ARCHIVE OF MECHANICAL ENGINEERING

DOI: 10.24425/ame.2024.151334 2024, Vol. 71, No. 3, pp. 375–398

Roman KRÓL [1]

Hand drill shape optimization using Open CASCADE library and the steepest descent method

Received 21 June 2024, **Revised** 17 September 2024, **Accepted** 20 September 2024, **Published online** 25 September 2024

Keywords: hand drills, drill bits, milling, finite element analysis, structural optimization, Open CASCADE, steepest descent method

In this article, compliance optimization with the steepest descent method of the hand drill bits shapes for metal drilling is presented. The analysis of stress, displacements and compliance of the solid with random shape can be performed using the finite element method. In the case of a high number of optimization iterations, each analysis needs automatic modifications of geometry, mesh and boundary conditions. The Open CASCADE library can be used in the fast and automatic construction of modified models. It allows for fast reanalysis for each derivative of the objective function in the optimization process. The motivation of this research is to fill the gap in the literature on drilling technology. Most of the contemporary research is devoted to the oil and gas industry, while optimization of the hand drill bits used in metal drilling is rare. Compliance optimization allows us to find the shape which guarantees a greater stiffness for the specified loading conditions and a given amount of material. Although the obtained compliance was low, further experimental research would be needed to apply new solutions in drilling practice. This will involve the construction of the drill bit tips and the development of a heat treatment process.

1. Introduction

The large shear of modern drilling science concentrates on the analysis and design of the PDC drill bits used in the oil and gas industry [1–6]. The main concerns of the ground drilling companies are vibrations [2], [7–10] and buckling of the string [3], wear of the PDC drill [6], friction [11] and friction excited vibrations [2]. High interest is devoted to medical applications of bone drilling

✉ Roman KRÓL, e-mail: r.krol@urad.edu.pl

[1]Faculty of Mechanical Engineering,Casimir Pulaski Radom University, Radom, Poland

[12] and the main concern here is thermal effects. Less scientific interest is devoted to wood drilling [13].

In the drilling science branch, various methods are used to investigate studied phenomena. The finite element method is often used in the modeling of wear [14], or torsional vibrations of the drill strings [7]. In [15] finite element method has been used for the solution of the cross-sectional entities of the hand drill bit. Various theoretical models have been proposed in [10, 16, 17] and the drilling parameters measurements appear in article [18] devoted to drill bits engineering.

Compared to the number of researches related to the oil and gas industry, there are very few works nowadays concerning machining drill bits and the ones used in hand drills for metal drilling. An interesting experimental study has been presented in [19], where the infrared sensor has been used for the detection of the drill bit damage. This work is devoted to the hand drills and there is an interesting conclusion that the damage of the drill bit occurs right after the wear of the cutting edges. The finite element method in the study of hand drill bits vibrations was used in [15] however these studies were performed in the 1980s. In [17] (1980s) holographic measurements of the natural frequencies of the hand drill bits were performed. The author of [18] presented measurements of the torque during the concrete sample drilling with a hand drill. In [14], the authors studied concrete drilling with hand drills using the finite element method.

It is worth noting, that the optimization subject in the research devoted to the drill bits concerns parameters of the drilling process or management of the laboratory machining stages. Few works concern optimization of the drill bit point angle using design of experiments [20] or the optimization performed with the use of an experimental stand (Taguchi method with measurement of cutting speed, feed rates and temperature [21]).

The topic of drill bits shape optimization using the finite element method is rare. Topology optimization is most often used to reduce the material of the optimized model. The most common applications of this kind of optimization are bridges and supporting structures of the machines. The optimization of the drill bit helix grove would be challenging using this method. The shape optimization (using the Hadamard shape derivatives) would be more suitable for this task. Authors of [22] presented software for 2D optimization and stated that in shape optimization it is difficult to scale the numerical algorithm which optimizes specified structure. When the time step is too small, the optimization process is too slow to be completed. With the step too large, the optimization algorithm is unstable.

Although shape optimization could be very promising when applied to the drill bits, it is difficult to implement a 3D algorithm which will meet the numerical requirements. It could be challenging to define 3D constraints, which would prevent the emergence of designs with improper chip removal or conserve point angle and cutting edge shapes.

Drill bit shape can be optimized using basic gradient optimization algorithms like the steepest descent method. The weakest chain link of this optimization method is reanalysis with drill bit shape modification, which should be automated. The reanalysis methods are used in truss structures [23], while it is difficult to use them in the optimization of models with solid bodies. The Open CASCADE (Computer Aided Software for Computer Aided Design and Engineering) library allows for calling the Python programming language script, which constructs the STEP (Standard for the Exchange of Product Data) file with the geometry of the drill bit. The geometry can be generated based on parameters passed to the Python programming language script (radius of the drill bit shank, radius of the helix grove, overall twist angle, which can be defined in the form of the helix angle, number of drill bit sections, length of the shank, length of the drill bit etc.).

In this article, the Open CASCADE library has been used to construct drill bit geometry with the steepest descent method implemented in Matlab. Optimization has been performed for various initial shapes of the drill bit, used in hand drills. The optimized drill bit is designed for metal drilling, with the exception that the shape of the tip has been modified to allow the automatic application of the twisting, bending and compression von Neumann boundary conditions. The modification of the tip should not significantly influence the optimization process as far as the wear and chip removal phenomena are not considered.

The optimization process was performed for various load cases. The torsional with compression or the bending with compression loads were applied during optimization. As mentioned above, the application of the Dirichlet and Neumann boundary conditions in the finite element analysis is automatic. The buckling of the drill bit (shank and body) is not considered in the optimization process. It needs additional finite element method linear buckling analysis performed after each static-stress analysis, which could be applied in further development of the software presented in this article. The application of the additional buckling analysis needs improvement in the optimization algorithms, which involve the application of the penalty function method in the case when FoS (the factor of stability) is lower than the critical value. The Matlab PDE (Partial Differential Equations) module does not allow for the dedicated buckling analysis. Instead, normal modes analysis can be performed with the solution of the eigenvalues. This approach needs the construction of a new geometry with applied loads included in the additional masses or modification of the system mass matrix. The buckling analysis is important in the design of the drill bits, therefore verification analysis of the optimal shapes has been performed in the Autodesk Inventor Nastran software, which is presented in Section 9.

The compliance optimization is one of the most often applied structural optimizations in the engineering constructions. Compliance is expressed in the energy units [J=Nm] and the physical interpretation of this entity is the internal work done by the elastic construction under the applied forces. The shape with the lowest compliance is the stiffest one. The displacements of such a construction are the lowest for the analyzed load case. Compliance optimization is applied in the civil engineering structures or supporting structures of the machines when the maximum strength is needed with the lowest amount of material applied in construction. The search for the stiffest drill bit is beneficial because it reveals the most optimal shape for the given amount of material. The optimal drill bit will carry the greatest possible amount of load for the specified conditions.

2. Structure of the optimization framework

Two main modules [24] were used in the optimization of the drill bit. The first module is implemented in Python and uses the Open CASCADE library for the construction of the parametrized geometry. The second module is implemented in Matlab and uses the steepest descent method for optimization. The Python script is called from the Matlab algorithm to construct geometry for the specified dimensional parameters when it is needed.

The optimization algorithm is presented in Fig. 1. The optimization aim is to minimize compliance of the drill bit with box constraints of the argument vector and maximum drill bit volume constraint, which uses the penalty function method.

