МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Сумський державний університет
Факультет технічних систем та енергоефективних технологій
Кафедра комп'ютерної механіки імені Володимира Марцинковського

«До захисту допущено»
Завідувач кафедри
_____Андрій ЗАГОРУЛЬКО
(підпис)
____ _____ 2023 р.

КВАЛІФІКАЦІЙНА РОБОТА
на здобуття освітнього ступеня магістр

зі спеціальності 131 Прикладна механіка, освітньо-професійної програми «Комп'ютерна механіка», на тему: Розроблення комп'ютерної програми з графічним інтерфейсом користувача для розрахунку критичних частот ротора на основі мови програмування C#.

Здобувача групи КМ.м-21 РОЩУПКІНА Олександра Вікторовича.

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

_____ Олександр РОЩУПКІН
(підпис)

Керівник: професор, д.т.н., професор Іван ПАВЛЕНКО _____
(підпис)

Суми – 2023

Анотація

В сучасному інженерному виробництві дослідження критичних частот обертання роторів є важливою складовою для забезпечення безпеки та ефективності роботи різноманітних механічних систем. Зокрема, у випадку обертових машин, таких як турбіни, насоси чи електродвигуни, виявлення критичних частот обертання ротора має вирішальне значення для запобігання виникненню небезпечних резонансних явищ.

Один із сучасних та ефективних підходів до вивчення динаміки обертання роторів є використання методу скінченних елементів (МСЕ). Метод скінченних елементів є числовим методом аналізу, що дозволяє моделювати та аналізувати складні інженерні структури, включаючи ротори, з високою точністю та ефективністю.

Основним перевагам використання МСЕ в дослідженнях критичних частот обертання роторів є можливість врахування різноманітних фізичних властивостей матеріалів, геометрії деталей та впливів експлуатаційних навантажень. Це дозволяє інженерам отримати більш точні та реалістичні результати, які можуть служити основою для прийняття обґрунтованих рішень щодо конструкції та експлуатації роторних систем.

У цьому контексті робота спрямована на дослідження та аналіз критичних частот обертання роторів з використанням методу скінченних елементів, зокрема, враховуючи його застосування для визначення стійкості та надійності роторних систем у різноманітних інженерних областях.

Ця робота націлена зробити внесок у розвиток сучасного програмного забезпечення для інженерного аналізу динаміки обертання роторів і розширити розуміння та можливості застосування методу скінченних елементів у вирішенні актуальних завдань інженерії роторних систем.

Зміст

1 Огляд літератури

Для вирішення поставленої задачі, тобто створення програмного забезпечення для знаходження власних частот коливань ротора будемо використовувати метод скінченних елементів. Як будь який математичний метод, метод скінченних елементів спирається на фундаментальні математичні методи та закони. Тож, щоб ефективно сприймати інформацію з наведених джерел та запрограмувати сам метод, необхідна додаткова інформація. Збір цієї інформації можна поділити на підзадачі:

- Пошук інформації з описом методу
- Пошук інформації з фундаментальної математики необхідної для розуміння методу
- Пошук інформації із специфічних проблем, які розглядаються в роботі
- Пошук інформації з необхідних допоміжних чисельних методів
- Пошук інформації, необхідних для розробки ефективних алгоритмів

1.1 Джерела з загальним описом методу скінченних елементів

Загальні теоретичні викладки метода добре описані та структуровані в [1] та [2].

Посібник [1] розроблений як підручник для бакалаврату та початкового рівня магістратури, залежно від того, чи розглядаються більш складні теми, які з’являються в кінці кожного розділу. Книга має рівень, зрозумілий студентам молодших і старших курсів у природничих науках і інженерії. З включеними темами для просунутого рівня книга може слугувати підручником для першого курсу в скінченних елементів на аспірантурі. Текстовий матеріал є результатом більш ніж 50 років спільного викладання досвіду авторів аспірантських та студентських робіт, присвячених методу скінченних елементів. Посібник містить теорію скінченних елементів, розробку коду скінченних елементів та застосування комерційного пакету програм. В посібнику присутні практичні роботи з розробки моделей кінцевих елементів з прикладами в MATLAB.

У [2] розглянуто основи методу скінченних елементів, охоплюючи фундаментальну теорію. Наведено та детально розібрані приклади моделей та розрахунків, необхідних для того, щоб читачі могли застосувати знання до власних інженерних проблем і зрозуміти більш складні програми. Також розглянуто деякі нові розділи, такі як слабкі форми моделей, варіативні форми моделей, задачі багатовимірного поля, автоматична генерація сітки, згинання пластин і оболонок, розвиток безсіткових технологій, тощо. Посібник є зведенням математичних та аналітичних інструментів, необхідних для успішного застосування, та є авторитетним ресурсом для студентів, дослідників та професійних інженерів, які займаються інженерним аналізом на основі скінченних елементів.

1.2 Джерела з фундаментальної математики

В якості посібників з лінійної алгебри розглянемо [3] та [4]. В джерелі [3] автор досить детально розглядає класичні задачі та методи лінійної алгебри з великою кількістю прикладів вирішень та роз’яснень в досить простих термінах та без перевантаження математичною нотацією. Тож цей посібник дуже добре підходить для поновлення навичок та знань з лінійної алгебри, які будуть в нагоді під час виконання роботи.

Джерело [4] є класичним посібником з лінійної алгебри для багатьох навчальних закладів Європи. Тут вичерпно та детально розглянуто такі теми, як вирішення векторних та матричних рівнянь, та систем лінійних та нелінійних рівнянь, які широко використовуються в методі скінченних елементів.

В теоретичних викладках методу скінчених елементів досить багато елементів диференційного та інтегрального числення, тож для повного оволодіння матеріалом також необхідний відповідний посібник. В якості такого розглянемо [5]. У посібнику докладно розглянуто необхідні теми, а також приведено багато прикладів вирішення найбільш типових задач.

1.3 Джерела з описом специфічних проблем, які вирішуються за допомогою методу скінченних елементів

Маючи необхідну математичну базу, для більш детального ознайомлення із застосуванням методу скінченних елементів безпосередньо в задачах супротиву матеріалів та теорії пружності, будемо використовувати посібник [6], в якому надано загальну інформації щодо використання методу в задачах механіки, та, що більш цінно, наведено багато прикладів побудування математичних моделей, та їх вирішення для найрізноманітніших технічних задач.

Для переходу від загальних задач механіки до власне задач динаміки роторних машин, будемо використовувати джерело [7], в якому зведено теоретичну інформацію щодо використання методу скінченних елементів для дослідження коливальних динамічних систем, до яких і належить задача аналізу динаміки роторних машин. Також в посібнику приведено багато прикладів розрахунків для різноманітних конфігурацій коливальних систем.

Так як застосування методу скінченних елементів для дослідження динаміки роторних машин спирається на дисципліну опору матеріалів, в розрахунках широко використовуються залежності та закони цієї дисципліни, тож для роботи також буде в нагоді посібник з опору матеріалів, в якості якого розглянемо [8]. В посібнику автор розкриває більшість класичних проблем дисципліни опору матеріалів, також навівши велику кількість інформативних схем та прикладів. Тож інформації цього посібника цілком достатньо для роботи.

1.4 Джерела з чисельних методів

Математичні викладки в аналітичному виді запрограмувати досить важко, тому щоб створити програму, яка базується на розробленому математичному апараті, спершу потрібно перевести цей математичний апарат до чисельного виду. В якості посібників по цій темі розглянемо [9] та [10]. Джерело [9] є загальновизнаним класичним посібником з чисельних методів. В посібнику розібрано більшість проблем, які вирішуються за допомогою чисельних методів, а також наведено стислі алгоритми в математичній нотації, які стануть в нагоді під час розробки програмних алгоритмів. Посібник [10] містить теорію щодо більшості чисельних методів, а також багато алгоритмів чисельних методів у виді псевдокоду, чим користуватися зручніше, ніж алгоритмами в математичній нотації. Крім того, в посібнику детально розглянуто алгоритми знаходження власних значень та векторів матриць. Цей математичний апарат буде використовуватись в роботі, тож цей посібник буде в нагоді для ознайомлення з цією темою.

1.5 Джерела з теорії алгоритмів

В якості посібника з теорії алгоритмів розглянемо [11]. Це підручник, який вже класичним для вивчення теорії алгоритмів, до розглянуто та детально розібрано безліч найбільш актуальних та розповсюджених алгоритмів.

У програмному забезпеченні, яке буде розроблено в рамках цього проєкту вочевидь буде проблема перемноження досить великих за розміром матриць, тож класичні методи перемноження можуть виявитися неефективними та повільними. Тож додатково слід вивчити методи швидкого перемноження матриць. В якості джерел для цього можна використати [12] та [13]. В джерелах описано два різних методи для перемноження матриць, які за ствердженням авторів швидше наївного алгоритму в два і більше разів.

2 Огляд існуючих рішень

Наразі існує велика кількість продуктів для інженерного аналізу, використовуючих метод скінченних елементів. Розглянемо кілька таких продуктів в контексті поточної задачі. Найбільш відомими представниками такого типу продуктів є ANSYS Workbench та COMSOL Multiphysics. Слід зазначити, що обидва продукта є платним програмним забезпеченням, і для їх використання необхідно придбати ліцензію. Також існують безкоштовні продукти з відкритим програмним кодом. В якості таких розглянемо Elmer та FEATool.

2.1 ANSYS Workbench

Ansys Workbench - це комплексне програмне забезпечення для комп'ютерного моделювання, яке дозволяє вирішувати широкий спектр інженерних задач. Воно складається з набору модулів, які можна використовувати окремо або разом, щоб створити модель для конкретного завдання.

Однією з основних переваг Ansys Workbench є його модульна структура. Це дозволяє користувачам вибирати лише ті модулі, які необхідні для їхньої задачі. Це робить продукт більш доступним, ніж інші програми для комп'ютерного моделювання, які вимагають придбання всієї програми.

Ansys Workbench також має потужний графічний інтерфейс, який робить його простим у використанні навіть для користувачів без досвіду в комп'ютерному моделюванні. Інтерфейс розділений на кілька вікон, кожне з яких відповідає за певну частину процесу моделювання. Це робить його легко зрозумілим і зручним у використанні.

Ansys Workbench використовується для розрахунків та моделювання в багатьох галузях науки і техніки, таких, як механіка, гідроаеродинаміка, термодинаміка, електрика та магнетизм, акустика тощо.

2.2 COMSOL Multiphysics

Comsol Multiphysics - це програмне забезпечення для комп'ютерного моделювання, яке дозволяє досліджувати взаємодію різних фізичних процесів. Воно використовується в різних галузях науки і техніки, включаючи інженерію, фізику, хімію, матеріалознавство, біологію та медицину.

Comsol Multiphysics має ряд переваг перед іншими програмами для комп'ютерного моделювання. По-перше, продукт дозволяє моделювати взаємодію різних фізичних процесів в єдиній моделі. Це дає можливість отримувати більш точні результати, ніж при використанні окремих моделей для кожного процесу. По-друге, Comsol Multiphysics має широкий спектр вбудованих моделей і модулів, що дозволяє швидко і легко створювати моделі для різних задач.

Comsol Multiphysics є потужним і універсальним інструментом для комп'ютерного моделювання. Воно використовується в багатьох галузях науки і техніки і дозволяє досліджувати взаємодію різних фізичних процесів, таких як:

- проектування інженерних систем, таких як автомобілі, літальні апарати, будівлі та мости;
- дослідження фізичних процесів, таких як теплопередача, електромагнетизм, гідродинаміка та механіка;
- розробки нових матеріалів і ліків;
- дослідження біологічних систем, та інше.

2.3 Elmer

Elmer — це програмне забезпечення для багатофізичного моделювання з відкритим кодом, розроблене в основному фінськими університетами, та дослідницькими інститутами. Після публікації з відкритим вихідним кодом використання та розробка Elmer стали значною мірою міжнародними.

Elmer включає фізичні моделі динаміки рідини, структурної механіки, електромагнетики, теплообміну та акустики тощо. Вони описуються диференціальними рівняннями в частинних похідних, які Elmer розв’язує методом кінцевих елементів (FEM). Elmer підтримує паралельні обчислення. Для багатьох проблем досягається хороша масштабованість на тисячі ядер.

Порівняно із попередніми комплексами, продукт виглядає досить скромно. Також Elmer відрізняється досить складним та не дуже зрозумілим для початківця інтерфейсом. Тож для початку роботи треба потратити деякий час на роботу із документацією та на вправи для початківця. Це стосується не тільки власне розрахунків, але й моделювання.

Тим не менш, палітра його можливостей досить велика.

2.4 FEATool Multiphysics

FEATool Multiphysics інструментарій для моделювання та симуляції змішаних фізичних та інженерних задач.

FEATool має на меті надати простий у використанні та повністю інтегрований комплексний пакет моделювання для всіх типів мультифізичного аналізу та поєднати найкраще з простоти використання, потужної функціональності та розширюваності як для початківців, так і для досвідчених користувачів. Такі функції, як інтуїтивно зрозумілий і простий у вивченні графічний інтерфейс користувача (GUI), повна бібліотека генерації сітки та функції постобробки, а також програмування інтерфейсу командного рядка (CLI), а також інтерактивне та інтерпретоване програмування та можливості створення сценаріїв, роблять FEATool придатним для всіх, від студентів, які вивчають математичне моделювання, до професіоналів та інженерів, які бажають досліджувати нові ідеї простим, швидким і зручним способом.

2.5 Порівняльний аналіз продуктів

Якісний порівняльний аналіз проведемо окремо для платних та безплатних продуктів, так як порівнювати продукти цих категорій між собою немає ніякого сенсу через масштабність та «вік» продуктів. На відміну від Elmer та FEATools, активна розробка Ansys та COMSOL ведеться вже кілька десятиліть. Наприклад, розробка COMSOL почалася в середині 80-х років минулого сторіччя, а розробка Ansys взагалі на початку 70-х років. Також, слід розуміти, що продукти з відкритим кодом розробляються ентузіастами, тож, що як правило, там немає чіткої місії та графіків розробки. Ansys та COMSOL в свою чергу розроблюється професійними інженерами з великим досвідом роботи. А так як це комерційні продукти, постійно обновлюються новими можливостями.

Ansys Workbench і Comsol Multiphysics - потужні програми для комп'ютерного моделювання, які, однак, мають свої переваги і недоліки.

Переваги Ansys Workbench:

- Модульна структура: Ansys Workbench складається з набору модулів, які можна використовувати окремо або разом, щоб створити модель для конкретного завдання. Це робить Ansys Workbench більш доступним, ніж інші програми для комп'ютерного моделювання, які вимагають придбання всієї програми.
- Потужний графічний інтерфейс: Ansys Workbench має потужний графічний інтерфейс, який робить його простим у використанні навіть для користувачів без досвіду в комп'ютерному моделюванні.
- Широкий спектр вбудованих моделей і модулів: Ansys Workbench має широкий спектр вбудованих моделей і модулів, що дозволяє швидко і легко створювати моделі для різних задач.
- Простіше в освоєнні, ніж Comsol Multiphysics: Ansys Workbench має простіший графічний інтерфейс і модульну структуру, ніж

Comsol Multiphysics, що робить його більш зручним для початківців.

Недоліки Ansys Workbench:

- Може бути менш точним, ніж Comsol Multiphysics, для задач, які вимагають моделювання взаємодії різних фізичних процесів.
- Немає деяких функцій, які є в Comsol Multiphysics, наприклад, можливості створення власних моделей.

Переваги Comsol Multiphysics:

- Може бути більш точним, ніж Ansys Workbench, для задач, які вимагають моделювання взаємодії різних фізичних процесів.
- Має широкий спектр вбудованих моделей і модулів, включаючи деякі, які не є доступні в Ansys Workbench.
- Дозволяє створювати власні моделі.

Недоліки Comsol Multiphysics:

- Немає модульної структури, як в Ansys Workbench. Це означає, що користувачі повинні мати більш глибокі знання про комп'ютерне моделювання, щоб використовувати Comsol Multiphysics.
- Графічний інтерфейс менш зручний в освоєнні, ніж в Ansys Workbench.

