МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЧЕРКАСЬКИЙ ДЕРЖАВНИЙ ТЕХНОЛОГІЧНИЙ УНІВЕРСИТЕТ

В. Я. ГАЛЬЧЕНКО
А. В. СТОРЧАК
Р. В. ТРЕМБОВЕЦЬКА
В. В. ТИЧКОВ

ВИХРОСТРУМОВІ ЕКСПРЕС-ВИМІРЮВАННЯ ПРОФІЛІВ ЕЛЕКТРОФІЗИЧНИХ ВЛАСТИВОСТЕЙ ЦИЛІНДРИЧНИХ ОБ'ЄКТІВ

Монографія

ХАРКІВ
2026

УДК 620.179.147:620.186:519.853] (02)
Г17

Рекомендовано до друку Вченою радою Черкаського державного технологічного університету Міністерства освіти і науки України, протокол № 9 від 16 лютого 2026 р.

Р е ц е н з е н т и :

Заболотний О. В., декан факультету систем управління літальними апаратами Національного аерокосмічного університету ім. М.Є. Жуковського «Харківський авіаційний інститут», доктор технічних наук, професор

Куц Ю. В., професор кафедри автоматизації та систем неруйнівного контролю Національного технічного університету України «Київський політехнічний інститут ім. Ігоря Сікорського», доктор технічних наук, професор

Засядько А. А., провідний науковий співробітник Державного науково-дослідного інституту випробувань і сертифікації озброєння та військової техніки, доктор технічних наук, професор

Гальченко В. Я., Сторчак А. В., Трембовецька Р. В., Тичков В. В. Вихрострумові експрес-вимірювання профілів електрофізичних
Г17 **властивостей циліндричних об’єктів**: монографія / В. Я. Гальченко, А. В. Сторчак, Р. В. Трембовецька, В. В. Тичков. – Харків: СГ НТМ «Новий курс», 2026. – 174 с.

ISBN 978-617-7886-58-6

Монографія присвячена розв’язанню прикладної науково-технічної задачі вихрострумового одночасного експрес-вимірювання приповерхневих радіальних профілів електрофізичних характеристик, а саме електричної провідності та магнітної проникності, циліндричних об’єктів контролю. Детально розглянута поетапна реалізація створення експрес-методу, який передбачає апріорне накопичення даних щодо процесу вимірювань у нейромережевій сурогатній моделі та її використання для швидкого пошуку розв’язку оберненої задачі за запропонованою технологією дворівневого динамічного Lookup table.

Для наукових співробітників та інженерів, що працюють в галузі вихрострумового неруйнівного контролю, а також для аспірантів і студентів відповідних спеціальностей.

УДК 620.179.147:620.186:519.853] (02)

ISBN 978-617-7886-58-6

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ОК - об’єкт контролю
ЕП - електрична провідність
МП - магнітна проникність
ПЕ - план експерименту
ЕРС - електрорушійна сила
LUT - lookup table
ВСП - вихрострумовий перетворювач
APDL - ANSYS parametric design language
ШНМ, НМ - штучна нейронна мережа
MARs - multivariate adaptive regression splines
NURBs - non-uniform rational B-splines
ANN - artificial neural networks
RBF - radial basis function
MLP - multilayer perceptron
SVM - support vector machine
GMDH - group method of data handling
TPs - thin plate splines
DNN, DANN - deep artificial neural networks
FEM - finite element method
RL - reinforcement learning
GAN - generative adversarial network
AE - autoencoder
VAE - variational autoencoder
INN - invertible neural network
DL - deep learning
CVNN - complex-valued neural network
SCVNN - splittable complex-valued neural network
RVNN - real-valued neural network
MAE - mean absolute error
MSE - mean square error
RMSE - root mean square error
MAPE - mean absolute percentage error
DDS - direct digital synthesizer

ПЕРЕДМОВА

Сучасний розвиток науково-технічних напрямів у галузі неруйнівного контролю зумовлює необхідність створення нових методів та засобів для підвищення ефективності діагностики матеріалів і виробів. Особливого значення набуває дослідження електрофізичних характеристик приповерхневих шарів металевих деталей, оскільки саме в цих зонах внаслідок термічної чи термохімічної обробки відбуваються зміни мікроструктури, що визначають міцність, зносостійкість та надійність конструкцій у цілому. Традиційні методи контролю часто вимагають значних ресурсних витрат або не забезпечують достатньої точності та оперативності отримання результатів.

У цьому контексті особливої уваги заслуговують методи вихрострумового контролю, які дозволяють безконтактно та у реальному масштабі часу відтворювати розподіли електричної провідності та магнітної проникності в об’єктах циліндричної форми. Поєднання вихрострумових вимірювань з сучасними підходами до розв’язання обернених задач – зокрема із застосуванням сурогатних моделей на основі штучних нейронних мереж – відкриває нові можливості для суттєвого підвищення швидкодії та точності діагностичних систем.

Робота присвячена розробленню та обґрунтуванню нових методів побудови сурогатних моделей процесу вихрострумового контролю, створенню оптимальних комп’ютерних планів експериментів та реалізації багатопараметрового експрес-методу вимірювання радіальних профілів електрофізичних характеристик приповерхневих шарів циліндричних об’єктів. Отримані результати мають як теоретичну, так і практичну цінність, забезпечуючи підґрунтя для вдосконалення існуючих та створення нових високопродуктивних систем неруйнівного контролю.

Монографія складається з чотирьох основних глав, у яких послідовно викладено теоретичні засади, методичні рішення та практичні результати.

Перша глава присвячена огляду сучасних методів і підходів до розв’язання обернених задач у вихрострумовому контролі. Проаналізовано переваги використання сурогатних моделей, зокрема на основі штучних нейронних мереж.

Друга глава містить опис запропонованого універсального підходу до створення сурогатних моделей із застосуванням глибоких нейронних мереж та квазі-випадкових планів експериментів. Запропоновано новий багатопараметровий експрес-метод вимірювань у реальному часі.

Третя глава описує алгоритмічне та програмне забезпечення для моделювання процесів вихрострумового контролю. Виконано верифікацію моделей на основі аналітичних рішень та методу скінченних елементів. Наведено приклади створення сурогатних моделей у середовищі Python із використанням глибоких нейронних мереж.

Четверта глава присвячена апаратній реалізації вимірювальної системи. Описано конструктивні рішення спеціалізованого вихрострумового структуроскопа та особливості застосування методу Lookup tables з динамічними таблицями другого рівня.

У завершальній частині роботи сформульовано основні наукові результати та практичні висновки, що підтверджують ефективність запропонованого підходу.

ГЛАВА 1
ОГЛЯД МЕТОДІВ ТА ЗАСОБІВ РОЗВ'ЯЗАННЯ ЗАДАЧІ ВСТАНОВЛЕННЯ СТРУКТУРНИХ ОСОБЛИВОСТЕЙ МАТЕРІАЛУ ЦИЛІНДРИЧНИХ ОБ'ЄКТІВ КОНТРОЛЮ

1.1. Огляд методів визначення електрофізичних характеристик об'єктів вихрострумовим методом

Наразі є відомими чимала кількість варіантів щодо підходів розв'язку досліджуваної проблеми або суміжних задач вихрострумового контролю. Критичний аналіз відомої науково-технічної літератури показав особливості, доцільність і ефективність цих варіантів. Нижче наведений опис підходів та методів розв'язання обернених задач вихрострумового контролю із зазначенням їхніх переваг та недоліків. В статті [1] наведено приклад розв'язання оберненої задачі багатопараметрового контролю структурних змін матеріалу ОК змінно-частотним методом. Метод забезпечує контроль певної товщини шару матеріалу, що є корисним, наприклад, при контролі глибини термічної обробки матеріалу. Та хоч він і належить до багатопараметрових, що дозволяє контролювати інтегральний електромагнітний параметр $\eta = f(\mu, \sigma)$, але він не допускає окремо контролювати ЕП та МП. Багатопараметровий контроль кожного з параметрів можливий за допомогою окремої математичної моделі, що описує залежність вихідного сигналу від параметрів матеріалу. Підхід до розв'язання оберненої задачі є оптимізаційним на основі методу Флетчера–Пауела, що дозволяє оцінити відхилення виміряного та модельованого сигналу. Загалом метод є досить точним, але не використовує інформативність сигналу в повному обсязі, зокрема не залучає амплітуду як інформативну складову. До того ж оптимізаційні підходи, як правило, не дозволяють проводити обробку сигналу в наближеному до реального масштабі часу.

В публікації [1] розв'язання оберненої задачі електродинаміки щодо реконструкції структури ОК за виміряними сигналами вихрострумового перетворювача (ВСП) рекомендується знаходити засобами лінійного програмування. Лінійні припущення, описані в роботі [2], які використано при побудові математичної моделі для конструкції рис.1.1, а відповідно й запропонований метод

суперпозиції, не є строгими і значно спрощують реальні фізичні процеси. Крім того, при проведенні вимірювальних операцій використовуються декілька частот, що ускладнює проведення процедури.

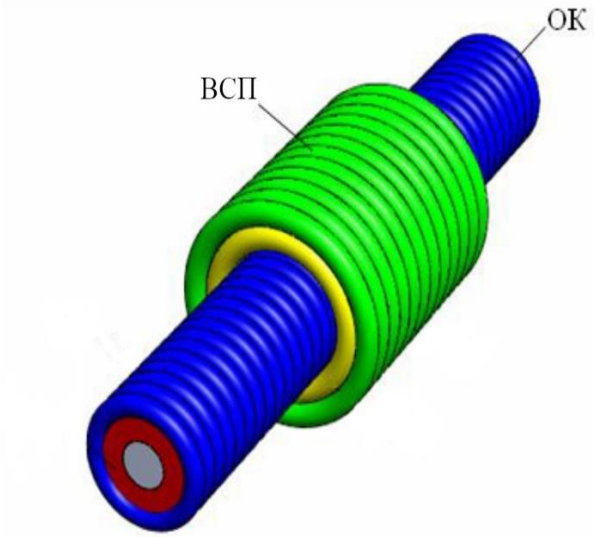


Рис. 1.1. Модель ВСП з циліндричним ОК, поверхня якого являє собою двошарову котушку.

У дослідженнях [3], [4] розглянуто електромагнітний перетворювач із просторово-періодичною структурою поля (рис. 1.2), що дозволяє проводити контроль та вимірювання параметрів ЕП та МП металевих виробів у формі протяжного феромагнітного циліндра.

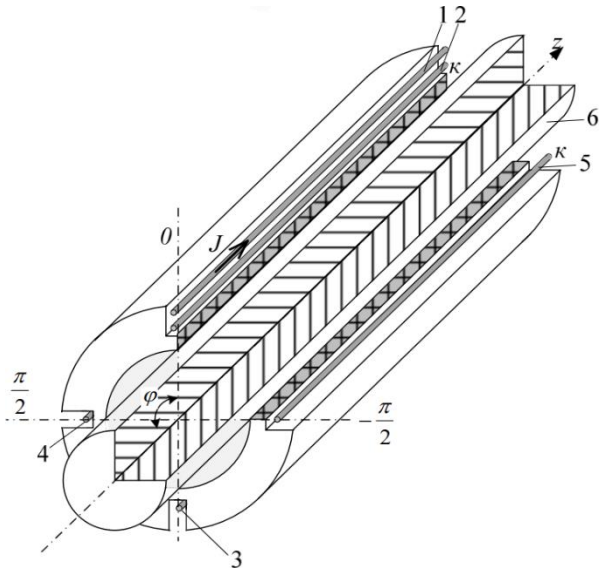


Рис. 1.2. Розташування котушок збудження вздовж металевого циліндра, де 1 - провідник збудження, 2-5 - вимірювальні провідники, 6 - ОК.

Автори пропонують використовувати специфічні гармоніки сигналу для визначення та виділення відхилень параметрів структури матеріалу ОК. Цей метод є досить вимогливим до якості сигналу самого ВСП, оскільки повздовжнє покриття циліндричного ОК збуджувальною системою робить його чутливість, а саме визначення локальних відхилень електрофізичних параметрів ОК, менш точною.

В наступній роботі [5] пропонується вимірювання параметрів ЕП та МП із використанням поздовжніх провідників за спеціальною методологією зустрічного та паралельного включення струмів збудження та комбінацій позиціонування вимірювальних та збуджуючих обмоток (рис.1.3). Метод є ефективним та досить перспективним в автоматизованих системах контролю, але вимога забезпечення комбінацій точного позиціонування котушок для системи вимірювання є недоліком, що призводить до ускладнення практичного застосування таких вимірювальних перетворювачів. Також для отримання інформативних результатів при спрощених конфігураціях запропонованих ВСП, а саме зменшення кількості вимірювальних та збуджуючих котушок, є необхідність проведення серії вимірювань з додатковим переміщенням ВСП навколо ОК (рис.1.4), що є небажаною технологічною операцією при впровадженні цього методу в промисловості. Крім того, можливими є похибки просторового позиціювання вимірювальних обмоток, що призводить до додаткових похибок вимірювання.

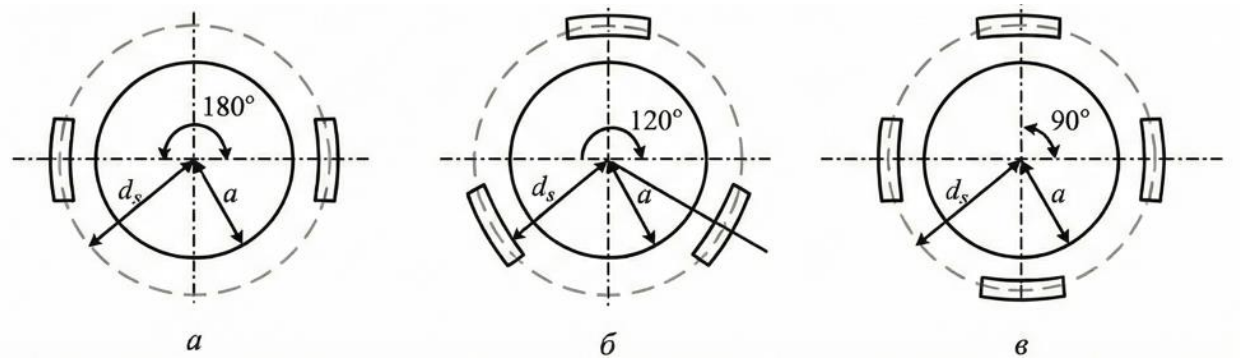


Рис. 1.3. Поперечний переріз намагнічувальних систем із однаковими за величиною струмами одного і того ж напряму.

В публікаціях [6]-[8] вирішення проблеми вихрострумового контролю товщини оболонок виробів та їх захисних покриттів, а також підвищення точності вимірювання контрольованих величин пропонується внаслідок впровадження автоматизованих систем з

використанням зворотної функції перетворення та розрахунку її наближеного значення. Автори окреслюють основні принципи та вимоги, яким має відповідати апаратура контролю параметрів ОК, і показують конкретний приклад реалізації основних елементів такого вимірювального приладу. Так як задача контролю параметрів ОК є оберненою і нелінійною, модель зворотної функції перетворення представлено багатовимірним поліномом у базисі інформаційних параметрів. Відповідно при використанні прямої функції перетворення відшукується розв'язок системи нелінійних рівнянь відносно невідомих параметрів.

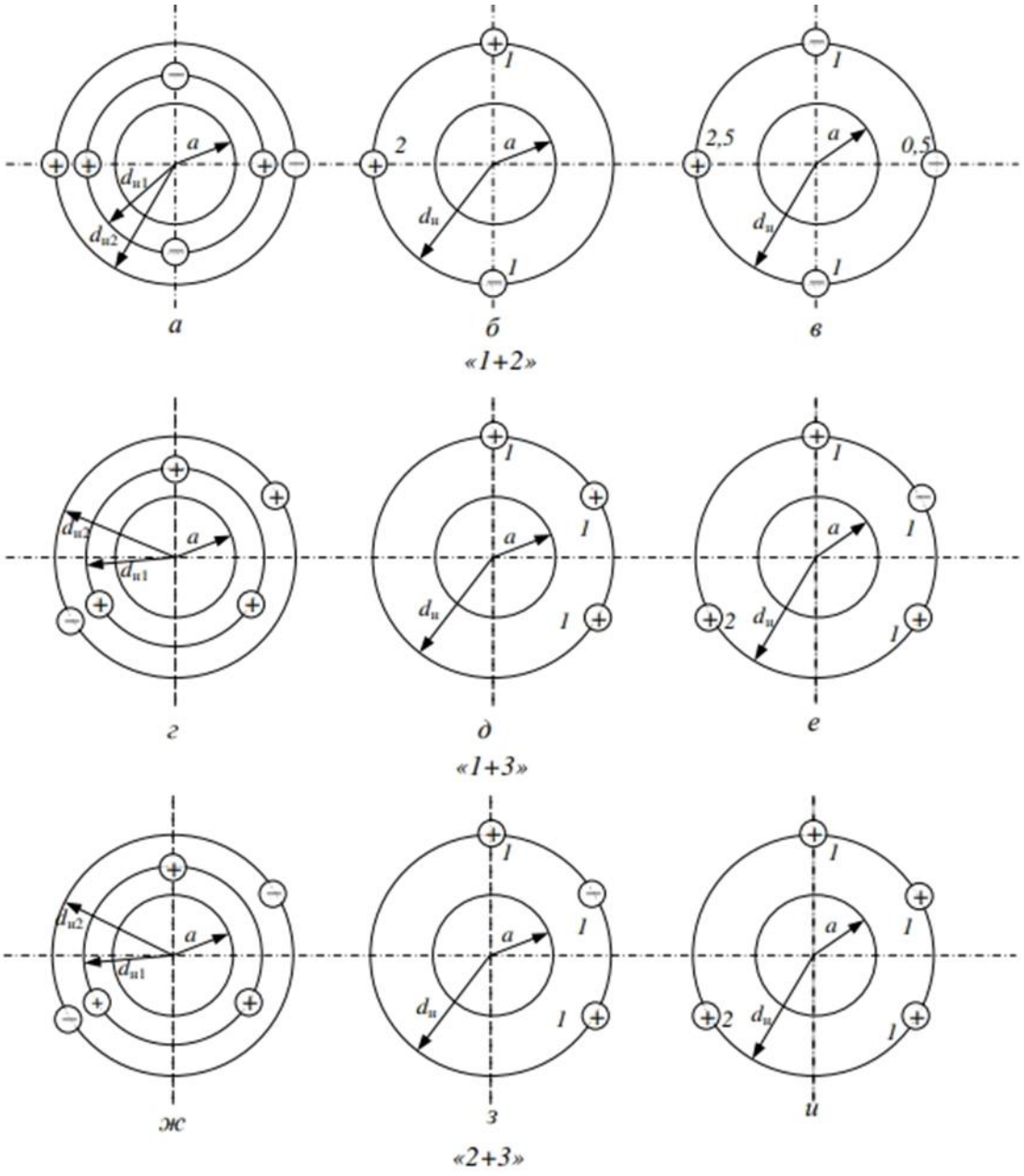


Рис. 1.4. Комбінації намагнічувальних систем перетворювачів із сумуванням переважаючих гармонік зондуючого поля.

У випадку використання зворотної функції для визначення параметрів ОК застосовується знайдена нелінійна поліноміальна залежність від компонент вектору інформаційних параметрів перетворювача. Зазначений вище метод може бути застосований в масштабі реального часу та потребує відносно малих обчислювальних ресурсів на обрахунок результату. До недоліків методу слід віднести певні труднощі вибору структури полінома, що апріорі є невідомою, яка би забезпечила прийнятну похибку апроксимації гіперповерхні відгуку. Також відзначимо, що з ростом числа невідомих параметрів ОК (наприклад, додатково повітряного зазору, товщини ОК), а відповідно розмірності гіперпростору, провести поліноміальну апроксимацію стає практично неможливим, а спрощена поліноміальна функція в свою чергу зменшує точність результату.

Хоч в статті [9] вирішується задача дефектоскопії, однак в цій роботі показано один з цікавих підходів розв'язання оберненої задачі досліджуваного фізичного процесу. У ході дослідження було використано ANSYS Parametric Design Language (APDL), що застосовує метод скінченних елементів для моделювання процесу розпізнавання дефектів накладним перетворювачем на основі виміру імпедансу котушки як характерної ознаки розпізнавання дефекту. За допомогою APDL автори створили модель, при якій імпеданс вимірювальної котушки в різних позиціях може бути обрахований автоматично. Дослідження порівнює результат отриманий за допомогою методу скінченних елементів і його апроксимаційний за допомогою штучних нейромереж (ШНМ) варіант. Обернена задача розпізнавання форми дефектів була вирішена навчанням ШНМ. Використання методу скінченних елементів є досить ресурсозатратним для формування навчальної вибірки для ШНМ, проте загалом підхід до розв'язання оберненої задачі є досить точним та відносно продуктивним і може бути застосований з певними модифікаціями в режимі реального часу.

В дослідженні [10] показано приклади виміру провідності матеріалу промисловим приладом вихрострумового контролю. Автори провели калібрування приладу і ряд експериментів на різних частотах і з різною кількістю вимірів на реальних зразках. Багатошаровий контрольний зразок був виготовлений накладанням металевих пластин з різною провідністю одну на одну. Наведені

результати досліджень та приведені графіки залежності від частоти струму збудження при вихрострумовому контролі товщини матеріалу та провідності шарів. Результати обчислення провідності матеріалу отримані на основі наближених значень каліброваних вимірів різних зразків реальних ОК. Варто відзначити проведення тестів на реальних зразках та різноманітність матеріалів для формування шарів ОК. Для розв'язку оберненої задачі такий підхід є важко реалізуємим та дуже затратним як по ресурсам, так і по часу.

Стаття [11] присвячена вимірюванню провідності пласких об'єктів накладним ВСП із застосуванням імпульсного збудження. Наведені результати експериментів вимірювання провідності листів вуглецевої сталі цим методом. Визначено переваги вимірювання провідності феромагнітних матеріалів імпульсним методом. Також створена аналітична модель для такого виду контролю. Показані результати впливу зазору між ОК і ВСП на вимірювання. В цій роботі описані експерименти тільки з феромагнітними матеріалами щодо визначення тільки одного електрофізичного параметру, а саме ЕП.

В наступній роботі [12] головною задачею вихрострумового неруйнівного контролю було визначення товщини труб, але підходи до вимірювання і проведення комп'ютерного моделювання є перспективними для використання щодо встановлення матеріальних властивостей матеріалу приповерхневого шару ОК. Автори створили модель в COMSOL Multiphysics (2D-модель) для реалізації їхнього концепту в подальших тестах за допомогою побудованої дослідницькою установки. Проведенням ряду вимірювань та використанням оптимізаційного методу отримано значення товщини матеріалу ОК. Розглядався однопараметровий контроль. Так як було використано оптимізаційні засоби розв'язання задачі, це робить отримання результатів в реальному часі важко реалізуємим.

Дослідження [13] описує процес контролю провідності вуглецевого волокна за допомогою вихрострумового перетворювача спеціальної конструкції. Він сформований з двох контурів і 12-ти сегментів прямокутних котушок в кожному контурі (рис.1.5), розташованих по колу так, що в них формується магнітне поле, що обертається, без необхідності обертати сам перетворювач

(рис.1.6). Це надає зручності та надійності у промисловому використанні. Чутливість таких перетворювачів досить висока і дозволяє розрізняти вид і товщину вуглецевого волокна. Обернена задача знайдення провідності за результатами вимірювання імпедансу котушок розв’язувалася за допомогою ШНМ, а точніше багатошаровим персептроном. Набір даних формувався внаслідок розв’язання прямої задачі за допомогою комплексу програм на мові Python та бібліотек для побудови 3D-моделей методом скінченних елементів FEniCS, Gmsh, бібліотек візуалізації ParaView.

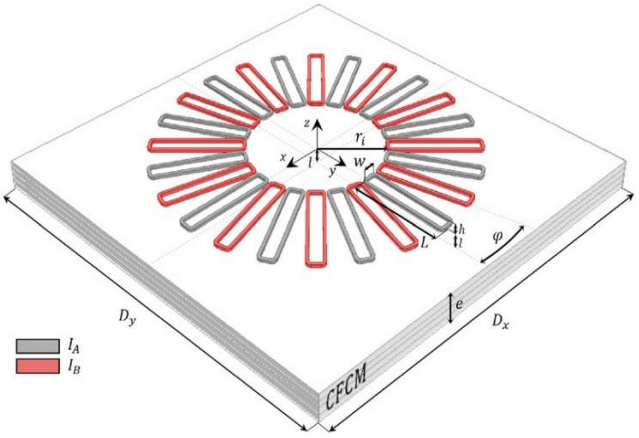


Рис. 1.5. Модель досліджуваної системи.

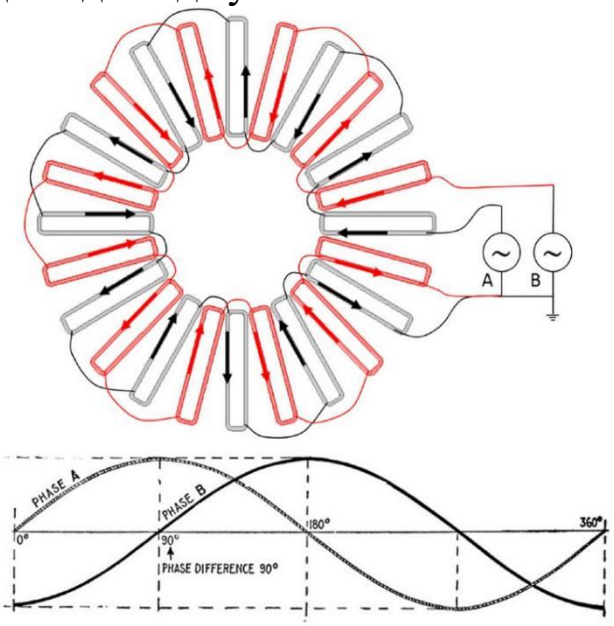


Рис. 1.6. Система збудження ВСП.

В дослідженнях [14], [15] показано розв’язання оберненої задачі встановлення профілів провідності пласких багатошарових об’єктів

за сигналом перетворювача, а саме імпедансу котушки (рис.1.7). Як засіб для розв'язання оберненої задачі також використовувались ШНМ. Для формування вибірки була використана аналітична модель процесу вихрострумового контролю плаского провідника накладним циліндричним перетворювачем. В дослідженнях висвітлюється проблема ефективного вибору частоти струму збудження. В цій роботі розв'язання задачі визначення контрольованих параметрів є лише однопараметровим.

В роботі [16] описується варіант розв'язання оберненої задачі визначення ступінчастих змін провідності матеріалу пласких багатошарових об'єктів симплекс-методом. В цьому дослідженні аналітична модель прямої задачі була спрощена з подальшим використанням тільки фази сигналу перетворювача як інформативного показника.

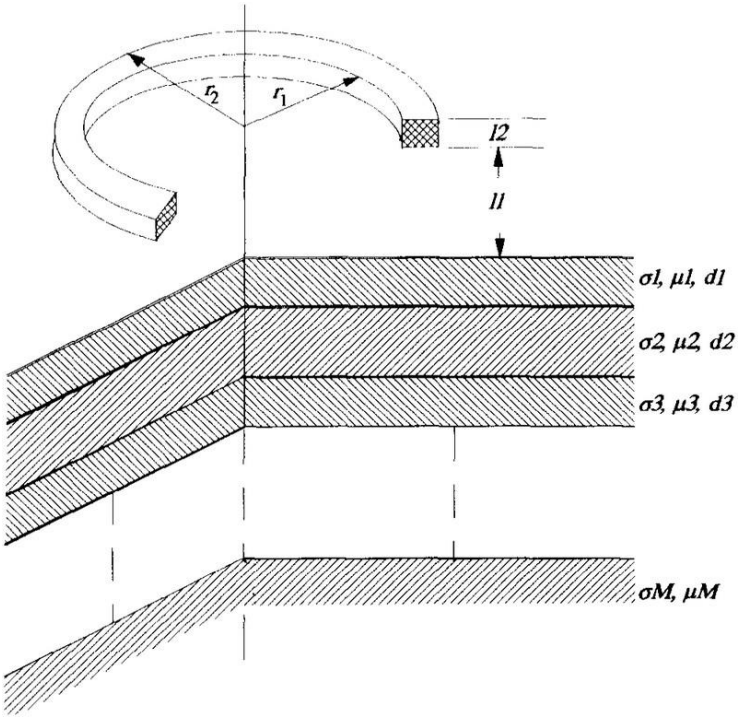


Рис. 1.7. Котушка над пласким багатошаровим об'єктом з М – шарів.

Отже, в наведених науково-технічних джерелах можна спостерігати схильність дослідників застосовувати різноманітні методи розв'язання прямих та обернених задач вихрострумових вимірювань. В більш простих випадках задач можливим є застосування спрощених оптимізаційних технік та апроксимацій для отримання адекватних моделей, але зі збільшенням розмірності

вхідних даних у математично некоректно поставлених задачах такі підходи стають малоефективними, призводять до збільшення похибки розв'язання і нелінійно суттєвого збільшення ресурсозатратності.

Проблема знаходження варіантів компромісу між точністю моделей та їх ресурсоємністю залежить від конкретного випадку, цілей і задач, які ставляться перед дослідниками. Можна зазначити, що сучасні тенденції розвитку обчислювальних технік спрямовані до застосування високоефективних і відносно точних в цьому плані штучних нейронних мереж. На сьогодні більшість підходів до вихрострумової ідентифікації матеріальних властивостей ОК є або вузькоорієнтованими, або неефективними і складними для реалізації через неможливість їх застосування в промисловому виробництві реальному часі, необхідності великої кількості вимірювань, додаткових ресурсозатратних технологічних операцій.

1.2. Огляд методів створення сурогатних моделей із накопиченням апріорних даних

Зазвичай розв'язання обернених задач в багатьох сферах науки та техніки потребують застосування оптимізаційних методів, що використовують цільові функції. Цільові функції в таких випадках часто обчислюються за допомогою досить «важких» в сенсі затрат часу чисельних методів, що призводить до практично непереборних перешкод. Заміна ресурсоємної цільової функції її апроксимованим аналогом, тобто сурогатною моделлю (метамоделлю або моделлю-замісником) (рис.1.8), яка відрізняється значно більшою обчислювальною продуктивністю, дає можливість пошуку розв'язку оптимізаційної задачі за реальний час [17].

Останніми роками такий підхід зі створенням сурогатних моделей застосовується в різноманітних галузях для вирішення складних проєктних завдань: в машинобудуванні [18], аерокосмічній промисловості [19], турбінобудуванні, будівництві [20]. Отже, під метамоделлю або сурогатною моделлю розуміють просту в обчислювальному сенсі формальну модель на складнішу модель, побудовану на фізичних законах, тобто вона є моделлю на модель. Загалом задача побудови сурогатної моделі зводиться до побудови апроксимаційної функції гіперповерхні відгуку, що визначається моделлю на фізичних законах. Це не проста задача, що

іноді, в складних випадках, потребує застосування комбінованих методів апроксимації, які поєднують в собі методи штучного інтелекту і традиційні математичні методи наближення та аналізу даних [21].

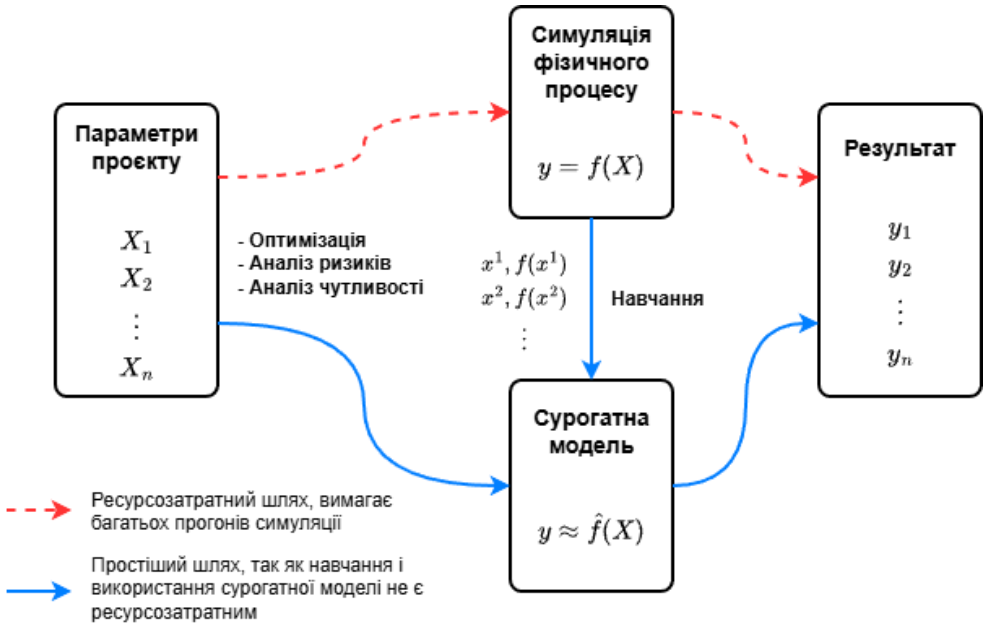


Рис. 1.8. Концепція сурогатного моделювання.

Наявні різноманітні методи побудови сурогатних моделей, які використовуються науковцями, можна класифікувати варіантом зображеним на рис.1.9, де MARs (Multivariate adaptive regression splines) - багатоваріантні адаптивні регресійні сплайни; Cubic splines - кубічні сплайни; NURBs (Non-uniform rational B-splines) - неоднорідні раціональні B-сплайни; ANN (artificial neural networks) - штучна нейронна мережа; RBF-ANN - нейронна мережа на радіально-базисних функціях; MLP-ANN - багатошаровий персептрон; SVM - метод опорних векторів; GMDH - метод групового врахування аргументів. При цьому можна виділити узагальнені класи сурогатних моделей, а саме геометричних, стохастичних та евристичних, та відповідних методів їх побудови [22]. Слід зазначити, що методи створення сурогатних моделей, які використовуються для технічних задач, відрізняються різноманітними підходами до апроксимації та складністю їх реалізації.

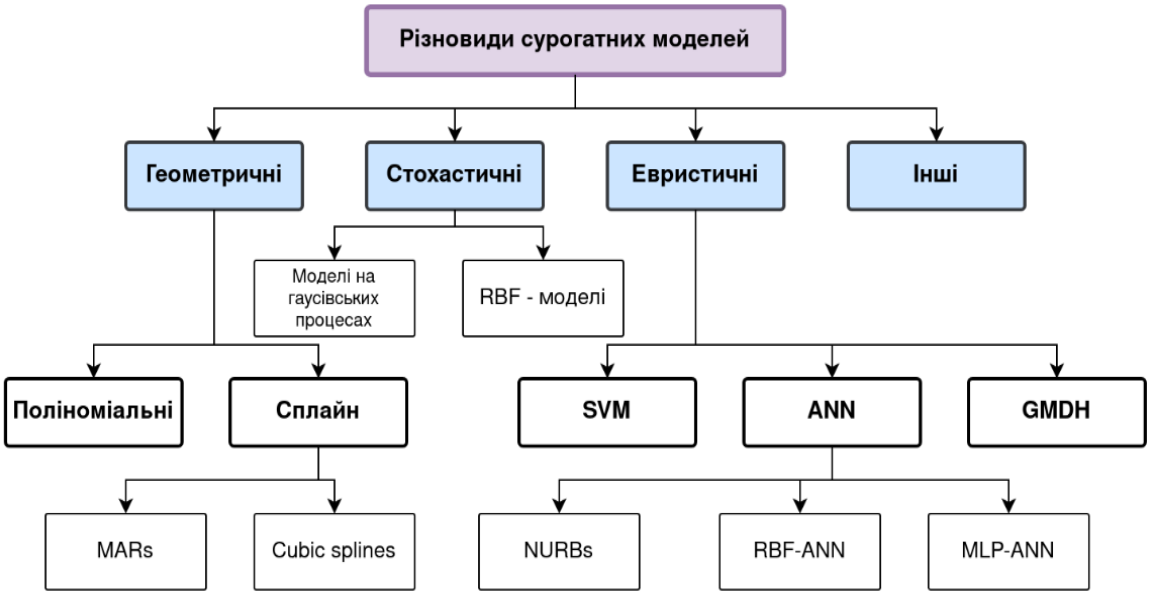


Рис. 1.9. Різновиди методів апроксимації, що застосовуються для побудови сурогатних моделей.

1.2.1. Геометричні сурогатні моделі

Згідно з наведеною класифікацією до першої групи методів побудови сурогатних моделей відносять геометричні, до яких належать всі види поліноміальних моделей та сплайн-моделі. Поліноміальні моделі, як одні з найпоширеніших, застосовуються науковцями для розв'язання різноманітних задач [23], [18]. Поліноміальна модель отримується, на відміну від лінійної, внесенням додаткових предикторів шляхом піднесення кожного початкового предиктора до певного степеня. Залежність між предикторами і відгуком описується в загальному вигляді регресійною функцією:

$$y = b_0 + b_1 \cdot f_1(x_1, x_2, \ldots, x_n) + \ldots + \\ + b_{m-1} \cdot f_{m-1}(x_1, x_2, \ldots, x_m) + \varepsilon, \qquad (1.1)$$

де $f_i(x_1, x_2, ..., x_n)$, $i = 1, ..., m\text{-}1$ - задані функції факторів $x_1, x_2, ..., x_n$;

$b_0, b_1, ..., b_{m-1}$ - коефіцієнти математичної моделі;

ε - залишок або випадкова складова.

В залежності від вибраної функції $f_1(x_1, x_2, ..., x_n)$ регресійна модель може бути представлена по-різному. Наприклад, для багатофакторної поліноміальної регресії функція

має вигляд (1.1), а поліноміальна модель 2-го порядку для трьох факторів містить головні ефекти (тобто ефекти першого порядку) та квадратичні ефекти (тобто ефекти другого порядку) і має вигляд:

$$y = b_0 + b_1 \cdot x_1 + b_2 \cdot x_1^2 + b_3 \cdot x_2 + b_4 \cdot x_2^2 + b_5 \cdot x_3 +$$

$$+ b_6 \cdot x_3^2 + \varepsilon.$$

Коефіцієнти математичної моделі оцінюються методом найменших квадратів. Зазвичай степінь поліному більше ніж 3 або 4 не використовується, оскільки в такому випадку поліноміальна крива стає надмірно гнучкою і приймає неадекватний вигляд. Класичний регресійний аналіз передбачає виконання низки передумов [24], які насправді можуть не виконуватися, а їх перевірка достатньо складна, оскільки передбачає проведення складних експериментів і вимагає значних ресурсів на її здійснення. В поліноміальних моделях в залежності від складності гіперповерхні відгуку завжди виникає проблема вибору порядку моделі, яка на практиці вирішується ітеративним методом в бік поступового підвищення з метою уникнення перенасичення. Одним з методів розв’язання регресійних задач є застосування багатовимірних адаптивних сплайнів MARs, що дозволяє встановлювати вигляд і параметри апроксимаційної функції, з заданою точністю відтворюючої початкові дані [25]. Простір пошуку значень вхідних змінних розбивається на області, в яких використовуються різні базисні функції певних видів та їх добутки з декількох співмножників. В основі роботи методу покладено вибір необхідної зваженої суми базисних функцій з їх загального набору (словника). Алгоритм MARs-сплайнів шукає в просторі всіх вхідних змінних місця розташування вузлових точок, а також взаємозв’язки між змінними. Коефіцієнти розкладання і сам робочий набір базисних функцій вибираються за допомогою ітеративної евристичної процедури включення-виключення, що дає дещо меншу точність апроксимації, ніж за використання повного словника. Метод MARs-сплайнів знаходить шукану залежність у два етапи. Перший етап полягає в додаванні базисних функцій до робочого набору, доки не буде мінімізовано загальний критерій якості моделі [25] або ж буде досягнута максимальна кількість базисних функцій. На другому етапі з робочого набору видаляються функції, які не впливають суттєво на критерій точності моделі, що

відображає зростання дисперсії з ростом числа базисних функцій. Для розрахунку невідомих коефіцієнтів розкладання використовують метод найменших квадратів. Маючи деякі переваги перед класичними статистичними методами побудови апроксимаційної моделі, MARs все ж є чутливим до початкових вхідних даних. Крім того, метод має значні часові затрати на розрахунок коефіцієнтів моделі методом найменших квадратів у випадку розв'язання задач великої розмірності.

1.2.2. Стохастичні сурогатні моделі

Представником стохастичних методів побудови сурогатних моделей є регресія на основі гаусівських процесів (або kriging) [26], що дозволяє створювати нелінійні апроксимаційні моделі. Методи побудови сурогатних моделей на основі гаусівських процесів та їх застосування в задачах оптимізації розглядаються в роботах [27] і [28]. Відомо, що будь-який випадковий процес визначається середнім значенням та коваріаційною функцією. У разі використання реальних даних коваріаційна функція гаусівського процесу невідома. Тому вводиться припущення, що коваріаційна функція належить до деякого параметричного сімейства. В залежності від апріорних уявлень про вигляд апроксимаційної залежності вибирається сімейство коваріаційних функцій. Так в роботі [29] передбачалося, що коваріаційна функція належить до експоненціального сімейства, а в [30] - до сімейства на основі відстані Махалонобіса. Коваріаційна функція другого сімейства дозволяє створити модель більш загальну, проте обмежує роботу з даними великої розмірності, оскільки значно збільшується кількість параметрів коваріаційної функції, які необхідно оцінити. У зв'язку з цим в роботі [30] вирішується актуальне завдання розробки алгоритму налагодження параметрів коваріаційної функції на основі відстані Махалонобіса, який дозволяє виконувати ці дії і для випадку даних великої розмірності. У порівнянні методів у двовимірному випадку стандартний і запропонований авторами [30] алгоритми налагодження параметрів коваріаційної функції мають однакову точність. Зі збільшенням розмірності стандартний метод з використанням відстані Махаланобіса має суттєво меншу точність у порівнянні із запропонованим алгоритмом повороту координатних осей [30], оскільки збільшується кількість

гіперпараметрів, які необхідно визначати в процесі навчання. З метою зменшення обчислювальних затрат та підвищення точності оцінювання параметрів моделі, в роботі [29] пропонується метод моделювання нестаціонарної коваріаційної функції на основі лінійного розкладання по словнику параметричних функцій та використання байєсівської регуляризації. Для байєсівської регуляризації використовуються нормальний та гамма-розподіл параметрів. Обидва розподіли дозволяють уникати виродження апроксимації, збільшувати узагальнювальну здатність і надійність алгоритмів. Використовуючи метод, слід враховувати, що для побудови моделі на основі гаусівських процесів необхідно оцінити вектор параметрів коваріаційної функції. Регресія на основі гаусівських процесів передбачає наявність заздалегідь заданої коваріаційної функції, яка необхідна для оцінювання параметрів цих процесів, що відповідно впливає на обчислювальну складність методу. Розрахунок параметрів моделі виконується методом максимальної правдоподібності, який передбачає виконання досить громіздких матричних перетворень, що суттєво впливає на затрати часу зі збільшенням розмірності задачі (1.2). У разі, коли є великий масив вихідних даних, найчастіше використовуються регресійні RBF-моделі, які так само створюються застосуванням стохастичних методів побудови [31], [32]. Радіальна базисна функція апроксимації використовує лінійні комбінації K радіально-симетричних функцій φ:

$$f(\boldsymbol{x}) = \sum_{j=1}^{K} \lambda_{\boldsymbol{j}} \cdot \varphi\left(\left\| \boldsymbol{x} - c^{(j)} \right\| \right), \qquad (1.2)$$

де $\lambda = [\lambda_1, \lambda_2, ..., \lambda_k]^T$ - вектор параметрів моделі;

$\boldsymbol{x}$ - вектор проєктних змінних;

φ - базисна функція;

$c^{(j)}, j = 1, ..., k$ - відомі центри базисних функцій.

Параметри моделі λ визначаються аналогічно, як і у випадку поліноміальної регресії. Тобто функція апроксимації $f(\boldsymbol{x})$ є лінійною комбінацією деяких базисних функцій з відповідними ваговими коефіцієнтами. Найрозповсюдженіші базисні функції - лінійна, кубічна, полігармонічний сплайн, TPs-сплайни (Thin Plate splines або сплайн-поверхні) [31], [32]. Проте більшу гнучкість мають параметричні базисні функції, наприклад, гаусівська,

зворотна квадратична, мультиквадратична, зворотна мультиквадратична та інші [31]. Розрахунок вектору параметрів моделі виконується методом найменших квадратів, що в задачах великої розмірності вимагає суттєвих обчислювальних ресурсів.

1.2.3. Евристичні моделі

Розглядаючи клас евристичних сурогатних моделей, можна виділити моделі з використанням методу групового урахування аргументів GMDH, методу опорних векторів (SVM) та штучних нейронних мереж (ANN).

Створення GMDH-сурогатних моделей засновано на сортуванні поступово ускладнених варіантів моделей з вибором їх оптимальної структури [33], [34]. Цей метод має переваги, коли відсутня або майже відсутня апріорна інформація про структуру моделі і розподіл її параметрів. Ідея методу полягає у формуванні за даними вибірки деякої множини моделей $\hat{y}_f$ різноманітної структури, що мають такий вигляд:

$$\hat{y}_f = f\left(\left\| X, \hat{\theta}_f \right\|\right),$$

де X - матриця значень змінних, що утворює вибірку спостережень;

$\hat{\theta}_f$ - оцінка параметрів моделі.

Далі з отриманої множини моделей $\hat{\hat{y}}_f$ визначається оптимальна модель f^* за критерієм мінімуму оцінки якості моделі C:

$$f^* = \arg\min C\left(\left\| y, \hat{y}_f \right\|\right),$$

де y - вектор вихідних значень у вибірці даних.

Оцінка параметрів для кожної моделі є також розв’язанням ще однієї екстремальної задачі. Зв’язок між вхідними і вихідними змінними описується у вигляді функціонального ряду Вольтерра, дискретним аналогом якого є узагальнений поліном Колмогорова-Габора:

$$\hat{y}_f = a_0 + \sum_{i=1}^{n} a_i \cdot x_i + \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij} \cdot x_i \cdot x_j + \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} a_{ijk} \cdot x_i \cdot x_j \cdot x_k ,$$

де a_0, a_i, a_{ij}, a_{ijk} - коефіцієнти поліному;

x_i, x_j, x_k - вхідні змінні;

n - кількість вхідних змінних.

Існує чотири базових алгоритми GMDH та велика кількість їх модифікацій: COMBI - комбінаторний алгоритм, MULTI - комбінаторно-селекційний алгоритм, MIA - багаторядний ітеративний алгоритм, RIA - релаксаційний ітеративний алгоритм. Всі базові алгоритми є багаторядними. В кожному ряді може знаходитися декілька моделей, які характеризуються однаковим рівнем складності. Алгоритми відрізняються між собою умовами формування і відбору змінних при переході від одного ряду до іншого. Ідея COMBI полягає в використанні всіх можливих моделей без їх пропуску, саме тому на кожному рівні складності розглядаються всі моделі і не проводиться селекція кращих комбінацій змінних. Проте його практичне застосування обмежено задачами з невеликою кількістю ознак n, оскільки кількість можливих комбінацій експоненціально збільшується зі збільшенням кількості змінних моделей. Тому в цьому алгоритмі кількість змінних обмежена, $n = 20$. Ідея алгоритму MULTI - зменшити кількість моделей, що розглядаються в кожному ряді, без втрати кращої комбінації змінних. На кожному рівні складності відбирається фіксована кількість кращих поєднань змінних моделі, а потім кращі поєднання комбінуються з усіма іншими змінними по черзі при переході на наступний рівень. Алгоритм MIA реалізує ідею зменшення кількості моделей, які розглядаються в кожному ряді, зі зменшенням кількості рядів, що дозволяє прискорити забезпечення оптимального рівня складності. На кожному ряді відбирається фіксована кількість кращих моделей, а потім кожна пара цих кращих моделей породжує нову змінну при переході на наступний рівень. Кількість кращих моделей та функцію перетворення необхідно задавати апріорі. Попри високу продуктивність цей алгоритм здатен пропускати оптимальні рішення та зі збільшенням кількості рядів характеризується різким ускладненням моделі, тоді як значення зовнішнього критерію якості при цьому зменшується несуттєво. Серед основних переваг

алгоритму GMDH [35], [36] можна виділити те, що за його допомогою для коротких, неточних або зашумлених даних може бути знайдена оптимальна нефізична модель. При цьому точність і структура такої моделі є прийнятними для подальшого застосування. Принциповим недоліком таких параметричних алгоритмів GMDH є необхідність оцінювання параметрів підсумкової моделі, які можуть бути зміщеними; додаткові витрати часу на пошук ефективного виду моделі. Також для задач нелінійної регресії використовується потужна і універсальна модель машинного навчання - так званий метод опорних векторів SVM (support vector machine) [37]. Цей метод передбачає створення набору гіперплощин в багатовимірному або нескінченному просторі, які використовуються для розв'язання задач регресії. Алгоритм SVM спочатку розроблявся для лінійної і нелінійної класифікації. Частіше за все використовується модель із застосуванням класу SVR з бібліотеки Scikit-Learn, яка підтримує ядровий трюк (kernel trick) і дозволяє керувати балансом між шириною смуги поміж класами та обмеженням кількості порушень зазору, використовуючи гіперпараметр *C*. Ядровий трюк передбачає додавання додаткових поліноміальних ознак до набору даних, що робить можливим реалізацію нелінійної класифікації в результаті переходу до нового простору ознак. На практиці використовуються поліноміальні і гаусівські ядра. Безпосередньо для задач регресії використовується параметрично редукована (kernelized) модель SVM. Під час розв'язку задач регресії SVM-методом застосовується ефективний математичний прийом. Його ідея полягає в інвертуванні мети: замість спроби пристосуватися до найширшої з можливих смуг між класами, одночасно обмежуючи порушення зазору, регресія SVM намагається помістити якомога більше зразків даних на смузі разом з обмеженням порушення зазору (тобто зразків поза смугою). Ширина смуги керується гіперпараметром ε. Розповсюджений підхід з пошуку доцільних значень гіперпараметрів полягає у використанні пошуку на гратці. Метод SVM найкраще підходить для невеликого і середнього наборів даних. На великих навчальних вибірках цей метод апроксимації характеризується великими обчислювальними затратами.

Також потужним апаратом для апроксимації складних залежностей є штучні нейронні мережі (НМ) [38], а саме НМ на

радіально-базисних функціях RBF-ANN та багатошарових персептронах MLP-ANN (рис.1.10). Універсальні апроксимаційні властивості НМ та відсутність вимог попереднього «точного, жорсткого» задання вигляду моделі є причиною їх широкого застосування при створенні сурогатних моделей у складних випадках топології гіперповерхні відгуку [39].

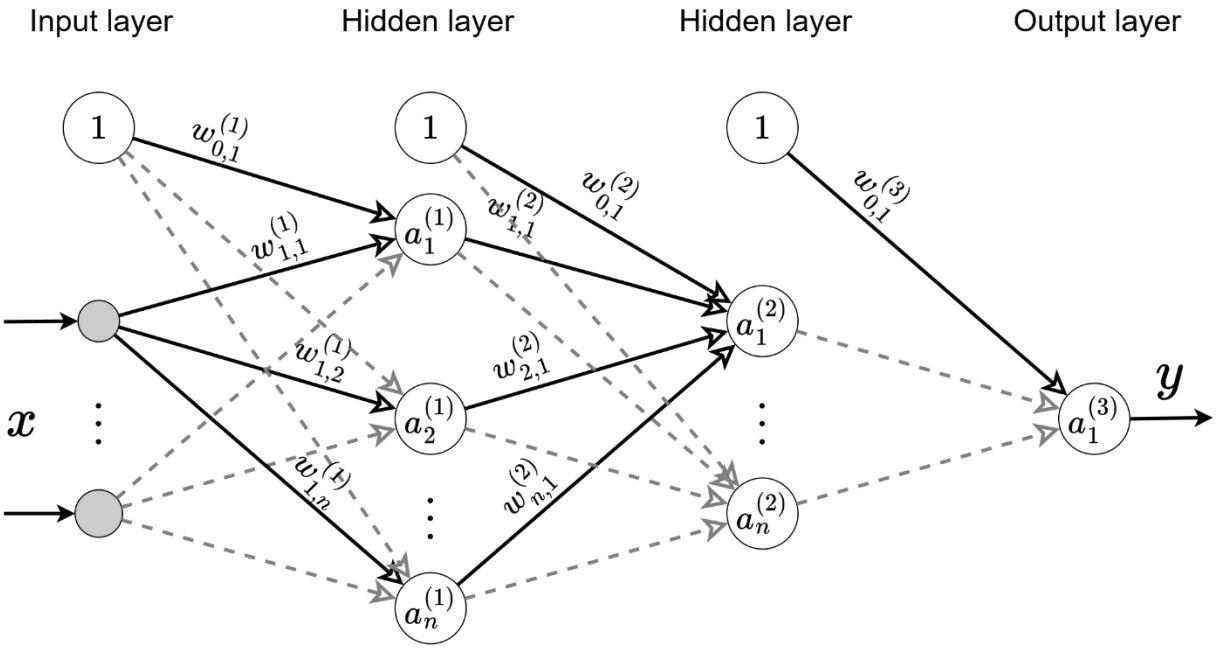


Рис. 1.10. Структура регресійної двошарової дійснозначної штучної MLP-ANN.

