 1863, long after Kirchhoff had solved all problems. Love did propose the first shell theory in 1888 [4], based on the same assumptions as Kirchhoff. Several books are on both plates and shells, which explains how the names got entangled. Love's assumptions are correct but his shell theory has several problems, which were solved by Lyell Sanders and Warner Koiter in 1959 [5].

3. Floor load

Floor load consists of self-weight, light walls, floor covering, furniture and people. Self-weight is

$$2400 \text{ kg/m}^3 \times 10 \text{ N/kg} \times 0.28 \text{ m} = 6.7 \text{ kN/m}^2 \quad (5)$$

The safety factor on self-weight is 1.35 [6 p. 54]. The other loads together are 3 kN/m^2 [7 p. 22]. The safety factor on the other loads is 1.5 [6 p. 54]. For the deflection (serviceability limit state) we do not use safety factors.

$$p = 6.7 + 3 = 9.7 \text{ kN/m}^2 \quad (6)$$

For strength (ultimate limit state), we do use the safety factors.

$$p = 6.7 \times 1.35 + 3 \times 1.5 = 13.5 \text{ kN/m}^2 \quad (7)$$

4. Deriving the molecule

In the finite difference method, we use a grid (fig. 3). At every grid point the displacement w of the plate is computed. The distance between the grid lines is h.

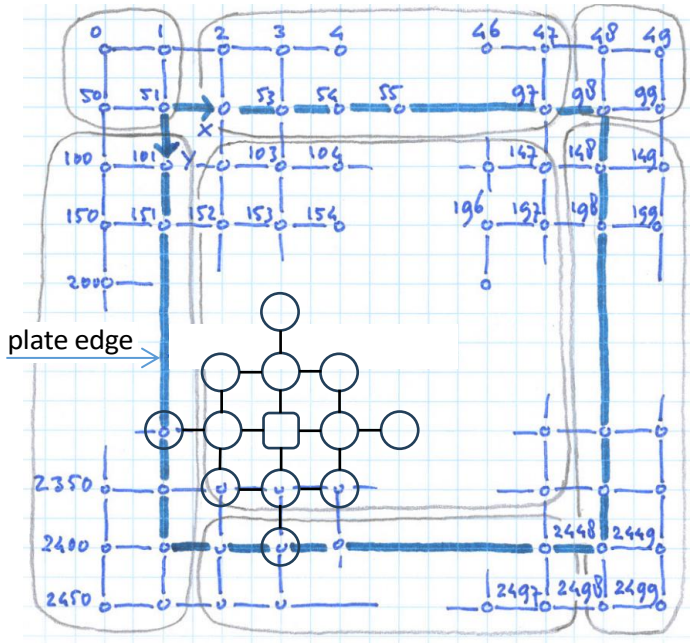

Figure 3. Finite difference grid

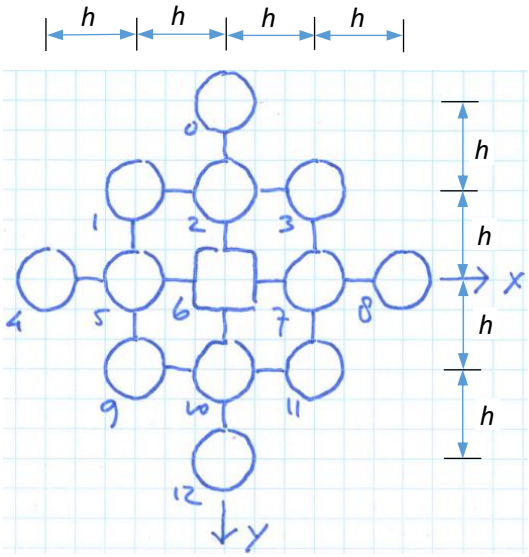

Figure 4. Finite difference molecule

Question 1: In structural mechanics we usually choose the z-axis down. Why would we do this? What consequence does this have for the y axis?

For every grid point we need the relation of adjacent displacements. We call this the molecule (fig. 4). Let's derive it. We pick a random grid point somewhere in the plate middle. We introduce a local coordinate system x-y with its origin in this point. We choose a displacement function.

```
inport sympy as sp
x,y = sp.symbols('x,y')
D,nu,h = sp.symbols('D,nu,h')
a0,a1,a2,a3,a4,a5,a6,a7,a8,a9,a10,a11,a12 = sp.symbols('a0,a1,a2,a3,a4,a5,a6,a7,a8,
a9,a10,a11,a12')
w0,w1,w2,w3,w4,w5,w6,w7,w8,w9,w10,w11,w12 = sp.symbols('w0,w1,w2,w3,w4,w5,w6,w7,w8,
w9,w10,w11,w12')

w = a0 + a1*x + a2*y + a3*x*x + a4*x*y + a5*y*y + a6*x*x*x + a7*x*x*y + a8*x*y*y +
a9*y*y*y + a10*x*x*x*x + a11*x*x*y*y + a12*y*y*y*y
```