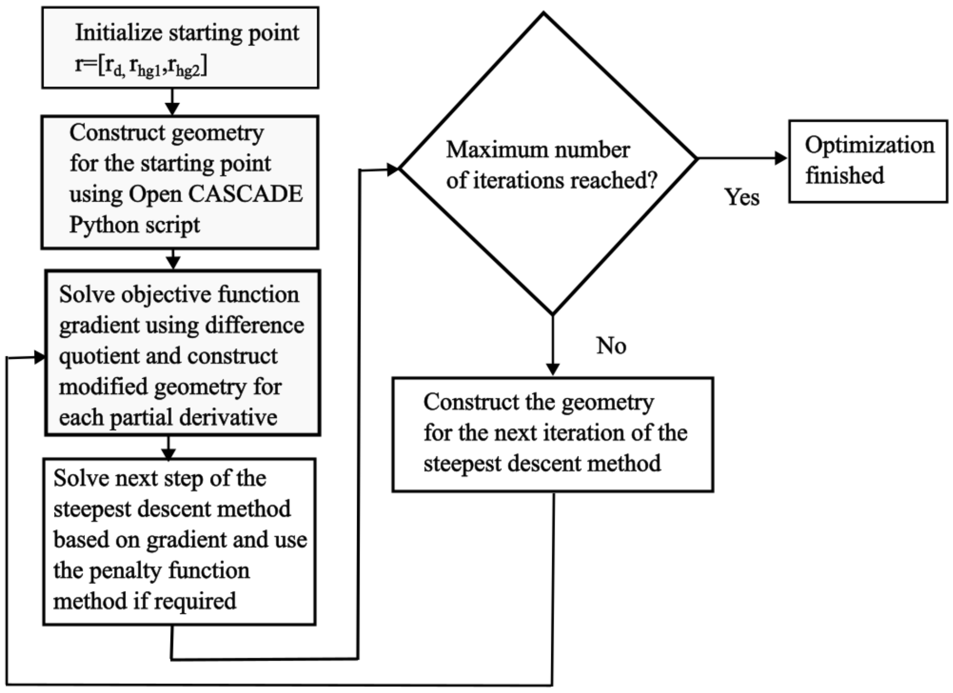
Initialize starting point
r=[r_d, r_hg1,r_hg2]
Construct geometry for the starting point using Open CASCADE Python script
Solve objective function gradient using difference quotient and construct modified geometry for each partial derivative
Solve next step of the steepest descent method based on gradient and use the penalty function method if required
Maximum number of iterations reached?
Yes
Optimization finished
No
Construct the geometry for the next iteration of the steepest descent method

Fig. 1. The drill bit optimization algorithm

At the beginning of the algorithm, the starting point is defined. The argument vector contains three variables: the radius of the shank (r_d), the radius of the first helix grove (r_{hg1}) and the second helix grove (r_{hg2}). After the initialization of the starting point, the geometry of the drill bit is constructed. The Python script which builds the geometry is called using the *system* command from the Matlab script. The construction of the geometry is described in Section 3. The optimized drill bit contains a shank, the drill bit body with two helix groves and a modified tip for easier application of the boundary conditions. When the initial geometry is constructed, Dirichlet and von Neumann boundary conditions are applied automatically, using the Matlab functions, which search for the geometry faces nearest to the specified point (Listing 4 and 5, Appendix B).

In the analyses performed in this article, two loading scenarios were considered. In the first scenario, the bending forces with compression forces are applied, and in the second scenario the torsional moment as the pair of forces and the compression forces are applied at the point of the drill bit.

After the initial analysis, the optimization algorithm computes the next iteration of the steepest descent method:

$$x_{i+1} = x_i - \alpha \cdot \nabla f(x_i), \tag{1}$$

where:x_{i+1} – argument vector at the next iteration, x_i – argument vector at the previous iteration, α – scaling coefficient, $\nabla f(x_i)$ – gradient of the objective function in the previous iteration.

To find the next value of the argument vector, the gradient of the objective function should be solved. For three arguments of the objective function, there are three partial derivatives, which are solved using the difference quotient (2), which is close to the partial derivative for small values of the Δx. For each direction, the $\Delta x = 10^{-6}$ m.

$$\frac{\partial f(x)}{\partial x} = \frac{f(x+\Delta x) - f(x)}{\Delta x}. \tag{2}$$

To solve eq. (2), the modified geometry for each difference quotient should be constructed. This issue influences significantly analysis time when other optimization algorithms using second derivatives are considered. However, sometimes it is possible to approximate the Hessian and shorten the analysis time.

The computation of the next step in the steepest descent method requires the penalty function method. If the box constraints for the objective function arguments are violated, or the maximum volume of the drill bit is exceeded, the value of the quadratic penalty function is added to the objective function. The value of the penalty function is adjusted by the scaling coefficient to fit the smoothing requirements for the objective function.

After each step of the steepest descent method, the optimization algorithm checks if the maximum number of iterations is reached. The first experiments with the optimization algorithm used only maximum iterations stopping criteria. This approach led to the oscillation of the objective function value. It was caused by an improperly scaled step coefficient α (see eq. (1)), which was kept constant in the steepest descent method. The optimization algorithm step was too long in the gradient direction and the value of the objective function could rise instead of the expected decrease in its value. The modification of the algorithm was introduced, which consisted in decreasing the step coefficient in the case of the objective function value rise. After the step decrease, the geometry construction and the static analysis in the current iteration are renewed until the drill bit compliance starts decreasing. If the value of the step is lower than $1 \cdot 10^{-13}$, then the optimization algorithm stops its execution and the assumption is made that the local minimum is reached.

3. Constructing geometry with Open CASCADE Python programming language script

The Open CASCADE [25] is the programming technology used for the design of CAD Systems [26–28]. One of the components of this library is a large set of functions used in the construction of the STEP files, which contain the geometry of the analyzed part. The STEP file format is widely used in the finite element analysis software and it can be applied in Matlab scripts using partial differential equations (PDE module). The PDE module solves physical problems in finite element meshes. The list of modules used in the Python script, which constructs drill bit geometry for the optimization algorithm is given in Appendix A. The *OCC.Core* is the core Open CASCADE library. The drill bit consists of three parts. The first is the drill bit shank, which is constructed in the form of a cylinder, the second is the drill bit body with two helical groves, which emerges after extrusion with the rotation of the drill bit cross-section, and the third is the tip of the drill bit. The shank cylinder is constructed using the function *BRepPrimAPI_MakeCylinder*, which receives the radius and the height of the cylinder as arguments.

Building the cross-section of the drill bit body requires the construction of three circles. Each circle requires the definition of the center point (*gp_Pnt* function), the direction of the axis (*gp_Dir* function) and the axis normal to the circle plane (*gp_Ax2* function). Each circle is constructed using *gp_Circ* function, which requires a circle radius as an argument. To design the cross-section of the drill bit body, the Boolean operation on the defined circles should be applied. The Boolean cut operation (*BRepAlgoAPI_Cut*) can be performed on the face structures. Therefore, the three circle geometries should be rebuilt in the form of edges, then based on the edges, the wires should be defined, and the last one can be used to build three circular faces. Edges, wires and faces are the names of the standard geometric structures used in the Open CASCADE library.

Construction of the shank and the circular faces is presented in the form of Python code examples in Listing 1 and 2 (Appendix B).

In Listing 1, the radius of the shank is read from the output file, created by the optimization algorithm. The file pointer is moved to the last line, which is being read using the function *re.findall* from the regular expressions module "*re*". The regular expression "[−+]?\d ∗ \.\d + |\d+" reads all floating point numbers from the last line of the file, which contains three radiuses (arguments of the objective function). The first argument (radius of the shank) and the height of the shank are substituted to the function *BRepPrimApi_MakeCylinder*, as it was mentioned before. The last line of the file always contains radiuses (the arguments of the objective function) from the last iteration of the steepest descent method.

In Listing 2, the component stages of the construction of the circular faces are presented. The basic functions for the definition of the center point, direction and axis are used first. Then the circular geometries are designed. In the next steps, these geometries are turned into faces using gradual transformation through the edge and wire forms. The last step is the Boolean operation, subtracting two eccentric circular faces from the center one. The eccentric circles, which subtract the material from the helix groves are localized at the distance 0.00625 m from the center of the shank circle. The exemplary resulting cross-section of the drill bit body is presented in Fig. 2. The geometry constructed by the Open CASCADE script is shown in Fig. 3. The cross-sectional geometry was simplified relative to the real metal drill bit cross-section, which contains additional details like margin. The simplification was applied to eliminate automatic mesh generation issues. The three radiuses presented in Fig. 2 are elements of the objective function argument vector $r = [r_d, r_{hg1}, r_{hg2}]$.