Відсутність «жорстко» заданого апріорного зв'язку шуканого розв'язку з конкретною моделлю надає переваги НМ, оскільки вона виявляється більш пристосованою до роботи в умовах невизначеності. Основні переваги НМ головним чином зумовлені: навчанням на прикладах; підвищенням завадостійкості до зашумлених та суперечливих даних; стійкістю до помірних змін побічних параметрів об'єкта, які не є шуканими в процесі розв'язку. Разом з тим їм притаманні і деякі недоліки, один з яких це відносно великий час для навчання мережі та відсутність аналітичного запису отриманої апроксимаційної функції. В класифікації нейронних мереж виділяють два фундаментальні класи: мережі прямого розповсюдження (одношарові та багатошарові) та рекурентні мережі або мережі зі зворотним зв'язком [40], але практичне застосування в апроксимаційних задачах знайшли перші з них. Завдяки великій кількості алгоритмів і методик навчання та багатьох видів функцій активації досягається

створення великого розмаїття НМ. Теоретичною основою і обґрунтуванням того, що НМ здатна апроксимувати будь-яку функціональну залежність є теорема Колмогорова-Арнольда про універсальну апроксимацію [38]. Будь-яка неперервна функція K аргументів в одиничному кубі $[0, 1]^K$ може бути представлена у вигляді суперпозицій неперервних функцій одного аргументу і операції додавання (1.3).

$$F\left(x_1, x_2, \ldots, x_K\right) = \sum_{i=1}^{N} a_i \cdot f\left(\sum_{k=1}^{K} a_{ik} \cdot x_k + a_{0k}\right), \qquad (1.3)$$

де $(x_1, x_2, ..., x_K)^T$ - вектор вхідних даних;

$f(\cdot)$ - обмежена непостійно монотонно висхідна неперервна функція;

K - кількість вхідних вузлів;

N - кількість нейронів прихованого шару;

a_{ik} - синаптичні ваги прихованого шару;

a_i - синаптичні ваги вихідного шару;

a_{0k} - зміщення.

В основі побудови RBF-мереж покладено розбиття простору пошуку гіперсферами, які задаються своїм центром та радіусом. RBF-мережа має у своєму складі: вхідний шар, що з'єднує мережу з середовищем спостереження; прихований шар (або проміжний), що складається з елементів з ядерними базисними функціями активації; лінійний вихідний шар - звичайний одношаровий персептрон, який в результаті налаштування ваг визначає вихід мережі [38], [40]. В якості функції активації нейронів прихованого шару часто використовують функцію Гауса, але можливі й інші різновиди функцій, наприклад, квадратична ядерна функція, ядро Єпанечникова, зворотна мульти-квадратична функція, сплайн-функція, функція Коші. Із застосуванням гаусівської функції активації вихід мережі формується як лінійна комбінація виходів нейронів прихованого шару і описується виразом:

$$u\left(\vec{x}\right) = \sum_{k=1}^{m} w_k \cdot \varphi_k\left(\vec{x}\right) = \sum_{k=1}^{m} w_k \cdot \exp\left(-\frac{r_k^2}{a_k^2}\right), \qquad (1.4)$$

де $\vec{x}$ - вхідний вектор $\vec{x} = \left(x_1, x_2, \ldots x_l\right)^T$;

$r_k = \|\vec{x} - \vec{c}_k\| = \sqrt{\left(x_1 - c_{x_1 k}\right)^2 + \left(x_2 - c_{x_2 k}\right)^2 + \ldots + \left(x_l - c_{x_l k}\right)^2}$ - радіус k-го нейрона;

l - кількість змінних цільової функції;

m - кількість нейронів прихованого шару;

w_k - вага зв'язку вихідного нейрона з k-м нейроном прихованого шару;

$\vec{c}_k$ - вектор координат центру k-го нейрона, який містить координати $(c_{x_1 k}, c_{x_2 k}, \ldots, c_{x_l k})^T$;

a_k - ширина k-го нейрона;

$\varphi_k(\vec{x})$ - гаусівська функція активації прихованого шару.

Задача апроксимації RBF-мережею зводиться до оптимального вибору ваг вихідного шару, кількості радіальних функцій (нейронів), а також їх параметрів: центрів розташування цих функцій та їх ширини, які є нелінійними параметрами прихованого шару. До переваг нейронних RBF-мереж відносять те, що вони мають лише один прихований шар нейронів, який істотно спрощує характерну для складніших НМ задачу вибору кількості прихованих шарів і робить цей вибір визначеним. Також ці мережі швидко навчаються, що зумовлено можливістю застосування добре вивчених методів лінійної оптимізації при підборі параметрів лінійної комбінації у вихідному шарі мережі. Але розмірність RBF-мереж експоненційно зростає зі збільшенням розмірності вихідних даних, а при їх навчанні є необхідність використання великої кількості прикладів. Крім того, RBF-мережа не здатна до екстраполяції даних при збільшенні ширини діапазону вхідних даних [39].

Багатошаровий персептрон прямого розповсюдження складається з множини сенсорних елементів (вхідних вузлів та вузлів джерела), які утворюють вхідний шар; одного або декількох прихованих шарів обчислювальних нейронів та одного вихідного шару. Вхідний сигнал розповсюджується мережею в прямому напрямку, від шару до шару. Навчання з «учителем» такої мережі виконується за допомогою алгоритму зворотного розповсюдження помилки, який ґрунтується на корекції похибок. В загальному випадку це відповідає популярному алгоритму адаптивної фільтрації — алгоритму мінімізації середньоквадратичної похибки. Існують також і інші алгоритми навчання MLP-мереж, які

використовують різноманітні стратегії найшвидшого просування до точки мінімуму [38], [40], наприклад, спуск по спряженим градієнтам і метод Левенберга-Марквардта.

Багатошарові персептрони мають декілька відмінних ознак:

- кожен нейрон мережі має переважно гладку функцію активації, наприклад, гіперболічного тангенса. Найбільш розповсюджена форма такої функції є сигмоїдальна та ReLU, яка визначається логістичною функцією. Теоретично доведено, що з використанням таких найпростіших перетворень можна наближувати достатньо складні багатовимірні функції, і, як наслідок, оцінювати складні залежності;
- мережа містить один або декілька шарів прихованих нейронів, які не належать до входу або виходу мережі. Ці нейрони дозволяють мережі навчатися розв'язувати складні задачі, послідовно враховуючи важливі ознаки з вхідного вектору;
- мережа має високий ступінь зв'язаності, що реалізується за допомогою синаптичних з'єднань. Зміна рівня зв'язаності мережі вимагає зміни множини синаптичних з'єднань або їх вагових коефіцієнтів.

Поєднання позитивних різноманітних властивостей MLP-мереж разом з урахуванням їх здатності до навчання забезпечує суттєву обчислювальну потужність регресійного багатошарового персептрона. Застосування MLP-мережі передбачає, крім вибору алгоритму навчання, необхідність застосування різноманітних методів оптимізації структури мережі для кожної конкретної задачі.

Для гіперповерхонь відгуку складної топології має сенс вживання технології декомпозиції областей пошуку та технік асоціативних машин. А рішення з вибору остаточної архітектури НМ може бути прийнято тільки після повного циклу навчання різноманітних варіантів їх структур та оцінки адекватності й інформативності отриманих сурогатних моделей за сукупністю статистичних показників.

Так як евристичні сурогатні моделі на основі ШНМ не потребують використання значних обчислювальних ресурсів та успішно виконують функції моделі-замісника — це є одним з найкращих варіантів вибору при розв'язку обернених задач.

Особливої уваги заслуговують глибокі ШНМ (DNN, DANN), вони забезпечують високу швидкодію та точність при необхідності

апроксимації складних нелінійних функцій з великою кількістю входів.

В рамках цього дослідження сурогатна модель може використовуватися дещо інакше. В задачах ідентифікації сурогатні моделі можуть виконувати функції накопичувачів апріорної інформації, яка отримана попередньо щодо досліджуваних об'єктів шляхом комп'ютерних чисельних експериментів, проведених за відповідними планами експериментів. Створені так апроксимаційні моделі, що є носіями апріорної інформації, можуть використовуватися безпосередньо у вимірювальних операціях, що дозволяє забезпечити розв'язання обернених задач в реальному масштабі часу.

1.3. Аналіз методів створення комп'ютерних однорідних планів експериментів

Як правило, для побудови адекватних та точних апроксимаційних моделей потрібні плани експерименту. Наприклад, для випадку, коли гіперповерхня відгуку заздалегідь не відома, щоб забезпечити точність апроксимації, потрібен однорідний комп'ютерний план експерименту.

Сурогатні моделі у вигляді штучних нейронних мереж відносяться до data driven методів, а це означає, що якість вибірки напряму впливає на якість відтворення результатів моделювання в подальшому.

Створення однорідного плану експерименту можливе за допомогою, наприклад, сукупності визначених в просторі точок, отриманих з використанням випадкових та квазі-випадкових послідовностей [41], [42], зокрема послідовності: Kronecker, Richtmyer, Ramshaw, Weyl, Van der Corput, Halton, Faure, Sobol, Niederreiter, R-послідовностей Робертса. Серед широко відомих квазі-випадкових послідовностей можна виділити послідовності Halton та Sobol.

На рис.1.11 зображено однорідні вибірки різних двовимірних розподілів. Як видно при звичайному випадковому розподілі точки схильні утворювати кластери і пробіли, а інші, тобто квазі-випадкові послідовності з низькою розбіжністю, є абсолютно визначеними послідовностями, що розподіляються по простору так рівномірно наскільки це можливо.

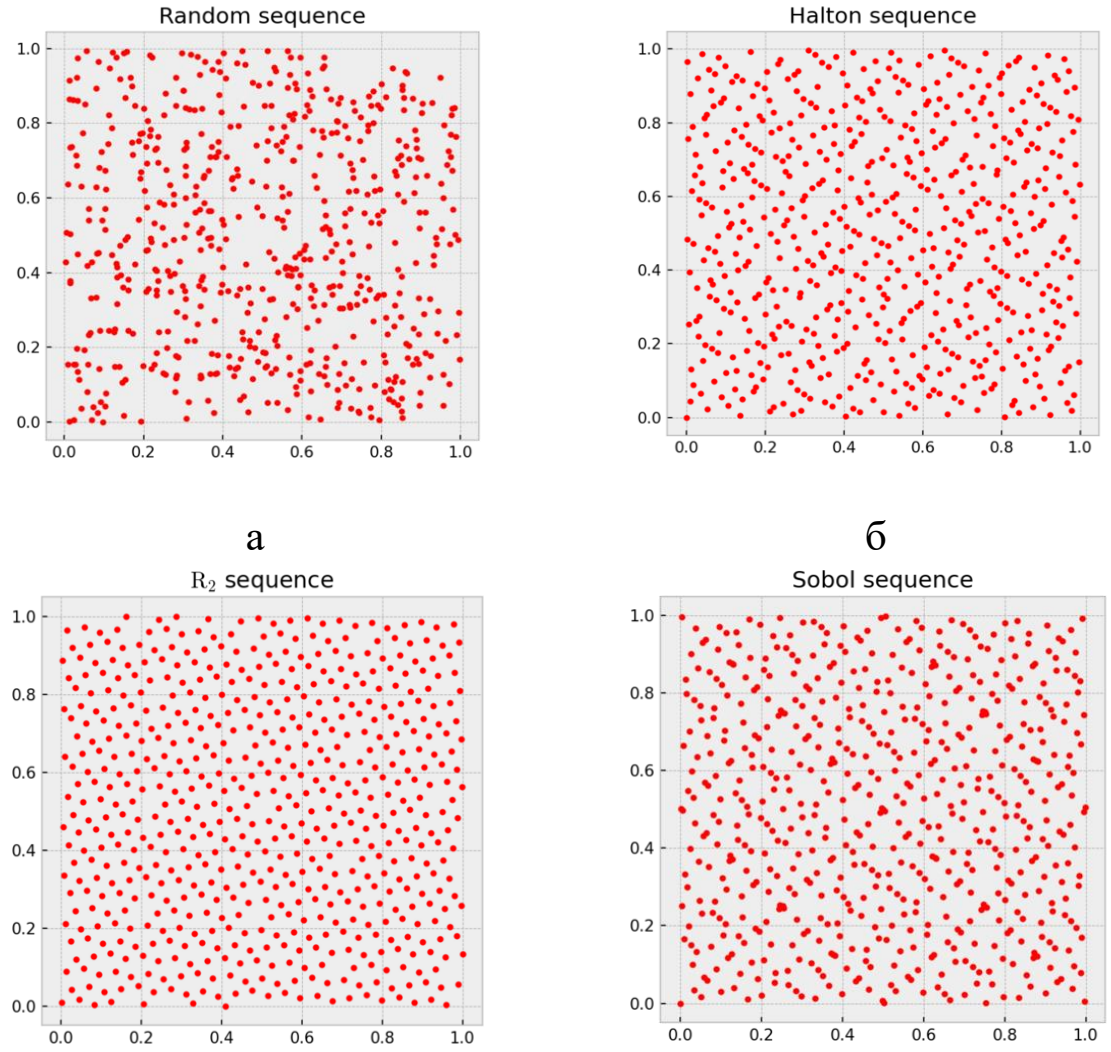


Рис. 1.11. Порівняння розподілів двовимірних квазі-випадкових та простої випадкової послідовностей.

При малій розмірності (одновимірному і двовимірному просторі) квазі-випадкові послідовності видаються менш однорідними ніж лінійне розбиття простору на однакові проміжки, але при збільшенні розмірності вони є найкращим вибором для формування вибірок.

Основними критеріями досконалості планів експерименту є оцінки гомогенності за показниками узагальнених розбіжностей, які кількісно характеризують відхилення згенерованого розподілу від ідеального рівномірного [43]. Окрім того, випадкові чи квазі-випадкові послідовності можуть бути оцінені за центрованою та циклічною розбіжністю, мінімальною дистанцією упаковки, візуально за діаграмами Вороного чи тріангуляцією Делоне, та оцінкою дискретного розміщення (Discretized Packing).

Хоча двовимірні Sobol-послідовності і відносяться до послідовностей з малим розходженням, проте існують і такі їх комбінації, що не демонструють хороших показників однорідності. Наприклад, комбінації які представлено на рис.1.12. Тому вибір «кращих» і «гірших» пар Sobol-послідовностей потребує додаткових досліджень.

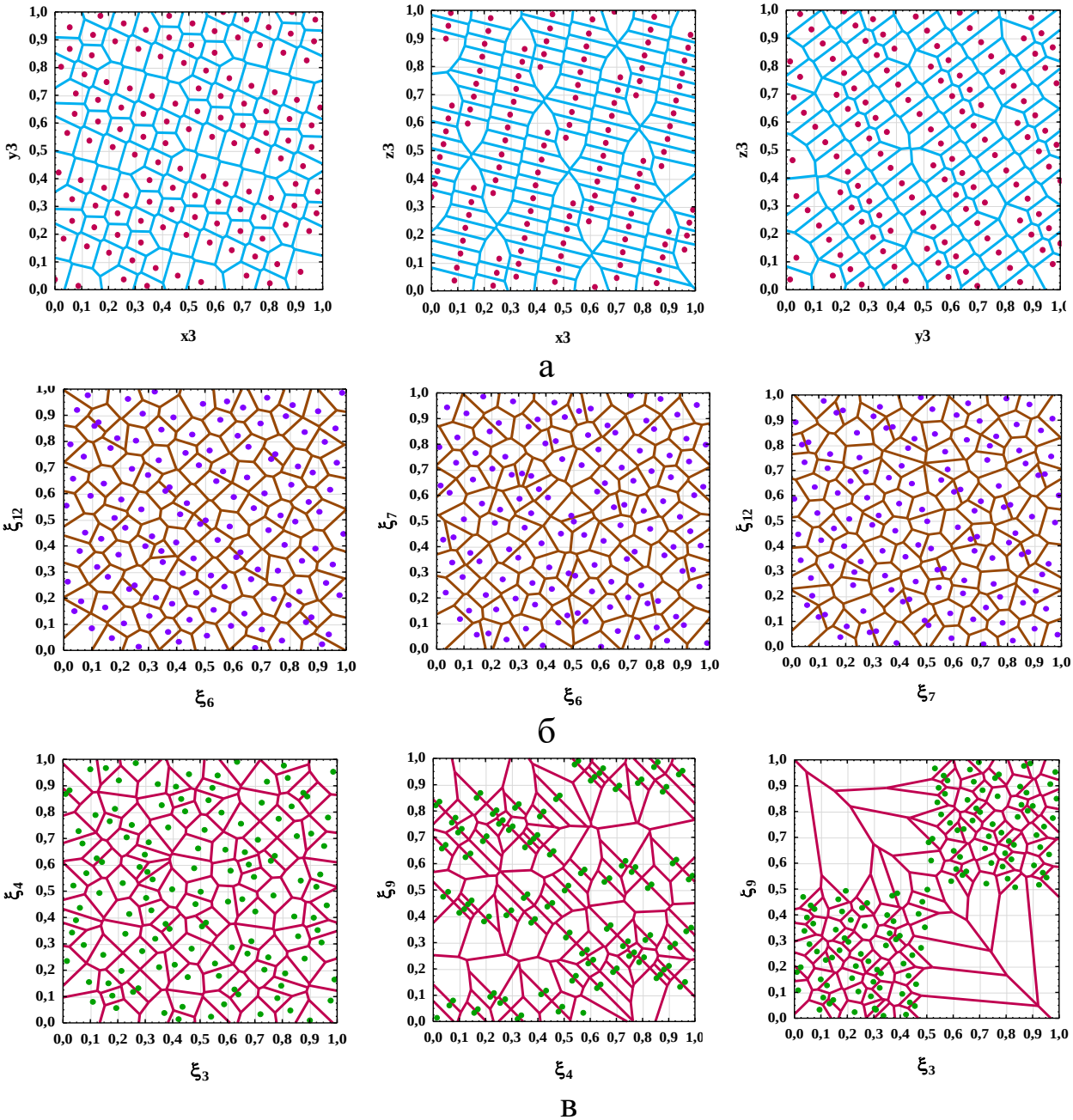


Рис. 1.12. Діаграми Вороного для проекцій тривимірних планів: а - R_3-послідовність; б, в - комбінації Sobol-послідовностей (ξ_6, ξ_7, ξ_{12}), (ξ_3, ξ_4, ξ_9) відповідно, де ξ_i - послідовність конкретного напрямку простору значень.

Із результатів, отриманих в дослідженнях [44], показників розбіжності відносно L_2-норми для двовимірних планів та візуального аналізу діаграм Вороного можна переконатися, що існують такі комбінації Sobol-послідовностей, які мають неоднозначні показники узагальненої розбіжності, деякі з них є кращими, а деякі є значно гіршими у порівнянні з *R*-послідовностями.

Для багатовимірних планів зі збільшенням розмірності простору все складніше відшуковувати комбінації Sobol-послідовностей, які мають найкращі показники узагальненої розбіжності, що потребує значних часових затрат. Хоча використання комбінацій Sobol-послідовностей все ж іноді показує кращі результати в результаті вдалого вибору направляючих чисел, однак проведення серії досліджень щодо автоматизації формування вибірок демонструє такий підхід досить затратним і не завжди раціональним. Тому перехід до *R*-послідовностей в цьому випадку має суттєву перевагу в простоті використання та досягнення гомогенності, так як вони гарантовано забезпечують рівномірне розподілення точок плану при збільшенні розмірності даних.

1.4. Огляд методів застосування штучних нейронних мереж для розв’язання обернених задач

1.4.1. Особливості розв’язання обернених задач з використанням нейромереж

Застосування нейромереж в дослідженнях на сьогодні досить розповсюджена практика. Вони мають здатність до узагальнення (generalization), а це означає, що вони можуть опрацьовувати результат на основі вхідних даних, які не були присутні в навчанні [38]. Крім того, нейромережам притаманні наступні якості: нелінійність, відображення вхідної інформації у вихідну; адаптивність, очевидність відповіді, масштабованість, уніфікація аналізу і проєктування.

Нещодавно data driven підходи з DNN розкрилися як нові потужні і гнучкі підходи до створення моделей для розв’язання прямих задач [45-52]. Подальшим розвитком використання DNN є припущення, що DNN також можуть розв’язувати обернені задачі, в тому числі реалізувати інверсію прямих задач [45].

Та незважаючи на цей факт, на практиці важко реалізуємо отримати розв'язок оберненої задачі. Тому для реалізації оберненої задачі доводиться багаторазово та ітеративно виконувати розв'язок прямої задачі. Такі прямі розв'язки зазвичай виведені з рівнянь Максвелла для кожної зони розташування точки розрахунку. Вони є досить обмеженими у використанні і відносно ресурсозатратними. Незважаючи на переваги аналітичних чи напіваналітичних методів розв'язання прямих задач, вони не є універсальними та адекватні лише для їх певних типів. Отже стає досить складно знайти таке рішення, яке може точно виявити чи відтворити нюанси фізичного явища конкретної задачі. Проблема загострюється, коли задача переходить від одно- чи двовимірних задач до трьох вимірностей і більше. В таких випадках використовуються чисельні методи, частіше за все метод скінченних елементів (Finite Element Method FEM), щоб отримати адекватні результати розрахунків. Альтернативою ресурсозатратних FEM чи аналітичних рішень є моделі, котрі основані на застосуванні DNN-апроксиматорів.

Особливістю DNN є те, що нейромережі здатні вчитися і відтворювати результат для досить специфічних задач, коли існує багато змінних чи поверхня відгуку має складну нелінійну топологію [46]. Іноді DNN корисні бо можуть знайти чи відкрити певні нелінійні зв'язки між вхідними змінними в латентному просторі [49]. Але на відміну від прямих задач, де спостерігається відношення "багато до одного" в рамках досліджуваної фізичної системи і її відповідним відгуком, інверсія прямих рішень часто зіштовхується з можливими проблемами неоднозначності. Тобто декілька вихідних сигналів можуть відповідати одному набору вхідних даних, що робить проблему ще тяжчою для вирішення [53]. Наразі відомі декілька методів обійти чи вирішити цю проблему, про такі методи з використанням нейромереж викладено нижче.

Навчання з учителем може бути визначене як задача віднайдення комплексних, зазвичай нелінійних, залежностей між двома наборами визначених даних [54]. Це випадок, коли мережа навчається відтворювати значення явно відомих зразків пар входів-виходів. Навчання з учителем підходить для розв'язку коректно поставлених задач [45], [53], [55]-[58]. Варто зазначити що, коли оптимізаційні підходи розв'язання цієї задачі застрягали в локальних мінімумах, то запропонований підхід з DNN приводив до

найбільш прийнятних результатів.

На відміну від навчання з учителем навчання без учителя може знаходити важливі патерни, кластери за певними ознаками чи особливостями, опрацьовуючи дані без явно визначених маркувань [59], [60]. Це дозволяє застосовувати DNN до менш чітко видимих, а значить і складніших задач. Виходячи з того, що нейромережі з навчанням без учителя навчаються самі по собі без спеціальної цілі, вони мають іншу природу ніж навчання з учителем, що дозволяє відшукати абсолютно нові латентні патерни, кластери, зв'язки чи закономірності в наборах даних [61], [62]-[65].

Навчання з підкріпленням (Reinforcement learning RL) це вид машинного навчання, що часто використовується в задачах теорії ігор і робототехніці [66]. Алгоритм RL проілюстрований на рис.1.13, де так званий "агент" вибирає з набору дій (d_g) куди рухатися та який стан (g) вибрати з ціллю отримати винагороду. Агент робить спроби знайти баланс між кількістю чи набором дій та пошуком шуканої винагороди, наприклад використовуючи жадібний (epsilon-greedy) алгоритм [67]. Середовище зазвичай представляється прямою DNN чи CEM моделлю, що обраховує винагороду з пари стан - дія (тобто вираховуючи s).

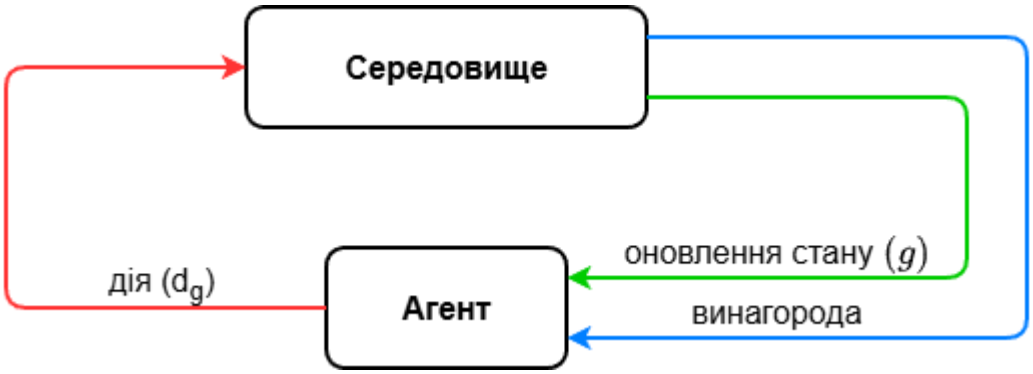


Рис. 1.13. Ілюстрація загального алгоритму навчання з підкріпленням.

З цього погляду розв'язок оберненої задачі як задачі для навчання з підкріпленням знаходиться більш випадковим шляхом, використовуючи бажану ціль s^* як винагороду та знаходячи найбільш підходящий набір параметрів з простору станів [68]-[71].

Агент зазвичай зіштовхується з досить динамічним середовищем, де отримання винагороди супроводжується створенням механізмів для пошуку чи розвідування, а не простого

вибору дії випадковим чином.

Якщо RL використовується як метод для розв'язку обернених задач, моделі не страждають від проблеми відношення "один до багатьох", так як застосування різних початкових параметрів та доданий фактор стохастичного пошуку надають декілька розв'язків для однієї цілі. Попри це, такий пошуковий алгоритм інколи може бути повільніший за звичайні методи оптимізації, особливо з погано підібраною функцією винагороди [72]. Також спостерігається взаємозалежність між пошуком рішення і його ефективністю, що значною мірою залежить від природи і структури даних.

Варто зазначити факт, що внаслідок апріорної невідомості наступного стану агенту запропоновані вектори даних можуть не бути явно оціненими. Отже, це призводить до необхідності прогонів прямої моделі при кожній ітерації , тобто епізоді.

RL є найбільш ефективним, коли простір даних залишається відносно малим [73], проте існує негативний фактор того, що при збільшенні простору він досить погано масштабується.

1.4.2. Тандемна нейронна мережа

Один з найбільш розповсюджених підходів, що долає проблему неунікальності, це додавання прямої моделі DNN до моделі оберненої задачі, тобто створення каскадної моделі, так званої тандемної нейронної мережі [74]-[76].

Покращений підхід використання тандемної нейромережі був описаний в роботі [74]. Авторами використовувалася тандемна структура DNN, що поєднувала мережу для оберненої задачі і вже навчену пряму модель (рис.1.14). При такому підході в першій фазі навчається мережа, що відтворює відгук прямої моделі. В другій фазі навчання ваги прямої моделі фіксуються і вона приєднується до мережі оберненої задачі та відбувається навчання мережі оберненої задачі. При цьому бажаний вихід служив як вхідні дані, та мережа навчалась мінімізуючи "втрати", тобто різницю між бажаним відгуком і відтвореним результатом. На етапі відтворення обернена модель самотужки здатна відтворювати потрібний відгук. Під час цього вихідні параметри бралися вже з проміжного шару. Як зазначається цей підхід допомагає побороти проблему неунікальності при розв'язанні обернених задач, так як він не вимагає специфічного набору параметрів для певного відгуку.

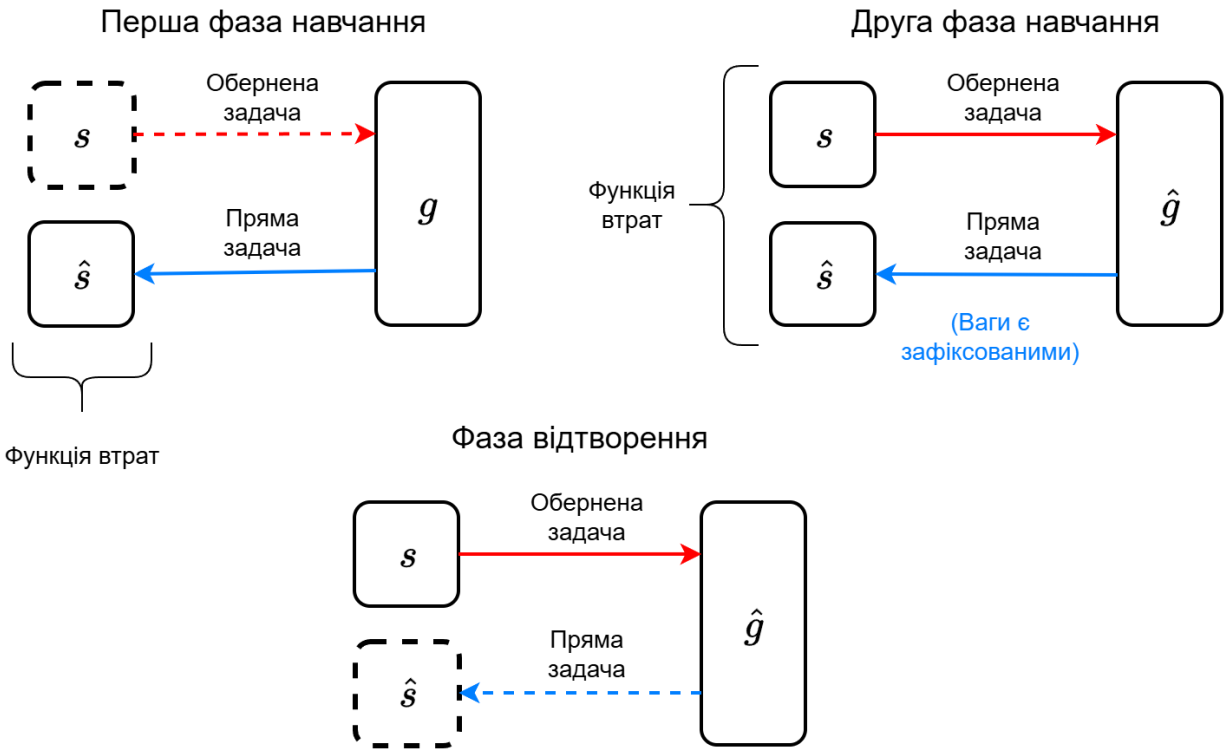


Рис. 1.14. Концепція тандемної нейронної мережі.

Тандемна мережа хоч і долає проблему неунікальності, але при цьому не має варіативності виходу, тобто в результаті можна отримати тільки один варіант розв'язку, який обумовлений навчанням. Іншим недоліком є те, що тандемні нейромережі потребують досить чималого набору даних для коректного навчання.

1.4.3. Генеративна змагальна мережа

Генеративна змагальна мережа (generative adversarial network GAN) є доволі новим розробленим алгоритмом машинного навчання, але цей алгоритм став найбільш цікавим з групи методів навчання без учителя [77].

GAN складаються з двох мереж, а саме генератора (generator) та дискримінатора (discriminator) (рис.1.15). Дві мережі поєднуються і одночасно навчаються по принципу гри з нульовою сумою. Мережа генератор приймає на вхід випадковий шум і генерує варіанти наборів даних, які мають бути бажаними вихідними параметрами. В той же час мережа дискримінатор перевіряє чи згенеровані виходи є даними, що задовольняють умови задачі. Задача генеративної мережі - це підміна даних (обман) для дискримінантної мережі, що вона реалізує генерацією реалістично подібних даних. Генератор

отримує випадковий шум z з можливим розподіленням по поверхні відгуку s. Дискримінатор намагається визначити чи вхідне значення $\hat{g}$ є істинним, тобто змагається з моделлю генератора. І тому після навчання генеративна мережа може створювати дизайни структур, що нагадують образи справжніх даних. В тому числі генеративна мережа добуває важливі закономірності з даних через зворотній зв'язок.

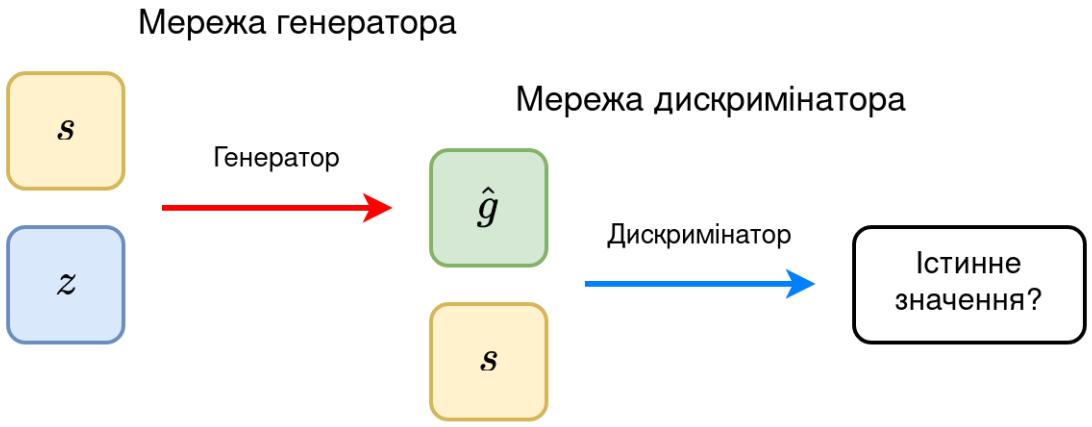


Рис. 1.15. Схема генеративно-змагальної мережі.

В дослідженні [61] автори додатково до моделі GAN додали мережу сурогатної симуляції, щоб апроксимувати вихідні параметри. Після навчання ця мережа здатна відтворювати потрібний вихід. Варто зазначити, що GAN генерують довільні структурні шаблони, які допомагають відкрити нові форми цих структур за межами людського розуміння, побудованого на досвіді та знаннях. Та попри це, хоч GAN і вирішують нестійку проблему рівноваги Неша (Nash Equilibrium), вони також не завжди дають стійке рішення.

Недавні дослідження показали, що глибокі згорткові GANs [78] є ефективними при розв'язку обернених задачах, та можуть вирішити проблему гри з нульовою сумою в більш стабільний спосіб [62].

Так як GAN часто використовуються для ідентифікації реальної структури параметрів вхідних даних із занадто параметризованого простору, як наприклад, дані зображень, тому нейромережі цього типу вимагають скрупульозного підбору даних при створенні випадкових вибірок для навчання [72]. Також відомо, що в GANs проявляється проблема вибору при деяких режимах (mode collapse) [79], коли мережа дискримінатора намагається обрати лише певні

набори згенерованих вихідних даних. І в подальшому, так би мовити, застрягає в певній підвибірці без дослідження всього простору поверхні відгуку. Так як GANs це по суті дві конкуруючі мережі, одночасне навчання цих двох DNNs є досить складним [80].

Навіть показуючи хороші результати при певних вибірках, GAN схильні до проблеми режимів. Коли GAN є добре та правильно натренованою, мережі цього типу можуть показувати вражаючі результати, особливо в поєднанні з традиційними методами оптимізації [81], [82], та зазвичай це супроводжується втратою додаткового часу на розробку і налагодження самої моделі.

Описані вище недоліки є перешкодою до вибору GAN як оптимального інструменту при розв'язанні обернених задач.

1.4.4. Автокодувальники та варіаційні автокодувальники

Автокодувальник (autoencoder AE) це загальна назва виду нейронних мереж, що по своїй структурі призначена без учителя навчатися та виділяти особливості з певного набору даних. Цей концепт машинного навчання відображений у [83].

Існує два види автокодувальників, що використовувалися для розв'язання обернених задач, а саме детермінований (deterministic autoencoder) та варіаційний автокодувальник (variational autoencoder VAE).

Детермінований автокодувальник служить для зменшення простору даних в менший латентний простір. В загальному структура автокодувальника складається з мережі кодувальника (encoder) та мережі декодувальника (decoder). Дані, проходячи через мережу кодувальника стискаються в менший кодований простір - латентний простір, а задача мережі декодувальника відтворювати оригінальний набір даних з значень латентного простору. Зазвичай значення латентного простору визначені як z, та не можуть бути прямо оцінені, але z є напряму визначеними з опрацьованих автокодувальником даних. Моделі з використанням автокодувальників є цікавими для стиснення простору даних в розв'язанні обернених задач [84], [85]. Завдяки автокодувальникам проблема відношення даних "один до багатьох" може бути неявно вирішена через латентний простір [85].

Варіаційний автокодувальник в розв'язанні обернених задач має дещо інше призначення та є найбільш використовуваним типом

автокодувальників. VAE належить до класу потужних генеративних мереж [86], що відтворюють вхід після того, як він був зжатий в декілька прихованих змінних. На відміну від інших методів розв'язання обернених задач мережа приймає на вхід дані, які потім закодовує в приховані змінні за визначеними раніше розподіленнями.

Моделі на основі VAE є генеративними моделями, основна задача яких навчитися розподілу вхідних даних, використовуючи ймовірність розподілення як значення латентного (μ_z, σ_z) простору [87] (рис.1.16).

Натренована нейромережа VAE може відтворювати зразки з нового заданого вектору латентного простору, що потім, будучи декодованим, може бути інтерпретований як дані. І саме тому VAE є типом нейромереж, що можуть генерувати нові структури даних з подібними властивостями до тих даних, на яких була ця нейромережа натренована [63], [88]-[93]. Так як VAE навчається як закодувати g в латентний простір, не існує ніякого зв'язку з цільовим розподілом s, що є вбудованим в базову структуру VAE.

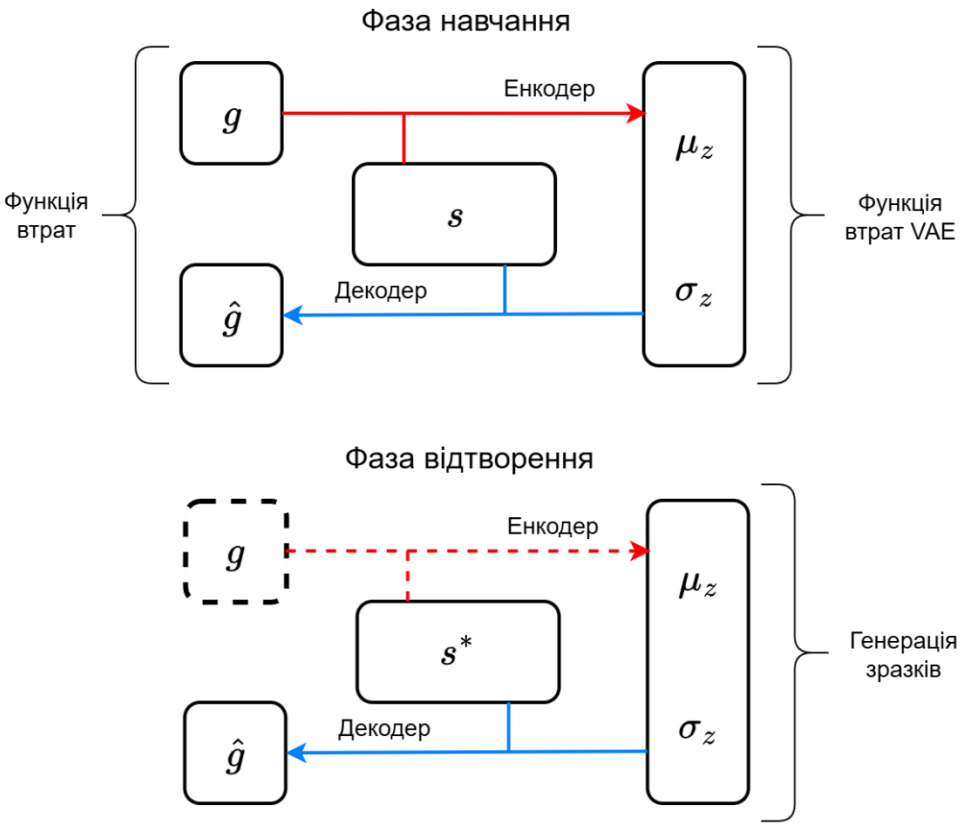


Рис. 1.16. Архітектура нейромережі варіаційного енкодера.

Тож для навчання потрібна пряма модель чи сурогатна модель

прямої задачі, щоб коректувати навчання самого VAE та додавати додаткові умови до функції втрат [91], або обраховувати чи відгук є задовільним перед мережею декодувальника [90].

VAE за своєю природою вирішує проблему відношення даних "один до багатьох" вважаючи, що для кожного s існує декілька можливих варіацій рішення $\hat{g}$ з різноманітним набором цих комбінацій з латентного простору z, що відповідає Гаусовому розподіленню. Так головною перевагою VAE є те, що він може давати декілька можливих рішень для певного s^*.

Розглядаючи недоліки VAE варто зазначити, що мережа намагається згенерувати можливі розподілення використовуючи варіаційний підхід та введення явної границі нижнього значення (the evidence lower bound ELBO), що є чисельно оптимізованою функцією втрат та є означеною як функція втрат для реконструкції поєднаною з регуляризаційною умовою дивергенції Кульбака–Лейблера (KL divergence). Такий тип ELBO як функції втрат є складним для обчислення [87].

До того ж отриманий латентний простір зазвичай є довільним та потребує підстройки параметрів протягом декількох ітерацій навчання. Відтворені дані від VAE є злегка зашумленими особливо у випадку невдалого вибору або великої розмірності латентного простору [72].

Так як мережі VAE навчаються відтворювати розподіл можливих g від s, природа таких залежностей невідома без послідовної побудови таких розподілень вручну. Отож хоч VAE і можуть моментально створювати вихід $\hat{g}$, проте перший отриманий результат не завжди є найкращим. Тому без знання закону розподілу все ж таки можна використовувати повторну симуляційну помилку, щоб оцінити здатність мережі відтворювати адекватний вихід $\hat{g}$, але це вимагає додаткових кроків оптимізації. Тож з практичної точки зору VAE є більш складними в навчанні ніж моделі на основі звичайних DNN та тандемні нейромережі.

1.4.5. Інверсійні нейронні мережі

Інверсні нейронні мережі Invertible Neural Networks (INNs) були в повній мірі представлені в роботі Ardizzone [94]. Дослідники показали ефективне застосування такого підходу для різного роду задач в природничих науках. INN покликані вирішити проблему

неоднозначності при розв'язку обернених задач без додаткових зусиль чи оптимізацій. Як зазначають автори при використані INN можна сфокусуватися на прямому навчанні нейромережі, використовуючи лише додаткові приховані змінні в функції втрат при навчанні.

Суть інверсної нейромережі полягає в наступному: мережу можна навчити добре зрозумілому прямому процесу $x \rightarrow y$ та потім просто отримати обернений розв'язок $y \rightarrow x$. Для досягнення такого результату створюється латентний вихідний простір z, що запобігає втраті прихованої інформації при прямому навчанні. Цей простір може містити в собі саме ту інформацію про x, що не міститься в y. Тож INN навчається зв'язувати приховані параметри щодо x в унікальних парах $[y, z]$, тобто змінних відгуку та латентних змінних. При навчанні прямій задачі INN намагається відобразити відношення $[y, z] = f(x)$, що дозволяє неявно визначити її інверсію $x = f^{-1}(y, z) = g(y, z)$. Латентний простір $p(z)$ формується за Гаусівським розподілом. Тож INN демонструє бажане відображення даних $p(x|y)$ за функцією визначником $x = g(y, z)$, що змушує за відомим розподілом $p(z)$ відповідати простору x при певних y.

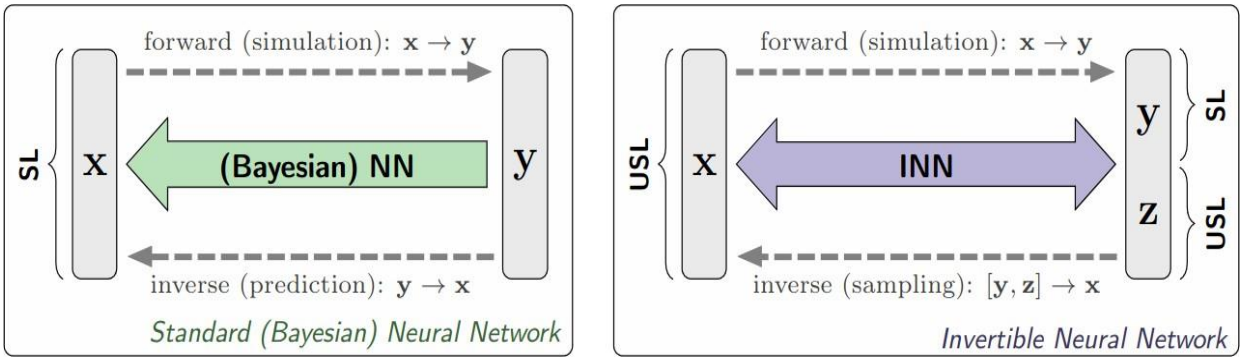


Рис. 1.17. Загальне порівняння звичайної і інверсної нейронної мережі [94].

Недоліком INN є те, що все ж таки зачасту доводиться вводити додаткові змінні при процесі навчання і використовувати їх у функції втрат як штрафи чи додаткові коефіцієнти. Іншим негативним фактором є симетрична архітектура таких мереж, тобто кількість виходів має дорівнювати кількості входів. Це призводить до додаткових незручностей та похибок при розрахунку функції цілі. Також слід зазначити, що ці мережі схильні до поганого опрацювання даних при досить великій розмірності входів-виходів.

1.4.6. Гібридні методи оптимізації з глибокими нейронними мережами

Довгий час для розв'язання обернених задач використовувалися числові методи оптимізації. Наприклад, такі оптимізаційні підходи, що ґрунтуються на градієнтних методах, як: level-set, density topology та interior point methods [95], а також генетичні (genetic) чи еволюційні (evolutionary) алгоритми [96], [97].

Багато підходів до розв'язання обернених задач покладаються на adjoint variables method [98]-[100], в якому ціллю є обрахунок зразків для f^{-1} для конкретної g, чим на тому щоб знайти саму функцію f^{-1}. Це притаманно випадкам, що вимагають специфічних знань в конкретній області навіть тоді, коли фізика процесу є менш зрозумілою.

Замість стандартних використань моделей DL DNN можуть бути поєднані з класичними оптимізаційними методами для розв'язання обернених задач. Такий підхід прийнято називати гібридним [72].

Як показано на рис.1.18, існує два базових сценаріїв гібридної оптимізації з DNN: це пряма оптимізація і оптимізація у латентному просторі.

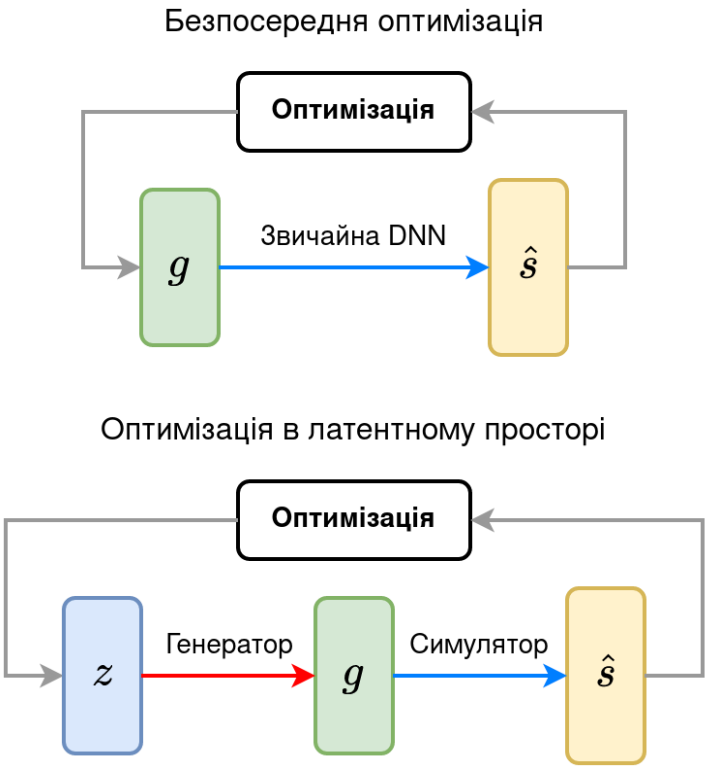


Рис. 1.18. Загальна схема підходів до розв'язання обернених задач засобами гібридної оптимізації.

У випадку прямої оптимізації сурогатна модель використовується як повний замінник прямої моделі. Наприклад, еволюційні алгоритми використовуються, щоб керувати простором даних g [101]. В моделях, що використовують оптимізацію у латентному просторі, основною метою є зменшення розмірності представлення латентного простору, що відповідає g, так як оптимізаційний процес проходить ефективніше у просторі з меншою розмірністю.

Моделі на основі сурогатних DNN є ключовим засобом покращення процесу оптимізації, так як через свою швидкодію дозволяють оптимізаційним алгоритмам бути ефективнішими, в тому числі у наданні кращих початкових точок пошуку екстремуму [91], [92]. Варто зазначити, що через зменшення розмірності можливо суттєво спростити розв'язання задачі без необхідності боротьби з проблемою відношення "один до багатьох" [85], так як оптимізаційні методи можуть покращити згенероване рішення $\hat{g}$ через ітеративні підходи.

Наразі вже відомі створені підходи до розв'язання обернених задач з застосуванням нейромережевої сурогатної оптимізації в поєднанні з традиційними оптимізаційними методами, такими як: еволюційні алгоритми (evolutionary strategies ES), генетичні алгоритми (genetic algorithms GA) та метод рою часток (particle swarm optimization PSO) [85], [91], [101]-[109].

Інша група гібридних моделей включає поєднання генераторних мереж з вторинними процедурами оптимізації, що ітеративно покращують рішення. Ці гібридні моделі використовують оптимізаційний метод, де для обчислення градієнтів чутливості (sensitivity analysis) застосовується ад'юнкт-рівняння (adjoint equation), що дозволяє ефективно розв'язувати задачі з великою кількістю змінних проектування [81], [82], [92], або еволюційні алгоритми [91].

Інакші підходи вдаються до розв'язання обернених задач з використанням DNN разом з Байєсовським (Bayesian) розв'язком [47], [110], [111], з методом внутрішніх точок [51], з усіченою Ньютонівською оптимізацією (truncated Newtonian optimization) [112], а також методом пошуку по таблицям (search-based lookup tables) [113]-[115].

Методи, що основані на гібридних підходах з DNN, можуть

справлятися з задачами швидше і ефективніше, так як вони спрощують обчислювальну складову оптимізаційного процесу. В загальному, методи з використанням сурогатних DNN та класичної оптимізації можуть бути застосовані до найпростішого розв'язання обернених задач.

Недоліком такого підходу є те, що здебільшого в задачах є спеціальні фізичні чи технічні обмеження, що, в свою чергу, призводить до застосування додаткових ускладнень загального алгоритму. Через це модифіковані методи стають вже вузькоспеціалізованими та доводиться витрачати додатковий час на тестування та налагодження таких алгоритмів.