The displacement *w* goes through the grid point displacements w_0 to w_{12} (fig. 4). This provides 13 equations.

```
eq0 = sp.Eq( w0, w.subs(x,   0).subs(y,-2*h))
eq1 = sp.Eq( w1, w.subs(x,  -h).subs(y, -*h))
eq2 = sp.Eq( w2, w.subs(x,   0).subs(y, -*h))
eq3 = sp.Eq( w3, w.subs(x,   h).subs(y, -*h))
eq4 = sp.Eq( w4, w.subs(x,-2*h).subs(y,   0))
eq5 = sp.Eq( w5, w.subs(x,  -h).subs(y,   0))
eq6 = sp.Eq( w6, w.subs(x,   0).subs(y,   0))
eq7 = sp.Eq( w7, w.subs(x,   h).subs(y,   0))
eq8 = sp.Eq( w8, w.subs(x, 2*h).subs(y,   0))
eq9 = sp.Eq( w9, w.subs(x,  -h).subs(y,   h))
eq10= sp.Eq(w10, w.subs(x,   0).subs(y,   h))
eq11= sp.Eq(w11, w.subs(x,   h).subs(y,   h))
eq12= sp.Eq(w12, w.subs(x,   0).subs(y, 2*h))
```

With the equations, we can solve the 13 polynomial coefficients a_0 to a_{12}.

```
a = sp.solve([eq0,eq1,eq2,eq3,eq4,eq5,eq6,eq7,eq8,eq9,eq10,eq11,eq12],a0,a1,a2,
a3,a4,a5,a6,a7,a8,a9,a10,a11,a12)
a0=a[a0]; a1=a[a1]; a2=a[a2]; a3=a[a3]; a4=a[a4]; a5=a[a5]; a6=a[a6]; a7=a[a7];
a8=a[a8]; a9=a[a9]; a10=a[a10]; a11=a[a11]; a12=a[a12]
```

We substitute the coefficients into the polynomial, and the polynomial into equation 1.

```
p = D*expand( diff(w,x,x,x,x) + 2*diff(w,x,x,y,y) + diff(w,y,y,y,y) )
```

The result is

$$p = D/(h^{**}4) * (w0 + 2*w1 - 8*w2 + 2*w3 + w4 - 8*w5 + 20*w6 - \\ - 8*w7 + w8 + 2*w9 - 8*w10 + 2*w11 + w12) \qquad (8)$$

This relates the grid displacements w_i to the load p on the centre molecule point. Figure 5 shows the complete molecule. We use this equation for every interior grid point. The free edge molecule is shown in figure 6. It can be derived in the same way.

Question 2: We can use a larger interpolation polynomial. What are advantages and disadvantages?

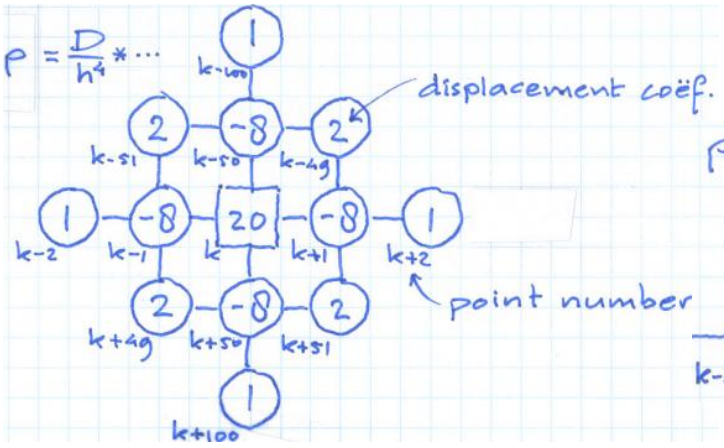

Figure 5. Internal molecule

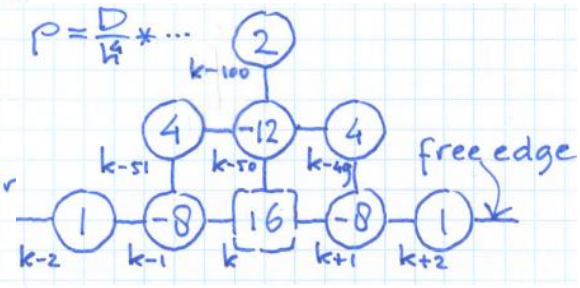

Figure 6. Free edge molecule

5. Writing the equations in the matrix

Some of the grid points are outside the plate (fig. 3). The displacements there are imaginary and determined by the boundary conditions. For example, on the fixed edge, the slope is zero, so the deformation is symmetrical over the edge, so the imaginary displacement outside the plate must be the same as the displacement just inside. For example, the boundary conditions of point 53 are `w53 = 0` and `w3 = w103`.

