Міністерство освіти і науки, молоді та спорту України
Міжнародний економіко-гуманітарний університет
імені академіка Степана Дем'янчука
Факультет кібернетики
Кафедра математичного моделювання

Костючок Сергій Васильович

Побудова моделі вивчення базової дисципліни в середовищі C++ і її використання в курсі «Педагогіка вищої школи»

Науковий керівник:
Р.М.Літнарович, доцент,
кандидат технічних наук

Рівне – 2012

УДК 004.42 **Костючок С.В** Побудова моделі вивчення базової дисципліни в середовищі С++ і її використання в курсі «Педагогіка вищої школи». Монографія. Науковий керівник Р.М.Літнарович. МЕГУ, Рівне, 2011.- 125 с.

Робота виконана на кафедрі математичного моделювання Міжнародного економіко-гуманітарного університету імені академіка Степана Дем'янчука

Рецензенти:В.Г.Бурачек, доктор технічних наук, професор
В.О.Боровий, доктор технічних наук, професор
…….Є.С.Парняков, доктор технічних наук, професор
Відповідальний за випуск: Й.В.Джунь, доктор фіз.-мат. наук, професор

Практична значимість і реалізація роботи полягає в розробці програмного продукту, який є перевірений , протестований та впроваджений в навчальний процес МЕГУ в курсі «Педагогіка вищої школи». Розроблена високорівнева модель алгоритму відповідає усім вимогам, які були поставлені до даного програмного забезпечення.

Ключові слова: математична модель, базова дисципліна, метод Монте-Карло, програмний продукт.

Практическая значимость и реализация работы заключается в разработке программного продукта, который проверен, протестированный и внедрен в учебный процесс МЕГУ в курсе "Педагогика высшей школы". Разработанная высокоуровневая модель алгоритма отвечает всем требованиям, которые были поставлены к данному программному обеспечению.

Ключевые слова: математическая модель, базовая дисциплина, метод Монте-Карло, программный продукт.

Practical meaningfulness and realization of work consist in software product that is tested, tested and inculcated in the educational process of IEGU in a course "Pedagogics of higher school" development. The worked out high-level model of algorithm answers all requirements that were put to this software.

Keywords: mathematical model, base discipline, method of Monte Carlo, software product.

Зміст

Вступ

Математичне моделювання зараз переживає час стрімкого злету. Цим воно, зокрема, завдячує бурхливому розвитку обчислювальної техніки, завдяки високим характеристикам якої стала можливою програмна реалізація ряду складних моделей. Крім того, інтерес до математичного моделювання зростає завдяки широкому поширенню обчислювального експерименту, результати якого прирівнюються до результатів реального, а вартість та часові затрати, як правило, значно нижчі. Саме тому галузі моделювання таких процесів є **актуальними і сучасними**.

Неможливо уявити собі сучасну науку без широко застосування математичного моделювання, суть якого полягає в заміні досліджуваного об'єкта його математичною моделлю – і в подальшому вивченні за допомогою відповідних обчислювально-логічних алгоритмів на ЕОМ. Робота не з об'єктом (явищем, процесом), а з його моделлю дає можливість без істотних затрат і відносно швидко дослідити його властивості і поведінку у різних ситуаціях. Обчислювальні (комп'ютерні, стимуляційні, імітаційні) експерименти з моделями об'єктів дозволяють детально вивчати об'єкти з достатньою повнотою, недоступної для чисто теоретичних досліджень. Традиційно математичні моделі будувалися в галузі фізики, і на сьогоднішній день в ряді випадків такі моделі є досить якісними та вичерпними. Педагогічна сфера є дещо новою для застосування математичного моделювання. Такий «запізнілий» інтерес до неї пов'язаний із тим, що поняття якими оперують відповідні науки досить важко формалізувати та кількісно виміряти. Разом з тим , завдання практики вимагають розробки ефективних математичних моделей, завдяки яким можна було б здійснювати довгострокові прогнози чи навіть керувати педагогічними процесами. При побудові моделей ті або інші вірогідні ситуації або гіпотези фахівців стають більш осяжними, можуть уточнюватися, а тому сприяють кращому розумінню ситуації. Моделювання прискорює підготовку рішень і страхує від грубих помилок в діяльності.

Одним з імітаційних методів є метод Монте-Карло. Цей метод дозволяє моделювати будь-який процес, на протікання якого впливають випадкові чинники. Ідея цього методу полягає в наступному: якщо нам треба приблизно вирахувати деяку величину **А**, то потрібно придумати таку випадкову величину **В**, що отримавши і обробивши множину її значень можна було отримати шукану величину. Для багатьох математичних завдань, не пов'язаних з якими-небудь випадковостями, можна штучно придумати імовірнісну модель, яка в деяких випадках є вигіднішою. Оскільки метод Монте-Карло вимагає проведення великого числа випробувань, його часто називають методом статистичних_випробувань. Метод Монте-Карло – могутній і універсальний інструмент для розв'язку задач в багатьох областях знань.

Проблема дослідження. створення математичної моделі якості засвоєння базової дисципліни і її дослідження методом статистичних випробувань Монте-Карло.

Мета дослідження. генерувати псевдовипадкові похибки, нормувати їх, побудувати спотворену модель, зрівноважити її і дослідити точність зрівноважених елементів.

Актуальність дослідження. В необхідності оптимізувати навчальний процес вузу з метою побудови математичної моделі якості засвоєння базової дисципліни.

Наукова новизна дослідження. В розробці програмного продукту на мові програмування С++ , який забезпечив би побудову математичної моделі якості засвоєння базової дисципліни з метою вдосконалення навчального процесу вузу.

Метод вирішення проблеми. Застосування методу статистичних випробувань Монте-Карло і методу множинної регресії способу найменших квадратів в основі розробки програми на С++.

Практична значимість і реалізація роботи полягає в розробці програмного продукту, який є перевірений , протестований та впроваджений в навчальний процес МЕГУ в курсі «Педагогіка вищої школи». Розроблена високорівнева

модель алгоритму відповідає усім вимогам, які були поставлені до даного програмного забезпечення.

Апробація роботи. Окремі розділи дисертації були доложені і отримали одобрення на наукових конференціях студентів і аспірантів у 2010 і 2011 роках, а також на науковому семінарі кафедри математичного моделювання.

Публікації. Основні положення дисертації опубліковані в монографії автора:Костючок С.В. Побудова моделі вивчення базової дисципліни в середовищі С++ і її використання в курсі «Педагогіка вищої школи». Науковий керівник Р.М.Літнарович. МЕГУ, Рівне, 2012.- 90 с.

Основні положення дисертації, що виносяться на захист:

- повний опис практичного застосування регресійного аналізу при моделюванні якості засвоєння базової дисципліни з метою вдосконалення педагогічного процесу вузу;
- розробка математичного апарату створення вісьми факторної математичної моделі;
- розробка математичного апарату отримання середньої квадратичної похибки зрівноваженої функції через допоміжну матрицю **Q';**
- розробка контрольної формули розрахунку середньої квадратичної похибки зрівноваженої функції Y' через середні квадратичні похибки знайдених коефіцієнтів новоствореної математичної моделі $m_{a0,}$ m_{a1}, m_{a2}, m_{a3}, m_{a4}, m_{a5}, m_{a6}, m_{a7}, $m_{a8;}$
- розробка програмного продукту;
- розробка зручного графічного інтерфейсу системи, який забезпечує комфортну роботу ;
- опис технологічної бази розробки і тестування алгоритму;
- середовище розробки.

Структура і об'єм роботи. Магістерська дисертація складається із вступу, трьох розділів, розбитих на підрозділи, висновків і списку використаних джерел із 27 найменувань, із них

4 на іноземній мові, та додатків. Обсяг дисертації 90 сторінок, 14 таблиць, 9 рисунків.

Розділ 1.Математичне моделювання

1.1 Поняття моделі та моделювання

Модель – речова, знакова або уявна (мислена) система, що відтворює, імітує, відображає принципи внутрішньої організації або функціонування, певні властивості, ознаки чи характеристики об’єкта дослідження (оригіналу).

Значення терміна “модель” багатопланове:

– зразок, взірцевий примірник чогось;
– тип, марка конструкції;
– те, що є матеріалом, натурою для відтворення;
– зразок, з якого знімається форма для відливання в іншому матеріалі;
– комп’ютерна модель,
– розрахункова модель,
– теоретична модель (процесу, конструкції тощо).

Розрізняють фізичні, математичні та ін. моделі.

Наприклад, модель — опис об'єкта (предмета, явища або процесу) на якій-небудь формалізованій мові, складений з метою вивчення його властивостей. Такий опис особливо корисний у випадках, коли дослідження самого об'єкта ускладнене або фізично неможливе.

Найчастіше в ролі моделі виступає інший матеріальний або уявний об'єкт, що замінює в процесі дослідження об'єкт – оригінал. Таким чином, модель виступає як своєрідний інструмент

для пізнання, який дослідник ставить між собою і об'єктом, і за допомогою якого вивчає об'єкт, що його цікавить.

Математична модель — це система математичних співвідношень, які описують досліджуваний процес або явище. Математична модель має важливе значення для таких наук, як: економіка, екологія, соціологія, фізика, хімія, механіка, інформатика, біологія та ін.

При одержанні математичної моделі використовують загальні закони природознавства, спеціальні закони конкретних наук, результати пасивних та активних експериментів, імітаційне моделювання за допомогою ЕОМ.

Математичні моделі дозволяють передбачити хід процесу, розрахувати цільову функцію (вихідні параметри процесу), керувати процесом, проектувати системи з бажаними характеристиками. Для їх створення можна використовувати будь які математичні засоби — мову диференційних або інтегральних рівнянь, теорії множин, абстрактної алгебри, математичну логіку, теорії ймовірностей, графи та інші.

Якщо відношення задаються аналітично, то їх можна розв'язати в замкнутому вигляді (явно) відносно шуканих змінних як функції від параметрів моделі, або в частково замкнутому вигляді (неявно), коли шукані змінні залежать від одного або багатьох параметрів моделі. До моделей цього класу належать диференційні, інтегральні, різницеві рівняння, ймовірнісні моделі, моделі математичного програмування та інші. Якщо не можна здобути точний розв'язок математичної моделі, використовуються чисельні (обчислювальні) методи або інші види моделювання.