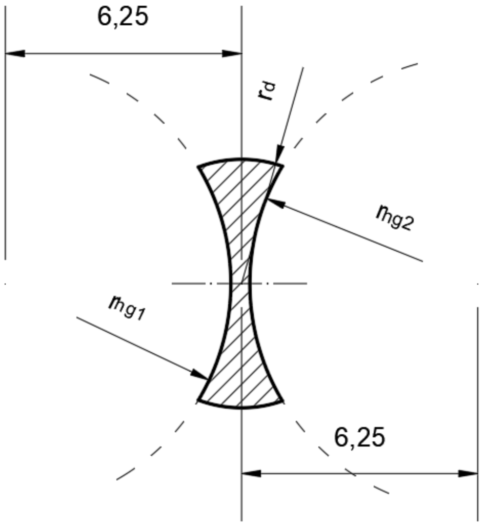

Fig. 2. Drill bit body cross-section. The radius of the drill bit shank is r_d and the radiuses of the helical groves are r_{hg1} and r_{hg2}

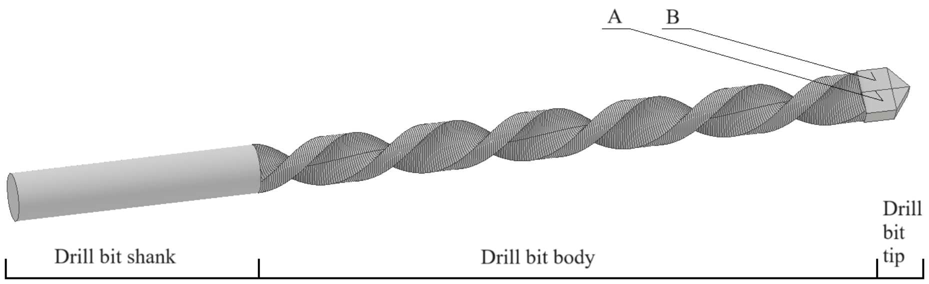

Fig. 3. Solid geometry of the drill bit used in the optimization process. The surfaces for the application of von Neumann boundary conditions are A and B. It can be normal to surface pair of bending forces or the pair of forces which generate a torsional moment, where one of the forces should be applied on the other side of the drill bit tip. The inclined surfaces at the tip are used for the application of the compression forces

4. Setting the environment for the Open CASCADE Python script

Automatic geometry construction using Open CASCADE needs the environment variable *PATH* set up before it is successfully applied in Matlab. The Open CASCADE library needs setting up the resources responsible for searching the Python libraries and modules. This is realized by setting up Anaconda, the Data Science platform, which contains the Python package manager *conda*. The script, which modifies the *PATH* system environment variable by adding specific paths to it is presented in Listing 3 (Appendix B) [29]. The *fileparts* function returns the base path of the *pyExec* variable. The function *getenv* returns the *PATH* system environment variable, *strsplit* splits this variable to the components separated by the semicolon, the *fullfile* function creates new paths and the function *unique* returns the elements which are not duplicated.

The script presented in Listing 3 should be called in Matlab before calling the Open CASCADE script using *system* command. In the other case, the Python interpreter can return errors due to the problems with finding the proper modules and libraries.

5. Application of the boundary conditions

The third part of the designed drill bit is the tip, which is simplified compared to the real drill bit for metal drilling. In such drill bits, the tip is conical and it contains cutting edges, which are raised above the conical surface. The drill bit tip was constructed from quads and triangles with extrusion operation and connected to the drill bit body through Boolean operations. The modified tip contains quad faces, which are easy for automatic search and application of the boundary conditions (Fig. 4).

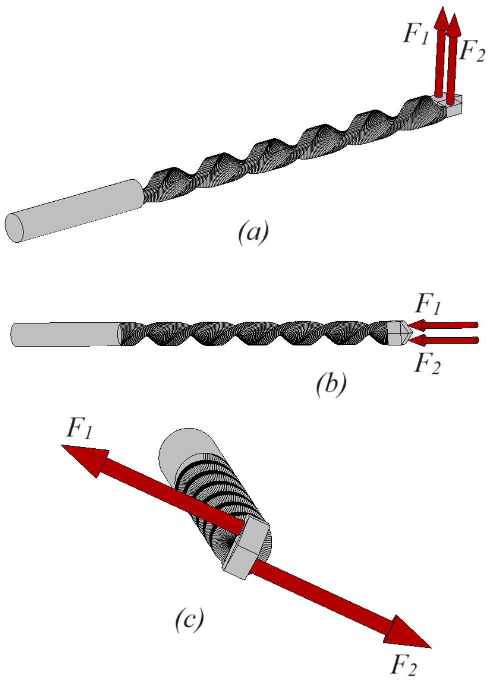

Fig. 4. The boundary conditions in the static analysis of the drill bit: bending (a), compression (b) and twisting (c)

As mentioned before, in this article, two variants of load are considered: bending with compression and twisting with compression (Fig. 4c). Let us assume that OZ axis is the longitudinal axis of the drill bit. The zero of the OZ axis is placed between the shank and the drill bit body. The shank is placed at the positive part of the OZ axis and the drill bit body is placed at the negative part of the OZ axis. The Matlab function *nearestFace* is used for the automatic search of the geometry face nearest to the specified point. The bottom of the shank is the face nearest to the (0,0,0.045) point and the Dirichlet fixed boundary conditions are applied at this place.

The compression load (von Neumann boundary condition) is placed on the face nearest to the $(0, r_d \cdot 10^{-3}/2, -0.108)$ point and on the face nearest to the $(0, -r_d \cdot 10^{-3}/2, -0.108)$ point, where r_d is the radius of the shank. The twisting torque is applied by placing two colinear forces at the opposite sides of the drill bit tip. The points $(-0.0025,\ r_d \cdot 10^{-3}/2,\ -0.103)$ and $(0.0025, -r_d \cdot 10^{-3}/2, -0.103)$ are nearest to the quad faces of the drill bit tip geometry, where the twisting pair of forces are applied (Fig. 4c). The collinear forces, which exert torque on the drill bit are applied in the OX axis direction in the form of surface traction in [N/m^2]. The application of the boundary conditions (the torque and the compression force) in the Matlab optimization script is presented in Listing 4 (Appendix B).

The boundary conditions of bending are applied to the similar quads of the twisting load. The difference is that both quads are localized at the same side of the drill bit tip and the bending forces have similar directions. The bending boundary conditions application is shown in Listing 5 (Appendix B).

It is worth noticing that the geometry from the Open CASCADE script is designed in [mm]. After reading the STEP file in Matlab, the geometry dimensions are expressed in [m] and the surface tractions in the *structuralBoundaryLoad* function should be expressed in [N/m^2]. The coordinates for the function *nearestFace* should also be given in [m].

After application of the Dirichlet and von Neumann conditions, the finite element mesh is generated and the structural static analysis is performed.

6. Estimation of the load torque

In one of the variants of the optimization process, the load torque is applied at the tip of the drill bit. The theoretical model is then constructed. Contrary to the optimization, it assumes that the tip of the drill bit has a conical shape due to chip formation, which is also a source of friction. The purpose of the model is the estimation of the torque arising due to friction in the metal drilling process. The torque component due to friction can be found from eq. (5) (see below). The shape of the point of the 3D drill bit geometry model is shown in Fig. 5. Fig. 6 shows the infinitesimal area used in the integration of the conical surface. In the drill bits used in the metal drilling, the conical surface is additionally inclined, which allows the cutting edges to come in contact with the machined material.