Вибір програмного забезпечення для конкретного завдання залежить від вимог завдання. Якщо задача вимагає моделювання взаємодії різних фізичних процесів, то Comsol Multiphysics може бути кращим вибором. Якщо задача не вимагає такого моделювання, то Ansys Workbench може бути більш доступним і простим у використанні варіантом.

Щодо безплатних продуктів, FEATool має досить простий та приємний інтерфейс, однак вимагає досить глибоких знань фізики, математики та самого

метода скінченних елементів. Наприклад, користувач сам має вирішити яке диференційне рівняння використати для розрахунку в поточному проєкті. Природно, що далеко не кожний користувач зможе одразу відповісти на подібне питання. Крім того, як виявилося, у спектрі видів розрахунків цього продукта немає ані модального ані гармонійного аналізу коливань. Тож у контексті теми роботи розглядати цей продукт більш детально немає сенсу.

Elmer дійсно досить потужний продукт в контексті функціонала, але має безнадійно застарілий та незрозумілий інтерфейс. Крім того, також потребує установки ще деяких додаткових продуктів для побудови моделей та сітки. Також досить серйозною проблемою є те, що проєкт вже давно не обновлювався, а в наш час це досить тривожний сигнал, котрий як правило означає, що проєкт не розвивається, та може остаточно зупинитися.

Зведемо інформацію у порівняльну таблицю. Деякі параметри оцінимо формальними балами від одиниці до п’яти, де один має значення дуже погано, а п’ять – дуже добре

Таблиця 2.1 – Порівняння продуктів

Продукт	Об’єм необхідних ресурсів ПК	Актуальність та підтримка	Зручність та дружність інтерфейсу	Наявність необхідного функціоналу	Вартість
Ansys Workbench	1	4	4	+	$30000
COMSOL Multiphysics	1	5	4	+	$8000
Elman	4	2	1	+	$0
FEATool	5	3	2	–	$0

З таблиці видно, що комерційні продукти, дивлячись на досить гарні показники функціональності та зручності, дуже дорогі. Також ці продукти потребують дуже великих ресурсів комп'ютера, тож для того, щоб з ними працювати, треба також мати досить потужне і дороге обладнання. Варто відзначити також високий рівень актуальності та підтримки цих продуктів. Вони постійно розвиваються, та мають широку базу знань із найрізноманітнішою інформацією та матеріалами для навчання.

Перевагами відкритих продуктів є їх безкоштовність та невисока потреба в ресурсах. За всіма іншими параметрами вони значно відстають. Вони не мають на стільки ж зручного та інтуїтивно зрозумілого інтерфейсу, та настільки ж розвинену інфраструктуру підтримки. Функціональність таких продуктів також значно менша ніж у комерційних продуктів.

Загальною рисою для всіх розглянутих продуктів є необхідність у попередньому навчанні роботи з ними. Це пов'язано з великою кількістю параметрів, які можуть впливати на розрахунок, або зі складністю самого інтерфейсу. Так чи інакше, маючи невелику задачу у вигляді розрахунку критичних частот, треба витратити чимало часу на побудову проєкту.

Тож нагальність поточної роботи полягає в розробці невеликого та швидкого застосунку для вирішення однієї конкретної задачі. Застосунок має бути зручним для використовування та не перевантаженим великою кількістю параметрів, щоб для роботи користувачу було достатньо поверхневих знань інженерії. Він не має бути універсальним вирішувачем для багатьох фізичних явищ, а має вирішувати лише одну задачу, а саме знаходження критичних частот завданої моделі ротора.

3 Методологія дослідження

3.1 Загальні положення методу скінченних елементів

Основною ідеєю методу є розбиття об’єкту дослідження на скінченну кількість елементів, які поєднуються вузлами. Для кожного елементу обирається апроксимуюча функція, коефіцієнти якої знаходяться із умов рівності у вузлах за допомогою систем алгебраїчних рівнянь.

Метод скінченних елементів можна розкласти на наступні етапи:

1. Поділ об’єкта задачі на скінченні елементи
2. Розробка рівнянь для елементів
3. Розробка рівняння для всієї системи
4. Накладання граничних умов
5. Рішення рівнянь
6. Знаходження необхідних значень

3.2 Прямий підхід метода скінченних елементів

3.2.1 Постановка задачі

Найпростішим підходом методу скінченних елементів є прямий підхід [1], [2]. Він прийнятний тільки у випадку, коли система може бути описана кусково-лінійною функцією. Таким чином, відпадає необхідність складних процедур з інтегрування та диференціювання цієї функції, і задача зводиться до досить простих дій з лінійними залежностями.

Розглянемо одиничний елемент ферми, до якого можуть бути застосовані тільки осьові навантаження. Відповідно, внутрішні зусилля також можуть виникати тільки впродовж осі.

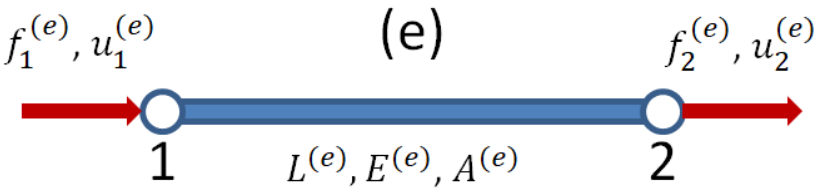


Рисунок 3.1 – Елемент ферми

У одномірного одиничного елемента є два кінця, які будемо називати вузлами. З рис.3.1 видно, що у вузлах на елемент діють деякі сили f, які визивають відповідні переміщення u. Нумерація параметрів вузла ведеться за схемою на рис. 3.2

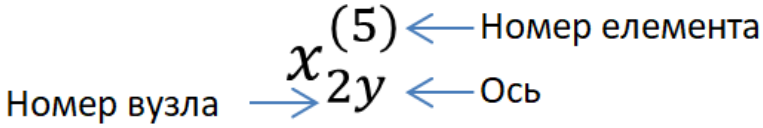


Рисунок 3.2 – Нумерація параметрів вузла

Назву осі можна опустити, коли вона (як у випадку з елементом ферми) очевидна.

Елемент має свій набір даних які характеризують властивості, пов’язані із геометрією та/або матеріалом об’єкту. Наприклад, в залежності від задачі це можуть бути такі властивості елемента, як маса, момент інерції, щільність, теплопровідність, електричний опір, тощо. Крім того, кожний елемент містить номер та дані про вузли, які йому належать. У нашому випадку елемент має умовний номер (e), та містить наступні дані: довжина ($L^{(e)}$), модуль Юнга ($E^{(e)}$), та площа перетину ($A^{(e)}$).

2.2.2 Поділ на елементи

Згідно зі списком етапів методу (див.3.1), першим кроком має бути поділ системи на скінченні елементи. Але в нашому випадку система не потребує додаткового поділу, так як він уже являє собою одиничний елемент. Звісно, можна вирішити задачу, поділивши цей елемент на безліч частин, і рішення буде вірним. Але в цьому немає жодного сенсу.

2.2.3 Рівняння для елементів

Наступним кроком треба скласти рівняння для елемента. Так як на елемент діють сили, які будемо вважати нерівними, в елементі виникають внутрішні зусилля. Якщо умовно розрізати елемент навпіл (рис.3), можна скласти рівняння рівноваги кожної частини елемента.

$$f_2^{(e)} = p^{(e)} \tag{3.1}$$

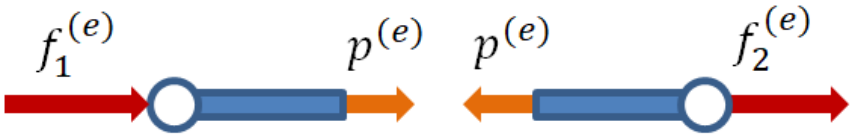


Рисунок 3.3 – До рівняння рівноваги елемента

Із визначення напруження [1], [8]

$$\sigma^{(e)} = \frac{p^{(e)}}{A^{(e)}} => \ p^{(e)} = \sigma^{(e)} A^{(e)} \tag{3.2}$$

Виходячи з закона Гука

$$\sigma^{(e)} = E^{(e)} \varepsilon^{(e)} \tag{3.3}$$

Де $\varepsilon^{(e)}$ відносна деформація

$$\varepsilon^{(e)} = \frac{\delta^{(e)}}{L^{(e)}} \tag{3.4}$$

Де $\delta^{(e)}$ видовження елемента під дією сил

$$\delta^{(e)} = u_2^{(e)} - u_1^{(e)} \tag{3.5}$$

Із приведених формул слідують наступні перетворення

$$f_2^{(e)} = p^{(e)} = \sigma^{(e)} A^{(e)} = E^{(e)} \varepsilon^{(e)} A^{(e)} = E^{(e)} A^{(e)} \frac{\delta^{(e)}}{L^{(e)}}$$

$$f_2^{(e)} = \frac{E^{(e)} A^{(e)}}{L^{(e)}} (u_2^{(e)} - u_1^{(e)}) \tag{3.6}$$

Маючи на увазі, що вираз перед скобкою це жорсткість при розтягуванні

$$\frac{EA}{L} = k \tag{3.7}$$

маємо рівняння для другого вузла

$$f_2^{(e)} = p^{(e)} = k^{(e)}(u_2^{(e)} - u_1^{(e)}) \tag{3.8}$$

Рівняння рівноваги всього елемента має вигляд

$$f_1^{(e)} + f_2^{(e)} = 0 => \ f_1^{(e)} = -f_2^{(e)} \tag{3.9}$$

Тоді маємо рівняння для першого вузла

$$f_1^{(e)} = -k^{(e)}\left(u_2^{(e)} - u_1^{(e)}\right) = k^{(e)}\left(u_1^{(e)} - u_2^{(e)}\right) \tag{3.10}$$

2.2.4 Рівняння для системи

Наступним кроком побудуємо рівняння для всієї системи. Для цього перепишемо рівняння для елементів в матричну форму

$$\begin{bmatrix} f_1^{(e)} \\ f_2^{(e)} \end{bmatrix} = \begin{bmatrix} u_1^{(e)}k^{(e)} - u_2^{(e)}k^{(e)} \\ u_2^{(e)}k^{(e)} - u_1^{(e)}k^{(e)} \end{bmatrix} = \begin{bmatrix} k^{(e)}(u_1 - u_2) \\ k^{(e)}(u_2 - u_1) \end{bmatrix} = \begin{bmatrix} k^{(e)} & -k^{(e)} \\ -k^{(e)} & k^{(e)} \end{bmatrix} \begin{bmatrix} u_1 \\ u_1 \end{bmatrix}$$

Остаточно маємо

$$\begin{bmatrix} f_1^{(e)} \\ f_2^{(e)} \end{bmatrix} = \begin{bmatrix} k^{(e)} & -k^{(e)} \\ -k^{(e)} & k^{(e)} \end{bmatrix} \begin{bmatrix} u_1 \\ u_1 \end{bmatrix} \tag{3.11}$$

Матрицю з коефіцієнтами жорсткості можна переписати у вигляді

$$K^{(e)} = \frac{E^{(e)}A^{(e)}}{L^{(e)}} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \tag{3.12}$$

Тоді рівняння для системи в скороченому вигляді буде

$$\boldsymbol{F} = \boldsymbol{K} \cdot \boldsymbol{U} \tag{3.13}$$

Невідомим членом тут є матриця $\boldsymbol{U}$. Рішенням цього рівняння буде

$$\boldsymbol{U} = \boldsymbol{K}^{-1}\boldsymbol{F} \tag{3.14}$$

Знаходження інших значень

Останнім кроком є знаходження інших значень. Знаючи переміщення, можна знайти видовження (3.5), відносну деформацію (3.4), та напруження (3.3).

2.2.5 Компоновка матриць

У попередньому прикладі було розглянуто одиничний елемент, на якого не було накладено жодних граничних умов. В реальності такого не буває. Для розбору накладення граничних умов на систему розглянемо стрижень змінного перетину, жорстко затиснену з одного боку, та на який діють відомі осьові сили. При цьому, також відомі розміри ділянок стрижнів та їх матеріал (рис. 3.4).

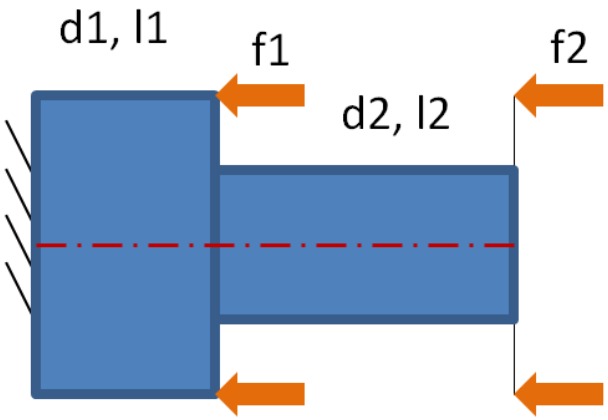


Рисунок 3.4 – Схема стрижня

По аналогії з 3.2.1-3.2.3 ділимо систему на елементи (рис. 3.4) та формуємо рівняння для елементів.

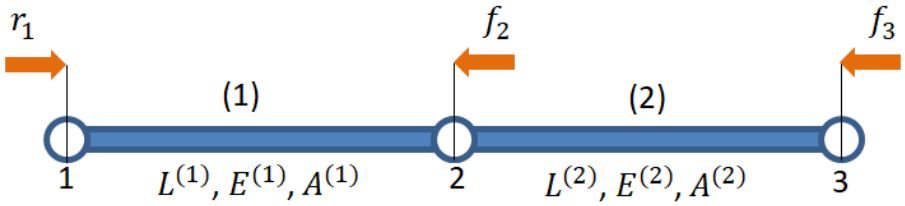


Рисунок 3.4.а – Поділ системи на елементи

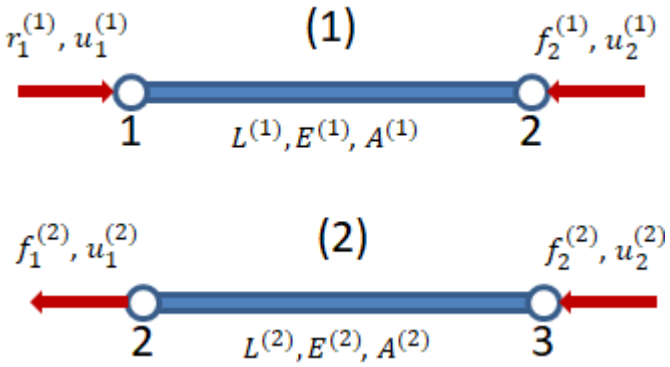


Рисунок 3.4.б – Схеми навантаження окремих елементів

Отримаємо наступні рівняння

$$\begin{bmatrix} r_1^{(1)} \\ f_2^{(1)} \end{bmatrix} = \begin{bmatrix} k^{(1)} & -k^{(1)} \\ -k^{(1)} & k^{(1)} \end{bmatrix} \begin{bmatrix} u_1^{(1)} \\ u_2^{(1)} \end{bmatrix}$$

$$\begin{bmatrix} f_1^{(2)} \\ f_2^{(2)} \end{bmatrix} = \begin{bmatrix} k^{(2)} & -k^{(2)} \\ -k^{(2)} & k^{(2)} \end{bmatrix} \begin{bmatrix} u_1^{(2)} \\ u_2^{(2)} \end{bmatrix}$$

Із рисунка 3.4 також очевидні наступні рівності

$$f_2^{(1)} = f_1^{(2)}$$

$$u_2^{(1)} = u_1^{(2)}$$

Тож, рівняння можна глобалізувати наступним чином

$$\begin{bmatrix} r_1 \\ f_2 \end{bmatrix} = \begin{bmatrix} k^{(1)} & -k^{(1)} \\ -k^{(1)} & k^{(1)} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$\begin{bmatrix} f_2 \\ f_3 \end{bmatrix} = \begin{bmatrix} k^{(2)} & -k^{(2)} \\ -k^{(2)} & k^{(2)} \end{bmatrix} \begin{bmatrix} u_2 \\ u_3 \end{bmatrix}$$

Для формування загального рівняння для системи необхідно скомпонувати ці рівняння. Матриці компонуються за номерами вузлів наступним чином:

1. В матриці жорсткості елемента умовно заміняємо натуральні індекси на номери вузлів
2. Створюємо квадратну одиничну матрицю K розмірністю N, де N – кількість вузлів в системі.

