УНИВЕРСИТЕТ ИТМО
НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ
сентябрь–октябрь 2019 Том 19 № 5 ISSN 2226-1494 http://ntv.itmo.ru/
SCIENTIFIC AND TECHNICAL JOURNAL OF INFORMATION TECHNOLOGIES, MECHANICS AND OPTCS
September–October 2019 Vol. 19 No 5 ISSN 2226-1494 http://ntv.itmo.ru/en/
НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК УНИВЕРСИТЕТ ИТМО ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

УДК 004.415.2 doi: 10.17586/2226-1494-2019-19-5-939-946

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ДЛЯ РЕШЕНИЯ ЗАДАЧ МЕХАНИКИ ДЕФОРМИРУЕМОГО ТВЕРДОГО ТЕЛА

А.А. Попов

Томский государственный университет, Томск, 634050, Российская Федерация
Адрес для переписки: popov.alexey1997@gmail.com

Информация о статье
Поступила в редакцию 04.06.19, принята к печати 23.07.19
Язык статьи — русский

Ссылка для цитирования: Попов А.А. Программное обеспечение для решения задач механики деформируемого твердого тела // Научно-технический вестник информационных технологий, механики и оптики. 2019. Т. 19. № 5. С. 939–946. doi: 10.17586/2226-1494-2019-19-5-939-946

Аннотация

Предмет исследования. Представлен способ создания программного обеспечения для решения задач механики деформируемого твердого тела. Программное обеспечение должно гарантировать высокую точность и скорость вычислений, а также простую подготовку начальных и обработку полученных данных даже для неопытного пользователя. При разработке программного обеспечения использовались интерфейс прикладного программирования (API) сеточного генератора с открытым исходным кодом GMSH и математическая библиотека Eigen. **Метод.** Разрабатываемое программное обеспечение состоит из трех модулей: GMSH_API, InputFile, FEMSolver и базы данных. Модуль GMSH_API, подготавливающий конечно-элементную модель исследуемого тела, написан с использованием API сеточного генератора GMSH. В модуле InputFile описаны методы взаимодействия с предварительно созданной базой данных, использование которой позволяет быстро и просто подготовить входной файл, необходимый для запуска расчета. Численный расчет методом конечных элементов реализован в модуле FEMSolver. При его реализации активно использовалась математическая библиотека Eigen, позволяющая строить разреженные матрицы, не хранящие в памяти нулевые элементы. Такая возможность избавляет от дополнительных преобразований глобальной матрицы жесткости, используемой в методе конечных элементов. **Основные результаты.** В качестве примера была решена тестовая задача Кирша в плоско-напряженной постановке: к верхней грани стальной пластинки с круглым вырезом в центре приложена распределенная растягивающая нагрузка, нижняя грань пластинки жестко закреплена. Проведя расчет, наблюдаем погрешность в 1,72 % относительно аналитического решения. Такое значение погрешности считается низким, следовательно, разработанное программное обеспечение не просто способствует простой подготовке данных для расчета, но и гарантирует высокую точность полученных результатов. **Практическая значимость.** Коммерческое программное обеспечение для решения задач механики деформируемого твердого тела, такое как ANSYS Mechanical APDL, Abaqus и т. д., является очень дорогим. Свободное программное обеспечение преимущественно ориентировано на научных сотрудников и, как правило, является сложным для освоения рядовым пользователем-инженером, а компромиссный вариант PDE Toolbox для MATHLAB применим только для задач в двумерной области и поддерживает только линейный треугольный конечный элемент. Однако использование API GMSH и библиотеки Eigen позволяет создать простой в использовании, но мощный инструмент для решения задач механики деформируемого твердого тела.

Ключевые слова

API, GMSH, базы данных, MySQL, Eigen, C++, механика деформируемого твердого тела

Благодарности

Работа выполнена при поддержке Российского научного фонда (РНФ) (проект № 16-19-10264).

doi: 10.17586/2226-1494-2019-19-5-939-946

SOFTWARE FOR DEFORMABLE SOLID MECHANICS

A.A. Popov

Tomsk State University, Tomsk, 634050, Russian Federation
Corresponding author: popov.alexey1997@gmail.com

Article info
Received 04.06.19, accepted 23.07.19
Article in Russian

For citation: Popov A.A. Software for deformable solid mechanics. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2019, vol. 19, no. 5, pp. 939–946 (in Russian). doi: 10.17586/2226-1494-2019-19-5-939-946