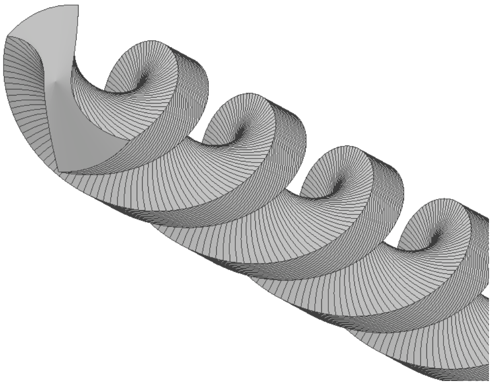

Fig. 5. The conical point of the drill bit. The point angle was assumed to be 120°. The shape is similar to the real drill bits used in hand drills for metal drilling

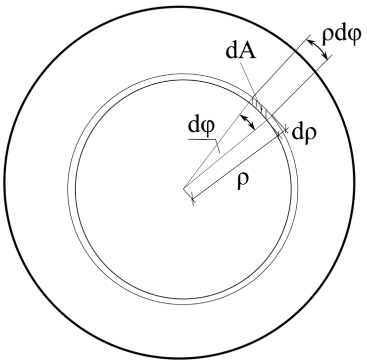

Fig. 6. Infinitesimal area dA of the drill bit contact surface used in the estimation of the load torque

The torque computations require the solution of a double integral. The infinitesimal area (Fig. 6) used in the double integral is given as:

$$\mathrm{d}A = \rho\,\mathrm{d}\phi\,\mathrm{d}\rho, \tag{3}$$

where: $\mathrm{d}\phi$ – infinitesimal angle, ρ – position of the $\mathrm{d}A$ infinitesimal area, $\mathrm{d}\rho$ – infinitesimal radial length of the infinitesimal area. The infinitesimal torque is:

$$\mathrm{d}M = \mathrm{d}T\rho = p2\pi\rho\,\mathrm{d}\rho\mu\rho, \tag{4}$$

where: $\mathrm{d}T$ – infinitesimal force acting on the infinitesimal $\mathrm{d}A$ area, $2\pi\rho$ – length of the ring, μ – friction coefficient. The torque value is solved using the double integral:

$$M = \int_0^a \int_0^{2\pi} \mu \frac{N\cos(30^\circ)}{0.5\pi a d}\rho^2\,\mathrm{d}\phi\,\mathrm{d}\rho, \tag{5}$$

where: M – load torque, a – slant height of the cone, d – diameter of the drill bit, N – compression force. The value of the torque M was solved for the data specified in Table 1.

Table 1. Data used in the solution of the load torque

Entity symbol	Description	Value
N	compression force	200 N
a	slant height of the cone	0.00461 m
d	diameter of the drill bit	0.008 m
μ	kinetic friction coefficient (dry, clean steel on steel)	0.42
M	Solved load torque	0.26 Nm

The obtained load torque $M \approx 0.26$ Nm. It is a small value compared to the maximum torque of the popular hand drilling tools, which is 19 Nm. The optimization was performed with a load torque of 20 Nm in the variant with a twisting and compression load of 200 N. In the optimization with bending, the bending force is 200 N and the compression force is also 200 N.

7. The finite element model and the solution of the compliance

The finite element PDE (Partial Differential Equations) model is created to solve the displacements of the drill bit. It is set up for the structural, static-solid analysis. The definition is presented in Listing 6 (Appendix B). The steel material properties (Young's modulus and Poisson's ratio) are assigned to the model.

The objective function return value is the compliance. It can be solved based on the displacements and the stiffness matrix:

$$C = u^T\mathbf{K}u, \tag{6}$$

where: C – compliance,u – displacements vector, $\mathbf{K}$ – stiffness matrix. The solution of compliance after the static analysis is finished is presented in Listing 7 (Appendix B).

The function *assembleFEMatrices*, presented in Listing 7, returns matrices used in the finite element analysis and stores them in the FEM structure. The **K** matrix of the structure FEM is the full stiffness matrix. The *result* variable contains the results of the static analysis.

The *result.Displacement.uN* contains displacement of the N-th direction, where N can be x, y or z (the directions of the Cartesian coordinate system in which static finite element analysis was performed). After the transposition of the uall displacements vector, the compliance is solved according to eq. (6). In the optimization process, the compliance (the internal energy of the drill bit loaded by the external forces) is minimized. The optimal drill bit has the lowest compliance. It should be also mentioned that the steepest descent method searches for the local minimum of the objective function, hence the optimization result may depend on the initial argument vector.

The penalty function adds the square of the constraint violation to the compliance calculated by the objective function. To verify the volume of the drill bit, this volume is determined based only on the radiuses of the shank and helix groves. The method of volume solution in the penalty function is shown in Listing 8 (Appendix B).

8. Results of the analysis

Application of the steepest descent method finds local minima of the objective function. Therefore it is important to perform optimization from the various starting points. The important issue in the application of the simple gradient optimization algorithms is the scaling of the step modification coefficients and the penalty function. In the steepest descent method, the step scaling coefficient can be too small (high analysis time is needed to find the minimum) or too high (the method will generate too large increments, which could lead the iteration process to the localization, where the value of the objective function increases).

The first attempts were made with stopping criteria set as the maximum function iterations. The stopping criterion, which stops the algorithm due to small changes in the argument vector, was excluded from the optimization algorithm. As a result, the compliance of the drill bit oscillated in the consecutive iterations.

The optimization algorithm of the steepest descent method was modified to accept an iteration only if the objective function value decreased as it was mentioned in Section 2.

The results of the optimization for the twisting and compression loads are presented in Figs. 7 and 8 and for the bending and compression in Figs. 9 and 10. The compliance is expressed in the energy units [J=Nm].

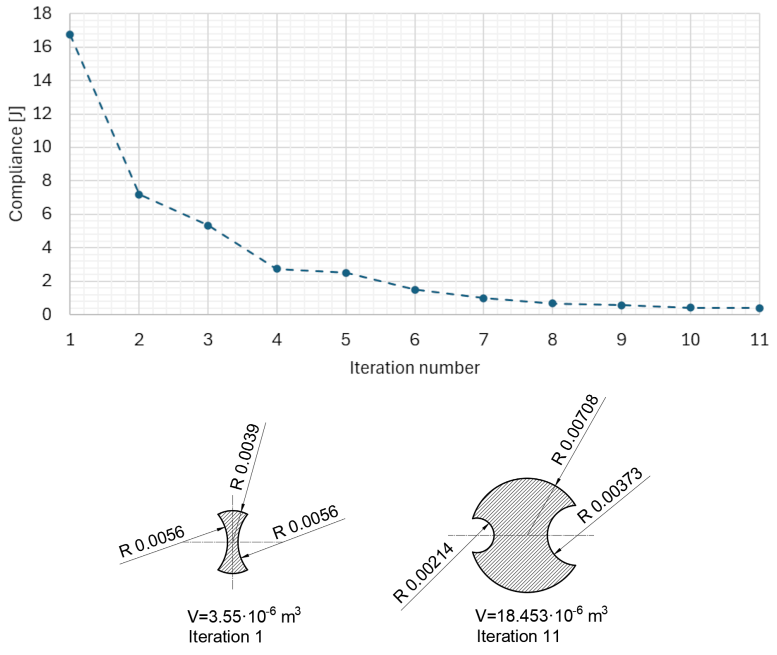

Fig. 7. Optimization for the twisting and compression loads for the starting point $r = [0.0039, 0.0056, 0.0056]$ [m]