3. Вставляємо матриці жорсткості окремих елементів так, щоб елементи цієї матриці зайняли в матриці K елементи з індексами відповідно номерам вузлів елемента. При цьому, якщо в один самий елемент K потрапляє декілька значень, вони складаються.
4. В вектори-стовпчики сил та переміщень розміром N проставляються відповідно сили та переміщення згідно з номерами вузлів (рис. 3.5)

$$\begin{matrix} & 1 & 2 \\ 1 & k^{(1)} & -k^{(1)} \\ 2 & -k^{(1)} & k^{(1)} \end{matrix} \quad \begin{matrix} & 2 & 3 \\ 2 & k^{(2)} & -k^{(2)} \\ 3 & -k^{(2)} & k^{(2)} \end{matrix}$$

$$\Downarrow$$

$$\begin{matrix} & 1 & 2 & 3 \\ 1 & k^{(1)} & -k^{(1)} & 0 \\ 2 & -k^{(1)} & k^{(1)} + k^{(2)} & -k^{(2)} \\ 3 & 0 & -k^{(2)} & k^{(2)} \end{matrix}$$

Рисунок 3.5.а – Компоновка матриці жорсткості

$$\begin{matrix} 1 \\ 2 \end{matrix}\begin{bmatrix} r_1 \\ f_2 \end{bmatrix} \quad \begin{matrix} 2 \\ 3 \end{matrix}\begin{bmatrix} f_2 \\ f_3 \end{bmatrix} \Rightarrow \begin{bmatrix} r_1 \\ f_2 \\ f_3 \end{bmatrix}\begin{matrix} 1 \\ 2 \\ 3 \end{matrix}$$

$$\begin{matrix} 1 \\ 2 \end{matrix}\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \quad \begin{matrix} 2 \\ 3 \end{matrix}\begin{bmatrix} u_2 \\ u_3 \end{bmatrix} \Rightarrow \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}\begin{matrix} 1 \\ 2 \\ 3 \end{matrix}$$

Рисунок 3.5.б – Компоновка стовпців сил та переміщень

Остаточно отримуємо рівняння

$$\begin{bmatrix} r_1 \\ f_2 \\ f_3 \end{bmatrix} = \begin{bmatrix} k^{(1)} & -k^{(1)} & 0 \\ -k^{(1)} & k^{(1)} + k^{(2)} & -k^{(2)} \\ 0 & -k^{(2)} & k^{(2)} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}$$

Отримане рівняння і є загальним рівнянням для всієї системи. Слід зазначити, що величини r_1, u_1, u_2, u_3 є невідомими.

Є помилкою вважати, що матриці елементів завжди компонуються блоками. У розглядуваному прикладі так сталося завдяки оптимальній нумерації вузлів, яка може вестися в довільному порядку. Якщо, наприклад, поміняти місцями номери вузлів 2 та 3 (буде 1-3-2) отримаємо наступне рівняння

$$\begin{bmatrix} r_1 \\ f_2 \\ f_3 \end{bmatrix} = \begin{bmatrix} k^{(1)} & 0 & -k^{(1)} \\ 0 & k^{(2)} & -k^{(2)} \\ -k^{(1)} & -k^{(2)} & k^{(1)} + k^{(2)} \end{bmatrix} \begin{bmatrix} u_1 \\ u_3 \\ u_2 \end{bmatrix}$$

2.2.6 Накладання граничних умов

У векторах-стовпцях повинно виконуватись правило – у одному рядку не може бути двох невідомих величин. Однак, як видно у нашому випадку, у першому рядку маємо тільки невідомі значення.

$$\begin{bmatrix} r_1 \\ f_2 \\ f_3 \end{bmatrix} \begin{bmatrix} u_1 \\ u_3 \\ u_2 \end{bmatrix}$$

Але з умови задачі нам відомо, що вузол 1 зафіксовано, тож переміщення u_1 взагалі-то відоме і дорівнює нулю. Тому маємо

$$\begin{bmatrix} r_1 \\ f_2 \\ f_3 \end{bmatrix} \begin{bmatrix} 0 \\ u_3 \\ u_2 \end{bmatrix}$$

А рівняння прийме вигляд

$$\begin{bmatrix} r_1 \\ f_2 \\ f_3 \end{bmatrix} = \begin{bmatrix} k^{(1)} & 0 & -k^{(1)} \\ 0 & k^{(2)} & -k^{(2)} \\ -k^{(1)} & -k^{(2)} & k^{(1)} + k^{(2)} \end{bmatrix} \begin{bmatrix} 0 \\ u_3 \\ u_2 \end{bmatrix}$$

Це обмеження необхідно також внести у загальне рівняння. Існує кілька способів це зробити [1], [2], [14]:

- метод обнулення: значення рядків та стовпчиків матриці жорсткості, відповідні до закріплених вузлів міняють на нулі, а діагональний елемент на одиницю

$$\begin{bmatrix} r_1 \\ f_2 \\ f_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & k^{(1)} + k^{(2)} & -k^{(2)} \\ 0 & -k^{(2)} & k^{(2)} \end{bmatrix} \begin{bmatrix} 0 \\ u_2 \\ u_3 \end{bmatrix}$$

- метод штрафів: рядок відповідний до закріпленого вузла помножується на деяке дуже велике число. Слід звернути увагу, що відоме значення

переміщення теж повинно бути помножене на це число, так як воно не обов'язково завжди дорівнює нулю в закріплені. Значення реакції на β немає сенсу, так як воно невідоме

$$\begin{bmatrix} r_1 \\ f_2 \\ f_3 \end{bmatrix} = \begin{bmatrix} \beta \cdot k^{(1)} & -k^{(1)} & 0 \\ -k^{(1)} & k^{(1)} + k^{(2)} & -k^{(2)} \\ 0 & -k^{(2)} & k^{(2)} \end{bmatrix} \begin{bmatrix} \beta \cdot 0 \\ u_2 \\ u_3 \end{bmatrix}$$

- метод блокових матриць: рівняння ділиться на матриці-блоки в залежності від того, чи відомі переміщення для рядків.

$$\begin{array}{cccc} \boldsymbol{K_E} & & \boldsymbol{K_{EF}} & \boldsymbol{U_E} \quad \boldsymbol{F_E} \\ \left[\begin{array}{c|cc} k^{(1)} & 0 & -k^{(1)} \\ \hline 0 & k^{(2)} & -k^{(2)} \\ -k^{(1)} & -k^{(2)} & k^{(1)} + k^{(2)} \end{array}\right] & & & \left[\begin{array}{c} 0 \\ u_3 \\ u_2 \end{array}\right] = \left[\begin{array}{c} r_1 \\ f_2 \\ f_3 \end{array}\right] \\ & & \boldsymbol{K_F} & \boldsymbol{U_F} \quad \boldsymbol{F_F} \end{array}$$

Рисунок 3.6 – Поділ рівняння на блоки

Тоді шляхом нескладних алгебраїчних перетворень, описаних в [1] отримаємо наступні рівняння в скороченому вигляді, які враховують граничні умови

$$U_F = K_F^{-1}(F_F - K_{EF}^T U_E)$$

$$F_E = K_E U_E + K_{EF} U_F$$

Слід підкреслити, що за рахунок різних в залежності від умов перетворень, даний метод є досить складним для програмування.

3.3 Метод скінченних елементів для коливальних систем

3.3.1 Деякі базові елементи теорії пружності

Розглядаючи деяке тіло, яке знаходиться під дією деяких сил, можна стверджувати, що таке тіло знаходиться у напружено-деформованому стані.

Напруження при цьому характеризується тензором напружень [6], [8]

$$S = \begin{bmatrix} \sigma_x & \tau_{xy} & \tau_{xz} \\ \tau_{yx} & \sigma_y & \tau_{yz} \\ \tau_{zx} & \tau_{zy} & \sigma_z \end{bmatrix} \tag{3.15}$$

де $\sigma_x, \sigma_y, \sigma_z$ – нормальні напруження, $\tau_{xy}, \tau_{xz}, \tau_{yz}, \tau_{yx}, \tau_{zx}, \tau_{zy}$ – дотичні напруження

Маючи на увазі симетричність матриці, тензор можна переписати у вигляді стовпчика

$$\sigma = \begin{Bmatrix} \sigma_x \\ \sigma_y \\ \sigma_z \\ \tau_{xy} \\ \tau_{yz} \\ \tau_{zx} \end{Bmatrix} \tag{3.16}$$

Деформації у будь-якій точці характеризується тензором деформацій

$$T = \begin{bmatrix} \varepsilon_x & \gamma_{xy} & \gamma_{xz} \\ \gamma_{yx} & \varepsilon_y & \gamma_{yz} \\ \gamma_{zx} & \gamma_{zy} & \varepsilon_z \end{bmatrix} \tag{3.17}$$

де $\varepsilon_x, \varepsilon_y, \varepsilon_z$ – лінійні деформації вздовж осей, $\gamma_{xz}, \gamma_{yz}, \gamma_{yx}, \gamma_{zx}, \gamma_{zy}$ – зсувні деформації

При цьому, кожна точка такого тіла отримує деяке переміщення, яке можна записати у вигляді

$$\vec{s} = u\vec{\imath} + v\vec{\jmath} + w\vec{k} \tag{3.18}$$

де u, v, k – проєкції переміщення на осі системи координат

Тож, можна ввести стовпець зі значеннями цих проєкцій

$$\boldsymbol{u} = \begin{Bmatrix} u \\ v \\ w \end{Bmatrix} \tag{3.19}$$

Тоді взаємозв’язок компонентів деформованого стану та вектора переміщень можна записати як рівняння Коші

$$\begin{cases} \varepsilon_x = \frac{\partial u}{\partial x} \\ \varepsilon_y = \frac{\partial v}{\partial y} \\ \varepsilon_z = \frac{\partial w}{\partial z} \end{cases} \qquad \begin{cases} \gamma_{xy} = \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \\ \gamma_{xy} = \frac{\partial v}{\partial z} + \frac{\partial w}{\partial y} \\ \gamma_{zx} = \frac{\partial u}{\partial z} + \frac{\partial w}{\partial x} \end{cases} \tag{3.20}$$

Рівняння можна записати у матричному вигляді

$$\varepsilon = Du \tag{3.21}$$

де D – спеціальна операторна матриця диференціювання

$$D = \begin{bmatrix} \frac{\partial}{\partial x} & 0 & 0 \\ 0 & \frac{\partial}{\partial x} & 0 \\ 0 & 0 & \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x} \end{bmatrix} \tag{3.22}$$

Для встановлення взаємозв’язку напружень та деформацій розглянемо узагальнений закон Гука

$$\begin{cases} \varepsilon_x = \frac{1}{E}\left(\sigma_x - \nu(\sigma_y + \sigma_z)\right) \\ \varepsilon_x = \frac{1}{E}\left(\sigma_y - \nu(\sigma_z + \sigma_x)\right) \\ \varepsilon_x = \frac{1}{E}\left(\sigma_z - \nu(\sigma_x + \sigma_y)\right) \end{cases} \qquad \begin{cases} \gamma_{xy} = \frac{\tau_{xy}}{G} \\ \gamma_{yz} = \frac{\tau_{yz}}{G} \\ \gamma_{zx} = \frac{\tau_{zx}}{G} \end{cases} \tag{3.22}$$

де E – модуль поздовжньої пружності

G – модуль зсуву

ν – коефіцієнт Пуассона

Також існує зв’язок

$$G = \frac{E}{2(1+\nu)} \tag{2.23}$$

Тож, закон Гука можна записати у матричній формі як

$$\varepsilon = M\sigma \tag{3.24}$$

де M – матриця коефіцієнтів Пуассона

$$M = \begin{bmatrix} 1 & -\nu & -\nu & 0 & 0 & 0 \\ -\nu & 1 & -\nu & 0 & 0 & 0 \\ -\nu & -\nu & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2(1+\nu) & 0 & 0 \\ 0 & 0 & 0 & 0 & 2(1+\nu) & 0 \\ 0 & 0 & 0 & 0 & 0 & 2(1+\nu) \end{bmatrix} \tag{3.25}$$

Зворотній закон Гука буде мати вигляд

$$\sigma = A\varepsilon \tag{3.26}$$

де A – матриця закона Гука

$$A = M^{-1} = \frac{E}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1-\nu & 0 & 0 & 0 & 0 & 0 \\ 0 & 1-\nu & 0 & 0 & 0 & 0 \\ 0 & 0 & 1-\nu & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1-\nu}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1-\nu}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1-\nu}{2} \end{bmatrix} \tag{3.27}$$

3.3.2 Рівняння руху для скінченного коливального елемента

Розглянемо скінченний елемент, який коливається [7]. При цьому, поле його переміщень можна виразити через значення його вузлових переміщень як

$$u = \Phi q \tag{3.27}$$

де Ф – матриця функцій форм коливань.

Так як ми не знаємо справжніх форм коливань, ми припускаємо, що їх можна описати деяким законом (наприклад поліноміальною або тригонометричною функцією), тож матриця функцій є нічим іншим, як матрицею інтерполяційних функцій. Тож ця матриця також називається інтерполяційною.

Із курсу будівельної механіки відомо, що переміщення пов'язані із деформаціями (див. 3.18), тож у матричному вигляді можна записати

$$\varepsilon = Du = D\Phi q = Bq \tag{3.28}$$

де D – спеціальна операторна матриця (див. 3.22)

B – матриця Коші

$$B = D\Phi \tag{3.29}$$

Перепишемо (3.26) з урахуванням (3.27)

$$\sigma = \mathrm{ABq} \tag{3.30}$$

Для знаходження квадратичної форма кінетичної енергії можна скористатися формулою

$$T = \frac{1}{2}\int_{\Omega} \dot{u}^T \rho \dot{u}\, d\Omega \tag{3.31}$$

де $\rho = \rho(x, y, z)$ – функція розподілу щільності матеріалу

Ω – область інтегрування, що відповідає скінченному елементу

Підставивши (3.27) в (2.31) отримаємо

$$T = \frac{1}{2}\dot{q}^T M \dot{q} \tag{3.32}$$

де M – матриця інерції скінченного елемента

$$M = \int_{\Omega} \Phi^T \rho \Phi\, d\Omega \tag{3.33}$$

Аналогічно з виведенням формули для кінетичної енергії, із квадратичної форми для потенційної енергії (3.34), використовуючи (3.28) та (3.29) отримаємо інтегральну формулу для потенціальної енергії (3.35)

$$\Pi = \frac{1}{2}\int_{\Omega} \varepsilon^T \sigma \, d\Omega \tag{3.34}$$

$$\Pi = \frac{1}{2} q^T C q \tag{3.35}$$

де C – матриця жорсткості елемента

$$C = \int_{\Omega} \mathrm{B}^T AB \, d\Omega \tag{3.36}$$

Для завдання руху скінченного коливального елемента за відсутності розсіювання енергії, скористаймося рівняння Лагранжа другого роду

$$\frac{\partial}{\partial t}\left(\frac{\partial T}{\partial \dot{q}}\right) - \frac{\partial T}{\partial q} + \frac{\partial \Pi}{\partial \dot{q}} = F \tag{3.37}$$

де F – вектор узагальнених сил

$$F = \int_{\Omega} \Phi^T p d\Omega \tag{3.26}$$

де $p = p(x, y, z)$ – функція розподілу навантаження, що діє на елемент

Підставивши (3.32) та (3.35) в (3.37), та виконавши деякі елементарні перетворення, отримаємо рівняння руху скінченного елемента в матричній формі

$$M\ddot{q} + Cq = F \tag{3.38}$$

3.3.4 Власні вектори та значення

Один з методів дослідження спирається на такі властивості матриці, як її власні вектори та значення. Тому в поточному розділі буде приведена коротка інформація щодо цих властивостей виходячи з [3], [4], [5].