Abstract

Subject of Research. The paper presents a software building method for the tasks of deformable solid mechanics. This software should guarantee high accuracy and speed of calculations, as well as simple preparation of the initial data and data processing even for an inexperienced user. The software was developed using the open source GMSH mesh generator application programming interface (API) and the Eigen mathematical library. **Method.** The developed software consists of three modules: GMSH_API, InputFile, FEMSolver and a database. The GMSH_API module, which prepares the finite element model, was written using the GMSH mesh generator API. The InputFile module describes methods for interacting with a previously created database, that provides quick and easy preparation of the input file needed to start the calculation. Numerical calculation by the finite element method is implemented in the FEMSolver module. The Eigen mathematical library was actively used for its implementation, and it can build sparse matrices that do not store zero elements in memory. This possibility obviates the need for additional transformations of the global stiffness matrix used in the finite element method. **Main Results.** The Kirsch task was solved as an example in a plane-stressed setting: a distributed tensile load was applied to the upper edge of a steel plate, with a round hole in the center, the lower edge of the plate was rigidly fixed. After calculating and obtaining the von Mises stress distribution field in the plate, we observe an error of 1.72% relative to the analytical solution. Such error value is considered low, therefore, the developed software not only facilitates the preparation of data for calculation, but also guarantees high accuracy of the obtained results. **Practical Relevance.** Commercial software for solving the problems of deformable solid mechanics, such as ANSYS Mechanical APDL, Abaqus, etc., is very expensive. Free software is primarily focused on researchers and, as a rule, is difficult for learning by an ordinary user-engineer, and the compromise version of the PDE Toolbox for MATHLAB is applicable only for tasks in a two-dimensional area and only supports a linear triangular finite element. However, the application of GMSH API and the Eigen library provides for creation of an easy-to-use but powerful tool for solving the problems of deformable solid mechanics.

Keywords

API, GMSH, databases, MySQL, Eigen, C++, deformable solids mechanics

Acknowledgments

This work was supported by the Russian Science Foundation (RSF) (project No. 16-19-10264).

Введение

В настоящее время существует значительное количество как коммерческого, так и свободного программного обеспечения, предназначенного для решения различных инженерных задач и задач механики деформируемого твердого тела (МДТТ) в частности. Коммерческое программное обеспечение, такое как ANSYS Mechanical APDL, Abaqus и т. д., хотя и способно решать сложные задачи МДТТ, но является очень дорогим. Свободное программное обеспечение преимущественно ориентировано на научных сотрудников и, как правило, является сложным для освоения рядовым пользователем-инженером, а компромиссный вариант в лице PDE Toolbox для MATHLAB применим только для задач в двумерной области и поддерживает только линейный треугольный конечный элемент [1]. В связи с этим актуальной остается проблема создания простого в использовании, но мощного инструмента для решения задач МДТТ [2]. Для предоставления широкого выбора в классе решаемых задач и типе конечного элемента, воспользуемся интерфейсом прикладного программирования сеточного генератора GMSH [3, 4].

GMSH представляет собой быстрый, легкий и удобный инструмент для создания конечно-элементных моделей, обладающий параметрическим вводом и расширенными возможностями визуализации [3]. Для создания геометрических моделей используются два связанных движка – собственный движок GMSH для создания примитивных геометрических моделей и движок Open CASCADE для создания моделей с конструктивными особенностями [3]. GMSH объединяет несколько алгоритмов для генерации конечно-элементных сеток. Каждый алгоритм имеет свои преимущества и недостатки. Выбор алгоритма будет зависеть от типа поверхности рассматриваемой геометрической модели. Также GMSH использует интерфейсы дополнительных внешних библиотек, таких как Tetgen, Netgen и Mmg3d [4, 5].

Для целесообразного использования интерфейса прикладного программирования (API) GMSH расчетный модуль, в котором реализован численный расчет методом конечных, написан таким образом, чтобы добавление возможности решения задач в трехмерных областях и/или использование другой физико-математической модели требовало минимум временных затрат. При написании этого модуля активно использовалась математическая библиотека Eigen, позволяющая строить разреженные матрицы, не хранящие в памяти нулевые элементы [6, 7]. Такая возможность избавляет от дополнительных преобразований глобальной матрицы жесткости, используемой в методе конечных элементов (МКЭ).

Архитектура разрабатываемого программного обеспечения

При проектировании данного программного обеспечения использовалась модульная архитектура (рис. 1) [8]. Программное обеспечение состоит из трех модулей: GMSH_API — модуль, в котором описан набор классов для создания и работы с различными геометрическими фигурами и генерации конечно-элементных сеток, InputFile — модуль, в котором описаны методы записи данных, полученных после работы сеточного генератора, в предварительно созданную базу данных (БД), а также методы их унификации и хранения для быстрой генерации входного файла, по которому будет проводиться расчет, FEMSolver —

модуль, в котором реализован численный расчет методом конечных элементов [9] на основании данных полученных из входного файла.