There are 50 × 50 = 2500 grid points. So, 2500 displacements and 2500 equations. The equation of a point follows from a molecule or a boundary conditions. The equations are written in a matrix *M*.

```
import numpy as np
from scipy.sparse import lil_matrix
from scipy.sparse.linalg import spsolve
M=lil_matrix((2500,2500))                # matrix with all equations
f=np.array([0.0 for i in range(2500)])   # {f} = [M]{w}, We solve {w}.
w=np.array([0.0 for i in range(2500)])

t = 280                          # mm
E = 30000                        # N/mm2
nu= 0.15                         # -
D = E*t**3/(12*(1-nu**2))        # Nmm
h = 7000/47                      # mm
p = 9.7/1000                     # N/mm2

row=-1
for i in range(2, 48):           # add internal molecules
   for j in range(2, 48):
      k=j*50+i
      row=row+1
      M[row,k-100]= 1
      M[row,k-51 ]= 2
      M[row,k-50 ]=-8
      M[row,k-49 ]= 2
      M[row,k-2  ]= 1
      M[row,k-1  ]=-8
      M[row,k    ]=20
      M[row,k+1  ]=-8
      M[row,k+2  ]= 1
      M[row,k+49 ]= 2
      M[row,k+50 ]=-8
```

```
      M[row,k+51 ]= 2
      M[row,k+100]= 1
      f[row]=p/D*(h**4)
for i in range(52, 98):      # add fixed edge
   row=row+1
   M[row,i]=1
   row=row+1
   M[row,i-50]= 1
   M[row,i+50]=-1
for j in range(2, 50):       # add left hinged edge
   row=row+1
   M[row,j*50+1]=1
   row=row+1
   M[row,j*50  ]=1
M[row,j*50+2]=1
for j in range(2, 50):       # add right hinged edge
   row=row+1
   M[row,j*50+48]=1
   row=row+1
   M[row,j*50+47]=1
   M[row,j*50+49]=1
for i in range(2402, 2448): # add bottom free edge molecules
   row=row+1
   M[row,i-100]=  2
   M[row,i-51 ]=  4
   M[row,i-50 ]=-12
   M[row,i-49 ]=  4
   M[row,i-2  ]=  1
   M[row,i-1  ]= -8
   M[row,i    ]= 16
   M[row,i+1  ]= -8
   M[row,i+2  ]=  1
   f[row]=p/D*(h**4)
   row=row+1
   M[row,i-50]=-1
   M[row,i   ]= 2
   M[row,i+50]=-1
row=row+1; M[row, 0]=1       # add top-left corner
row=row+1; M[row, 1]=1
row=row+1; M[row,50]=1
row=row+1; M[row,51]=1
row=row+1; M[row,48]=1       # add top-right corner
row=row+1; M[row,49]=1
row=row+1; M[row,98]=1
row=row+1; M[row,99]=1
```

6. Solving the matrix

NumPy solves the 2500 displacements *w* from the 2500 equations.

```
M=M.tocsr()
w=spsolve(M,f)
del M
del f
```

Surely, you solved a system of equations by hand once. So, you can appreciate that your computer is doing a lot of work for you.

7. Checking the deformation

Let's see the displacement in a contour plot.

```
import matplotlib.pyplot as plt
Z=[[w[50*j+i] for i in range(1,49)] for j in range(48,0,-1)]
plt.figure()
cp=plt.contourf(Z,10,cmap=plt.cm.jet)
plt.colorbar(cp)
plt.show()
```

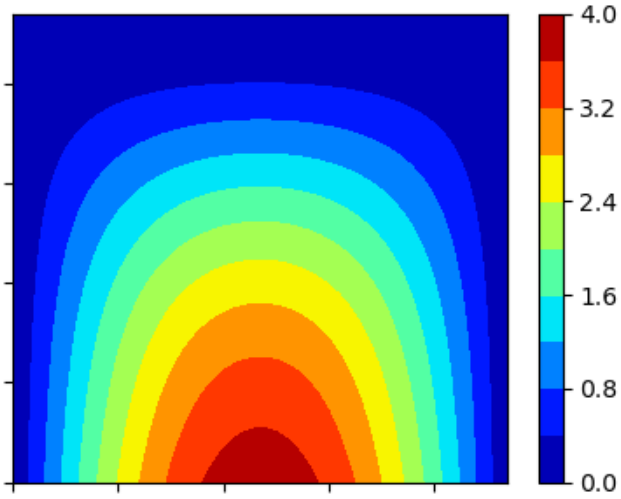

Figure 7. Computed deformation [mm]

First we check ourselves (fig. 7).
- Is the displacement at supports zero? ..... It is dark blue between 0 en 0.4 mm. Seems okay.
- Does the free edge displace? ..... It does. So, okay.
- Is the slope at the fixed edge zero? ..... We cannot see this but the slope is certainly smaller than at the hinged edges. Okay.
- Is the deflection in the direction of the load? .. Both are positive, so in positive z direction. Okay

The plate has also been computed by the finite element method (Appendix 2). The deflections are the same. Okay.