У залежності від того, якими є параметри системи та зовнішні збурення математичні моделі можуть бути детермінованими та стохастичними. Детерміновані пов’язані з дослідженням моделей з чітко заданими параметрами (завдання початкових і граничних умов значень функцій на вході і т. д.). Стохастичні – пов’язані з випадковими значеннями. Останні мають особливо важливе значення при дослідженні і проектуванні

великих систем зі складними зв’язками і властивостями, які важко врахувати.

Для розробки математичних моделей широко використовується диференційне числення, теорія множин, матриці і графи, а також планування експерименту. Відповідно розрізняють теоретико-множинні, матричні, топологічні та поліномні математичні моделі.

Приклади математичних моделей:

1. Модель Мальтуса – закон про пропорційну залежність між швидкістю росту і розміром популяції.

2. Система хижак-жертва (Вольтера-Лотки) – показує залежність між чисельністю хижаків та жертв.

3. Модель оптимальної поведінки покупця – виражає вибір покупця між множиною продуктів при обмеженому бюджеті.

Процес побудови, вивчення й використання математичних моделей називається математичним моделюванням. Це найзагальніший та найбільш використовуваний в науці, зокрема, в кібернетиці, метод досліджень. Це метод дослідження процесів або явищ шляхом створення їхніх математичних моделей і дослідження цих моделей. Він тісно поєднаний з такими категоріями, як абстракція, аналогія, гіпотеза тощо.

В основу методу покладено ідентичність форми рівнянь і однозначність співвідношень між змінними в рівняннях оригіналу і моделі, тобто, їх аналогії. Математичні моделі досліджуються, як правило, із допомогою аналогових обчислювальних машин, цифрових обчислювальних машин, комп'ютерів.

Математичне моделювання тією чи іншою мірою застосовують всі природничі і суспільні науки, що використовують математичний апарат для одержання спрощеного опису реальності за допомогою математичних понять. Воно дозволяє замінити реальний об’єкт його моделлю і потім вивчати останню.

В залежності від характеру процесів, що вивчаються, в системі всі види моделювання можуть бути розділені на аналітичні та комп'ютерні (Рис.1)

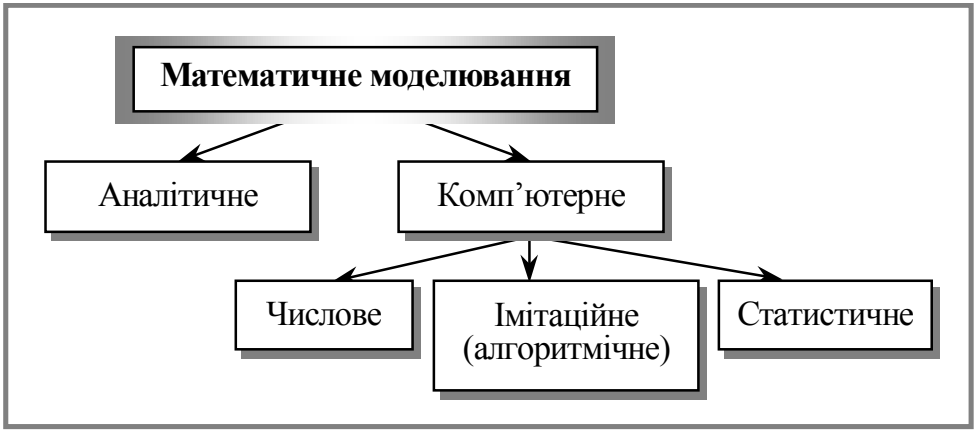


Рис. 0.1. Аналітичне та комп'ютерне моделювання

Для аналітичного моделювання характерним є те, що процеси функціонування елементів системи записують у вигляді деяких математичних співвідношень (алгебраїчних, інтегро-диференційних, кінцево-різницевих тощо) чи логічних умов.

Аналітична модель може досліджуватися такими методами:

1. Аналітичним, коли прагнуть у загальному вигляді отримати деякі залежності для шуканих характеристик;
2. Числовим;
3. Якісним, коли, не маючи явного розв'язку, все ж знаходять деякі властивості рішень.

Комп'ютерне моделювання характеризується тим, що математична модель системи (використовуючи основні співвідношення аналітичного моделювання) подається у вигляді деякого алгоритму та програми, придатної для її реалізації на комп'ютері, що дозволяє проводити з нею обчислювальні експерименти. Залежно від математичного інструментарію, що використовується в побудові моделі, та способу організації обчислювальних експериментів можна виокремити три

взаємопов’язані види моделювання: числове, алгоритмічне (імітаційне) та статистичне.

За числового моделювання для побудови комп’ютерної моделі використовуються методи обчислювальної математики.

Алгоритмічне (імітаційне) моделювання (може бути як детермінованим, так і стохастичним) — це вид комп’ютерного моделювання, для якого характерним є відтворення на комп’ютері (імітація) процесу функціонування досліджуваної складної системи. Тут імітуються (з використанням аналітичних залежностей і моделей) елементарні явища, що становлять процес, зі збереженням їхньої логічної та семантичної структури, послідовності плину в часі, що дозволяє отримати нову інформацію про стан системи *S* у задані моменти часу.

Статистичне моделювання — це вид комп’ютерного моделювання, який дозволяє отримати статистичні дані відносно процесів у модельованій системі *S*.

Все частіше використовується комбіноване моделювання, системотвірним елементом якого є аналітичні моделі. У побудові та використанні комбінованих моделей попередньо проводять декомпозицію процесу функціонування моделі на складові елементи.

З розвитком математичних досліджень ускладнюється й проблема класифікації моделей, що використовуються. Разом із виникненням нових типів моделей (особливо змішаних типів) і нових ознак їх класифікації здійснюється процес інтеграції моделей різних типів у більш складні модельні конструкції.

1.2. Імітаційне моделювання та метод статистичних випробувань

Існує клас об'єктів, для яких з різних причин не розроблені аналітичні моделі або не розроблені методи розв'язування задач про такі моделі. В цьому випадку математична модель замінюється імітатором або імітаційною моделлю.

Імітаційна модель — логіко-математичний опис об'єкту, який може бути використаний для експериментування на

комп'ютері в цілях проектування, аналізу і оцінки функціонування об'єкту. Імітаційне моделювання — це окремий випадок математичного моделювання, метод дослідження, заснований на тому, що система, яка вивчається, замінюється імітатором і з ним проводяться експерименти з метою отримання інформації про цю систему. Експериментування з імітатором називають імітацією (імітація — це збагнення суті явища, не вдаючись до експериментів на реальному об'єкті).

Імітаційне моделювання (машинна імітація) - особлива форма проведення експериментів на ЕОМ з математичними моделями, які з певним ступенем ймовірності описують закономірності функціонування реальних систем і об’єктів. Це метод, що дозволяє будувати моделі процесів, що описують, як ці процеси проходили б насправді. Таку модель можна «програти» в часі як для одного випробування, так і заданої їх кількості. При цьому результати визначатимуться випадковим характером процесів. За цими даними можна отримати достатньо стійку статистику.

Імітація як метод розв'язування нетривіальних задач отримала початковий розвиток у зв'язку із створенням ЕОМ в 1950х — 1960х роках.

Імітаційні (алгоритмічні) моделі можуть бути детермінованими і стохастичними. В останньому випадку за допомогою датчиків (генераторів) випадкових чисел імітується вплив (дія) невизначених і випадкових чинників. Такий метод імітаційного моделювання дістав назву методу статистичного моделювання (статистичних прогонів, чи методу Монте-Карло). На даний час цей метод вважають одним із найефективніших методів дослідження складних систем, а часто і єдиним практично доступним методом отримання нової інформації щодо поведінки гіпотетичної системи (на етапі її проектування).

Винахідником методу Монте-Карло називають Станіслава Улама (Stanislaw Ulam), американського математика, який народився у м. Львові. С.Улам перш за все відомий як людини яка брала участь в проектуванні водневої бомби з Едуардом Теллером

на початку 50-х років. Під час II Світової Війни Станіслав Марцин Улам і Джон фон Нейман (Neumann) працювали в Лос-Аламосі на Манхеттенському Проекті над моделюванням нейтронної дифузії в розщеплюваному матеріалі. Метод Монте-Карло він винайшов в 1946 р., коли одужуючи після хвороби, і, розкладаючи пасьянси, задався питанням, яка вірогідність того, що пасьянс «складеться». Йому в голову прийшла ідея, що замість того, щоб використовувати звичайні для подібних завдань міркування комбінаторики, можна просто поставити «експеримент» велике число раз і, таким чином, підрахувавши число вдалих результатів, оцінити їх вірогідність. Він же запропонував використовувати комп'ютери для розрахунків методом Монте-Карло. Працюючи з Джоном фон Нейманом і Ніколасом Метрополісом (N. Metropolis), він розвивав алгоритми для комп'ютерних виконань, також як і досліджував засіб перетворення не випадкових проблем у випадкові форми, які полегшили б їх розв’язок через статистичне здійснення вибірки. Ця робота перетворила статистичне здійснення вибірки з математичної цікавості- на формальну методологію, що застосовується до широкого кола різноманітних проблем.

Поява перших електронних комп'ютерів, які могли з великою швидкістю генерувати псевдовипадкові числа, різко розширила круг завдань, для вирішення яких стохастичний підхід виявився ефективнішим, ніж інші математичні методи. Після цього відбувся великий прорив і метод Монте-Карло застосовувався в багатьох завданнях, проте його використання не завжди було виправдано через велику кількість обчислень, необхідних для отримання відповіді із заданою точністю.

Роком народження методу Монте-Карло вважається 1949 рік, коли в світ виходить стаття Метрополіса і Улама «Метод Монте-Карло» в журналі Journal of American Statistical Association (Журнал американської статистичної звітності). Назва методу походить від назви міста в князівстві Монако, широко відомого своїми численними казино, оскільки саме рулетка є одним з найвідоміших генераторів випадкових чисел. Станіслав Улам пише в своїй автобіографії «Пригоди математика», що назва була

запропонована Метрополісом на честь його дядька, який був азартним гравцем.