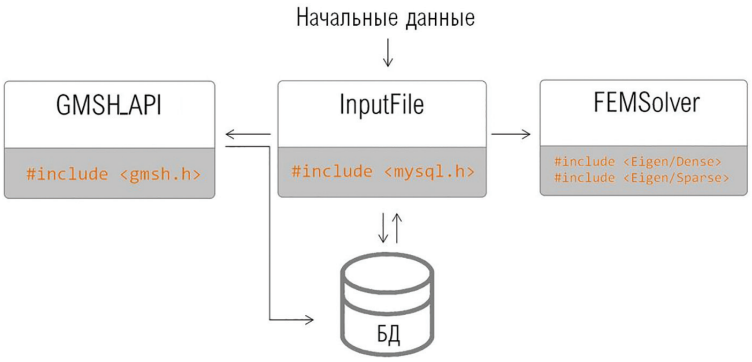

Рис. 1. Архитектура разрабатываемого программного обеспечения

В качестве начальных данных выступают параметры простых геометрических фигур, из которых состоит геометрическая модель. Эти данные выступают в качестве входных параметров для конструкторов соответствующих классов в модуле GMSH_API, и именно с этих данных начинается формироваться входной файл.

Создание геометрической модели и генерация конечно-элементной сетки

Пожалуй, самым важным этапом при подготовке данных для расчета является этап создания геометрической модели исследуемого тела и генерации для этой модели качественной конечно-элементной сетки [10].

При создании модуля, отвечающего за построение конечно-элементной модели, будем использовать API свободного сеточного генератора с открытым исходным кодом GMSH.

В GMSH имеется несколько способов для управления размерами конечных элементов, но самым гибким и универсальным подходом является передача специального параметра — шага сетки в функцию для добавления точки. Данный подход позволяет не только получить возможность управления размерами конечных элементов, но и, если это необходимо, построить как структурированные (рис. 2), так и неструктурированные (рис. 3) конечно-элементные сетки.

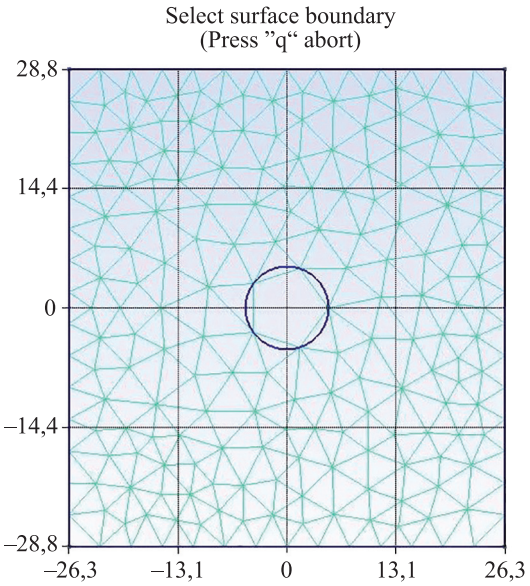

Рис. 2. Структурированная конечно-элементная сетка

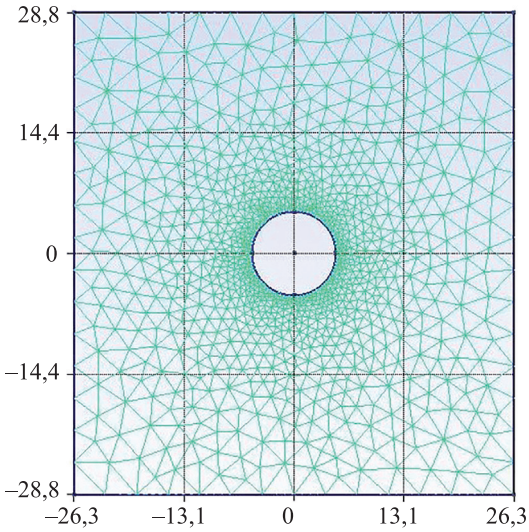

Рис. 3. Неструктурированная конечно-элементная сетка со сгущением элементов в области выреза

В связи с этим целесообразно, используя API GMSH, написать собственные классы для работы с различными геометрическими фигурами, методы которых описываются с учетом особенностей GMSH, а также класс для работы с конечно-элементной моделью в целом [11].

Создан базовый класс Contour, который принимает в конструктор значения количества точек и количества линий, а также значение шага сетки. Используя метод getContourTag(), получаем идентификатор созданного контура. Базовый класс не является абстрактным, поэтому, создавая объекты данного класса, можно получить произвольный контур. Часто для построения геометрической модели требуются примитивные фигуры. Следовательно, целесообразно написать классы наследники для работы с такими геометрическими фигурами. Схема иерархии классов представлена на рис. 4. Каждая созданная фигура имеет свой уникальный идентификатор.