Список використаних джерел до глави 1

1. Розв'язання задачі багатопараметричного контролю металевих виробів змінно-частотним методом вихрових струмів / Б. М. Березюк, У. Б. Марікуца, Т. В. Свірідова // *Вісн. Нац. ун-ту "Львів. політехніка"*. - 2006. - № 564. - С. 67-71. https://ena.lpnu.ua:8443/server/api/core/bitstreams/1d73c113-6793-469a-9f8c-6d1ce3e361bf/content
2. Метод суперпозиции при определении глубины упрочненного слоя вихретоковым методом / Б. М. Горкунов, А. А. Тищенко // *Вестник Нац. техн. ун-та "ХПИ" : сб. науч. тр. Темат. вып. : Электроэнергетика и преобразовательная техника*. – Харьков : НТУ "ХПИ". – 2011. – № 19. – С. 94-97. https://repository.kpi.kharkov.ua/handle/KhPI-Press/15716
3. Электромагнитный преобразователь с пространственно-периодическим полем для систем многопараметрового контроля / Б. М. Горкунов [и др.] // *Вісник Національного технічного університету "ХПІ". Сер. : Нові рішення в сучасних технологіях = Bulletin of the National Technical University "KhPI". Ser. : New solutions in modern technology* : зб. наук. пр. – Харків : НТУ "ХПІ", 2018. – № 26 (1302), т. 1. – С. 80-85. https://doi.org/10.20998/2413-4295.2018.26.12
4. Uncertainty estimation while proceeding multi-parameter eddy current testing / B. Gorkunov, S. Lvov, Y. Borysenko // *Системи обробки інформації* : зб. наук. пр. / гол. ред. О. І. Тимочко. – Харків : ХУПС, 2018. – Вип. 4 (155). – С. 92-97. https://doi.org/10.30748/soi.2018.155.12

5. Горкунов, Б. М., Львов, С. Г., Тамер, Ш., & Борисенко, Е. А. (2018). Экспериментальные исследования вихретокового преобразователя с пространственно-периодическими полями. *Метрологія та прилади*, (4), 45-50. https://library.kpi.kharkov.ua/files/Vestniki/2015_19.pdf

6. Тетерко, А. Я., & Гутник, В. І. (2011). Побудова зворотної функції перетворення приладів вихрострумового багатопараметрового контролю. *Фізико-хімічна механіка матеріалів*, (47, № 3), 103-108. http://nbuv.gov.ua/UJRN/PHKhMM_2011_47_3_17

7. Концепція побудови апаратури багатопараметрового вихрострумового контролю / А. Я. Тетерко, В. І. Гутник // *Відбір і оброб. інформації* : міжвід. зб. наук. пр. - 2010. - Вип. 33. - С. 9-14. https://nasplib.isofts.kiev.ua/server/api/core/bitstreams/2d218894-3039-469c-8169-95d37a11025a/content

8. Тетерко, А. Я., Гутник, В. И., Луценко, Г. Г., & Тетерко, А. А. (2014). Метод многопараметровых вихретоковых измерений толщины, электропроводности материала и толщины диэлектрического покрытия элементов конструкций. *Техническая диагностика и неразрушающий контроль*, (3), 55-60. http://nbuv.gov.ua/UJRN/TDNK_2014_3_8

9. Ayad, A., Benhamida, F., Bendaoud, A., Le Bihan, Y., & Bensetti, M. (2011). Solution of inverse problems in electromagnetic NDT using neural networks. *Przegląd Elektrotechniczny*, *87*(9a), 330-333. https://www.academia.edu/69050913/Solution_of_Inverse_Problems_in_Electromagnetic_NDT_Using_Neural_Networks

10. Burkhardt, J. (2019). Determination of the conductivity and thickness of conductive layers on conductive base materials. *Advances in Mechanical Engineering*, *11*(7), 1687814019854234. https://doi.org/10.1177/16878140198542

11. Chen, X., & Lei, Y. (2015). Electrical conductivity measurement of ferromagnetic metallic materials using pulsed eddy current method. *Ndt & E International*, *75*, 33-38. https://doi.org/10.1016/j.ndteint.2015.06.005

12. Ulapane, N., Alempijevic, A., Vidal Calleja, T., & Valls Miro, J. (2017). Pulsed eddy current sensing for critical pipe condition assessment. *Sensors*, *17*(10), 2208. https://doi.org/10.3390/s17102208

13. Lahrech, A. C., Abdelhadi, B., Feliachi, M., Zaoui, A., & Naïdjate, M. (2018). Electrical conductivity identification of a carbon fiber composite material plate using a rotating magnetic field and multi-coil eddy current sensor. *The European Physical Journal Applied Physics*, *83*(2), 20901. https://doi.org/10.1051/epjap/2018170411

14. Rekanos, I. T., Theodoulidis, T. P., Panas, S. M., & Tsiboukis, T. D. (1997). Impedance inversion in eddy current testing of layered planar structures via neural networks. *NDT & E International*, *30*(2), 69-74. https://doi.org/10.1016/S0963-8695(96)00047-3

15. Theodoulidis, T. (2019). Impedance of a coil above a planar conductor with an arbitrary continuous conductivity depth profile. *International Journal of Applied Electromagnetics and Mechanics*, *59*(4), 1179-1185. https://doi.org/10.3233/JAE-171122

16. Yin, W., & Peyton, A. J. (2006, April). Determining the step-change conductivity profile within layered metal structures using inductance spectroscopy. In *2006 IEEE Instrumentation and Measurement Technology Conference Proceedings* (pp. 2127-2131). IEEE. https://doi.org/10.1109/IMTC.2006.328503

17. Forrester, A., Sobester, A., & Keane, A. (2008). *Engineering design via surrogate modelling: a practical guide*. John Wiley & Sons. https://books.google.com.ua/books?hl=uk&lr=&id=ulMHmeMnRCcC&oi=fnd&pg=PR5&dq=Engineering+Design+via+Surrogate+Modelling:+A+Practical+Guide&ots=ggJN3tJd3G&sig=WJepzqXoNPz395TzKclknguNY2o&redir_esc=y#v=onepage&q=Engineering%20Design%20via%20Surrogate%20Modelling%3A%20A%20Practical%20Guide&f=false

18. Чубань, М. А. (2015). Аппроксимация поверхности отклика для использования в процессе параметрического синтеза машиностроительных конструкций. *Вісник Національного технічного університету ХПІ. Серія: Транспортне машинобудування*, (43), 161-164. http://www.irbis-nbuv.gov.ua/cgi-bin/irbis_nbuv/cgiirbis_64.exe?C21COM=2&I21DBN=UJRN&P21DBN=UJRN&IMAGE_FILE_DOWNLOAD=1&Image_file_name=PDF/vcpitm_2015_43_31.pdf

19. Espinosa Barcenas, O. U., Quijada Pioquinto, J. G., Kurkina, E., & Lukyanov, O. (2023). Surrogate aerodynamic wing modeling based on a multilayer perceptron. *Aerospace*, *10*(2), 149.

https://doi.org/10.3390/aerospace10020149

20. Garifullin, M. R., Barabash, A. V., Naumova, E. A., Zhuvak, O. V., Jokinen, T., & Heinisuo, M. (2016). Surrogate modeling for initial rotational stiffness of welded tubular joints. *Magazine of Civil Engineering*, (3 (63)), 53-76. https://doi.org/10.5862/MCE.63.4

21. Koziel, S., & Yang, X. S. (Eds.). (2011). *Computational optimization, methods and algorithms* (Vol. 356). Springer. https://books.google.com.ua/books?hl=uk&lr=&id=HOHwCAAAQBAJ&oi=fnd&pg=PR3&dq=Computational+Optimization,+Methods+and+Algorithm&ots=1FYU9j_LjJ&sig=E4VF3-Vs9cQbhvCk2_OjabYQ1Q8&redir_esc=y#v=onepage&q=Computational%20Optimization%2C%20Methods%20and%20Algorithm&f=false

22. Гальченко, В. Я., Трембовецька, Р. В., Тичков, В. В., Сторчак, А. В. (2020). Методи створення метамоделей: стан питання. *Вісник Вінницького політехнічного інституту*.№ 4: 74-88. https://doi.org/10.31649/1997-9266-2020-151-4-74-88

23. Grimstad, B., Robertson, P. M., & Foss, B. (2015). Virtual flow metering using B-spline surrogate models. *IFAC-PapersOnLine*, *48*(6), 292-297. https://doi.org/10.1016/j.ifacol.2015.08.046Get rights and content

24. Радченко, С. Г. (2015). Анализ методов моделирования сложных систем. *Математические машины и системы*, (4), 123-127. https://nasplib.isofts.kiev.ua/handle/123456789/113684

25. Fridedman, J. H. (1991). Multivariate adaptive regression splines (with discussion). *Ann Stat*, *19*(1), 79-141. https://doi.org/10.1214/aos/1176347963

26. MacKay, D. J. (2003). *Information theory, inference and learning algorithms*. Cambridge university press. https://books.google.com.ua/books?hl=uk&lr=&id=AKuMj4PN_EMC&oi=fnd&pg=PR11&dq=Information+Theory,+Inference+and+Learning+Algorithms&ots=EOnmhdaDAj&sig=EbSm7S_y70nsQLn4R4M7yRJlwhc&redir_esc=y#v=onepage&q=Information%20Theory%2C%20Inference%20and%20Learning%20Algorithms&f=false

27. Bilicz, S., Lambert, M., Gyimothy, S., & Pavo, J. (2012). Solution of inverse problems in nondestructive testing by a kriging-based surrogate model. *IEEE Transactions on Magnetics*, *48*(2), 495-498. https://doi.org/10.1109/TMAG.2011.2172196

28. Burnaev, E., Panov, M., Kononenko, D., & Konovalenko, I.

(2015). *Comparative analysis of optimization procedures based on Gaussian processes*. http://itas2012.iitp.ru/pdf/1569602385.pdf

29. Burnaev, E.V., Panov, M.E. & Zaytsev, A.A. Regression on the basis of nonstationary Gaussian processes with Bayesian regularization. *J. Commun. Technol. Electron.* 61, 661–671 (2016). https://doi.org/10.1134/S1064226916060061

30. Burnaev E., Erofeev P., Prikhodko P. Identification of main directions in approximation problems based on Gaussian processes. *Proceedings of the Moscow Institute of Physics and Technology*. 5.3 (19) (2013): 024-035.

31. Fang, H., & Horstemeyer, M. F. (2006). Global response approximation with radial basis functions. *Engineering optimization*, *38*(04), 407-424. https://doi.org/10.1080/03052150500422294

32. De Marchi, S., & Perracchiono, E. (2018). *Lectures on radial basis functions*. Preprint. https://www.math.unipd.it/~demarchi/RBF/LectureNotes_new.pdf

33. Ивахненко, А. Г. (1982). *Индуктивный метод самоорганизации моделей сложных систем*. http://ir.nmu.org.ua/handle/GenofondUA/34782

34. Ivakhnenko, A. G., & Ivakhnenko, G. A. (1995). The review of problems solvable by algorithms of the group method of data handling (GMDH). *Pattern recognition and image analysis c/c of raspoznavaniye obrazov i analiz izobrazhenii*, 5, 527-535. https://articles.gmdh.net/review/algorith.pdf

35. "GMDH – General description of the GMDH". https://www.gmdh.net/GMDH_abo.htm

36. "GMDH – Spectrum of the GMDH algorithms". https://www.gmdh.net/GMDH_alg.htm

37. Parrella, F. (2007). Online support vector regression. *Master's Thesis, Department of Information Science, University of Genoa, Italy*, *69*. https://d1wqtxts1xzle7.cloudfront.net/81740267/OnlineSVR_Thesis-libre.pdf?1646465219=&response-content-disposition=inline%3B+filename%3DOnline_Support_Vector_Regression.pdf&Expires=1764498072&Signature=bRfhA3OZsLN~~P1e6Ksthk2v7~v-xl66FMNuIL4zQcYud08NY4DDABmow3onpxwGrtTq6gNCT8WS~q

nialU5jujETJOY4XmUXj7FX7GthW0oX3JntLNzkGq~GxiTQDPCfDMxX~jeyoQd76vgAQJhMacG7PW~lzYx6Yk4D5jIOakPUYapdBeLqI1p56oK0ILMZp1~VJfv-xSA0wK4QjgSWnXIxqutedE3D~LlDOZCFO8lADo3yNhxvkk7LheKO8ik1tmCL8lDODnrirxXfQnnz1oo8Mv49Z1~Lb018jbS~I4H-cKJIrPftgUWdVRrwhhKiJpfdt09bzQBoR42OGUiMw__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA

38. Haykin, S. (2009). *Neural networks and learning machines, 3/E*. Pearson Education India. https://dai.fmph.uniba.sk/courses/NN/haykin.neural-networks.3ed.2009.pdf

39. Трембовецька, Р. В., Гальченко, В. Я., & Тичков, В. В. (2018). Побудова MLP-метамоделі накладного вихрострумового перетворювача для задач сурогатного оптимального синтезу. *Технічні вісми*, *47*(1-2), 27-31. https://er.chdtu.edu.ua/bitstream/ChSTU/462/1/27.pdf

40. Géron, A. (2022). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*. "O'Reilly Media, Inc.". https://books.google.com.ua/books?hl=uk&lr=&id=X5ySEAAAQBAJ&oi=fnd&pg=PT17&dq=Hands-On+Machine+Learning+with+Scikit-Learn,+Keras,+and+TensorFlow&ots=yC3utl00AP&sig=PAuUpPFTSt9Hxmm4-6gKrQnGLEM&redir_esc=y#v=onepage&q=Hands-On%20Machine%20Learning%20with%20Scikit-Learn%2C%20Keras%2C%20and%20TensorFlow&f=false

41. Roberts, M. (2018). The unreasonable effectiveness of quasirandom sequences. *Extreme Learning*, *575*. https://extremelearning.com.au/unreasonable-effectiveness-of-quasirandom-sequences/

42. Struckmeier, J. (1995). Fast generation of low-discrepancy sequences. *Journal of Computational and Applied Mathematics*, *61*(1), 29-41. https://doi.org/10.1016/0377-0427(94)00054-5

43. Galchenko, V. Y., Koshevoy, M. D., & Trembovetskaya, R. .V. (2022). Homogeneous plans of multi-factory experiments on quasi-random R-Roberts sequences for surrogate modeling in a vortex style structuroscopy . *Radio Electronics, Computer Science, Control*, (3), 22. https://doi.org/10.15588/1607-3274-2022-3-2

44. Halchenko, V., Trembovetska, R., Tychkov, V., & Storchak, A. V. (2020). The construction of effective multidimensional computer

designs of experiments based on a quasi-random additive recursive Rd–sequence. *Applied Computer Systems*, *25*(1), 70-76. https://doi.org/10.2478/acss-2020-0009

45. Peurifoy, J., Shen, Y., Jing, L., Yang, Y., Cano-Renteria, F., DeLacy, B. G., ... & Soljačić, M. (2018). Nanophotonic particle simulation and inverse design using artificial neural networks. *Science advances*, *4*(6), eaar4206. https://doi.org/10.1126/sciadv.aar4206

46. Qu, Y., Jing, L., Shen, Y., Qiu, M., & Soljacic, M. (2019). Migrating knowledge between physical scenarios based on artificial neural networks. *ACS Photonics*, *6*(5), 1168-1174. https://doi.org/10.1021/acsphotonics.8b01526

47. Balin, I., Garmider, V., Long, Y., & Abdulhalim, I. (2018). Training artificial neural network for optimization of nanostructured VO_2-based smart window performance. *Optics express*, *27*(16), A1030-A1040. https://doi.org/10.1364/OE.27.0A1030

48. Baxter, J., Calà Lesina, A., Guay, J. M., Weck, A., Berini, P., & Ramunno, L. (2019). Plasmonic colours predicted by deep learning. *Scientific reports*, *9*(1), 8074. https://doi.org/10.1038/s41598-019-44522-7

49. Kiarashinejad, Y., Zandehshahvar, M., Abdollahramezani, S., Hemmatyar, O., Pourabolghasem, R., & Adibi, A. (2020). Knowledge discovery in nanophotonics using geometric deep learning. *Advanced Intelligent Systems*, *2*(2), 1900132. https://doi.org/10.1002/aisy.201900132

50. Sajedian, I., Kim, J., & Rho, J. (2019). Finding the optical properties of plasmonic structures by image processing using a combination of convolutional neural networks and recurrent neural networks. *Microsystems & nanoengineering*, *5*(1), 27. https://doi.org/10.1038/s41378-019-0069-y

51. Inampudi, S., & Mosallaei, H. (2018). Neural network based design of metagratings. *Applied Physics Letters*, *112*(24). https://doi.org/10.1063/1.5033327

52. Nadell, C. C., Huang, B., Malof, J. M., & Padilla, W. J. (2019). Deep learning for accelerated all-dielectric metasurface design. *Optics express*, *27*(20), 27523-27535. https://doi.org/10.1364/OE.27.027523

53. Liu, D., Tan, Y., Khoram, E., & Yu, Z. (2018). Training deep

neural networks for the inverse design of nanophotonic structures. *Acs Photonics*, *5*(4), 1365-1369. https://doi.org/10.1021/acsphotonics.7b01377

54. Caruana, R., & Niculescu-Mizil, A. (2006, June). An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd international conference on Machine learning* (pp. 161-168). https://doi.org/10.1145/1143844.1143865

55. So, S., Mun, J., & Rho, J. (2019). Simultaneous inverse design of materials and structures via deep learning: demonstration of dipole resonance engineering using core–shell nanoparticles. *ACS applied materials & interfaces*, *11*(27), 24264-24268. https://doi.org/10.1021/acsami.9b05857

56. Malkiel, I., Mrejen, M., Nagler, A., Arieli, U., Wolf, L., & Suchowski, H. (2018). Plasmonic nanostructure design and characterization via deep learning. *Light: Science & Applications*, *7*(1), 60. https://doi.org/10.1038/s41377-018-0060-7

57. Asano, T., & Noda, S. (2018). Optimization of photonic crystal nanocavities based on deep learning. *Optics express*, *26*(25), 32704-32717. https://doi.org/10.1364/OE.26.032704

58. Hemmatyar, O., Abdollahramezani, S., Kiarashinejad, Y., Zandehshahvar, M., & Adibi, A. (2019). Full color generation with fano-type resonant hfo 2 nanopillars designed by a deep-learning approach. *Nanoscale*, *11*(44), 21266-21274. https://doi.org/10.1039/C9NR07408B

59. Barlow, H. B. (1989). Unsupervised learning. *Neural computation*, *1*(3), 295-311. https://doi.org/10.1162/neco.1989.1.3.295

60. Hofmann, T. (2001). Unsupervised learning by probabilistic latent semantic analysis. *Machine learning*, *42*(1), 177-196. https://doi.org/10.1023/A:1007617005950

61. Liu, Z., Zhu, D., Rodrigues, S. P., Lee, K. T., & Cai, W. (2018). Generative model for the inverse design of metasurfaces. *Nano letters*, *18*(10), 6570-6576. https://doi.org/10.1021/acs.nanolett.8b03171

62. So, S., & Rho, J. (2019). Designing nanophotonic structures using conditional deep convolutional generative adversarial networks. *Nanophotonics*, *8*(7), 1255-1261. https://doi.org/10.1515/nanoph-2019-0117

63. Ma, W., Cheng, F., Xu, Y., Wen, Q., & Liu, Y. (2019). Probabilistic representation and inverse design of metamaterials based on

a deep generative model with semi-supervised learning strategy. *Advanced Materials*, *31*(35), 1901111. https://doi.org/10.1002/adma.201901111

64. Jiang, J., Sell, D., Hoyer, S., Hickey, J., Yang, J., & Fan, J. A. (2019). Free-form diffractive metagrating design based on generative adversarial networks. *ACS nano*, *13*(8), 8872-8878. https://doi.org/10.1021/acsnano.9b02371

65. An, S., Zheng, B., Tang, H., Shalaginov, M. Y., Zhou, L., Li, H., ... & Zhang, H. (2019). Generative multi-functional meta-atom and metasurface design networks. *arXiv preprint arXiv:1908.04851*. https://doi.org/10.48550/arXiv.1908.04851

66. Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., & Riedmiller, M. (2013). Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*. https://arxiv.org/pdf/1312.05602

67. Sutton, R. S., & Barto, A. G. (1998). *Reinforcement learning: An introduction* (Vol. 1, No. 1, pp. 9-11). Cambridge: MIT press. https://doi.org/10.1017/S0263574799271172

68. Sajedian, I., Badloe, T., & Rho, J. (2019). Optimisation of colour generation from dielectric nanostructures using reinforcement learning. *Optics express*, *27*(4), 5874-5883. https://doi.org/10.1364/OE.27.005874

69. Huang, Z., Liu, X., & Zang, J. (2019). The inverse design of structural color using machine learning. Nanoscale, 11(45), 21748-21758. https://doi.org/10.1039/C9NR06127D

70. Badloe, T., Kim, I., & Rho, J. (2020). Biomimetic ultra-broadband perfect absorbers optimised with reinforcement learning. *Physical Chemistry Chemical Physics*, *22*(4), 2337-2342. https://doi.org/10.1039/C9CP05621A

71. Wang, H., Zheng, Z., Ji, C., & Guo, L. J. (2021). Automated multi-layer optical design via deep reinforcement learning. *Machine Learning: Science and Technology*, *2*(2), 025013. https://doi.org/10.1088/2632-2153/abc327

72. Khatib, O., Ren, S., Malof, J., & Padilla, W. J. (2021). Deep learning the electromagnetic properties of metamaterials—a comprehensive review. *Advanced Functional Materials*, *31*(31), 2101748. https://doi.org/10.1002/adfm.202101748

73. Kober, J., Bagnell, J. A., & Peters, J. (2013). Reinforcement

learning in robotics: A survey. *The International Journal of Robotics Research*, *32*(11), 1238-1274. https://doi.org/10.1177/0278364913495721

74. Padilla, W. J., Aronsson, M. T., Highstrete, C., Lee, M., Taylor, A. J., & Averitt, R. D. (2007). Electrically resonant terahertz metamaterials: Theoretical and experimental investigations. *Physical Review B—Condensed Matter and Materials Physics*, *75*(4), 041102. https://doi.org/10.1103/PhysRevB.75.041102

75. Liu, X., Tyler, T., Starr, T., Starr, A. F., Jokerst, N. M., & Padilla, W. J. (2011). Taming the blackbody with infrared metamaterials as selective thermal emitters. *Physical review letters*, *107*(4), 045901. https://doi.org/10.1103/PhysRevLett.107.045901

76. Jahani, S., & Jacob, Z. (2016). All-dielectric metamaterials. *Nature nanotechnology*, *11*(1), 23-36. https://doi.org/10.1038/nnano.2015.304

77. Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2014). Generative adversarial nets. *Advances in neural information processing systems*, *27*. https://doi.org/10.48550/arXiv.1406.2661

78. Radford, A., Metz, L., & Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*. https://doi.org/10.48550/arXiv.1511.06434

79. Kodali, N., Abernethy, J., Hays, J., & Kira, Z. (2017). On convergence and stability of gans. *arXiv preprint arXiv:1705.07215*. https://doi.org/10.48550/arXiv.1705.07215

80. Ganguly, K. (2017). *Learning generative adversarial networks: next-generation deep learning simplified*. Packt Publishing. https://www.scribd.com/document/809939523/9781788396417

81. Jiang, J., & Fan, J. A. (2019). Global optimization of dielectric metasurfaces using a physics-driven neural network. *Nano letters*, *19*(8), 5366-5372. https://doi.org/10.1021/acs.nanolett.9b01857

82. Jiang, J., Sell, D., Hoyer, S., Hickey, J., Yang, J., & Fan, J. A. (2019). Free-form diffractive metagrating design based on generative adversarial networks. *ACS nano*, *13*(8), 8872-8878. https://doi.org/10.1021/acsnano.9b02371

83. Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). Deep Learning Cambridge. *MA: MIT Press http://www.*

deeplearningbook. org. https://doi.org/10.1007/s10710-017-9314-z

84. Kiarashinejad, Y., Abdollahramezani, S., Zandehshahvar, M., Hemmatyar, O., & Adibi, A. (2019). Deep learning reveals underlying physics of light–matter interactions in nanophotonic devices. *Advanced Theory and Simulations*, *2*(9), 1900088. https://doi.org/10.1002/adts.201900088

85. Kiarashinejad, Y., Abdollahramezani, S., & Adibi, A. (2020). Deep learning approach based on dimensionality reduction for designing electromagnetic nanostructures. *npj Computational Materials*, *6*(1), 12. https://doi.org/10.1038/s41524-020-0276-y

86. Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., ... & Lerchner, A. (2017, February). β-vae: Learning basic visual concepts with a constrained variational framework. In *International conference on learning representations*. https://www.cs.toronto.edu/~bonner/courses/2022s/csc2547/papers/generative/disentangled-representations/beta-vae,-higgins,-iclr2017.pdf

87. Kingma, D. P., & Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*. https://indico.math.cnrs.fr/event/11377/attachments/4589/6915/18012024_Kingma-and-Welling-2022%20Auto-Encoding%20Variational%20Bayes.pdf

88. Qiu, T., Shi, X., Wang, J., Li, Y., Qu, S., Cheng, Q., ... & Sui, S. (2019). Deep learning: a rapid and efficient route to automatic metasurface design. *Advanced Science*, *6*(12), 1900128. https://doi.org/10.1002/advs.201900128

89. Shi, X., Qiu, T., Wang, J., Zhao, X., & Qu, S. (2020). Metasurface inverse design using machine learning approaches. *Journal of Physics D: Applied Physics*, *53*(27), 275105. https://doi.org/10.1088/1361-6463/ab8036

90. Ma, W., & Liu, Y. (2020). A data-efficient self-supervised deep learning model for design and characterization of nanophotonic structures. *Science China Physics, Mechanics & Astronomy*, *63*(8), 284212. https://doi.org/10.1007/s11433-020-1575-2

91. Liu, Z., Raju, L., Zhu, D., & Cai, W. (2020). A hybrid strategy for the discovery and design of photonic structures. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, *10*(1), 126-135. https://doi.org/10.1109/JETCAS.2020.2970080

92. Kudyshev, Z. A., Kildishev, A. V., Shalaev, V. M., &

Boltasseva, A. (2020). Machine learning–assisted global optimization of photonic devices. *Nanophotonics*, *10*(1), 371-383. https://doi.org/10.1515/nanoph-2020-0376

93. Kudyshev, Z. A., Kildishev, A. V., Shalaev, V. M., & Boltasseva, A. (2020). Machine-learning-assisted metasurface design for high-efficiency thermal emitter optimization. *Applied Physics Reviews*, *7*(2). https://doi.org/10.1063/1.5134792

94. Ardizzone, L., Kruse, J., Wirkert, S., Rahner, D., Pellegrini, E. W., Klessen, R. S., ... & Köthe, U. (2018). Analyzing inverse problems with invertible neural networks. *arXiv preprint arXiv:1808.04730*. https://doi.org/10.48550/arXiv.1808.04730

95. Waltz, R. A., Morales, J. L., Nocedal, J., & Orban, D. (2006). An interior algorithm for nonlinear optimization that combines line search and trust region steps. *Mathematical programming*, *107*(3), 391-408. https://doi.org/10.1007/s10107-004-0560-5

96. Yu, X., & Gen, M. (2010). *Introduction to evolutionary algorithms*. London: Springer London. https://doi.org/10.1007/978-1-84996-129-5_3

97. Molesky, S., Lin, Z., Piggott, A. Y., Jin, W., Vucković, J., & Rodriguez, A. W. (2018). Inverse design in nanophotonics. *Nature Photonics*, *12*(11), 659-670. https://doi.org/10.1038/s41566-018-0246-9

98. Bendsøe, M. P., & Kikuchi, N. (1988). Generating optimal topologies in structural design using a homogenization method. Computer methods in applied mechanics and engineering, 71(2), 197-224. https://doi.org/10.1016/0045-7825(88)90086-2

99. Herskovits, J. (Ed.). (2012). *Advances in structural optimization* (Vol. 25). Springer Science & Business Media. https://books.google.com.ua/books?hl=uk&lr=&id=bgbwCAAAQBAJ&oi=fnd&pg=PR7&dq=Advances+in+Structural+Optimization.&ots=08W6HkRg2x&sig=BRxUoVDMKwKGtcBCBPXFkS5qofc&redir_esc=y#v=onepage&q=Advances%20in%20Structural%20Optimization.&f=false

100. El Bechari, R., Guyomarch, F., & Brisset, S. (2022). The adjoint variable method for computational electromagnetics. *Mathematics*, *10*(6), 885. https://doi.org/10.3390/math10060885

101. Miyatake, Y., Sekine, N., Toprasertpong, K., Takagi, S., & Takenaka, M. (2020). Computational design of efficient grating couplers

using artificial intelligence. *Japanese Journal of Applied Physics*, *59*(SG), SGGE09. https://doi.org/10.7567/1347-4065/ab641c

102. da Silva, M. R., Nóbrega, C. D. L., Silva, P. H. D. F., & D'Assunção, A. G. (2014). Optimization of FSS with Sierpinski island fractal elements using population-based search algorithms and MLP neural network. *Microwave and Optical Technology Letters*, *56*(4), 827-831. https://doi.org/10.1002/mop.28214

103. Liu, Z., Zhu, D., Lee, K. T., Kim, A. S., Raju, L., & Cai, W. (2020). Compounding meta-atoms into metamolecules with hybrid artificial intelligence techniques. *Advanced Materials*, *32*(6), 1904790. https://doi.org/10.1002/adma.201904790

104. Zhang, Q., Liu, C., Wan, X., Zhang, L., Liu, S., Yang, Y., & Cui, T. J. (2019). Machine-learning designs of anisotropic digital coding metasurfaces. *Advanced theory and simulations*, *2*(2), 1800132. https://doi.org/10.1002/adts.201800132

105. Hegde, R. S. (2019). Photonics inverse design: pairing deep neural networks with evolutionary algorithms. *IEEE Journal of Selected Topics in Quantum Electronics*, *26*(1), 1-8. https://doi.org/10.1109/JSTQE.2019.2933796

106. Hegde, R. S. (2019). Accelerating optics design optimizations with deep learning. *Optical Engineering*, *58*(6), 065103. https://doi.org/10.1117/1.OE.58.6.065103

107. Zhang, T., Liu, Q., Dan, Y., Yu, S., Han, X., Dai, J., & Xu, K. (2020). Machine learning and evolutionary algorithm studies of graphene metamaterials for optimized plasmon-induced transparency. *Optics Express*, *28*(13), 18899-18916. https://doi.org/10.1364/OE.389231

108. Kalt, V., González-Alcalde, A. K., Es-Saidi, S., Salas-Montiel, R., Blaize, S., & Macías, D. (2018). Metamodeling of high-contrast-index gratings for color reproduction. *Journal of the Optical Society of America A*, *36*(1), 79-88. https://doi.org/10.1364/JOSAA.36.000079

109. González-Alcalde, A. K., Salas-Montiel, R., Kalt, V., Blaize, S., & Macías, D. (2019). Engineering colors in all-dielectric metasurfaces: metamodeling approach. *Optics Letters*, *45*(1), 89-92. https://doi.org/10.1364/OL.45.000089

110. Li, Y., Xu, Y., Jiang, M., Li, B., Han, T., Chi, C., ... & Fang, Z. (2019). Self-learning perfect optical chirality via a deep

neural network. *Physical review letters*, *123*(21), 213902. https://doi.org/10.1103/PhysRevLett.123.213902

111. Sakurai, A., Yada, K., Simomura, T., Ju, S., Kashiwagi, M., Okada, H., ... & Shiomi, J. (2019). Ultranarrow-band wavelength-selective thermal emission with aperiodic multilayered metamaterials designed by Bayesian optimization. *ACS central science*, *5*(2), 319-326. https://doi.org/10.1021/acscentsci.8b00802

112. Hammond, A. M., & Camacho, R. M. (2019). Designing integrated photonic devices using artificial neural networks. *Optics express*, *27*(21), 29620-29638. https://doi.org/10.1364/OE.27.029620

113. El-Mosalmy, D. D., Hameed, M. F. O., Areed, N. F., & Obayya, S. S. A. (2014). Novel neural network based optimization approach for photonic devices. *Optical and Quantum Electronics*, *46*(3), 439-453. https://doi.org/10.1007/s11082-013-9869-8

114. Gostimirovic, D., & Winnie, N. Y. (2018). An open-source artificial neural network model for polarization-insensitive silicon-on-insulator subwavelength grating couplers. *IEEE Journal of Selected Topics in Quantum Electronics*, *25*(3), 1-5. https://doi.org/10.1109/JSTQE.2018.2885486

115. Gabr, A. M., Featherston, C., Zhang, C., Bonfil, C., Zhang, Q. J., & Smy, T. J. (2019). Design and optimization of optical passive elements using artificial neural networks. *Journal of the Optical Society of America B*, *36*(4), 999-1007. https://doi.org/10.1364/JOSAB.36.000999

ГЛАВА 2
МЕТОД ВИМІРЮВАННЯ ПРОФІЛІВ ЕЛЕКТРОФІЗИЧНИХ ХАРАКТЕРИСТИК МАТЕРІАЛУ ЦИЛІНДРИЧНИХ ОБ'ЄКТІВ З АПРІОРНИМ НАКОПИЧЕННЯМ ДАНИХ

2.1. Концептуальна постановка задачі

Визначення приповерхневих радіальних профілів електрофізичних характеристик матеріалу циліндричних ОК неруйнівним вихрострумовим методом вимірювань дозволяє відслідковувати структурні особливості матеріалу виробів, що є дуже важливим при контролі проведення багатьох технологічних операцій щодо зміцнення їх поверхні, спостереження модифікації стану в процесі експлуатації, дослідження хімічного та фазового складу тощо [1], [2].

Суть вимірювання приповерхневих радіальних профілів електрофізичних характеристик полягає в наступному. Після термічного чи термохімічного оброблення деталей азотуванням, гартуванням, цементацією, покриттям нітридом титану тощо змінюються фізико-механічні властивості матеріалу поверхневого шару матеріалу виробів. Завдяки кореляційним зв'язкам ЕП та МП з фізико-механічними властивостями матеріалу, в свою чергу, за допомогою вихрострумових вимірювань можна отримати інформацію про такі властивості приповерхневого шару матеріалу як в'язкість, пластичність, твердість, теплоємність, міцність, а крім того, хімічний і фазовий склад [3].

Вважатимемо, що електрофізичні властивості матеріалу ОК змінюються вздовж радіуса неперервно відповідно до певних законів розподілу $\sigma = \sigma(r)$, $\mu = \mu(r)$ (рис.2.1). Бажано отримувати результат вимірювання при одній частоті струму збудження та в реальному масштабі часу за одну вимірювальну операцію.

Негативними факторами впливу на процес вимірювання є нестабільність частоти струму збудження і вірогідна непостійність радіусу ОК (тобто зазору між перетворювачем і поверхнею ОК), можливі зміни яких треба враховувати при проведенні вимірювань.

З математичної точки зору визначення зазначених профілів параметрів за отриманим сигналом ВСП є некоректно поставленою задачею, розв'язання якої розглядається в цьому досліджені.

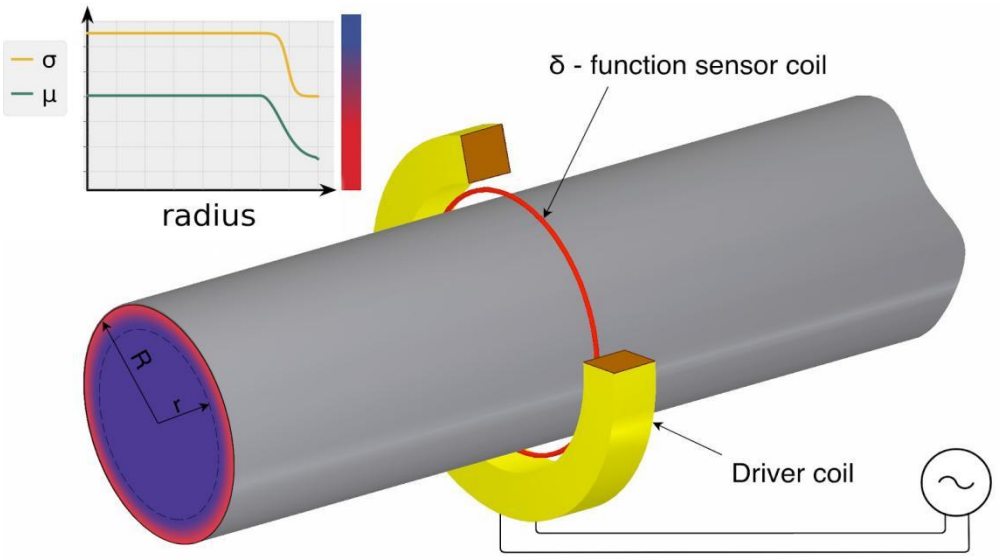


Рис. 2.1. Вихрострумовий метод вимірювання властивостей матеріалу приповерхневого шару циліндричних ОК: $(R-r)$ - приповерхневий шар із змінними електрофізичними характеристиками матеріалу; Driver coil - котушка збудження; δ-function sensor coil - вимірювальна котушка у вигляді нескінченно тонкого витка.

Враховуючи багатопараметровість задачі ідентифікації профілів та з метою скорочення обсягу вимірювальних операцій, накопичення апріорної інформації щодо ОК має сенс виконувати в сурогатній моделі процесу контролю при варіюванні суттєвими технічними параметрами, зокрема ЕП, магнітною проникністю МП приповерхневого шару, діаметром ОК, частотою збудження вихрових струмів [4]. Слід зауважити, що сурогатна модель створюється заздалегідь перед проведенням вимірювань на основі “точної” електродинамічної моделі, шляхом застосування багатовимірних апроксимаційних технік та характеризується мінімальною обчислювальною ресурсоємністю. Поєднання швидкості обрахунків й їх точності забезпечується використанням методології замісного моделювання, тобто сурогатним моделюванням, як складової загального процесу реконструкції (ідентифікації) приповерхневих профілів властивостей матеріалу ОК (рис.2.2). Завдання побудови сурогатної моделі складає один з етапів алгоритму розв’язання задачі, що розглядається.

Умова застосування методу вимірювання в реальному масштабі часу накладає відповідні додаткові умови на техніку обчислювальних складових методу в сенсі їх швидкості.

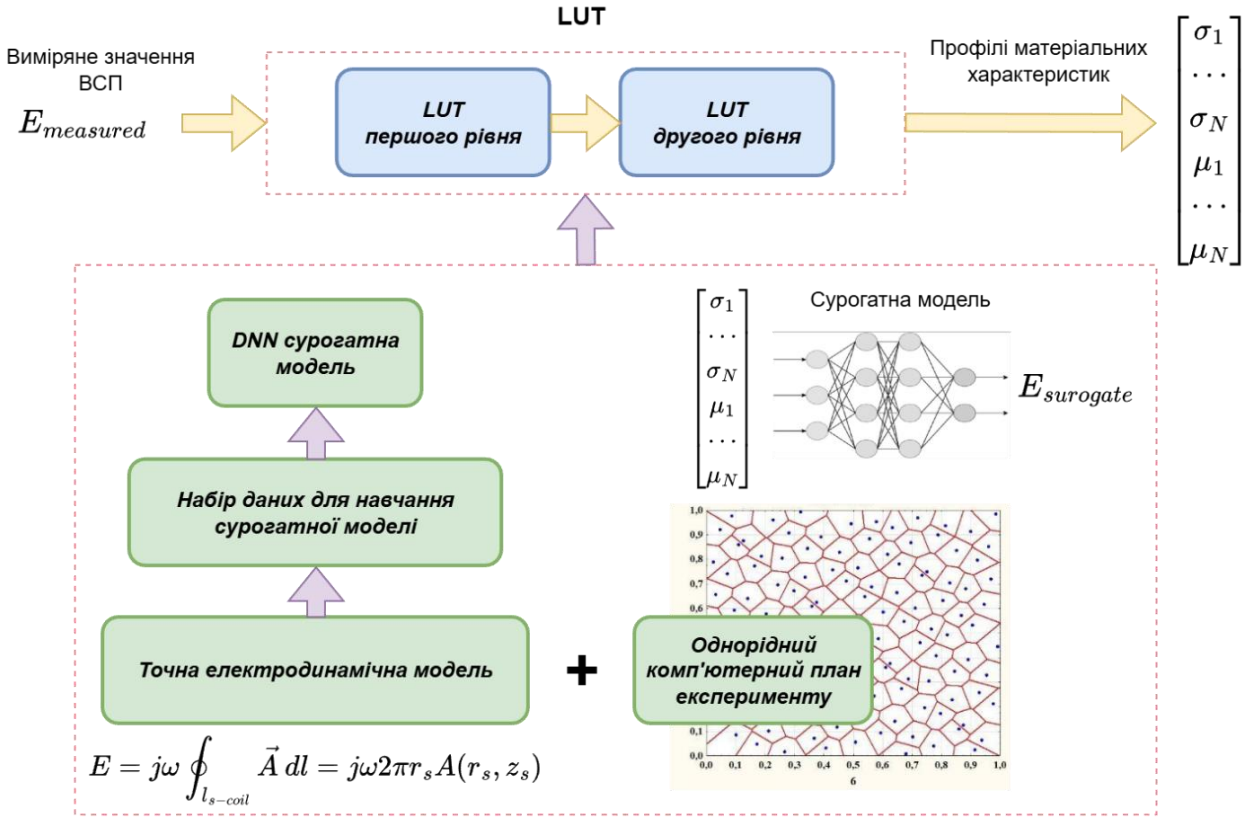


Рис. 2.2. Загальна схема процесу ідентифікації радіальних приповерхневих профілів матеріальних характеристик об’єктів контролю.

Надалі зосередимося на основних етапах реалізації процесу вихрострумового вимірювання радіальних профілів електрофізичних характеристик матеріалу ОК.

2.2. “Точна” електродинамічна модель процесу вимірювання вихрострумовим перетворювачем електрофізичних властивостей циліндричних об’єктів контролю

Математична модель складалася на основі загальних законів Максвелла теорії електромагнітного поля та описує процес контролю прохідним круговим ВСП циліндричного співвісного ОК. Відомими є математичні моделі процесу вихрострумового контролю циліндричних ОК [5], [6], але внаслідок універсальності має сенс використовувати аналітичну модель Dodd-Deeds [7]. Для спрощення представлення профілів електрофізичних параметрів у її контексті пропонується використовувати кусково-постійну апроксимацію, коли ОК

вважається умовно багатошаровим та електрофізичні параметри в кожному n-му шарі із його (K - 1) шарів приймаються сталими: $\sigma_n = \sigma(r)$, $\mu_n = \mu(r)$, де $n = 1, 2, \ldots, (K - 1)$ (рис.2.3).

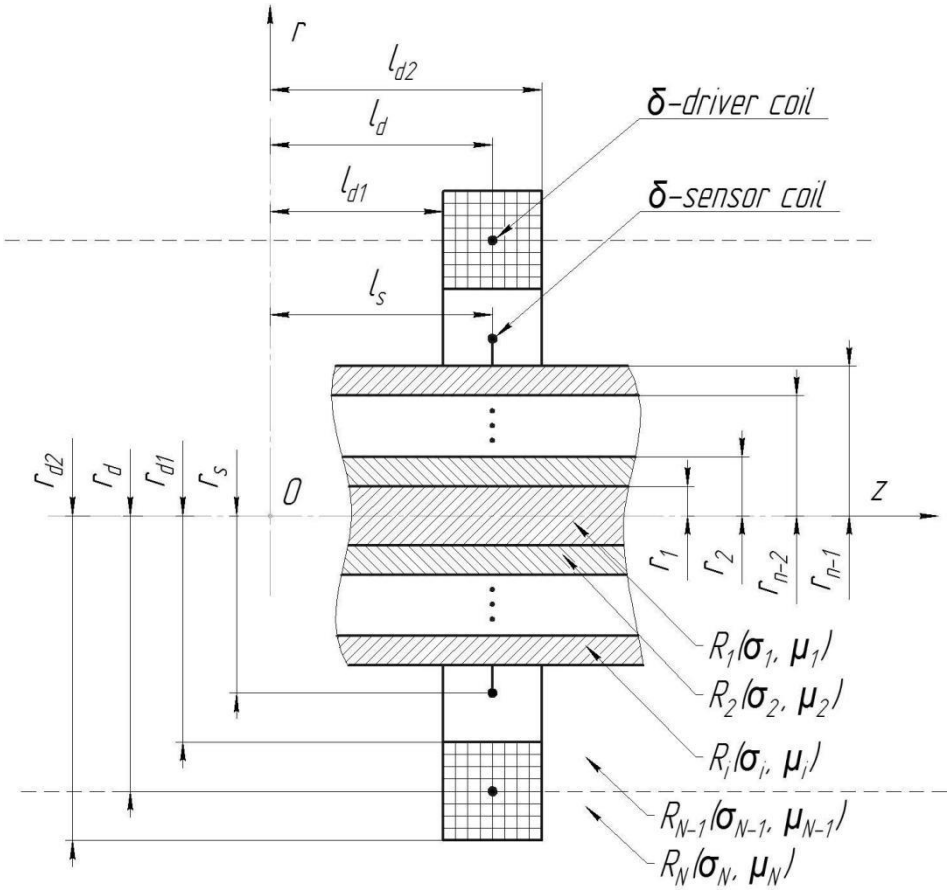


Рис. 2.3. Геометрична модель прохідного ВСП з циліндричним ОК: r_{d1}, r_{d2} - внутрішній та зовнішній радіуси котушки збудження відповідно; r_n - зовнішній радіус n-го шару; l_{d1}, l_{d2} - відстані до граней котушки збудження, r_s - радіус вимірювального витка, l_s - відстань до вимірювального витка.

Для коректного опису математичної моделі введено поняття регіону. Кожен регіон у циліндричній системі координат може бути визначений системою таких нерівностей:

$$R_1 = \{0 \le r \le r_1, 0 \le \varphi \le 2\pi, -\infty < z < \infty\}$$
$$\vdots$$
$$R_i = \{r_{i-1} \le r \le r_i, 0 \le \varphi \le 2\pi, -\infty < z < \infty\}$$
$$\vdots$$
$$R_N = \{r_{N-1} < r, 0 \le \varphi \le 2\pi, -\infty < z < \infty\},$$

де $i = 2, 3, ..., (N - 1)$;

N - загальна кількість регіонів;

$r_{N-1} = r_d$.

Математична модель складалася при прийнятті таких припущень [7]: середовища є лінійними, ізотропними та однорідними; струм збудження I є синусоїдальним, що змінюється з кутовою частотою ω.

Котушка збудження на початковому етапі розглядається як нескінченно тонкий виток з радіусом r_d. Також приймається, що осі ВСП та циліндричного ОК співпадають. Густина струму збудження та векторний потенціал у циліндричній системі координат за таких умов мають тільки азимутальну складову:

$$\vec{A}(r, \varphi, z) = A(r, z)\vec{e}_\varphi,\ \vec{J}(r, \varphi, z) = J(r, z)\vec{e}_\varphi.$$

Диференціальні рівняння для векторного потенціалу в регіонах із номерами $(N-1)$ та N можна записати у вигляді:

$$R_{N-1} \cup R_N: \quad \frac{\partial^2 A}{\partial r^2} + \frac{1}{r}\frac{\partial A}{\partial r} - \frac{A}{r^2} + \frac{\partial^2 A}{\partial z^2} = 0,$$

а в регіонах $R_1, R_2, ..., R_{N-2}$ відповідно:

$$\frac{\partial^2 A^{(i)}}{\partial r^2} + \frac{1}{r}\frac{\partial A^{(i)}}{\partial r} - \frac{A^{(i)}}{r^2} + \frac{\partial^2 A^{(i)}}{\partial z^2} = j\omega\mu_i\sigma_i A^{(i)}, \qquad (2.1)$$

де $i = 1, 2, ..., (N-2)$;

μ_i - абсолютна магнітна проникність;

$j = \sqrt{-1}$.

Враховуючи, що для векторного потенціалу з фізичних міркувань виконуються такі умови: а) A є скінченним при $r = 0$,

б) $A \to 0$ при $r \to \infty$, та беручи до уваги граничні умови:

$$A^{(i)}(r,z)\Big|_{r=r_i} = A^{(i+1)}(r,z)\Big|_{r=r_i},$$

$$\frac{1}{\mu_i}\frac{\partial A^{(i)}}{\partial r}(r,z)\Big|_{r=r_i} = \frac{1}{\mu_{i+1}}\frac{\partial A^{(i+1)}}{\partial r}(r,z)\Big|_{r=r_i},$$

де $i = 1,2,...,(N-2)$,

$$A^{(N-1)}(r,z)\Big|_{r=r_{N-1}} = A^{(N)}(r,z)\Big|_{r=r_{N-1}},$$

$$\frac{1}{\mu_{N-1}}\frac{\partial A^{(N-1)}}{\partial r}(r,z)\Big|_{r=r_{N-1}} = \frac{1}{\mu_N}\frac{\partial A^{(N)}}{\partial r}(r,z)\Big|_{r=r_{N-1}} + I\delta(r-r_d)\delta(z-z_d),$$

де δ - дельта-функція Дірака,
отримані рівняння для векторного потенціалу в будь-якому регіоні в середині витка зі струмом, що має такий вигляд:

$$A(r,z,r_d,z_d) = \frac{I\mu_0 r_d}{\pi}\int_0^{\infty}\frac{Q1\,Q2}{U_{22}V_{11}-U_{12}V_{21}}\cos(\alpha(z-z_d))\,d\alpha,$$

$$Q1 = V_{11}I_1(\alpha_n r) + V_{21}K_1(\alpha_n r), \qquad (2.2)$$

$$Q2 = U_{12}I_1(\alpha r_d) + U_{22}K_1(\alpha r_d),$$

де $\mu_0 = 4\pi\cdot 10^{-7}$ Гн/м - магнітна стала;

$$V_{11}(n+1,n) = \left(K_0(\alpha_{n+1}r_n)I_1(\alpha_n r_n) + \frac{\beta_n}{\beta_{n+1}}I_0(\alpha_n r_n)K_1(\alpha_{n+1}r_n)\right)\alpha_{n+1}r_n;$$

$$U_{12}(n+1,n) = \left(K_0(\alpha_{n+1}r_n)K_1(\alpha_n r_n) - \frac{\beta_n}{\beta_{n+1}}K_0(\alpha_n r_n)K_1(\alpha_{n+1}r_n)\right)\alpha_{n+1}r_n;$$

$$V_{21}(n+1,n) = \left(I_0(\alpha_{n+1}r_n)I_1(\alpha_n r_n) - \frac{\beta_n}{\beta_{n+1}}I_0(\alpha_n r_n)I_1(\alpha_{n+1}r_n)\right)\alpha_{n+1}r_n;$$

$$U_{22}(n+1,n) = \left(I_0(\alpha_{n+1}r_n)K_1(\alpha_n r_n) + \frac{\beta_n}{\beta_{n+1}}K_0(\alpha_n r_n)I_1(\alpha_{n+1}r_n)\right)\alpha_{n+1}r_n;$$

$$\beta_n = \left(\frac{\mu_0}{\mu_n} \right) \alpha_n ;$$

I_0, I_1 - модифіковані функції Бесселя першого роду нульового та першого порядків комплексного аргументу;

K_0, K_1 - модифіковані функції Бесселя другого роду нульового та першого порядків комплексного аргументу;

$\alpha_n = \sqrt{\alpha^2 - j\mu_n\sigma_n}$, $n = 1, 2, ..., K$.

Векторний потенціал в середині котушки збудження, яка має прямокутний поперечний переріз та однорідний розподіл густини струму збудження, можна записати у вигляді:

$$A(r,z) = IN_d \int_{l_{d1}}^{l_{d2}} \int_{r_{d1}}^{r_{d2}} A(r,z,r_d,z_d)\, dr_d\, dz_d \,, \qquad (2.3)$$

де $N_d = \dfrac{W}{(r_{d2} - r_{d1})(l_{d2} - l_{d1})}$;

W - кількість витків котушки збудження.

Зміною порядку інтегрування та в результаті його виконання отримано вираз для векторного потенціалу:

$$A(r,z) = \frac{IN_d \mu_0 r_d}{\pi} \int_0^{\infty} \frac{Q1\, Q2}{\alpha^3 (U_{22}V_{11} - U_{12}V_{21})} Q3\, d\alpha,$$

$$Q1 = \sin(\alpha(z - l_{d1})) - \sin(\alpha(z - l_{d2})), \qquad (2.4)$$

$$Q2 = V_{11} I_1(\alpha_n r) + V_{21} K_1(\alpha_n r),$$

$$Q3 = U_{12} I(r_{d2}, r_{d1}) + U_{22} K(r_{d2}, r_{d1}),$$

де $I(r_{d2}, r_{d1}) = \int_{\alpha r_{d1}}^{\alpha r_{d2}} t I_1(\alpha t)\, dt$;

$K(r_{d2}, r_{d1}) = \int_{\alpha r_{d1}}^{\alpha r_{d2}} t K_1(\alpha t)\, dt$.

Наведена в круговому вимірювальному витку напруга з урахуванням (2.4) обчислюється відповідно до співвідношення:

$$E = j\omega \oint_{l_{s-coil}} \vec{A}\, dl = j\omega 2\pi r_s A\left(r_s, z_s\right). \qquad (2.5)$$

Отже, задаючи обране радіальне розподілення параметрів матеріалу ОК, модель дозволяє отримати відповідний відгук перетворювача у вигляді напруги в комплексній формі представлення.

2.3. Створення сурогатної моделі процесу контролю з апріорним накопиченням інформації

В попередній главі було зроблено висновок, що використання DNN як основи для створення сурогатної моделі - це універсальний і надійний підхід в контексті апроксимації функцій з заздалегідь невідомою формою поверхні відгуку.

Виходячи з точної аналітичної моделі процесу вимірювання, створення сурогатної моделі ускладнене тим, що напруга ВСП представлена комплексним числом, тобто характеризується модулем А та фазою φ. Зазвичай нейронні мережі в їх класичному варіанті використання оперують дійсними числами. Це вимагає дещо інших підходів їх застосування в рамках досліджуваної задачі, які відрізняються від загальноприйнятих. Тобто є необхідність використовувати комплекснозначні нейронні мережі CVNNs (Complex-Valued Neural Networks) або комплекснозначні нейронні мережі, що розщеплюються SCVNNs (Splittable Complex-Valued Neural Networks). Слід відзначити, що для регресійних комплекснозначних нейронних мереж входи, виходи, зміщення та синаптичні ваги мають мати комплексні значення. Для комплекснозначних мереж, що розщеплюються, зазвичай використовують дійснозначні мережі окремо для дійсної та уявної частин вхідних сигналів з наступним об’єднанням в комплексний вихід. Використання таких нейронних мереж в дослідженнях, що вимагають розрахунків величин, які представляються комплексними числами, є більш природним з математичної точки зору та демонструє кращі результати в порівнянні з комплекснозначними нейронними мережами, але є більш складним в реалізації [8].

Архітектура комплекснозначної штучної нейронної мережі SCVNNs, яка використовується в цих дослідженнях, ілюструється

на рис.2.4. Для реалізації регресійної комплексозначної штучної нейронної мережі застосовувалися дві глибокі окремі дійснозначні мережі RVNNs (Real-Valued Neural Networks) прямого поширення з навчанням методом зворотного розповсюдження помилки (Back Propagation) для визначення оптимальних наборів вагових коефіцієнтів [9].

В більшості випадків двох-трьох прихованих шарів цілком достатньо для адекватної апроксимації будь-якої неперервної нелінійної гіперповерхні відгуку, що доведено теоретично [10].