Будь яку квадратну матрицю можна розглядати як оператор перетворення вектора розмірністю такою, як і матриця в деякому базисі. Наприклад, застосуємо матрицю A як оператор перетворення для двомірних векторів, які завдані нижче.

$$\mathrm{A} = \begin{bmatrix} 3 & 2 \\ 0 & 1 \end{bmatrix}, v_1 = (0,1), v_2 = (2,1), v_3 = (1,0)$$

Отриманий результат можна побачити на рис.8. Можна зазначити, що вектори v_1 та v_2 після перетворення змінили напрямок та довжину. Але вектор v_3 змінив тільки довжину, але залишився колінеарним самому собі. Це можна записати як

$$A \cdot \boldsymbol{v_3} = \begin{bmatrix} 3 & 2 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 3 \\ 0 \end{bmatrix} = 3 \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix} = 3 \cdot \boldsymbol{v_3}$$

Тобто, після перетворення отримуємо той же вектор, скалярно помножений на деяке число. В нашому разі, це число 3.

Такий вектор, називається власним для матриці A, а множник цього вектора називається власним значенням матриці.

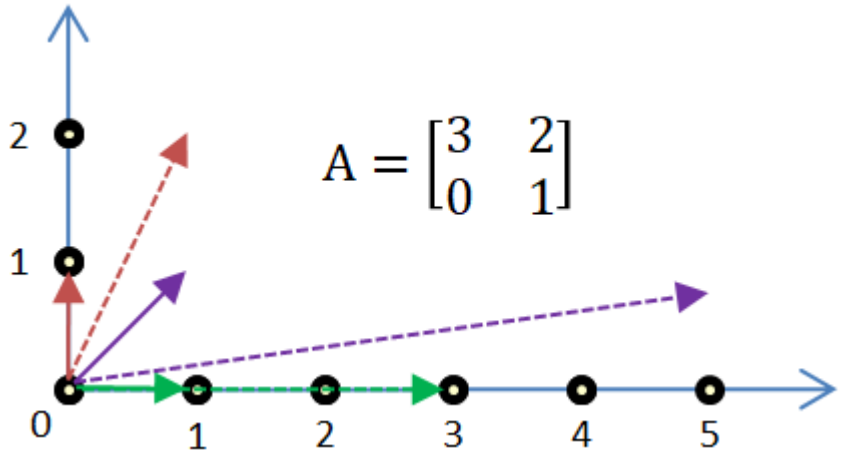


Рисунок 3.7 – Перетворення векторів за допомогою матриці

Тобто, у загальному випадку можна записати

$$A\boldsymbol{v} = \lambda \boldsymbol{v} \tag{3.39}$$

де $\boldsymbol{v}$ – власний вектор

λ – власне значення, відповідне до вектора

Формулу 2.х можна переписати у суто матричному вигляді як

$$A\boldsymbol{v} = \lambda I \boldsymbol{v} \tag{3.40}$$

де I – одинична матриця

Після переносів та групування отримаємо

$$(A - \lambda I)\boldsymbol{v} = 0 \tag{3.41}$$

або

$$\begin{bmatrix} a_{11}-\lambda & a_{12} & a_{1n} \\ a_{21} & a_{22}-\lambda & a_{2n} \\ a_{n1} & a_{n2} & a_{nn-\lambda} \end{bmatrix} \cdot \boldsymbol{v} = 0 \qquad (3.42)$$

Фізичний сенс власного вектора можна проілюструвати наступним прикладом. Припустимо деяке тіло, що довільно обертається у тримірному просторі. Таке обертання можна завдати матрицею перетворення. В такому контексті власний вектор завдасть вісь обертання цього тіла.

3.3.4 Дискретизація коливального скінченного елемента

Розглянемо кінцевий елемент, що являє собою прямолінійний брус із наступними відомими характеристиками: площею поперечного перерізу A, довжиною L, щільністю матеріала ρ, моментом інерції перерізу I, та модулем пружності E. Елемент обмежений вузлами 1 та 2, які під дією зовнішніх сил можуть мати деякі переміщення (u), та повертатися відносно осі x на деякі кути (Θ) [7]. Такий елемент зображено на рис.7

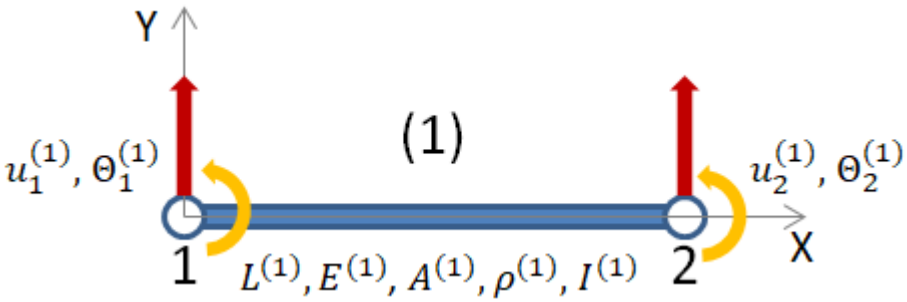


Рисунок 3.8 – Кінцевий елемент балки

Як було зазначено у коментарях до (2.15), матриця форм коливань, яка у даному випадку є матрицею форм прогину є матрицею інтерполяційних формул, тип яких ми обираємо самі. Для даного випадку оберемо в якості інтерполяційної формули кубічний поліном. Тож, значення переміщення u будуть залежати від координати x за деяким законом (2.28)

$$u(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 \qquad (3.43)$$

Маючи на увазі, що кут повороту є похідна від переміщення, можна записати залежність кутів повороту від координати

$$\Theta(x) = \frac{du(x)}{dx} = a_1 + 2a_2 x + 3a_3 x^2 \qquad (3.44)$$

Граничні умови приймуть вигляд

$$\begin{cases} v(0) = a_0 = u_1 \\ \Theta(0) = a_1 = \Theta_1 \\ v(L) = a_0 + a_1 L + a_2 L^2 + a_3 L^3 = u_2 \\ \Theta(L) = a_1 + a_2 L + a_3 L^2 = u_2 \end{cases} \quad (3.45)$$

Розв'язавши цю систему рівнянь отримаємо

$$\begin{cases} a_0 = u_1 \\ a_1 = \Theta_1 \\ a_2 = -\frac{3}{l^2}u_1 - \frac{2}{l}\Theta_1 + \frac{3}{l^2}u_2 - \frac{1}{l}\Theta_2 \\ a_3 = \frac{2}{l^3}u_1 - \frac{1}{l^2}\Theta_1 + \frac{2}{l^3}u_2 - \frac{1}{l^2}\Theta_2 \end{cases} \quad (3.46)$$

Так як коефіцієнти a_i є коефіцієнтами функції форми елемента, згідно (2.28) можна скласти рівняння

$$u(x) = \left(1 - 3\frac{x^2}{l^2} + 2\frac{x^3}{l^3}\right)u_1 + l\left(\frac{x}{l} - 2\frac{x^2}{l^2} + \frac{x^3}{l^3}\right)\Theta_1 + \left(3\frac{x^2}{l^2} - 2\frac{x^3}{l^3}\right)u_2 +$$

$$l\left(-\frac{x^2}{l^2} + \frac{x^3}{l^3}\right)\Theta_2 = \Phi_1 u_1 + \Phi_1\Theta_1 + \Phi_1 u_2 + \Phi_1\Theta_2 \quad (3.47)$$

Або

$$[\Phi_1 \quad \Phi_2 \quad \Phi_3 \quad \Phi_4]\begin{bmatrix} u_1 \\ \Theta_1 \\ u_2 \\ \Theta_2 \end{bmatrix} = \Phi q \quad (3.48)$$

Де

$$\Phi = \begin{cases} \Phi_1 = 1 - 3\frac{x^2}{l^2} + 2\frac{x^3}{l^3} \\ \Phi_2 = l\left(\frac{x}{l} - 2\frac{x^2}{l^2} + \frac{x^3}{l^3}\right) \\ \Phi_3 = \left(3\frac{x^2}{l^2} - 2\frac{x^3}{l^3}\right) \\ \Phi_4 = l\left(-\frac{x^2}{l^2} + \frac{x^3}{l^3}\right) \end{cases} \quad (3.49)$$

Тоді за формулою (2.21) маємо вираз для матриці інерції

$$M = \int_0^l \Phi^T \rho \Phi A dx = \begin{bmatrix} \frac{13}{35}m & \frac{11}{210}ml & \frac{9}{70}m & -\frac{13}{420}ml \\ \frac{11}{210}ml & \frac{1}{105}ml^2 & \frac{13}{420}ml & -\frac{1}{140}ml^2 \\ \frac{9}{70}ml^2 & \frac{13}{420}ml & \frac{13}{35}m & -\frac{11}{210}ml \\ -\frac{13}{420}ml & -\frac{1}{140}ml^2 & -\frac{11}{210}ml & \frac{1}{105}ml^2 \end{bmatrix} \quad (3.50)$$

Маючи на увазі, що для випадку поперечного прогину значення потенційної енергії визначається формулою

$$\Pi = \frac{1}{2}\int_0^l EI\left(\frac{d^2u}{dx^2}\right)dx \tag{3.51}$$

А вираз для знаходження значень матриці жорсткості

$$C_{i,j} = \frac{\partial^2 \Pi}{\partial q_i \partial q_j} \tag{3.52}$$

Із формул отримаємо

$$C = \begin{bmatrix} \frac{12EI}{l^3} & \frac{6EI}{l^2} & -\frac{12EI}{l^3} & \frac{6EI}{l^2} \\ \frac{6EI}{l^2} & \frac{4EI}{l} & -\frac{6EI}{l^2} & \frac{2EI}{l} \\ -\frac{12EI}{l^3} & -\frac{6EI}{l^2} & \frac{12EI}{l^3} & -\frac{6EI}{l^2} \\ \frac{6EI}{l^2} & \frac{2EI}{l} & -\frac{6EI}{l^2} & \frac{4EI}{l} \end{bmatrix} \tag{3.53}$$

3.4 Методи розрахунку критичних частот

В поточному розділі буде розглянуто методи, якими у розроблюваному додатку будуть розраховуватися значення критичних частот. Для цього будемо використовувати два методи:

- метод простих ітерацій
- метод власних значень

В ході роботи буде реалізовано обидва методи, а результати розрахунків будуть порівняні з відповідними висновками.

3.4.1 Метод простих ітерацій

Розглянемо метод знаходження критичних частот шляхом послідовного перебору аргументів у деякому діапазоні. За основу методу узятий метод простих ітерацій для вирішення нелінійних алгебраїчних рівнянь [10], [11]. Суть метода полягає в послідовній підстановці аргументів в рівняння через деякий невеликий крок. Корні рівняння знаходяться на перетині графіка функції рівняння з віссю X (див. рис. 3.9). Тож, якщо для двох сусідніх аргументів функція приймає різні за знаком значення, рішення знаходиться в поточному діапазоні. Якщо вибрати крок сумірним із похибкою обчислень, то в якості рішення можна взяти один із поточних аргументів.

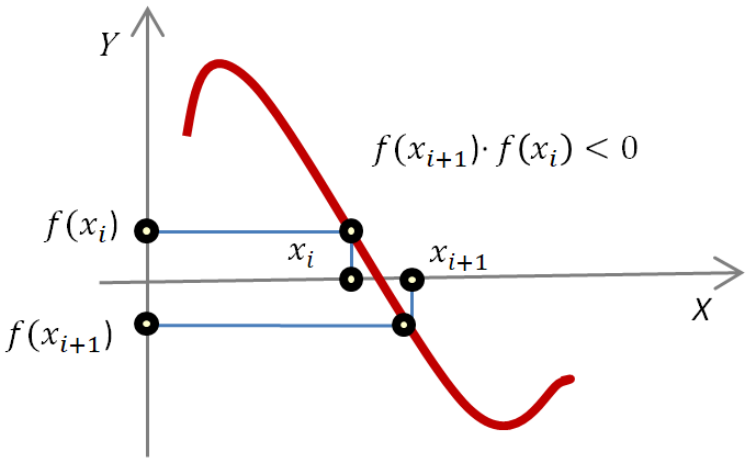


Рисунок 3.9 – Метод простих ітерацій

Розглянемо рівняння 3.38, беручи до уваги, що функція $y(t)$ є функцією гармонічних коливань від часу [6], [7], тобто

$$y(t) = A \cdot \sin(\omega t) \tag{3.54}$$

де A – амплітуда коливань

ω – частота коливань

Підставивши 3.54 в 3.38, та розкривши похідну маємо

$$(-\boldsymbol{M}\omega^2 + \boldsymbol{C}) \cdot A \cdot \sin(\omega t) = (\boldsymbol{C} - \boldsymbol{M}\omega^2) \cdot A \cdot \sin(\omega t) = 0 \tag{3.55}$$

Вираз 3.55 буде дорівнювати нулю, коли один з множників буде дорівнювати нулю. У даному випадку нас не цікавлять тривіальні рішення, коли $\mathrm{A} = 0$, або $\sin(\omega \mathrm{t}) = 0$.

Тож, рівняння 3.55 може бути спрощено до виду 3.56

$$\boldsymbol{C} - \boldsymbol{M}\omega^2 = \boldsymbol{0} \tag{3.56}$$

Підкреслимо, що 3.56 є матричним рівнянням, тож щоб уникнути плутанини перепишемо його у вигляді

$$det(\boldsymbol{C} - \boldsymbol{M}\omega^2) = 0 \tag{3.57}$$

$$\left|\boldsymbol{C} - \boldsymbol{M}\omega_i^2\right| \cdot \left|\boldsymbol{C} - \boldsymbol{M}\omega_{i+1}^2\right| < 0$$

Щоб вирішити це рівняння методом простої ітерації, треба завдати деякий інтервал частот. Це може бути, наприклад, інтервал від нуля до максимальної робочої частоти ω_{max}. Також треба завдати крок ітерації, скажімо в 1 Гц. І далі ітеративно додавати значення кроку до поточного значення частоти, та перевіряти, чи є результат множення поточного та попереднього значення детермінантів від'ємним. Коли отримуємо такий результат, то поточне значення і буде шуканим значенням критичної частоти в ступені двійки.

3.4.2 Метод власних значень

Розглянемо рівняння 3.56, та ізолюємо аргумент функції шляхом помноження рівняння на обернену матрицю M^{-1}. Для алгебраїчного рівняння це фактично означало б ділення рівняння на M [5], [14]. Для компактності припустимо, що добуток $M^{-1}C$ дорівнює деякій матриці D

$$\boldsymbol{M}^{-1} \cdot \boldsymbol{C} - \omega^2 = \boldsymbol{D} - \omega^2 = \boldsymbol{0} \tag{3.58}$$

Так як рівняння є матричним, вираз $\boldsymbol{D} - \omega^2$, як різниця матриці та числа не має сенсу, тож від'ємник теж має бути матрицею. Для цього представимо від'ємник у виді добутку $\omega^2 E$, де E – це одинична діагональна матриця відповідного розміру. Тоді вираз 3.58 набуде виду

$$\boldsymbol{D} - \omega^2 \boldsymbol{E} = \boldsymbol{0} \tag{3.59}$$

Підставимо отриманий вираз у 3.55 та припустимо замість виразу $A \cdot \sin(\omega t)$ деякий вектор A^*. З такими змінами рівняння 3.55 прийме вигляд

$$(\boldsymbol{D} - \omega^2 \boldsymbol{E}) \cdot A^* = 0 \tag{3.60}$$

При цьому маємо визначення власних значень для матриці у вигляді 3.41. Тут спостерігається явна подібність, тож можна припустити, що власні значення матриці D дорівнюють другому ступеню значень критичних частот, а власні вектори матриці D завдають значення, з яких можна знайти амплітуди коливань. Тобто, можна записати

$$\{\omega\} = \sqrt{EigVals(\boldsymbol{D})} \tag{3.31}$$
$$\{A\} = F(EigVecs(\boldsymbol{D}))$$

3.5 Метод розрахунку форм відносних деформацій

Маючи значення критичних частот, а також маючи на увазі, що і значення у ступені двійки є власними значеннями для матриці D, скориставшись 3.59 та 3.42 можна знайти форми відносних деформацій. Для кожної з частот можна записати рівняння, де x_i значення координати форми відносної деформації

$$\begin{bmatrix} (d_{1,1} - \omega_i^2)x_1 & d_{1,2}x_2 & d_{1,n}x_n \\ d_{2,1}x_1 & (d_{2,2} - \omega_i^2)x_2 & d_{2,n}x_n \\ d_{n,1}x_1 & d_{n,2}x_2 & (d_{n,n} - \omega_i^2)x_n \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Вочевидь, за рахунок нульового стовпчика результатів рішення такої системи будуть тривіальними. Але маючи на увазі, що нам не так важливі конкретні значення x_i, як співвідношення між ними, можна припустити одне із значень відомим, та яке дорівнює одиниці [4], [5]. Нехай таким значенням буде

x_n. Зробивши так ми позбавляємося від останнього рядка рівняння, та отримуємо останній стовпчик без змінних, тож їх можна перенести за знак рівності. Тепер стовпчик результатів ненульовий, і ми можемо отримати нетривіальні рішення

$$\begin{bmatrix} (d_{1,1} - \omega_i^2)x_1 & d_{1,2}x_2 & d_{1,n-1}x_{n-1} \\ d_{2,1}x_1 & (d_{2,2} - \omega_i^2)x_2 & d_{2,n-1}x_{n-1} \\ d_{n-1,1}x_1 & d_{n-1,2}x_2 & (d_{n-1,n-1} - \omega_i^2)x_{n-1} \end{bmatrix} = \begin{bmatrix} -d_{1,n} \\ -d_{2,n} \\ -d_{n-1,n} \end{bmatrix}$$

Вирішивши таку систему, отримаємо матрицю розміром $[n-1, n-1]$, стовпчики якої будуть містити значення зміщень та кутів прогину для поточної координати. При цьому, прийнявши перший індекс за одиницю, значення зміщень будуть займати непарні рядки, а значення кутів прогину парні. Для віднульового індексу навпаки.