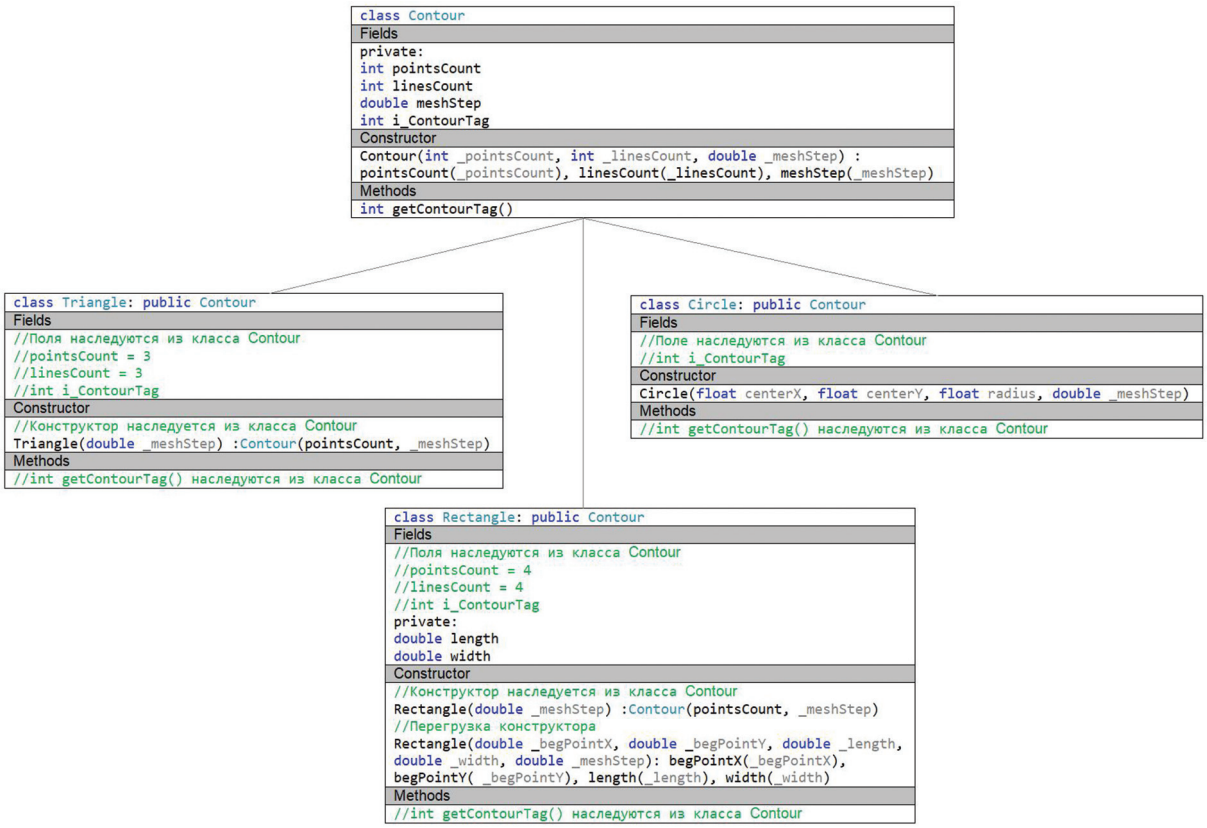

Рис. 4. Схема иерархии классов для работы с различными геометрическими фигурами

Класс Model (рис. 5) необходим для работы с геометрической моделью, построенной из таких геометрических фигур и произвольных контуров. Также в данном классе описаны методы генерации различных конечно-элементных сеток для конкретной геометрической модели. Конструктор этого класса принимает в качестве входных параметров: вектор идентификаторов фигур, необходимых для построения геометрической модели, имя этой геометрической модели и ее толщину (следует отметить, что данная перегрузка конструктора подходит только в случае решения задачи в плоско-напряженной постановке).

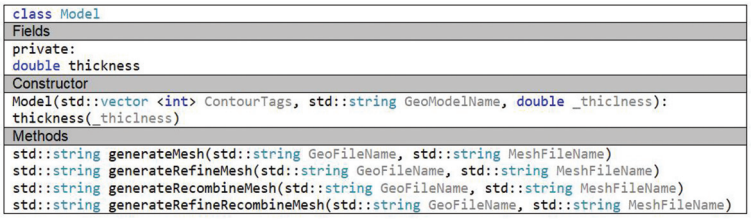

Рис. 5. Класс Model

Метод generateRecombineMesh (std::string GeoFileName, std::string MeshFileName) позволяет изменить тип конечных элементов с треугольных на четырехугольные. Методы generateRefineMesh (std::string GeoFileName, std::string MeshFileName) и generateRefineRecombineMesh (std::string GeoFileName, std::string MeshFileName) позволяют построить улучшенные треугольные и четырехугольные сетки соответственно.