Now, we check our floor. The deflection is 4 mm. Concrete cracking will make this 3 times larger. Concrete creep will double this. So, in ten years the deflection will be 4 × 3 × 2 = 24 mm. Is this too large? If the deflection is less than 1/250 of the span, most people in a house or office do not notice [8 p. 126]. 7 000 mm / 250 = 28 mm. So, it is okay.

Something else can go wrong: If the floor supports light interior walls, they may crack. Obviously, the walls are built after the floor. So, we need to look at the additional deflection after floor self-weight and concrete cracking. Self-weight is 6.7/9.8 × 100% = 69% of the load. 69% of 4 mm deflection × 3 = 8.3 mm. The additional deflection after construction is 24 – 8.3 = 15.7 mm. Experience shows that walls rarely crack if the extra deflection is less than 1/360 of the span. 7 000 mm / 360 = 19 mm. Okay.

Question 3: If the deflection were too large, what can we do to reduce it?

Question 4: Above we reduced the load to 69% and we did not redo the computation. Why not?

Question 5: A construction worker wants to build a light interior wall up to the concrete floor above. No dilatation filled with foam applied. What would happen after ten years? Do you allow building the wall like this?

8. Computing moments

For computing moments we use the moment molecules in figure 8 (derivation in Appendix 3).

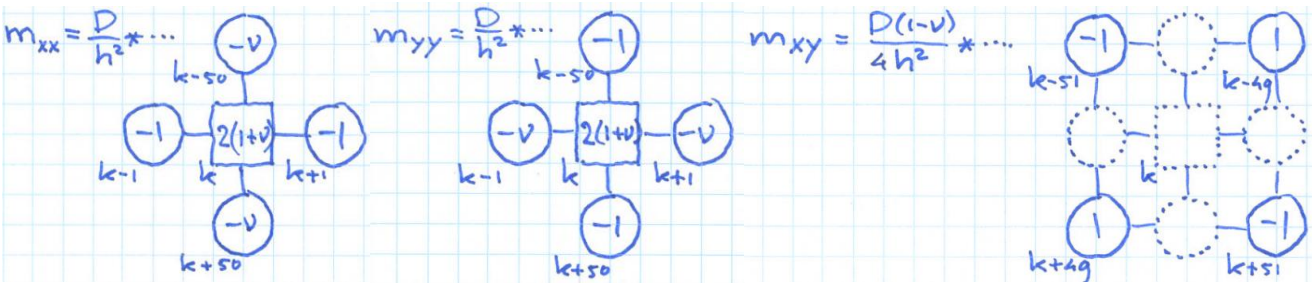

Figure 8. Moment molecules

```
mxx=np.array([0.0 for i in range(2500)])
myy=np.array([0.0 for i in range(2500)])
mxy=np.array([0.0 for i in range(2500)])
for j in range(1, 49):
   for i in range(1, 49):
      k=j*50+i
      mxx[k] = D/(h**2)*( -nu*w[k-50] -w[k-1] +2*(1+nu)*w[k] -w[k+1] -nu*w[k+50])
      myy[k] = D/(h**2)*( -w[k-50] -nu*w[k-1] +2*(1+nu)*w[k] -nu*w[k+1] -w[k+50])
      mxy[k] = D/(h**2)*(1-nu)/4*( -w[k-51] +w[k-49] +w[k+49] -w[k+51])
```

Let's plot the moments (fig. 9).

```
Z=[[mxx[50*j+i] for i in range(1,49)] for j in range(48,0,-1)]
plt.figure()
cp=plt.contourf(Z,10,cmap=plt.cm.jet)
plt.colorbar(cp)
plt.show()
```

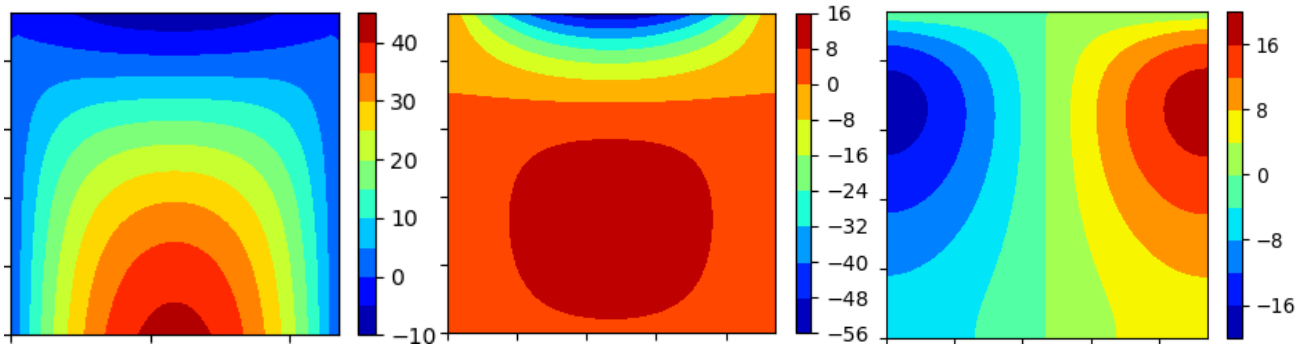

Figure 9. Contour plots of the moments m_{xx}, m_{yy} and m_{xy} [kNm/m]