Метод Монте-Карло — це сукупність формальних процедур, засобами яких відтворюються на ЕОМ будь-які випадкові фактори (випадкові події, випадкові величини з довільним розподілом, випадкові вектори тощо). У межах цього підходу будується ймовірнісна модель, яка відповідає математичній чи фізичній задачі, і на ній реалізується випадкова вибірка. «Розігрування» вибірок за методом Монте-Карло є основним принципом імітаційного моделювання систем із стохастичними (випадковими, імовірними) елементами.

Метод Монте-Карло — це метод імітації для приблизного відтворення реальних явищ. Він об'єднує аналіз чутливості (сприйнятливості) і аналіз розподілювання ймовірностей вхідних змінних. Цей метод дає змогу побудувати модель, мінімізуючи дані, а також максимізувати значення даних, які використовуються в моделі. Побудова моделі починається з визначення функціональних залежностей у реальній системі. Після чого можна одержати кількісне рішення, використовуючи теорію ймовірності й таблиці випадкових чисел. [2, стр.70]

Метод статистичного моделювання (чи метод Монте-Карло) — це спосіб дослідження невизначених (стохастичних) економічних об’єктів і процесів, коли не повністю (до певної міри) є відомими внутрішні взаємодії в цих системах.

Цей метод полягає у модельному відтворенні процесу за допомогою стохастичної математичної моделі та обчисленні характеристик цього процесу. Одне таке відтворення можливого (випадкового) стану функціонування модельованої системи називають реалізацією (чи імітаційним прогоном). Після кожного прогону реєструють сукупність параметрів, що характеризують випадкову подію (її реалізацію). Метод ґрунтується на багатократних прогонах (випадкових реалізаціях) на підставі побудованої моделі з подальшим статистичним опрацюванням отриманих даних з метою визначення числових характеристик досліджуваного об’єкта (процесу) у вигляді статистичних оцінок його параметрів. Процес моделювання економічної системи

зводиться до машинної імітації досліджуваного процесу, котрий моделюється на ЕОМ з усіма суттєвими невизначеностями, випадковостями і породженим ними ризиком.

Метод Монте-Карло широко використовується у всіх випадках імітації на ЕОМ. На сьогодні він охоплює будь-яку техніку статистичного здійснення вибірки, яке використовується для приблизних рішень кількісних проблем.

Він приміняється:

1. Для визначення площі довільних фігур.
2. Для вибору найкращих стратегій в задачах, де присутні багато випадкових факторів.
3. Для визначення ймовірності, чи відбудеться якась подія.
4. Для побудови різних геометричних об"єктів, в тому числі лабіринтів та фракталів.
5. Для моделювання поведінки складних екологічних та економічних систем.

Розв'язування задач методом статистичного моделювання полягає в слідуючому:

1. Опрацювання й побудова структурної схеми процесу, виявлення основних взаємозв'язків;
2. Формалізований опис процесу;
3. Моделювання випадкових явищ (випадкових подій, випадкових величин, випадкових функцій), що притаманні досліджуваній системі;
4. Моделювання процесу функціонування системи (на підставі використання даних, що отримані на попередньому етапі) — відтворення процесу відповідно до розробленої структурної схеми і формалізованого опису (імітаційні прогони);
5. Накопичення результатів моделювання (імітаційних прогонів), статистичне опрацювання, аналіз та інтерпретація їх.

Будь-які твердження стосовно характеристик модельованої системи повинні ґрунтуватися на результатах відповідних перевірок за допомогою методів математичної статистики.

Оскільки випадкові події й випадкові функції можуть подаватися з використанням випадкових величин, то й

моделювання випадкових подій і випадкових функцій проводиться за допомогою випадкових величин.

1.3. Побудова регресійних моделей

При вивчені статистичних зв'язків між різними ознаками економічного, наукового і т.д. (досліджуваного) об'єкта головною задачею є встановлення виду кореляційної залежності результуючої ознаки Y, від факторної X, тобто виду функціональної ознаки $\overline{Y} = f(x)$. В першу чергу це пов'язано з необхідністю прогнозування досліджуваних процесів. Математико-статистичний апарат, що дозволяє встановити вид кореляційної залежності називається **регресійний аналіз**, а функція що описує цю залежність, — **рівнянням регресії**.

Встановлення виду кореляційної залежності

Регресійний аналіз проводиться за такими етапами:

1. Встановлення виду кореляційної залежності результативної ознаки Y від факторної ознаки X;
2. Побудова регресійної моделі;
3. Перевірка статистичної значущості побудованої моделі.

Перший етап регресійного аналізу є найважливішим, оскільки помилки у виборі виду залежності призводять до побудови регресійної моделі, що не відповідає емпіричним даним і не може використовуватись для прогнозування.

Вибіркові дані для вивчення кореляційного зв'язку між ознаками Y та X зазвичай мають вигляд пар їх значень: $(x_1; y_1)$, $(x_2; y_2)$, …, $(x_n; y_n)$, x_i – значення величини X, y_i – значення Y, n – кількість пар значень, $i = \overline{1,n}$. Якщо їх кількість достатньо велика, то для зручності розрахунків дані групуються і будується статистичний ряд, що містить значення X, відповідні середні значення Y та частоти (Табл. 1.1)

Таблиця 0.1. Груповані статистичні дані

x_i	x_1	x_2	…	x_k
$\overline{y}_{x_i}$	$\overline{y}_{x_1}$	$\overline{y}_{x_2}$	…	$\overline{y}_{x_k}$
n_i	n_1	n_2	…	n_k

Згруповані дані (Табл. 1.1) зображуються графічно, що часто дозволяє визначити вид залежності *Y* від *X*.

Ламана лінія, що сполучає крапки з координатами $(x_i; \overline{y_{x_i}})$, називається **емпіричною лінією регресії**.

Якщо емпірична лінія регресії значно наближається до прямої лінії, то висувається гіпотеза про наявність лінійного зв'язку між досліджуваними ознаками (Рис. 1.2).

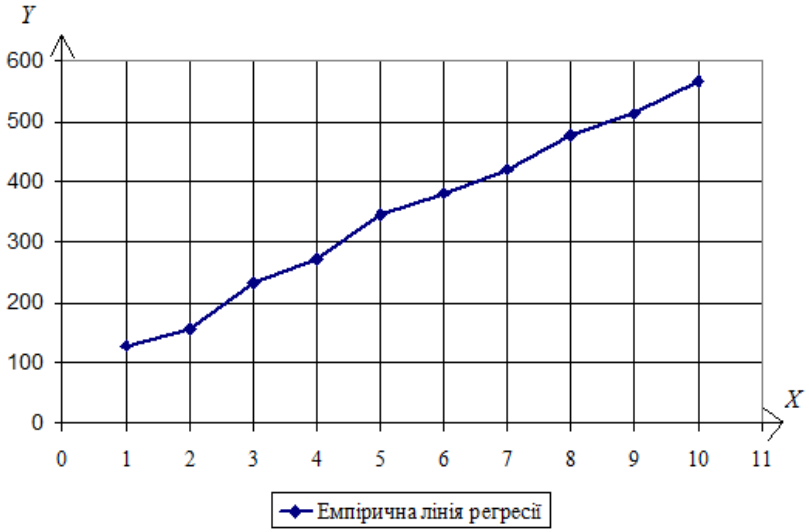


Рис. 0.2. Гіпотетична лінійна залежність

В іншому випадку висувається гіпотеза про наявність нелінійного зв'язку (Рис. 1.3).

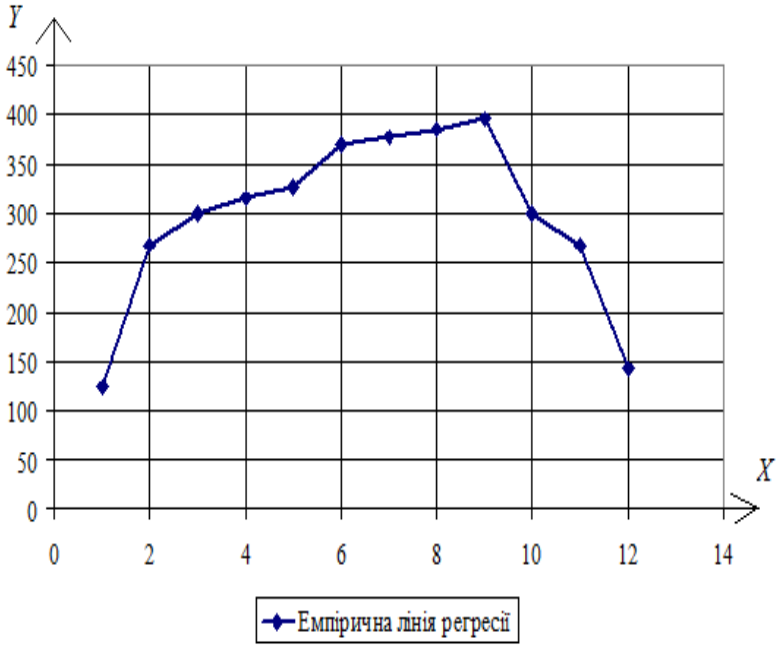


Рис. 0.3. Гіпотетична нелінійна залежність

1.3.1. Лінійна регресія

Якщо висунуто гіпотезу про наявність лінійної залежності результативної ознаки (Y) від факторної (X), то рівняння регресії має вид:

$$\overline{y_x} = ax + b, \qquad (1.1)$$

де a, b - параметри моделі.

Побудова лінійної регресійної моделі – це знаходження параметрів рівняння (1.1). Параметри рівняння регресії зазвичай знаходяться за **методом найменших квадратів**.

Ідея методу найменших квадратів

Нехай при вивчення залежності *Y* від *X* було отримано вибіркові дані: $x_1, x_2, ..., x_n$ – значення величини *X*, $y_1, y_2, ..., y_n$ – відповідні значення *Y*. За вибірковими даними було побудовано рівняння регресії $y = ax + b$. Якщо в рівняння підставити замість *x* значення $x_1, x_2, ..., x_n$, то будуть отримані теоретичні значення *Y*: $y_{1,теор}, y_{2,теор}, ..., y_{n,теор}$, які відрізняються від $y_1, y_2, ..., y_n$. Різниця значень $y_{i,теор} - y_i$ називається помилкою регресійної моделі і позначається e_i. Якщо параметри рівняння підбираються так, щоб сума квадратів помилок була мінімальною, то говорять, що вони отримані за методом найменших квадратів.