Организация работы с базой данных и генерация входного файла посредствам обращения к ней

После получения конечно-элементной сетки для составления и последующего решения системы дифференциальных уравнений необходимо задать граничные условия и характеристики материала. Для того чтобы сделать это было максимально быстро и удобно, результат работы сеточного генератора передается в предварительно созданную реляционную базу данных, состоящую из следующих таблиц [12, 13]:

— Nodes, которая хранит в себе номера и координаты узлов конечных элементов;

— Elements, содержащую номера элементов, номера узлов, входящих в каждый элемент, и номер материала конечного элемента;

— Materials, хранящую в себе номер и название материала, а также его константы;

— Displacements и Loadings – таблицы граничных условий.

Схема «сущность–связь» используемой базы данных представлена на рис. 6. Номер узла приложения граничного условия в таблицах Displacements и Loadings является внешним ключом к столбцу idNode таблицы Nodes. Некоторые поля этих таблиц могут быть нулевыми в зависимости от вида граничных условий. В качестве системы управления базой данных используется MySQL [14]. Для обеспечения взаимодействия между программой и базой данных используется MySQL Connector/C++.

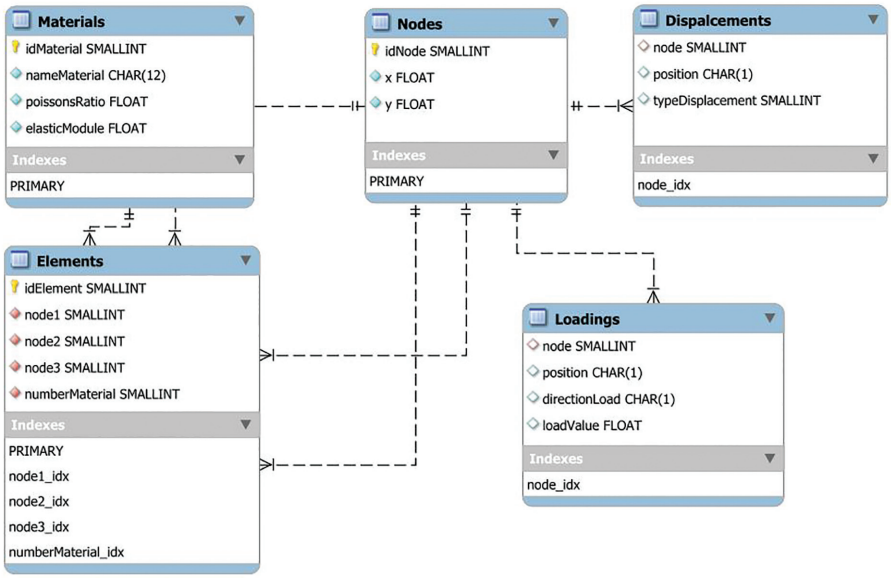

Рис. 6. Схема «сущность–связь» используемой базы данных

Использование базы данных не только обеспечивает удобное хранение, обращение и манипулирование данными, но и в случае необходимости способствует более легкой модификации программы [14, 15]. Ведь без труда можно добавить по столбцу в таблицы Nodes и Elements при изменении размерности решаемых задач или дополнительные столбцы для констант материала при использовании другой физико-математической модели. В результате обращения к базе данных формируется входной файл, на основании которого проводится расчет.

Реализация метода конечных элементов

Расчет, реализованный в модуле FEMSolver, производится на основании данных, полученных из входного файла. Для хранения этих данных, а также промежуточных данных, созданы две структуры, описывающие конечный элемент и граничные условия [6]. В процессе реализации метода конечных элементов активно использовалась математическая библиотека Eigen [6, 7], которая предоставляет возможность строить разреженные матрицы, не хранящие в памяти нулевые элементы. Так как глобальная матрица жесткости, используемая в методе конечных элементов, содержит в себе большое число нулевых элементов, такая возможность позволяет эффективно работать с памятью, избавляя от дополнительных преобразований этой матрицы.

Собственно работа метода конечных элементов и сводится к построению глобальной матрицы жесткости и решению образованной от нее системы линейных уравнений [16, 17]. Глобальная матрица жесткости представляет собой суперпозицию локальных матриц жесткости каждого конечного элемента и является коэффициентом пропорциональности между узловыми силами и перемещениями. Для вычисления значений локальных матриц написан специальный метод, в конце которого заполняется вектор triplets, каждый элемент которого создается, используя массив индексов узлов элементов для определения его положения

в глобальной матрице жесткости. Пройдясь в цикле по каждому элементу и заполнив вектор, используя вышеописанный метод, значениями локальных матриц, будет построена глобальная матрица жесткости.