Так як для подальших розрахунків нам не знадобляться значення кутів прогину, відповідні рядки слід виключити.

В результаті отримаємо квадратну матрицю, де рядки будуть містити значення відносних деформації для певної координати.

Останньою дією над такою матрицею буде нормалізація. Сенс процедури це привести значення до такого вигляду, щоб у кожному рядку була хоча б одна одиниця.

Щоб виконати нормалізацію, треба знайти максимальне по модулю значення у кожному стовбці, та поділити стовпець на це значення.

4 Розробка додатку

4.1 Вимоги до додатку

Основні задачі програми це опрацювання вхідного набору даних, що описують систему деякого ротора, розрахунок критичних частот для завданої системи, та вивід або збереження результатів роботи.

Програма має бути розроблена для сімейства операційних систем Windows від Windows 7 до Windows 10, повинна мати сучасний та зручний віконний інтерфейс, високу відмовостійкійсть, та забезпечувати коректні результати розрахунків.

Програма повинна мати простий та дружній інтерфейс, який би дозволяв наступні операції:

- створення нового проєкту
- збереження поточного проєкту
- відкривання збереженого проєкту
- побудова моделі ротора та її відображення
- розміщення позиційних елементів на моделі, таких як опори, додані маси та жорсткості
- для кожного елемента схеми повинна бути можливість назначати та змінювати параметри цього елементу. Наприклад значення довжини та діаметра для елемента ротора, значення маси та моменту інерції для доданих мас тощо
- можливість обирати деякі глобальні параметри розрахунку, такі як кількість значень в результуючому спектрі частот, максимальна частота, до якої треба шукати результати, тощо
- виконання розрахунку критичних частот для завданої схеми
- представлення результатів розрахунку у вигляді списку критичних частот та діаграм відносних деформацій ротора при цих частотах
- можливість зберегти результати розрахунку у вигляді веб сторінки
- можливість завантажувати вихідні дані із певним чином відформатованого файлу MS Excel

У разі виникнення помилок у розрахунках, або вихідних даних, програма повинна видавати зрозумілі повідомлення, які б надавали користувачу інформацію, що він може зі свого боку зробити, щоб виправити помилку.

Для діагностики роботи програми всі етапи мають бути покриті логами з можливістю вмикання та вимикання логування.

4.2 Синтез структури додатку

Виходячи із досвіду та практик програмування, програму краще розділити на модулі, які б відповідали за якісь конкретні етапи роботи програми. Такий підхід спрощує пошук помилок та тестування програми, а також дозволяє досить гнучко масштабувати програму. Для цього необхідно проаналізувати які етапи проходять вхідні дані та як перетворюються. Для цих цілей побудуємо структурну схему додатку. Маючи таку схему можна попередньо визначити які моделі та функціональні модулі повинні бути розроблені, а також як їх краще скомпонувати.

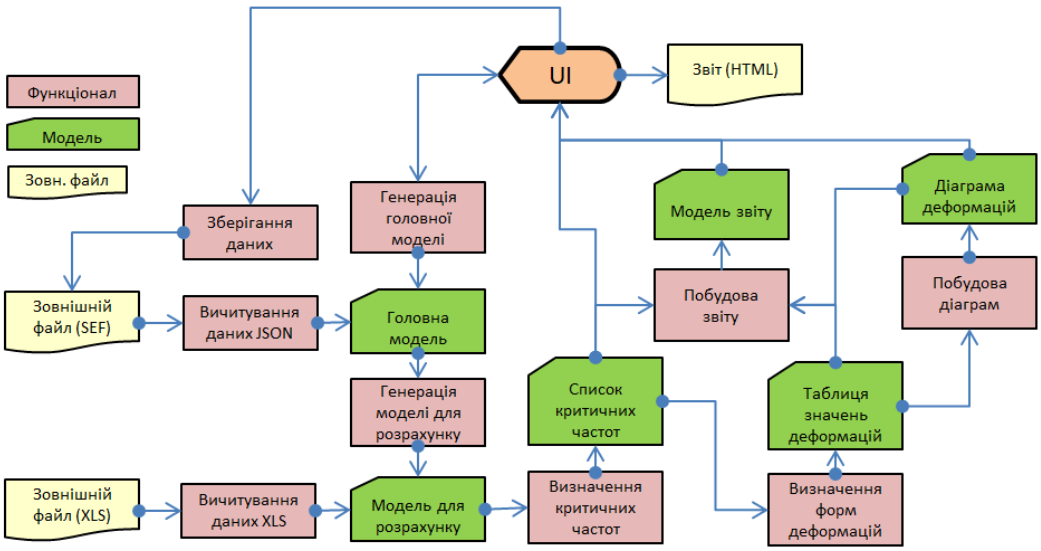


Рисунок 4.1 – Структурна схема додатку

Маючи структурну схему, можемо попередньо скласти перелік необхідних функціональних модулів

- Core. Відповідальністю модуля є генерацію головної моделі та її взаємодія з інтерфейсом. Тут процеси генерації та взаємодії моделі з інтерфейсом є неперервними та неподільними в силу особливостей фреймворку WPF.

- JsonHelper. Відповідальністю модуля є серіалізація та десеріалізація даних в формат JSON, в якому будуть зберігатися файли проєкту
- IOHelper. Відповідальністю модуля є елементарна робота з файловою системою, така як вичитування даних з файлів, створення нових файлів, збереження даних, тощо.
- XlsHelper. Відповідальністю модуля є вичитування даних з таблиць Excel
- WorkModelGenerator. Відповідальністю модуля є генерація робочої моделі для розрахунку із головної моделі.
- Solver. Відповідальністю модуля є розрахунки критичних частот та відповідних форм деформацій.
- DiagramBuilder. Відповідальністю модуля є побудова діаграм. Тут будемо використовувати сторонню бібліотеку OxyPlot, яка дозволяє досить легко будувати на формі діаграми різних типів.
- ReportBuilder. Відповідальністю модуля є генерація звіту

4.3 Розробка графічного дизайну

Графічний дизайн буде розроблюватись за допомогою фреймворка WPF від Microsoft на мові C#. Особливістю фреймвока є те, що всі графічні елементи спираються на відповідні моделі. Тож перш за все, має бути розроблений макет відображення, на базі якого мають бути розроблені інтерфейс та відповідні моделі.

За основу дизайну візьмемо стандартний дизайн ANSYS, який уже багато років обновлювався та редагувався під бажання користувачів, тож може вважатися оптимальним.

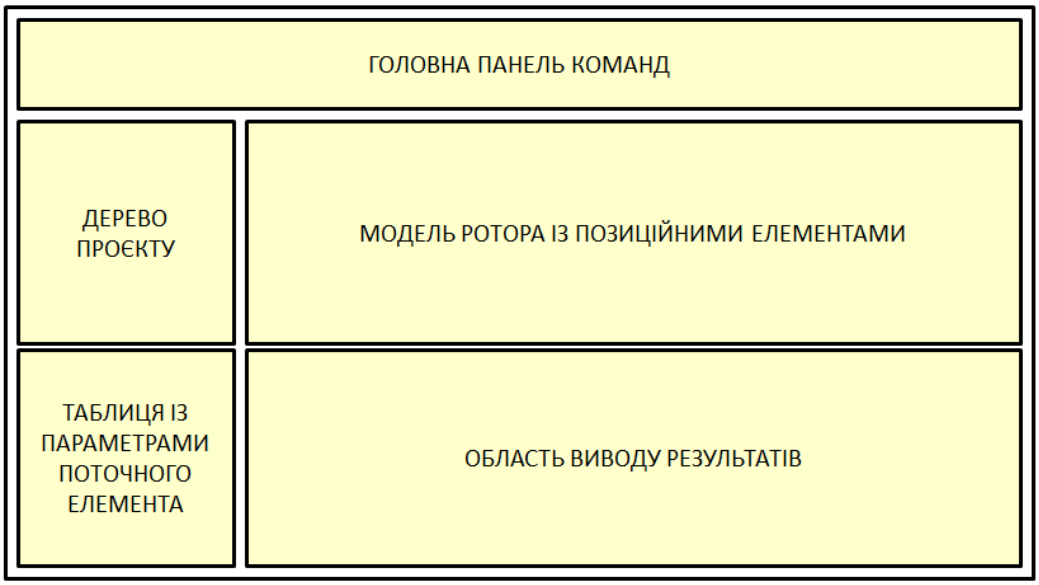


Рисунок 4.2 – Макет головного вікна програми

Розробити макет для WPF додатку можна за допомогою Microsoft Blend. Нижче будуть подаватися макети елементів, створених у цьому додатку. Іконки елементів меню вибрані довільно.

4.3.1 Головна панель команд

В головну панель команд як правило додаються ключові команди програми. В нашому випадку додамо на панель наступні команди:

1. Створити новий проєкт.
2. Відкрити проєкт. Тут має сенс реалізувати можливість відкрити як файл проєкту програми, так і файл MS Excel
3. Зберегти проєкт. Проєкт за допомогою стандартних діалогів зберігається у спеціальний файл, який можна відкрити цією програмою.
4. Додати елемент схеми. Цей елемент інтерфейсу можна зробити комбінованим, тобто надати можливість за його допомогою додавати всі необхідні елементи схеми. А це елемент ротора, опора, додана маса, та додана жорсткість
5. Розрахувати схему

6. Сформувати звіт. Звіт повинен містити інформацію про саму модель, та результати розрахунків. Найпростіше такий звіт буде виконати як html сторінку.

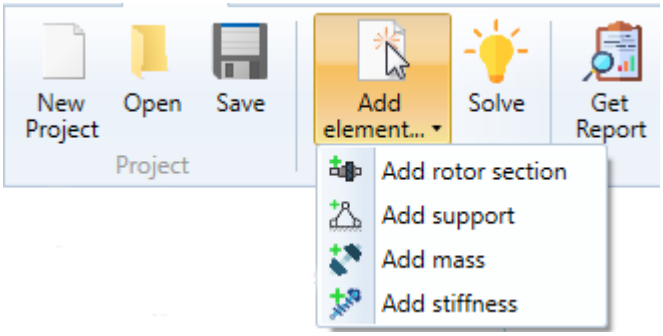


Рисунок 4.3 – Головна панель команд

4.3.2 Дерево проєкту

Дерево проєкту буде містити структуровану інформацію по поточному проєкту, а також дублювати можливість додавання елементів. При цьому (як і при додаванні з головної панелі команд) дерево має обновитися і показати новий елемент. На рівні безпосередніх потомків кореня дерева мають знаходитися категорії елементів, які можуть бути додані, при цьому додавання елементів на рівні категорій має бути заборонено. Тобто, кількість категорій обмежена тим списком, що був запрограмований, і користувач не може додавати нових категорій.

Самих категорій буде п’ять:

- Налаштування (Settings). Цей розділ буде містити різноманітні налаштування програми. Наразі він буде містити тільки один елемент налаштувань проєкту (Project Settings). На відміну від інших категорій, додавання користувачем елементів до категорії налаштувань, та до всіх потомків цієї категорії має бути заборонено.
- Елементи ротора (Rotor Sections). Категорія буде містити елементи ротора, який досліджується. При додаванні нової секції вона має з’явитися в області моделі ротора, та відповідати параметрам, які для неї завдані.

- Опори (Supports). Категорія буде містити опори, на яких знаходиться ротор. Наразі передбачується тільки один тип опори, а саме шарнірна нерухома опора. У разі необхідності змоделювати жорстке защемлення, можна використати натомість дві шарнірні нерухомі опори на невеликій відстані одна від одної.
- Додані маси (Added mass). Категорія буде містити додані маси, які будуть імітувати реальні тіла, що містяться на роторі, такі як зубчасті колеса, шківи, колеса турбін тощо.
- Додані жорсткості (Stiffness). Категорія буде містити елементи, що додають елементам ротора додаткової жорсткості.

Додавання дочірніх елементів для всіх категорій має бути дозволено. При додаванні елемента в категорії, в дереві має з'явитись відповідний дочірній елемент. Також має бути можливість видалення існуючих дочірніх елементів.

Операції додавання та видалення дочірніх елементів мають міститись в контекстному меню для відповідних елементів дерева – додавання для елементів категорій, видалення для дочірніх елементів.

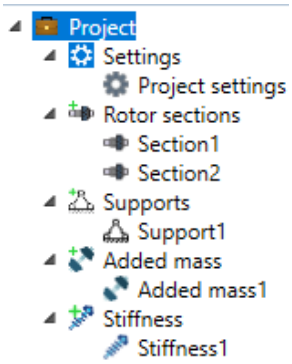


Рисунок 4.4 – Дерево проєкту

4.3.3 Таблиця параметрів

Кожна категорія має свій набір параметрів для своїх елементів. В таблиці параметрів буде наведено список таких параметрів, частину яких треба задавати користувачеві, а частина яких розраховується автоматично. Інформація в таблиці має змінюватись в залежності від того, який елемент дерева наразі вибраний, та

відображати параметри для нього. Нижче наведено списки параметрів для кожної з категорій

Таблиця 4.1 – Параметри категорії налаштувань

Номер параметру	Назва параметру	Відображення	Завдається користувачем	Діапазон значень
1	Кількість критичних частот в результаті розрахунку	Freq. count	Так	1..10
2	Максимальна частота для розрахунку	Max frequency	Так	300..20000
3	Тип глобального вирішувача	Global solver type	Так	0..2
4	Тип вирішувача для розрахунку критичних частот	Crit. freq. solver type	Так	1..2
5	Тип вирішувача для розрахунку форм відносних деформацій	Def. form solver type	Так	1..2
6	Точність кроку курсора в області моделі ротора	Picker step prec.	Так	0..5

Як відомо, кількість критичних частот у ротора нескінченно багато. Але знати їх всі немає сенсу, так як більшість з них лежить за межею режимів роботи ротора. Тому кількість шуканих частот обмежується за допомогою параметра 1, що дозволяє зекономити ресурси комп’ютера на розрахунки. Додатково, щоб не розрахувати частоти, які не лежать в межах режимів роботи ротора додано параметр 2, який встановлює верхню межу значення частоти.

Типи вирішувачів будуть обговорюватись у відповідному пункті, тож призначення значень параметрів, пов’язаних із ними буде розкрито далі.

Для області моделі ротора передбачається розробити функціонал, яким можливо було б знімати значення координат і переводити їх в координати реального масштабу ротора. Це може бути корисним для додавання позиційних елементів. Проте якщо знімати реальні координати курсора, вони будуть мати близько п’яти знаків після коми, та змінюватись від найменшого руху. Це зводить до нуля корисність такого функціоналу. Тож параметр 6 визначає кількість знаків після коми, на зміну яких відбувається оновлення інформації курсора, а також ця кількість знаків і буде подана в результаті.