First, we check ourselves again:
- On the hinge edges; is the bending moment m_{xx} zero? …… It is between 0 and 5. So, okay.
- On the hinge edges; is the bending moment m_{yy} also zero? … Okay.
- On free edge; is the bending moment m_{yy} zero? …… Okay.
- On the fixed edge; is the bending moment m_{yy} not zero? …… Okay.
- On the fixed edge; is the torsion moment m_{xy} zero? …… Okay.

We compare the moments with the finite element method (Appendix 2). … Okay.

The moments do not depend on plate stiffness *D*. We can see this in the equations: If *D* is doubled, then *w* is halved (see equation 1), then m_{xx} stays the same (see equation 3). The moments do depend on Poisson's ratio ν but not much.

Question 6: What happens to the moments, when the plate thickness is increased?

Question 7: If a support settlement occurs, do the moments still not depend on the plate stiffness?

Challenge 8: The free edge moment is an exception in that it does not depend on Poisson's ratio. How can this be explained?

Conundrum 9: In figure 9 we see torsion moment m_{xy} on the edges that are hinged. A hinge does not carry any torsion moment. Also, we did not apply any torsion moment load. It must be the Oompa-loompas again. They are applying torsion moments to the plate edges. Do you have a better explanation?

9. Checking reinforcement

Reinforced concrete plate moment capacity is calculated the same way as for beams:
Bar yield strength: 500 N/mm^2 / 1.15 = 435 N/mm^2 [8, p. 24]
Bar diameter: 12 mm
Bar spacing: 150 mm centre-to-centre
Bar layer: bottom, inner
Bar cover: 30 mm
Floor depth: t = 280 mm

Bar force $F = 1/4\ \pi\ 12^2 \times 435 \times 1000 / 150 = 328$ kN/m
Plate moment capacity $m_{bx} = F\ 0.9\ (t - 30 - 12 - 6) = 68$ kNm/m
This is the moment capacity due to the bottom bars in the *x* direction.
The other three bar layers are calculated in the same way.

Yielding of plate reinforcement is checked with the following equations. For a derivation see course [9].

$$
\begin{aligned}
&-m_{tx} < m_{xx} < m_{bx} \\
&-m_{ty} < m_{yy} < m_{by} \\
&m_{xy}^2 < (m_{bx} - m_{xx})(m_{by} - m_{yy}) \\
&m_{xy}^2 < (m_{tx} + m_{xx})(m_{ty} + m_{yy})
\end{aligned}
\tag{9}
$$

The plate moments must include load factors and material factors because now we check the ultimate limit state. So, we increase the load to 13.5 kN/m^2 and compute figures 10 and 11. We allow a bit of local yielding so that other bars pick up load (moment redistribution).

```
mtx = 68000                    # Nmm/mm
mty = 68000
mbx = 68000
mby = 68000
check=np.array([0.0 for i in range(2500)])
for j in range(1, 49):
   for i in range(1, 49):
      k=j*50+i
      if   mxx[k] > mbx: check[k] = 1
      elif mxx[k] <-mtx: check[k] = 1
      elif myy[k] > mby: check[k] = 1
      elif myy[k] <-mty: check[k] = 1
      elif (mbx - mxx[k])*(mby - myy[k]) < mxy[k]**2: check[k] = 1
      elif (mtx + mxx[k])*(mty + myy[k]) < mxy[k]**2: check[k] = 1
```