Для существования единственного решения полученной системы уравнений необходимо добавить граничные условия. Для этого была написана специальная функция, принимающая в качестве входных параметров глобальную матрицу жесткости и вектор граничных условий. Для решения системы уравнений используется прямой решатель из библиотеки Eigen — SimplicialLDLT [6]. Результат решения представляет собой вектор перемещений. Для анализа результата целесообразно от перемещений перейти к деформациям, а далее к напряжениям и уже из полученных напряжений получить инвариантную величину – напряжения по Мизесу.

Полученные результаты и их обсуждение

В качестве примера рассмотрим тестовую задачу Кирша в плоско-напряженной постановке. Имеем стальную пластину с круглым отверстием в центре, нижняя грань которой закреплена, а к верхней грани приложена растягивающая распределенная нагрузка. Для того чтобы легко и быстро определить, сходится ли полученное решение с аналитическим, зададим продольную силу в номинальном сечении таким образом, чтобы номинальное напряжение было равным 1 кг/см^2. Ведь в случае малого отверстия наибольшее напряжение должно быть равным трем номинальным. Завершив расчет и получив поэлементные значения напряжений в пластине (рис. 7, *а*), получаем погрешность в 1,72 % относительно аналитического решения. Аналогичная задача также была решена в ANSYS Mechanical APDL [18, 19], и погрешность составила целых 7,81 % (рис. 7, *б*).

а *б*

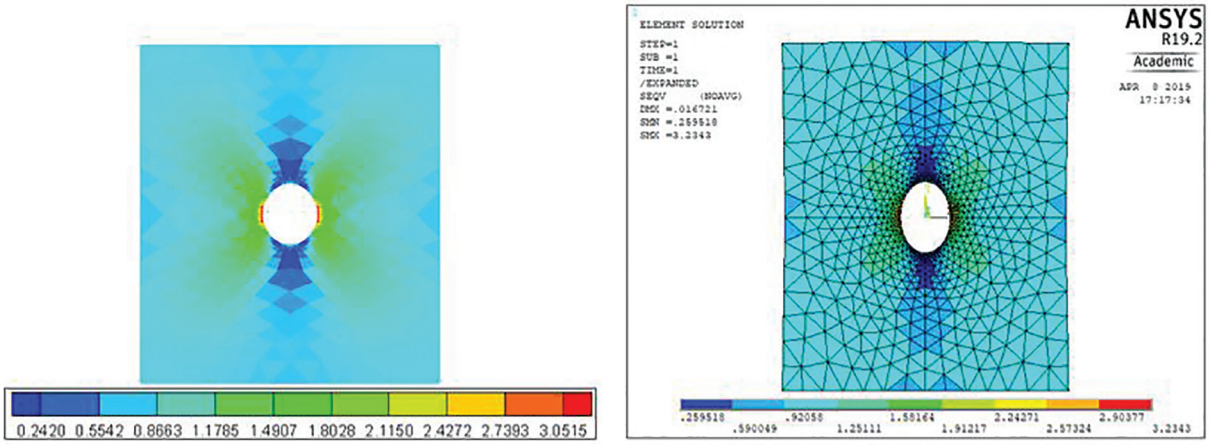

Рис. 7. Поэлементное распределение напряжений по Мизесу: *а* — полученное с помощью разрабатываемого программного обеспечения; *б* — полученное в ANSYS Mechanical APDL

Только при проведении в ANSYS Mechanical APDL перерасчета значений напряжений по Мизесу с элементов на узлы конечно-элементной сетки, проведя аппроксимацию, получили значение погрешности равное 1,56 % (рис. 8), которое всего на 0,16 % меньше значения погрешности, полученного нами, притом что в нашем случае перерасчет и аппроксимация не проводились.