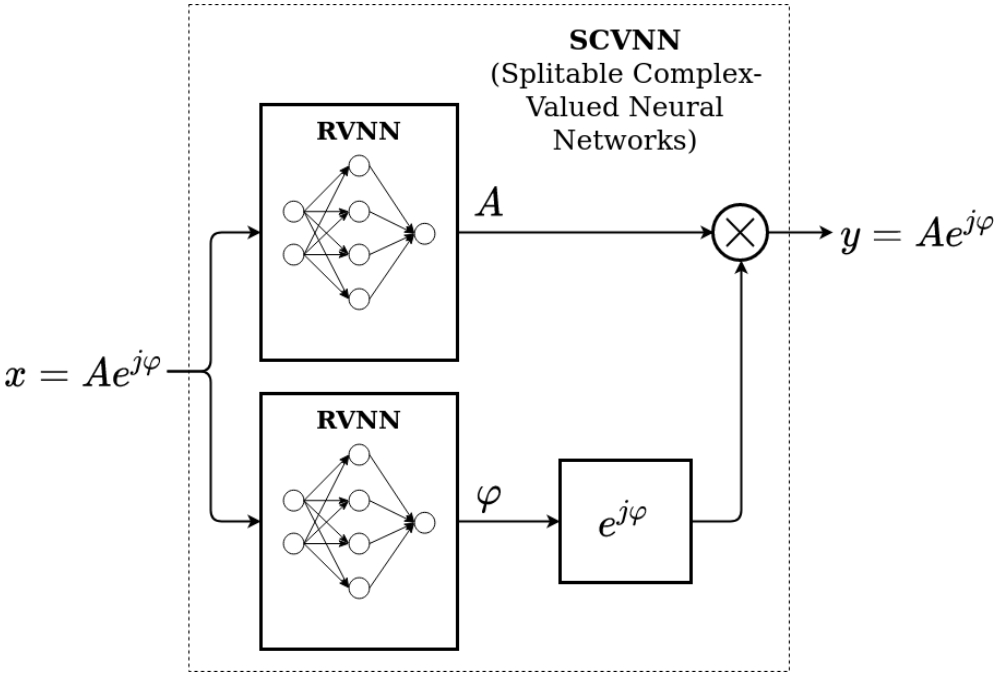


Рис. 2.4. Архітектура комплекснозначної штучної нейронної мережі з розщепленням.

Вихідний відгук *k*-шарової нейромережі, необхідної для нелінійних задач апроксимації, визначається за наступними формулами для вихідних значень нейронів відповідно:

$$a_j^{(1)} = f^{(1)}\left(w_{0,j}^{(1)} + \sum_{i=1}^{N^{(Input\ layer)}} x_i w_{i,j}^{(1)} \right),\ j = 1..N^{(1)};$$

$$a_j^{(k)} = f^{(k)}\left(w_{0,j}^{(k)} + \sum_{i=1}^{N^{(k-1)}} a_i^{(k-1)} w_{i,j}^{(k)} \right),$$

$$j = 1..N^{(k)};\ k = 2,\ldots,Output\ layer;$$

$$y = a^{(Output\ layer)},$$

де x - вектор вхідних даних;

$w^{(k)}$ - вагові коефіцієнти k-го прихованого шару;

$w_0^{(k)}$ - зміщення шару нейронів;

$N^{(Input\ layer)}$, $N^{(k)}$ - кількість нейронів відповідно вхідного та прихованого шарів;

$f^{(k)}$ - функція активації шару нейронів (вважається, що функція активації для всіх нейронів в шарі однакова);

$a^{(Output\ layer)}$ - вихід останнього шару нейронної мережі з одиничною функцією активації.

В наведених формулах враховується факт того, що кількість нейронів в різних шарах нейромережі не є однаковою.

Попередньо перед процедурою навчання мережі слід проводити нормування початкових даних з метою приведення їх до єдиної “шкали”. При цьому має бути дотриманий наступний підхід до розділення даних для навчання нейромережевої сурогатної моделі: навчальна частина загальної вибірки даних (Training Set) - це саме та підвибірка, що використовується в процесі навчання; підвибірка для застосування механізму контрольної кросс-валідації (Validation Set) для недопущення ефекту «перенавчання» (Overfitting) і контрольна тестова підвибірка (Control Test Set), що використовується виключно для розрахунку помилки апроксимації, тобто оцінки якості моделі. Одночасне спостереження за зміною в динаміці помилки навчання та помилки валідації на тестових даних дозволяє не тільки фіксувати факт «перенавчання», але й вчасно зупиняти процес підгонки (Early Stopping), забезпечуючи здатність мережі узагальнювати результат на нові вибіркові дані. Для навчання нейромереж доцільно застосовувати сучасні стохастичні оптимізаційні методи градієнтного спуску із накопиченням імпульсу: Adam (Adaptive Moment Estimation), Нестерова (Nesterov accelerated gradient), Adagrad (Adaptive Gradient). Вибір найкращих мереж із множини їх варіантів варто здійснювати за результатами перевірки на контрольній частині вибірки. В результаті така стратегія дозволяє мінімізувати не помилку навчання, а досягти мінімізації помилки узагальнення, що значно важливіше.

Адекватність побудованих нейромереж зручно контролювати

певною сукупністю об’єктивних показників [10], серед яких першочергово слід відзначити продуктивність, що обчислюється за наступною формулою:

$$ex = \frac{100}{m}\sum_{i=1}^{m}\left(1-\left|\frac{\hat{y}_i - y_i}{\hat{y}_i}\right|\right), \tag{2.6}$$

де m - кількість зразків даних для перевірки;

y - бажаний вихід мережі;

$\hat{y}$ - вихід мережі при верифікації за створеною сурогатною моделлю.

Серед інших показників валідації мереж є поширеними середнє абсолютне значення похибки моделі MAE (Mean Absolute Error), середньоквадратична похибка MSE (Mean Square Error), RMSE (Root Mean Square Error), гістограма розподілу залишків, діаграма розсіювання тощо:

$$MAE = \frac{\sum_{i=1}^{m}\left|y_i - \hat{y}_i\right|}{m}, \tag{2.7}$$

$$MSE = \frac{\sum_{i=1}^{m}\left(y_i - \hat{y}_i\right)^2}{m}, \tag{2.8}$$

$$RMSE = \sqrt{MSE}\,. \tag{2.9}$$

Використовуючи такий підхід реалізація адекватної сурогатної моделі не є проблемною, не вимагає надто великої кількості даних для навчання і має гарні узагальнюючі властивості та накопичує апріорну інформації щодо процесу вихрострумового контролю [11].

Варто зазначити, що описаний підхід для створення сурогатних моделей за допомогою DNN, є універсальним до задач аналогічного типу. Тобто при застосуванні його до іншої задачі, достатньо при необхідності змінити лише структуру нейронної мережі.

2.4. Метод створення однорідних планів експериментів на R-послідовностях Робертса

Обчислювальна технологія побудови сурогатних моделей [12]-

[15], яка включає планування комп'ютерного експерименту, формування репрезентативної вибірки, створення ефективної нейромережевої сурогатної моделі та перевірки її валідності, є перспективною для застосування в багатьох технічних додатках для розв'язання обернених задач.

Формування навчальної вибірки для побудови сурогатної моделі є одним з найважливіших етапів дослідження, оскільки суттєво впливає на подальші його результати. Вибірка має забезпечити максимальний обсяг інформації щодо топології поверхні відгуку для її ефективної апроксимації [4]. Якщо попередні відомості щодо її поведінки відсутні, то доцільним є обрання однорідного плану експерименту. Тому має сенс використовувати комп'ютерний план експерименту, створений на основі *R*-послідовностей.

Рекурсивну *R*-послідовність в *d*-вимірному просторі математично можна записати у вигляді:

$$R_s\left(\phi_s\right): \quad t_n = \left\{N \cdot \alpha\right\}, \tag{2.10}$$

де N - кількість точок плану $N = 1,2,3,\ldots$;

α - ірраціональне число, $\alpha = \left(\frac{1}{\phi_d}, \frac{1}{\phi_d^2}, \frac{1}{\phi_d^3}, \ldots \frac{1}{\phi_d^d}\right)$;

ϕ_s - унікальний позитивний корінь рівняння $x^{s+1} = x + 1$.

Для $s = 1$, $\phi_1 = 1,618033...$; для $s = 2$, $\phi_2 = 1,324717...$; для $s = 3$, $\phi_3 = 1,220744...$

В якості оцінки неоднорідності планів експериментів на множині послідовностей, що спостерігаються в одиничному гіперкубі, зазвичай використовуються два типи розбіжностей відносно L_2-норми: центровану розбіжність (the centered discrepancy) та циклічну розбіжність (the wrap-around discrepancy), які є інваріантними щодо перемаркування й упорядкування факторів та відносно обертання координат.

Показники розбіжності для N точок плану в *d*-вимірному просторі обчислюються відповідно до співвідношень [16]:

а) центрована розбіжність

$$\left[CD(P)\right]^2 = \left(\frac{13}{12}\right)^d - \frac{2}{N}\cdot\sum_{k=1}^{N}\prod_{j=1}^{d}\left[1+\frac{1}{2}\cdot\left|x_{kj}-0.5\right|-\frac{1}{2}\cdot\left|x_{kj}-0.5\right|^2\right]+$$

$$+\frac{1}{N^2}\cdot\sum_{k=1}^{N}\sum_{j=1}^{N}\prod_{i=1}^{d}\left[1+\frac{1}{2}\cdot\left|x_{kj}-0.5\right|+\frac{1}{2}\cdot\left|x_{ji}-0.5\right|-\frac{1}{2}\cdot\left|x_{kj}-x_{ji}\right|\right] \quad ;$$

б) циклічна розбіжність

$$\left[WD(P)\right]^2 = \left(\frac{4}{3}\right)^d + \frac{1}{N^2}\cdot\sum_{k=1}^{N}\sum_{i=1}^{N}\prod_{j=1}^{d}\left[\frac{3}{2}-\left|x_{ki}-x_{ji}\right|\cdot\left(1-\left|x_{ki}-x_{ji}\right|\right)\right].$$

Вважається, що більш низьке значення розбіжності при порівнянні планів експерименту притаманно більш рівномірному, а, відповідно, більш бажаному плану. Для наочності порівняльного аналізу планів доцільно використовувати графічне представлення згенерованих даних у вигляді діаграм Вороного [17].

Технологічна обробка поверхні виробу зачасту призводить до зміни електрофізичних параметрів матеріалу лише в приповерхневій зоні ОК. Очевидно, що приповерхневий матеріал ОК має неперервну зміну електрофізичних параметрів уздовж його радіусу $\sigma = \sigma(r)$, $\mu = \mu(r)$, а глибинний матеріал має сталі параметри. Так як в обраній математичній моделі процесу вимірювань [7] ЕП та МП в умовних шарах матеріалу ОК вважаються сталими, застосовується кусково-постійна апроксимація розподілу електрофізичних властивостей матеріалу приповерхневої зони.

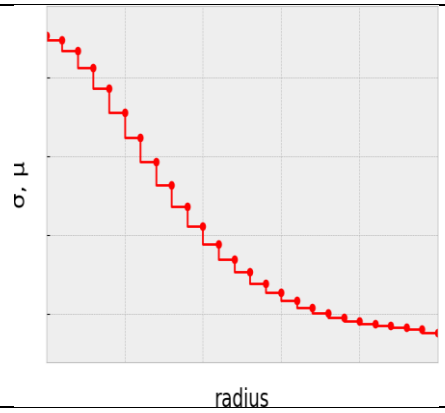
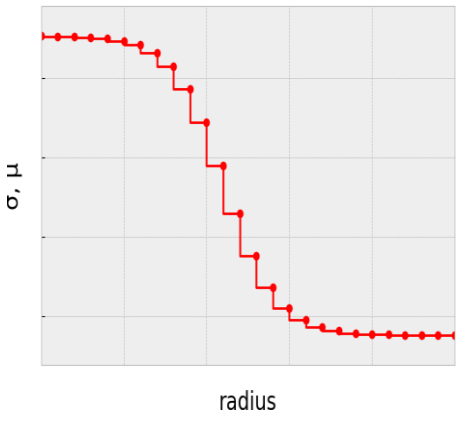
В [18] були запропоновані базові функції законів розподілу параметрів і наведено приклад їх застосування для апроксимації змін властивостей матеріалу приповерхневої зони товщиною 1 мм 50-ма умовними шарами. Для цього дослідження було обрано чотири типи апроксимаційних функцій [5], [18], які показано в табл.2.1. Окрім того, також на практиці профілі електрофізичних характеристик, які вважатимемо “нормою”, тобто взірцем, що отриманий внаслідок коректної технологічної обробки поверхні ОК одним із відомих способів, можуть бути визначені експериментально.

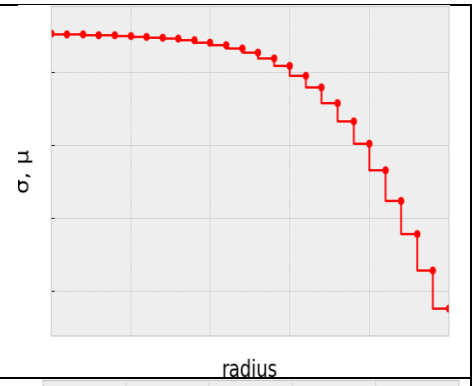
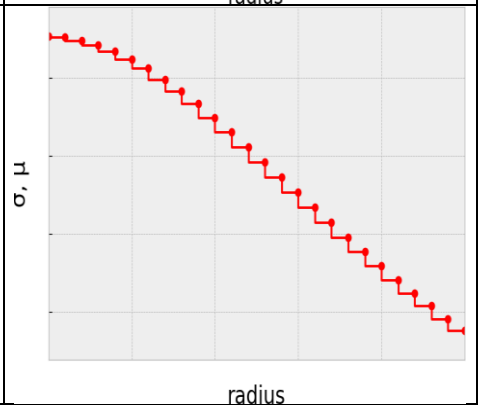
У табл.2.1 σ_1, μ_1; σ_2, μ_2 - початкові та кінцеві значення відповідних параметрів зони апроксимації; a, b, c, g, r - параметри,

що задають вигляд апроксимаційної моделі.

План експерименту може в себе включати наступні параметри: усі значення фізичних величин в апроксимаційних умовних шарах, частоту струму збудження та радіус ОК. Спростити заповнення планів експерименту можна варіюванням тільки кінцевими значеннями матеріальних параметрів приповерхневої зони, а проміжні набори значень формувати за допомогою обраної функції апроксимації. В свою чергу, застосування *R*-послідовностей при створенні таких планів експерименту з великою вірогідністю забезпечує рівномірне дослідження поверхні відгуку, що дозволяє отримувати вибірку з інформаційно високою цінністю даних.

Таблиця 2.1 - Апроксимаційні функції профілів електрофізичних параметрів

Вид апроксимації	Апроксимаційна модель	Графічне зображення
Гаусіан	$\sigma(r) = \sigma_1 + (\sigma_2 - \sigma_1)e^{\frac{-r^2}{g^2}}$ $\mu(r) = \mu_1 + (\mu_2 - \mu_1)e^{\frac{-r^2}{g^2}}$	
гіперболічний тангенс	$\sigma(r) = \sigma_1 + + \frac{\sigma_2 - \sigma_1}{2}\left(1 + \tanh\frac{r+c}{2a}\right)$ $\mu(r) = \mu_1 + + \frac{\mu_2 - \mu_1}{2}\left(1 + \tanh\frac{r+c}{2a}\right)$	

експоненційна функція	$\sigma(r) = \sigma_1 + (\sigma_2 - \sigma_1)e^{\frac{r}{b}}$ $\mu(r) = \mu_1 + (\mu_2 - \mu_1)e^{\frac{r}{b}}$	
степенева функція	$\sigma(r) = \sigma_1 + (\sigma_2 - \sigma_1)r^{-2}$ $\mu(r) = \mu_1 + (\mu_2 - \mu_1)r^{-2}$	

2.5. Експрес-метод розв'язання оберненої задачі вимірювання

Отже, на останньому етапі застосування методу передбачається класична однократна процедура вимірювань прохідним ВСП на фіксованій частоті збудження, причому саме тій, що використовувалася в процесі моделювання під час створення сурогатної моделі.

Початковою процедурою вимірювань профілів електрофізичних характеристик матеріалу є зняття та збереження значень амплітуди та фази ЕРС вимірювальної котушки трансформаторного прохідного ВСП, тобто значень модуля та аргументу комплекснозначної величини. Подальше віднайдення результату виконується за допомогою таблиць пошуку (Lookup tables, LUT) по амплітуді і фазі сигналу.

В загальному розумінні ключовою ідеєю такого підходу є абстрагування розв'язання складної моделі оберненої задачі у формі LUT. LUT генеруються симулятором для набору еталонних зразків, що має використовуватися один раз для кожного специфічного ОК. Потім можна використовувати LUT для пошуку швидкого розв'язку задачі без виклику симулятора, досягаючи при цьому результатів, співставлених з точністю симулятора [19]-[22].

Простіше кажучи, таблиця пошуку LUT - це масив, який містить набір попередньо обчислених результатів для вимірів профілів електрофізичних параметрів. Цей масив надає доступ до результатів швидше, ніж потребує обчислення кожного разу результату операції. Зазвичай LUT використовуються в системах збору та обробки даних у реальному часі, часто в вбудованих системах (embedded systems), оскільки такі типи систем є вимогливими до часових обмежень. Економія часу обробки може бути значною, тому що отримання значення з пам'яті часто швидше, ніж виконання ресурсозатратного обчислення.

В рамках задачі цього дослідження розглядається наступна імплементація LUT. Складовою методу є генерування множини екземплярів-кандидатів розв'язку, які різняться профілями в загальному випадку в залежності від *f*, *R*, ЕП та МП, та заповнення таблиці LUT. Такі кандидати на розв'язок мають бути попередньо підготовлені чисельним моделюванням за значно ширшим та детальним планом експерименту, що відрізняється суттєво більшим обсягом даних. Кандидатів необхідно генерувати з урахуванням властивих для конкретного технологічного процесу законів розподілу ЕП та МП в приповерхневому шарі ОК. Також при їх підготовці треба брати до уваги фактично можливий розкид параметрів σ_1, μr_1; σ_N, μr_N [23]. Підготовлені таким чином кандидати на розв'язок і становлять собою таблицю пошуку.

Метод є доволі універсальним внаслідок можливості внесення в перелік впливових факторів додаткових параметрів для накопичення апріорної інформації в сурогатній моделі. Зокрема таких, які сприяють ефективному проведенню вимірювань, наприклад, варіювання частотою збудження.

Список використаних джерел до глави 2

1. Ida, N., & Meyendorf, N. (Eds.). (2019). *Handbook of advanced nondestructive evaluation* (Vol. 10, pp. 978-3). Cham, Switzerland: Springer International Publishing. https://mz.zut.edu.pl/papers/2019/2019_HAN.pdf
2. Dorofeev, A. L. *Vihrevye toki[Eddy currents]*, in Russian. Moskva: Izdatel'stvo Jenergija, 1977. 72 p.
3. Dorofeev, A. L. *Indukcionnaja strukturoskopija[Induction*

structuroscopy], in Russian. Izdatel'stvo "Jenergija", 1973. 176 p.

4. Гальченко, В. Я., Тичков, В. В., Сторчак, А. В., & Трембовецька, Р. В. (2020). Відновлення приповерхневих радіальних профілів електрофізичних характеристик циліндричних об'єктів при вихрострумових вимірюваннях із наявністю апріорних даних. Формування вибірки для побудови сурогатної моделі. *Український метрологічний журнал/Ukrainian Metrological Journal*, (1), 35-50. https://doi.org/10.24027/2306-7039.1.2020.204226

5. Koliškina, V. (2014). Analytical and quasi-analytical solutions of direct problems in eddy current testing. https://dspace.lu.lv/handle/7/4555

6. Kolyshkin, A. A., & Vaillancourt, R. (1994). Analytical solution to eddy current testing of cylindrical problems with varying properties. *Canadian Applied Mathematics Quarterly*, *2*(3), 349-360.

7. Nestor Jr, C. W., Dodd, C. V., & Deeds, W. E. (1979). Analysis and computer programs for eddy current coils concentric with multiple cylindrical conductors. *NASA STI/Recon Technical Report N*, *80*, 15359. https://doi.org/10.2172/6145607

8. Dramsch, J. S., Lüthje, M., & Christensen, A. N. (2021). Complex-valued neural networks for machine learning on non-stationary physical data. *Computers & Geosciences*, *146*, 104643. https://doi.org/10.1016/j.cageo.2020.104643

9. Руденко, О. Г., & Бодянський, Є. В. (2006). *Штучні нейронні мережі*. Харків: Компанія СМІТ. 404 с.

10. Gulli, A., & Pal, S. (2017). *Deep learning with Keras*. Packt Publishing Ltd. https://books.google.nl/books?hl=uk&lr=&id=20EwDwAAQBAJ&oi=fnd&pg=PP1&dq=10.%09A.+Gulli+and+S.+Pal,+Deep+Learning+with+Keras&ots=lJez9i9TP7&sig=GgSdEpf03X7AzwbqNSZJ8j8LZnA&redir_esc=y#v=onepage&q=10.%09A.%20Gulli%20and%20S.%20Pal%2C%20Deep%20Learning%20with%20Keras&f=false

11. Гальченко, В. Я., Сторчак, А., Трембовецька, Р. В., & Тичков, В. В. (2020). Створення сурогатної моделі для відновлення приповерхневих профілів електрофізичних характеристик циліндричних об'єктів. *Український метрологічний журнал*, (3), 27-35. https://doi.org/10.24027/2306-7039.3.2020.216824

12. Halchenko, V. Y., Trembovetska, R. V., & Tychkov, V. V. (2019). Development of excitation structure RBF-metamodels of moving

concentric eddy current probe. *Электротехника и электромеханика*, (2 (eng)), 28-38. https://doi.org/10.20998/2074-272X.2019.2.05

13. Гальченко, В. Я., Трембовецька, Р. В., Тичков, В. В. (2018). Застосування нейрокомп'ютинга на етапі побудови метамоделей в процесі оптимального сурогатного синтезу антен. *Вісник НТУУ "КПІ". Серія Радіотехніка. Радіоапаратобудування*, № 74, с. 60—72. https://ela.kpi.ua/server/api/core/bitstreams/da9834be-45c8-4805-96fb-f861fca61a87/content .

14. Трембовецька, Р. В., Гальченко, В. Я., & Тичков, В. В. (2018). Побудова MLP-метамоделі накладного вихрострумового перетворювача для задач сурогатного оптимального синтезу. *Технічні вісті*, *47*(1-2), 27-31. https://er.chdtu.edu.ua/bitstream/ChSTU/462/1/27.pdf

15. Трембовецька, Р. В., Гальченко, В. Я., & Тичков, В. В. (2019). Multiparameter hybrid neural network metamodel of eddy current probes with volumetric structure of excitation system. In *Mathematical modeling: Theoretical foundations and specificity of mathematical modelling. Mathematical modelling of technological processes and systems: proceedings of the III International scientific conference, 11–14.12. 2019, Borovets, Bulgaria* (pp. 56-59). Mathematical modeling: Theoretical foundations and specificity of mathematical modelling. Mathematical modelling of technological processes and systems: proceedings of the III International scientific conference, 11–14.12. 2019, Borovets, Bulgaria. https://er.chdtu.edu.ua/handle/ChSTU/1106

16. ELsawah, A. M. (2014). *Constructing uniform experimental designs: in view of centered and wrap-around discrepancy*. LAP LAMBERT Academic Publishing. 168 p.

17. Halchenko, V., Trembovetska, R., Tychkov, V., & Storchak, A. V. (2020). The construction of effective multidimensional computer designs of experiments based on a quasi-random additive recursive Rd–sequence. *Applied Computer Systems*, *25*(1), 70-76. https://doi.org/10.2478/acss-2020-0009

18. Uzal, E. "Theory of eddy current inspection of layered metals", Ph.D. dissertation, 1992. https://www.proquest.com/openview/06972356f1b282d7831062f658520e1a/1?pq-origsite=gscholar&cbl=18750&diss=y

19. Youssef, A. A., Murmann, B., & Omran, H. (2020). Analog

IC design using precomputed lookup tables: Challenges and solutions. *IEEE Access*, *8*, 134640-134652. https://doi.org/10.1109/ACCESS.2020.3010875

20. Murmann, B. (2016). Systematic Design of Analog Circuits Using Pre-Computed Lookup Tables. https://ieeetoronto.ca/wp-content/uploads/2020/06/20160226toronto_sscs.pdf

21. Sabry, M. N., Omran, H., & Dessouky, M. (2018). Systematic design and optimization of operational transconductance amplifier using gm/ID design methodology. *Microelectronics journal*, *75*, 87-96. https://doi.org/10.1016/j.mejo.2018.02.002

22. Sabry, M. N., Nashaat, I., & Omran, H. (2020). Automated design and optimization flow for fully-differential switched capacitor amplifiers using recycling folded cascode OTA. *Microelectronics Journal*, *101*, 104814. https://doi.org/10.1016/j.mejo.2020.104814

23. Гальченко, В. Я., Сторчак, А., Тичков, В. В., & Трембовецька, Р. В. (2022). Вимірювання приповерхневих радіальних профілів електрофізичних характеристик циліндричних об'єктів вихрострумовим методом із застосуванням апріорних даних. *Український метрологічний журнал*, (1), 5-11. https://doi.org/10.24027/2306-7039.1.2022.258678

ГЛАВА 3
АЛГОРИТМІЧНЕ І ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ДЛЯ ВИМІРЮВАНЬ ПРОФІЛІВ ЕЛЕКТРОФІЗИЧНИХ ХАРАКТЕРИСТИК МЕТОДОМ З АПРІОРНИМ НАКОПИЧЕННЯМ ДАНИХ

3.1. Програмне забезпечення для “точного” моделювання процесів вихрострумового контролю об’єктів циліндричної форми

Зазвичай для побудови точних електрофізичних моделей використовуються числові методи моделювання, такі як, наприклад, метод скінченних елементів. В випадку вихрострумового контролю в цьому дослідженні через специфічну зміну електрофізичних параметрів ОК вздовж його радіусу в приповерхневому шарі на відносно малій ділянці використання таких інструментів для побудови моделі за допомогою числових методів практично не є реалістичним. Тому доводиться реалізовувати модель на базі аналітичного розв’язання рівнянь Максвелла.

Точна електродинамічна модель процесу вимірювання вихрострумовим перетворювачем електрофізичних параметрів циліндричних об’єктів контролю, яка описана в пункті 2.2, має ряд спеціальних математичних функцій та виразів з їх використанням, що потребують специфічних інструментів та прийомів їх обчислення. По своїй суті задача зводиться до обчислення невласного інтегралу і функціями Бесселя комплексних аргументів першого та другого роду різних порядків у підінтегральній функції.

Аналіз найкращих програмних засобів та застосунків, що можуть бути використані для розв’язання поставленої задачі, дозволив обрати як найефективніше програмне середовище Python. Цей вибір обумовлений набором готових необхідних математичних процедур, зручністю роботи з даними, гнучкістю у розробці, вводі-виводі інформації і візуалізації інформації, кросплатформністю та відкритим програмним кодом.

Відповідно до “точної” електродинамічної моделі для обчислення векторного потенціалу і напруги ВСП було складено програмне забезпечення, яке реалізовано за допомогою мови програмування Python 3 та з використанням бібліотек NumPy і SciPy. Для оптимізації моделювання деякі функції і підінтегральні

вирази з функціями Бесселя були замінені апроксимаціями. Інтегрування проводилося методом усічення з адаптивною межею та розбиттям діапазону інтегрування на частини. Також була використана логіка обчислення матричних елементів із відношеннями властивостей матеріалу в шарах ОК у відповідності до [1].

Верифікацію комплексу програм було проведено за допомогою аналітичних моделей для більш простих випадків, тобто двошарових ОК [2], [3], які дозволяють отримати значення векторного потенціалу в області розміщення вимірювального витка. Також з цією метою використовувався програмний продукт мультифізичного моделювання COMSOL Multiphysics (AC/DC Module), що застосовує для розрахунків FEM.

Спрощена тестова модель для двошарового ОК із немагнітного матеріалу [2] має вигляд:

$$A(r,z)=\frac{IN_d\mu_0}{\pi}\int_0^{\infty}\left\{\frac{K(r_{d2},r_{d1})}{\alpha^3}\left\{I_1(\alpha r)-\left[\frac{K_1(\alpha_2 r_2)}{r_2 D(\alpha)K_1(\alpha r_2)}\times\right.\right.\right.$$

$$\times\left(\alpha_1 I_1(\alpha_2 r_1)I_0(\alpha_1 r_1)-\alpha_2 I_1(\alpha_1 r_1)I_0(\alpha_2 r_1)\right)-\frac{I_1(\alpha_2 r_2)}{r_2 D(\alpha)K_1(\alpha r_2)}\times$$

$$\left.\left.\left.\times\left(\alpha_2 K_0(\alpha_2 r_1)I_1(\alpha_1 r_1)-\alpha_2 K_1(\alpha_2 r_1)I_0(\alpha_1 r_1)\right)+\frac{I_1(\alpha r_2)}{K_1(\alpha r_2)}\right]K_1(\alpha r)\times\right.\right.$$

$$\left.\left.\times\left[\sin(\alpha(z-l_{d1}))-\sin(\alpha(z-l_{d2}))\right]\right\}\right\}d\alpha,$$

$$D(\alpha)=\left[\alpha_2 K_0(\alpha_2 r_2)K_1(\alpha r_2)-\alpha K_0(\alpha r_2)K_1(\alpha_2 r_2)\right]\times$$

$$\times\left[\alpha_1 I_1(\alpha_2 r_1)I_0(\alpha_1 r_1)-\alpha_2 I_1(\alpha_1 r_1)I_0(\alpha_2 r_1)\right]+$$

$$+\left[\alpha_2 K_0(\alpha_2 r_1)I_1(\alpha_1 r_1)+\alpha_1 K_1(\alpha_2 r_1)I_0(\alpha_1 r_1)\right]\times$$

$$\times\left[\alpha I_1(\alpha_2 r_2)K_0(\alpha r_2)+\alpha_2 I_0(\alpha_2 r_2)K_1(\alpha r_2)\right].$$

Друга тестова модель [3] двошарового ОК з магнітного матеріалу представлена рівнянням:

$$A(r,z)=\frac{IN_d\mu_0}{\pi}\int_0^{\infty}\left\{\frac{K(r_{d2},r_{d1})}{\alpha^3}\left\{I_1(\alpha r)+S(\alpha)K_1(\alpha r)\right\}\times\right.$$
$$\left.\times\left[\sin(\alpha(z-l_{d1}))-\sin(\alpha(z-l_{d2}))\right]\right\}d\alpha,$$

$$D(\alpha)=\left[\alpha K_1'(\alpha r_2)K_1(\alpha_2 r_2)-\beta_2 K_1'(\alpha_2 r_2)K_1(\alpha r_2)\right]\times$$
$$\times\left[\beta_1 I_1(\alpha_2 r_1)I_1'(\alpha_1 r_1)-\beta_2 I_1(\alpha_1 r_1)I_1'(\alpha_2 r_1)\right]+$$
$$+\left[\beta_1 K_1(\alpha_2 r_1)I_1'(\alpha_1 r_1)-\beta_2 K_1'(\alpha_2 r_1)I_1(\alpha_1 r_1)\right]\times$$
$$\times\left[\beta_2 I_1'(\alpha_2 r_2)K_1(\alpha r_2)-\alpha I_1(\alpha_2 r_2)K_1'(\alpha r_2)\right],$$

$$S(\alpha)=\frac{1}{r_2 D(\alpha)K_1(\alpha r_2)}\left\{K_1(\alpha_2 r_2)\left[\beta_2 I_1(\alpha_1 r_1)I_1'(\alpha_2 r_1)-\right.\right.$$
$$-\beta_1 I_1(\alpha_2 r_1)I_1'(\alpha_1 r_1)\left.\left.\right]+I_1(\alpha_2 r_2)\left[\beta_1 K_1(\alpha_2 r_1)I_1'(\alpha_1 r_1)-\right.\right.$$
$$\left.\left.-\beta_2 K_1'(\alpha_2 r_1)I_1(\alpha_1 r_1)\right]-r_2 D(\alpha)I_1(\alpha r_2)\right\}.$$

Для верифікації з допомогою пакету COMSOL Multiphysics використовувалася модель, побудована у вісесиметричній системі координат. ЕП повітря регіону моделі (Air box) при розрахунках дорівнювала 1 См/м, оскільки це значно зменшує час розрахунку МСЕ, але суттєво не спотворює числовий результат. Загальний вигляд геометричної моделі показано на рис.3.1. Геометричні розміри цього регіону штучно обмежувалися та становили: радіус r_{ab} = 80 мм; відстані до обмежувальних границь l_{ab1} = 0 мм, l_{ab2} = 100 мм відповідно. Котушка збудження вважалася однорідною й багатовитковою (Homogenized multi-turn coil). Вимірювальна котушка розглядалась як функція Дірака.

В табл.3.1 наведено вихідні дані для розрахунків з використанням моделей немагнітного ОК, де μ_r - відносна МП *n*-го шару ОК.

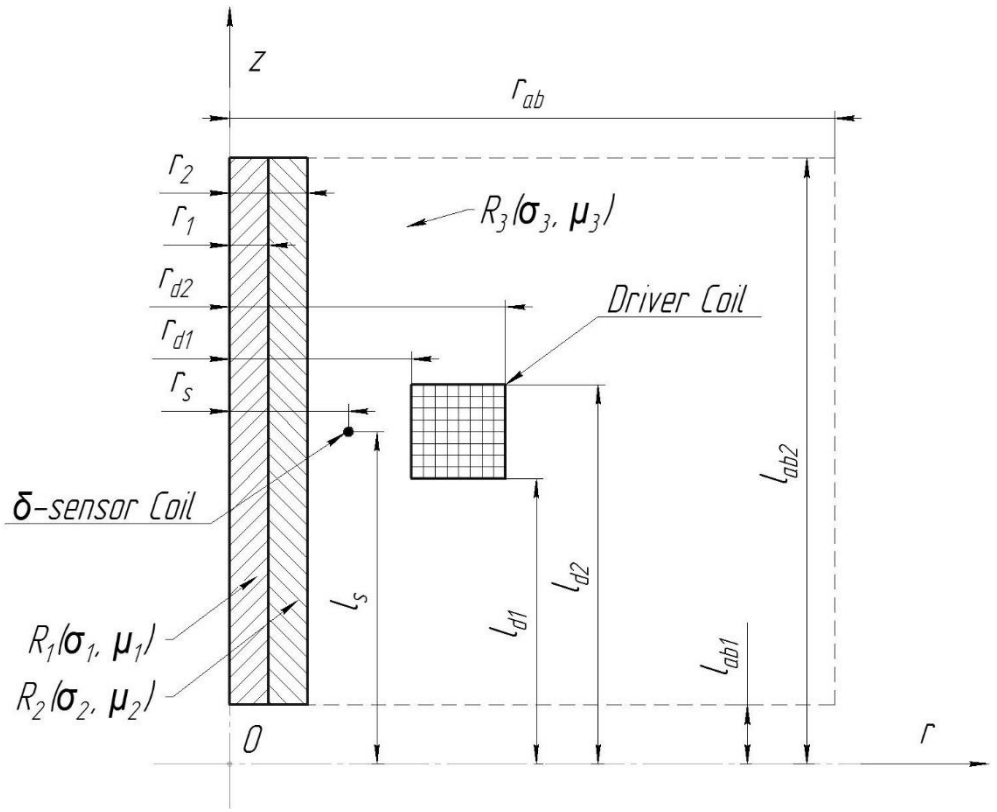


Рис. 3.1. Геометрична модель ВСП для моделювання в пакеті COMSOL Multiphysics.

Таблиця 3.1 - Вихідні дані для моделей немагнітного ОК

r_{d1}, мм	r_{d2}, мм	l_{d1}, мм	l_{d2}, мм	r_s, мм	l_s, мм	r_1, мм	r_2, мм	σ_1, См/м	σ_2, См/м	μ_{r_1}	μ_{r_2}	W	I, А
16	19	47,5	52,5	13,5	50	9	10	$3{,}766 \cdot 10^7$	$1{,}88 \cdot 10^7$	1	1	100	1

Результати обчислення векторного потенціалу для чотирьох випадків використання вищевказаних моделей наведено в табл.3.2.

Таблиця 3.2 - Результати обчислень дійсної та уявної частин векторного потенціалу для немагнітного ОК

Частота, Гц	Векторний потенціал А, Вб/м							
	модель [2]		модель [3]		модель [1], [4]		модель МСЕ	
	Re A $\cdot 10^{-5}$	Im A $\cdot 10^{-6}$	Re A $\cdot 10^{-5}$	Im A $\cdot 10^{-6}$	Re A $\cdot 10^{-5}$	Im A $\cdot 10^{-6}$	Re A $\cdot 10^{-5}$	Im A $\cdot 10^{-6}$
100	2,89007	-2,41258	2,89007	-2,41258	2,89007	-2,41258	2,86060	-2,34160
200	2,72492	-3,48933	2,72492	-3,48933	2,72492	-3,48933	2,70060	-3,40250

300	2,59301	-3,69192	2,59301	-3,69192	2,59301	-3,69192	2,57190	-3,61150
…	…	…	…	…	…	…	…	…
2400	2,16979	-2,28339	2,16979	-2,28339	2,16979	-2,28339	2,15530	-2,25060
2500	2,16370	-2,25452	2,16370	-2,25452	2,16370	-2,25452	2,14930	-2,22240
2600	2,15794	-2,22722	2,15794	-2,22722	2,15794	-2,22722	2,14360	-2,19570
…	…	…	…	…	…	…	…	…
4800	2,07383	-1,82864	2,07383	-1,82864	2,07383	-1,82864	2,06060	-1,80640
4900	2,07157	-1,81796	2,07157	-1,81796	2,07157	-1,81796	2,05830	-1,79590
5000	2,06936	-1,80756	2,06936	-1,80756	2,06936	-1,80756	2,05610	-1,78580

Результати обчислень свідчать про досить високу точність розрахунків, при яких максимальна відносна похибка для дійсної та уявної частин векторного потенціалу не перевищує 3% (1,7% в області частот вище 1 кГц), що дозволяє вважати програмну модель, котра підлягає верифікації, адекватною.

Вихідні дані для верифікації моделі слабомагнітного ОК наведено в табл.3.3.

Таблиця 3.3 - Вихідні дані для моделей слабомагнітного ОК

r_{d1}, мм	r_{d2}, мм	l_{d1}, мм	l_{d2}, мм	r_s, мм	l_s, мм	r_1, мм	r_2, мм	σ_1, См/м	σ_2, См/м	μr_1	μr_2	W	I, А
16	19	47,5	52,5	13,5	50	9	10	$6{,}99 \cdot 10^6$	$3{,}495 \cdot 10^6$	25	15	100	1

Порівняльний аналіз результатів обчислення векторного потенціалу на основі моделі [1], [4] та моделі МСЕ можливий з урахуванням даних табл.3.4.

Таблиця 3.4 - Результати обчислень дійсної та уявної частин векторного потенціалу для слабомагнітного ОК

Частота, Гц	Векторний потенціал А, Вб/м			
	модель [1], [4]		модель МСЕ	
	Re A $\cdot 10^{-5}$	Im A $\cdot 10^{-5}$	Re A$\cdot 10^{-5}$	Im A $\cdot 10^{-5}$
100	8,36504	-2,77174	7,89160	-1,94760
200	7,05051	-2,50664	6,83770	-2,00940
300	6,40149	-2,33445	6,26320	-1,97350

…	…	…	…	…
2400	3,92757	-1,38956	3,88870	-1,33390
2500	3,89292	-1,37199	3,85430	-1,31850
2600	3,86010	-1,35520	3,82160	-1,30370
…	…	…	…	…
4800	3,40407	-1,10740	3,36550	-1,07590
4900	3,39051	-1,09961	3,35200	-1,06850
5000	3,37732	-1,09202	3,33880	-1,06120

Максимальна похибка обчислень з використанням моделі [1], [4] є дещо більшою ніж для попереднього випадку немагнітного ОК та становить 7,6% в області частот вище 1 кГц, але і в цьому випадку з магнітним ОК точність результатів можна вважати прийнятною для розрахунків. Графік розбіжностей чисельних значень дійсної та уявної частин векторного потенціалу (табл. 3.4) в залежності від частоти збудження для наочності показано на рис.3.2 та 3.3.

Аналіз даних, наведених на цих рисунках, свідчить щодо більшої похибки в області частот нижчих за 1 кГц, які практично майже не використовуються при вихрострумових вимірюваннях.

Верифікацію обчислень напруги ВСП немає сенсу проводити, так як вона напряму залежить від точності розрахунку векторного потенціалу відповідно до формули 2.4.

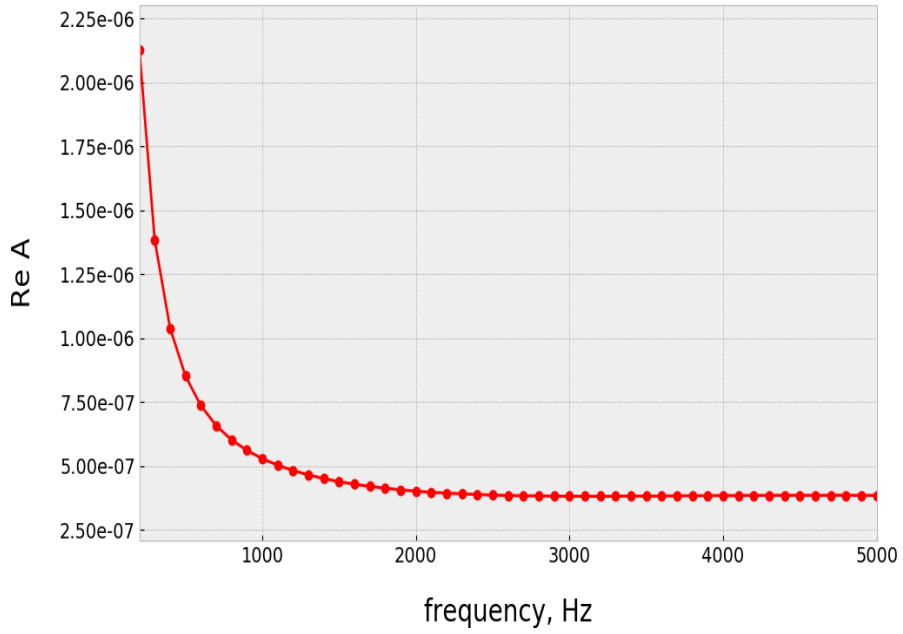


Рис. 3.2. Абсолютна похибка розрахунку значень дійсної складової векторного потенціалу для слабомагнітного ОК.

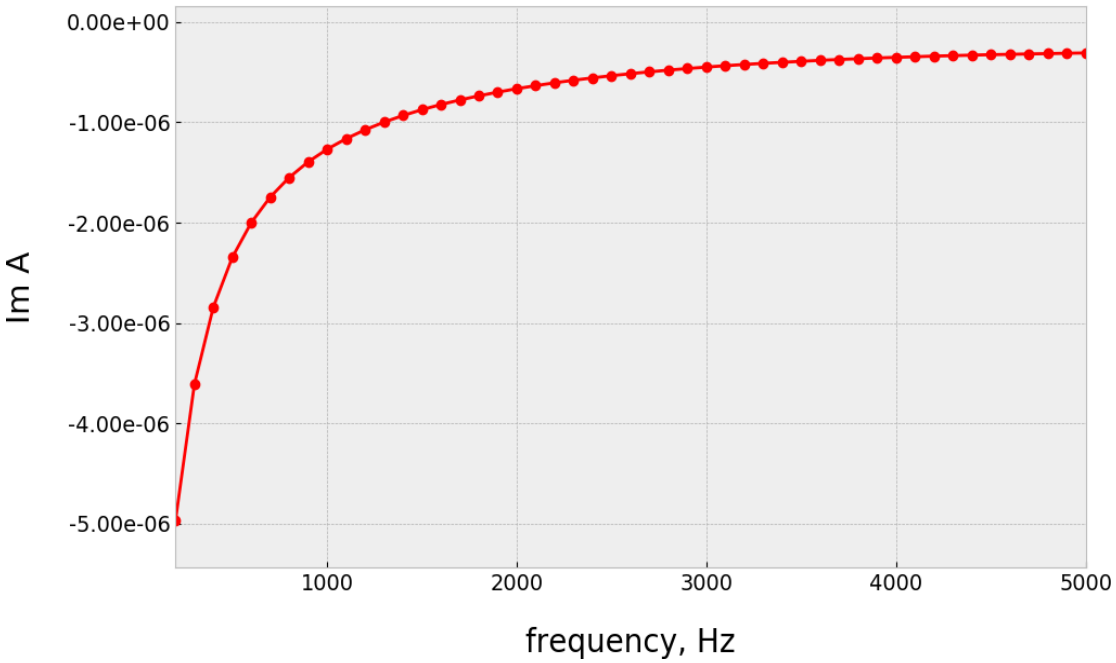


Рис. 3.3. Абсолютна похибка розрахунку значень уявної складової векторного потенціалу для слабомагнітного ОК.

Отже, запропоновану математичну модель та створений комплекс програм її реалізації можна вважати придатними для подальших чисельних експериментів, а можливість варіювати електрофізичними характеристиками умовних шарів і їх товщиною дозволяє з легкістю обчислювати наведену ЕРС вихрострумового перетворювача.

3.2. Програмне забезпечення для створення комп'ютерних однорідних планів експериментів

Для формування коректного плану експерименту спочатку варто визначитися з конкретними параметрами, які будуть змінюватися при розрахунках за цими планами. Так як вважається, що приповерхневий матеріал ОК має неперервну зміну електрофізичних параметрів уздовж його радіусу $\sigma = \sigma(r)$ та $\mu = \mu(r)$ в зоні приповерхневого шару (рис. 3.4), запропоновано наступний концепт побудови плану: електрофізичні характеристики глибинного матеріалу ОК вважалися сталими і задавалися як характеристики 1-го умовного шару, а приповерхневий матеріал був представлений сукупністю 50-ти апроксимаційних умовних шарів з 2-го по 51-й. Отже, зміна їхніх матеріальних характеристик відбувалася у відповідності до певної апроксимізаційної функції. Вибір розбиття приповерхневої зони матеріалу на 50 шарів є

умовним, що забезпечує доволі точну апроксимацію реального розподілу його властивостей, а збільшення цього числа не призводить до якоїсь видимої зміни результатів обчислення. При цьому більш ґрунтовна дискретизація приповерхневої зони для апроксимації змін електрофізичних властивостей матеріалу негативно впливає на швидкість обчислення результату за “точною” моделлю та збільшує розмірність сурогатної моделі, що є не бажаним [5].

На рис.3.5 зображено приклад радіального профілю електричної провідності матеріалу приповерхневого шару ОК, отриманого з використанням функції гіперболічного тангенса для апроксимації.

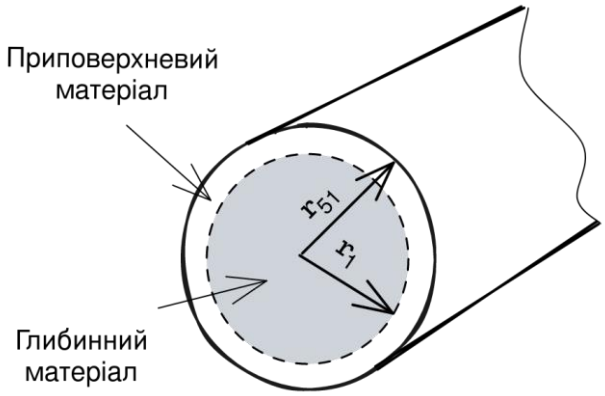


Рис. 3.4. Представлення приповерхневого та глибинного матеріалу циліндричного ОК.

Аналогічним чином, як показано у прикладі вище з ЕП, задавалися значення МП матеріалу ОК. За цим підходом набір значень МП і ЕП, тобто $\vec{\sigma} = [\sigma_1, \sigma_2, ..., \sigma_{51}]$ та $\vec{\mu} = [\mu_1, \mu_2, ..., \mu_{51}]$ розглядається, як набір вхідних даних моделі $U = F(\vec{\sigma}, \vec{\mu}, f, r, ...)$. Опорними змінними, що визначаються видом застосованої апроксимаційної залежності для опису розподілів властивостей матеріалу ОК, є тільки граничні σ_1, σ_{51} і μ_1, μ_{51}, значення яких залавалися при моделюванні, а всі проміжні значення електрофізичних характеристик обчислювалися у відповідності до функціональних залежностей, що описані в пункті 2.4.

Для того, щоб коректно спланувати наступні числові експерименти з визначення можливості розрізнення профілів електрофізичних параметрів ОК, також варто визначитися з

вибором частоти струму збудження. З цією метою для наведеного прикладу було проведено низку числових експериментів з наступним порівняльним аналізом їх результатів, які дозволяють визначитися з оптимальною частотою.

В табл.3.5 наведено вихідні дані для розрахунків з використанням моделей немагнітного ОК, де μ_r - відносна МП n-го шару ОК. На рис.3.6 зображено графіки зміни напруги, індукованої у вимірювальному витку ВСП, в залежності від зміни виду профілів ЕП. Інші базові вхідні дані для моделювання бралися з табл.3.5.

На рис.3.7 зображено залежності зміни напруги, індукованої у вимірювальному витку ВСП, від зміни віиду профілів МП ОК. Відносна МП першого та останнього шарів ОК була фіксованою, а для шарів з 2-го по 50-й включно задавалася змінною відповідно до однієї з функцій апроксимації профілю в визначеному діапазоні. Інші базові вхідні дані для моделювання бралися з табл.3.6.

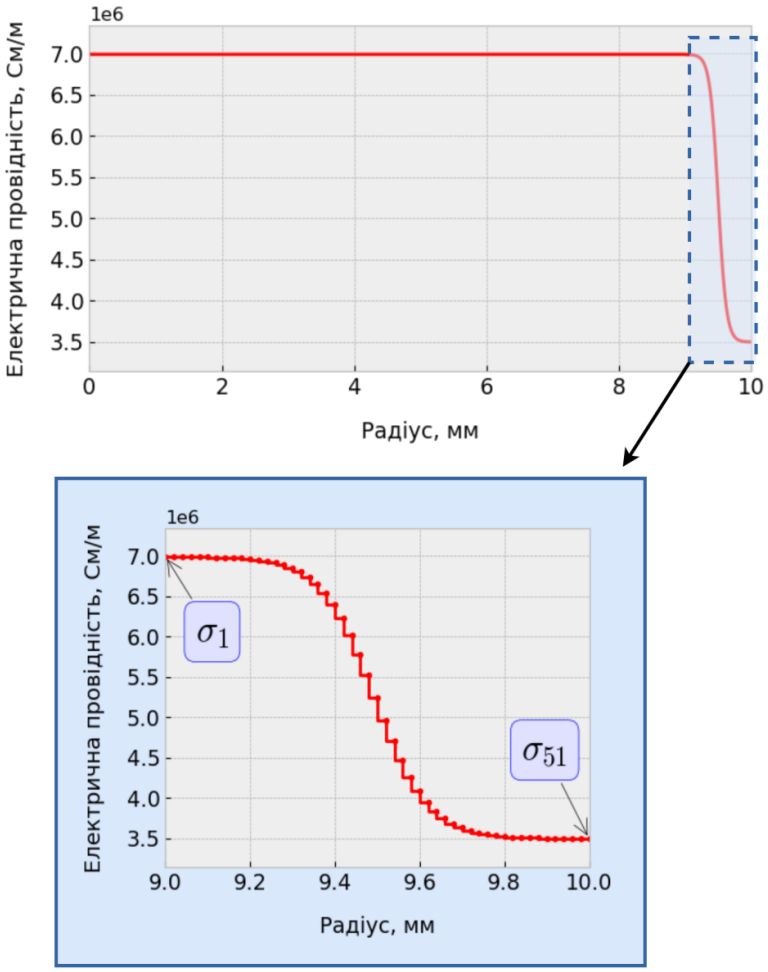


Рис. 3.5. Приклад апроксимації неперервної зміни електрофізичних параметрів приповерхневого матеріалу ОК.

На рис.3.8 зображено характер зміни напруги, індукованої у вимірювальному витку ВСП, внаслідок зміни виду профілів провідності та магнітної проникності ОК одночасно. ЕП та відносна МП першого та останнього шарів ОК були фіксованими, а для шарів з 2-го по 50-й включно задавалися змінними відповідно до однієї з видів функцій апроксимації профілів. ЕП змінювалася в зазначених діапазонах. Інші базові вхідні дані для моделювання бралися з табл.3.6.

Таблиця 3.5 - Вихідні дані для моделі немагнітного ОК

r_{d1}, мм	r_{d2}, мм	l_{d1}, мм	l_{d2}, мм	r_s, мм	l_s, мм	r_1, мм	r_2, мм	σ_{51}, См/м	σ_1, См/м	μr_{51}	μr_1	W	I, А
16	19	47,5	52,5	13,5	50	9	10	$3{,}766 \cdot 10^7$	$1{,}88 \cdot 10^7$	1	1	100	1

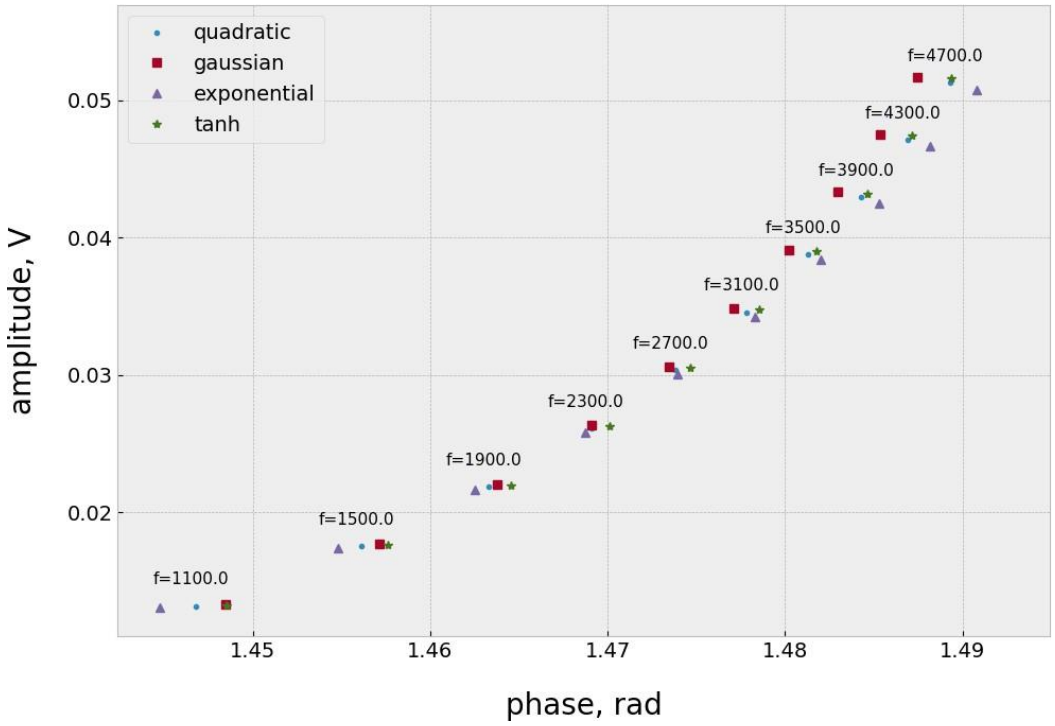


Рис. 3.6. Індукована напруга ВСП при різних видах профілів провідності для немагнітного ОК.