Property	Value	Unit
Freq. count	3	
Max frequency	10000	**Hz**
Global solver type	0	
Crit. freq. solver type	2	
Def. form solver type	1	
Picker step prec.	3	

Рисунок 4.5 – Таблиця параметрів для налаштувань проєкту

Параметри 1-3 в таблиці 3.2 завдають геометрію елемента ротора, такі як зовнішній та внутрішній діаметри, та довжину. За замовчуванням ці значення будуть рівними нулю, а користувач має виставити необхідні йому значення. Якщо у елемента ротора немає отвору, слід залишити значення внутрішнього діаметра рівним нулю. Слід зазначити, що наразі в застосунку не передбачається системи переводу величин, тож всі значення лінійних розмірів завдаються в метрах.

Параметри 4-5, відповідно завдають значення для властивостей матеріалу елемента, таких як модуль Юнга та щільність, і за замовчуванням мають значення для конструкційної сталі 40.

Параметр 6 визначає позицію лівого торця елемента. Ця позиція розраховується автоматично, як позиція правого торця попереднього елемента, або нуль.

Таблиця 4.2 – Параметри категорії елементів ротора

Номер параметру	Назва параметру	Відображення	Завдається користувачем	Діапазон значень
1	Зовнішній діаметр	Outer diameter	Так	> 0
2	Внутрішній діаметр	Inner diameter	Так	≥ 0
3	Довжина	Length	Так	> 0
4	Модуль Юнга	Young’s modulus	Так	> 0
5	Щільність	Density	Так	> 0
6	Координата лівого торця	Position	Ні	≥ 0
7	Маса	Mass	Ні	> 0
8	Момент інерції перерізу	Moment of inertia	Ні	> 0

Параметри 7 та 8 визначають масу та момент інерції перерізу елемента ротора. Наразі надається можливість будувати елементи тільки круглого перерізу, тож ці значення також вираховуються автоматично в залежності від завданих користувачем даних.

Можливість зміни користувачем значень параметрів 6-8 має бути заборонена.

Property	Value	Unit
Outer diameter	0.03	**m**
Inner diameter	0	**m**
Length	0.05	**m**
Young's modulus	1970000000(	**Pa**
Density	7755	**kg/m**
Position	0	**m**
Mass	0.274084324	**kg**
Moment of inertia	3.976078202	**kg*m**

Рисунок 4.6 – Таблиця параметрів для елемента ротора

Таблиця 4.3 – Параметри категорії опор

Номер параметру	Назва параметру	Відображення	Завдається користувачем	Діапазон значень
1	Координата прикладання опори	Position	Так	$L > x \geq 0$
2	Жорсткість опори	Stiffness	Так	$\geq 10^7$

Параметр 1 визначає координату прикладання опори, і вочевидь не може приймати значення більші, ніж довжина ротора.

Параметр 2 визначає умовну жорсткість опори. В розрахунках накладання граничних умов виконується за методом штрафів (див. 3.2.6), тож щоб зробити виключення зафіксованих елементів там використовується деяке велике число. За замовчуванням це 10^7. Тож, жорсткість для закріпленої частини ротора буде умовно вважатися дуже великою, і таким чином відрізнятися від незафіксованих елементів. Також слід зазначити, що ця умова буде діяти від значення координати до наступного елемента ротора, або до кінця ротора. Тож якщо треба змоделювати ротор одного діаметра, частина якого закріплена, закріплену частину треба вводити як окремий елемент, інакше умова закріплення поширюється на весь ротор.

Property	Value	Unit
Position	0.02	**m**
Stiffness	10000000	**N/m**

Рисунок 4.7 – Таблиця параметрів для опори

Параметр 1 визначає координату прикладання маси, і вочевидь не може приймати значення більші, ніж довжина ротора.

Таблиця 4.4 – Параметри категорії доданої маси

Номер параметру	Назва параметру	Відображення	Завдається користувачем	Діапазон значень
1	Координата прикладання маси	Position	Так	$L > x \geq 0$
2	Значення маси	Added mass	Так	> 0
3	Значення моменту інерції	Added inertia moment	Так	> 0

Параметр 2 визначає власне значення доданої маси, а параметр 2 момент інерції цієї маси. В поточній версії програми не передбачається додаткового функціоналу для визначення цих значень для типових деталей, тож, наприклад, щоб додати умовне зубчасте колесо, користувач сам повинен заздалегідь визначити його масу та момент інерції.

Для доданої маси діє таке ж правило розповсюдження, як і для опори.

Property	Value	Unit
Position	0.03	**m**
Added mass	5	**kg**
Added inertia momen	0.01	**kg*m**

Рисунок 4.8 – Таблиця параметрів для доданої маси

Таблиця 4.5 – Параметри категорії доданої жорсткості

Номер параметру	Назва параметру	Відображення	Завдається користувачем	Діапазон значень
1	Координата прикладання опори	Position	Так	$L > x \geq 0$
2	Жорсткість опори	Stiffness	Так	> 0

Категорія доданої жорсткості по своїй суті дуже близька до категорії опор, с тією різницею, що значення жорсткості може бути будь-яким.

Property	Value	Unit
Position	0.015	**m**
Stiffness	12.5	**N/m**

Рисунок 4.9 – Таблиця параметрів для доданої жорсткості

Окрім самої таблиці в області ще є кнопка Submit. Її призначення передавати зміни в таблиці в головну модель. На момент того, як таблиця заповнена або зміни до неї занесені, головна модель ще лишається неоновленою, і буде оновлюватись тільки після натискання цієї кнопки. Оновлення даних досить важка з точки зору ресурсоємності процедура, тож виконувати її на кожну зміну в таблиці не розумно. Тому оновлення відбувається по команді користувача, коли він впевнений, що всі дані актуальні.

4.3.4 Область ескізу моделі ротора

Під час моделювання роторам додаванням його елементів та позиційних елементів користувачу потрібно бачити саму модель ротора, яка виходить, чи вірно вона згенерована, і чи вірно розставлені позиційні елементи. Для цього в додатку в області моделі динамічно відображається ескіз моделі в масштабі. Позиційні елементи відображаються на ескізі умовними позначеннями:

- Опора – зелений трикутник
- Додана маса – зелений квадрат
- Додана жорсткість – помаранчевий квадрат

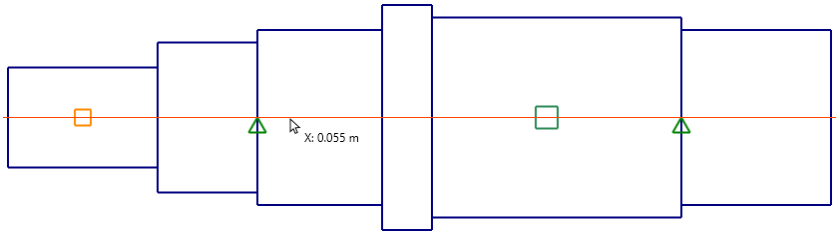


Рисунок 4.9 – Ескіз моделі ротора

Для більшої зручності завдання позиційних елементів, поруч із курсором вказується координата довжини в реальному масштабі. Також для області ескізу моделі ротора буде додано контекстне меню з наступними командами:

- Pick X coord – встановить значення параметра Position поточного елемента таким, яке займає курсор на ескізі

- Add Support, Add Mass, Add Stiffness – команди додадуть відповідний елемент за поточною координатою

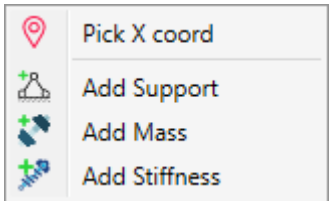


Рисунок 4.10 – Контекстне меню для області ескізу ротора

4.3.5 Область результатів

Область результатів містить дані, які були отримані під час розрахунку, і взагалі є кінцевою метою роботи додатку. Тут після проведення розрахунку можна побачити список значень критичних частот та очікувану форму відносних деформацій в залежності від частоти.

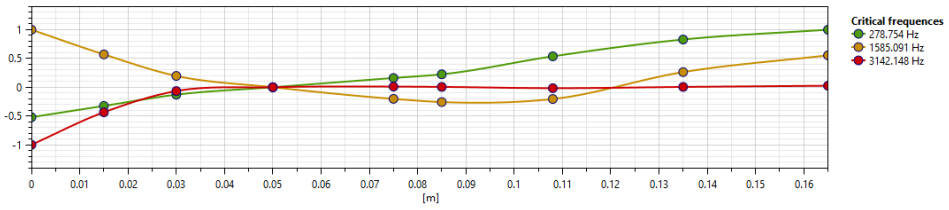


Рисунок 4.11 – Контекстне меню для області ескізу ротора

4.4 Розробка моделей

4.4.1 Модель дерева проєкту

Як було сказано вище, особливістю фреймворка WPF є те, що модель повинна відзеркалювати елементи інтерфейсу. Тож, маючи макет інтерфейсу, можна почати будувати моделі.

Для відображення деревовидної структури добре підійде ієрархічна модель. Однак у нашому випадку, вочевидь, елементи дерева мають різну структуру. Перший тип елементів це категорії. Вони не мають власних параметрів, однак мають дочірні елементи, які представляють другий тип елементів. Другий тип навпаки повинен мати набір параметрів, але не мати дочірніх елементів. Крім того, елементи другого типу також можна поділити на позиційні елементи, тобто такі, де користувач сам завдає їх положення, та елементи ротору, де користувач може завдати тільки геометрію елемента, а сама позиція розраховується автоматично. Проте всі ці елементи мають бути одного типу.

Щоб вийти з ситуації можна скористатися таким механізмом, як наслідування. Тобто, створити мінімалістиний батьківський клас, від якого наслідувати всі необхідні класи, які в свою чергу будуть зберігати в собі всі необхідні дані. Тоді модель дерева буде являти собою ієрархію елементів одного батьківського типу, але, привівши конкретний елемент до необхідного дочірнього типу, можна отримати з нього всю необхідну інформацію.

Діаграму класів моделі дерева представлено на рис. 4.12. Центральним елементом тут є абстрактний клас TgvItem який є батьківським класом для всіх елементів дерева, та містить члени, які буде містить будь який елемент дерева. Детальний опис членів класу наведено в таблиці 4.6.

В діаграмі також є два незалежні переліки CategoryType та ItemPropertyType. Перший перелік завдає типи категорій, які будуть доступні в дереві. Другий перелік завдає всі параметри, які будуть завдані для елементів дерева. Ці переліки потрібні для одностайного визначення категорій та властивостей елементів.

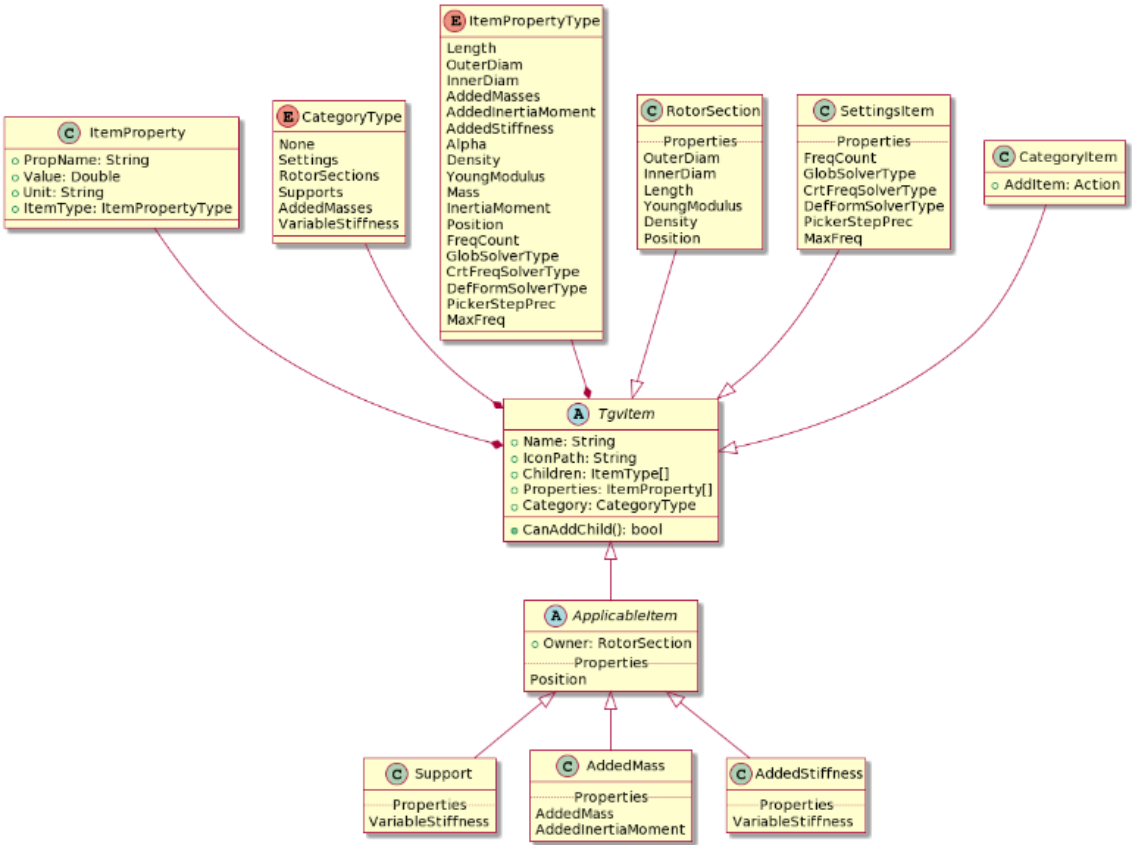


Рисунок 4.12 – Діаграма класів моделі дерева проєкту

Серед незалежних елементів на діаграмі присутній клас ItemProperty, що являє собою структуру для збереження однієї властивості елементу.

Таблиця 4.7 – Опис елементів класу TgvItem

Назва елемента класу	Опис
Name	Ім’я поточного елемента, яке буде відображатися в дереві
IconPath	Відносний шлях до файлу із зображенням для поточного елемента дерева
Category	Категорія поточного елемента дерева згідно з переліком CategoryType
Children	Колекція дочірніх елементів поточного елементу
Properties	Колекція властивостей поточного елементу
CanAddChild	Функція, що визначає, чи можна додавати дочірні елементи до поточного

Передбачається, що кожний елемент дерева буде мати зумовлену колекцію властивостей, які характерні саме для нього. В свою чергу, така колекція необхідна як модель для таблиці з параметрами. Опис елементів класу наведено в таблиці 4.9

Таблиця 4.9 – Опис елементів класу ItemProperty

Назва елемента класу	Опис
PropName	Ім’я поточної властивості
Value	Значення поточної властивості
Unit	Одиниця виміру для поточної властивості
ItemType	Тип поточної властивості згідно з переліком ItemPropertyType

Також на рис. 4.12 можна побачити декілька класів, успадкованих від класу TgvItem. Це таці класи як:

- RotorSection
- SettingsItem
- ApplicableItem
- CategoryItem

Перші три класи представляють листові елементи дерева, тобто конкретні елементи ротору, опори, маси тощо. Такі елементи не можуть мати дочірніх елементів, тож їх метод CanAddChild має вертати false.

Окремо слід роздивитися клас CategoryItem, який представляє собою категорію, що згруповує під собою деякі сутності дерева проекту, такі як елемент ротора, опора тощо. У такого класу немає властивостей, натомість повинна бути функція, що буде створювати нові елементи відповідного типу та додавати їх в поточну категорію. Користувач не може створювати свої категорії, їх кількість та параметри визначається програмно. Із списком та призначенням категорій можна ознайомитись в п.4.2.2.

Листові елементи дерева, як вказувалося раніше, можна поділити на дві групи – позиційні та непозиційні (див. п. 4.3). До непозиційних елементів входить тільки елемент ротору (RotorSection). Його властивості наведено в таблиці 4.2. Тож функція додавання елементів ротору повинна додати в колекцію властивостей відповідні властивості із значеннями за замовчуванням.

До позиційних елементів відносяться опори, додані маси та жорсткості. Їх особливість в тому, що ключовою властивістю для кожного з них є координата додавання елемента, яку користувач може безпосередньо завдати. Крім того, в розрахунках буде корисно знати до якого саме елемента ротора належить той чи інший позиційний елемент. Тож, можна створити батьківський абстрактний клас, який буде одразу містити властивість Position та посилання на батьківський елемент ротора. Від цього класу успадкувати конкретні класи для елементів, що перелічені вище, додаючи кожному з них необхідні властивості. Саме це і відображено на діаграмі класів на рис. 4.13.