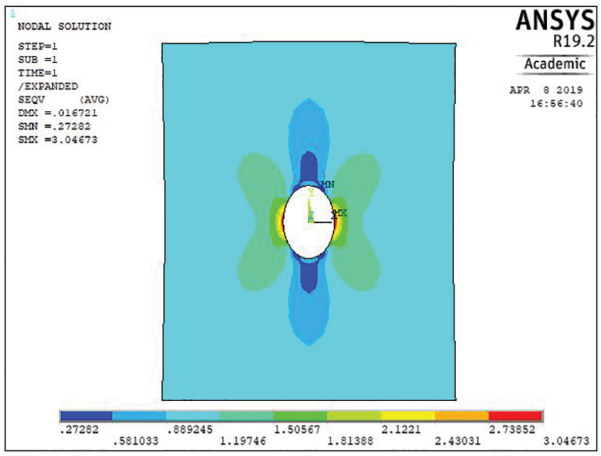

Рис. 8. Поле распределения напряжения, полученное в ANSYS Mechanical APDL, после проведения аппроксимации

Заключение

При создании простого в использовании, но мощного инструмента для решения задач механики деформируемого твердого тела было предложено использовать API сеточного генератора GMSH и математическую библиотеку Eigen.

Модуль, отвечающий за создание конечно-элементной модели, был написан с использованием API GMSH. Описанный в нем набор классов позволяет создавать качественные конечно-элементные модели для широкого класса задач. Для целесообразного использования API GMSH, расчетный модуль, в котором реализован метод конечных элементов, при минимальных изменениях может быть применим для решения задач в трехмерных областях и/или с использованием другой физико-математической модели.

Использование библиотеки Eigen при написании расчетного модуля позволяет не только работать с разреженными матрицами, не хранящими в памяти нулевые элементы, но и предоставляет возможности для их удобного создания, заполнения и обработки. Также использование этой библиотеки позволяет применять уже готовый решатель для решения полученной системы линейных уравнений, к которой сводится метод конечных элементов.

На данном этапе работы разработанное программное обеспечение позволяет решать только задачи теории упругости в плоско-напряженной постановке, но модули, из которых состоит программа, написаны таким образом, чтобы последующая модификация программного обеспечения требовала минимума временных затрат программиста. Простой модификации способствуют применение мощного сеточного генератора GMSH и использование базы данных, ведь без труда можно добавить по столбцу в таблицы Nodes и Elements при изменении размерности решаемых задач или дополнительные столбцы для констант материала при использовании другой физико-математической модели.

Литература

1. Ануфриев И.Е. Применение PDE Toolbox при изучении некоторых разделов вычислительной математики // Труды III научной конференции «Проектирование инженерных и научных приложений в среде MATLAB», 23-26 октября 2007 г. С. 42–54.
2. Курепин М.П., Сербиновский М.Ю. Эффективные методики конечно-элементного моделирования сложных конструкций энергетического машиностроения // Современные наукоемкие технологии. 2017. № 10. С. 19–25.
3. Geuzaine C., Remacle J.-F. Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities // International Journal for Numerical Methods in Engineering. 2009. V. 79. N 11. P. 1309–1331. doi: 10.1002/nme.2579
4. Geuzaine C., Remacle J. Gmsh Reference Manual. 2001. March. 252 p.
5. Avdis A., Mouradian S.L. A Gmsh tutorial. Imperial College London, Applied Modelling and Computation Group (AMCG), 2012. 29 p.
6. Подгорский С. Написание МКЭ расчетчика в менее чем 180 строк кода. Habr [Электронный ресурс]. URL: https://habr.com/ru/post/271723/, свободный. Яз. рус. (дата обращения: 03.06.2019).
7. Никехин А.А. Основы C++ для моделирования и расчетов. Часть 2. Библиотеки для научных вычислений: Учебное пособие. СПб.: Университет ИТМО, 2016. С. 15–23.
8. Mistrik I., Bahsoon R., Eeles P., Roshandel R., Stal M. Relating System Quality and Software Architecture. Morgan Kaufman, 2014. 420 p.
9. Lui G.R., Quek S.S. The Finite Element Method: A Practical Course. Butterworth-Heinemann, 2003. 384 p.
10. Guo J., Ding F., Jia X., Yan D-M. Automatic and high-quality surface mesh generation for CAD models // Computer-Aided Design. 2019. V. 109. P. 49–59. doi: 10.1016/j.cad.2018.12.005
11. Lattanzi M., Henry S. Software reuse using C++ classes: The question of inheritance // Journal of Systems and Software. 1998. V. 41. N 2. P. 127–132. doi: 10.1016/S0164-1212(97)10013-9
12. Harrington J.L. Relational Database Design and Implementation. 4th ed. Morgan Kaufman, 2016. 712 p.
13. Федорук В.Г. Основы языка SQL. Учебное пособие. МГТУ имени Н.Э. Баумана. [Электронный ресурс]. URL: http://rk6.bmstu.ru/electronic_book/iosapr/sql/sql_tutor.html/, свободный. Яз. рус. (дата обращения: 03.06.2019).
14. Harrington J.L. SQL Clearly Explained. A volume in The Morgan Kaufmann Series in Data Management Systems. 3rd ed. Morgan Kaufman, 2003. 352 p.