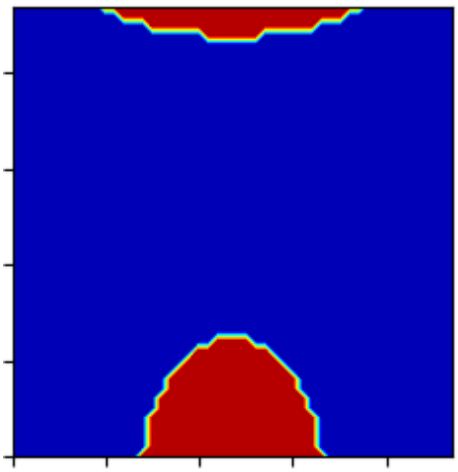

Figure 10. Reinforcement Ø12 - 200 … not okay

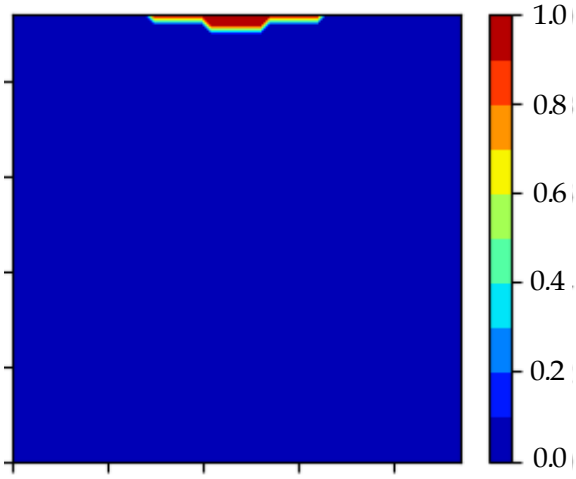

Figure 11. Reinforcement Ø12 - 150 … okay

We tell construction workers to put Ø12 – 200 in every layer. This is not enough at the fixed edge and at the free edge (fig. 10). There, we put extra bars Ø12. Every two bars, one extra bar will do: (200 + 100 + 100)/3 = 133 < 150. Obviously, at the fixed edge we put the extra bars perpendicular to the edge in the top layer. At the free edge we put the extra bars parallel to the edge in the bottom layer.

We will inspect the reinforcement just before casting concrete. If necessary, the construction workers just have to wait. We do not want the reinforcement to go wrong.

10. Checking crack width

Concrete should not crack too much. Corrosion of the rebars is not a problem indoors but we do not want people to get worried. This is a serviceability issue, so we go back to the load of 9.7 kN/m^2 and the moment of -56 kNm/m. The calculation is rather long and will be explained in course Concrete Structures [10]. The result is crack widths of 0.27 mm at the fixed support. This is acceptable. Anyway, the cracks will be hidden by floor covering.

11. Costs

For making design decisions, it is useful to know the costs.

Design	and inspection, 6 hours × 100 euro/hour = ...	600 euro
Formwork	(7 m × 7 m + 0.28 m × 28 m) × 100 euro/m^2 = ...	6 000 euro
Reinforcement	¼ π × (0.012 m)^2 × 7 m × (7 m)/(0.2 m) × 4 = 0.11 m^3	
	0.11 m^3 × 7000 kg/m^3 × 1.5 euro/kg = ...	1 000 euro
Concrete	0.28 m × 7 m × 7 m × 150 euro/m^3 = ...	2 000 euro
Labour	3 men × 7 days x 8 hours x 60 euro/hour = ...	10 000 euro
		---------- +
		19 600 euro

Question 10: What is better? A design with a few more rebars than required or a reinforcement mistake that needs to be fixed?

12. Conclusion

The model has been validated. It not probable that modelling errors or programming errors were made.

The floor deflection is acceptable (SLS).
The moments can be carried by the reinforcement (ULS).
The crack widths are acceptable (SLS).
This floor can be built.

13. Check the floor you are standing on.

- Is the floor, you are standing on, made of reinforced concrete?
- What is the span of this floor?
- How is it supported?
- How thick is it?
- What concrete strength and what rebars are often used?
- Is the deflection okay?
- Is the estimated reinforcement sufficient?
- What does an engineer do, if something is wrong?

Literature

1 A. Tsouvalas et al., CIEM5000, Base: Structural Engineering, Course Delft University of Technology, Faculty of Civil Engineering and Geosciences, 2024.
2 S. Timoshenko, History of strength of materials, McGraw-Hill, York, PA, USA, 1953
3 G. Kirchhoff, *Über das Gleichgewicht und die Bewegung einer elastischen Scheibe*, Journal für die reine und angewandte Mathematik (Crelles Journal) Volume 1850 Issue 40, 1850
4 A.E.H. Love, The small free vibrations and deformation of a thin elastic shell, *Philosophical transactions of the Royal Society of London A,* 1888, Volume 179, pp. 491–546.
5 P.C.J. Hoogenboom, *Notes on shell structures*, 2024, pp. 216, online: https://phoogenboom.nl/b17_book.pdf
6 Anonymous, Eurocode 0, Basis of structural design, European Union, 2002
7 Anonymous, Eurocode 1, Actions on structures, Part 1-1: General actions - Densities, self-weight, imposed loads for buildings, European Union, 2002
8 Anonymous, Eurocode 2 Design of concrete structures, Part 1-1: General rules and rules for buildings, European Union, 2004
9 P.C.J. Hoogenboom, CIEM5210-3 Advanced mechanics of structural elements, course Delft University of Technology, Faculty of Civil Engineering and Geosciences, 2024/2026, online: https://phoogenboom.nl/CIEM5210-3_schedule.html
10 Yuguang Yang et al., Course CIEM5230 Concrete Structures, Delft University of Technology, Faculty of Civil Engineering and Geosciences, 2024.

Appendix 1. Advice for teachers

Students can do this exercise alone or in a team. If a teacher is present, he or she may encourage students in the following ways.