У випадку лінійної регресії параметри рівняння регресії за методом найменших квадратів знаходяться з системи лінійних алгебраїчних рівнянь:

$$\begin{cases} a\sum_{i=1}^{k} x_i^2 n_i + b\sum_{i=1}^{k} x_i n_i = \sum_{i=1}^{k} x_i n_i \overline{y_{x_i}} \\ a\sum_{i=1}^{k} x_i n_i + b\sum_{i=1}^{k} n_i = \sum_{i=1}^{k} n_i \overline{y_{x_i}} \end{cases} \qquad (1.2)$$

Якщо вибіркові дані не згруповані, то система (1.2) значно спрощується:

$$\begin{cases} a\sum_{i=1}^{n} x_i^2 + b\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} x_i y_i \\ a\sum_{i=1}^{n} x_i + bn = \sum_{i=1}^{n} y_i \end{cases} \qquad (1.3)$$

Перевірка правильності побудови рівняння регресії здійснюється за основним варіаційним рівнянням:

$$Q = Q_p + Q_o, \qquad (1.4)$$

де $Q = \sum_{i=1}^{k} \left(\overline{y_{x_i}} - \overline{y} \right)^2 n_i$ - загальна варіація, тобто сума квадратів відхилень емпіричних значень *Y* від середнього, $\overline{y} = \frac{\sum_{i=1}^{k} \overline{y_{x_i}} n_i}{n}$;

$Q_p = \sum_{i=1}^{k} \left(y_{i,теор} - \overline{y} \right)^2 n_i$ - варіація регресії, тобто сума квадратів відхилень теоретичних значень *Y* від середнього, що обумовлена регресією;

$Q_o = \sum_{i=1}^{k} \left(y_{i,теор} - \overline{y}_{x_i} \right)^2 n_i$ - варіація залишків, тобто сума квадратів відхилень теоретичних значень *Y* від емпіричних.

У випадку незгрупованих даних загальна варіація, варіації регресії і залишків знаходяться за формулами: $Q = \sum_{i=1}^{n} \left(y_i - \overline{y} \right)^2$;

$Q_p = \sum_{i=1}^{n} \left(y_{i,\text{теор}} - \overline{y} \right)^2$; $Q_o = \sum_{i=1}^{n} \left(y_{i,\text{теор}} - y_i \right)^2$; а середнє значення за формулою $\overline{y} = \frac{\sum_{i=1}^{n} y_i}{n}$.

Для перевірки статистичної значущості рівняння регресії розраховується F-статистика за формулою:

$$F = \frac{Q_p \left(n - l \right)}{Q_o \left(l - 1 \right)}, \quad (1.5)$$

де *n* – кількість наглядів, *l* – кількість груп у кореляційній таблиці або кількість параметрів моделі у випадку незгрупованих даних. Розраховане значення F-статистики порівнюється з критичним значенням $F_{кр}$ розподілу Фішера, яке можна знайти за статистичними таблицями або за допомогою вбудованої функції

Excel $FPACПОБР(\alpha, k_1, k_2)$, де $k_1 = l - 1$; $k_2 = n - l$ - ступені волі, α - рівень значущості.

Адекватність моделі вибірковим даним можна оцінити за коефіцієнтом детермінації R^2, що показує частину варіації значень результативної ознаки *Y*, що пояснюється рівнянням регресії. Коефіцієнт детермінації розраховується за формулою:

$$R^2 = 1 - \frac{Q_o}{Q} = \frac{Q_p}{Q}.$$

Значення коефіцієнта детермінації знаходяться в інтервалі $[0;1]$, тобто $0 \le R^2 \le 1$. Чим ближче R^2 до 1, тим краще отримане рівняння регресії пояснює поведінку результативної ознаки. Наприклад, якщо R^2=0,98, то 98% варіації результативної ознаки *Y* пояснюється рівнянням регресії.

ПРИКЛАД 1.1.

Побудувати регресійну модель, що описує залежність сумарних виробничих затрат *Y* (тис. грн.) від об'ємів виробництва *X* (тис. од.). Відповідні статистичні дані надано у таблиці 1.2.

Таблиця 0.2. Відповідні статистичні дані

X	41	44	52	57	59	64	68	70	73	75
Y	670	657	713	736	778	812	833	876	911	932

Розв'язок. В таблиці 1.2 надано вибіркові дані: значення $x_i, i = 1, n$ величини *X* та відповідні значення $y_i, i = 1, n$; кількість пар – $n = 10$ невелика, тому для проведення регресійного аналізу їх можна не групувати.

Перший етап аналізу: визначимо вид залежності *Y* від *X*. Побудуємо емпіричну лінію регресії (Рис. 1.4).

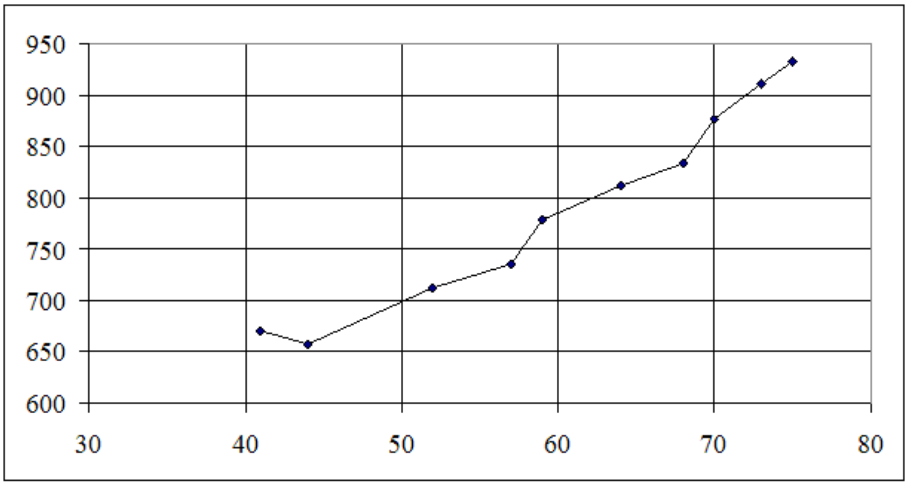


Рис. 0.4. Емпірична лінійна регресія

Оскільки емпірична лінія регресії наближається до прямої лінії, то висуваємо гіпотезу про лінійну залежність Y від X, тобто рівняння регресії будемо шукати у вигляді $y = ax + b$.

Другий етап: знайдемо параметри a, b рівняння регресії, для чого складемо систему (1.3) для даних, що не згруповані. Необхідні розрахунки для зручності оформимо у вигляді таблиці (табл. 1.3).

Таблиця 0.3. Приклади розрахунків

Розрахункова таблиця											Сум и
x_i	41	44	52	57	59	64	68	70	73	75	603
y_i	670	657	713	736	778	812	833	876	911	932	791 8

x_i^2	1681	1936	2704	3249	3481	4096	4624	4900	5329	5625	37625
$x_i y_i$	27470	28908	37076	41952	45902	51968	56644	61320	66503	69900	487643

Отже, складемо систему для знаходження параметрів рівняння регресії та розв'яжемо її за правилом Крамера:

$$\begin{cases} a\sum_{i=1}^{n} x_i^2 + b\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} x_i y_i \\ a\sum_{i=1}^{n} x_i + bn = \sum_{i=1}^{n} y_i \end{cases} \Rightarrow \begin{cases} 37625a + 603b = 487643 \\ 603a + 10b = 7918 \end{cases}.$$

Знайдемо головний визначник системи, що складений із коефіцієнтів перед невідомими:

$$\Delta = \begin{vmatrix} 37625 & 603 \\ 603 & 10 \end{vmatrix} = 37625 \cdot 10 - 603^2 = 12641.$$

Знайдемо допоміжні визначники, що отримуються із головного заміною відповідного стовпця коефіцієнтів на стовпець вільних членів:

$$\Delta a = \begin{vmatrix} 487643 & 603 \\ 7918 & 10 \end{vmatrix} = 487643 \cdot 10 - 603 \cdot 7918 = 101876;$$

$$\Delta b = \begin{vmatrix} 37626 & 487643 \\ 603 & 7918 \end{vmatrix} = 37626 \cdot 7918 - 48743 \cdot 603 = 3866021$$

.

Знайдемо невідомі за формулами Крамера:

$$a = \frac{\Delta a}{\Delta} = \frac{101876}{12641} \approx 8{,}06; \quad b = \frac{\Delta b}{\Delta} = \frac{3866021}{12641} \approx 305{,}83.$$

Отже, шукане рівняння регресії має вигляд

$$y = 8{,}06x - 305{,}83.$$

Третій етап: перевіримо правильність побудови моделі за рівнянням (1.4), її статистичну значущість за F-статистикою (1.5) і адекватність вибірковим даним за коефіцієнтом детермінації (1.6). Для чого знайдемо загальну варіацію, варіації регресії та залишків; необхідні розрахунки оформимо у вигляді таблиці (табл. 1.4).

Передусім знайдемо $\bar{y}$: $\bar{y} = \frac{\sum_{i=1}^{n} y_i}{n} \approx 791{,}8$.

Таблиця 0.4. Приклад розрахунків

x_i	y_i	$y_{i,\text{теор}}$	$\left(y_i - \bar{y}\right)^2$	$\left(y_{i,\text{теор}} - \bar{y}\right)^2$	$\left(y_{i,\text{теор}} - y_i\right)^2$
41	670	636,26	14835,24	24193,32	1138,52
44	657	660,44	18171,04	17256,64	11,80
52	713	724,91	6209,44	4474,42	141,82
57	736	765,20	3113,64	707,31	852,92
59	778	781,32	190,44	109,77	11,04
64	812	821,62	408,04	889,17	92,52
68	833	853,86	1697,44	3850,90	434,96
70	876	869,97	7089,64	6111,17	36,31
73	911	894,15	14208,64	10475,83	283,87
75	932	910,27	19656,04	14035,10	472,20
		Суми	85579,6	82103,63	3475,974

Отже, $Q = 85579{,}6$; $Q_p = 82103{,}63$; $Q_o = 3475{,}974$; тоді основне варіаційне рівняння $Q = Q_p + Q_o$ для побудованої моделі має вигляд: $85579{,}6 = 82103{,}63 + 3475{,}974$ і є тотожністю, тому рівняння регресії побудовано правильно.