Виходячи з вищесказаного, дерево на рис.4.4 можна представити як ієрархію зазначених класів (рис. 4.13)

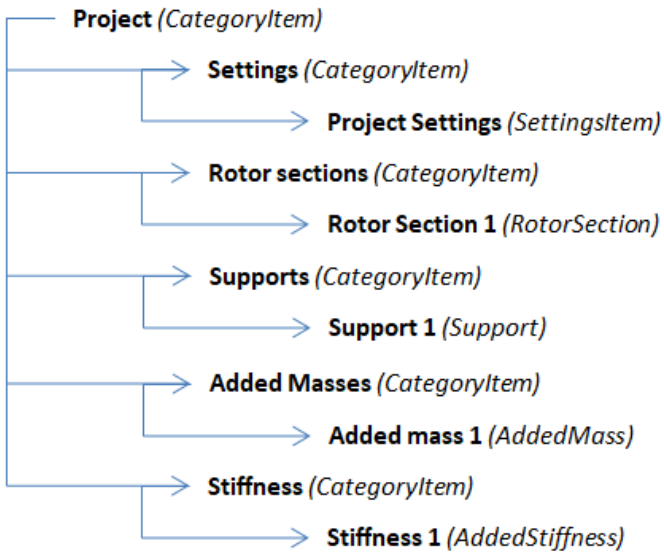
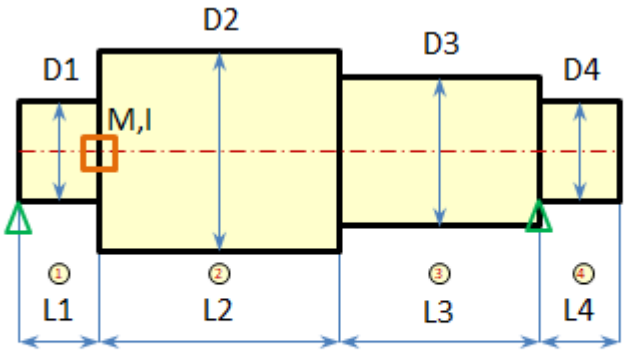


Рисунок 4.13 – Ієрархія класів в дереві проєкту

4.4.2 Модель для розрахунків

Модель описана в 4.3.1 добре підходить для відображення на формі, але погано підходить для розрахунків. Тож, є необхідність генерувати більш компактну і просту модель, з якої було б зручно формувати матриці та і в цілому вибирати необхідні для розрахунків дані. Такою моделлю може бути звичайна двомірна таблиця, рядки якої задавались би типами властивостей, я стовпчики елементами ротора.

Реалізувати таку таблицю було б доцільно за допомогою мапи, ключами якої були б значення переліку ItemPropertyType, а значеннями масиви чисел, де індекси співпадали б з умовним номером елемента ротора, а значення містили б значення конкретних властивостей для цих елементів ротора (рис. 4.14).



Тип параметра (ItemPropertyType)	1	2	3	4
Length	L1	L2	L3	L4
OuterDiam	D1	D2	D3	D4
InnerDiam	0	0	0	0
AddMass	0	M	0	0
AddInertMoment	0	I	0	0
Stiffness	1E7	0	0	1E7
...				

Рисунок 4.15 – Приклад побудови моделі для розрахунків

4.4.3 Модель для області ескізу

Область ескізу необхідна для відображення ескізу ротора, тож в цій області повинні бути відмальовані певним чином лінії, які б склалися в контур ротора в масштабі. Найпростіший підхід для цього – це зберігати колекцію ліній, а при оновленні даних затирати її та генерувати знову.

З іншого боку, зберігати суто колекцію ліній не практично, так як вони можуть мати різні кольори та товщину. Річ у тому, що такі властивості, як колір і товщина не зберігаються в самій лінії.

Щоб вирішити цю проблему, створимо спеціальний клас-обгортку Segment, який буде зберігати всю необхідну інформацію.

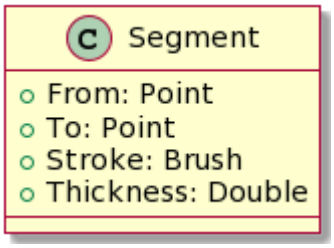


Рисунок 4.16 – Структура класу для відображення ліній

Тут Point та Brush це стандартні класи мови C# з неймспейсу System.Windows. Клас Point реалізує двумірну точку із координатами у вигляді чисел із плаваючою точкою. Клас Brush містить інформацію стосовно кольору та прозорості намальованого примітиву

Завдяки простоті дана модель легко реалізується і впроваджується, але її недоліком є те, що так можна рисувати тільки прямі відрізки. Але на даному етапі розвитку проєкту цього цілком достатньо.

4.4.3 Модель результату

Як було зазначено в 4.2, результати розрахунку доцільно об’єднати в одну структуру. Така структура має складатися з двох частин. Першою має бути перелік критичних частот, другою – колекція масивів значень деформацій, іншими словами двомірна таблиця. Вочевидь, рядки такої таблиці мають відповідати елементам переліку частот. Слід також зазначити, що двомірна

таблиця із значеннями деформацій добре підходить як модель для побудування діаграм за допомогою OxyPlot. Тож, таким чином також закривається питання щодо моделі для побудови діаграм.

4.5 Розробка функціоналу

Із структурної схеми додатку (рис.4.1) добре видно функціональні блоки, які потрібно реалізувати. Більшість блоків з цього набору є суто технічними, і не містить нічого цікавого з точки зору методології поточного дослідження. Тому не будемо загострювати на них увагу. Натомість розглянемо блоки, які безпосередньо відповідають за ключові розрахунки. У нашому випадку це

- Визначник критичних частот
- Визначник форм відносних деформацій

Для обох метод в якості вихідних даних необхідні матриці мас та жорсткостей. Тому доцільно створити структуру, яка б розраховувала загальні дані, а згодом викликала ту чи іншу реалізацію для розрахунку власне частот.

Така архітектура представлена на рис. 4.17

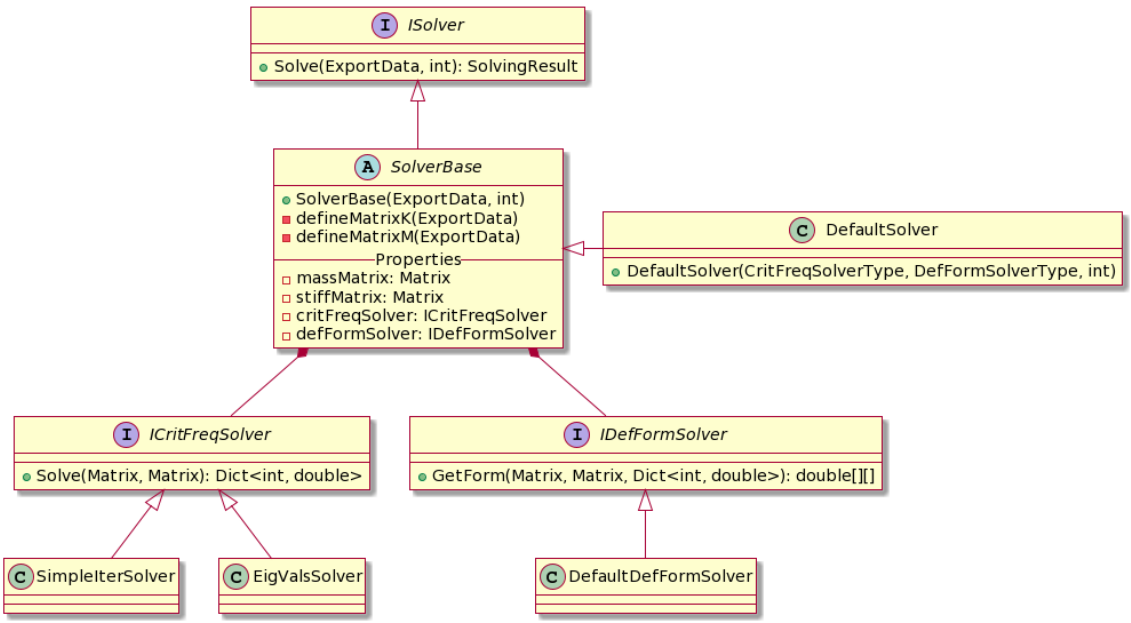


Рисунок 4.17 – Структура підпорядкування вирішувачів

Тут центральним елементом є абстрактний клас SolverBase, який реалізує інтерфейс ISolver. В ньому обчислюються матриці мас та жорсткостей. Крім того, клас містить посилання на інтерфейси ICritFreqSolver та IDefFormSolver, реалізації яких будуть розраховувати значення критичних частот, та форм відносних деформацій. Такий підхід дозволяє гнучко перемикати реалізації вирішувачив, та додавати нові реалізації.

Так як абстрактний клас не може мати власні екземпляри, його теж треба дореалізувати конкретним класом. На діаграмі це клас DefaultSolver. Конструктор класу приймає спеціальні значення, за допомогою яких створюються конкретні реалізації вирішувачів, та максимальне значення частоти.

4.5.1 Алгоритми для визначення критичних частот

Як було показано в 3.4, до реалізації маємо два методи знаходження критичних частот.

У класах SimpleIterSolver та EigValsSolver будуть реалізовані вирішувачі за методами простої ітерації та власних значень відповідно.

Так як вони є потомками інтерфейсу ICritFreqSolver, то також мають по одному методу Solve з такими ж вхідними параметрами та результатом, як і у відповідного інтерфейсу. Метод Solve приймає матриці мас та жорсткості, і вертає мапу, де значеннями є власне критичні частоти, а ключами їх індекси.

Алгоритм для визначення критичних частот методом простих ітерацій виконаний за методологією з 3.4.1. Його блок-схема представлена на рис. 4.18.

Із блок-схеми видно, що маючи на вході модель, перш за все формуються матриці мас та жорсткостей. Цей процес має відбуватися у рамках класу SolverBase.

Далі формується пустий масив результатів, та ініціалізується початкова частота, яка і буде змінюватись під час ітерацій. На кожному кроці додаємо одиницю до значення частоти, та знаходимо детермінанти матриць для поточного та попереднього значення частоти. Якщо результат множення детермінантів від’ємний, додаємо поточне значення частоти до масиву результатів, інакше повертаємось до початку ітерації.

Ітерація має відпрацьовувати до тих пір, поки значення поточної частоти не перевищить максимальне значення частоти, або не буде знайдена необхідна кількість частот.

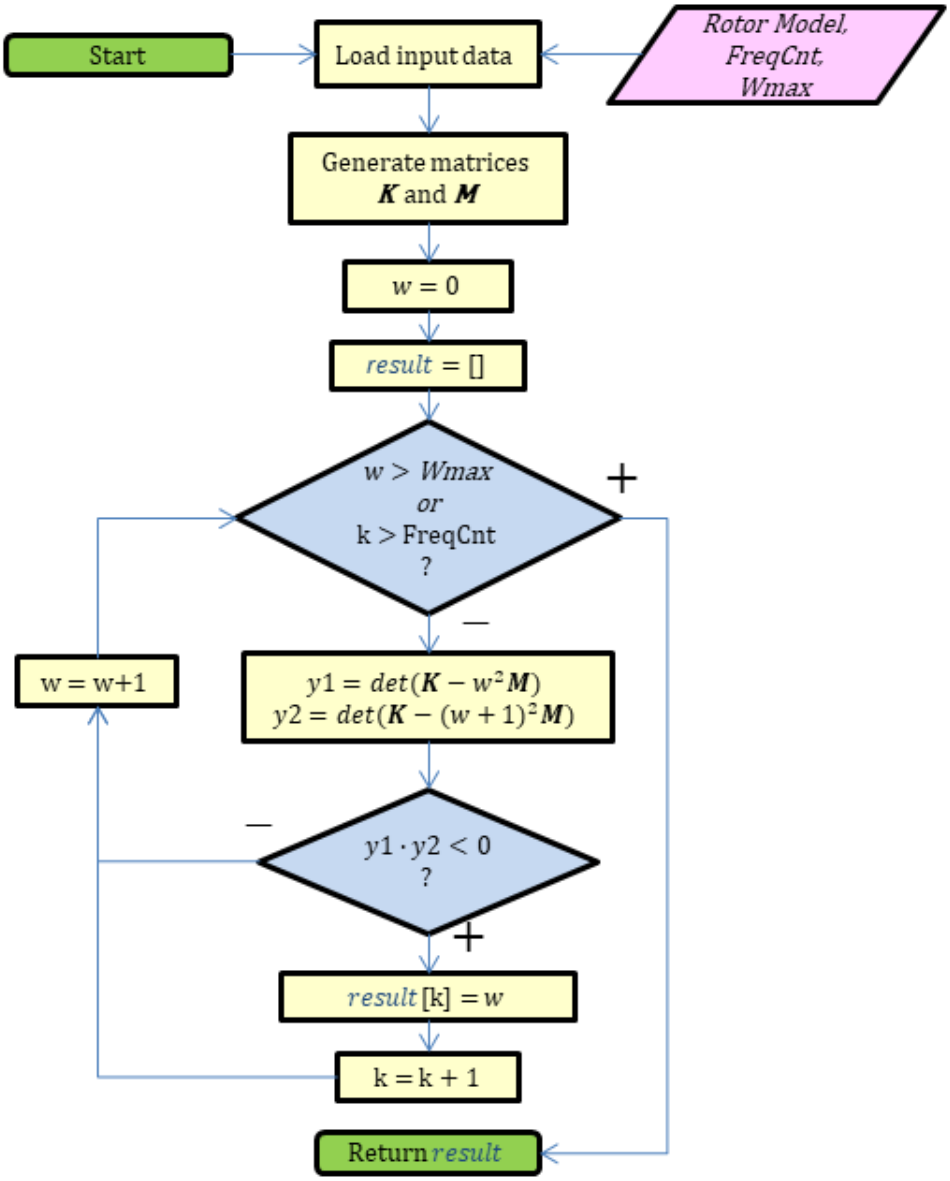


Рисунок 4.18 – Блок-схема алгоритму знаходження критичних частот за методом простих ітерацій

Алгоритм для визначення критичних частот методом простих ітерацій виконаний за методологією з 3.4.2. Його блок-схема представлена на рис. 4.19.

Із блок-схеми видно, перші кроки алгоритму такі ж самі, як у попередньому випадку. Далі розраховується спеціальна матриця D, для якої розраховуються власні значення під квадратним коренем. Власне значення може бути комплексним числом, але в нашому випадку такі значення не є валідними. Тож наступними кроками алгоритм відфільтровує комплексні значення, та залишає тільки реальні.