- Print copies of this text to hand out.
- Bring Eurocodes 0 [6], 1 [7], 2 [8].
- Bring a long tape measure for measuring the floor span.
- Bring a mall tape measure for measuring floor thickness. Thickness can be measured in a stair case or by opening the "kruipluik".
- Perhaps you can bring the floor design drawings, with thickness and reinforcement.

Answers to the questions

1 The *z* axis is in the direction of gravity and the direction of the deflection. Consequently, the load and the deformation are positive. If we stand on the plate and point *x* to the right, then *y* points towards us. It follows from the Cartesian coordinate system.
2 Advantage: more accuracy. Disadvantage: larger molecules, more molecules and more programming mistakes.
3 Increase the thickness; reduce the span; reduce the load; add an edge beam; apply camber; apply pre-stress; more reinforcement but the effect of the latter is difficult to calculate.
4 Equation 1 is a linear differential equation: Twice the load gives twice the deflection and twice the moments.
5 The above concrete floor creeps and after a few year it will load the light wall. The wall is not made to carry the floor and will crack. The below concrete floor is not made to carry the floor above too and may get large cracks. However, there is no danger of collapse (ULS). It is a SLS issue. Most structural designers will not make a fuss about it; an architect may.
6 Nothing to the plate moments. The moment capacities become larger.
7 No.
8 The free edge is like a beam. There is free contraction in the lateral direction, so, no constraint that makes it stiffer.
9 Computed torsion moments on plate edges do not occur in reality. In reality, the shear stresses stay in the plate and go round like torsion stresses in a beam. We need to interpret a torsion moment on a plate edge as a shear force in the plate edge. Historically, the first to figure this out was Lord Kelvin in 1855 [2 p. 266].
10 A few more rebars cost less than halve a day delay.

Appendix 2. Finite element check

The plate has been entered in the finite element program SCIA Engineer. The element size is 200 mm. The Kirchhoff theory is used. The load is -9.7 kN/m^2. The finite difference results agree well with the finite element results.