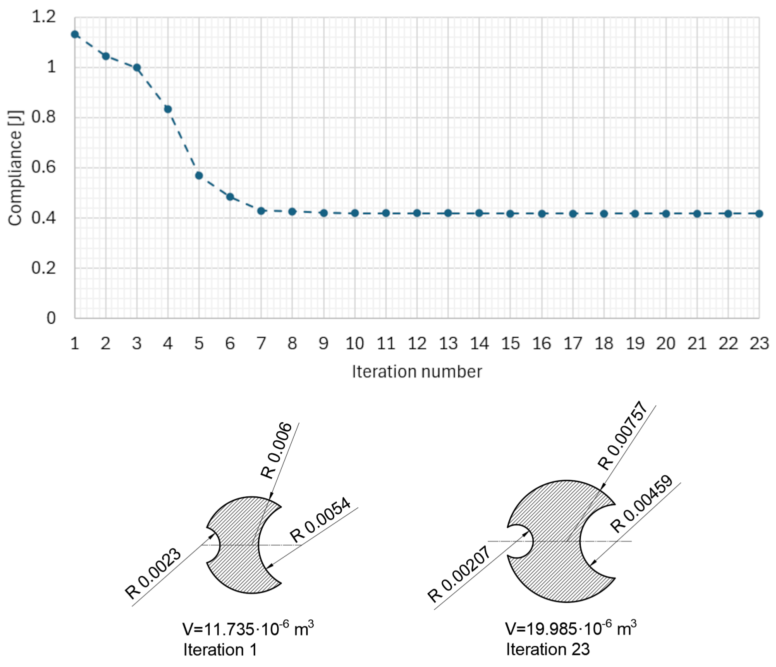

Fig. 8. Optimization for the twisting and compression loads for the starting point $r = [0.006, 0.0023, 0.0054]$ [m]

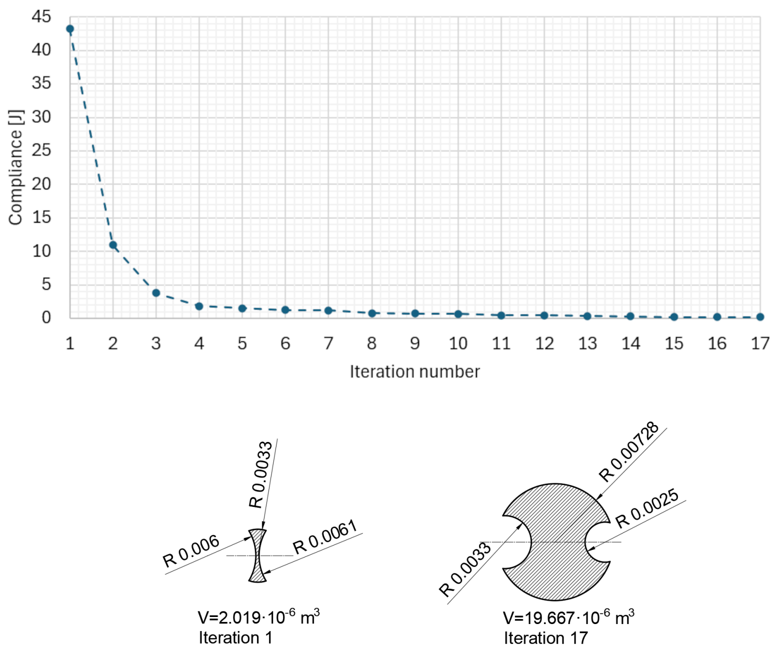

Fig. 9. Optimization for the bending with compression load for the starting point $r = [0.0033, 0.006, 0.0061]$ [m]

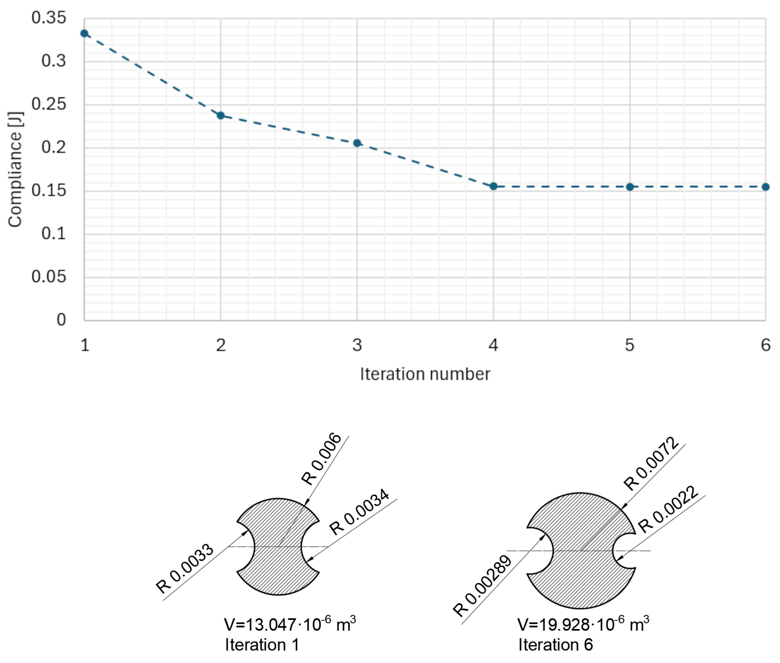

Fig. 10. Optimization for the bending with compression load for the starting point $r = [0.006, 0.0033, 0.0034]$ [m]

9. Verification of the results with the finite element linear buckling analysis

After the optimization, the finite element linear buckling analysis in the Autodesk Inventor Nastran has been performed. The results of the analysis are FoS and the buckling modes. In the optimal shapes, the diameter of the drill bits exceeds 0.014 m. The analysis showed that all shapes have high FoS values, which means that loss of stability will occur when the load case is 33.3-36.3 or 84.5–95.9 times higher relative to applied one in the load case with twisting and compression or with bending and compression, respectively. In the metal drilling practice, it is difficult to break the drill with a diameter of 0.008 m or higher, while breaking the drill bits with diameters lower than 0.004 m often happens. Optimization of compliance guarantees that the shape obtained with the specified amount of material will be the stiffest one. The displacements in the optimal solution will be the smallest and it will be more difficult to break the tool. Despite the minimum compliance, the optimal shape can lose stability, especially in the analyses where the material limit is low. Linear buckling analysis is important for verification of stability of the optimal shapes. Unfortunately, performing linear buckling analysis using theoretical methods is laborious because of the helical shape of the drill bit body. Finite element analysis is much faster to perform, while the Euler equation for the simplified case can be used to estimate the result. The buckling modes with the FoS coefficients for the optimal results are presented in Fig. 11.

10. Discussion of results

Finding a proper configuration of the optimization process needs many attempts. Analyses are time-consuming (single analysis duration lasts from one to several hours) and the process of software corrections is also long-lasting. First attempts bounded the optimization algorithm only with the number of iterations, which led to oscillating compliance in the consecutive steps. Significant influence on the oscillations of the results had the method of boundary conditions application. In one of the first optimization attempts, a twisting pair of forces was applied to the side surfaces of the groves. The faces of the groves were found as the faces nearest to the points lying on the axis crossing the center of the shank circle in the section lying in the drill bit tip. This approach caused changes in the twisting moment due to different force pair distances in consecutive iterations. The grove radiuses changed their values and the distance between the forces in the twisting pair oscillated. The application of the simplified drill bit tip solved this problem. The geometry of the tip has a variable width, dependent on the shank radius, therefore the force arm should be corrected in each iteration. It is more straightforward in a case of a simplified drill bit tip, because it is predictable, which faces will be