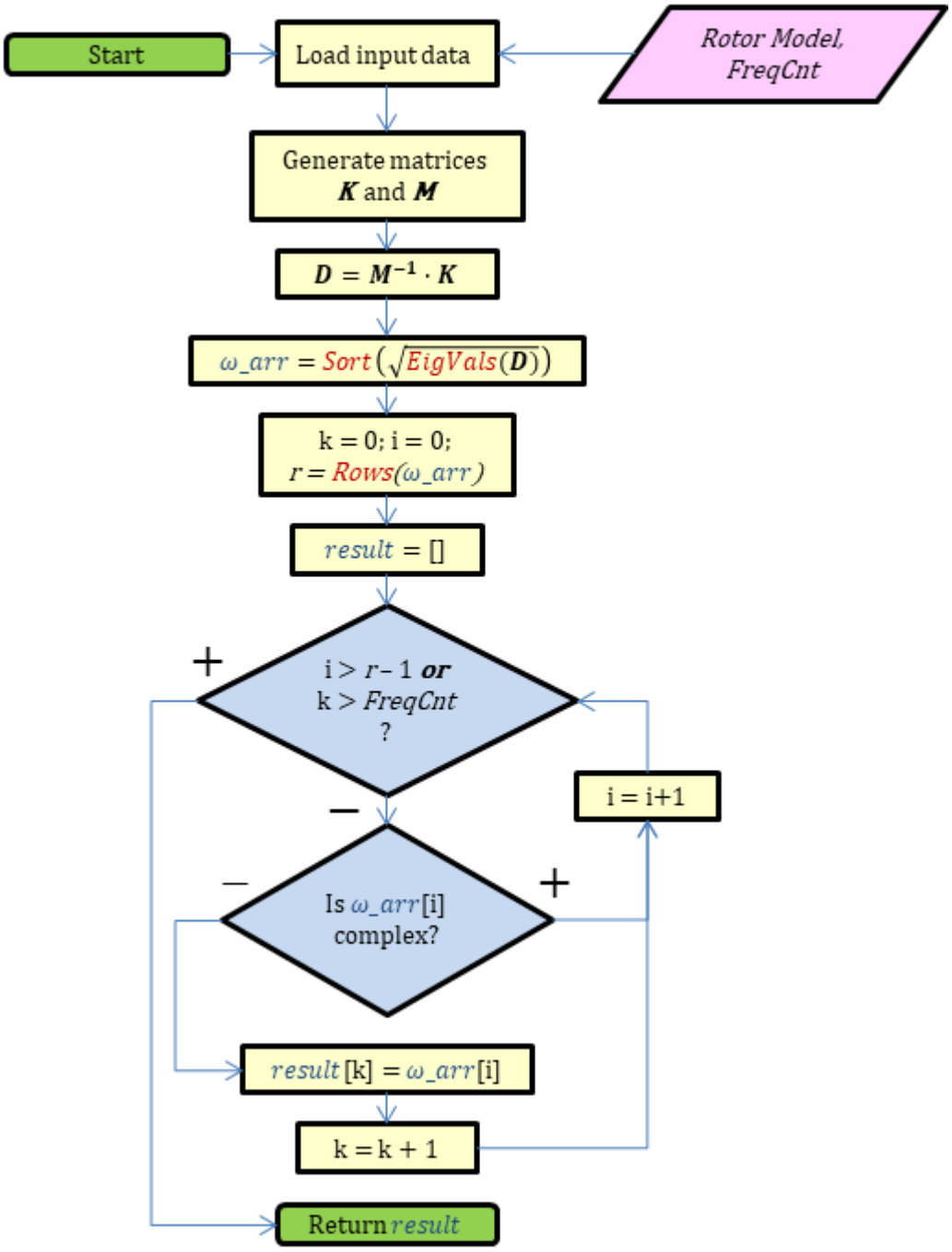


Рисунок 4.19 – Блок-схема алгоритму знаходження критичних частот за методом власних значень

4.5.2 Алгоритм для визначення форм відносних деформацій

Розрахунок форм відносних деформацій реалізовано в класі DefaultDefFormSolver.

Алгоритм для визначення форм відносних деформацій виконаний за методологією з 3.5. Його блок-схема представлена на рис. 4.20.

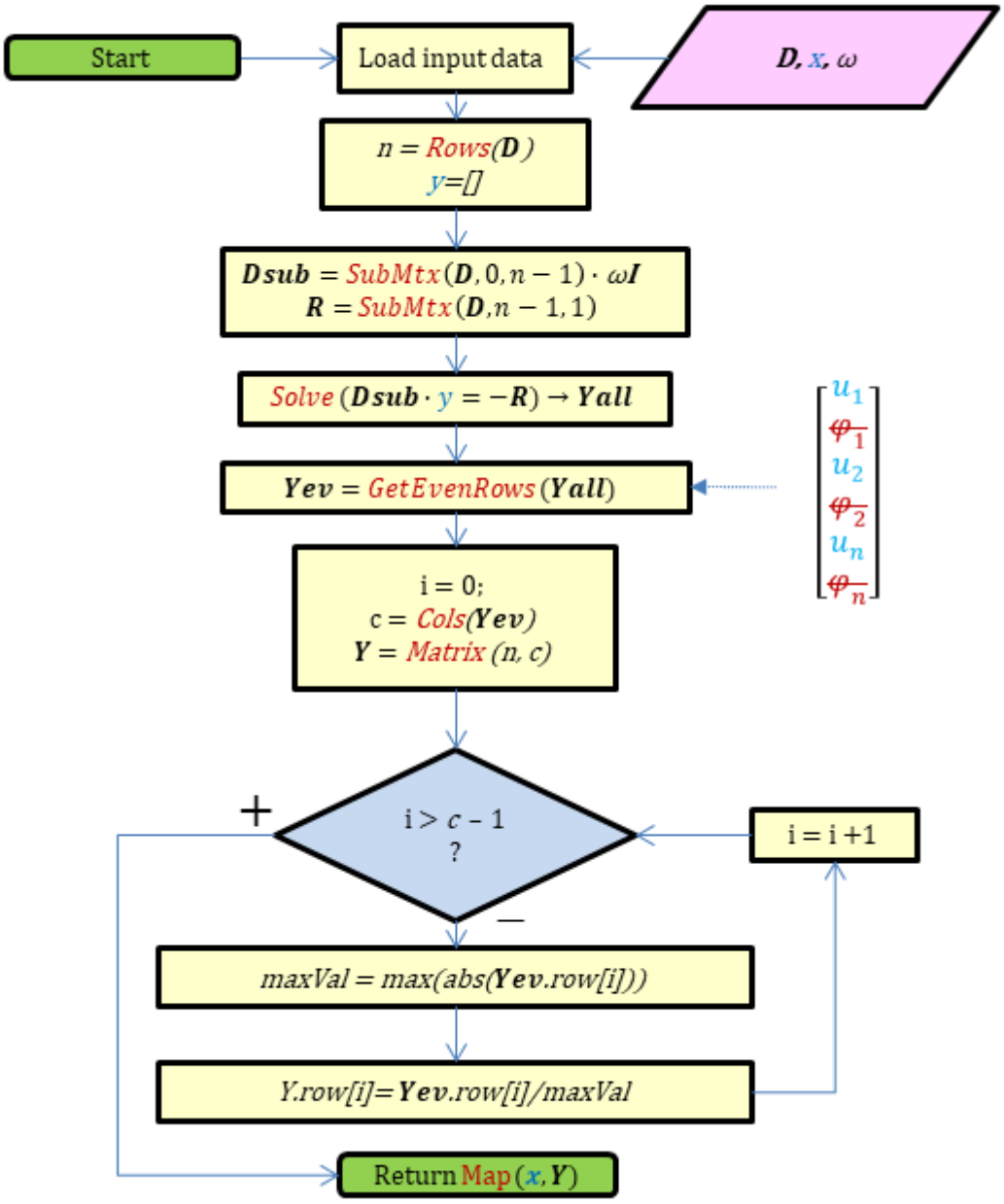


Рисунок 4.20 – Алгоритм розрахунку форм відносних деформацій

Вихідними даними алгоритму є матриця D (див. 3.58), вектор координат особливих точок ротора (таких як точки прикладання опор, мас або змінної

жорсткості, або точки, де змінюється геометрія перетину ротора) та деяке значення критичної частоти.

Першим кроком із вихідної матриці формується система рівнянь з нетривіальними рішеннями (див. п. 3.5). Для цього від матриці відкидається останній рядок та останній стовпець.

Від останнього стовпчика відкидається останнє значення (що відповідає відкинутому рядку). Це буде вектор вільних членів для матриці коефіцієнтів, що була сформована вище.

Наступним кроком отримана система вирішується, при цьому отримуємо вектор рішень, який послідовно містить пари значень відносних деформацій та кутів прогину перетину. Оскільки нам потрібні тільки відносні деформації, з вектору рішень відкидаємо значення з непарними індексами, які є значеннями кутів прогину.

Кожний стовпчик отриманої матриці містить значення відносних переміщень у деякій точці ротора. Так як у нашому випадку важливі саме форми деформацій, а не їх значення, слід нормувати значення для одної частоти відносно одиниці. Для цього для кожного рядка знаходимо максимальний за модулем елемент, та ділимо рядок на це значення.

Результатом роботи алгоритму є мапа, ключем якої буде значення координати точки ротора, а значенням набір значень форм для різних частот для цієї точки.

5 Аналіз результатів роботи застосунку

В якості аналізу для оцінки коректності роботи застосунку було обрано три моделі валів різного ступеня загальної жорсткості, та з різними схемами закріплень. Для аналізу змоделюємо ці вали з такими ж граничними умовами та навантаженнями в ANSYS Workbench та COMSOL Multiphysics. Проведемо модальний аналіз моделей та порівняймо результати.

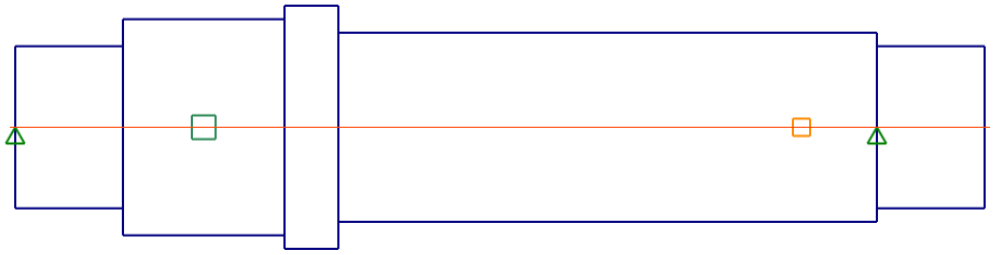


Рисунок 5.1 – Модель короткого валу з двома шарнірними опорами, завантаженого додатковою масою та з ділянкою певної жорсткості

Таблиця 5.1 – Порівняльна таблиця значень критичних частот

Номер моди	Значення власних частот, рад/с			Відносна похибка, %
	Застосунок	ANSYS	COMSOL	
1	219	216	225	1.4/3.1
2	595	615	611	3.5/2.8
3	1711	1777	1815	3.9/6.1

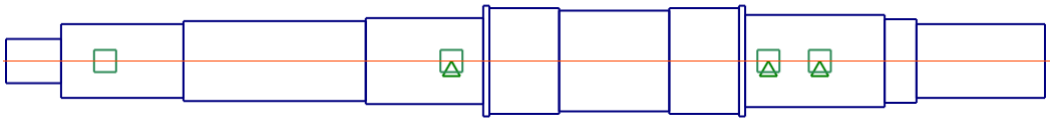


Рисунок 5.2 – Модель довгого складного валу з трьома шарнірними опорами та несиметричним навантаженням

Таблиця 5.2 – Порівняльна таблиця визначення критичних частот

Номер моди	Значення власних частот, рад/с			Відносна похибка, %
	Застосунок	ANSYS	COMSOL	
1	321	338	328	5.0/2.1
2	1125	1203	1185	6.4/5.0
3	1902	1857	1934	2.4/2.7

Як видно з таблиць, розбіжність між результатами застосунку та результатами професійних комплексів складає 1,5-7,0%. Але тут також варто підкреслити розбіжності практично такого ж порядку між результатами розрахунків ANSYS та COMSOL.

Така розбіжність зумовлена різними підходами до розрахунку, розмірністю та способом побудування моделей, та таке інше. Інакше кажучи, всі три моделі технічно різні, тож і результати розрахунків відрізняються.

Таким чином, результати роботи застосунку можна вважати релевантними.

6 Перспективи розвитку

В сучасному світі користувачі програмного забезпечення досить вибагливі, тож програмні продукти потребують постійного удосконалення якості та відмовостійкості, розширення функціоналу, поліпшення ергономічності та зручності, тощо. Інакше кажучи, продукт має постійно розвиватися.

Тож розглянемо перспективи розвитку для поточного продукту.

6.1 Перспективи у контексті поточного функціоналу

1. Додати базу матеріалів. Наразі властивості матеріалів, такі як щільність, модуль Юнга та інше, завдаються безпосередньо. База матеріалів дозволила б заповнювати властивості автоматично, залежно від обраного матеріалу.
2. Додати базу стандартних перетинів. Наразі можна працювати тільки з перетинами круглої форми. База стандартних перетинів, таких як перетини для шпоночного та шліцьового з’єднань з автоматичним розрахуванням мас та моментів інерції.
3. Вдосконалити графічну частину. Наразі графічна частина моделі ротора досить примітивна, та не дозволяє, наприклад, відобразити вал із осьовими отворами. Використання OpenGL або DirectX для графічної частини дозволило б виводити більш якісний, інтерактивний та інформативний контент.
4. Додати можливість побудови інших діаграм (АЧХ, діаграму Кампбела тощо)
5. Додати функціонал оптимізації завданої моделі. Було б корисно, якщо застосунок не тільки вираховував критичні частоти, а й пропонував варіанти їх збільшення за рахунок оптимізації геометрії вала.

6.2 Перспективи у контексті загального потенціалу продукту

1. Глибокий рефакторинг коду. Наразі застосунок можна вважати тільки прототипом. Його архітектура досить жорстка та примітивна. Глибокий рефакторинг, спрямований на розмежування та інкапсуляцію окремих модулів застосунку був би корисним для розширення функціональних можливостей застосунку.
2. Вдосконалення графічної частини. Як було сказано вище, удосконалення графічної частини добре сказалося б на зручності та функціональності продукту. Окрім того, таке удосконалення дозволило б перейти на відображення неодномірних задач.
3. Пристосування графічної оболонки для розробки інших видів аналізу. Попередні два пункти підводять до поточного. Загалом розроблену оболонку можливо б було використовувати для розробки дуже широкого спектра інструментів інженерного аналізу. В ідеальному варіанті, щоб використати оболонку під інший тип задач (наприклад розрахунок ферми), усе що було б потрібно, це розробити відповідні моделі у вигляді динамічної бібліотеки, та через спеціальний публічний інтерфейс приєднати до оболонки.

Висновки

1. У результаті проведеної роботи було проведено систематизацію та аналіз методології різних методів розрахунку критичних частот ротора, а на базі вивченої методології було розроблено прототип відповідного застосунку.
2. Застосунок виявився компактним та швидким. Інтерфейс є зручним та зрозумілим, тож дозволяє працювати з ним практично без попереднього навчання. Функціонал застосунку є достатнім для виконання поставленої мети.
3. Результати роботи застосунку мають деякі розбіжності із результатами, отриманими за допомогою професійних комплексів. Але, зважаючи на розбіжності між результатами самих комплексів, можна вважати результати релевантними.
4. Було виділено кілька перспективних напрямків подальшого розвитку продукту, які б удосконалили та розширили функціональні можливості застосунку, зберігши його зручність і простоту.

Тож загально можна зробити висновок, що сама ідея розробки подібного продукта є доцільною, оскільки це дозволило б мати простий та корисний на практиці інструмент.

Перелік джерел посилань

[1] J. Fish and T. Belytschko, A first course in finite elements. Chichester [U.A.] Wiley, 2012.

[2] O. C. Zienkiewicz, R. L. Taylor, and J. Z. Zhu, The Finite Element Method: Its Basis and Fundamentals. Elsevier, 2005.

[3] G. Strang, Introduction To Linear Algebra. S.L.: Wellesley-Cambridge Press, 2021.

[4] V. Mehrmann, Numerical Linear Algebra. Maynooth, 2010.

[5] V. A. Zorich, Mathematical Analysis. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015.

[6] И. В. Павленко, Метод конечных элементов в задачах сопротивления материалов и линейной теории упругости. СумДУ , 2006.

[7] І. В. Павленко, Метод скінченних елементів в задачах коливань механічних систем. СумДУ, 2007.

[8] R. K. Bansal, A TEXTBOOK OF STRENGTH OF MATERIALS. LAXMI PUBLICATIONS, 2009.

[9] L. N. Trefethen and D. Bau, Numerical Linear Algebra. SIAM, 1997.

[10] J. Solomon, Numerical Algorithms. CRC Press, 2015.

[11] S. S. Skiena, The algorithm design manual. London: Springer, 2008.

[12] “Strassen algorithm,” Wikipedia, May 04, 2022. https://en.wikipedia.org/wiki/Strassen_algorithm

[13] A. Matthew and S. Barman, The Coppersmith-Winograd Matrix Multiplication Algorithm. 2006.

[14] K.-J. Bathe, Finite Element Procedures. S.L.: S.N, 2014.

[15] Wikipedia Contributors, “Eigenvalues and eigenvectors,” Wikipedia, Mar. 29, 2019. https://en.wikipedia.org/wiki/Eigenvalues_and_eigenvectors