Для перевірки статистичної значущості рівняння регресії знайдемо F-статистику, враховуючи, що n=10, l=2 – оскільки шукали рівняння з двома параметрами:

$$F = \frac{Q_p(n-l)}{Q_o(l-1)} = \frac{82103(10-2)}{3475,974(2-1)} \approx 188,96.$$

Знайдемо $F_{кр}$: $F_{кр} = FРАСПОБР(0,001;2-1;10-2) \approx 25,41$. Розраховане значення F-статистики більше критичного, тому регресійна модель є статистично значущою на рівні 0,001.

Знайдемо коефіцієнт детермінації R^2: $R^2 = \frac{Q_p}{Q} = \frac{82103,63}{85579,6} \approx 0,96$. Значення коефіцієнта детермінації свідчить, що 96% варіації результативної ознаки Y пояснюються рівнянням регресії.

Висновок: Сумарні виробничі затрати Y (тис. грн.) лінійно залежать від об'єму виробництва X (тис. од.). Залежність описується рівнянням $y = 8,06x - 305,83$, яке є статистично значимим на рівні значущості 0,001 та описує 96% вибіркових даних.

Розділ 2. Представлення моделі якості засвоєння базової дисципліни

2.1 Розробка методологічних основ побудови математичної моделі базової дисципліни в рамках роботи наукової школи

Нехай, Y — екзаменаційна оцінка студента (від 0 до 100 балів за шкалою EST — результуюча ознака) [2].

Досліджувані фактори:

X_1 – інтерес до вивчення дисципліни (0-5 балів);
X_2 – оцінка студентами роботи викладача (0-5 балів);

X_3 – трудність вивчення дисципліни (0-5 балів);
X_4 – елементи наукового пошуку (0-5 балів);
X_5 – зв'язок зі спеціальністю (0-5 балів);
X_6 – степінь самостійності в написанні першої монографії (0-5 балів);
X_7 – степінь самостійності в написанні другої монографії (0-5 балів);
X_8 – оцінка студентами створеної наукової школи (0-5 балів);

X_1 – інтерес до вивчення дисципліни:

«0 балів» – інтерес до вивчення дисципліни відсутній; «В мене абсолютно відсутнє бажання вивчати дану дисципліну і оцінка на екзамені мене не цікавить»;

«1 бал» – інтерес до вивчення дисципліни обумовлений необхідністю отримати задовільну оцінку на екзамені «50-59 балів» – E;

«2 бали» – інтерес до вивчення дисципліни обумовлений необхідністю отримати задовільну оцінку що відповідає шкалі EST D «60-75 балів»; «Пристойно, але зі значними недоліками»;

«3 бали» – «Мені потрібна оцінка C «76-79 балів» для того, щоб була четвірка у виписці до диплому»;

«4 бали» – інтерес до дисципліни високий, відповідає шкалі EST «80-89 балів» – «Дуже добре, вище середнього стандарту»;

«5 балів» – підвищений інтерес; «Я бажаю внести свій внесок в дану дисципліну» – рівень творчості.

X_2 – оцінка студентами роботи викладача: – відповідає традиційній екзаменаційній оцінці роботи студента «від 0 до 5 балів» з тією різницею, що оцінку роботи студента за семестр ставить викладач, а оцінку роботи викладача за семестр ставить студент.

X_3 – трудність вивчення дисципліни:

« 0 балів» – ніякої складності у вивченні даної дисципліни немає;

«1 бал» – при вивченні даної дисципліни потрібні мінімальні затрати сил і часу;

«2 бали» – до вивчення дисципліни необхідно прикласти деякі зусилля і час;

«3 бали» – методика викладання дисципліни автоматично забезпечує добру оцінку на екзамені;

«4 бали» – до вивчення дисципліни потрібна значна концентрація зусиль і часу;

«5 балів» – максимальна концентрація зусиль і часу гарантує високу оцінку на екзамені.

X_4 – елементи наукового пошуку:

«0 балів» – вся інформація при вивченні даної дисципліни добре представлена у рекомендованій літературі;

«1 бал» – необхідно вести конспект лекцій, в якому висвітлюються матеріали, яких не можна почерпнути із відомих літературних джерел;

«2 бали» – без конспекту лекцій неможливо пробробляти практичні заняття;

«3 бали» – на практичних роботах вирішуються задачі, які потребують творчого підходу і максимального використання комп'ютерної техніки;

«4 бали» – максимальне використання теоретичного матеріалу лекційного курсу в поєднанні із максимальним використанням комп'ютерної техніки;

«5 балів» – написання власних монографій під керівництвом наукового керівника.

X_5 – зв'язок зі спеціальністю:

«0 балів» – «Я не можу відмітити зв'язку зі спеціальністю;

«1 бал» – зв'язок зі спеціальністю незначний;

«2 бали» – зв'язок зі спеціальністю помірний;

«3 бали» – зв'язок зі спеціальністю добрий;
«4 бали» – зв'язок зі спеціальністю високий;
«5 балів» – зв'язок зі спеціальністю повний.

X_6, X_7 – степінь самостійності в написанні монографії:

«0 балів» – я не зміг завершити дослідження, щоб написати монографію;
«1 бал» – монографія не завершена;
«2 бали» – «Мені допомогли завершити роботу над монографією»;
«3 бали» – «Я сам написав монографію при консультації і наявності допоміжних матеріалів»;
«4 бали» – «Необхідні розрахункові файли створені мною особисто»;
«5 балів» – « Монографія написана, набрана на комп'ютері і видана при моїй же власній авторській редакції».

X_8 – оцінка студентами створеної наукової школи:

«0 балів» – наукова школа не відбулась, монографії не написані;
«1 бал» – 10 відсотків студентів написали власні монографії;
«2 бали» – 25 відсотків студентів написали монографії;
«3 бали» – 50 відсотків студентів написали монографії;
«4 бали» – 75 відсотків студентів написали монографії;
«5 балів» – 85 відсотків студентів написали монографії.

Після проведення екзаменаційної сесії студенти провели експертну оцінку і була отримана наступна зведена таблиця за результатами анкетування. Даний базовий курс вивчало 38 студентів.

Таблиця 0.1. Зведена таблиця успішності по шкалі EST

№п.п.	Екз.оц.		Інтерес вивчення дисципл.	Оцінка викладачу	Трудність вивчення дисципліни	Елем.наук. пошуку	Зв'язок зі спец.	Оцінка моногр.1	Оцінка моногр.2	Оцінка Наук.школ.
	У	X0	X1	X2	X3	X4	X5	X6	X7	X8
1	100	1	5	5	4	4	4	5	5	5
2	90	1	5	5	5	5	5	5	5	5
3	90	1	5	5	5	5	5	5	5	5
4	100	1	5	5	3	5	5	5	5	5
5	89	1	4	4	3	4	4	5	4	5
6	89	1	5	5	3	4	5	5	5	5
7	95	1	5	5	5	5	5	5	5	5
8	100	1	5	5	2	5	5	5	5	5
9	90	1	5	5	5	5	5	5	5	5
10	89	1	4	5	4	5	4	5	0	5
11	100	1	5	5	5	5	5	5	5	5
12	80	1	4	5	4	5	4	0	0	4
13	89	1	4	5	4	4	4	5	4	5
14	90	1	5	5	3	5	5	5	5	5
15	100	1	5	5	4	3	5	5	5	5
16	90	1	5	5	4	4	5	5	5	5
17	100	1	4	5	4	4	4	5	5	5
18	100	1	5	5	5	5	5	4	5	5
19	77	1	5	5	3	5	5	4	0	5
20	77	1	5	5	3	5	5	5	5	5
21	100	1	5	5	5	5	5	5	5	5

22	100	1	5	5	4	4	5	5	5	5
23	90	1	4	5	4	4	4	5	4	4
24	100	1	5	5	3	5	5	5	5	5
25	100	1	5	5	3	5	5	5	5	5
26	100	1	5	5	4	4	5	5	5	5
27	100	1	5	5	3	5	5	5	5	5
28	100	1	5	5	5	5	5	5	5	5
29	100	1	5	5	3	5	5	5	5	5
30	85	1	4	5	5	5	5	5	5	5
31	90	1	5	5	3	5	5	5	5	5
32	90	1	4	5	4	5	5	5	5	5
33	86	1	5	5	5	5	5	5	5	5
34	86	1	5	5	3	5	5	5	5	5
35	100	1	5	5	3	5	5	5	5	5
36	90	1	5	5	5	5	5	5	5	5
37	95	1	5	5	3	4	5	5	5	4
38	100	1	5	5	5	5	5	5	5	5

2.2 Представлення істинної моделі.

За результатами строгого зрівноваження отримана емпірична модель базової дисципліни , яку приймаємо за істинну модель [2,18]

$$Y_{моделі}' = 53.933095X_0 + 5.379875X_1 + 5.14170X_2 - \\ -0.063645X_3 - 1.049493X_4 - 6.503593X_5 - \\ 0.1142141X_6 + 2.433299X_7 + 2.890344X_8.$$

Таблиця 0.2. Вихідні дані істинної моделі у табличному вигляді

x	1,6	2	2,1	2,3	2,5	2,8	2,9	3	3,1	3,3
y	18,0	13,8	13,1	11,9	10,8	8,9	8,1	7,1	5,9	2,9

	21	64	67	86	98	49	01	08	39	65

Побудувавши ймовірнішу модель за способом найменших квадратів (тобто знайшовши параметри (коефіцієнти) емпіричної формули) і зробивши оцінку точності її елементів, в подальшому необхідно провести дослідження точності істинної якості засвоєння базової дисципліни методом статистичних випробувань Монте- Карло. Для цього необхідно генерувати істинні похибки за допомогою генератора випадкових чисел.

2.3. Генерування істинних похибок для дослідження математичної моделі методом статистичних випробувань Монте- Карло.

Приведемо методику розрахунку випадкових чисел, які приймемо в подальшому, як істинні похибки для побудови спотвореної моделі [23- с.30].