Аналіз результатів чисельних експериментів показав можливість вибору частоти збудження ВСП, на якій найкраще проявляється розрізнення профілів розподілу електрофізичних параметрів матеріалу циліндричних виробів. Отже, було обрано частоту 2,5 кГц як раціональну, бо саме на цій частоті

спостерігається найбільша різниця амплітуди та фази сигналу при всіх вище перелічених числових експериментах.

Таблиця 3.6 - Вихідні дані для моделей слабомагнітного ОК

r_{d1}, мм	r_{d2}, мм	l_{d1}, мм	l_{d2}, мм	r_s, мм	l_s, мм	r_1, мм	r_2, мм	σ_1, См/м	σ_2, См/м	μr_{51}	μr_1	w	I, А
16	19	47,5	52,5	13,5	50	9	10	$6{,}99 \cdot 10^6$	$3{,}495 \cdot 10^6$	25	15	100	1

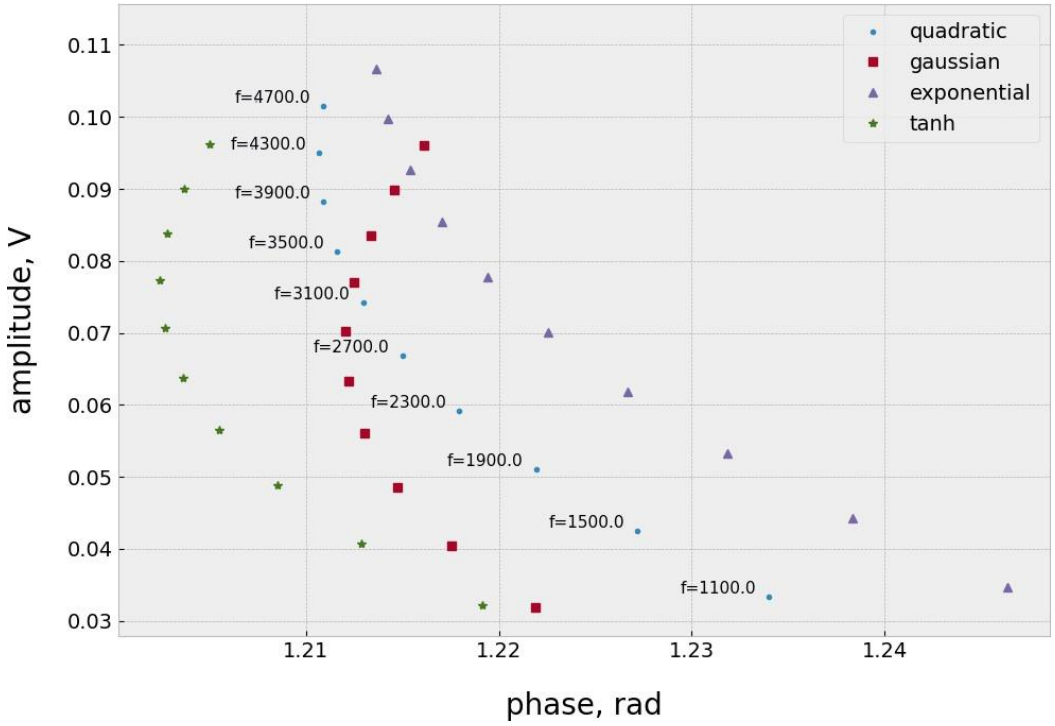


Рис. 3.7. Індукована напруга ВСП при різних видах профілів магнітної проникності для слабомагнітного ОК.

Як зазначено в пункті 2.4, для створення навчальної вибірки доцільно використовувати *R*-послідовності Робертса. Вони мають низьку розбіжність при малій вибірці та різній розмірності простору проєктування. При цьому не вимагають ретельного підбору послідовностей для кожного виміру простору для підтримки низької розбіжності для багатофакторних планів.

В зазначених рамках дослідження для формування комп'ютерного плану експерименту з накопиченням різної апріорної інформації щодо процесу вимірювань можна виділити такі варіативні параметри: опорні значення ЕП σ_{51} та МП μ_{51} із врахуванням розкиду від норми, частоту струму збудження f та радіус ОК r. Основні комбінації яких мають вигляд:

1) $U = F(\sigma_{51}, \mu_{51});$

2) $U = F(\sigma_{51}, \mu_{51}, f);$

3) $U = F(\sigma_{51}, \mu_{51}, f, r).$

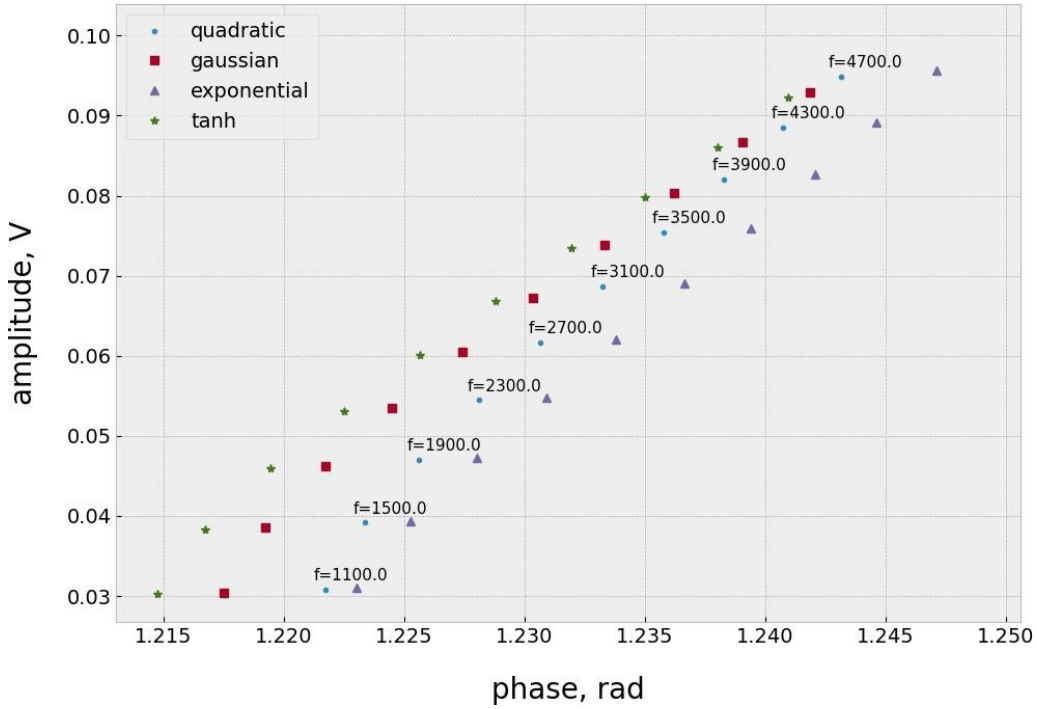


Рис. 3.8. Індукована напруга ВСП при різних видах профілів провідності та магнітної проникності для слабомагнітного ОК.

Скорочені приклади таких наборів в одиничному гіперкубі представлено в таблицях 3.7, 3.8, де $d_1, d_2, ..., d_n$ - номер виміру в просторі.

Таблиця 3.7 - Приклад двофакторного плану експерименту на *R*-послідовностях в одиничному квадраті

№	d_1	d_2
1	0,2549	0,0698
2	0,0098	0,6397
3	0,7646	0,2095
4	0,5195	0,7794
5	0,2744	0,3492
…	…	…
2498	0,1844	0,961
2499	0,9392	0,5308

2500	0,6941	0,1006
2501	0,449	0,6705
2502	0,2039	0,2403
…	…	…
4996	0,8687	0,4219
4997	0,6236	0,9918
4998	0,3785	0,5616
4999	0,1333	0,1315
5000	0,8882	0,7013

Таблиця 3.8 - Приклад трифакторного плану експерименту на *R*-послідовностях в одиничному кубі

№	d_1	d_2	d_3
1	0,3192	0,171	0,0497
2	0,1383	0,8421	0,5994
3	0,9575	0,5131	0,1491
4	0,7767	0,1842	0,6988
5	0,5959	0,8552	0,2485
…	…	…	…
2498	0,7929	0,7669	0,6518
2499	0,6121	0,438	0,2015
2500	0,4313	0,109	0,7512
2501	0,2505	0,7801	0,3009
2502	0,0696	0,4511	0,8506
…	…	…	…
4996	0,0859	0,0339	0,8036
4997	0,905	0,7049	0,3533
4998	0,7242	0,3759	0,903
4999	0,5434	0,047	0,4527
5000	0,3626	0,718	0,0024

Таблиця 3.9 - Приклад чотирифакторного плану експерименту на *R*-послідовностях в одиничному гіперкубі

№	d_1	d_2	d_3	d_4
1	0,3567	0,2339	0,1287	0,0386
2	0,2133	0,9678	0,7574	0,5772
3	0,07	0,7017	0,3861	0,1158
4	0,9267	0,4356	0,0148	0,6544
5	0,7834	0,1695	0,6435	0,193
…	…	…	…	…
2498	0,4739	0,7619	0,0094	0,9159
2499	0,3305	0,4957	0,6381	0,4545
2500	0,1872	0,2296	0,2668	0,9931
2501	0,0439	0,9635	0,8955	0,5317
2502	0,9006	0,6974	0,5242	0,0703
…	…	…	…	…
4996	0,4477	0,0237	0,5188	0,3319
4997	0,3044	0,7576	0,1475	0,8705
4998	0,1611	0,4915	0,7762	0,4091
4999	0,0177	0,2254	0,4049	0,9477
5000	0,8744	0,9593	0,0336	0,4863

Якість характеристик однорідності планів на зазначених послідовностях демонструється центрованою розбіжністю $(CD(D_n))^2$ та циклічною розбіжністю $(WD(P))^2$. Числові значення для наборів *R*-послідовностей з різною кількістю вимірів простору *d* та кількістю значень в вимірі *n* наведено в таблиці 3.10.

Таблиця 3.10 - Характеристики розбіжностей планів на *R*-послідовностях при різному наборі розмірностей

d	n	$(CD(D_n))^2$	$(WD(P))^2$
2	1000	4,2152E-06	3,5556E+00
2	2500	6,3493E-07	3,5556E+00
2	5000	2,8499E-07	3,5556E+00
3	1000	4,3859E-05	4,7408E+00
3	2500	1,5454E-05	4,7408E+00

3	5000	1,3425E-05	4,7408E+00
4	1000	2,8774E-04	6,3214E+00
4	2500	8,0705E-05	6,3211E+00
4	5000	3,9753E-06	6,3210E+00
5	1000	2,3086E-04	8,4284E+00
5	2500	1,4915E-04	8,4282E+00
5	5000	1,0777E-04	8,4282E+00
6	1000	5,0100E-04	1,1239E+01
6	2500	2,6120E-04	1,1238E+01
6	5000	4,1721E-05	1,1237E+01
7	1000	5,6472E-04	1,4985E+01
7	2500	2,1451E-04	1,4984E+01
7	5000	9,1620E-05	1,4983E+01
8	1000	9,4584E-04	1,9980E+01
8	2500	2,9802E-04	1,9978E+01
8	5000	1,5428E-04	1,9978E+01
9	1000	1,1893E-03	2,6642E+01
9	2500	3,1416E-04	2,6638E+01
9	5000	1,3303E-04	2,6637E+01
10	1000	2,1625E-03	3,5526E+01
10	2500	7,1506E-04	3,5519E+01
10	5000	2,6856E-04	3,5517E+01

Для прикладу масштабування даних для реальних значень факторів взято базові значення з таблиці 3.11 і виконано розрахунки на основі планів із наборів *R*-послідовностей в одиничному гіперкубі з наведених вище таблиць та прийнятних значень можливих відхилень факторів.

Таблиця 3.11 - Вихідні дані для створення масштабованих планів експерименту

	σ_{51}, См/м	μr_{51}	f, Гц	r, мм
Базове значення (BaseValue)	6990000	10	2500	10
Максимальне відносне відхилення, %	15,00	15,00	20,00	2,50

Абсолютне значення максимального відхилення (DeltaMax)	1048500	1,5	500	0,25
Нижня границя відхилення (BaseValue-DeltaMax)	5941500	8,5	2000	9,75
Верхня границя відхилення (BaseValue+DeltaMax)	8038500	11,5	3000	10,25

Таблиця 3.12 - Приклад масштабованого двофакторного плану експерименту

№	σ_{51}, См/м	μr_{51}
1	6,4852E+06	8,7095E+00
2	5,9705E+06	1,0419E+01
3	7,5557E+06	9,1286E+00
4	7,0410E+06	1,0838E+01
5	6,5262E+06	9,5476E+00
…	…	…
2498	6,3372E+06	1,1383E+01
2499	7,9224E+06	1,0092E+01
2500	7,4076E+06	8,8019E+00
2501	6,8929E+06	1,0511E+01
2502	6,3781E+06	9,2210E+00
…	…	…
4997	7,2595E+06	1,1475E+01
4998	6,7448E+06	1,0185E+01
4999	6,2300E+06	8,8944E+00
5000	7,8153E+06	1,0604E+01

Таблиця 3.13 - Приклад масштабованого трифакторного плану експерименту

№	σ_{51}, См/м	μr_{51}	f, Гц
1	6,6203E+06	9,0131E+00	2,0497E+03
2	6,2405E+06	1,1026E+01	2,5994E+03
3	7,9608E+06	1,0039E+01	2,1491E+03

4	7,5810E+06	9,0525E+00	2,6988E+03
5	7,2013E+06	1,1066E+01	2,2485E+03
…	…	…	…
2498	7,6152E+06	1,0801E+01	2,6518E+03
2499	7,2354E+06	9,8139E+00	2,2015E+03
2500	6,8557E+06	8,8270E+00	2,7512E+03
2501	6,4760E+06	1,0840E+01	2,3009E+03
2502	6,0962E+06	9,8533E+00	2,8506E+03
…	…	…	…
4997	7,8506E+06	1,0615E+01	2,3533E+03
4998	7,4709E+06	9,6278E+00	2,9030E+03
4999	7,0911E+06	8,6409E+00	2,4527E+03
5000	6,7114E+06	1,0654E+01	2,0024E+03

Таблиця 3.14 - Приклад масштабованого чотирифакторного плану експерименту

№	σ_{51}, См/м	μr_{51}	f, Гц	r, мм
1	6,4852E+06	8,7095E+00	6,4852E+06	8,7095E+00
2	5,9705E+06	1,0419E+01	5,9705E+06	1,0419E+01
3	7,5557E+06	9,1286E+00	7,5557E+06	9,1286E+00
4	7,0410E+06	1,0838E+01	7,0410E+06	1,0838E+01
5	6,5262E+06	9,5476E+00	6,5262E+06	9,5476E+00
…	…	…	…	…
2498	6,3372E+06	1,1383E+01	6,3372E+06	1,1383E+01
2499	7,9224E+06	1,0092E+01	7,9224E+06	1,0092E+01
2500	7,4076E+06	8,8019E+00	7,4076E+06	8,8019E+00
2501	6,8929E+06	1,0511E+01	6,8929E+06	1,0511E+01
2502	6,3781E+06	9,2210E+00	6,3781E+06	9,2210E+00
…	…	…	…	…
4996	7,7743E+06	9,7658E+00	7,7743E+06	9,7658E+00
4997	7,2595E+06	1,1475E+01	7,2595E+06	1,1475E+01
4998	6,7448E+06	1,0185E+01	6,7448E+06	1,0185E+01
4999	6,2300E+06	8,8944E+00	6,2300E+06	8,8944E+00
5000	7,8153E+06	1,0604E+01	7,8153E+06	1,0604E+01

На основі масштабованих даних з використанням опорних значень МП і ЕП формуються проміжні значення електрофізичних параметрів в умовних шарах приповерхневого матеріалу з застосуванням функціональних апроксимаційних залежностей.

Після цього набори даних є придатними до чисельних експериментів щодо створення з застосуванням “точної” електродинамічної моделі процесу вимірювань навчальної вибірки для навчання сурогатної моделі.

3.3. Програмні засоби створення сурогатної моделі

Вибір програмного засобу для створення сурогатної моделі на основі DNN не тільки залежить від конкретної задачі, особливостей даних та вимог до швидкості обрахунку та точності моделі, а й від затрат часу на прототипування та зручності використання цих засобів. Також варто враховувати наявність документації та підтримки спільнотою користувачів і розробників вибраного програмного засобу, оскільки це може значно полегшити розробку та підтримку засобів реалізації моделі у майбутньому.

Створення сурогатної моделі виконувалося в середовищі програмування Python 3 з використанням відкритої бібліотеки Keras (Tensorflow backend). При числових експериментах нейронні мережі з найкращою продуктивністю було отримано при навчанні із використанням оптимізаційного методу Adam. Це є метод стохастичного градієнтного спуску, який базується на адаптивній оцінці моментів першого та другого порядку. Цей метод є ефективним в обчислювальному відношенні та потребує незначних ресурсів пам’яті [6].

Приклад вхідних даних для створення сурогатної моделі, отриманих за методикою, що викладено у пункті 3.2, наведено у табл.3.6. Індукована напруга $\hat{y} = U$ в вимірювальному витку прохідного ВСП була обрахована при сталій частоті струму збудження 2,5 кГц. При створенні сурогатної моделі є зручним використання значення напруги ВСП у представленні комплексним числом в алгебраїчній формі [7]. Дані сформованої вибірки попередньо були довільно розділені на три частини: навчальну (64%), тестову (16%), контрольну (20%).

В результаті створення сурогатної моделі отримано глибокі нейронні мережі відповідно для дійсної та уявної складових

вихідного сигналу ВСП із однаковими структурами 102-256(Sigmoid)-128(Sigmoid)-64(Sigmoid)-1. В яких Sigmoid - логістична функція активації нейронів. Варто зазначити, що використання альтернативної функції ReLU, яка вважається оптимальною для такого типу задач, давало схожі результати. ReLU (Rectified Linear Unit) –- функція активації, яка реалізує простий пороговий перехід в нулі, тобто повертає значення аргументу, якщо він позитивний, і 0 в іншому випадку. Об'єктивні чисельні показники, що характеризують обидві нейромережі на тестовій вибірці, мають наступні значення продуктивності для дійсної складової значення напруги 96,05% та для уявної складової - 95,94%. MSE нейронних мереж для розрахунку дійсної та уявної складових при нормованих значеннях становить відповідно 0,00044784 і 0,00019249.

Також створені моделі перевірялися на адекватність розрахунком коефіцієнту детермінації R^2 за формулою (3.1) та обчисленням середньої абсолютної відсоткової помилки (Mean Absolute Percentage Error, MAPE):

$$R^2 = 1 - \frac{\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2}{\sum_{i=1}^{n}\left(y_i - \bar{y}_i\right)^2}, \tag{3.1}$$

$$MAPE = \left(\frac{100\%}{n}\right)\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{3.2}$$

де y - бажаний вихід моделі;

$\hat{y}$ - вихідні значення сурогатної моделі;

$\bar{y}$ - середнє значення спостережуваних даних;

n - кількість зразків даних для перевірки.

Так значення коефіцієнту детермінації R^2 становлять 0,999578 та 0,9997864, в свою чергу MAPE дорівнює 0,039814% і 0,026852% для сурогатних моделей обчислення дійсної та уявної складових напруги відповідно.

Результати верифікаційних обчислень з використанням створених сурогатних моделей на даних контрольної вибірки, яка взагалі не використовувалася на етапі їх побудови, наведено в

табл.3.15, табл.3.16 та табл.3.17.

Таблиця 3.15 - Вхідні значення параметрів електричної провідності ОК для тестування відтворення вихідного сигналу $\hat{y} = U$ прохідного ВСП на даних контрольної вибірки

№	σ_1, См/м	σ_2, См/м	…	σ_{51}, См/м
1	6,9900E+06	6,9884E+06	…	3,4950E+06
2	6,9900E+06	6,9885E+06	…	3,6916E+06
3	6,9900E+06	6,9882E+06	…	3,0035E+06
4	6,9900E+06	6,9883E+06	…	3,3312E+06
5	6,9900E+06	6,9884E+06	…	3,5933E+06
…	…	…	…	…
996	6,9900E+06	6,9882E+06	…	3,0771E+06
997	6,9900E+06	6,9882E+06	…	3,1426E+06
998	6,9900E+06	6,9885E+06	…	3,6669E+06
999	6,9900E+06	6,9882E+06	…	3,0812E+06
1000	6,9900E+06	6,9884E+06	…	3,4744E+06

Таблиця 3.16 - Вхідні значення параметрів магнітної проникності ОК для тестування відтворення вихідного сигналу $\hat{y} = U$ прохідного ВСП на даних контрольної вибірки

№	μr_1	μr_2	…	μr_{51}
1	1,0000E+00	1,0042E+00	…	1,0000E+01
2	1,0000E+00	1,0042E+00	…	1,0188E+01
3	1,0000E+00	1,0040E+00	…	9,7188E+00
4	1,0000E+00	1,0039E+00	…	9,5313E+00
5	1,0000E+00	1,0036E+00	…	8,7813E+00
…	…	…	…	…
996	1,0000E+00	1,0036E+00	…	8,6959E+00
997	1,0000E+00	1,0035E+00	…	8,5084E+00
998	1,0000E+00	1,0042E+00	…	1,0008E+01
999	1,0000E+00	1,0048E+00	…	1,1286E+01
1000	1,0000E+00	1,0046E+00	…	1,0911E+01

Таблиця 3.17 - Результати тестування вихідного сигналу $\hat{y} = U$ прохідного ВСП на даних контрольної вибірки

№	Тестове значення Re(U) вимірювального витка ВСП	Відтворене значення Re(U) вимірювального витка ВСП	Тестове значення Im(U) вимірювального витка ВСП	Відтворене значення Im(U) вимірювального витка ВСП
1	5,1486E-03	5,1499E-03	4,4372E-02	4,4374E-02
2	5,2425E-03	5,2461E-03	4,4522E-02	4,4533E-02
3	4,9370E-03	4,9376E-03	4,4208E-02	4,4198E-02
4	5,0447E-03	5,0424E-03	4,3874E-02	4,3864E-02
5	5,0703E-03	5,0691E-03	4,2887E-02	4,2871E-02
…	…	…	…	…
996	4,8917E-03	4,8867E-03	4,2948E-02	4,2933E-02
997	4,9011E-03	4,8976E-03	4,2698E-02	4,2701E-02
998	5,2143E-03	5,2167E-03	4,4322E-02	4,4326E-02
999	5,0986E-03	5,0982E-03	4,6014E-02	4,6032E-02
1000	5,2323E-03	5,2348E-03	4,5437E-02	4,5446E-02

Порівняльний аналіз отриманих результатів свідчить щодо досить точної апроксимації нелінійних поверхонь відгуку та відповідну адекватність побудованих дійснозначних сурогатних моделей.

На рис.3.9 - 3.12 ілюструються діаграми розсіювання та гістограми розподілу абсолютних похибок відповідно для дійсної й уявної складових значень індукованої в вимірювальному витку ВСП напруги, що отримані з використанням створених сурогатних моделей на даних контрольної вибірки. Наведений наочний графічний матеріал підтверджує достатньо високі апроксимаційні можливості створених сурогатних моделей та доводить їх адекватність.

Отже, використовуючи DNN для багатовимірної апроксимації гіперповерхні відгуку, побудовано адекватну сурогатну MLP-модель процесу вихрострумового контролю циліндричного слабомагнітного ОК прохідним перетворювачем у вигляді сукупності двох дійснозначних глибоких нейронних мереж.

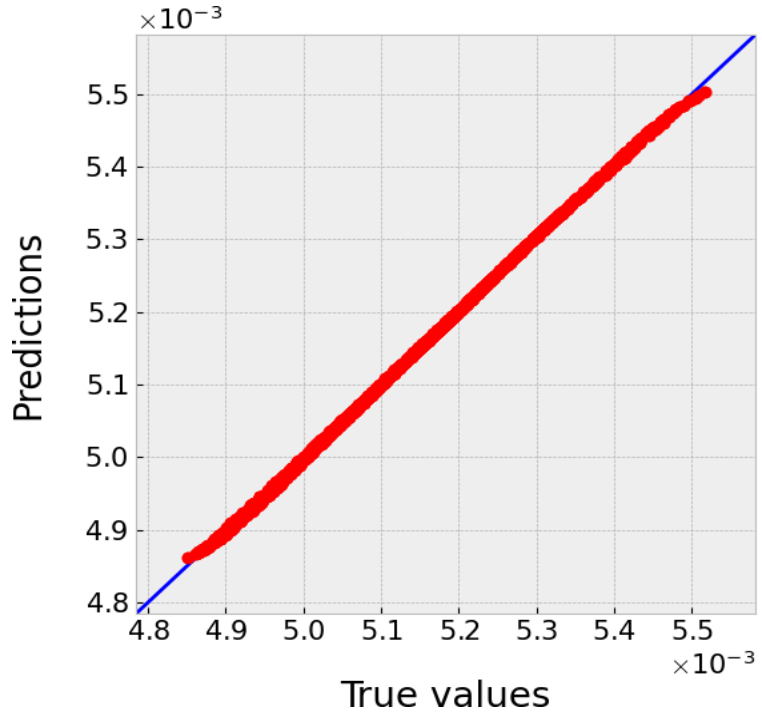


Рис. 3.9. Діаграма розсіювання індукованої Re(U) у вимірювальному витку ВСП.

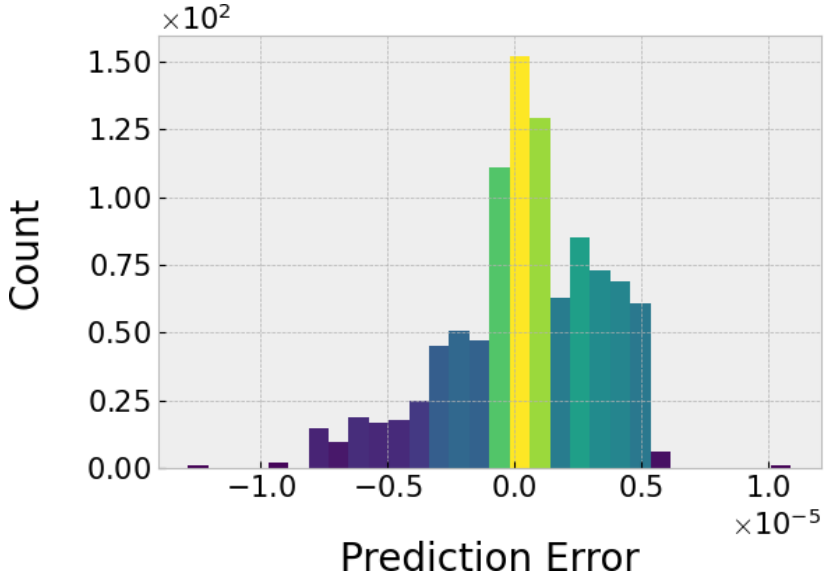


Рис. 3.10. Гістограма розподілу абсолютних похибок індукованої Re(U) у вимірювальному витку ВСП.

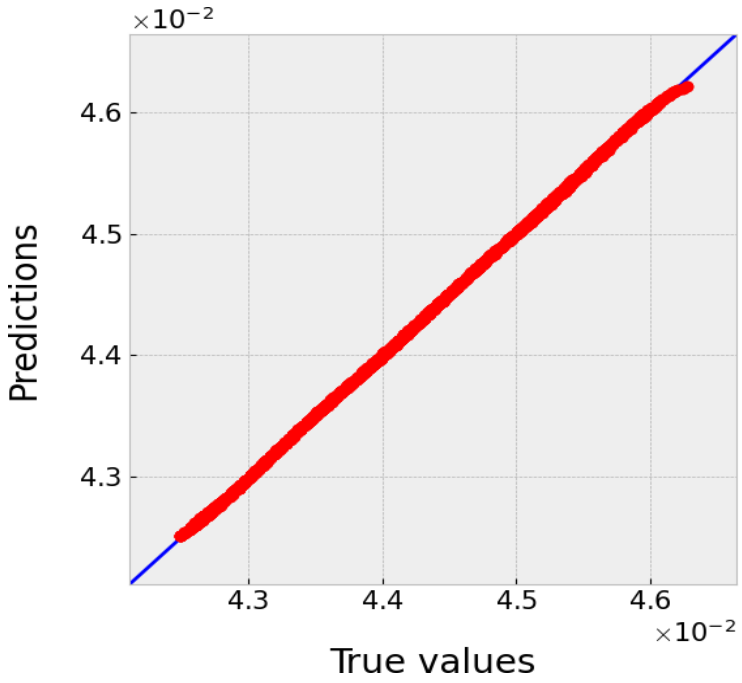


Рис. 3.11. Діаграма розсіювання індукованої Im(U) у вимірювальному витку ВСП.

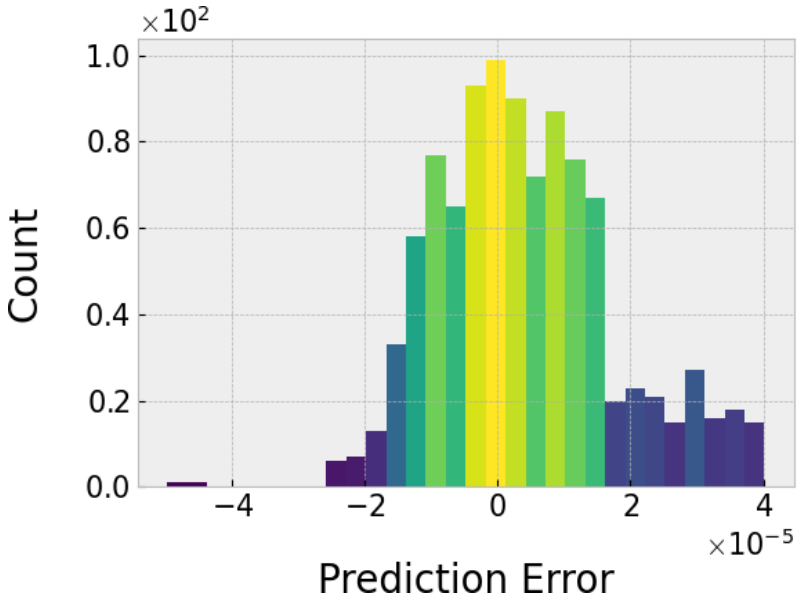


Рис. 3.12. Гістограма розподілу абсолютних похибок індукованої Im(U) у вимірювальному витку ВСП.

Сурогатна модель виконує функції носія апріорних знань щодо ОК, забезпечує високу обчислювальну ефективність та незначні часові затрати на розрахунки, що дозволяє розв'язання задачі параметричної ідентифікації радіальних приповерхневих профілів електрофізичних характеристик ОК у реальному часі в процесі виконання вимірювальних операцій.

3.4. Розв’язання обернених вимірювальних задач вихрострумової структуроскопії методом Lookup Tables

Таблиці пошуку LUT є одним з найкращих засобів для відновлення значень параметрів об’єкта вимірювання на основі вимірювань вихідного сигналу ВСП у контексті отримання результату в реальному масштабі часу. Застосування LUT є поширеним методом при розв’язанні обернених вимірювальних задач, хоч цей підхід інколи може погіршити точність результатів, але при цьому значно спрощується процес обробки результатів вимірювань.

Перевагою методу є ефективність використання ресурсів, швидкість обчислення та надійність. Застосування LUT є ефективним з точки зору використання ресурсів, оскільки значення вимірюваних результатів вже обчислені і збережені, що дозволяє уникнути зайвого використання ресурсів під час повторних обчислень у порівнянні з складними альтернативними алгоритмами оптимізації. Оскільки значення параметрів вимірювання вже попередньо обчислені і збережені в таблиці, для отримання результату не потрібно проводити повторні обчислення, що забезпечує високу швидкість роботи. Використання LUT може зменшити вплив помилок на відтворені дані електрофізичних параметрів ОК, адже цей метод є детермінованим і не залежить від початкових умов, що обумовлює його надійність [8].

LUT можна поєднувати з іншими методами для розв’язання обернених задач для отримання кращих результатів. Варто виділити інтерполяційні та оптимізаційні методи серед засобів для поєднання з LUT. Інтерполяційні методи використовуються для отримання значень шуканих електрофізичних параметрів в точках, які не були попередньо збережені в таблиці LUT. В такому випадку, значення електрофізичних параметрів об’єкта отримуються шляхом інтерполяції між ближчими до заданого вимірювання значень вихідного сигналу ВСП, які були збережені в таблиці LUT. Також метод LUT ефективно поєднується з оптимізаційними методами для пошуку підсумкового розв’язку задачі.

Для розв’язання оберненої задачі вимірювання профілів електрофізичних параметрів ОК в рамках цих досліджень пропонується застосування дворівневого пошуку по апріорі підготовленій таблиці. На першому етапі пошуку застосовується

заздалегідь нарахована за точною моделлю процесу вимірювань таблиця “грубих значень” сигналу ВСП з досить великою дискретизацією і відповідних цьому сигналу параметрів ОК, тобто профілів провідності σ та магнітної проникності μ_r або інших параметрів у випадку розширеної мультифакторної таблиці.

На другому етапі за знайденими “грубими значеннями” параметрів ОК динамічно формується новий набір проміжних значень в діапазоні найближчих табличних значень. Новий набір створюється за допомогою нового комп’ютерного плану експериментів на R-послідовностях і на основі цієї вибірки нараховується і зберігається таблиця другого рівня. Обрахунок значень вихідного сигналу ВСП в таблиці здійснюється з використанням сурогатної моделі, що дозволяє виконати це в реальному масштабі часу. До того ж це дозволяє застосовувати низку сурогатних моделей з дещо різним переліком факторів накопичення апріорної інформації, що є важливою особливістю застосування динамічної LUT другого рівня.

Нова “детальна” таблиця другого рівня тимчасово зберігається в пам’яті, що дозволяє перевикористання результату для таких же або подібних за значеннями вхідних параметрів, тобто застосовується метод мемоізації (memoization). Метод мемоізації, що зазвичай використовується для викликів важких для обчислення функцій в комп’ютерних обчисленнях, а також хешування таблиць значно збільшує швидкість алгоритму (рис.3.13).

Варто зазначити, що всі операції з пошуку значень в таблицях чи формування нової таблиці за допомогою сурогатної моделі слід виконувати з даними в нормованому вигляді.

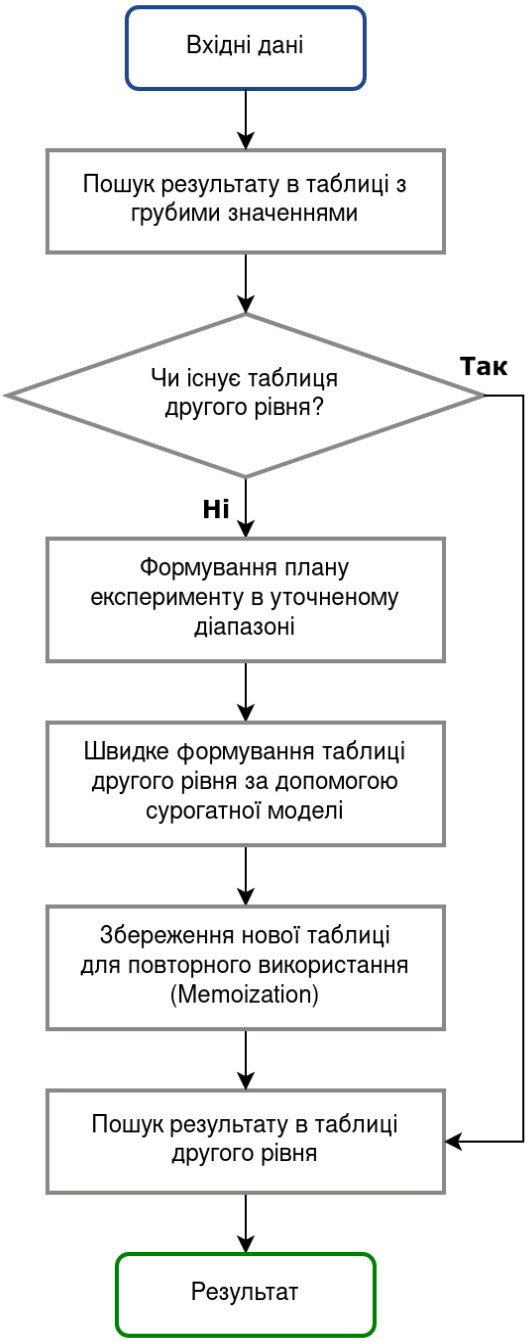


Рис. 3.13. Алгоритм використання дворівневого динамічного LUT.

Наочно цей алгоритм демонструє наступний приклад.

Таблиця 3.18 - Таблиця першого рівня (параметри матеріалу ОК)

№	σ_1, См/м	…	σ_{51}, См/м	μr_1	…	μr_{51}
1	6,9900E+06	…	3,2380E+06	1,0000E+00	…	8,7095E+00
2	6,9900E+06	…	2,9810E+06	1,0000E+00	…	1,0419E+01

3	6,9900E+06	…	3,7725E+06	1,0000E+00	…	9,1286E+00
4	6,9900E+06	…	3,5155E+06	1,0000E+00	…	1,0838E+01
5	6,9900E+06	…	3,2584E+06	1,0000E+00	…	9,5476E+00
…	…	…	…	…	…	…
996	6,9900E+06	…	3,3463E+06	1,0000E+00	…	8,6827E+00
997	6,9900E+06	…	3,0892E+06	1,0000E+00	…	1,0392E+01
998	6,9900E+06	…	3,8807E+06	1,0000E+00	…	9,1017E+00
999	6,9900E+06	…	3,6237E+06	1,0000E+00	…	1,0811E+01
1000	6,9900E+06	…	3,3667E+06	1,0000E+00	…	9,5208E+00

Таблиця 3.19 - Таблиця першого рівня (параметри сигналу ВСП)

№	Re(U)	Im(U)	Амплітуда, В	Фаза, рад
1	4,9468E-03	4,2914E-02	4,3198E-02	1,4560E+00
2	4,9815E-03	4,5044E-02	4,5319E-02	1,4607E+00
3	5,1623E-03	4,3246E-02	4,3553E-02	1,4520E+00
4	5,2418E-03	4,5339E-02	4,5641E-02	1,4557E+00
5	5,0194E-03	4,3918E-02	4,4204E-02	1,4570E+00
…	…	…	…	…
996	4,9808E-03	4,2846E-02	4,3135E-02	1,4551E+00
997	5,0239E-03	4,4975E-02	4,5255E-02	1,4596E+00
998	5,1959E-03	4,3177E-02	4,3489E-02	1,4510E+00
999	5,2834E-03	4,5268E-02	4,5575E-02	1,4546E+00
1000	5,0568E-03	4,3850E-02	4,4140E-02	1,4560E+00

За сигналом ВСП вибираємо профілі з таблиці грубих значень. Далі використовується динамічна побудова таблиці другого рівня. За новим планом експериментів масштабуються проміжні значення змінних, за якими потім формується весь набір значень для вхідних даних сурогатної моделі

В новій таблиці другого рівня аналогічно до пошуку в таблиці першого рівня проводиться пошук найближчих значень вихідного сигналу ВСП для тестового зразка, використовуючи для цього значення його амплітуди та фази, які трансформовано в алгебраїчну форму представлення комплексного числа. Результати вимірювання

виділено напівжирним в таблицях другого рівня.

Наведений приклад показує, що метод LUT є ефективним інструментом для розв'язання обернених вимірювальних задач вихрострумової структуроскопії. Використання LUT значно зменшує витрати часу та обчислювальних ресурсів на вирішення завдання, що дозволяє розв'язати задачу одночасного визначення профілів параметрів електрофізичних параметрів ОК в реальному масштабі часу.

Таблиця 3.20 - Тестовий зразок даних для перевірки працездатності алгоритму (параметри матеріалу ОК)

σ_1, См/м	…	σ_{51}, См/м	μr_1	…	μr_{51}
6,9900E+06	…	3,6450E+06	1,0000E+00	…	9,3134E+00

Таблиця 3.21 - Тестовий зразок даних для перевірки працездатності алгоритму (параметри сигналу ВСП)

Re(U)	Im(U)	Амплітуда, В	Фаза, рад
5,1368E-03	4,3509E-02	4,3811E-02	1,4533E+00

Таблиця 3.22 - Результати пошуку найближчих значень в таблиці першого рівня (параметри матеріалу ОК)

№	σ_1, См/м	…	σ_{51}, См/м	μr_1	…	μr_{51}
590	**6,9900E+06**	…	**3,6341E+06**	**1,0000E+00**	…	**9,3268E+00**
239	6,9900E+06	…	3,6739E+06	1,0000E+00	…	9,2850E+00
855	6,9900E+06	…	3,6788E+06	1,0000E+00	…	9,3498E+00
125	6,9900E+06	…	3,6151E+06	1,0000E+00	…	9,3996E+00

Таблиця 3.23 - Результати пошуку найближчих значень в таблиці першого рівня (параметри сигналу ВСП)

№	Re(U)	Im(U)	Амплітуда, В	Фаза, рад
590	**5,1343E-03**	**4,3528E-02**	**4,3830E-02**	**1,4534E+00**
239	5,1441E-03	4,3465E-02	4,3768E-02	1,4530E+00
855	5,1521E-03	4,3541E-02	4,3844E-02	1,4530E+00
125	5,1346E-03	4,3621E-02	4,3923E-02	1,4536E+00

Таблиця 3.24 - Таблиця другого рівня (параметри матеріалу ОК)

№	σ_1, См/м	…	σ_{51}, См/м	μr_1	…	μr_{51}
1	6,9900E+06	…	3,6314E+06	1,0000E+00	…	9,2930E+00
2	6,9900E+06	…	3,6158E+06	1,0000E+00	…	9,3583E+00
3	6,9900E+06	…	3,6638E+06	1,0000E+00	…	9,3090E+00
4	6,9900E+06	…	3,6482E+06	1,0000E+00	…	9,3743E+00
5	6,9900E+06	…	3,6326E+06	1,0000E+00	…	9,3250E+00
…	…	…	…	…	…	…
12	6,9900E+06	…	3,6507E+06	1,0000E+00	…	9,3237E+00
13	6,9900E+06	…	3,6351E+06	1,0000E+00	…	9,3891E+00
14	6,9900E+06	…	3,6195E+06	1,0000E+00	…	9,3398E+00
15	6,9900E+06	…	3,6675E+06	1,0000E+00	…	9,2904E+00
16	6,9900E+06	…	3,6519E+06	1,0000E+00	…	9,3558E+00

Таблиця 3.25 - Таблиця другого рівня (параметри сигналу ВСП)

№	Re(U)	Im(U)	Амплітуда, В	Фаза, рад
1	5,1281E-03	4,3493E-02	4,5544E-02	1,4500E+00
2	5,1288E-03	4,3574E-02	4,5463E-02	1,4500E+00
3	5,1413E-03	4,3501E-02	4,5523E-02	1,4503E+00
4	5,1422E-03	4,3582E-02	4,5442E-02	1,4503E+00
5	5,1317E-03	4,3530E-02	4,5503E-02	1,4501E+00
…	…	…	…	…
12	5,1380E-03	4,3522E-02	4,5504E-02	1,4503E+00
13	5,1389E-03	4,3603E-02	4,5423E-02	1,4502E+00
14	5,1284E-03	4,3551E-02	4,5486E-02	1,4500E+00
15	5,1408E-03	4,3478E-02	4,5546E-02	1,4503E+00
16	5,1417E-03	4,3559E-02	4,5465E-02	1,4503E+00

Таблиця 3.26 - Результати пошуку найближчих значень в таблиці другого рівня (параметри матеріалу ОК)

№	σ_1, См/м	…	σ_{51}, См/м	μr_1	…	μr_{51}
11	**6,9900E+06**	…	**3,6507E+06**	**1,0000E+00**	…	**9,3237E+00**
7	6,9900E+06	…	3,6494E+06	1,0000E+00	…	9,2917E+00
2	6,9900E+06	…	3,6638E+06	1,0000E+00	…	9,3090E+00
14	6,9900E+06	…	3,6675E+06	1,0000E+00	…	9,2904E+00

Таблиця 3.27 - Результати пошуку найближчих значень в таблиці другого рівня (параметри сигналу ВСП)

№	Re(U)	Im(U)	Амплітуда, В	Фаза, рад
11	**5,1380E-03**	**4,3522E-02**	**4,5504E-02**	**1,4503E+00**
7	5,1344E-03	4,3486E-02	4,5544E-02	1,4502E+00
2	5,1413E-03	4,3501E-02	4,5523E-02	1,4503E+00
14	5,1408E-03	4,3478E-02	4,5546E-02	1,4503E+00

Список використаних джерел до глави 3

1. Nestor Jr, C. W., Dodd, C. V., & Deeds, W. E. (1979). Analysis and computer programs for eddy current coils concentric with multiple cylindrical conductors. *NASA STI/Recon Technical Report N*, *80*, 15359. https://www.osti.gov/biblio/6145607
2. Dodd, C. V., & Deeds, W. E. (1968). Analytical solutions to eddy-current probe-coil problems. *Journal of applied physics*, *39*(6), 2829-2838. https://doi.org/10.1063/1.1656680
3. Dodd, C. V., Luquire, J. W., Deeds, W. E., & Spoeri, W. G. (1969). *Some eddy-current problems and their integral solutions* (No. ORNL-4384). Oak Ridge National Lab.(ORNL), Oak Ridge, TN (United States). https://www.osti.gov/servlets/purl/4783007
4. Uzal, E. (1992). *Theory of eddy current inspection of layered metals*. Iowa State University. https://www.proquest.com/openview/06972356f1b282d7831062f658520e1a/1?pq-origsite=gscholar&cbl=18750&diss=y
5. Гальченко, В. Я., Тичков, В. В., Сторчак, А. В., & Трембовецька, Р. В. (2020). Відновлення приповерхневих радіальних профілів електрофізичних характеристик циліндричних об’єктів при вихрострумових вимірюваннях із наявністю апріорних даних. Формування вибірки для побудови сурогатної моделі. *Український метрологічний журнал/Ukrainian Metrological Journal*, (1), 35-50. https://doi.org/10.24027/2306-7039.1.2020.204226
6. Kingma, D. P. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*. https://doi.org/10.48550/arXiv.1412.6980
7. Гальченко, В. Я., Сторчак, А., Трембовецька, Р. В., & Тичков, В. В. (2020). Створення сурогатної моделі для відновлення приповерхневих профілів електрофізичних характеристик

циліндричних об’єктів. *Український метрологічний журнал*, (3), 27-35. https://doi.org/10.24027/2306-7039.3.2020.216824

8. Гальченко, В. Я., Сторчак, А., Тичков, В. В., & Трембовецька, Р. В. (2022). Вимірювання приповерхневих радіальних профілів електрофізичних характеристик циліндричних об’єктів вихрострумовим методом із застосуванням апріорних даних. *Український метрологічний журнал*, (1), 5-11. https://doi.org/10.24027/2306-7039.1.2022.258678

ГЛАВА 4
ПРОГРАМНО-АПАРАТНИЙ КОМПЛЕКС ДЛЯ ВИМІРЮВАННЯ ПРОФІЛІВ ЕЛЕКТРОФІЗИЧНИХ ВЛАСТИВОСТЕЙ ЦИЛІНДРИЧНИХ ОБ'ЄКТІВ КОНТРОЛЮ

4.1. Апаратна частина комплексу

Однією з головних переваг підходу до розв'язання задачі, яка досліджується, є те, що для реалізації процесу вимірювання може бути використаний класичний прохідний трансформаторний ВСП разом із вже існуючими серійними дефектоскопами з синусоїдальним струмом збудження та вихідною інформацією про параметри сигналу, які можна зчитати в цифровому вигляді. Отже, основна суть методу полягає в обробці сигналу ВСП, який має бути представленим у вигляді даних про його амплітуду і фазу. Також в контексті цього дослідження пропонується варіант реалізації спеціалізованого вихрострумового вимірювача для визначення електрофізичних параметрів ОК.

Функціональна схема вихрострумового вимірювача (рис.4.1) включає наступні основні компоненти:

- Генератор струму збудження, до якого входить генератор прямого цифрового синтезу DDS (Direct digital synthesizer) та підсилювач сигналу синтезатора. Він генерує синусоїдальний струм заданої частоти, який подається на обмотку збудження ВСП;
- Прохідний ВСП трансформаторного типу у вигляді двох котушок, а саме котушки збудження та вимірювальної котушки;
- Модуль вимірювання параметрів сигналу. Вихідний сигнал ВСП в цьому модулі подається на фазовий та амплітудний детектори для виділення потрібної інформації. Надалі отримані сигнали згладжуються фільтрами низьких частот другого порядку (ФНЧ) і подаються до АЦП для перетворення в цифрову форму;
- Модуль первинного оброблення сигналів та узгодження інтерфейсів. Цей модуль взаємодіє з генератором та модулем вимірювання через два інтерфейси SPI (один до синтезатора, інший - до АЦП), а за допомогою USB (Virtual COM Port, VCP) з модулем керування, обробки та відображення інформації і є зв'язувальною ланкою між ними. Також в ньому відбувається перетворення формату представлення параметрів сигналу та первинне його оброблення;

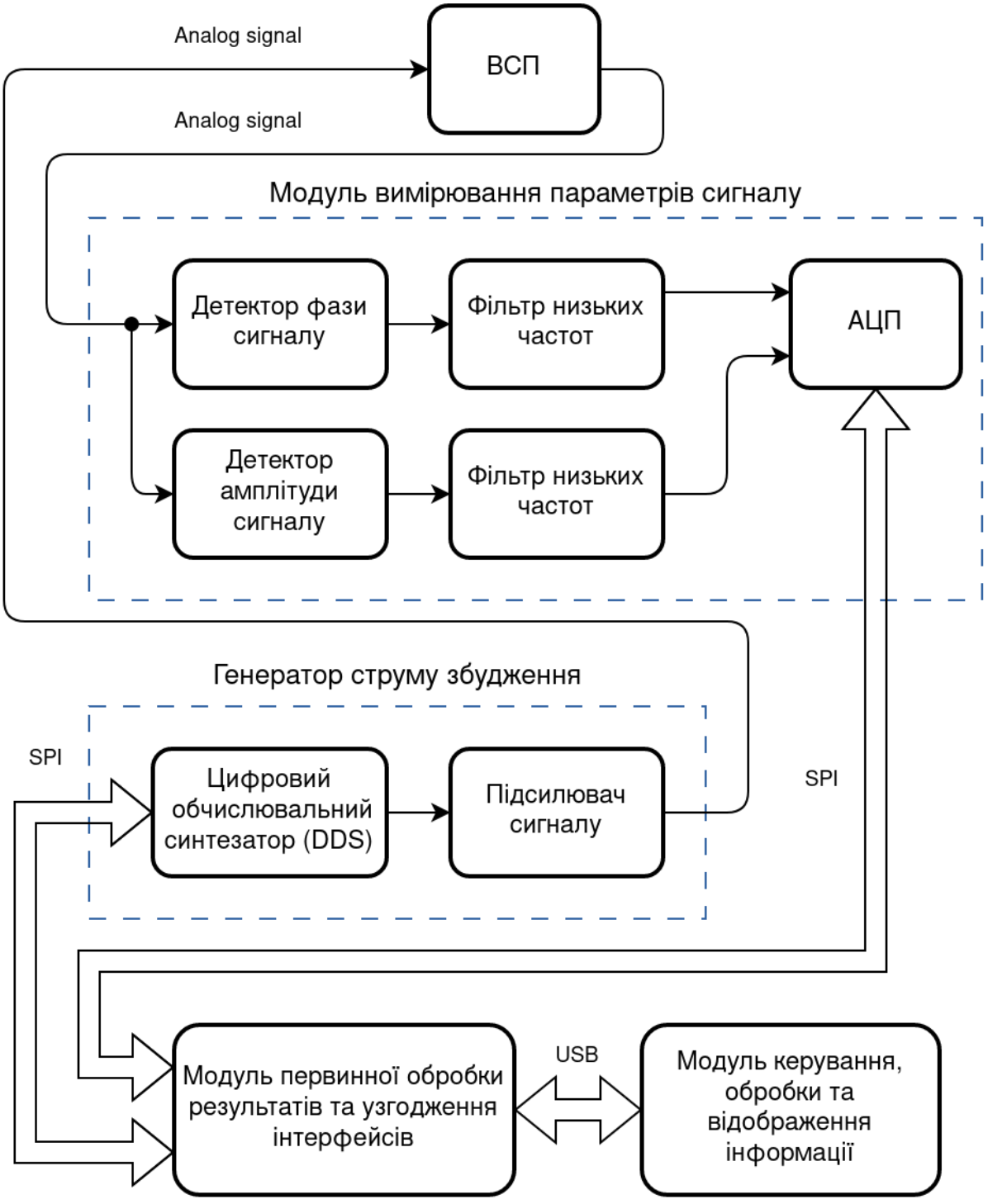


Рис. 4.1. Функціональна схема вихрострумового вимірювача профілів властивостей матеріалу ОК.