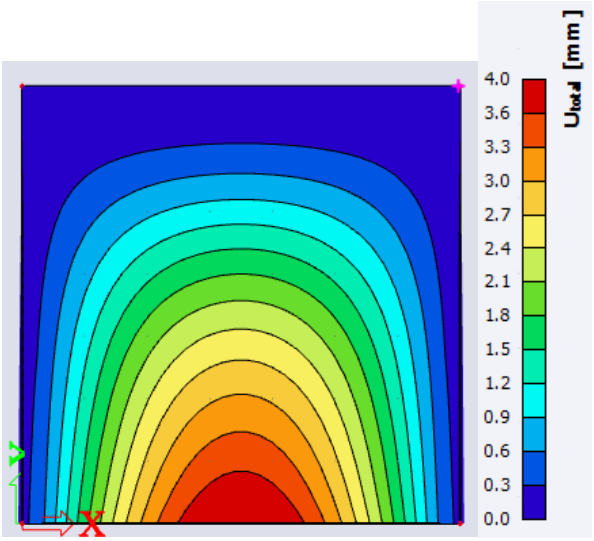

Figure 12. Displacement w

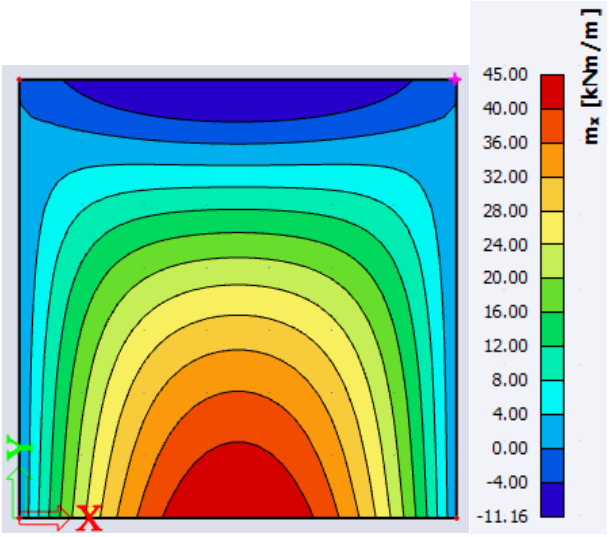

Figure 13. Bending moment m_{xx}

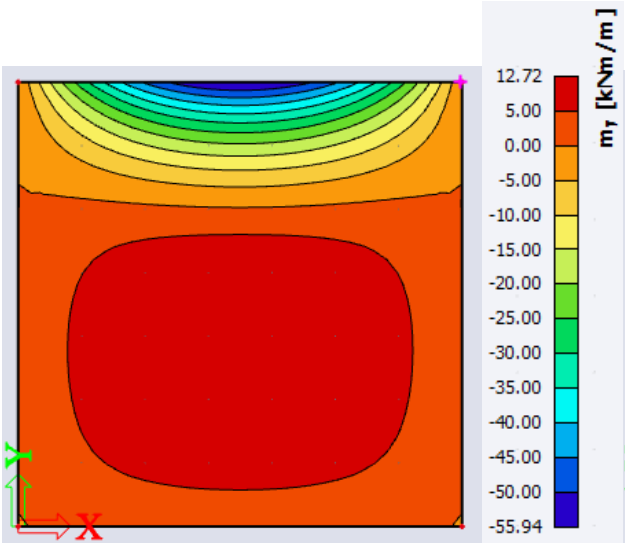

Figure 14. Bending moment m_{yy}

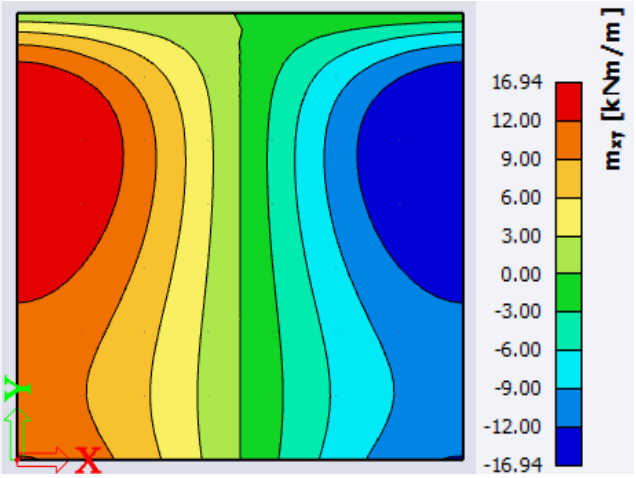

Figure 15. Torsion moment m_{xy}

Our torsion moment and SCIA's torsion moment are mirrored. This is caused by different positive directions. This exercise uses the sign convention that is used by most books, most software and all Delft mechanics courses (fig. 2).

Appendix 3. Derivation of the moment molecules

```
import sympy as sp
x,y               = sp.symbols('x,y')
D,nu,h            = sp.symbols('D,nu,h')
a0,a1,a2,a3,a4,a5 = sp.symbols('a0,a1,a2,a3,a4,a5')
w0,w1,w2,w3,w4,w5 = sp.symbols('w0,w1,w2,w3,w4,w5')

w = a0 + a1*x + a2*y + a3*x*x + a4*y*y
eq0 = sp.Eq( w0, w.subs(x, 0).subs(y,-h) )
eq1 = sp.Eq( w1, w.subs(x,-h).subs(y, 0) )
eq2 = sp.Eq( w2, w.subs(x, 0).subs(y, 0) )
eq3 = sp.Eq( w3, w.subs(x, h).subs(y, 0) )
eq4 = sp.Eq( w4, w.subs(x, 0).subs(y, h) )
a = sp.linsolve([eq0,eq1,eq2,eq3,eq4],a0,a1,a2,a3,a4)
print(a)

a0 = w2
a1 = (-w1 + w3)/(2*h)
a2 = (-w0 + w4)/(2*h)
a3 = (w1 - 2*w2 + w3)/(2*h**2)
a4 = (w0 - 2*w2 + w4)/(2*h**2)
w = a0 + a1*x + a2*y + a3*x*x + a4*y*y
mxx = -D*(sp.diff(w,x,x) + nu*sp.diff(w,y,y))
mxx0 = D/(h**2) * ( -nu*w0 - w1 + 2*(nu + 1)*w2 - w3 - nu*w4 )
print(sp.simplify(mxx-mxx0))
```

```
0
```

```
myy = -D*(sp.diff(w,y,y) + nu*sp.diff(w,x,x))
myy0 = D/(h**2) * ( -w0 - nu*w1 + 2*(nu + 1)*w2 - nu*w3 - w4 )
print(sp.simplify(myy-myy0))
```

```
0
```

```
a0,a1,a2,a3,a4,a5 = sp.symbols('a0,a1,a2,a3,a4,a5')
w = a0 + a1*x + a2*y + a3*x*y
eq0 = sp.Eq( w0, w.subs(x,-h).subs(y,-h) )
eq1 = sp.Eq( w1, w.subs(x, h).subs(y,-h) )
eq2 = sp.Eq( w2, w.subs(x,-h).subs(y, h) )
eq3 = sp.Eq( w3, w.subs(x, h).subs(y, h) )
a = sp.linsolve([eq0,eq1,eq2,eq3],a0,a1,a2,a3)
print(a)

a0 = ( w0 + w1 + w2 + w3)/4
a1 = (-w0 + w1 - w2 + w3)/(4*h)
a2 = (-w0 - w1 + w2 + w3)/(4*h)
a3 = ( w0 - w1 - w2 + w3)/(4*h**2)
w = a0 + a1*x + a2*y + a3*x*y
mxy = -D*(1 - nu)*sp.diff(w,x,y)
mxy0 = D*(1 - nu)/(4 * h**2) * ( -w0 + w1 + w2 - w3 )
print(sp.simplify(mxy-mxy0))
```

```
0
```