1.Отримавши ряд випадкових (а точніше псевдовипадкових) чисел $\xi_{ср.}$, розраховують середнє арифметичне генерованих псевдовипадкових чисел $\xi_{\text{ср}}$:

$$\xi_{\text{ср}} = \frac{\sum_{i=1}^{n} \xi_i}{n} \qquad (2.1)$$

де n – сума випадкових чисел.

2. Розраховуються попередні значення істинних похибок Δ_i' за формулою

$$\Delta_i' = \xi_i - \xi_{\text{ср}} \qquad (2.2)$$

3. Знаходять середню квадратичну похибку попередніх значень істинних похибок :

$$m_{\Delta'} = \sqrt{\frac{\sum_{i=1}^{n} \Delta'^2 i}{n}} \qquad (2.3)$$

4. Знаходять коефіцієнт пропорційності K, для визначення істинних похибок необхідності точності

$$\mathrm{K} = \frac{c}{m'_{\Delta}}, \qquad (2.4)$$

де c – необхідна константа.

Так, наприклад, при $m'_{\Delta} = 0{,}63254$ і необхідності побудови математичної моделі з точністю $c=0{,}1$, будемо мати

$$\mathrm{K}_{0,1} = \frac{0{,}1}{0{,}63254} = 0{,}15809,$$

при $c=0{,}05$ K буде дорівнювати 0,07905.

5. Істинні похибки розраховуються за формулою

$$\Delta_i = \Delta'_i \cdot \mathrm{K} \qquad (2.5)$$

6. Заключним контролем служить розрахунок середньої квадратичної похибки m_{Δ} генерованих істинних похибок Δ

$$m_{\Delta} = \sqrt{\frac{\sum_{i=1}^{n} \Delta^2}{n}} \qquad (2.6)$$

і порівняння

$$m_{\Delta} = C \qquad (2.7)$$

Генеровані нами похибки , розрахунок попередніх значень істинних похибок , самі істинні похибки представлені в таблиці 2.

Таблиця 0.3. Генерування псевдовипадкових чисел і розрахунок істинних похибок

	ξ=слчис()*0,01*N	ξсередн.	$\Delta_i^{'} = \xi_i - \xi_{ср.}$	$\Delta i^{'2}$	$\Delta i = k * \Delta_i^{'}$	Δ_I^2
2	0,0165	0,063	-0,046	0,002	-0,633	0,401
3	0,0269	0,063	-0,036	0,001	-0,490	0,241
4	0,0609	0,063	-0,002	0,000	-0,025	0,001
5	0,084	0,063	0,021	0,000	0,292	0,085
6	0,0693	0,063	0,007	0,000	0,090	0,008
7	0,0532	0,063	-0,010	0,000	-0,130	0,017
8	0,0478	0,063	-0,015	0,000	-0,204	0,042
9	0,0208	0,063	-0,042	0,002	-0,574	0,330
10	0,121	0,063	0,058	0,003	0,799	0,638
11	0,1247	0,063	0,062	0,004	0,849	0,721
12	0,0013	0,063	-0,061	0,004	-0,841	0,708
13	0,139	0,063	0,076	0,006	1,045	1,092
14	0,06	0,063	-0,003	0,000	-0,037	0,001
15	0,068	0,063	0,005	0,000	0,073	0,005
16	0,0592	0,063	-0,004	0,000	-0,048	0,002
17	0,062	0,063	-0,001	0,000	-0,010	0,000
18	0,0616	0,063	-0,001	0,000	-0,015	0,000
19	0,0773	0,063	0,015	0,000	0,200	0,040
20	0,105	0,063	0,042	0,002	0,579	0,336
21	0,1121	0,063	0,049	0,002	0,677	0,458
22	0,0422	0,063	-0,021	0,000	-0,281	0,079
23	0,0383	0,063	-0,024	0,001	-0,334	0,112
24	0,0198	0,063	-0,043	0,002	-0,588	0,345
25	0,0625	0,063	0,000	0,000	-0,003	0,000
26	0,0004	0,063	-0,062	0,004	-0,854	0,728
27	0,1267	0,063	0,064	0,004	0,877	0,769
28	0,0215	0,063	-0,041	0,002	-0,564	0,319
29	0,0572	0,063	-0,006	0,000	-0,075	0,006
30	0,0082	0,063	-0,055	0,003	-0,747	0,558
31	0,0938	0,063	0,031	0,001	0,426	0,181
32	0,102	0,063	0,039	0,002	0,538	0,290

33	0,0622	0,063	-0,001	0,000	-0,007	0,000
34	0,0465	0,063	-0,016	0,000	-0,222	0,049
35	0,1241	0,063	0,061	0,004	0,841	0,707
36	0,0381	0,063	-0,025	0,001	-0,337	0,114
37	0,0707	0,063	0,008	0,000	0,110	0,012
38	0,0392	0,063	-0,024	0,001	-0,322	0,104
39	0,0588	0,063	-0,004	0,000	-0,053	0,003
40	2,383	2,383	0,000	0,051	0,000	9,500

Середня квадратична похибка попередніх істинних похибок

$$m\Delta i' = \sqrt{\left(\left[\Delta i'^{2} / n\right]\right)} = 0,036499235$$

Коефіцієнт пропорційності

$$K = \frac{0,5}{0,036499235} = 13,698917..$$

Середня квадратична похибка при генеруванні випадкових чисел з точністю $c = 0,5$

$$m_{\Delta_i} = \sqrt{\frac{9,500}{38}} = 0,5 \ .$$

Отже, $m_{\Delta} = c = 0,5$.

2.4. Побудова спотвореної моделі

Визначимо $x_{спотворене}$ за формулою

$$x_{спотв.} = x_{іст} + \Delta_i \qquad (2.8)$$

Дані занесено в таблицю 3.

Таблиця 0.4. Побудова спотвореної моделі

№	Істинна модель		Δ_i	$Y_{спотв.} = Y_{іст.} + \Delta_i$
	Екз.оцін.	Yіст.=X*A		
2	100	102,4597125	-0,633	101,82675
3	90	94,44050746	-0,490	93,95001
4	90	94,44050746	-0,025	94,41578
5	100	94,58812998	0,292	94,87984

6	89	89	0,090	89,09034
7	89	95,55514436	-0,130	95,42493
8	95	94,44050746	-0,204	94,23632
9	100	94,66194123	-0,574	94,08788
10	90	94,44050746	0,799	95,23908
11	89	82,81828264	0,849	83,66754
12	100	94,44050746	-0,841	93,59932
13	80	80,19449082	1,045	81,23965
14	89	94,12678395	-0,037	94,08972
15	90	94,58812998	0,073	94,66066
16	100	96,44834749	-0,048	96,40033
17	90	95,48133311	-0,010	95,47167
18	100	96,71215568	-0,015	96,69701
19	100	94,40339101	0,200	94,60332
20	77	81,62415487	0,579	82,20355
21	77	94,58812998	0,677	95,26478
22	100	94,44050746	-0,281	94,15961
23	100	95,48133311	-0,334	95,14701
24	90	91,68857438	-0,588	91,10082
25	100	94,58812998	-0,633	94,58532
26	100	94,58812998	-0,854	93,73462
27	100	95,48133311	0,877	96,35799
28	100	94,58812998	-0,564	94,02366
29	100	94,44050746	-0,075	94,36509
30	100	94,58812998	-0,747	93,84147
31	85	88,69295063	0,426	89,11891
32	90	94,58812998	0,538	95,12643
33	90	88,76676189	-0,007	88,75984
34	86	94,44050746	-0,222	94,21851
35	86	94,58812998	0,841	95,42917
36	100	94,58812998	-0,337	94,25106
37	90	94,44050746	0,110	94,55003
38	95	93,11693479	-0,322	92,79494
39	100	94,44050746	-0,053	94,38701
∑	3547	3547	0,000	3547,00000

По даним спотвореної моделі виконують строге зрівноваження методом найменших квадратів і отримують ймовірніші моделі, яким роблять оцінку точності зрівноважених елементів і дають порівняльний аналіз на основі якого заключають на предмет

поширення даної моделі для рішення даної проблеми в цілому.

2.5. Підбір параметрів способом найменших квадратів.

Метод найменших квадратів (МНК) — це математично-статистичний метод, який полягає в тому, що функція (котра може бути відомою, або заданою динамічним рядом чи таблицею експериментальних даних) для опису деякого явища апроксимується більш простою функцією (лінійною функцією, параболою, поліномами різного ступеня тощо). Апроксимуюча функція добирається таким чином, щоб середньоквадратичне відхилення (сума квадратів відхилень) фактичних рівнів функції в спостережуваних точках від вирівняних було найменшим.

На практиці часто приходиться розв"язувати таку задачу: для двох функціонально зв"язаних величин величин x та y відомі n пар відповідних значень (x_1 ; y_1), (x_2 ; y_2), …(x_n ; y_n). Потрібно в наперед заданій формулі $y = f(x,\ a_1, a_2, \ldots, a_m)$ визначити m параметрів $a_1,\ a_2,\ \cdots,\ a_m$ $(m < n)$ так щоб в цю формулу найкращим чином "вкладалися" б відомі n значень x та y.

Вважається (виходячи з принципів теорії ймовірності), що найкращими будуть такі значення $a_1,\ a_2,\ \cdots,\ a_m$, які перетворюють в мінімум суму

$$\sum_{k=1}^{k=n} \left[f\ (x_k,\ a_1,\ a_2, \cdots,\ a_{m)} - y_k \right]^2$$

(тобто суму квадратів відхилень значень y, котрі вирахувані по формулі, від заданих, тому сам спосіб і отримав назву способу найменших квадратів).

Ця умова дає систему m рівнянь, з котрих визначаються $a_1,\ a_2,\ \cdots,\ a_m$:

$$\sum_{k=1}^{k=n}\left[f\ (x_k,\ a_1,\ a_2,\cdots,\ a_m)-y_k \right]\cdot \frac{df(x_k,\ a_1,\ a_2,\ \cdots,\ a_m)}{da_j}=0, \tag{2.9}$$

де $(j=1,\ 2,\ \cdots,\ m)$.