- Модуль керування, оброблення та відображення інформації, який призначений для конфігурації необхідних параметрів сигналу збудження ВСП, опрацювання параметрів сигналу і відображення результату, що дозволяє оператору приладу спостерігати отриману інформацію про властивості приповерхневого матеріалу досліджуваного ОК.

Генератор DDS побудований на мікросхемі AD9834BRUZ (рис.4.2) та кварцовому генераторі DSC1001CI1-050.0000. AD9834BRUZ - це малопотужний DDS-пристрій з тактовою частотою 50 МГц, здатний генерувати високоякісні синусоїдальні сигнали. Він також має вбудований компаратор, який дозволяє отримати прямокутний сигнал для генерації тактових імпульсів. Останній, в свою чергу, використовується в фазовому детекторі. Частотні регістри цього DDS мають розрядність в 28 біт. Запис в AD9834BRUZ здійснюється за допомогою 3-провідного послідовного інтерфейсу (SPI). Цей послідовний інтерфейс працює на тактовій частоті до 40 МГц і сумісний зі стандартами DSP і мікроконтролерів. Аналогова і цифрова його частини є незалежними і працюють від різних джерел живлення, що зменшує шум від цифрової частини схеми.

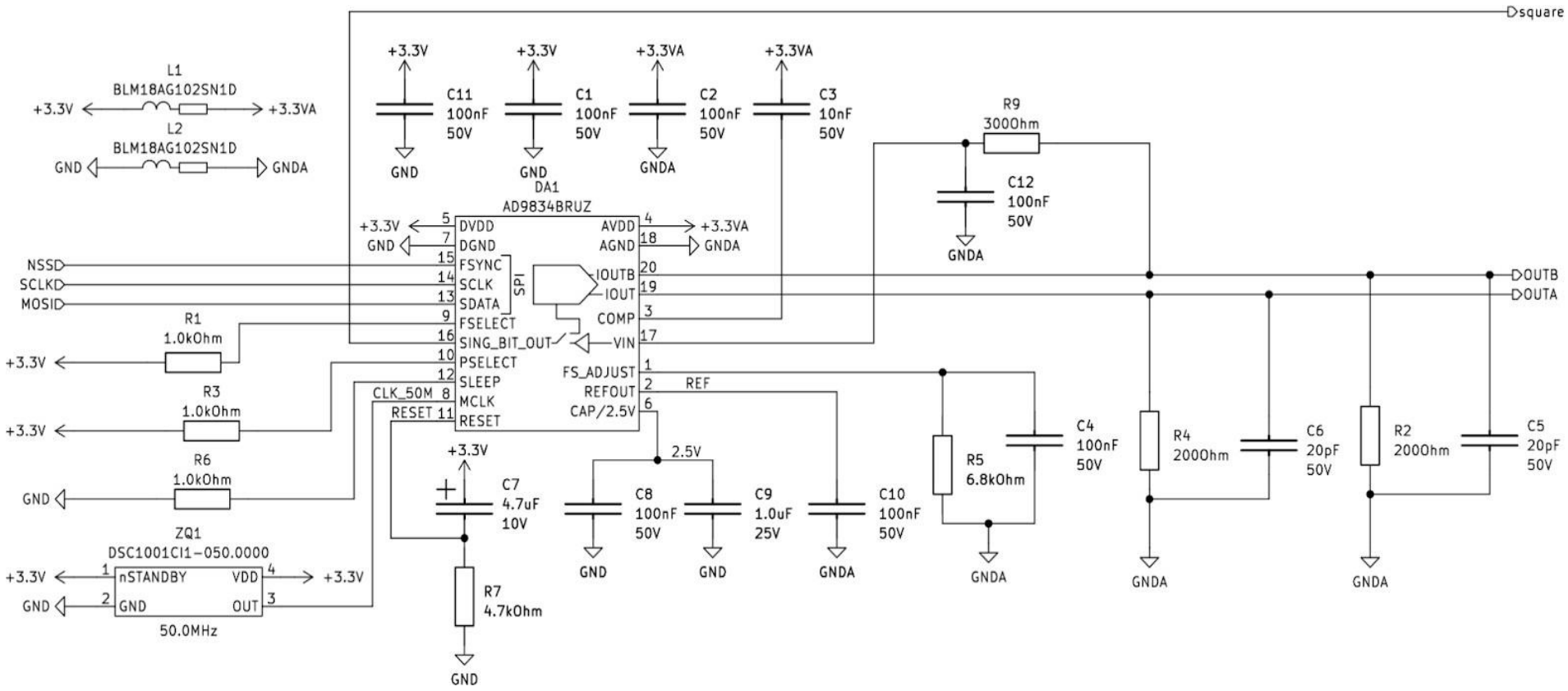


Рис. 4.2. Електрична принципова схема генератора прямого цифрового синтезу.

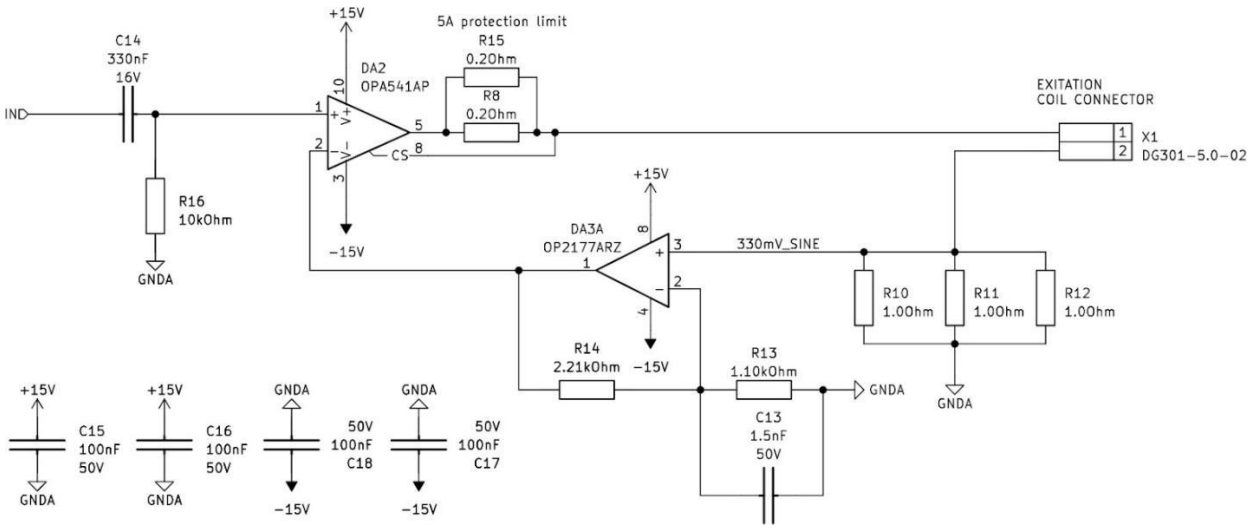


Рис. 4.3. Електрична принципова схема підсилювача сигналу збудження.

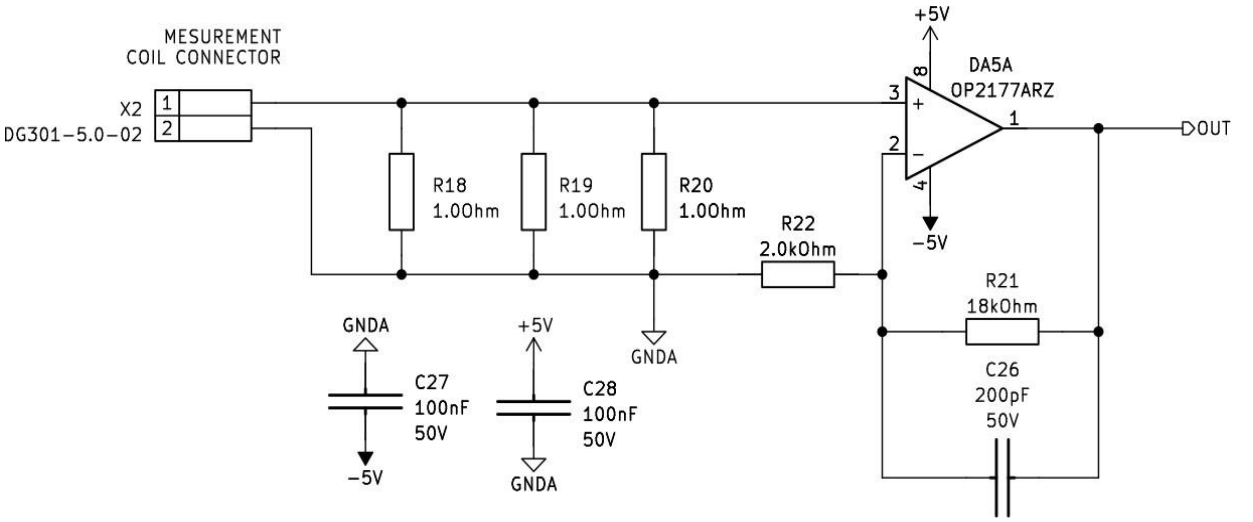


Рис. 4.4. Електрична принципова схема підсилювача сигналу ВСП.

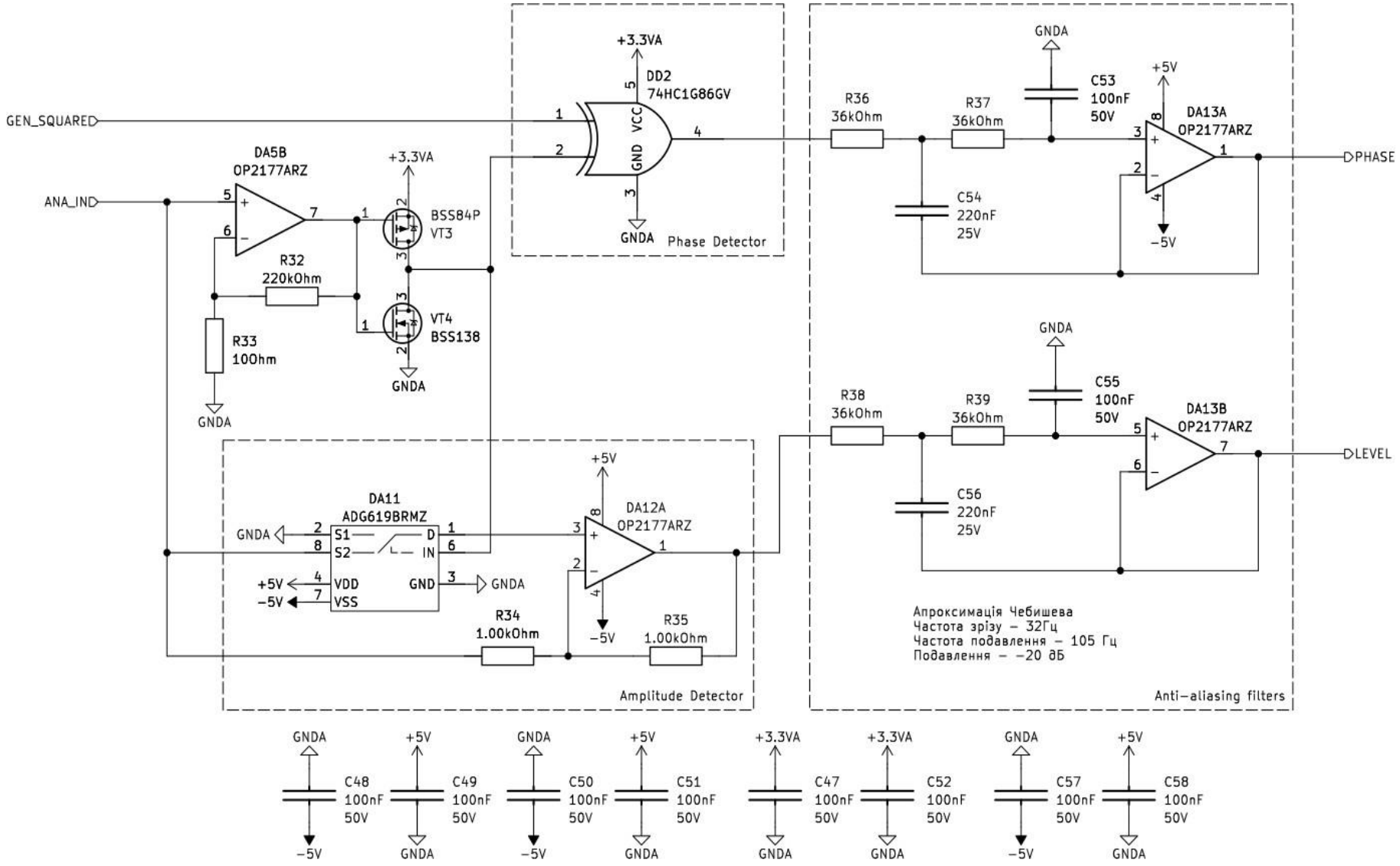


Рис. 4.5. Електричні принципові схеми фазового та амплітудного детекторів сигналу ВСП з фільтрами низьких частот другого порядку.

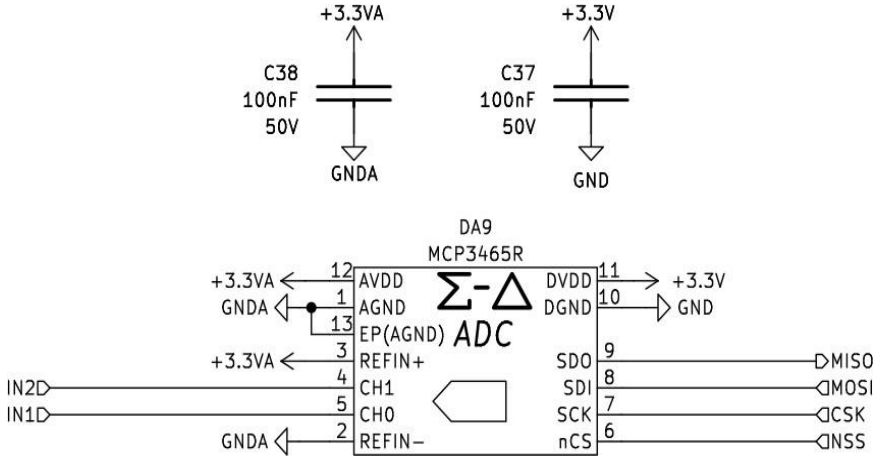


Рис. 4.6. Електрична принципова схема АЦП.

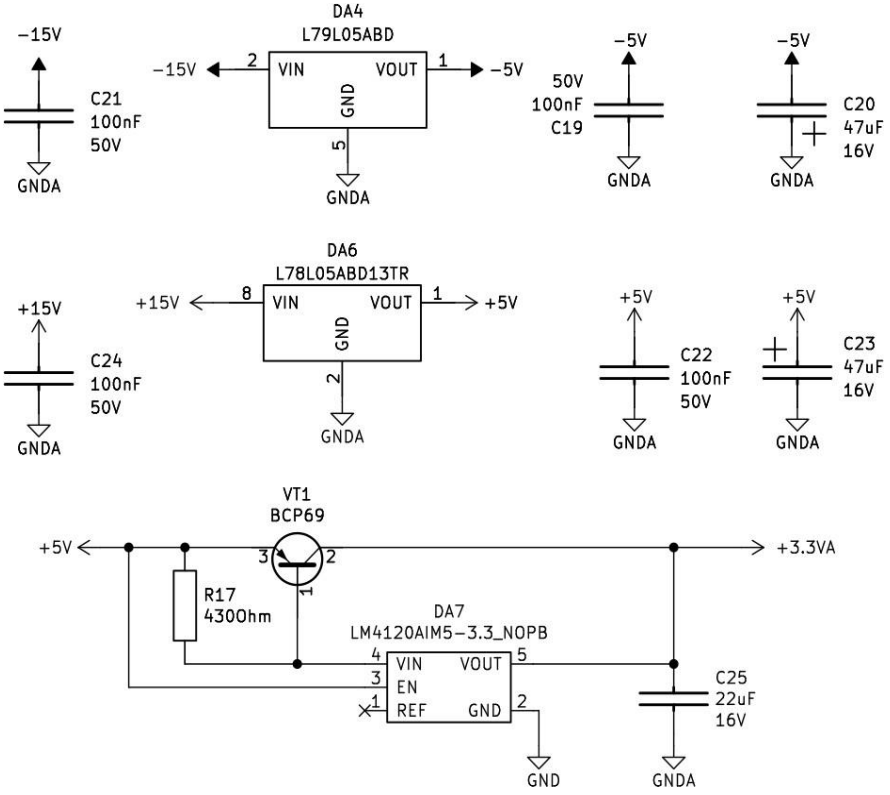


Рис. 4.7. Електричні принципові схеми блоків живлення для цифрових та аналогових частин пристрою.

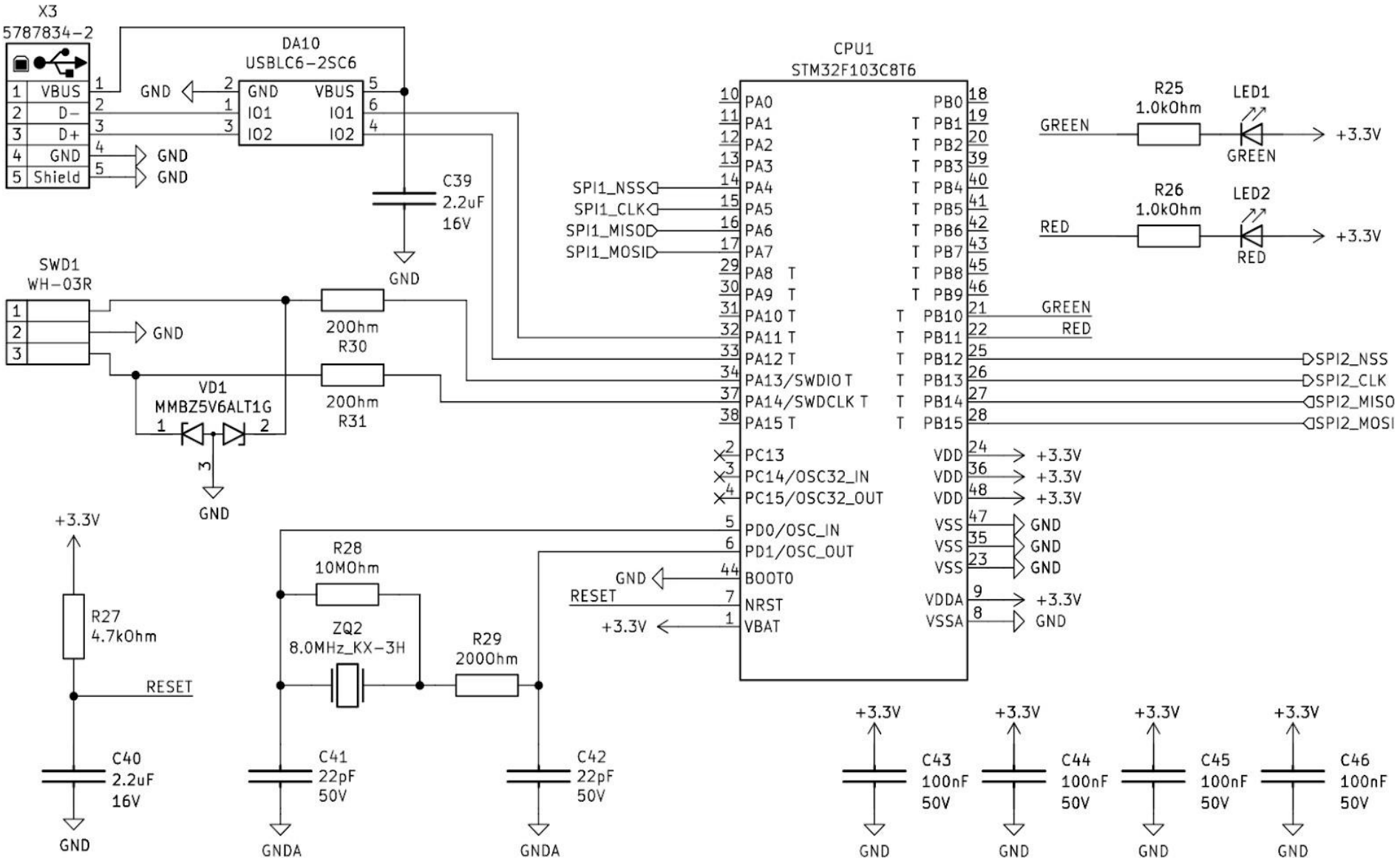


Рис. 4.8. Електрична принципова схема модуля первинного оброблення сигналу та узгодження інтерфейсів.

Підсилювачем сигналу збудження (рис.4.3) є операційний підсилювач (ОП) високої потужності OPA541. Цей ОП живиться від ±15 В (допускається ±40 В max) [1]. Внутрішня схема обмеження струму може бути налагоджена за допомогою одного зовнішнього резистора, що захищає підсилювач від перенавантаження. Максимальний струм такого обмеження може становити до 5 А. Операційний підсилювач DA3A доданий в петлю негативного зворотного зв'язку для узгодження падіння напруги на токовому шунті R10..R12 з вхідною напругою. Отже, каскад працює в режимі перетворення напруга-струм збудження ВСП.

Мікросхема OP2177 складається з двох прецизійних підсилювачів, що мають надзвичайно низьку вхідну напругу зміщення і низький температурний дрейф, характеризуються малим рівнем шуму та низьким енергоспоживанням. Виходи цього ОП стабільні з ємнісними навантаженнями понад 1000 пФ без зовнішньої компенсації. Вона використовується як в схемі драйвера, так і в схемі вимірювання. ФНЧ для фази та амплітуди є активними фільтрами Чебишева 2-го порядку [2], [3] з частотою зрізу 32 Гц, частотою подавлення 105 Гц і коефіцієнтом подавлення 20 дБ.

Щоб без втрати перетворити змінний сигнал ВСП в позитивні півхвилі без падіння напруги, що властиво класичним детекторам на діоді, використовується схема синхронного амплітудного детектора. Це дозволяє забезпечити лінійну передаточну функцію детектора в більшому динамічному діапазоні (до 4 порядків). Схематично цей вузол виконано на операційному підсилювачі, схему включення якого змінює електронний перемикач ADG619 [4]. В одній позиції перемикач забезпечує неінвертуючу схему включення ОП, в іншій - інвертуючу. ADG619 - це монолітний однополюсний CMOS-перемикач з подвійним перекиданням (SPDT) та низьким опором 4 Ом, який узгоджується з точністю до 0,7 Ом між каналами. Ці перемикачі також забезпечують низьке розсіювання потужності та при цьому мають високу швидкість перемикання [5], [6].

Фазовий детектор побудований на простому логічному елементі XOR (виключне АБО). Вхідними сигналами є сформовані внутрішнім компаратором синхроімпульси в синтезаторі AD9834BRUZ та перетворений в прямокутні імпульси сигнал з вимірювальної котушки. Це перетворення забезпечує підсилювач

DA5B. А транзистори VT3, VT4 забезпечують узгодження з TTL-рівнями логічного елемента. Далі отриманий за рахунок логічного елемента XOR ШІМ сигнал передається на ФНЧ. Така схема забезпечує перетворення діапазону фазових здвигів 0…180 градусів в постійну напругу 0…3.3 В.

Варто зазначити, що аналогова частина приладу живиться від окремого джерела живлення, побудованого на мікросхемі LM4120, що є прецизійним малопотужним джерелом опорної напруги, яка керує транзистором BCP69 для підвищення вихідної потужності.

Отримані з ФНЧ амплітудний та фазовий сигнал ВСП фіксуються мікросхемою MCP3465R, що є малошумним 16-розрядним дельта-сигма аналого-цифровим перетворювачем з двома однонаправленими каналами і програмованою швидкістю передачі даних до 153,6 kSPS [7]. АЦП MCP3465R конфігурується в схему з коефіцієнтом передискретизації (OSR) від 32 до 98304 і коефіцієнтом підсилення від 0,33X до 64X. Він включає блок внутрішньої автоматичної вибірки (режим SCAN) та 24-бітний таймер, що дозволяє автоматично створювати послідовності циклів перетворення без необхідності зв’язку з мікроконтролером. MCP3465R також має SPI-сумісний послідовний інтерфейс, що працює на частоті до 20 МГц. Комунікація значно спрощується завдяки 8-бітним командам, включаючи різні режими безперервного читання/запису і 24/32-розрядні формати даних, до яких можна отримати доступ через прямий доступ до пам’яті (DMA) 8-розрядного, 16-розрядного або 32-розрядного мікроконтролера.

Модуль первинної обробки сигналів та узгодження інтерфейсів сформований на 32 бітному ARM мікроконтролері (МК). Даний МК має ядро Cortex™-M3 CPU та може працювати з тактовою частотою до 72 MHz [8]. Для пам’яті програм в МК є 64 Kbytes вбудованої flash пам’яті і 20 Kbytes оперативної пам’яті (SRAM). Також в ньому присутні апаратні модулі SPI та USB (VCP). Це дозволяє йому з легкістю взаємодіяти з генератором та АЦП по інтерфейсу SPI та обмінюватися командами та інформацією про виміри з модулем керування, оброблення та відображення інформації [9], [10].

Модуль керування, оброблення та відображення інформації може бути представлений як персональним так і одноплатним комп’ютером, або планшетом з підтримкою USB (VCP).

Представлений варіант архітектури схемотехніки приладу є

недорогим у собівартості, але завдяки високоточним генератору струму збудження і АЦП дозволяє проводити достовірні вимірювання амплітуди та фази сигналу ВСП.

4.2. Особливості технології виконання вимірювань в реальному масштабі часу

Швидкодія застосування LUT залежить від багатьох факторів, включаючи її розмір, швидкість доступу до пам’яті, оптимізації апаратного засобу та алгоритму використання LUT. Наприклад, чим більший розмір LUT, тим більше часу може зайняти пошук в таблиці. Більші LUT потребують більше циклів для доступу до пам’яті та обробки даних. В свою чергу, якщо доступ до пам’яті повільний, це теж уповільнює пошук. Покращити швидкодію застосування LUT можна, використовуючи апаратні засоби, що мають спеціалізовані апаратні ресурси, такі як FPGA або ASIC. Алгоритм пошуку в LUT також має суттєвий вплив на швидкодію. Якщо алгоритм пошуку оптимізований для мінімізації затримок та використання кешу, то це значно пришвидшує використання LUT. В загальному випадку швидкодію LUT оцінити складно, оскільки вона залежить від конкретного апаратного засобу та застосування. Тож необхідно проводити специфічні випробування та вимірювання для визначення швидкодії застосування LUT у конкретному контексті.

В таблиці 4.1 наведено затрати часу на виконання етапів алгоритму LUT на ПК з операційною системою Ubuntu 16.04 та процесором Intel Core i3- 4150. При цьому для порівняння швидкодії створення динамічної таблиці другого рівня відбувалося двома різними бібліотеками: Keras та Numpy. В обох з них використовувалися однакові коефіцієнти та ваги для побудови сурогатної моделі на основі DNN, при цьому обчислення проводилися на CPU.

Таблиця 4.1 - Витрати часу на виконання етапів LUT

Назва етапу	Обсяг таблиці	Час, мс
Пошук в таблиці 1-го рівня	1000	1,11
	2000	1,29
	3000	1,23

	4000	1,39
	5000	1,39
	10000	1,75
Створення таблиці 2-го рівня (Keras)	8	304,92
	16	291,43
	32	304,68
	64	316,87
	128	307,51
Створення таблиці 2-го рівня (Numpy)	8	35,92
	16	34,12
	32	35,15
	64	39,01
	128	43,53
Пошук в таблиці 2-го рівня	8	1,99
	16	2
	32	2
	64	2,1
	128	3,13

Як приклад, сумарна витрата часу на виконання алгоритму, показана в таблиці 4.2. При цьому, таблиця другого рівня була створена сурогатною моделлю з використанням бібліотеки Numpy.

Таблиця 4.2 - Витрати часу (мс) на застосування LUT з динамічним створенням таблиці другого рівня

Характеристики таблиць LUT		Обсяг таблиці 1-го рівня		
		1000	3000	5000
Обсяг таблиці 2-го рівня	8	39,02	39,14	39,3
	16	37,23	37,35	37,51
	32	38,26	38,38	38,54
	64	42,22	42,34	42,5
	128	47,77	47,89	48,05

На прикладі продемонструємо пошук за виміряним вихідним сигналом ВСП інформації щодо профілів характеристик матеріалу, який виконувався в LUT. В якості “невідомого профілю” був взятий

контрольний зразок вимірювання, що був заздалегідь обчислений за точною моделлю. Числові значення параметрів контрольного зразка знаходяться в межах значень таблиці першого рівня, але завідомо не співпадають з ними. Вихідний сигнал ВСП складається з двох числових значень, представлених дійсною Re(U) та уявною Im(U) складовими. Визначення найближчих профілів з таблиці пошуку виконується з використанням критерію евклідової відстані в просторі ознак (4.1), що застосована до n-вимірного простору [11].

$$\rho(x, x') = \sqrt{\sum_{i}^{n} \left(x_i - x_i^* \right)^2}, \qquad (4.1)$$

де x та x^* - координати векторів.

Числові значення, по яким проводиться пошук, мають бути попередньо нормалізованими. Точність визначення профілів МП та ЕП залежить від кількості зразків профілів в динамічній таблиці другого рівня та складає не більше половини кроку між найбільш віддаленими точками сусідніх профілів.

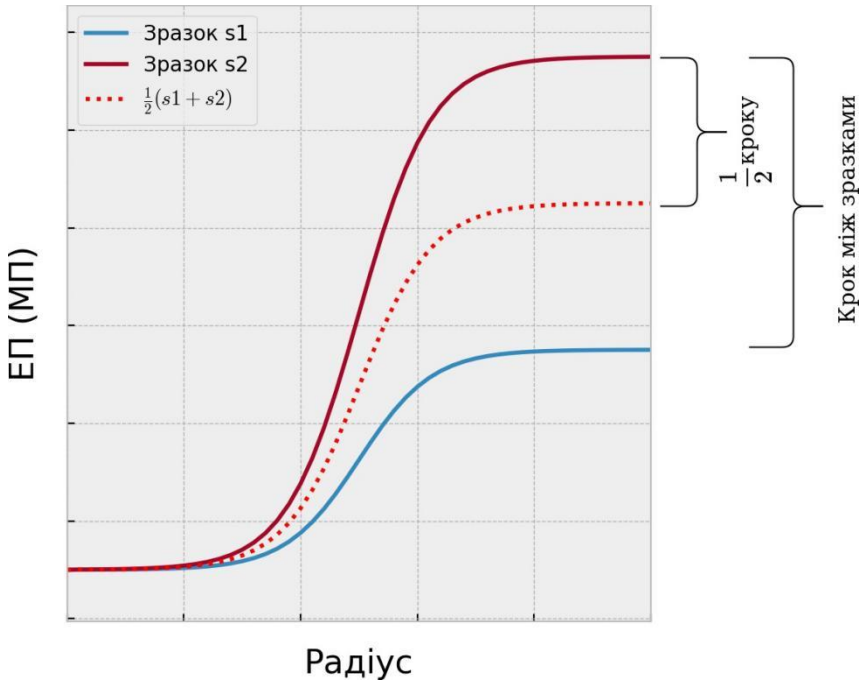


Рис. 4.9. Визначення кроку між зразками профілів в LUT.

Приклад визначення кроку сусідніх профілів ЕП чи МП в найбільш віддалених їх точках показано на рисунку 4.9. Математично крок для ЕП (См/м) розраховується відповідно до:

$$\delta_\sigma = \frac{\sigma_{s2} - \sigma_{s1}}{2}.$$

Для МП крок визначається за аналогічною формулою.

Приклад радіальних розподілів ЕП та МП наведено на рисунках 4.10 - 4.21 в вигляді графіків. Графіки для їх наочності подано в різних масштабах. Обсяг таблиці першого і другого рівня складає 1000 і 16 зразків відповідно.

Отже, середні затрати часу на виконання повного алгоритму розв'язання оберненої вимірювальної задачі за допомогою дворівневого LUT складають близько 38 мс, що дозволяє стверджувати про виконання його в реальному масштабі часу.

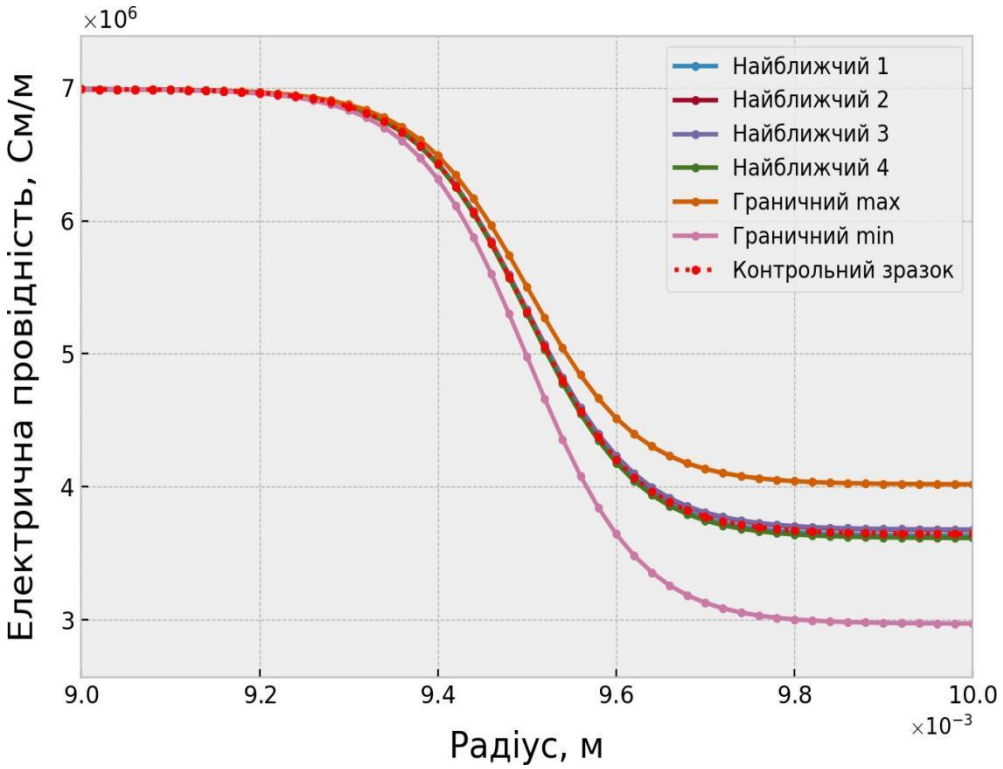


Рис. 4.10. Графічна інтерпретація пошуку найближчих до контрольного зразка профілів ЕП в таблиці 1-го рівня (Радіальні профілі ЕП матеріалу ОК на приповерхневій глибині 1 мм, зокрема контрольний, найближчі і граничні).

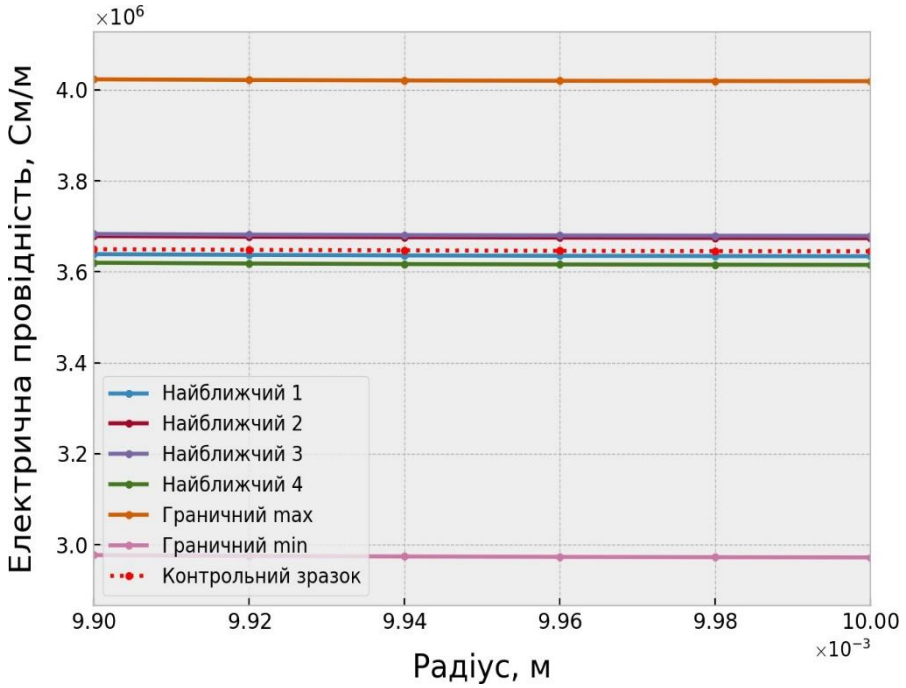


Рис. 4.11. Графічна інтерпретація пошуку найближчих до контрольного зразка профілів ЕП в таблиці 1-го рівня (Радіальні профілі ЕП матеріалу ОК на приповерхневій глибині 0,1 мм, зокрема контрольний, найближчі і граничні).

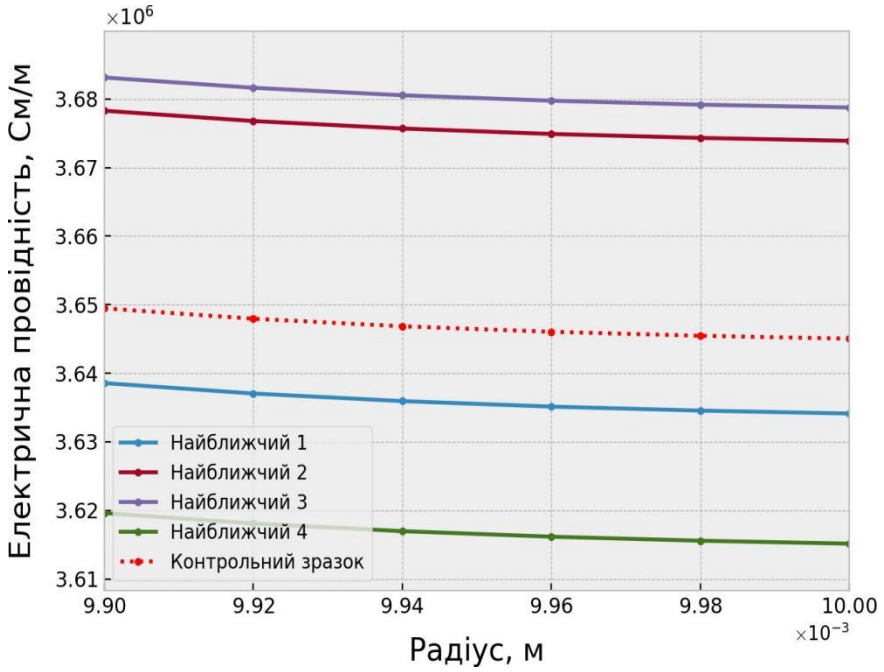


Рис. 4.12. Графічна інтерпретація пошуку найближчих до контрольного зразка профілів ЕП в таблиці 1-го рівня. (Радіальні профілі ЕП матеріалу ОК на приповерхневій глибині 0,1 мм, зокрема контрольний і найближчі).

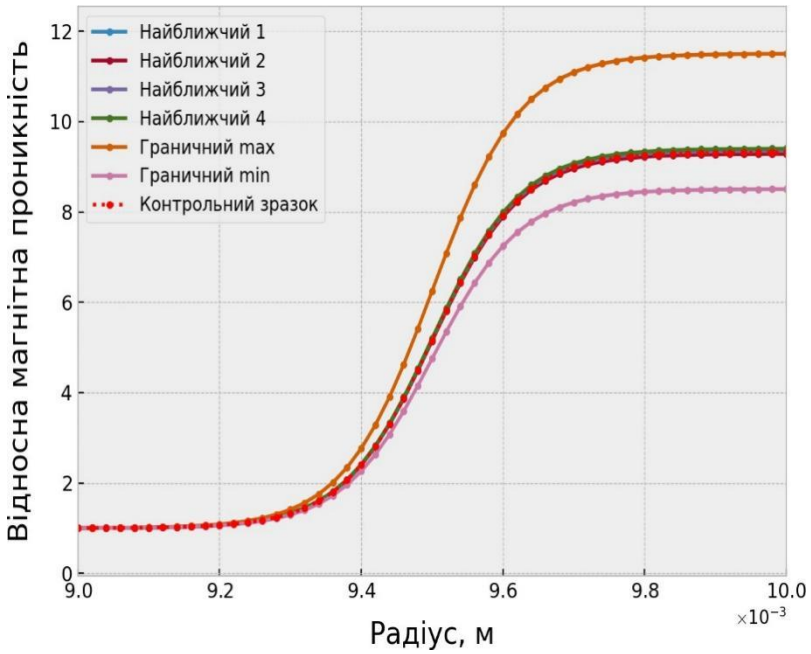


Рис. 4.13. Графічна інтерпретація пошуку найближчих до контрольного зразка профілів МП в таблиці 1-го рівня. (Радіальні профілі МП матеріалу ОК на приповерхневій глибині 1 мм, зокрема контрольний, найближчі і граничні).

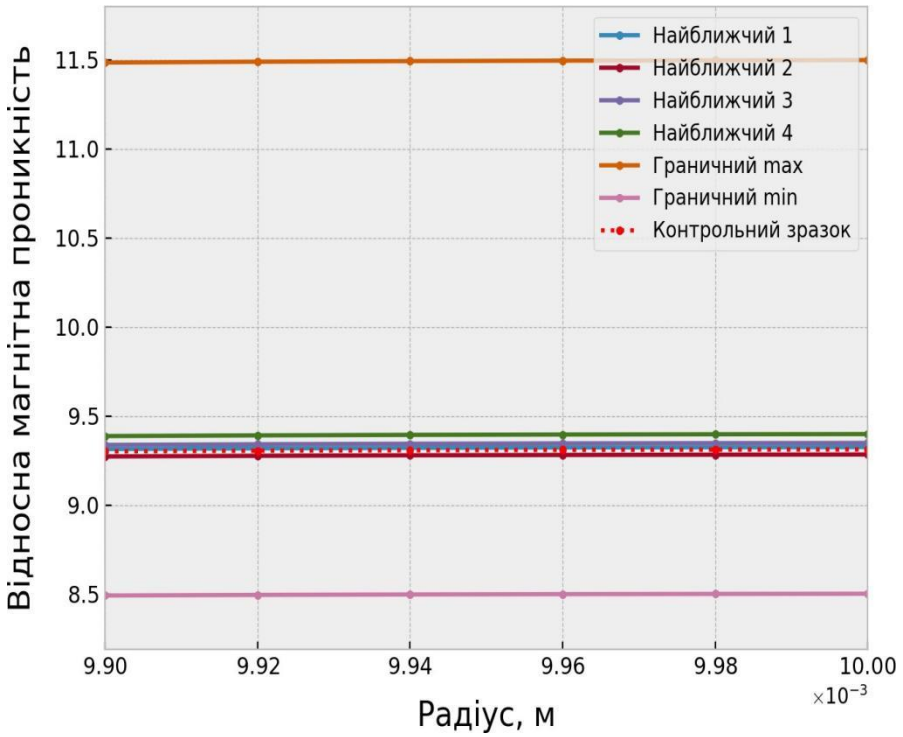


Рис. 4.14. Графічна інтерпретація пошуку найближчих до контрольного зразка профілів МП в таблиці 1-го рівня. (Радіальні профілі МП матеріалу ОК на приповерхневій глибині 0,1 мм, зокрема контрольний, граничні і найближчі).

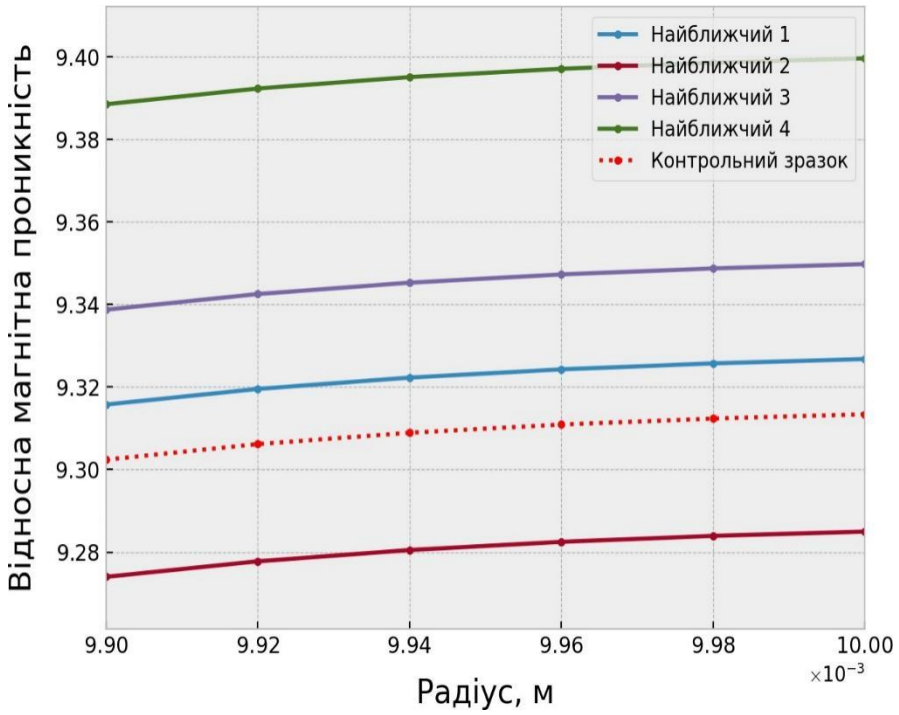


Рис. 4.15. Графічна інтерпретація пошуку найближчих до контрольного зразка профілів МП в таблиці 1-го рівня. (Радіальні профілі МП матеріалу ОК на приповерхневій глибині 0,1 мм, зокрема контрольний і найближчі).

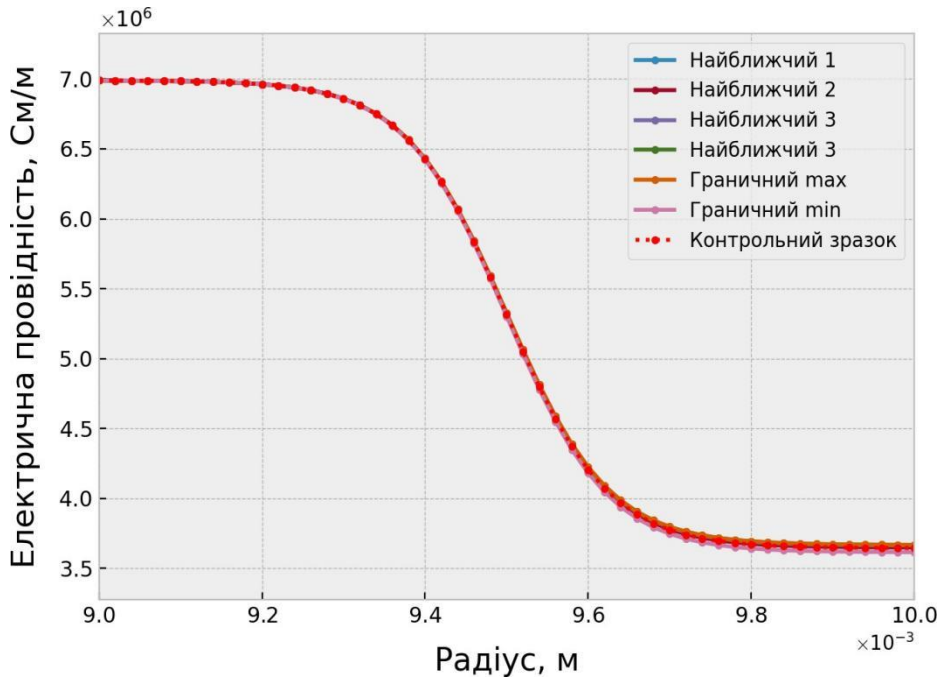


Рис. 4.16. Графічна інтерпретація пошуку найближчих до контрольного зразка профілів ЕП в таблиці 2-го рівня. (Радіальні профілі ЕП матеріалу ОК на приповерхневій глибині 1 мм, зокрема контрольний, граничні і найближчі).

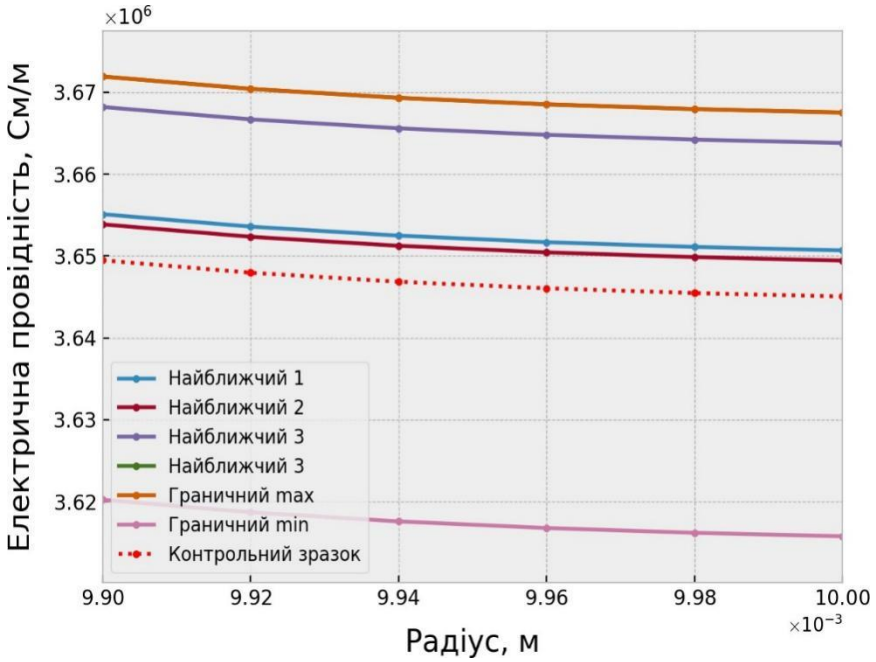


Рис. 4.17. Графічна інтерпретація пошуку найближчих до контрольного зразка профілів ЕП в таблиці 2-го рівня. (Радіальні профілі ЕП матеріалу ОК на приповерхневій глибині 0,1 мм, зокрема контрольний, граничні і найближчі).

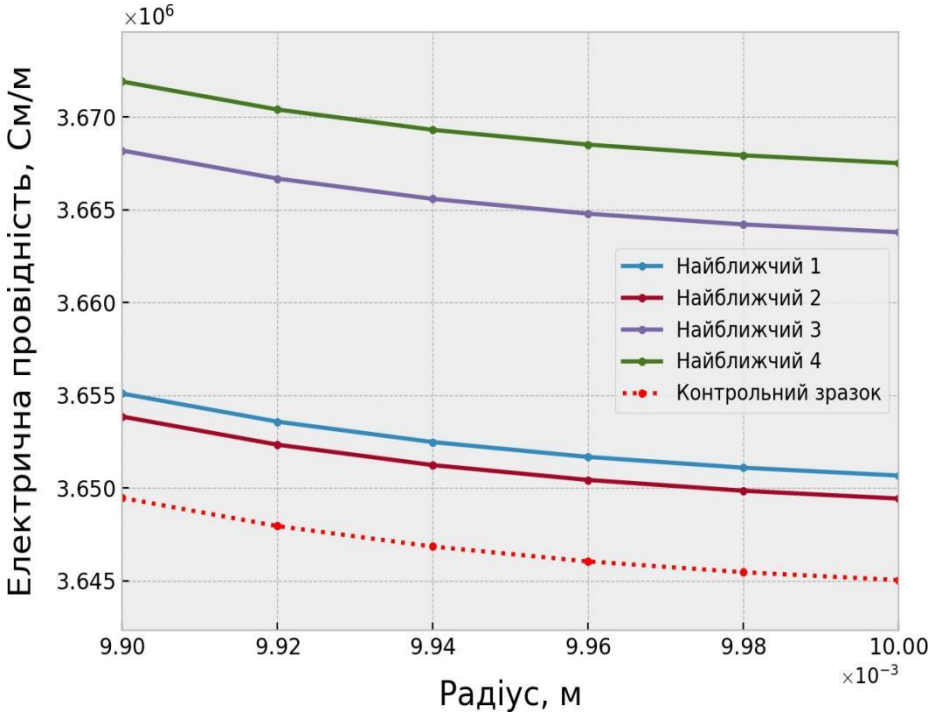


Рис. 4.18. Графічна інтерпретація пошуку найближчих до контрольного зразка профілів ЕП в таблиці 2-го рівня. (Радіальні профілі ЕП матеріалу ОК на приповерхневій глибині 0,1 мм, зокрема контрольний і найближчі).