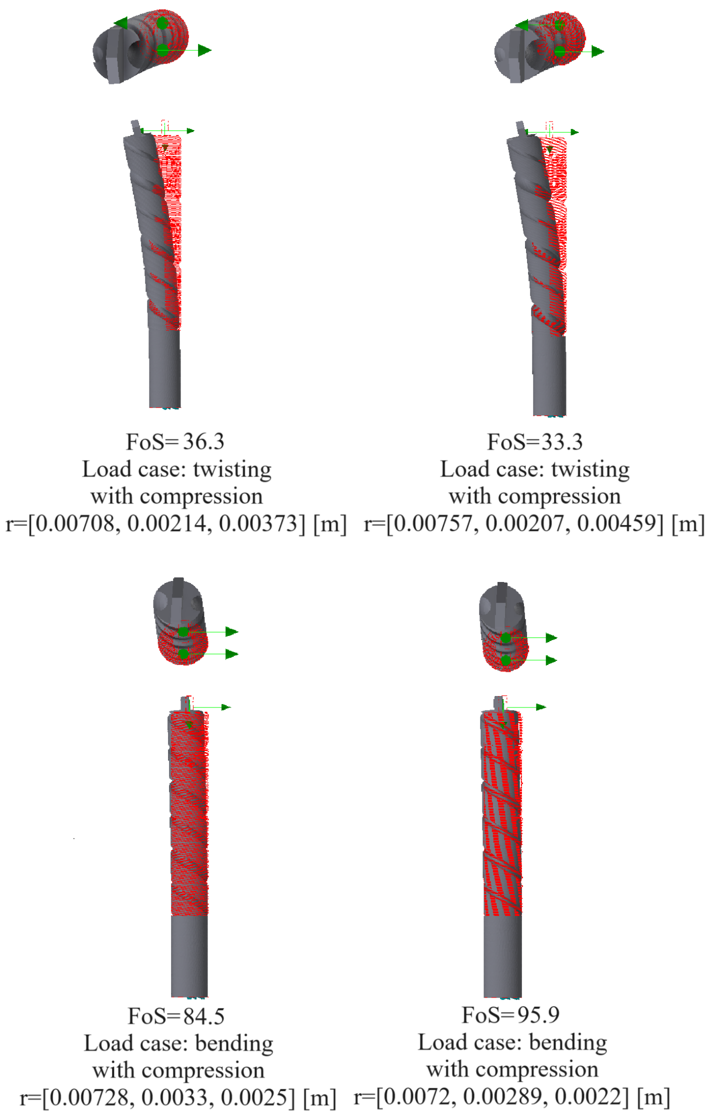

Fig. 11. The top view and the side view of buckling modes with FoS for the optimal results from Section 8. The final argument vector containing radiuses of the shank and helix groves is given in r. The red mesh presents undeformed shape, the green arrows show the force vectors

found in the search of the nearest face operation performed during the application of the boundary conditions.

In the four optimization processes presented in Figs. 7–10, all optimizations stopped near the maximum volume constraint, which was set to $20 \cdot 10^{-6}$ m^3. In general, when the load case with twisting and compression is considered, one of the helix groves tends to have a large radius and the other one is twice as small as the first one. The amount of analyses presented here is too small for general conclusions. However, a crescent shape with only one large helix grove would have the compliance smaller than traditional, symmetrical shapes of the hand drill bits for metal drilling. When both helix groves have large radiuses, there is a small amount of material between the arches of the groves, which decreases the reliability of the drill bit. The crescent shape would be stiffer and more reliable for the twisting load case.

For the bending with compression load case, the difference in the helix groves radiuses is smaller. The ratio of these radiuses for the twisting and compression load cases is 1.74 and 2.22, respectively, while for the bending with compression, it is 1.31 and 1.32.

Although it is obvious that when the bending loading is considered, the cross-sectional moment of inertia depends on the square distance from the center of mass, it is much more difficult to describe the emergence of the crescent shapes for optimization under the twisting load. When the drill bit undergoes bending, the helix groves distances from the center of mass tend to be equal. Under the twisting load, the moment of inertia for twisting is equal to the polar moment of inertia only for circular shapes. The solution for the twisting angle or the twisting stiffness becomes very complicated for the random cross-sectional shapes. In most cases, the cross-sectional deplanation should be taken into account.

The optimal shapes are the locally best solutions when compliance is considered. To develop new drill bit shapes used in hand drills, more physical phenomena should be considered. The eccentricity of the mass in the crescent shapes can lead to a rise in centrifugal force. The drill bit tip should be designed with proper cutting edges and experimental studies should be performed, which will show if the chip removal is correctly designed in new shapes.

The linear buckling analysis showed that loss of stability will occur when the loads are approximately 30 times or 80 times greater than those assumed in the optimization process. The FoS would be much smaller for the analyses with the lower volume constraint. Therefore, for the optimization of the small diameters of drill bits, it would be practical to include the penalty function for the FoS in the additional normal modes analysis with eigenvalues calculation.

11. Conclusions

Summing up the results, the application of the Open CASCADE library allows for gradient optimization of the milling tools. The optimization processes are very time-consuming. However in simple structures, like drill bits, a single personal computer can perform an optimization process in several hours. This article presents possibilities for compliance optimization. The final product of drill bits needs further laboratory testing, which includes measurements of the cutting resistance, chip removal and thermal treatment experiments to improve the endurance of the cutting edges. The topic of strength analysis with finite element method is rare in the literature on hand drill bits engineering. A high percentage of articles in drilling science are devoted to the experiments or studies of PDC drills in the oil and gas industry. The improvements of the hand drill bits for metal in the private company sector include modification of the tip, while there is a lack of works with studies of mechanical properties of the overall drill bit shape including helix groves. Application of the crescent cross-sections can be beneficial, especially in drill bits with small diameters. It would significantly increase strength and decrease compliance while improving the fatigue life of the drill bits.

The high value of FoS for the bending loads and more than twice smaller values for the twisting loads indicate that twisting is a much more dangerous loading case for drill bits stability. It is also confirmed by the static analysis. In civil engineering, constructions are designed to avoid twisting load cases because these are more dangerous than bending and compression. When the drill bit locks in the opening, the twisting action of the hand drill motor leads to the breaking of the tool. To improve the strength of the small diameter drill bits it is beneficial to increase the effective cross-section by applying the crescent shapes.

It is beneficial to perform optimization for multiple load cases. In this kind of analysis, the values of the objective functions for various loads (bending, compression, twisting) can be added to each other, possibly considering weighting factors. In this article, the optimization was performed for the separate load cases to compare the results for various loads. The results show that it is sufficient to consider only twisting load, because it is most important in the strength analysis of the drill bits. The results for the bending load are almost symmetric. In the optimal shapes presented in Figs. 7 and 8, the compliance for the twisting load case is ≈ 0.41 J and ≈ 0.42 J, respectively, while for Figs. 9 and 10, where bending optimization is presented, it is ≈ 0.17 J and ≈ 0.16 J. To apply multiple load case optimization in further research, the objective functions which calculate compliance for each load case should be prepared and added. The gradient of the objective function should also include these modifications. However, the resulting optimal shapes probably would be significantly influenced by the twisting component, while bending would have a smaller impact.

Appendix A
Python libraries used in drill bit geometry construction

Python module name	Python function	Description
math		contains mathematical constants (π)
numpy		contains the function linspace()
re		regular expressions used in reading text file data from the optimization algorithm
OCC.Core.BRepPrimAPI	BRepPrimAPI_MakeCylinder	draw the cylinder (used in the construction of the drill bit shank geometry)
OCC.Core.gp	gp_Pnt	point definition
	gp_Dir	direction definition (used in extrusion)
	gp_Circ	Circle definition (used in drill bit cross-section construction)
	gp_Ax3, gp_Ax2, gp_Ax1	axis definition (used in extrusion)
	gp_Trsf	Transformation definition (used in extrusion and needs a set of gp_Pnt, gp_Dir, gp_Ax1, gp_Ax2 and gp_Ax3 definitions)
OCC.Core.BRepBuilderAPI	BRepBuilderAPI_MakeFace	builds face from edges
	BRepBuilderAPI_MakeEdge	builds edge from wires
	BRepBuilderAPI_MakeWire	builds wire
	BRepBuilderAPI_Transform	executes transformation of the cross-section built as face
OCC.Core.BRepAlgoAPI	BRepAlgoApi_Fuse BRepAlgoApi_Cut	Boolean operations on cross-sectional circles or the shank and drill bit solids (fuse and cut)
OCC.Core.BRepOffsetAPI	BRepOffsetAPI_ThruSections	used in the extrusion of the drill bit through the transformed and rotated cross-sections to create solid geometry
OCC.Core.TopoDS	topods	needed in topology explorer, to construct cross-section from edges
OCC.Core.TopExp	TopExp_Explorer	topology explorer used in construction of the wires
OCC.Core.TopAbs	TopAbs_EDGE	needed in topology explorer in definition of the edge
OCC.Core.STEPControl	STEPControl_Writer	used to create the writer, which saves the solid geometry to the STEP file
	STEPControl_AsIs	used in the STEP writer