В випадку $y=a_0x^m+a_1x^{m-1}+\cdots+a_m \qquad (m+1)$ параметрів $a_0,\ a_1,\ \cdots,\ a_m;\ n>m+1)$, система (4.1) приймає вигляд:

$$\begin{aligned}
&a_0\cdot\sum_{k=1}^{k=n}x_k^m\ +\ a_1\cdot\sum_{k=1}^{k=n}x_k^{m-1}\ +\cdots+\ n\cdot a_m=\sum_{k=1}^{k=n}y_k,\\
&a_0\cdot\sum_{k=1}^{k=n}x_k^{m+1}\ +\ a_1\cdot\sum_{k=1}^{k=n}x_k^m\ +\cdots+\ a_m\cdot\sum_{k=1}^{k=n}x_k=\sum_{k=1}^{k=n}x_k.y_k,\\
&a_0\cdot\sum_{k=1}^{k=n}x_k^{m+2}\ +\ a_1\cdot\sum_{k=1}^{k=n}x_k^{m+1}\ +\cdots+\ a_m\cdot\sum_{k=1}^{k=n}x_k^2=\sum_{k=1}^{k=n}x_k^2.y_k,\\
&\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\\
&a_0\cdot\sum_{k=1}^{k=n}x_k^{2m}\ +\ a_1\cdot\sum_{k=1}^{k=n}x_k^{2m-1}\ +\cdots+\ a_m\cdot\sum_{k=1}^{k=n}x_k^m=\sum_{k=1}^{k=n}x_k^m.y_k
\end{aligned} \tag{2.10}$$

Для визначення коефіцієнтів системи (2.10) зручно складати допоміжну таблицю, де в останньому рядку записуються суми елементів кожного стовбчика які і є коефіцієнтами системи (2.10).

2.5.1. Реалізація процедури строгого зрівноваження

За формулою

$$\begin{bmatrix} Y_1 \\ Y_2 \\ Y_3 \\ \bullet \\ \bullet \\ \bullet \\ Y_{38} \end{bmatrix} * = \begin{bmatrix} \sum_{i=1}^{38} Y_i \\ \sum_{i=1}^{38} Y_i X_{1i} \\ \sum_{i=1}^{38} Y_i X_{2i} \\ \sum_{i=1}^{38} Y_i X_{3i} \\ \cdot \\ \cdot \\ \cdot \\ \sum_{i=1}^{38} Y_i X_{8i} \end{bmatrix}$$

отримаємо вектор вільних членів нормальних рівнянь

=МУМНОЖ(A46:AL54;AI2:Ai39) F2,Ctrl+Shift+Enter

L'=Xт*Yспт.
3547
17021,23616
17645,90966
13821,35646
16594,19815
17097,28816
17151,9949
16225,16543
17469,8646

Вектор вільних членів

Вектор коефіцієнтів математичної моделі ,побудованої в даній монографії, отримаємо за формулою

$$A' = QL' \quad (2.11)$$

І в нашому випадку
=МУМНОЖ(A68:I76;R68:R76) F2,Ctrl+Shift+Enter

A'=Q*L'	
53,933095	a0
5,379875	a1
5,141703	a2
-0,063645	a3
-1,049493	a4
-6,503593	a5
-0,142141	a6
2,433299	a7
2,890344	a8
Вектор коефіцієнтів	
зрівноваженої моделі	

Таким чином, на основі проведених нами досліджень, отримана емпірична формула математичної моделі якості базової дисципліни в рамках наукової школи

$$Y_{моделі}' = 53.933095X_0 + 5.379875X_1 + 5.14170X_2 - \\ -0.063645X_3 - 1.049493X_4 - 6.503593X_5 - \\ 0.1142141X_6 + 2.433299X_7 + 2.890344X_8. \quad (4.12)$$

2.5.2. Контроль зрівноваження

Перший контроль виконання процедури зрівноваження виконується за формулою

$$L' = N * A' \qquad (2.13)$$

або для нашого розрахункового файла
=МУМНОЖ(A57:I65;T68:T76)) F2,Ctrl+Shift+Enter (4.6)

І в нашому випадку

L'=N*A'
3547,000
17021,236
17645,910
13821,356
16594,198
17097,288
17151,995
16225,165
17469,865
Контроль1

Другий контроль процедури зрівноваження виконується за формулою

$$\begin{aligned}&[YY]-a_0[Y]-a_1[YX_1]-a_2[YX_2]-a_3[YX_3]-a_4[YX_4]- \\ &-a_5[YX_5]-a_6[YX_6]-a_7[YX_7]-a_8[YX_8]=[VV]\end{aligned} \qquad (2.14)$$

У формулі (6.3) символом [ ] позначені суми за Гаусом. Розрахунок був проведений в MS Excel за формулою

=S40-МУМНОЖ(ТРАНСП(T68:T76);R68:R76))
F2,Ctrl+Shift+Enter (2/15)

В чарунку S40 знаходилася сума квадратів [YY], в діапазоні (T68:T76) знаходилися значення $a_0, a_1, a_2, \ldots, a_8$, в діапазоні $(R68:R76)$ знаходилися вільні члени нормальних рівнянь.

В матричній формі запис формули контролю зрівноваження буде

$$\left|\mathrm{Y}^T\mathrm{Y}\right| - \ell K^T = \left|V^T V\right|$$

В нашому випадку отримали

$$\left|\mathrm{Y}^T\mathrm{Y}\right| - \ell K^T = 6{,}4643652,$$
$$\left|V^T V\right| = 6{,}4643652.$$

Різниця між даними числами склала Δ=0,00000002, що говорить про коректність процедури зрівноваження в цілому.

Третім контролем процедури зрівноваження був розрахунок за формулою

=ЛИНЕЙН(R2:R39;H2:P39;1;1) F2,Ctrl+Shift+Enter

Діапазоном (R2:R39) відмічені екзаменаційні оцінки $Y_{сп}$, діапазоном (H2:P39) відмічені результати експертних оцінок студентів.

Як видно із табл.6.1 , лише для коефіцієнтів a8,a7,a5,a4,a2,a1 і a0 середні квадратичні похибки менші самих коефіцієнтів.

Таблиця 0.5. Другий контроль процедури зрівноваження

a8	a7	a6	a5		
2,890343953	2,433299023	-0,14214063	6,503592932	=ai	A"трансп
0,36853683	0,083959954	0,13876732	0,350531961	стандарт S	ai=S√dii
0,988708458	0,47213255	#Н/Д	#Н/Д	R^2	μ
317,4117628	29	#Н/Д	#Н/Д	Fкритерій	n-m-1
566,0318761	6,464365192	#Н/Д	#Н/Д	[(Y'-Yср)^2]	[VV]
a8	a7	a6	a5		

a4	a3	a2	a1	a0
-1,049493	-0,063645	5,14170255	5,379875	53,933095
0,177414	0,091649	0,54223826	0,298263	2,8954443
#Н/Д	#Н/Д	#Н/Д	#Н/Д	#Н/Д
#Н/Д	#Н/Д	#Н/Д	#Н/Д	#Н/Д
#Н/Д	#Н/Д	#Н/Д	#Н/Д	#Н/Д

Розраховуючи зрівноважені значення $\widetilde{Y}$, отримали

Коваріаційна матриця $K=N^{-1}\mu^2$

8,383597499	0,038852781	-1,307238261	0,021825
0,038852781	0,088960696	-0,027024645	0,004352
-1,307238261	-0,02702464	0,294022336	-0,01043
0,021825238	0,004352011	-0,010429213	0,008399
0,021722177	0,007141627	-0,008883669	-0,0011
0,068521047	-0,06348244	-0,031833165	0,000844
0,024426591	0,003295768	-0,00202444	0,001768
0,007408396	-0,00294619	0,004154258	-0,00151
-0,554745001	-0,01579676	0,040799699	-0,0049
X0	X1	X2	X3

Продовження коваріаційної матриці $K=N^{-1}\mu^2$

0,0217222	0,068521047	0,024426591	0,007408	-0,55475
-0,0071416	0,063482444	0,003295768	-0,00295	-0,0158
-0,008884	0,031833165	-0,00202444	0,004154	0,0408
-0,001098	0,000844334	0,001768282	-0,00151	-0,0049
-0,0314757	0,023835076	0,004692745	0,00294	-0,01545
-0,023835	0,122872656	-0,000794078	-0,0085	-0,00959
-0,0046927	0,000794078	0,01925637	-0,00601	-0,02453
-0,0029397	0,008503962	-0,00601369	0,007049	0,003284

-0,015449	0,009591138	-0,024527702	0,003284	0,135819
X4	X5	X6	X7	X8

Кореляційна матриця факторних ознак R

	Столбец 1	*Столбец 2*	*Столбец 3*	*Столбец 4*
Столбец 1	1	0,31835727	-0,06161	0,1835129
Столбец 2	0,31835727	1	0,166723	0,2169984
Столбец 3	-0,06161142	0,166722763	1	0,0999413
Столбец 4	0,183512877	0,216998446	0,099941	1
Столбец 5	0,753690358	0,345964044	0,020248	0,3653088
Столбец 6	0,276727230	-0,036817127	-0,02672	-0,136432
Столбец 7	0,439704044	0,063992219	0,090561	-0,100782
Столбец 8	0,327569210	-0,048131095	0,075683	0,1981753
	X1	X2	X3	X4

Продовження кореляційної матриці факторних ознак R

Столбец 5	*Столбец 6*	*Столбец 7*	*Столбец 8*
0,753690358	0,27672723	0,439704	0,327569
0,345964044	-0,036817127	0,063992	-0,04813
0,020248226	-0,026719455	0,090561	0,075683
0,365308801	-0,136431967	-0,10078	0,198175
1	0,306225931	0,486201	0,364366
0,306225931	1	0,631377	0,527649
0,486201157	0,631376931	1	0,330486
0,364366275	0,527648579	0,330486	1
X5	X6	X7	X8

Обернена кореляційна матриця Z=1/R

1 2 3 4

1	2,520565168	-0,300645696	0,26685	0,2572807
2	-0,3006457	1,284312076	-0,25109	-0,12566
3	0,266850063	0,251086901	-1,114567	-0,085592
4	0,257280689	0,125660166	-0,08559	-1,4417696
5	-1,71032107	-0,336743381	0,049228	-0,81649
6	0,188407423	-0,04544034	0,218761	0,3410975
7	-0,27685817	0,153279758	-0,30721	0,3512488
8	-0,29604444	0,300220944	-0,19873	-0,368132
	X1	X2	X3	X4