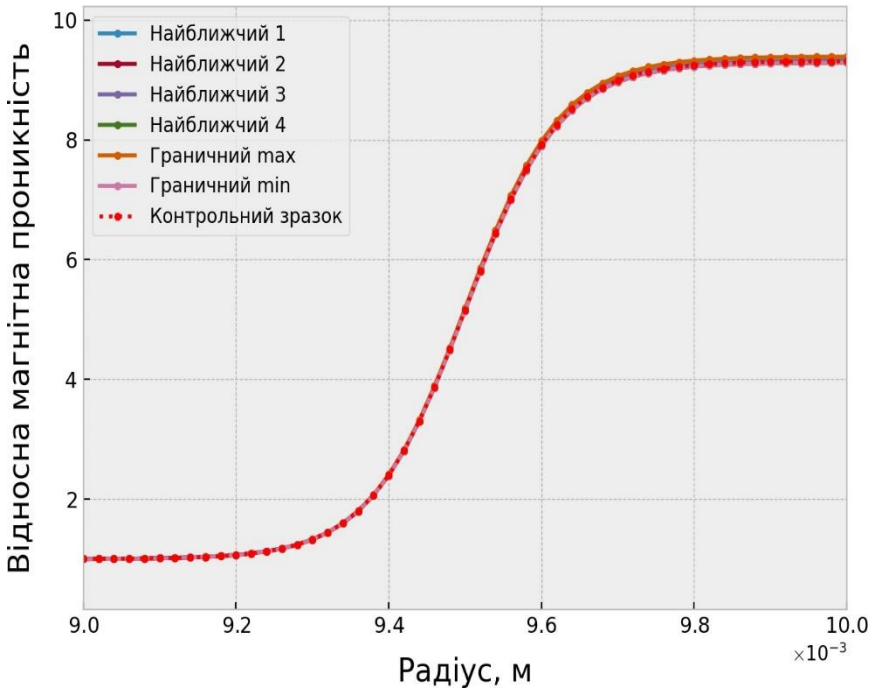


Рис. 4.19. Графічна інтерпретація пошуку найближчих до контрольного зразка профілів МП в таблиці 2-го рівня. (Радіальні профілі МП матеріалу ОК на приповерхневій глибині 1 мм, зокрема контрольний, граничні і найближчі).

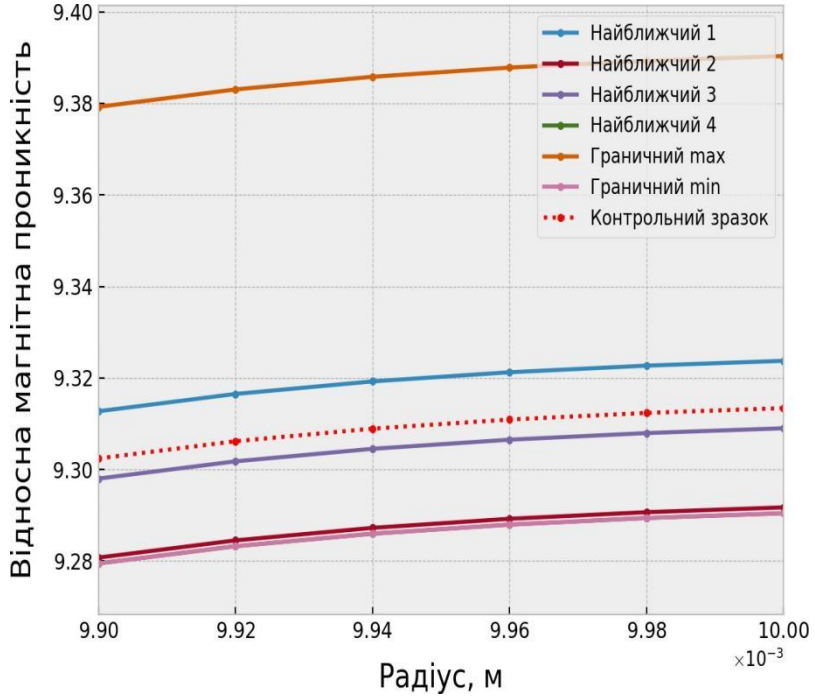


Рис. 4.20. Графічна інтерпретація пошуку найближчих до контрольного зразка профілів МП в таблиці 2-го рівня. (Радіальні профілі МП матеріалу ОК на приповерхневій глибині 0,1 мм, зокрема контрольний, граничні і найближчі).

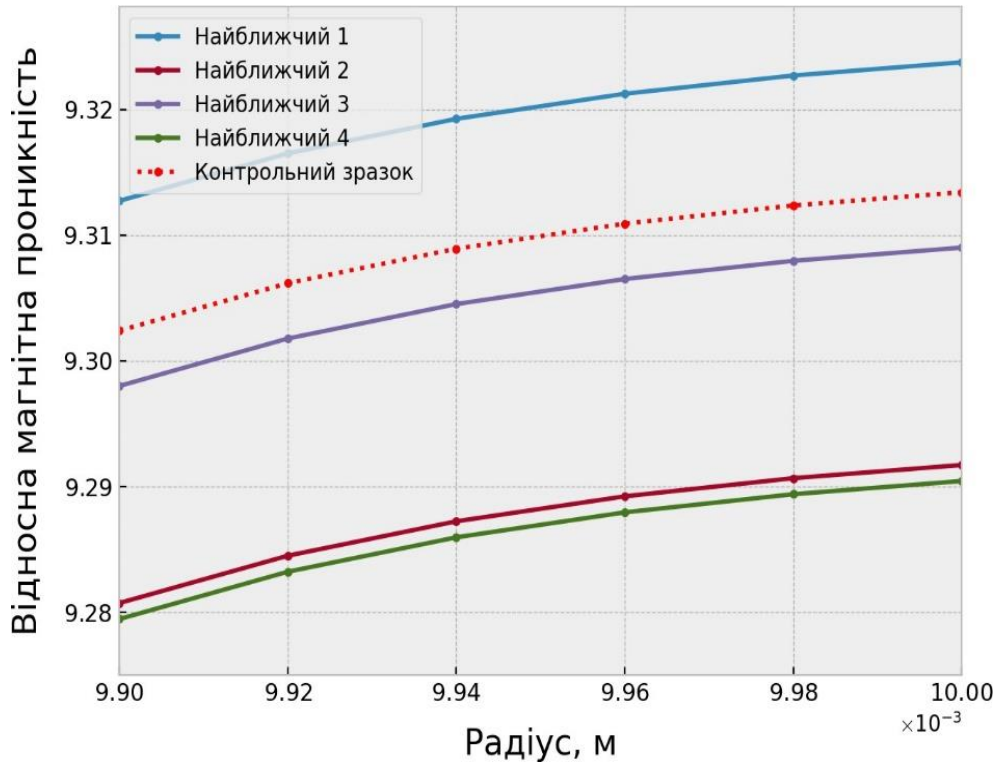


Рис. 4.21. Графічна інтерпретація пошуку найближчих до контрольного зразка профілів МП в таблиці 2-го рівня. (Радіальні профілі МП матеріалу ОК на приповерхневій глибині 0,1 мм, зокрема контрольний і найближчі).

Слід зазначити, що застосування мемоїзації таблиці другого рівня зачасту виключає повторне їх створення знову і скорочує загальний час пошуку результуючої інформації до декількох мілісекунд. Точність алгоритму є змінною та залежить від кількості зразків в таблиці, а отже дорівнює половині кроку між найближчими сусідніми, що дозволяє збільшувати точність вимірювань тільки у випадках необхідності і зменшувати її, коли потрібна максимальна швидкодія.

Список використаних джерел до глави 4

1. OPA541 High Power Monolithic Operational Amplifier. https://www.ti.com/lit/ds/symlink/opa541.pdf?ts=1754440642617

2. Roberge, J. K. (1975). *Operational amplifiers: theory and practice* (Vol. 197). New York: Wiley. https://d1wqtxts1xzle7.cloudfront.net/49038537/opamps181-libre.pdf?1474556082=&response-content-disposition=inline%3B+filename%3DOperational_Amplifiers_Theo

ry_and_Practi.pdf&Expires=1764514108&Signature=PF5FTkzy99OB8RNaLZnSHSqrhFti0YrYAJNUwsaprKyXEoZPKT0snJGeYXf30z7vKQdM0e7nOUk6AwJZyJcjf98txbGn0r-635M4nh8Mts4goiexKK6zxx-3tofC2jEJaFg-qgGuoCi0UL1Qo4lzs4rHXJEFf4i7TjXyvTOD2dLBzINaIqhPPbWU5oJBwcwU1UsGYG75ud1FuZduSX5Rsm5rB1UaqD1XdOUSQnWHlzY3DEJ54val6JWEPK8Pk0dhE7y4cXwpEOktASkgr57GfclooocXbI9tJZGwfwTNrUC2GrmE2BbKv1ZhJIJ5-KEhb8nkYjEHr5dnVxUswlryIyA__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA

3. Steer, M. (2019). Microwave and RF design. *NC State University: Raleigh, NC, USA*. https://www.fccdecastro.com.br/MO/Bibliografia/Microwave%20and%20RF%20Design...%20A%20Systems%20Approach%20-%20Michael%20Steer%20-%20(%20SciTech%20Publishing%20-%202010%20-%20pp.979).pdf

4. Analog Devices, Inc., *AD9834*. https://www.analog.com/media/en/technical-documentation/data-sheets/ad9834.pdf

5. Войцицький А. П. Електроніка і мікросхемотехніка : підручник / А. П. Войцицький, М. А. Войцицький / Вид. 2-ге, виправ. / Житомир: ЖНАЕУ. – Херсон: «Олді-Плюс», 2018. – 300 с https://library.kpi.kharkov.ua/files/new_postupleniya/elimiks.pdf

6. Сташук В.Д. *Радіоелектронні кола і пристрої*. Університет "Україна", 2007. – 360 с.

7. *MCP3465R*.16-Bit, 153.6kSPS, Single/Dual Channel ADC w/Vref. https://www.microchip.com/en-us/product/mcp3465r

8. *STM32F103C8*. Mainstream Performance line, Arm Cortex-M3 MCU with 64 Kbytes of Flash memory, 72 MHz CPU, motor control, USB and CAN. https://www.st.com/en/microcontrollers-microprocessors/stm32f103c8.html

9. Рябенький, В. М., Жуйков, В. Я., & Гулий, В. Д. Цифрова схемотехніка: навчальний посібник. Львів:" Новий Світ-2000", 2020. 736 с. https://ns2000.com.ua/wp-content/uploads/2019/07/TSyfrova_skhemotekhnika.pdf

10. Матвієнко М. П. *Проектування цифрових пристроїв*, К.: Ліра-К, 2018, - 364 с. https://knushop.com.ua/image/catalog/lira20230617/pdf/12423.pdf?srslt

id=AfmBOooLGGI3rcFzmcogoJ0sCUA6eHg0IFlHRUdxs_X6Wjw6_tGOTfQt .

11. Аналіз систем розпізнавання образів структури композитів : монографія / Добротвор І.Г., Стухляк П.Д., Микитишин А.Г., Митник М.М. – Тернопіль : Тернопільський національний технічний університет імені Івана Пулюя, 2018. – 224 с. http://elartu.tntu.edu.ua/handle/lib/25615

ЗАКЛЮЧЕННЯ

У монографії досліджується актуальна прикладна науково-технічна задача, яка полягає у створенні нового підходу до вихрострумових вимірювань профілів електрофізичних властивостей циліндричних металевих об’єктів. Запропоновано методологію, що об’єднує математичне моделювання, алгоритмічні та програмні засоби, а також апаратну реалізацію для забезпечення можливості одночасного визначення обох радіальних приповерхневих профілів електрофізичних характеристик матеріалу.

Важливим результатом дослідження є створення багатопараметрового експрес-методу, що дозволяє у реальному масштабі часу здійснювати визначення профілів електропровідності та магнітної проникності матеріалів. Метод ґрунтується на використанні апріорної інформації, накопиченої у нейромережевій сурогатній моделі процесу вихрострумового контролю, а також на застосуванні дворівневих таблиць пошуку Lookup tables. Це дало змогу досягти високої точності при одночасному значному скороченні обчислювальних витрат у порівнянні з традиційними аналітичними чи числовими методами.

Інноваційна новизна дослідження підтверджується низкою досягнень, які отримано в його межах. Зокрема обґрунтовано доцільність використання та створено однорідні комп’ютерні плани експериментів на основі R-послідовностей Робертса, що забезпечують рівномірність розподілу точок в проєктному просторі та низькі показники розбіжності, що є критично важливим для коректного навчання нейромережевих моделей. Запропоновано та реалізовано підхід до побудови сурогатної моделі процесу вихрострумового контролю на основі глибоких нейронних мереж, яка виконує не лише функції апроксимації, але й слугує накопичувачем апріорної інформації. Продемонстровано високу точність цієї моделі, що підтверджено результатами верифікації із застосуванням як спрощених аналітичних рішень, так і числових методів скінченних елементів.

Практичне значення отриманих результатів полягає у створенні спеціалізованого вихрострумового вимірювача профілів електрофізичних характеристик матеріалів у вигляді апаратно-програмного комплексу, який реалізує розроблений експрес-метод.

Виконані дослідження довели можливість одночасного вимірювання та ідентифікації розподілів електропровідності та магнітної проникності з похибкою, що не перевищує 0,5% для реальних прикладів. Це свідчить про придатність методу до практичного використання в умовах промисловості, де необхідні швидкість, точність і надійність діагностичного контролю.

Завдяки інтеграції сучасних методів математичного моделювання, технологій штучного інтелекту та експериментальних вимірювань сформовано універсальний підхід, що може бути поширений і на інші задачі багатопараметрового неруйнівного контролю. Запропоновані рішення здатні підвищити ефективність технологічних процесів, забезпечити своєчасне виявлення змін мікроструктури матеріалів та поліпшити надійність технічних систем у різних галузях промисловості.

Отже, виконані дослідження роблять внесок у розвиток теорії та практики вихрострумового контролю, створюючи наукове підґрунтя та прикладні інструменти для подальшого вдосконалення систем неруйнівної діагностики.

ДОДАТОК А

Програма створення багатовимірних комп'ютерних однорідних планів експериментів на основі R-послідовностей

Програма створення багатовимірного комп'ютерного плану експерименту на основі R-послідовностей

Область 1. Розрахунок коренів рівняння (розв'язок яких використовується в якості коефіцієнтів для створення ПЕ

Область 2. Генерування точок ПЕ для двовимірного факторного простору

$d := 2 \quad g2 := 1.3247179572447460 2596$

$N := 1000 \qquad i := 1..N$

$a1 := \frac{1}{g2} \qquad a2 := \frac{1}{g2 \cdot g2}$

$x_i := \mathrm{mod}(0.5 + i \cdot a1, 1) \quad y_i := \mathrm{mod}(0.5 + i \cdot a2, 1) \qquad X := \mathrm{augment}(x, y)$

$x_i =$

0.255
$9.755 \cdot 10^{-3}$
0.765
0.52
0.274
0.029
0.784
0.539
0.294
0.049
0.804
0.559
0.313
0.068
...

$y_i =$

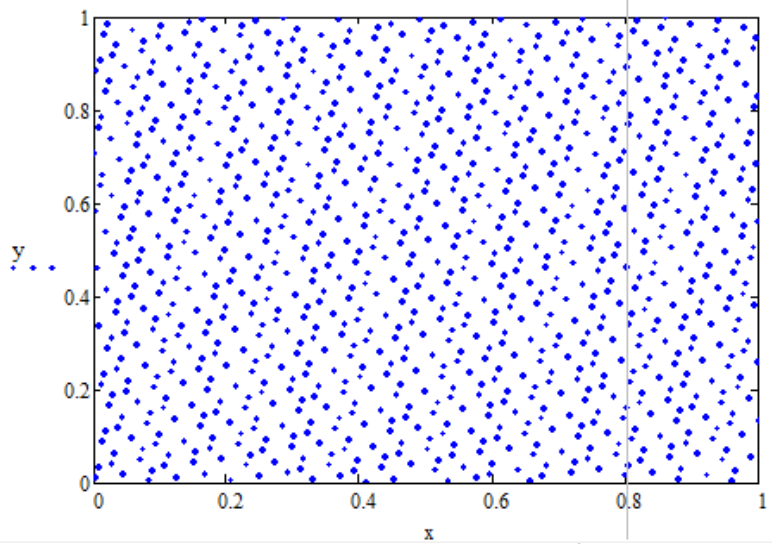
0.07
0.64
0.21
0.779
0.349
0.919
0.489
0.059
0.629
0.198
0.768
0.338
0.908
0.478
...

$X =$

	1	2
1	0.255	0.07
2	$9.755 \cdot 10^{-3}$	0.64
3	0.765	0.21
4	0.52	0.779
5	0.274	0.349
6	0.029	0.919
7	0.784	0.489
8	0.539	0.059
9	0.294	0.629
10	0.049	0.198
11	0.804	0.768
12	0.559	0.338
13	0.313	0.908
14	0.068	0.478
15	0.823	0.048
16	0.578	...

$X =$

	1	2
1	0.255	0.07
2	$9.755 \cdot 10^{-3}$	0.64
3	0.765	0.21
4	0.52	0.779
5	0.274	0.349
6	0.029	0.919
7	0.784	0.489
8	0.539	0.059
9	0.294	0.629
10	0.049	0.198
11	0.804	0.768
12	0.559	0.338
13	0.313	0.908
14	0.068	0.478
15	0.823	0.048
16	0.578	...



$$CD2 := \left(\frac{13}{12}\right)^d - \left[\left(\frac{2}{N}\right) \cdot \sum_{k=1}^{N} \prod_{j=1}^{d} \left[1 + \left(\frac{1}{2}\right) \cdot |X_{k,j} - 0.5| - \left(\frac{1}{2}\right) \cdot (|X_{k,j} - 0.5|)^2\right]\right] \ldots$$
$$+ \frac{1}{N^2} \cdot \left[\sum_{k=1}^{N} \sum_{j=1}^{N} \prod_{i=1}^{d} \left[1 + \left(\frac{1}{2}\right) \cdot |X_{k,i} - 0.5 + \left(\frac{1}{2}\right) \cdot |X_{j,i} - 0.5| - \left(\frac{1}{2}\right) \cdot |X_{k,i} - X_{j,i}|\right]\right] \qquad CD2 = 4.200576 \times 10^{-6}$$

$$WD2 := \left(\frac{4}{3}\right)^d + \left(\frac{1}{N^2}\right) \cdot \sum_{k=1}^{N} \sum_{j=1}^{N} \prod_{i=1}^{d} \left[\left(\frac{3}{2}\right) - |X_{k,i} - X_{j,i}| \cdot (1 - |X_{k,i} - X_{j,i}|)\right] \qquad WD2 = 3.555562$$

Область 2. Генерування точок ПЕ для двовимірного факторного простору

Область 3. Генерування точок ПЕ для тривимірного факторного простору

$d := 3 \qquad g3 := 1.220744084605759475361$

$N := 1000 \qquad i := 1..N$

$a1 := \frac{1}{g3} \qquad a2 := \frac{1}{g3 \cdot g3} \qquad a3 := \frac{1}{g3 \cdot g3 \cdot g3}$

$x_i := \mathrm{mod}(0.5 + i \cdot a1, 1) \qquad y_i := \mathrm{mod}(0.5 + i \cdot a2, 1) \qquad z_i := \mathrm{mod}(0.5 + i \cdot a3, 1) \qquad X := \mathrm{augment}(x, y, z)$

$x_i =$	$y_i =$	$z_i =$
0.319	0.171	0.05
0.138	0.842	0.599
0.958	0.513	0.149
0.777	0.184	0.699
0.596	0.855	0.249
0.415	0.526	0.798
0.234	0.197	0.348
0.053	0.868	0.898
0.873	0.539	0.447
0.692	0.21	0.997
0.511	0.881	0.547
0.33	0.553	0.096
0.149	0.224	0.646
0.968	0.895	0.196
0.788	0.566	0.746
0.607	0.237	0.295
0.426	0.908	0.845
0.245	0.579	0.395
0.064	0.25	0.944
0.883	0.921	0.494
0.703	0.592	0.044
0.522	0.263	0.593
0.341	0.934	0.143
0.16	0.605	0.693
0.979	0.276	0.243
0.798	0.947	0.792
0.618	0.618	0.342
0.437	0.289	0.892
0.256	0.96	0.441
0.075	0.631	0.991
0.894	0.302	0.541
0.714	0.973	0.09
0.533	0.644	0.64
0.352	0.315	0.19
0.171	0.987	0.74
0.99	0.658	0.289
0.809	0.329	0.839
0.629	1	0.389
0.448	0.671	0.938
0.267	0.342	0.488
0.086	...	...
...		

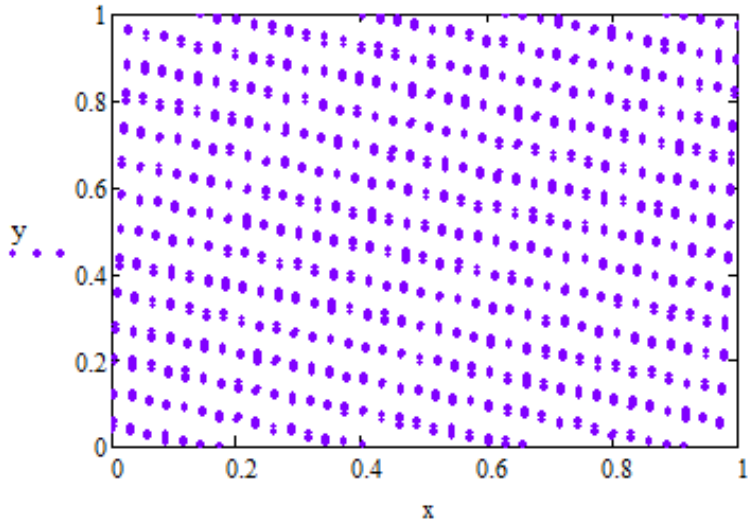
$X =$

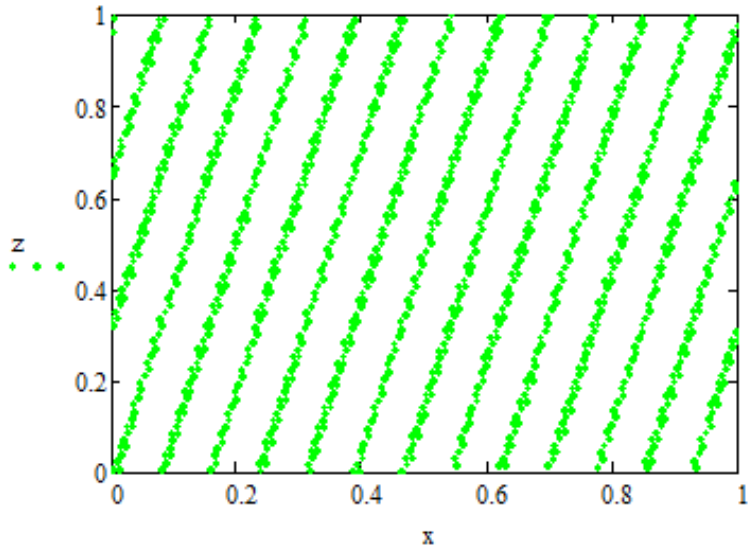
	1	2	3
1	0.319	0.171	0.05
2	0.138	0.842	0.599
3	0.958	0.513	0.149
4	0.777	0.184	0.699
5	0.596	0.855	0.249
6	0.415	0.526	0.798
7	0.234	0.197	0.348
8	0.053	0.868	0.898
9	0.873	0.539	0.447
10	0.692	0.21	0.997
11	0.511	0.881	0.547
12	0.33	0.553	0.096
13	0.149	0.224	0.646
14	0.968	0.895	0.196
15	0.788	0.566	0.746
16	0.607	0.237	0.295
17	0.426	0.908	0.845
18	0.245	0.579	0.395
19	0.064	0.25	0.944
20	0.883	0.921	0.494
21	0.703	0.592	0.044
22	0.522	0.263	0.593
23	0.341	0.934	0.143
24	0.16	0.605	0.693
25	0.979	0.276	0.243
26	0.798	0.947	0.792
27	0.618	0.618	0.342
28	0.437	0.289	0.892
29	0.256	0.96	0.441
30	0.075	0.631	0.991
31	0.894	0.302	0.541
32	0.714	0.973	0.09
33	0.533	0.644	0.64
34	0.352	0.315	0.19
35	0.171	0.987	0.74
36	0.99	0.658	0.289
37	0.809	0.329	0.839
38	0.629	1	0.389
39	0.448	0.671	0.938
40	0.267	0.342	0.488
41	0.086	0.013	0.038
42	0.905	0.684	...

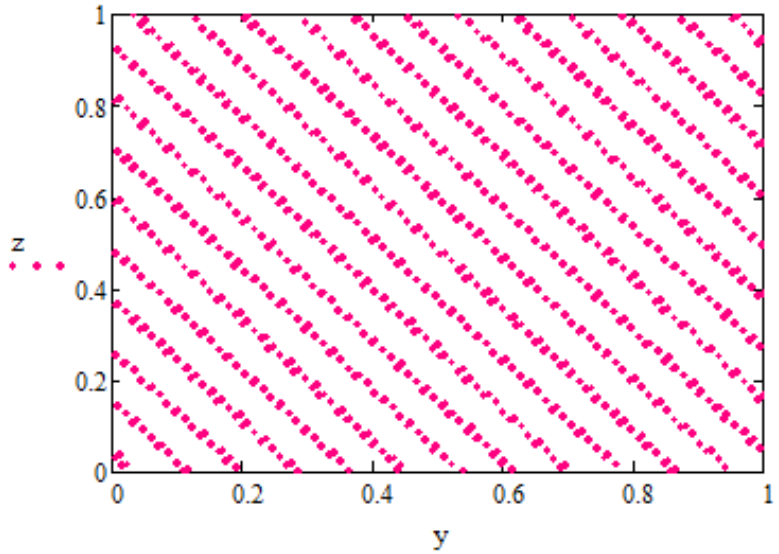
$$CD3 := \left(\frac{13}{12}\right)^d - \left[\left(\frac{2}{N}\right) \cdot \sum_{k=1}^{N} \prod_{j=1}^{d} \left[1 + \left(\frac{1}{2}\right) \cdot |X_{k,j} - 0.5| - \left(\frac{1}{2}\right) \cdot \left(|X_{k,j} - 0.5|\right)^2\right]\right] \ldots$$

$$+ \frac{1}{N^2} \cdot \left[\sum_{k=1}^{N} \sum_{j=1}^{N} \prod_{i=1}^{d} \left[1 + \left(\frac{1}{2}\right) \cdot |X_{k,i} - 0.5 + \left(\frac{1}{2}\right) \cdot |X_{j,i} - 0.5| - \left(\frac{1}{2}\right) \cdot |X_{k,i} - X_{j,i}\right]\right] \qquad CD3 = 4.386 \times 10^{-5}$$

$$WD3 := \left(\frac{4}{3}\right)^d + \left(\frac{1}{N^2}\right) \cdot \sum_{k=1}^{N} \sum_{i=1}^{N} \prod_{i=1}^{d} \left[\left(\frac{3}{2}\right) - |X_{k,i} - X_{j,i}| \cdot \left(1 - |X_{k,i} - X_{j,i}|\right)\right] \qquad WD3 = 4.740774$$







Область 3. Генерування точок ПЕ для тривимірного факторного простору

ДОДАТОК Б
Комплекс програм реалізації моделі Dood

```
# Документація скриптів розрахунку моделі Dodd - векторний потенціал та напруга

## Огляд системи

Цей проект реалізує аналітичну модель Додда для вихрострумового контролю, засновану на документі ORNL-5220 "Analysis and Computer Programs for Eddy Current Coils Concentric with Multiple Cylindrical Conductors" (1979). Система обчислює електромагнітні поля в циліндричних провідникових системах з використанням функцій Гріна.

## Основні компоненти

### 1. VectorPotentialInsideCoilGreenFunction

**Розташування:** `vector_potential_calculation/vector_potential_inside_coil_green_function.py`

**Призначення:** Основний клас для обчислення векторного потенціалу всередині котушки з використанням підходу функцій Гріна.

**Ключові методи:**
- `normalization_1_coil__sigma_mu_r(input_params)` - нормалізує параметри для розрахунків однієї котушки
- `calculate(norm_parameters, integ_top_range)` - виконує основне обчислення векторного потенціалу
- `frequency_depend_part1()`, `frequency_depend_part2()` - обчислює частотно-залежні компоненти
- `integrand()` - підінтегральна функція для чисельного інтегрування

**Теоретична основа:**
- Реалізує рівняння 49 з документа ORNL-5220
- Використовує функції Бесселя для циліндричних координат
- Враховує багатошарові провідникові системи
- Застосовує адаптивне інтегрування з квадратурами Гауса

### 2. VoltageFromVectorPotential

**Розташування:** `calculation_helpers/voltage_from_vector_potential.py`

**Призначення:** Конвертує векторний потенціал у вимірювану напругу, використовуючи закон Фарадея.

**Ключовий метод:**
- `from_delta_func(vec_pot, frequency, radius)` - статичний метод для перетворення

**Формула:** `V = jωA × 2πr`
- `j` - уявна одиниця (√-1)
- `ω` = 2πf (кутова частота)
- `A` - векторний потенціал
- `r` - радіус
### 3. Робочий процес в скрипті `method_lut/4_calc_analytic_result.py`

Скрипт реалізує метод lookup table (LUT) для аналітичних розрахунків з наступним алгоритмом:

```python
# Крок 1: Завантаження вхідних даних
with open(input_file_name) as json_file:
    packed_data = json.load(json_file)

# Крок 2: Розпакування даних шарів
raw_data_list = unpack_layers_data(copy.deepcopy(packed_data))

# Крок 3: Створення об'єкта розрахунку
calcObject = VectorPotentialInsideCoilGreenFunction()
```
```

```
# Крок 4: Нормалізація параметрів для кожного набору даних
normalized_data_list = []
for data in raw_data_list:
    normalized_data = calcObject.normalization_1_coil__sigma_mu_r(data)
    normalized_data_list.append(normalized_data)

# Крок 5: Обчислення векторного потенціалу для всіх наборів
results_list = []
integration_range = 50
for input_data in normalized_data_list:
    result = calcObject.calculate(input_data, integration_range)
    results_list.append(result)

# Крок 6: Перетворення у напругу та збереження результатів
for index, result in enumerate(results_list):
    data_item = packed_data[index]
    vec_pot_complex = result
    frequency = data_item['frequency'][0]
    radius = data_item['calc_point']['r']

    # Перетворення векторного потенціалу в напругу
    voltage_complex = VoltageFromVectorPotential.from_delta_func(
        vec_pot_complex, frequency, radius)

    # Перетворення у полярні координати
    voltage_polar = Convertations.complex_to_polar(voltage_complex)
    amplitude, phase = voltage_polar

    # Збереження результатів у вихідній структурі даних
    packed_data[index]["result_voltage"] = {
        "real": voltage_complex.real,
        "imag": voltage_complex.imag,
    }
    packed_data[index]["result_voltage_polar"] = {
        "amplitude": amplitude,
        "phase": phase,
    }
    packed_data[index]["result_vector_potential"] = {
        "real": result.real,
        "imag": result.imag,
    }
```

Особливості методу LUT:
- **Структура каталогів**: `method_lut/datasets_output/model_input_data/` → `method_lut/datasets_output/calculated/`
- **Пакетна обробка**: Обробляє великі набори даних (до 10,000 зразків)
- **Багатопроцесорність**: Використовує `multiprocessing.Process` для паралельної обробки
- **Збереження результатів**: Зберігає як комплексні, так і полярні представлення результатів

Фізичне значення

Векторний потенціал - проміжна електромагнітна величина, що представляє розподіл магнітного поля в просторі.

Напруга - кінцева вимірювана величина, що з'являється в приймальній котушці внаслідок електромагнітної індукції.

Зв'язок: Напруга обчислюється з векторного потенціалу через закон Фарадея, що описує електромагнітну індукцію.

Математичні основи

- **Функції Гріна:** Аналітичні рішення для електромагнітних польових задач
- **Багатошарова аналіза:** Обробка складних геометрій з декількома провідними шарами
- **Частотний аналіз:** Обчислення частотно-залежних електромагнітних відгуків
- **Функції Бесселя:** Спеціальні математичні функції для циліндричних координат

Повний список файлів скриптів

Основні розрахункові файли

1. Головний скрипт
- **`method_lut/4_calc_analytic_result.py`** - **ОСНОВНИЙ СКРИПТ** - Аналітичні розрахунки методом lookup table

2. Розрахунки векторного потенціалу (`vector_potential_calculation/`)

- **`vector_potential_calculation/vector_potential_inside_coil_green_function.py`** - Основний клас з функціями Гріна
- **`vector_potential_calculation/vector_potential.py`** - Альтернативна реалізація (рівняння 59 ORNL-5220)
- **`vector_potential_calculation/vector_potential_greens_function.py`** - Спеціалізовані функції Гріна
- **`vector_potential_calculation/vector_potential_2_layers.py`** - Розрахунки для 2-шарових систем
- **`vector_potential_calculation/vector_potential_2_layers_coil_3reg_67.py`** - 3-регіонний підхід (версія 67)
- **`vector_potential_calculation/vector_potential_2_layers_coil_3reg_69.py`** - 3-регіонний підхід (версія 69)
- **`vector_potential_calculation/vector_potential_2_layers_delta_f67.py`** - Дельта-функціональний підхід (v67)
- **`vector_potential_calculation/vector_potential_2_layers_delta_f69.py`** - Дельта-функціональний підхід (v69)
- **`vector_potential_calculation/vector_potential_2_layers_delta_f2010.py`** - Дельта-функції (формулювання 2010)
- **`vector_potential_calculation/vector_potential_3reg.py`** - Трирегіонний метод
- **`vector_potential_calculation/vector_potential_parts.py`** - Компонентні розрахунки

3. Допоміжні розрахунки (`calculation_helpers/`)

- **`calculation_helpers/voltage_from_vector_potential.py`** - Перетворення потенціалу в напругу
- **`calculation_helpers/convertations.py`** - Перетворення комплексних чисел
- **`calculation_helpers/gauss_integration.py`** - Квадратури Гауса
- **`calculation_helpers/adaptive_integration_functions.py`** - Адаптивне інтегрування
- **`calculation_helpers/cycles_adaptive_integration.py`** - Циклічне адаптивне інтегрування
- **`calculation_helpers/normalization_functions.py`** - Нормалізація параметрів
- **`calculation_helpers/convertation_si.py`** - Перетворення в систему СІ

4. Спеціальні функції (`special_functions/`)

- **`special_functions/integral_bessel_functions.py`** - Інтеграли функцій Бесселя
- **`special_functions/matrices_calculation.py`** - Матричні операції для багатошарових систем
- **`special_functions/bessel_functions.py`** - Функції Бесселя
- **`special_functions/greens_function.py`** - Реалізації функцій Гріна

5. Генератори вхідних даних (`input_data_generators/`)

- **`input_data_generators/helpers.py`** - Упакування/розпакування параметрів
- **`input_data_generators/get_input_parameters.py`** - Генерація стандартних параметрів
- **`input_data_generators/test_data_generator.py`** - Генерація тестових випадків
- **`input_data_generators/layers_generator.py`** - Генерація багатошарових систем

```
- **`input_data_generators/thin_input_parameters.py`** - Параметри для тонких провідників
- **`input_data_generators/get_parameters_okita.py`** - Параметри за підходом Окіти
- **`input_data_generators/dist_test.py`** - Тестування розподілів параметрів
- **`input_data_generators/easing_functions/easing.py`** - Функції згладжування переходів

### 6. Тестові та валідаційні скрипти

**Основні тести:**
- **`vector_potential_main_test.py`** - Головний тест векторного потенціалу
- **`impendance_main_test.py`** - Тестування імпедансу
- **`impendance_dodd69_test.py`** - Валідація моделі Додда 1969
- **`vector_potential_2_layers_coil_test.py`** - Тест 2-шарових котушок

**Додаткові тести:**
- `impendance_main_test_without_loop.py`
- `impendance_main_thin_test.py`
- `impendance_okita_compare_test.py`
- `impendance_okita_test.py`
- `vector_potential_2_layers_delta_f_test.py`
- `vector_potential_2_layers_main_test.py`
- `vector_potential_calc_lpt_expreriment_1.py`
- `vector_potential_okita_compare_test.py`
- `vector_potential_params_distribution_test.py`
- `vector_potential_params_distribution_test_radius.py`
- `vector_potential_to_voltage_lpt_experiments_1.py`

### 7. Методи lookup table (`method_lut/`)

- **`method_lut/3_create_model_input_data.py`** - **ОСНОВНИЙ ГЕНЕРАТОР** - Створення модельних вхідних даних
- **`method_lut/4_calc_analytic_result.py`** - **ОСНОВНИЙ РОЗРАХУНКОВИЙ СКРИПТ** - Аналітичні розрахунки методом LUT

## Архітектура системи

Система побудована за модульним принципом:

1. **Фізичні моделі** - Реалізують теорію електромагнетизму
2. **Математичні методи** - Спеціальні функції та чисельні методи
3. **Допоміжні утиліти** - Перетворення даних та одиниць
4. **Генератори параметрів** - Створення тестових конфігурацій
5. **Валідаційні тести** - Перевірка точності розрахунків

Ця система забезпечує повний цикл моделювання вихрострумового контролю від задання параметрів до отримання кінцевих результатів у вигляді напруги.

---

# Список файлів, пов'язаних з VectorPotentialInsideCoilGreenFunction

## 1. Визначення класу

### Основний файл класу:
- **`vector_potential_calculation/vector_potential_inside_coil_green_function.py`**
  - Містить повне визначення класу `VectorPotentialInsideCoilGreenFunction`
  - Реалізує підхід функцій Гріна для обчислення векторного потенціалу
  - Базується на рівнянні 49 з документа ORNL-5220

## 2. Батьківські класи (ієрархія наслідування)

Клас використовує множинне наслідування від шести батьківських класів:
```

```
### Математичні методи:
- **`calculation_helpers/gauss_integration.py`**
  - Батьківський клас `GaussIntegration`
  - Забезпечує методи інтегрування за квадратурами Гауса

- **`calculation_helpers/adaptive_integration_functions.py`**
  - Батьківський клас `AdaptiveIntegrationFunctions`
  - Забезпечує функціональність адаптивного інтегрування

- **`calculation_helpers/cycles_adaptive_integration.py`**
  - Батьківський клас `CyclesAdaptiveIntegration`
  - Забезпечує циклічне адаптивне інтегрування

### Допоміжні функції:
- **`calculation_helpers/normalization_functions.py`**
  - Батьківський клас `NormalizationFunctions`
  - Містить утиліти нормалізації параметрів

### Спеціальні математичні функції:
- **`special_functions/integral_bessel_functions.py`**
  - Батьківський клас `IntegralBesselFunctions`
  - Забезпечує методи інтегрування функцій Бесселя

- **`special_functions/matrices_calculation.py`**
  - Батьківський клас `MatricesCalculation`
  - Забезпечує утиліти матричних обчислень

## 3. Файли, що активно використовують клас

### Основні розрахункові скрипти:
- **`method_lut/4_calc_analytic_result.py`** - **ОСНОВНИЙ СКРИПТ** - Аналітичні розрахунки методом lookup table (LUT)

### Тестові скрипти:
- **`vector_potential_2_layers_coil_test.py`** - Тестування векторного потенціалу для 2-шарових котушок
- **`vector_potential_2_layers_coil_test_2.py`** - Додаткові тести 2-шарових конфігурацій котушок
- **`vector_potential_calc_lpt_expreriment_1.py`** - Експерименти з розрахунками векторного потенціалу
- **`vector_potential_params_distribution_test.py`** - Тестування розподілів параметрів
- **`vector_potential_params_distribution_test_radius.py`** - Тестування варіацій радіуса

### Експериментальні скрипти:
- **`article_1_verification_vecpot_calc.py`** - Верифікаційні розрахунки для статті
- **`article_1_experiments_vecpot_calc.py`** - Експериментальні розрахунки для дослідження
- **`article_1_lpt_vecpot_calc.py`** - LPT розрахунки векторного потенціалу для статті

### Аналіз напруги та багатошарових систем:
- **`voltage_diff_layers_quantity.py`** - Аналіз різниць напруги для різної кількості шарів
- **`tandem_1_4_0__wrong_input_normalize.py`** - Тандемний аналіз з тестуванням нормалізації
- **`tandem_1_5_sample_test.py`** - Тандемне тестування зразків

## 4. Файли з закоментованим використанням

### Скрипти візуалізації та обробки:
- **`plots_dist.py`** - Побудова графіків розподілів (закоментований імпорт)
- **`disertation_plot__plots_dist.py`** - Графіки для дисертації (закоментований імпорт)

### Генерація даних:
- **`method_lut/3_create_model_input_data.py`** - **ОСНОВНИЙ ГЕНЕРАТОР** - Створення вхідних даних для методу lookup table
- **`tandem_1_3_input_data_gen.py`** - Генерація вхідних даних для тандемного аналізу
```

5. Супутні класи векторного потенціалу

Альтернативні реалізації:
- **`vector_potential_calculation/vector_potential_greens_function.py`** - Споріднена реалізація функцій Гріна
- **`vector_potential_calculation/vector_potential.py`** - Базова реалізація векторного потенціалу
- **`vector_potential_calculation/vector_potential_3reg.py`** - 3-регіонна реалізація

Спеціалізовані 2-шарові реалізації:
- **`vector_potential_calculation/vector_potential_2_layers.py`** - Базова 2-шарова реалізація
- **`vector_potential_calculation/vector_potential_2_layers_coil_3reg_67.py`** - 2-шарова котушка (версія 67)
- **`vector_potential_calculation/vector_potential_2_layers_coil_3reg_69.py`** - 2-шарова котушка (версія 69)
- **`vector_potential_calculation/vector_potential_2_layers_delta_f67.py`** - Дельта-функціональний підхід (v67)
- **`vector_potential_calculation/vector_potential_2_layers_delta_f69.py`** - Дельта-функціональний підхід (v69)
- **`vector_potential_calculation/vector_potential_2_layers_delta_f2010.py`** - Дельта-функції (формулювання 2010)

6. Документаційні файли

- **`ДОКУМЕНТАЦІЯ_СКРИПТІВ_РОЗРАХУНОК_МОДЕЛІ_DODD.md`** - Українська документація з поясненням функціональності класу
- **`method_lut/README_METHOD_LUT.md`** - Документація призначення скриптів у каталозі method_lut

Використання в проекті

`VectorPotentialInsideCoilGreenFunction` є центральним компонентом проекту аналітичного моделювання для аналізу вихрострумових котушок. Клас активно використовується в **14 файлах** і згадується в **5 додаткових файлах**.

Основний робочий процес (method_lut):

```
method_lut/3_create_model_input_data.py → method_lut/4_calc_analytic_result.py
        ↓ (генерує)                              ↓ (обробляє)
    model_input_data/                        calculated/
    input_data_q10000_var0.3_0.3.json  →    full_data_q10000_var0.3_0.3.json
```

Основні сфери застосування:

1. **Основний конвеєр LUT** - `method_lut/4_calc_analytic_result.py` для промислового використання
2. **Експериментальна валідація** - верифікація та експерименти для статей
3. **Параметричне тестування** - тести розподілів, варіації радіуса
4. **Багатошаровий аналіз** - тестування 2-шарових конфігурацій котушок
5. **Розрахунки напруги** - через обчислення векторного потенціалу

Архітектурні особливості:

Клас використовує множинне наслідування для об'єднання математичних можливостей з 6 різних батьківських класів, що робить його комплексним інструментом для розрахунків векторного потенціалу в аналізі вихрострумових полів. Основний робочий процес зосереджений на методі lookup table, що забезпечує ефективну обробку великих обсягів даних (до 10,000 зразків) з використанням багатопроцесорності.

```
# Повний список файлів коду, залучених до VectorPotentialInsideCoilGreenFunction

## Основний файл класу

**`vector_potential_calculation/vector_potential_inside_coil_green_function.py`**
- Містить головний клас VectorPotentialInsideCoilGreenFunction
- Реалізує підхід функцій Гріна для обчислення векторного потенціалу в точках всередині котушок
- Базується на документі ORNL-5220

## Батьківські класи (множинне наслідування)

### 1. Математичні методи інтегрування

**`calculation_helpers/gauss_integration.py`**
- **Клас:** `GaussIntegration`
- **Функціональність:** Методи інтегрування Гауса для чисельного інтегрування
- **Методи:** `gauss_integration()`, `gauss2_integration()` з N-точковими формулами (N=2,4,6,8,10)

**`calculation_helpers/adaptive_integration_functions.py`**
- **Клас:** `AdaptiveIntegrationFunctions`
- **Функціональність:** Алгоритми адаптивного інтегрування з тестуванням збіжності
- **Методи:** `adapt()`, `adapt2()` для адаптивного чисельного інтегрування

**`calculation_helpers/cycles_adaptive_integration.py`**
- **Клас:** `CyclesAdaptiveIntegration`
- **Функціональність:** Циклічні методи інтегрування для ітеративного інтегрування по діапазонах
- **Методи:** `adapt_integ_cycle()`, `adapt2_integ_cycle()` (статичні методи)

### 2. Допоміжні функції

**`calculation_helpers/normalization_functions.py`**
- **Клас:** `NormalizationFunctions`
- **Функціональність:** Нормалізація параметрів котушки та перетворення координат
- **Методи:** методи нормалізації параметрів та конвертації одиниць

### 3. Спеціальні математичні функції

**`special_functions/integral_bessel_functions.py`**
- **Клас:** `IntegralBesselFunctions`
- **Функціональність:** Спеціалізовані інтеграли функцій Бесселя для електромагнітних розрахунків
- **Методи:** `python_xiint()`, `python_xkknt()` та споріднені методи інтегралів Бесселя

**`special_functions/matrices_calculation.py`**
- **Клас:** `MatricesCalculation`
- **Функціональність:** Обчислення матриць U та V для розрахунків провідників
- **Методи:** `calc_uv_matrices()` та функції матричних обчислень

## Непрямі залежності (через ланцюг наслідування)

**`special_functions/bessel_functions.py`**
- **Клас:** `BesselFunctions`
- **Роль:** Базовий клас (дідівський клас) для MatricesCalculation
- **Функціональність:** Базові реалізації функцій Бесселя з використанням SciPy special functions
- **Методи:** `bess_i0()`, `bess_k0()`, `bess_i1()`, `bess_k1()`, `cmd_bess()`

## Конфігураційні файли пакетів

**`calculation_helpers/__init__.py`**
- Файл ініціалізації пакету calculation_helpers (порожній)

**`special_functions/__init__.py`**
- Файл ініціалізації пакету special_functions (порожній)
```

```
394
395  ## Зовнішні залежності (стандартні бібліотеки)
396
397  - **numpy** - Чисельні масиви та математичні операції
398  - **scipy.constants** - Фізичні константи (μ₀, π)
399  - **scipy.special** - Функції Бесселя (через BesselFunctions)
400  - **scipy.integrate** - Чисельне інтегрування (через IntegralBesselFunctions)
401  - **logging** - Функціональність логування
402
403  ## Підсумок залежностей
404
405  VectorPotentialInsideCoilGreenFunction залежить від **9 файлів Python коду** (включаючи себе):
406
407  ### Структура наслідування:
408  ```
409  VectorPotentialInsideCoilGreenFunction
410  ├── GaussIntegration (calculation_helpers/gauss_integration.py)
411  ├── AdaptiveIntegrationFunctions (calculation_helpers/adaptive_integration_functions.py)
412  ├── CyclesAdaptiveIntegration (calculation_helpers/cycles_adaptive_integration.py)
413  ├── NormalizationFunctions (calculation_helpers/normalization_functions.py)
414  ├── IntegralBesselFunctions (special_functions/integral_bessel_functions.py)
415  └── MatricesCalculation (special_functions/matrices_calculation.py)
416      └── BesselFunctions (special_functions/bessel_functions.py)
417  ```
418
419  ### Категорії файлів:
420  - **1** головний файл класу
421  - **6** батьківських класів (множинне наслідування)
422  - **1** дідівський клас (через ланцюг наслідування)
423  - **1** додаткова залежність через ланцюг імпортів
424
425  Всі ці файли працюють разом для реалізації складної системи розрахунку електромагнітного поля,
     заснованої на аналітичній моделі, описаній в ORNL-5220 для аналізу вихрових струмів у циліндричних
     провідниках.
```

Модель Dood

Розрахунок векторного потенціалу

`vector_potential_calculation/vector_potential_inside_coil_green_function.py`

```python
#!/usr/bin/env python
# coding: utf-8

import numpy as np
from scipy import constants

from calculation_helpers.gauss_integration import GaussIntegration
from calculation_helpers.adaptive_integration_functions import AdaptiveIntegrationFunctions
from calculation_helpers.cycles_adaptive_integration import CyclesAdaptiveIntegration
from calculation_helpers.normalization_functions import NormalizationFunctions
from special_functions.integral_bessel_functions import IntegralBesselFunctions
from special_functions.matrices_calculation import MatricesCalculation

import logging
logger = logging.getLogger(__name__)
logger.setLevel(logging.INFO)

formatter = logging.Formatter('%(asctime)s:%(name)s:%(message)s')
log_file_name = 'logs/vector_potential_inside_coil_green_function.log'
```

```
20
21  # clear log
22  with open(log_file_name, 'w'):
23      pass
24
25  file_handler = logging.FileHandler(log_file_name)
26  file_handler.setLevel(logging.DEBUG)
27  file_handler.setFormatter(formatter)
28
29  stream_handler = logging.StreamHandler()
30  stream_handler.setFormatter(formatter)
31
32  logger.addHandler(file_handler)
33  logger.addHandler(stream_handler)
34
35
36  # VectorPotential class
37  # computes vector potential in point
38  # induced by 1 coil (eq. 49, p. 10)
39  # ORNL-5220
40  # UC32-Mathematics and Computers
41  # ANALYSIS AND COMPUTER PROGRAMS FOR
42  # EDDY CURRENT COILS CONCE TRIC WITH
43  # MULTIPLE CYLINDRICAL CONDUCTORS
44  # C.W. Nestor, Jr., C.V.Dodd, W.E. Deeds, July (1979))
45  class VectorPotentialInsideCoilGreenFunction(
46          GaussIntegration,
47          AdaptiveIntegrationFunctions,
48          CyclesAdaptiveIntegration,
49          NormalizationFunctions,
50          IntegralBesselFunctions,
51          MatricesCalculation):
52
53      def __init__(self):
54          pass
55
56      # numerator part 1 divided by alpha^3
57      def frequency_depend_part1(self, alpha, parameters):
58          calc_point = parameters['calc_point']
59          z = calc_point["z"]
60
61          coils_parameters = parameters['coils_parameters']
62          l1 = coils_parameters[0]['l1']
63          l2 = coils_parameters[0]['l2']
64
65          numerator = np.sin(alpha * (z - l1)) - np.sin(alpha * (z - l2))
66          denominator = alpha**3
67
68          result = numerator / denominator
69
70          return result
71
72      def frequency_depend_part2(self, alpha, parameters):
73          coils_parameters = parameters['coils_parameters']
74          r1 = coils_parameters[0]["r1"]
75          r2 = coils_parameters[0]["r2"]
76
77          inner_parameters = parameters['inner_parameters']
78          outer_parameters = parameters['outer_parameters']
79
```

```
        calc_point = parameters['calc_point']
        r = calc_point["r"]

        u_col = np.zeros(4)
        v_col = np.zeros(4)

        ir1r2 = super().python_xiint(alpha, r1, r2)
        kr1r2 = super().python_xkknt(alpha, r1, r2)
        i1a0r = super().bess_i1(alpha * r)
        k1a0r = super().bess_k1(alpha * r)

        # set up U matrix for no external conductors
        u_col[2] = 1.0

        # calculate U matrix for external conductors (if any)
        # (flag INN for UVMAT = 1)
        if len(outer_parameters) != 0:
            u_col = super().calc_uv_matrices(alpha, 1, u_col, parameters)

        # set up V matrix for no internal conductors
        v_col[0] = 1.0

        # calculate V matrix for internal conductors (if any)
        # (flag INN for UVMAT = 0)
        if len(inner_parameters) != 0:
            v_col = super().calc_uv_matrices(alpha, 0, v_col, parameters)

        u12 = complex(u_col[0], u_col[1])
        u22 = complex(u_col[2], u_col[3])
        v11 = complex(v_col[0], v_col[1])
        v21 = complex(v_col[2], v_col[3])

        numerator_part1 = v11 * i1a0r + v21 * k1a0r
        numerator_part2 = u12 * ir1r2 + u22 * kr1r2

        numerator = (numerator_part1 * numerator_part2)
        denominator = u22 * v11 - u12 * v21

        result = numerator / denominator

        return result

    def frequency_depend_part(self, alpha, parameters):
        part1 = self.frequency_depend_part1(alpha, parameters)
        part2 = self.frequency_depend_part2(alpha, parameters)

        result = part1 * part2

        return result

    def integrand(self, alpha, parameters):
        frequency_depend_result = self.frequency_depend_part(alpha, parameters)
        complex_result = frequency_depend_result

        result = complex_result.real, complex_result.imag

        return result
```

```
    # number of turns per unit area
    def turns_per_meter(self, parameters):
        coils_parameters = parameters['coils_parameters']
        r1 = coils_parameters[0]["r1"]
        r2 = coils_parameters[0]["r2"]
        l1 = coils_parameters[0]["l1"]
        l2 = coils_parameters[0]["l2"]
        n = coils_parameters[0]["n"]

        nc = (n / ((r2 - r1) * (l2 - l1)))

        return nc

    def constant_expresion(self, parameters):
        electrical_values = parameters['electrical_values']
        current = electrical_values['current']

        nc = self.turns_per_meter(parameters)

        numerator = nc * current * constants.mu_0

        result = numerator / constants.pi

        return result

    def calculate(self, norm_parameters, integ_top_range):
        constant_expresion = self.constant_expresion(norm_parameters)
        calc_func = self.integrand
        integr_method = super().gauss2_integration
        adapt2 = super().adapt2

        result_adapt2 = super().adapt2_integ_cycle(
            calc_func, integr_method, adapt2, norm_parameters, integ_top_range)

        result = constant_expresion * result_adapt2

        return result
```