Appendix B

Listing 1. Construction of the drill bit shank geometry (programming language: Python)

```python
with open(radius_path) as myfile_radius:
    for line_radius in myfile_radius:
        pass
    lastline_radius=line_radius

radiuses_floatsstr=re.findall(r"[-+]?\d*\.\d+|\d+",lastline_radius)

myradiuses=[]
for radfs in radiuses_floatsstr:
    myradiuses.append(float(radfs))

center_radius=myradiuses[0]
height=40.0

cylinder = BRepPrimAPI_MakeCylinder(center_radius, height).Shape()
```

Listing 2. Construction of the drill bit body cross-section geometry (programming language: Python)

```python
p1=gp_Pnt(0.0, 0.0, 0.0)
p2=gp_Pnt(6.25, 0.0, 0.0)
p3=gp_Pnt(-6.25, 0.0, 0.0)
dir1=gp_Dir(0,0,1)
dir2=gp_Dir(0,0,1)
dir3=gp_Dir(0,0,1)
ax1=gp_Ax2(p1,dir1)
ax2=gp_Ax2(p2,dir2)
ax3=gp_Ax2(p3,dir3)
myCirc=gp_Circ(ax1, myradiuses[0])
myCirc1=gp_Circ(ax2, myradiuses[1])
myCirc2=gp_Circ(ax3, myradiuses[2])

edge=BRepBuilderAPI_MakeEdge(myCirc).Edge()
edge1=BRepBuilderAPI_MakeEdge(myCirc1).Edge()
edge2=BRepBuilderAPI_MakeEdge(myCirc2).Edge()

wire=BRepBuilderAPI_MakeWire()
wire1=BRepBuilderAPI_MakeWire()
wire2=BRepBuilderAPI_MakeWire()

wire.Add(edge)
wire1.Add(edge1)
wire2.Add(edge2)

body_base_face=BRepBuilderAPI_MakeFace(wire.Wire()).Face()
body_base1_face=BRepBuilderAPI_MakeFace(wire1.Wire()).Face()
body_base2_face=BRepBuilderAPI_MakeFace(wire2.Wire()).Face()
Cut1 = BRepAlgoAPI_Cut(body_base_face, body_base1_face).Shape()
bodydrill = BRepAlgoAPI_Cut(Cut1, body_base2_face).Shape()
```

Listing 3. Setting the Anaconda environment [29] in Matlab (programming language: Matlab)

```
pyExec = 'C:\Users\user_name\.conda\envs\pyoccenv\python.exe';
pyRoot = fileparts(pyExec);
p = getenv('PATH');
p = strsplit(p, ';');
addToPath = {
    pyRoot
    fullfile(pyRoot, 'Library', 'mingw-w64', 'bin')
    fullfile(pyRoot, 'Library', 'usr', 'bin')
    fullfile(pyRoot, 'Library', 'bin')
    fullfile(pyRoot, 'Scripts')
    fullfile(pyRoot, 'bin')
};
p = [addToPath(:); p(:)];
p = unique(p, 'stable');
p = strjoin(p, ';');
setenv('PATH', p);
```

Listing. 4. Application of the boundary conditions of twisting and compression. The surface traction is given in [N/m^2] (programming language: Matlab)

```
faceDC = nearestFace(model.Geometry, [0 0 45e-3]);
faceNC_n1 = nearestFace(model.Geometry, [0 x(1)/2.0 -108e-3]);
faceNC_n2 = nearestFace(model.Geometry, [0 -x(1)/2.0 -108e-3]);
faceNC_t1= nearestFace(model.Geometry,[-2.5e-3,x(1)/2.0,-103e-3]);
faceNC_t2= nearestFace(model.Geometry,[2.5e-3,-x(1)/2.0,-103e-3]);
structuralBC(model,"Face",faceDC,"Constraint","fixed");
structuralBoundaryLoad(model, ...
                     "Face",faceNC_n1, ...
                     "SurfaceTraction",[0;0;100/(sqrt(0.003^2+x(1)^2)*0.003)]);
structuralBoundaryLoad(model, ...
                     "Face",faceNC_n2, ...
                     "SurfaceTraction",[0;0;100/(sqrt(0.003^2+x(1)^2)*0.003)]);
structuralBoundaryLoad(model, ...
                     "Face",faceNC_t1, ...
                     "SurfaceTraction",[(20/x(1))/(x(1)*0.005);0;0]);
structuralBoundaryLoad(model, ...
                     "Face",faceNC_t2, ...
                     "SurfaceTraction",[(-20/x(1))/(x(1)*0.005);0;0]);
```

Listing 5. Application of the boundary conditions of bending and compression. The surface traction is given in [N/m^2] (programming language: Matlab)

```
faceDC = nearestFace(model.Geometry, [0 0 45e-3]);
faceNC_b1= nearestFace(model.Geometry,[2.5e-3,x(1)/2.0,-103e-3]);
faceNC_b2= nearestFace(model.Geometry,[2.5e-3,-x(1)/2.0,-103e-3]);
faceNC_n1 = nearestFace(model.Geometry, [0 x(1)/2.0 -108e-3]);
faceNC_n2 = nearestFace(model.Geometry, [0 -x(1)/2.0 -108e-3]);
structuralBC(model,"Face",faceDC,"Constraint","fixed");
structuralBoundaryLoad(model, ...
                     "Face",faceNC_n1, ...
                     "SurfaceTraction",[0;0;100/(sqrt(0.003^2+x(1)^2)*0.003)]);
```

```
structuralBoundaryLoad(model, ...
                    "Face",faceNC_n2, ...
                    "SurfaceTraction",[0;0;100/(sqrt(0.003^2+x(1)^2)*0.003)]);
structuralBoundaryLoad(model, ...
                    "Face",faceNC_b1, ...
                    "SurfaceTraction",[-100/(x(1)*0.005);0;0]);
structuralBoundaryLoad(model, ...
                    "Face",faceNC_b2, ...
                    "SurfaceTraction",[-100/(x(1)*0.005);0;0]);
```

Listing 6. Definition of the PDE finite element model with the material properties for steel (programming language: Matlab)

```
    model=createpde("structural","static-solid");
    model.Geometry=importGeometry(model, geompathstr);

    structuralProperties(model,"YoungsModulus",200E9, ...
                     "PoissonsRatio",0.3);
```

Listing. 7. Solution of the compliance of the drill bit (programming language: Matlab)

```
generateMesh(model, Hmax=5e-3);
result=solve(model);
FEM = assembleFEMatrices(model);
Kmtx=FEM.K;
sizeu=length(result.Displacement.ux);
uall=zeros(sizeu*3,1);
uall(1:sizeu)=result.Displacement.ux;
uall(sizeu+1:2*sizeu)=result.Displacement.uy;
uall(2*sizeu+1:3*sizeu)=result.Displacement.uz;
compliance=uall'*Kmtx*uall;
```