References

1. Anufriev I.E. The usage of the PDE Toolbox in the investigation on certain sections of computational mathematics. *Proc. 3rd scientific conference "Designing Engineering and Scientific Applications in the MATLAB Environment", October 23-26*, 2007, pp. 42–54. (in Russian)
2. Kurepin M.P., Serbinovskiy M.Yu. Efficient methods of finite-element analysis of energetic machinery complex structures. *Modern high technologies*, 2017, no. 10, pp. 19–25. (in Russian)
3. Geuzaine C., Remacle J.-F. Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering*, 2009, vol. 79, no. 11, pp. 1309–1331. doi: 10.1002/nme.2579
4. Geuzaine C., Remacle J. *Gmsh Reference Manual*, 2001, March, 252 p.
5. Avdis A., Mouradian S.L. *A Gmsh tutorial*. Imperial College London, Applied Modelling and Computation Group (AMCG), 2012, 29 p.
6. Podgorsky S. *Coding the FEM of the calculator in less than 180 lines of code*. *Habr*. Available at: https://habr.com/ru/post/271723/ (accessed: 03.06.2019). (in Russian)
7. Nikekhin A.A. *The basics of C ++ for modeling and calculations. Part 2. Libraries for Scientific Computing*. Handbook. St. Petersburg, ITMO University, 2016, pp. 15–23. (in Russian)
8. Mistrik I., Bahsoon R., Eeles P., Roshandel R., Stal M. *Relating System Quality and Software Architecture*. Morgan Kaufman, 2014, 420 p.
9. Lui G.R., Quek S.S. *The Finite Element Method: A Practical Course*. Butterworth-Heinemann, 2003, 384 p.
10. Guo J., Ding F., Jia X., Yan D-M. Automatic and high-quality surface mesh generation for CAD models. *Computer-Aided Design*, 2019, vol. 109, pp. 49–59. doi: 10.1016/j.cad.2018.12.005
11. Lattanzi M., Henry S. Software reuse using C++ classes: The question of inheritance. *Journal of Systems and Software*, 1998, vol. 41, no. 2, pp. 127–132. doi: 10.1016/S0164-1212(97)10013-9
12. Harrington J.L. *Relational Database Design and Implementation*. 4th ed. Morgan Kaufman, 2016, 712 p.
13. Fedoruk V.G. *Concepts on SQL. Handbook – Bauman MSTU*. Available at: http://rk6.bmstu.ru/electronic_book/iosapr/sql/sql_tutor.html/ (accessed: 03.06.2019). (in Russian)
14. Harrington J.L. *SQL Clearly Explained. A volume in The Morgan Kaufmann Series in Data Management Systems*. 3rd ed. Morgan Kaufman, 2003, 352 p.
15. Donahoo M.J., Speegle G.D. *SQL: Practical Guide for Developers. A volume in The Morgan Kaufmann Practical Guide Series*. Morgan Kaufman, 2005, 272 p.

15. Donahoo M.J., Speegle G.D. SQL: Practical Guide for Developers. A volume in The Morgan Kaufmann Practical Guide Series. Morgan Kaufman, 2005. 272 p.
16. Зенкевич О.К. Метод конечных элементов в технике. М.: Мир, 1975. 541 с.
17. Rao S.S. The Finite Element Method in Engineering. Butterworth-Heinemann, 2018. 782 p.
18. Каплун А.Б., Морозов Е.М., Шамраева М.А. ANSYS в руках инженера: практическое руководство. 5-е изд. М.: URSS ЛЕНАНД, 2017. 269 с.
19. Thompson M., Thompson J. ANSYS Mechanical APDL for Finite Element Analysis. Butterworth-Heinemann, 2017. 466 p.
16. Zienkiewiez O.C. *The finite element method in engineering science*. London, 1971.
17. Rao S.S. *The Finite Element Method in Engineering*. Butterworth-Heinemann, 2018, 782 p.
18. Kaplun A.B., Morozov E.M., Shamraeva M.A. *ANSYS in the hands of an engineer*. Moscow, Librocom, 2017, 269 p. (in Russian)
19. Thompson M., Thompson J. *ANSYS Mechanical APDL for Finite Element Analysis*. Butterworth-Heinemann, 2017, 466 p.

Авторы

Попов Алексей Алексеевич — студент, Томский государственный университет, Томск, 634050, Российская Федерация, Scopus ID: 57203556236, ORCID ID: 0000-0002-0260-2297, popov.alexey1997@gmail.com

Authors

Alexey A. Popov — student, Tomsk State University, Tomsk, 634050, Russian Federation, Scopus ID: 57203556236, ORCID ID: 0000-0002-0260-2297, popov.alexey1997@gmail.com