Модель Dood

Альтернативний розрахунок векторного потенціалу

`vector_potential_calculation/vector_potential.py`- (рівняння 59 ORNL-5220)

```
#!/usr/bin/env python
# coding: utf-8

import numpy as np
from scipy import constants

from calculation_helpers.gauss_integration import GaussIntegration
from calculation_helpers.adaptive_integration_functions import AdaptiveIntegrationFunctions
from calculation_helpers.cycles_adaptive_integration import CyclesAdaptiveIntegration
from calculation_helpers.normalization_functions import NormalizationFunctions
from special_functions.integral_bessel_functions import IntegralBesselFunctions
from special_functions.matrices_calculation import MatricesCalculation
```

```
13
14  import logging
15  logger = logging.getLogger(__name__)
16  logger.setLevel(logging.INFO)
17
18  formatter = logging.Formatter('%(asctime)s:%(name)s:%(message)s')
19  log_file_name = 'logs/vector_potential.log'
20
21  # clear log
22  with open(log_file_name, 'w'):
23      pass
24
25  file_handler = logging.FileHandler(log_file_name)
26  file_handler.setLevel(logging.DEBUG)
27  file_handler.setFormatter(formatter)
28
29  stream_handler = logging.StreamHandler()
30  stream_handler.setFormatter(formatter)
31
32  logger.addHandler(file_handler)
33  logger.addHandler(stream_handler)
34
35
36  # VectorPotential class
37  # computes vector potential in point
38  # induced by 1 coil (eq. 59, p. 14)
39  # ORNL-5220
40  # UC32-Mathematics and Computers
41  # ANALYSIS AND COMPUTER PROGRAMS FOR
42  # EDDY CURRENT COILS CONCE TRIC WITH
43  # MULTIPLE CYLINDRICAL CONDUCTORS
44  # C.W. Nestor, Jr., C.V.Dodd, W.E. Deeds, July (1979))
45  class VectorPotential(
46          GaussIntegration,
47          AdaptiveIntegrationFunctions,
48          CyclesAdaptiveIntegration,
49          NormalizationFunctions,
50          IntegralBesselFunctions,
51          MatricesCalculation):
52
53      def __init__(self):
54          pass
55
56      def frequency_depend_part1(self, alpha, parameters):
57          calc_point = parameters['calc_point']
58          z = calc_point["z"]
59
60          coils_parameters = parameters['coils_parameters']
61          l1 = coils_parameters[0]['l1']
62          l2 = coils_parameters[0]['l2']
63
64          numerator = np.sin(alpha * (z - l1)) - np.sin(alpha * (z - l2))
65          denominator = alpha**3
66
67          result = numerator / denominator
68
69          return result
```

```
71      def frequency_depend_part2(self, alpha, parameters):
72          coils_parameters = parameters['coils_parameters']
73          r1 = coils_parameters[0]["r1"]
74          r2 = coils_parameters[0]["r2"]
75
76          inner_parameters = parameters['inner_parameters']
77          outer_parameters = parameters['outer_parameters']
78
79          calc_point = parameters['calc_point']
80          r = calc_point["r"]
81
82          u_col = np.zeros(4)
83          v_col = np.zeros(4)
84
85          ir1r2 = super().python_xiint(alpha, r1, r2)
86          kr1r2 = super().python_xkknt(alpha, r1, r2)
87          i1a0r = super().bess_i1(alpha * r)
88          k1a0r = super().bess_k1(alpha * r)
89
90          # set up U matrix for no external conductors
91          u_col[2] = 1.0
92
93          # calculate U matrix for external conductors (if any)
94          # (flag INN for UVMAT = 1)
95          if len(outer_parameters) != 0:
96              u_col = super().calc_uv_matrices(alpha, 1, u_col, parameters)
97
98          # set up V matrix for no internal conductors
99          v_col[0] = 1.0
100
101         # calculate V matrix for internal conductors (if any)
102         # (flag INN for UVMAT = 0)
103         if len(inner_parameters) != 0:
104             v_col = super().calc_uv_matrices(alpha, 0, v_col, parameters)
105
106         u12 = complex(u_col[0], u_col[1])
107         u22 = complex(u_col[2], u_col[3])
108         v11 = complex(v_col[0], v_col[1])
109         v21 = complex(v_col[2], v_col[3])
110
111         numerator_part1 = u12 * v11 * i1a0r * ir1r2
112         numerator_part2 = u22 * v21 * k1a0r * kr1r2
113         numerator_part3 = u12 * v21 * (i1a0r * kr1r2 + k1a0r * ir1r2)
114
115         numerator = (numerator_part1 + numerator_part2 + numerator_part3)
116         denominator = u22 * v11 - u12 * v21
117
118         result = numerator / denominator
119
120         return result
```

```
121
122     def frequency_depend_part(self, alpha, parameters):
123         part1 = self.frequency_depend_part1(alpha, parameters)
124         part2 = self.frequency_depend_part2(alpha, parameters)
125
126         result = part1 * part2
127
128         return result
129
130     def coil_in_air_part(self, alpha, parameters):
131         coils_parameters = parameters['coils_parameters']
132         r1 = coils_parameters[0]["r1"]
133         r2 = coils_parameters[0]["r2"]
134         l1 = coils_parameters[0]['l1']
135         l2 = coils_parameters[0]['l2']
136
137         calc_point = parameters['calc_point']
138         r = calc_point["r"]
139         z = calc_point["z"]
140
141         jr1r2 = super().python_xjjnt(alpha, r1, r2)
142         j1a0r = super().bess_j1(alpha * r)
143
144         e1 = np.exp(alpha * (z - l2))
145         e2 = np.exp(-alpha * (z - l1))
146
147         part1 = constants.pi / (2 * alpha**3)
148         part2 = jr1r2 * j1a0r
149         part3 = 2 - e1 - e2
150
151         result = part1 * part2 * part3
152
153         return result
154
155     def integrand(self, alpha, parameters):
156         coil_in_air_result = self.coil_in_air_part(alpha, parameters)
157         frequency_depend_result = self.frequency_depend_part(alpha, parameters)
158         complex_result = coil_in_air_result + frequency_depend_result
159
160         result = complex_result.real, complex_result.imag
161
162         return result
163
164     def constant_expresion(self, parameters):
165         coils_parameters = parameters['coils_parameters']
166         n = coils_parameters[0]['n']
167         electrical_values = parameters['electrical_values']
168         current = electrical_values['current']
169
170         numerator = n * current * constants.mu_0
171
172         result = numerator / constants.pi
173
174         return result
```

```
175
176     def calculate(self, norm_parameters, integ_top_range):
177         constant_expresion = self.constant_expresion(norm_parameters)
178         calc_func = self.integrand
179         integr_method = super().gauss2_integration
180         adapt2 = super().adapt2
181
182         result_adapt2 = super().adapt2_integ_cycle(
183             calc_func, integr_method, adapt2, norm_parameters, integ_top_range)
184
185         result = constant_expresion * result_adapt2
186
187         return result
```

Модель Dood

Розрахунок векторного потенціалу

`vector_potential_calculation/vector_potential_greens_function.py`

```
1   #!/usr/bin/env python
2   # coding: utf-8
3
4   # import numpy as np
5   # from scipy import constants
6
7   from calculation_helpers.gauss_integration import GaussIntegration
8   from calculation_helpers.adaptive_integration_functions import AdaptiveIntegrationFunctions
9   from calculation_helpers.cycles_adaptive_integration import CyclesAdaptiveIntegration
10  from calculation_helpers.normalization_functions import NormalizationFunctions
11  from special_functions.greens_function import GreensFunction
12
13  import logging
14  logger = logging.getLogger(__name__)
15  logger.setLevel(logging.INFO)
16
17  formatter = logging.Formatter('%(asctime)s:%(name)s:%(message)s')
18  log_file_name = 'logs/vector_potential_greens_function.log'
19
20  # clear log
21  with open(log_file_name, 'w'):
22      pass
23
24  file_handler = logging.FileHandler(log_file_name)
25  file_handler.setLevel(logging.DEBUG)
26  file_handler.setFormatter(formatter)
27
28  stream_handler = logging.StreamHandler()
29  stream_handler.setFormatter(formatter)
30
31  logger.addHandler(file_handler)
32  logger.addHandler(stream_handler)
33
34
35  class VectorPotentialGreensFunction(
36          GaussIntegration,
37          AdaptiveIntegrationFunctions,
38          NormalizationFunctions,
39          CyclesAdaptiveIntegration,
40          GreensFunction,):
```

```
41
42      def __init__(self):
43          pass
44
45      def calculate(self, norm_parameters, integ_top_range):
46          electrical_values = norm_parameters['electrical_values']
47          current = electrical_values['current']
48
49          constant_expresion = super().constant_expresion(norm_parameters)
50          calc_func = super().integrand
51          integr_method = super().gauss2_integration
52          adapt2 = super().adapt2
53
54          result_adapt2 = super().adapt2_integ_cycle(
55              calc_func, integr_method, adapt2, norm_parameters, integ_top_range)
56
57          result = current * constant_expresion * result_adapt2
58
59          return result
```

Модель Dood

Допоміжні розрахунки. Розрахунок вихідного сигналу

`calculation_helpers/voltage_from_vector_potential.py`

```
1
2   from scipy import constants
3
4
5   class VoltageFromVectorPotential():
6       # https://drive.google.com/file/d/1EMtBFP1dnea4vhW3W8N1mvmsRp-nTo54/view?usp=sharing
7       @staticmethod
8       def from_delta_func(vec_pot, frequency, radius):
9           omega = 2 * constants.pi * frequency
10          result = complex(0, 1) * omega * 2 * constants.pi * radius * vec_pot
11
12          return result
```

Модель Dood

Допоміжні розрахунки. Конвертор комплексних чисел

`calculation_helpers/convertations.py`

```
1   import cmath
2
3
4   class Convertations():
5       # get polar coordinates. Returns (length, angle).
6       @staticmethod
7       def complex_to_polar(complex_form):
8           return cmath.polar(complex_form)
9
10      # Returns a complex number
11      @staticmethod
12      def polar_to_complex(amplitude, phase):
13          return cmath.rect(amplitude, phase)
```

Модель Dood

Допоміжні розрахунки. Квадратури Гауса

`calculation_helpers/gauss_integration.py`

```python
#!/usr/bin/env python
# coding: utf-8


import numpy as np
import logging

logger = logging.getLogger(__name__)
logger.setLevel(logging.DEBUG)

formatter = logging.Formatter('%(asctime)s:%(name)s:%(message)s')

log_file_name = 'logs/gauss_integration.log'
# clear log
with open(log_file_name, 'w'):
    pass

file_handler = logging.FileHandler(log_file_name)
file_handler.setFormatter(formatter)

logger.addHandler(file_handler)


class GaussIntegration:
    def __init__(self):
        # todo rewrite
        pass

    # N-point gaussian integration from a to b (here a1, a2) of
    # integrand calculated  by subprogram F (here integrand).
    # N = 2, 4, 6, 8, or 10.
    def gauss_integration(self, a1, a2, integrand, coils_params, n):
        logger.debug('================================')
        logger.debug('gauss_integration input arguments a1 {}, a2 {}, coils_params {}, n {}'.format(
            a1, a2, coils_params, n))
#         print_coil_params = json.dumps(coils_params, indent=4)
#         print(f"call gauss_integration: \
#         a1 = {a1}, \
#         a2 = {a2}, \
#         integrand = {integrand}, \
#         params = {print_coil_params}, \
#         n = {n}")

#         abscissae = np.array([0.5773502691896257, 0.3399810435848563, 0.8611363115940526, 0.238619186
#                               0.6612093864662645, 0.9324695142031521, 0.1834346424956498, 0.525532409
#                               0.7966664774136267, 0.9602898564975363, 0.1488743389816312, 0.433395394
#                               0.6794095682990244, 0.8650633666889845, 0.9739065285171717])

#         weights =   np.array([1.0,               0.6521451548625461, 0.3478548451374538, 0.4679139345726
#                             0.3607615730481386, 0.1713244923791704, 0.3626837833783620, 0.3137066458778
#                             0.2223810344533745, 0.1012285362903763, 0.2955242247147529, 0.2692667193099
#                             0.2190863625159820, 0.1494513491505806, 0.0666713443086881])

        abscissae = np.array([0.577350269, 0.339981044, 0.861136312, 0.238619186,
                              0.661209386, 0.932469514, 0.183434642, 0.525532410,
                              0.796666477, 0.960289856, 0.148874339, 0.433395394,
                              0.679409568, 0.865063367, 0.973906529])

        weights = np.array([1.0,         0.652145155, 0.347854845, 0.467913935,
                            0.360761573, 0.171324492, 0.362683783, 0.313706646,
                            0.222381034, 0.101228536, 0.295524225, 0.269266719,
                            0.219086363, 0.149451349, 0.0666713443])
```

```
        i_t = n / 2
        i_1 = (i_t * (i_t - 1)) / 2
        i_2 = i_1 + i_t
        bpla = a2 + a1
        add = bpla * 0.5
        bmao2 = (a2 - a1) * 0.5
        sum_ = .0

        i_1 = int(i_1)
        i_2 = int(i_2)

        for i in range(i_1, i_2):
            alpha = add + bmao2 * abscissae[i]
            sum_ = (sum_ + weights[i] * (integrand(alpha, coils_params) +
                                         integrand(bpla - alpha, coils_params)))

        result = sum_ * bmao2
        logger.debug('gauss_integration output result = {}'.format(result))

        return result

    # N-point gaussian integration from a to b (here a1, a2) of
    # two integrands calculated by subprogram F (here integrand).
    # N = 2, 4, 6, 8, or 10.
    # result returned in ANS1 and ANS2 (here just list)
    def gauss2_integration(self, a1, a2, integrand, params, n):
        logger.debug('gauss_integration2 input arguments a1 {}, a2 {}, params {}, n {}'.format(
            a1, a2, params, n))
#         print_coil_params = json.dumps(coils_params, indent=4)
#         print(f"call gauss_integration: \
#         a1 = {a1}, \
#         a2 = {a2}, \
#         integrand = {integrand}, \
#         params = {print_coil_params}, \
#         n = {n}")

#         abscissae = np.array([0.5773502691896257, 0.3399810435848563, 0.8611363115940526, 0.23861918608
#                               0.6612093864662645, 0.9324695142031521, 0.1834346424956498, 0.52553240991
#                               0.7966664774136267, 0.9602898564975363, 0.1488743389816312, 0.43339539412
#                               0.6794095682990244, 0.8650633666889845, 0.9739065285171717])

#         weights =   np.array([1.0,                0.6521451548625461, 0.3478548451374538, 0.46791393457
#                               0.3607615730481386, 0.1713244923791704, 0.3626837833783620, 0.31370664587
#                               0.2223810344533745, 0.1012285362903763, 0.2955242247147529, 0.26926671930
#                               0.2190863625159820, 0.1494513491505806, 0.0666713443086881])


        abscissae = np.array([0.577350269, 0.339981044, 0.861136312, 0.238619186,
                              0.661209386, 0.932469514, 0.183434642, 0.525532410,
                              0.796666477, 0.960289856, 0.148874339, 0.433395394,
                              0.679409568, 0.865063367, 0.973906529])

        weights = np.array([1.0,         0.652145155, 0.347854845, 0.467913935,
                            0.360761573, 0.171324492, 0.362683783, 0.313706646,
                            0.222381034, 0.101228536, 0.295524225, 0.269266719,
                            0.219086363, 0.149451349, 0.0666713443])

        i_t = n / 2
        i_1 = (i_t * (i_t - 1)) / 2
        i_2 = i_1 + i_t
        bpla = a2 + a1
        add = bpla * 0.5
        bmao2 = (a2 - a1) * 0.5
        sum_1 = .0
        sum_2 = .0

        i_1 = int(i_1)
        i_2 = int(i_2)
```

```
131
132         for i in range(i_1, i_2):
133             temp_result_1 = 0
134             temp_result_2 = 0
135             #  ----1
136             alpha = add + bmao2 * abscissae[i]
137             temp_result_1 = integrand(alpha, params)
138             #  ----2
139             alpha = bpla - alpha
140             temp_result_2 = integrand(alpha, params)
141
142             sum_1 = (sum_1 + weights[i] *
143                      (temp_result_1[0] + temp_result_2[0]))
144             sum_2 = (sum_2 + weights[i] *
145                      (temp_result_1[1] + temp_result_2[1]))
146
147         result = [sum_1 * bmao2, sum_2 * bmao2]
148
149         logger.debug('gauss_integration2 output result = {}'.format(result))
150
151         return result
```

Модель Dood

Допоміжні розрахунки. Адаптивне інтегрування

`calculation_helpers/adaptive_integration_functions.py`

```
1   #!/usr/bin/env python
2   # coding: utf-8
3
4   # import json
5   import numpy as np
6
7
8   class AdaptiveIntegrationFunctions:
9       # ADAPT Subroutine
10      # adaptive integration of function from a1 to a2, with the function
11      # specified by the function subprogram FINT (here calc_func) and
12      # the method of numerical integration specified by the subroutine
13      # METHD(here integr_method). An example of the use of ADAPT is in subroutine COLAIR.
14      # returns: real int result, bool convergence
15      def adapt(self, a1, a2, integr_method, calc_func, thresh, tol, coils_params):
16          # required functions:
17          #   > integr_method - integration function (METHD):
18          #     > calc_func - integrand (FINT)
19
20          #          print_coil_params = json.dumps(coils_params, indent=4)
21          #         print(f"call adapt: \
22          #         a1 = {a1}, \
23          #         a2 = {a2}, \
24          #         integr_method = {integr_method}, \
25          #         calc_func = {calc_func}, \
26          #         thresh = {thresh}, \
27          #         tol = {tol} \
28          #         params = {print_coil_params}")
29
```

```
    # local variables
    a_left = np.zeros(10)
    a_right = np.zeros(10)
    n_p = np.array([2, 4, 6, 8, 10])
    smult = 1.953125 * 1.0e-3

    # convergence flag values???
    # 1H+  - 0 lines (type over)
    # 1H   - 1 line (single space)
    # variables to return
    # plus = '1H+'
    # blank = '1H'
    # char = blank
    convergence = True

    # local variables
    a_left[0] = a1
    a_right[0] = a2
    smin = smult * (a2 - a1)
    l_index = 0  # 1 is original because of fortran indexes
    t1 = 0  # buffer
    s1 = 0  # buffer
    q1 = 0  # buffer

    # 120 Program pointer of original program
    # converged for level L (here l_index). Add contribution to total,
    # see if more levels need to be done

    while True:
        # compute first estimate for current level
        # 100
        t1 = integr_method(a_left[l_index], a_right[l_index],
                           calc_func, coils_params, n_p[l_index])

        need_120 = False

        # computes additional estimates for this this level,
        # and test convergence on:
        #   - the absolute error, if the newest estimate is
        #     less than thresh, or
        #   - the relative error
        for k in range(1, len(n_p)):
            kp = n_p[k]
            s1 = integr_method(a_left[l_index], a_right[l_index],
                               calc_func, coils_params, kp)
            ds = abs(s1 - t1)

            if (abs(s1) > thresh):
                ds = ds / abs(s1)

            if (ds < tol):
                # goto 120
                need_120 = True
                break

            t1 = s1
```

```
86
87              if need_120:
88                  q1 = q1 + s1
89
90                  if (l_index == 0):
91                      break  # goto 140
92
93                  else:
94                      l_index = l_index - 1
95                      a_left[l_index] = a_right[l_index + 1]
96
97                      continue  # goto 100
98
99              # no covergence at this level. see if there more
100             # see if the step size is bigger than the minimum allowed
101             if ((l_index >= (len(a_left) - 1)) or ((a_right[l_index] - a_left[l_index]) < smin)):
102                 # goto 130
103                 # 130
104                 # set nonconvergence signal
105                 # char = plus
106                 convergence = False
107                 # goto 120
108
109                 q1 = q1 + s1
110
111                 if (l_index == 0):
112                     break  # goto 140
113
114                 else:
115                     l_index = l_index - 1
116                     a_left[l_index] = a_right[l_index + 1]
117
118                     continue  # goto 100
119
120             # increase level index, adjust limits
121             l_index += 1
122
123             a_left[l_index] = a_left[l_index - 1]
124             a_right[l_index] = (.5 * (a_left[l_index - 1] +
125                                       a_right[l_index - 1]))
126             # goto 100
127
128         # 140
129         # return {'result': q1, 'char': char}
130         return (q1, convergence)
131
132     # ADAPT2 Subroutine
133     # adaptive integration of two function from a1 to a2, with the method
134     # specified by the subprogram FINT (here  integr_method) and
135     # integrands calculated by the subroutine
136     # METHD(here calc_func). An example of the use of ADAPT2 is in subroutine QIDF, GAUSS2.
137     def adapt2(self, a1, a2, integr_method, calc_func, thresh, tol, params):
138         # required functions:
139         #   > integr_method - integration function (METHD):
140         #     > calc_func - integrand (FINT)
141
142         #         print_coil_params = json.dumps(params, indent=4)
143         #         print(f"call adapt: \
144         #         a1 = {a1}, \
145         #         a2 = {a2}, \
146         #         integr_method = {integr_method}, \
147         #         calc_func = {calc_func}, \
148         #         thresh = {thresh}, \
149         #         tol = {tol} \
150         #         params = {print_coil_params}")
```

```
152         # local variables
153         a_left = np.zeros(10)
154         a_right = np.zeros(10)
155         n_p = np.array([2, 4, 6, 8, 10])
156         klim = len(n_p) - 1
157         SMULT = 1.953125 * 1.0e-3
158 
159         # convergence flag values???
160         # 1H+  - 0 lines (type over)
161         # 1H   - 1 line (single space)
162         # variables to return
163         # blank = '1H'
164         # plus = '1H+'
165         # char = blank
166         convergence = True
167 
168         # local variables
169         a_left[0] = a1
170         a_right[0] = a2
171         smin = SMULT * (a2 - a1)
172         l_index = 0  # 1 is original because of fortran indexes
173         t = 0  # buffer
174         s = 0  # buffer
175         q1 = 0  # buffer
176         q2 = 0  # buffer
177 
178         # 120 Program pointer of original program
179         # converged for level L (here l_index). Add contribution to total,
180         # see if more levels need to be done
181 
182         while True:
183             # compute first estimate for current level
184             # 100
185             # print("_________\nssss",)
186             # print("a_left[l_index]", a_left[l_index])
187             # print("a_right[l_index]", a_right[l_index])
188             # t = integr_method(
189             #     a_left[l_index], a_right[l_index], calc_func, params, n_p[l_index])
190             t = integr_method(
191                 a_left[l_index], a_right[l_index], calc_func, params, n_p[0])
192             t1 = t[0]
193             t2 = t[1]
194 
195             if (t1 == float("nan") or t2 == float("nan")):
196                 raise Exception('result', 'is nan')
197             # print("t", t)
199             need_120 = False
200 
201             # computes additional estimates for this this level,
202             # and test convergence on:
203             #   - the absolute error, if the newest estimate is
204             #     less than thresh, or
205             #   - the relative error
206             for k in range(1, klim):
207                 kp = n_p[k]
208                 s = integr_method(
209                     a_left[l_index], a_right[l_index], calc_func, params, kp)
210                 s1 = s[0]
211                 s2 = s[1]
```

```
            ds1 = abs(s1 - t1)
            if (abs(s1) > thresh):
                ds1 = ds1 / abs(s1)

            ds2 = abs(s2 - t2)
            if (abs(s2) > thresh):
                ds2 = ds2 / abs(s2)

            ds = ds1 + ds2
            if (ds < tol):
                # goto 120
                need_120 = True
                break

            t1 = s1
            t2 = s2

        if need_120:
            need_120 = False
            q1 = q1 + s1
            q2 = q2 + s2

            if (l_index == 0):
                break  # goto 140

            else:
                l_index = l_index - 1
                a_left[l_index] = a_right[l_index + 1]

                continue  # goto 100

        # no covergence at this level. see if there more
        # see if the step size is bigger than the minimum allowed
        if ((l_index >= (len(a_left) - 1)) or ((a_right[l_index] - a_left[l_index]) < smin)):
            # goto 130
            # 130
            # set nonconvergence signal
            # char = plus
            convergence = False
            # goto 120

            q1 = q1 + s1
            q2 = q2 + s2

        if (l_index == 0):
            print("# goto 140")
            break  # goto 140

        else:
            l_index = l_index - 1
            a_left[l_index] = a_right[l_index + 1]

            continue  # goto 100

    # increase level index, adjust limits
    l_index = l_index + 1

    # print("!!!! l_index", l_index, q1, q2)
    a_left[l_index] = a_left[l_index - 1]
    a_right[l_index] = (.5 * (a_left[l_index - 1] +
                              a_right[l_index - 1]))
    # goto 100

# 140
# return {'result': [q1, q2], 'char': char}
return (q1, q2, convergence)
```

Модель Dood

Допоміжні розрахунки. Циклічне адаптивне інтегрування

`calculation_helpers/cycles_adaptive_integration.py`

```python

class CyclesAdaptiveIntegration():
    @staticmethod
    def adapt_integ_cycle(calc_func, integr_method, adapt, normalized_params, range_to):
        a1 = 0

        THRESH = 1.0e-3
        TOL = 5.0e-7

        result = 0
        for a2 in range(1, range_to + 1):
            temp = adapt(a1, a2, integr_method, calc_func,
                         THRESH, TOL, normalized_params)
            a1 = a2
            result += temp[0]
            convergence = temp[1]
            print(f"a {a2}, result {result}, convergence {convergence}")

        return result

    @staticmethod
    def adapt2_integ_cycle(calc_func, integr_method, adapt2, normalized_params, range_to):
        a1 = 0

        THRESH = 1.0e-3
        TOL = 5.0e-7

        result = [0, 0]
        for a2 in range(1, range_to + 1):
            temp = adapt2(a1, a2, integr_method, calc_func,
                          THRESH, TOL, normalized_params)
            a1 = a2
            result[0] += temp[0]
            result[1] += temp[1]
            convergence = temp[2]
            print(f"a {a2}, result {result}, convergence {convergence}")

        z1 = result[0]
        z2 = result[1]
        print(f"z1 {z1}, z2 {z2}")

        return complex(z1, z2)

```

Модель Dood

Допоміжні розрахунки. Нормалізація даних

`calculation_helpers/normalization_functions.py`

```python
#!/usr/bin/env python
# coding: utf-8
from scipy import constants


def inches_to_meters(inches):
    return inches * 0.0254


def meters_to_inches(meters):
    return meters / 0.0254


def convert_p_to_sigma(p):
    sigma = (10**8) / p

    return sigma


def convert_sigma_to_p(sigma):
    p = (10**8) / sigma

    return p


class NormalizationFunctions:
    # calculating RRB
    def _get_normalization_scale(self, input_coils_params):
        r1 = input_coils_params[0]['r1']
        r2 = input_coils_params[0]['r2']

        rrb = 2.0 / (r1 + r2)

        return rrb

    # NORMCO normalize coil dimensions by multiplying by the
    # reciprocal of the mean radius of coil No. 1
    def _coils_normalization(self, input_coils_params, coeff):
        for coil in input_coils_params:
            for param in coil:
                if param == 'n':
                    continue

                coil[param] = coil[param] * coeff

        return input_coils_params

    def mult_m_on_ratio(self, params, ratio):
        inner_params = params["inner_parameters"]
        for layer in inner_params:
            layer['m'] *= ratio
        outer_params = params["outer_parameters"]
        for layer in outer_params:
            layer['m'] *= ratio

        return params
```

```
58
59      def normalization(self, input_params):
60          normalized_params = input_params.copy()
61
62          input_coils_params = normalized_params["coils_parameters"]
63
64          r1 = input_coils_params[0]['r1']
65          r2 = input_coils_params[0]['r2']
66
67          rrb = self._get_normalization_scale(input_coils_params)
68          normalized_params["coils_parameters"] = self._coils_normalization(
69              input_coils_params, rrb)
70
71
72  #           THRESH = 1.0e-3
73  #           TOL = 5.0e-7
74  #           DEGPRA = 57.2957795
75          CONV = 0.0254
76          UNITS_CONV_COEFF = 509.39792
77          # OMEGA = 6283.18531
78
79          rbar = 0.5 * (r1 + r2)
80          rb2 = rbar**2
81          rbar = rbar * CONV
82
83          inner_params = normalized_params["inner_parameters"]
84
85          outer_params = normalized_params["outer_parameters"]
86
87          for layer in inner_params:
88              layer['r'] *= rrb
89              layer['m'] = (
90                  ((layer['mu_r'] * UNITS_CONV_COEFF) / layer['rho']) * rb2)
91
92          for layer in outer_params:
93              layer['r'] *= rrb
94              layer['m'] = (
95                  ((layer['mu_r'] * UNITS_CONV_COEFF) / layer['rho']) * rb2)
96
97          frequency = normalized_params['frequency'][0]
98
99          fold = 1.0
100         ratio = frequency / fold
101         normalized_params = self.mult_m_on_ratio(normalized_params, ratio)
102
103         return normalized_params
104
105     def calc_m_without_inches(self, input_params):
106         normalized_params = input_params.copy()
107
108          # input_coils_params = normalized_params["coils_parameters"]
109
110          # r1 = input_coils_params[0]['r1']
111          # r2 = input_coils_params[0]['r2']
112
113          # rrb = self._get_normalization_scale(input_coils_params)
114          # normalized_params["coils_parameters"] = self._coils_normalization(
115          # input_coils_params, rrb)
```

```
116
117         CONV = 0.0254
118         UNITS_CONV_COEFF = (509.39792 / (CONV * CONV)) / 1000
119
120         # rbar = 0.5 * (r1 + r2)
121         # rb2 = rbar**2
122         # rbar = rbar * CONV
123
124         inner_params = normalized_params["inner_parameters"]
125
126         outer_params = normalized_params["outer_parameters"]
127
128         for layer in inner_params:
129             # layer['r'] *= rrb
130             # layer['mu_r'] *= constants.mu_0
131             layer['m'] = (((layer['mu_r'] * UNITS_CONV_COEFF) /
132                            layer['rho']))
133
134         for layer in outer_params:
135             # layer['r'] *= rrb
136             # layer['mu_r'] *= constants.mu_0
137             layer['m'] = (((layer['mu_r'] * UNITS_CONV_COEFF) /
138                            layer['rho']))
139
140         frequency = normalized_params['frequency'][0]
141
142         fold = 1.0
143         ratio = frequency / fold
144         normalized_params = self.mult_m_on_ratio(normalized_params, ratio)
145
146         return normalized_params
147
148     # frequency in kHz
149     def normalization_1_coil(self, input_params):
150         normalized_params = input_params.copy()
151
152         input_coils_params = normalized_params["coils_parameters"]
153
154         r1 = input_coils_params[0]['r1']
155         r2 = input_coils_params[0]['r2']
156
157         rrb = self._get_normalization_scale(input_coils_params)
158         normalized_params["coils_parameters"] = self._coils_normalization(
159             input_coils_params, rrb)
160
161         CONV = 0.0254
162         UNITS_CONV_COEFF = 509.39792
163         # THE FACTOR 509.39792 is THE CONVERSION FACTOR NEEDED PtJ
164         # ACCOUNT FOR DISTANCES IN INCHES, FREQUENCIES IN KILOHERTZ
165         # AND PESISTPVXTIES IN HICRO-OHII CENTKHETERS,
166         # avg = 2 * constants.pi * constants.mu_0 * (0.0254 * 0.0254)
167         rbar = 0.5 * (r1 + r2)
168         rb2 = rbar**2
169         rbar = rbar * CONV
```

```
170
171         inner_params = normalized_params["inner_parameters"]
172
173         outer_params = normalized_params["outer_parameters"]
174
175         for layer in inner_params:
176             layer['r'] *= rrb
177             layer['m'] = (((layer['mu_r'] * UNITS_CONV_COEFF) /
178                            layer['rho']) * rb2)
179
180         for layer in outer_params:
181             layer['r'] *= rrb
182             layer['m'] = (((layer['mu_r'] * UNITS_CONV_COEFF) /
183                            layer['rho']) * rb2)
184
185         if normalized_params["calc_point"]:
186             calc_point = normalized_params["calc_point"]
187             for point_params in calc_point:
188                 calc_point[point_params] *= rrb
189         else:
190             print("normalized_params do not have calc_point")
191
192         frequency = normalized_params['frequency'][0]
193
194         fold = 1.0
195         ratio = frequency / fold
196         normalized_params = self.mult_m_on_ratio(normalized_params, ratio)
197
198         return normalized_params
199
200     def normalization_1_coil__sigma_mu_r(self, input_params):
201         normalized_params = input_params.copy()
202
203         input_coils_params = normalized_params["coils_parameters"]
204
205         r1 = input_coils_params[0]['r1']
206         r2 = input_coils_params[0]['r2']
207
208         rrb = self._get_normalization_scale(input_coils_params)
209         normalized_params["coils_parameters"] = self._coils_normalization(
210             input_coils_params, rrb)
211
212         rbar = 0.5 * (r1 + r2)
213         rb2 = rbar**2
214
215         inner_params = normalized_params["inner_parameters"]
216         outer_params = normalized_params["outer_parameters"]
217
218         frequency = normalized_params['frequency'][0]
```

```
220         def convert_layers(layers_params):
221             for index, layer in enumerate(layers_params):
222                 new_params = {}
223                 new_params['r'] = layer['r'] * rrb
224                 new_params['mu_r'] = layer['mu_r']
225                 new_params['m'] = (2 * constants.pi * constants.mu_0 *
226                                    frequency * layer['mu_r'] * layer['sigma'] * rb2)
227 
228                 inner_params[index] = new_params
229 
230         convert_layers(inner_params)
231         convert_layers(outer_params)
232 
233         if normalized_params["calc_point"]:
234             calc_point = normalized_params["calc_point"]
235             for point_params in calc_point:
236                 calc_point[point_params] *= rrb
237         else:
238             print("normalized_params do not have calc_point")
239 
240         return normalized_params
```

Модель Dood

Допоміжні розрахунки. Конвертор даних в систему CI

`calculation_helpers/convertation_si.py`

```
1   #!/usr/bin/env python
2   # coding: utf-8
3   from scipy import constants
4 
5 
6   class ConvertationSI:
7 
8       def normalization_1_coil(self, input_params):
9           INCHES_TO_METRE = 0.0254
10          CONV_COEFF = 509.39792 / (INCHES_TO_METRE**2)
11          print('CONV_COEFF', CONV_COEFF)
12 
13          normalized_params = input_params.copy()
14 
15          input_coils_params = normalized_params["coils_parameters"]
16 
17          for coil in input_coils_params:
18              for param in coil:
19                  if param == 'n':
20                      continue
21 
22                  coil[param] = coil[param] * INCHES_TO_METRE
23 
24          inner_params = normalized_params["inner_parameters"]
25 
26          outer_params = normalized_params["outer_parameters"]
27 
28          frequency = normalized_params['frequency'][0]
```

```
29
30          for layer in inner_params:
31              layer['r'] *= INCHES_TO_METRE
32              layer['m'] = (((frequency * layer['mu_r'] * CONV_COEFF) /
33                             layer['rho']))
34              layer['mu_r'] *= 1.2566370614359172e-05
35
36          for layer in outer_params:
37              layer['r'] *= INCHES_TO_METRE
38              layer['m'] = (((frequency * layer['mu_r'] * CONV_COEFF) /
39                             layer['rho']))
40              layer['mu_r'] *= 1.2566370614359172e-05
41
42          if normalized_params["calc_point"]:
43              calc_point = normalized_params["calc_point"]
44              for point_params in calc_point:
45                  calc_point[point_params] *= INCHES_TO_METRE
46          else:
47              print("normalized_params do not have calc_point")
48
49          return normalized_params
50
51      def calculate_m_only_from_si(self, input_params):
52          normalized_params = input_params.copy()
53
54          inner_params = normalized_params["inner_parameters"]
55          outer_params = normalized_params["outer_parameters"]
56
57          frequency = normalized_params['frequency'][0]
58
59          for layer in inner_params:
60              layer['mu_r'] *= constants.mu_0
61              layer['m'] = (2 * constants.pi * frequency *
62                            layer['mu_r'] * layer['sigma'])
63
64          for layer in outer_params:
65              layer['mu_r'] *= constants.mu_0
66              layer['m'] = (2 * constants.pi * frequency *
67                            layer['mu_r'] * layer['sigma'])
68
69          # for layer in inner_params:
70          #     mu = layer['mu_r'] * 1.2566370614359172e-05
71          #     layer['m'] = (2 * constants.pi * frequency *
72          #                   mu * layer['sigma'])
73
74          # for layer in outer_params:
75          #     mu = layer['mu_r'] * 1.2566370614359172e-05
76          #     layer['m'] = (2 * constants.pi * frequency *
77          #                   mu * layer['sigma'])
78
79          return normalized_params
```

Lookup Table. Створення модельних вхідних даних

7. Методи lookup table (`method_lut/`)
ОСНОВНИЙ ГЕНЕРАТОР

```python
#!/usr/bin/env python
# coding: utf-8

# import copy
import json
import numpy as np
import csv

from pathlib import Path, PurePath

# from decimal import Decimal

# from csv_helpers.csv_helpers import CSVHelpers
# from vector_potential_calculation.vector_potential_inside_coil_green_function import Ve

# from vector_potential_calculation.vector_potential_2_layers_coil_3reg_67 import VectorP
# from vector_potential_calculation.vector_potential_2_layers_coil_3reg_69 import VectorP

import input_data_generators.easing_functions.easing as easing
import input_data_generators.dist_test as data_generator
from input_data_generators.helpers import pack_layers_data

get_thin_input_parameters_n_layer = data_generator.get_thin_input_parameters_n_layer
get_raw_input_parameters = data_generator.get_raw_input_parameters

# aluminium
# sigma2 = 1.883e7
# sigma1 = 3.766e7

# carbon steel
# _conductivity
# __6.99e6/2 ???
# __6.99e6
# _permeability
# __60
# __200

# Linear
# CircularEaseIn
# SineEaseIn
# QuadEaseIn
# TanHiperbolic
# Gausian


# str_sigma2_dist_list = [
#     # 'None',
#     # 'QuadEaseIn',
#     # 'Gausian',
#     # 'ExponentialEaseIn',
#     'TanHiperbolic',
# ]

```

```
# sigma2_dist_list = [
#     # None,
#     # easing.QuadEaseIn,
#     # easing.Gausian,
#     # easing.ExponentialEaseIn,
#     easing.TanHiperbolic,
# ]


# ------------------------------------------------------------------------
# ------------------------------------------------------------------------
# ------------------------------------------------------------------------


def readCSVFile(path_to_file, file_name):
    data_holder = []

    with open(path_to_file + file_name, mode='r') as csv_file:
        line_count = 0
        csv_reader = csv.reader(csv_file, delimiter=',')

        for row in csv_reader:
            line_count += 1
            if line_count == 1:
                print('csv file keys: ', row)
            else:
                data_holder.append(np.array(row))

        print(f'Processed {line_count} lines.')

    return np.array(data_holder)


def create_datasets_from_sequences(path_to_sequences_file, sequences_file_name,
    str_file_data = readCSVFile(path_to_sequences_file, sequences_file_name)
    file_data = str_file_data.astype(np.float64)
    file_data_columns = file_data.transpose()
    sigma_r2_list = file_data_columns[0]
    mu_r2_list = file_data_columns[1]

    list_len = len(mu_r2_list)

    input_data_list = []

    for i in range(0, list_len):
        layersQuantity = 50
        mu_r1 = 1
        mu_r2 = mu_r2_list[i]
        sigma1 = 6.99e6
        sigma2 = sigma_r2_list[i]
        mu_r2_dist = easing.TanHiperbolic
        sigma2_dist = easing.TanHiperbolic

        frequency = 2500
```

```
108
109             raw_params = get_raw_input_parameters(
110                 sigma1=sigma1,
111                 sigma2=sigma2,
112                 sigma2_dist=sigma2_dist,
113                 aprox_layers_quantity=layersQuantity,
114                 mu_r1=mu_r1,
115                 mu_r2=mu_r2,
116                 mu_r2_dist=mu_r2_dist,
117                 frequency=frequency,
118                 r1=0.009,
119                 r2=0.010,
121                 calc_r=0.0135,
122                 calc_z=0.05,
123
124                 coil_r1=0.016,
125                 coil_r2=0.021,
126                 coil_l1=0.0475,
127                 coil_l2=0.0525,)
128
129             input_data_list.append(raw_params)
130
131         pack_layers_data(input_data_list)
132         # test_data = pack_layers_data(input_data_list)
133         # print(test_data)
134         # test_data = unpack_layers_data(test_data)
135         # print(test_data)
137         new_path = PurePath('./').joinpath(path_to_dataset_file, dataset_file_name)
138         Path(path_to_dataset_file).mkdir(parents=True, exist_ok=True)
139
140         with open(new_path, 'w') as fp:
141             json.dump(input_data_list, fp)
142
143
144     # ----------------------------------------------------------------------------
145     # ----------------------------------------------------------------------------
146
147     def main():
148         return
149
150
151     if __name__ == "__main__":
152
153         main_directory = 'method_lut/datasets_output/'
154
155         path_to_sequences_file = main_directory + 'sequences_scaled_r/'
156         path_to_dataset_file = main_directory + 'model_input_data/'
157
158         # quantity = 5 * 1000
159         # dim_n_1 = 3
160         # dim_n_2 = 4
161         # file_name_base = f"lpt_2x{quantity}_dim({dim_n_1},{dim_n_2})"
162         # quantity = 8
163         # quantity = 5005
164         quantity = 10000
165         variance = 0.3
166         variation_coeff_x = variance
167         variation_coeff_y = variance
```

```
    file_name_base = f"R_sequence__d_2__n_{quantity}"
    # sequences_file_name = f'scaled_variance{variation_coeff_x}_{variation_coeff_y}__{file_name_
    sequences_file_name = f'scaled__{file_name_base}__sigma_{variation_coeff_x}_mu_r_{variation_
    dataset_file_name = f'input_data_q{quantity}_var{variation_coeff_x}_{variation_coeff_y}.json

    create_datasets_from_sequences(
        path_to_sequences_file, sequences_file_name, path_to_dataset_file, dataset_file_name)
    # ----------------------------------------------------------------------------------------
    # ----------------------------------------------------------------------------------------

    # quantity = 10 * 1000
    # dim_n_1 = 0
    # dim_n_2 = 2
    # file_name_base = f"lpt_2x{quantity}_dim({dim_n_1},{dim_n_2})"
    # variance = 0.3
    # variation_coeff_x = variance
    # variation_coeff_y = variance
    # sequences_file_name = f'scaled_variance{variation_coeff_x}_{variation_coeff_y}__{file_name_
    # dataset_file_name = f'input_data_q{quantity}_var{variation_coeff_x}_{variation_coeff_y}.jso

    # create_datasets_from_sequences(
    #     path_to_sequences_file, sequences_file_name, path_to_dataset_file, dataset_file_name)
    # # ----------------------------------------------------------------------------------------
    # # ----------------------------------------------------------------------------------------
```

Lookup Table. Реалізація методу LUT

****ОСНОВНИЙ РОЗРАХУНКОВИЙ СКРИПТ****

```
#!/usr/bin/env python
# coding: utf-8

import copy
import json
# import numpy as np
# import csv

from pathlib import Path, PurePath
from vector_potential_calculation.vector_potential_inside_coil_green_function import Vector


from calculation_helpers.voltage_from_vector_potential import VoltageFromVectorPotential
from input_data_generators.helpers import unpack_layers_data
from calculation_helpers.convertations import Convertations

from multiprocessing import Process


def calc_and_save(input_file_name, result_file_name):
    # packed_data_raw = None
    # with open(input_file_name) as json_file:
    #     packed_data_raw = json.load(json_file)

    # # raw_data_list = unpack_layers_data(copy.deepcopy(packed_data))
    # packed_data = copy.deepcopy(packed_data_raw)

    # for data_item in packed_data:
    #     vec_pot_dict = data_item['result_vector_potential']
    #     vec_pot_complex = complex(vec_pot_dict['real'], vec_pot_dict['imag'])
    #     frequency = data_item['frequency'][0]
    #     radius = data_item['calc_point']['r']
```

```
#     # print('raw_data_list', data_item)
#     # print('result_vector_potential', vec_pot_dict)
#     # print('result_voltage', data_item['result_voltage'])
#     # print('complex_to_polar', Convertations.complex_to_polar(vec_pot_complex))

#     voltage_complex = VoltageFromVectorPotential.from_delta_func(
#         vec_pot_complex, frequency, radius)

#     voltage_polar = Convertations.complex_to_polar(voltage_complex)
#     amplitude, phase = voltage_polar

#     # print('voltage_complex', voltage_complex)
#     # print('voltage_polar', voltage_polar)
#     # print('amplitude, phase', amplitude, phase)

#     # mutate data !!!!!!!
#     data_item["result_voltage"] = {
#         "real": voltage_complex.real,
#         "imag": voltage_complex.imag,
#     }
#     data_item["result_voltage_polar"] = {
#         "amplitude": amplitude,
#         "phase": phase,
#     }

# with open(result_file_name, 'w') as fp:
#     json.dump(packed_data, fp)

# ------------------------------------------------------------------
# ------------------------------------------------------------------
# ------------------------------------------------------------------
packed_data = None
with open(input_file_name) as json_file:
    packed_data = json.load(json_file)

raw_data_list = unpack_layers_data(copy.deepcopy(packed_data))

# print('raw_data_list', raw_data_list[0])

calcObject = VectorPotentialInsideCoilGreenFunction()

normalized_data_list = []

# print(raw_data_list[0])
for data in raw_data_list:
    normalized_data = calcObject.normalization_1_coil__sigma_mu_r(data)
    normalized_data_list.append(normalized_data)
results_list = []
integration_range = 50
for input_data in normalized_data_list:
    # print('adfadsfasfdasf')
    # print(input_data)
    # break
    result = calcObject.calculate(input_data, integration_range)

    # print("adsfasdf result", result)

    # result2 = calcObject.calculate(input_data, integration_range * 2)
    # print(result, result2)
    results_list.append(result)
    # break
```

```
    # print(normalized_data_list[1])

    # print("normalized_data_list", normalized_data_list)
    for index, result in enumerate(results_list):
        amplitude, phase = Convertations.complex_to_polar(result)
        # print("-----\nadsfasdf result", result)
        # print("complex_to_polar", amplitude, phase)
        # print("polar_to_complex", Convertations.polar_to_complex(amplitude, phase))

        # packed_data[index]["result_voltage"] = {
        #     "amplitude": amplitude,
        #     "phase": phase,
        # }

        data_item = packed_data[index]
        vec_pot_complex = result
        frequency = data_item['frequency'][0]
        radius = data_item['calc_point']['r']

        vec_pot_complex = result
        voltage_complex = VoltageFromVectorPotential.from_delta_func(
            vec_pot_complex, frequency, radius)

        voltage_polar = Convertations.complex_to_polar(voltage_complex)
        amplitude, phase = voltage_polar

        packed_data[index]["result_voltage"] = {
            "real": voltage_complex.real,
            "imag": voltage_complex.imag,
        }
        packed_data[index]["result_voltage_polar"] = {
            "amplitude": amplitude,
            "phase": phase,
        }

        packed_data[index]["result_vector_potential"] = {
            "real": result.real,
            "imag": result.imag,
        }

    print(packed_data[0])

    with open(result_file_name, 'w') as fp:
        json.dump(packed_data, fp)
def main():
    return


if __name__ == "__main__":
    # quantity = 5 * 1000
    # quantity = 5000
    # quantity = 50
    # quantity = 5005
    quantity = 10000
    variance = 0.3
    variation_coeff_x = variance
    variation_coeff_y = variance

    main_directory = 'method_lut/datasets_output/'
```

```
    path_to_input_dataset_file = PurePath(
        main_directory).joinpath('model_input_data/')
    path_to_output_dataset_file = PurePath(
        main_directory).joinpath('calculated/')

    Path(path_to_output_dataset_file).mkdir(parents=True, exist_ok=True)

    # dataset_file_name = f'input_data_q{quantity}_var{variation_coeff_x}_{variation_coeff_y}
    dataset_file_name = f'input_data_q{quantity}_var{variation_coeff_x}_{variation_coeff_y}.
    result_dataset_file_name = f'full_data_q{quantity}_var{variation_coeff_x}_{variation_coe

    input_file_name = PurePath(
        path_to_input_dataset_file).joinpath(dataset_file_name)

    result_file_name = PurePath(
        path_to_output_dataset_file).joinpath(result_dataset_file_name)
    # calc_and_save(input_file_name, result_file_name)
    process1 = Process(target=calc_and_save, args=(
        input_file_name, result_file_name))

    # ================================================================
    # quantity = 10 * 1000
    # variance = 0.3
    # variation_coeff_x = variance
    # variation_coeff_y = variance
    # # dataset_file_name = f'input_data_q{quantity}_var{variation_coeff_x}_{variation_coeff_
    # dataset_file_name = f'full_data_q{quantity}_var{variation_coeff_x}_{variation_coeff_y}
    # result_dataset_file_name = f'corrected_full_data_q{quantity}_var{variation_coeff_x}_{va

    # input_file_name = PurePath(
    #     path_to_input_dataset_file).joinpath(dataset_file_name)

    # result_file_name = PurePath(
    #     path_to_output_dataset_file).joinpath(result_dataset_file_name)
    # # calc_and_save(input_file_name, result_file_name)
    # process2 = Process(target=calc_and_save, args=(
    #     input_file_name, result_file_name))

    # construct a different process for each function
    processes = [process1,
                 # process2,
                 # process3,
                 # process4,
                 ]

    # kick them off
    for process in processes:
        process.start()

    # now wait for them to finish
    for process in processes:
        process.join()
```

ОСНОВНІ ТЕРМІНИ ТА ВИЗНАЧЕННЯ ВИХРОСТРУМОВОГО КОНТРОЛЮ

Автокодувальник (autoencoder AE) це загальна назва виду нейронних мереж, що по своїй структурі призначена без учителя навчатися та виділяти особливості з певного набору даних**.**

Вихрострумовий контроль (eddy current testing) – неруйнівний метод, при якому використовуються електромагнітні ефекти індукованого струму в контрольованому виробі.

Вихрові струми (eddy currents) – електричний струм, індукований в провідному матеріалі змінним магнітним полем.

Вихрострумова дефектоскопія (eddy current flaw detection) – методи та засоби на основі вихрових струмів, що використовують для виявлення дефектів типу порушення суцільності, які виходять на поверхню ОК або розташовані на невеликій глибині під поверхнею, наприклад, різні тріщини, розшарування, раковини, неметалеві включення тощо.

Вихрострувова структуроскопія (eddy current structuroscopy) – структурний аналіз металів і сплавів, який використовують для контролю варіації хімічного складу сплавів, сортування і розбраковки металевих матеріалів за їх марками, визначати механічні напругу в них, контролю якості термообробки, стану поверхневих шарів деталей після механічної обробки та інше.

Вимірювальна котушка (pick-up coil) – котушка для вимірювання напруженості магнітного поля, яка приймає результуюче магнітне поле.

Динамічні струми (dynamic currents) – додаткові вихрові струми, які наводяться переміщенням вихрострумового перетворювача і об’єкту контролю відносно один одного.

Зона контролю (tested zone) – зона контролю виробу в напрямку сканування об’єкту.

Ефективна глибина проникнення (effective depth of penetration) – глибина матеріалу, за якою електромагнітне поле вихрових струмів неможливо використовувати при контролі за допомогою вибраної схеми.

Ефективна магнітна проникність (effective permeability) – комплексний коефіцієнт, введений для оцінювання ослаблення напруженості магнітного поля в циліндричному об’єкті через вихрові струми. Використовують в розрахунку вихідної напруги

вторинної обмотки коаксіального ВСП.

Ефект швидкості (velocity effect) – ефект, викликаний динамічними струмами.

Ефект матеріалу (material effect) – вплив на вихрострумовий сигнал змін електромагнітних параметрів контрольованої деталі в зоні взаємодії ВСП.

Котушка збудження **(**excitation coil) – котушка, що підключена до джерела живлення і створює в контрольованій деталі електромагнітне поле збудження.

Обмотка котушки (coil winding) – один або більше витків провідника.

Розподіл вихрових струмів (eddy current distribution) – векторне поле густини вихрових струмів.

Струм збудження (excitation current) – значення струму у котушці збудження.

Скін-ефект (skin effect) – концентрація електромагнітних полів і вихрових струмів в околі поверхні виробу, що контролюється, яка є результатом самоіндукції і залежить від частоти, електропровідності і проникності.

Стандартна глибина проникнення δ (standard depth of penetration) – глибина, на якій напруженість електромагнітного поля або індукована густина вихрових струмів зменшується на 37 % від їх значення на поверхні.

Технологія вихрострумового контролю (eddy current technique) – сукупність технічних та програмних засобів, методів, алгоритмів, методик та способів реалізації вихрострумового контролю.

Частота збудження (excitation frequency) – номінальна частота струму збудження.

Наукове видання

Гальченко Володимир Якович
Сторчак Анатолій Вячеславович
Трембовецька Руслана Володимирівна
Тичков Володимир Володимирови

ВИХРОСТРУМОВІ ЕКСПРЕС-ВИМІРЮВАННЯ ПРОФІЛІВ ЕЛЕКТРОФІЗИЧНИХ ВЛАСТИВОСТЕЙ ЦИЛІНДРИЧНИХ ОБ'ЄКТІВ

Монографія
В авторській редакції

Видання українською мовою

Підписано до друку 03.04.2026
Формат 60х84/16. Папір офсетний
Друк. Арк. – 4,6
Друк цифровий
Наклад 100 прим.

Видавець: СГ НТМ «Новий курс»
Наукова установа
пр. Перемоги, 77, оф. 179
м. Харків, 61174, Україна
Свідоцтво про внесення суб’єкта видавничої справи
до державного реєстру видавцівб виготовлювачів і розповсюджувачів
видавничої продукції
ДК № 8013 від 22.11.2023