Listing 8. The penalty function, which solves volume of the drill bit geometry based on radiuses (programming language: Matlab)

```
function y=drill_penalty(x)
    VolumeMax=20000;
    r1=x(1);
    r2=x(2);
    d=6.25;
    A=pi*x(1)*x(1);
    A1=r1^2*acos((d^2+r1^2-r2^2)/(2*d*r1))+...
        r2^2*acos((d^2+r2^2-r1^2)/(2*d*r2))-...
        0.5*sqrt((-d+r1+r2)*(d+r1-r2)*(d-r1+r2)*(d+r1+r2));
    r2=x(3);
    A2=r1^2*acos((d^2+r1^2-r2^2)/(2*d*r1))+...
        r2^2*acos((d^2+r2^2-r1^2)/(2*d*r2))-...
        0.5*sqrt((-d+r1+r2)*(d+r1-r2)*(d-r1+r2)*(d+r1+r2));
    Area=A-A1-A2;
    Volume=A*40+Area*100;
    y=0;
    if (Volume>VolumeMax)
        y=y+((Volume-VolumeMax)*1e-2)^2;
    end
    if (x(1)+x(2)<6.4)
        y=y+(6.4-(x(1)+x(2)))^2;
    end
    if (x(1)+x(3)<6.4)
        y=y+(6.4-(x(1)+x(3)))^2;
    end
    if (x(2)>6.1)
        y=y+(x(2)-6.1)^2;
    end
    if (x(3)>6.1)
        y=y+(x(3)-6.1)^2;
    end
end
```

References

[1] L.Y. Ropyak, T.O. Pryhorovska, and K.H. Levchuk. Analysis of materials and modern technologies for PDC drill bit manufacturing. *Progress in Physics of Metals*, 21(2):274–301, 2020, doi: 10.15407/ufm.21.02.274.

[2] I. Kessai, S. Benammar, M.Z. Doghmane, and K.F. Tee. Drill bit deformations in rotary drilling systems under large-amplitude stick-slip vibrations.*Applied Sciences*, 10(18):6523, 2020, doi: 10.3390/APP10186523.

[3] Z. Wu, W. Zhang, R. Yuan, and J. Liu. Buckling and dynamic analysis of drill string system in horizontal wells. *Nonlinear Dynamics*, 112:4147–4168, Mar. 2024, doi: 10.1007/s11071-023-09100-7.

[4] L. Chen, Q. cao, Q. Qi, S. Niu, Y. Yang, X. Chen, Z. Zhao, and B. Wu. Development and application of hobbing-cone hybrid PDC bit. *Geomechanics and Geophysics for Geo-Energy and Geo-Resources*, 9:139, 2023, doi: 10.1007/s40948-023-00679-0.

[5] A. Nautiyal and A. K. Mishra. Drill bit selection and drilling parameter optimization using machine learning. In *IOP Conference Series: Earth and Environmental Science*, 1261:012027, 2023. doi: 10.1088/1755-1315/1261/1/012027.

[6] L.Q. Wang, M.J. Shao, W. Zhang, Z.P. Xiao, S. Yang, and M.H. Yang. Prediction and analysis of PDC bit wear in conglomerate layer with machine learning and finite-element method. *Geofluids*, 2022:4324202, 2022. doi: 10.1155/2022/4324202.

[7] M. Kapitaniak, V.V. Hamaneh, and M. Wiercigroch. Torsional vibrations of helically buckled drill-strings: Experiments and FE modelling. *Journal of Physics: Conference Series*, 721:012012, 2016. doi: 10.1088/1742-6596/721/1/012012.

[8] O.V. Vashchilina, I.V. Lebedyeva, and O.I. Bilobrytska. Modeling and numerical research of the self-excitation phenomenon of the drill bit whirlings vibrations. Bulletin of the Taras Shevchenko National University of Kyiv. Physics and Mathematics, 2019(1):28–33, 2019, doi: 10.17721/1812-5409.2019/1.5. (in Ukrainian).

[9] M. Kanzari, I.M. Shahin, M.Y. Alqaradawi, and B. Balachandran. Drill string nonlinear vibrations: Experimental studies and finite-element analysis. *Journal of Physics: Conference Series*, 1075:012010, 2018. doi: 10.1088/1742-6596/1075/1/012010.

[10] V. Svitlytskyi, S. Iagodovskyi, and N. Bilenko. Effect of vibration dampers on the dynamic state of a drill string. *Technology Audit and Production Reserves*, 5(1(73)):32–36, 2023, doi: 10.15587/2706-5448.2023.290145.

[11] M. Bembenek, Y. Grydzhuk, B. Gajdzik, L. Ropyak, M. Pashechko, O. Slabyi, A. Al-Tanakchi, and T. Pryhorovska. An analytical–numerical model for determining “drill string–wellbore” frictional interaction forces. *Energies*, 17(2):301, 2024, doi: 10.3390/en17020301.

[12] G. Alajmo, U. Schlegel, B. Gueorguiev, R. Matthys, and E. Gautier. Plunging when drilling: effect of using blunt drill bits. *Journal of Orthopaedic Trauma*, 26(8):482–487, 2012, doi: 10.1097/BOT.0b013e3182336ec3.

[13] E. Sharapov, X. Wang, E. Smirnova, and J.P. Wacker. Wear behavior of drill bits in wood drilling resistance measurements. *Wood and Fiber Science*, 50(2):154–166, 2018, doi: 10.22382/wfs-2018-017.

[14] S.G. Moseley. Reinforced concrete drilling with cemented tungsten carbide drill bits: wear and fracture mechanisms and predictive failure analysis based on finite element modelling and weibull. [Online]. Available: https://www.researchgate.net/publication/353038357

[15] O. Tekinalp and A.G. Ulsoy. Modeling and finite element analysis of drill bit vibrations. *Journal of Vibration and Acoustics*, 111(2):148–155, 1989. doi: 10.1115/1.3269835.

[16] A.G. Ulsoy, O. Tekinalp, and E. Lenz. Dynamic modeling of transverse drill bit vibrations. *CIRP Annals*, 33(1):253–258, 1984. doi: 10.1016/S0007-8506(07)61420-6.

[17] O. Tekinalp and A.G. Ulsoy. Modeling of drill bit transverse vibrations. In *Proceedings SPIE 0955, Industrial Laser Interferometry II*, 1988. doi: 10.1117/12.947677.

[18] Ł. Bołoz. Influence of the drill bit tip geometry on the rotary drilling process performed with a hand-held hydraulic drill. *Production Engineering Archives*, 29(3):304–310, 2023, doi: 10.30657/pea.2023.29.35.

[19] M.J. Jeong, S.W. Lee, W.K. Jang, H.J. Kim, Y.H. Seo, and B.H. Kim. Prediction of drill bit breakage using an infrared sensor. *Sensors*, 21(8):2808, 2021, doi: 10.3390/s21082808.

[20] R. Zulrian Aldio and Z. Mustafa. Drill bit selection using Design of Experiments (DoE) method. *Journal of Renewable Energy and Mechanics*, 03(01):39–44, 2020. doi: 10.25299/rem.2020.vol3(01).4597.

[21] R. Çakiroğlu and A. Acir. Optimization of cutting parameters on drill bit temperature in drilling by Taguchi method. *Measurement*, 46(9):3525–3531, 2013. doi: 10.1016/j.measurement.2013.06.046.

[22] G. Allaire and O. Pantz, Structural Optimization with FreeFem++, Nov. 2005.

[23] P. Kołakowski, M. Wikło, and J. Holnicki-Szulc. The virtual distortion method—a versatile reanalysis tool for structures and systems. *Structural and Multidisciplinary Optimization*, 36(3): 217–234, 2008. doi: 10.1007/s00158-007-0158-7.

[24] R. Król. The software for hand drill bit gradient shape optimization (Matlab + Open CASCADE Python script). Zenodo, Aug. 2024. doi: 10.5281/zenodo.13365716.

[25] T. Paviot et al. Open Cascade Software Library. https://dev.opencascade.org/project/pythonocc, 2024.

[26] EDF-CEA, Salome Meca. www.salome-platform.org, 2021.

[27] FreeCAD Team, FreeCAD. www.freecad.org, 2001.

[28] C. Geuzaine, J.-F. Remacle, GMSH. gmsh.info, 1997.

[29] J. Hapke. MATLAB Crashes when using conda environment other than base. In Matlab Software Discussion Forum, MathWorks, 2019.