Продовження матриці Z=1/R

5	6	7	8
-1,710321069	0,188407423	-0,27686	-0,29604
-0,336743381	-0,04544034	0,15328	0,300221
0,049228404	0,218761335	-0,30721	-0,19873
-0,816489851	0,341097548	0,351249	-0,36813
-3,147773661	0,043164736	-0,75987	-0,17092
-0,043164736	2,221045801	-1,14019	-0,92744
-0,7598741	-1,140194061	2,197034	0,204115
-0,170916182	-0,927443161	0,204115	1,683603
X5	X6	X7	X8

Частинні коефіцієнти кореляції $r_{ij} = z_{ij}/\sqrt{(z_{ii}*z_{jj})}$

	1	2	3	4
1	1	-0,167097804	0,159208	0,1349616
2	-0,1670978	1	-0,20986	-0,092345
3	0,159208161	-0,20986294	1	-0,06752
4	0,134961625	-0,09234522	-0,06752	1
5	-0,60719325	-0,167479647	0,026282	-0,383267
6	0,079628821	-0,026904661	0,13904	0,1906127
7	-0,11764947	0,091249629	-0,19632	0,1973552
8	-0,14371038	0,204167165	-0,14507	-0,236285
	X1	X2	X3	X4

Продовження матриці

r ij = zij/√(zii*zjj)

-0,607193246	0,079628821	-0,11765	-0,14371
-0,167479647	-0,026904661	0,09125	0,204167
0,026282156	0,139039689	-0,19632	-0,14507
-0,383266724	0,190612735	0,197355	-0,23628
1	-0,016324827	-0,28895	-0,07424
-0,016324827	1	-0,51616	-0,47961
-0,288949188	-0,51615674	1	0,10613
-0,074244046	-0,479610565	0,10613	1
X5	X6	X7	X8

Кореляційна матриця результатів педагогічного експерименту $R(Y_{спотв.},X1,X2,X3,X4,X5,X6,X7,X8)$ (Yспотв.)

	Столбец 1	*Столбец 2*	*Столбец 3*	*Столбец 4*
Столбец 1	1			
Столбец 2	0,566005136	1		
Столбец 3	0,146649357	0,317324	1	
Столбец 4	0,045732457	-0,08112	0,164747 2	1
Столбец 5	-0,296765707	0,176532	0,215453 2	0,08146175
Столбец 6	0,26677499	0,752063	0,345032 8	0,004263239
Столбец 7	0,576167275	0,27477 9	-0,037852	0,035077255
Столбец 8	0,874113627	0,43711 6	0,062562 3	0,080557082
Столбец 9	0,368553147	0,32503 2	-0,049507	0,067288671

	Yспотв.	X1	X2	X3

Продовження кореляційної матриці
$R(Y_{спотв.},X1,X2,X3,X4,X5,X6,X7,X8)$

Столбец 5	*Столбец 6*	*Столбец 7*	*Столбец 8*	*Столбец 9*

1				
0,36039265	1			
-0,140922936	0,304486	1		
-0,107266457	0,483982	0,63073	1	
0,19455738	0,362131	0,526844	0,328639	1
X4	X5	X6	X7	X8

Кореляційна матриця результатів педагогічного експерименту R(X1,X2,X3,X4,X5,X6,X7,X8, $Y'_{\text{зрівн.}}$)

	Столбец 1	*Столбец 2*	*Столбец 3*	*Столбец 4*
Столбец 1	1			
Столбец 2	0,31835727	1		
Столбец 3	-0,061611418	0,166723	1	
Столбец 4	0,183512877	0,216998	0,0999413	1
Столбец 5	0,753690358	0,345964	0,0202482	0,365308801
Столбец 6	-0,27672723	-0,03682	-0,026719	-0,136431967
Столбец 7	0,439704044	0,063992	0,0905606	0,100781854
Столбец 8	0,32756921	-0,04813	0,0756825	0,198175279
Столбец 9	0,561456924	0,18371	0,0235668	-0,266855865
	X1	X2	X3	X4

Продовження кореляційної матриці R(X1,X2,X3,X4,X5,X6,X7,X8, $Y'_{\text{зрівн.}}$)

1				
0,306225931	1			
0,486201157	0,631377	1		
0,364366275	0,527649	0,330486	1	
0,25068415	0,609396	0,864651	0,386272	1
X5	X6	X7	X8	Y'зрівн.

Кореляційна матирця істинної моделі
R(X1,X2,X3,X4,X5,X6,X7,X8,X9,Yістн.)

	Столбец 1	*Столбец 2*	*Столбец 3*	*Столбец 4*
Столбец 1	1			
Столбец 2	#ДЕЛ/0!	1		
Столбец 3	#ДЕЛ/0!	0,3183573	1	
Столбец 4	#ДЕЛ/0!	-0,061611	0,166722763	1
Столбец 5	#ДЕЛ/0!	0,1835129	0,216998446	0,099941282
Столбец 6	#ДЕЛ/0!	0,7536904	0,345964044	0,020248226
Столбец 7	#ДЕЛ/0!	0,2767272	-0,036817127	-0,026719455
Столбец 8	#ДЕЛ/0!	0,439704	0,063992219	0,090560639
Столбец 9	#ДЕЛ/0!	0,3275692	-0,048131095	0,075682513
Столбец 10	#ДЕЛ/0!	0,5468201	0,174084146	0,046324245
		X1	X2	X3

Продовженя кореляційної матирці істинної моделі

Столбец 5	*Столбец 6*	*Столбец 7*	*Столбец 8*	*Столбец 9*	*Столбец 10*

1					
0,365309	1				
-0,13643	0,306226	1			
-0,10078	0,486201	0,631377	1		
0,198175	0,364366	0,527649	0,330486	1	
-0,27107	0,271415	0,596041	0,864581	0,357617	1
X4	X5	X6	X7	X8	Yістн.

Кореляційна матриця результатів екзамену

	Столбец 1	*Столбец 2*	*Столбец 3*	*Столбец 4*
Столбец 1	1			
Столбец 2	#ДЕЛ/0!	1		
Столбец 3	0,323592401	#ДЕЛ/0!	1	
Столбец 4	0,103017993	#ДЕЛ/0!	0,3183573	1
Столбец 5	0,027413357	#ДЕЛ/0!	-0,061611	0,166722763
Столбец 6	-0,160412227	#ДЕЛ/0!	0,1835129	0,216998446
Столбец 7	0,16061577	#ДЕЛ/0!	0,7536904	0,345964044
Столбец 8	0,352719734	#ДЕЛ/0!	0,2767272	0,036817127
Столбец 9	0,511634281	#ДЕЛ/0!	0,439704	0,063992219
Столбец 10	0,211627189	#ДЕЛ/0!	0,3275692	0,048131095
	Yекзам.	X0	X1	X2

Продовження кореляційної матриці результатів екзамену

Столбец 5	*Столбец 6*	*Столбец 7*	*Столбец 8*	*Столбец 9*	*Столбец 10*

1					
0,099941282	1				
0,020248226	0,365309	1			
-0,026719455	-0,13643	0,306226	1		
0,090560639	-0,10078	0,486201	0,631377	1	
0,075682513	0,198175	0,364366	0,527649	0,330486	1
X3	X4	X5	X6	X7	X8

Оберненими вагами встановлених нами коефіцієнтів математичної моделі будуть діагональні елементи оберненої матриці Q.

Середні квадратичні похибки коефіцієнтів розраховують за формулою

$$m_a = \mu\sqrt{Q_{I=J}}, \qquad (2.19)$$

Таблиця 6. Обернені ваги встановлених нами коефіцієнтів математичної моделі і їх середні квадратичні похибки

1/Pa	√(1/Pa)	ma	
37,60993	6,1326936	2,895444	a0
0,399089	0,6317353	0,298263	a1
1,319023	1,14848736	0,542238	a2
0,037681	0,19411615	0,091649	a3
0,141204	0,37577152	0,177414	a4
0,551223	0,74244396	0,350532	a5
0,086387	0,29391603	0,138767	a6
0,031624	0,17783132	0,08396	a7
0,609304	0,78057916	0,368537	a8

Значимість коефіцієнтів встановлюється за формулою

$$t_a = a / m_a , \qquad (2.20)$$

І в нашому випадку отримаємо Таблиця 7.

t=a/ma	
18,62688	
18,03737	Інтерес
9,482368	Роб.викл.
0,694451	Трудність
5,915502	Наук.пош.
18,55349	Зв'яз.спец
1,024309	Моногр.1
28,98166	Моногр.2
7,842755	Наук.школ
Значимість	

t(0,05;30)=	2,042272

Для коефіцієнтів регресії a0,a1,a2,a4,a5,a7,a8 t> t(0,05;30), тобто коефіцієнти регресії статистично значимі, а значить і сама математична модель адекватно описує якість засвоєння дисципліни.

Коефіцієнти a3, a6 незначимі і їх можна виключити з розгляду.

Згідно таблиці 4 коефіцієнт детермінації R^2 =0,9885511, тобто маємо дуже тісну кореляцію з моделлю.

За критерієм Фішера-Снедекора ми отримали

F0,05;8;29	2,278251
F=356,405	F>Fтабл.

Оскільки F>F(0.05;8;29), тобто (356,405>2,278) , то згідно критерію Фішера з надійністю P=0,95 математичну модель

$$Y_{моделі}' = 53.933095X_0 + 5.379875X_1 + 5.14170X_2 - \\ -0.063645X_3 - 1.049493X_4 - 6.503593X_5 - \\ 0.1142141X_6 + 2.433299X_7 + 2.890344X_8.$$

можна вважати адекватною експериментальним даним і на підставі прийнятої моделі можна проводити педагогічний аналіз.

Знайдемо значення оберненої ваги зрівноваженої функції 1/Ру' за Формулою

$$\frac{1}{P_\phi} = \phi Q \phi^T \quad . \qquad (2.21)$$

Для цього попередньо перемножим матриці

$$Q' = XN^{-1}, \qquad (2.22)$$

=МУМНОЖ(H2:P39;A68:I76) F2,Ctrl+Shift+Enter.

Допоміжна матриця Q'

0,5589	-0,2841	0,1696	0,0036	0,0004	-0,4619	-0,0133	0,0302	0,1049
-0,0562	0,0509	-0,0598	0,0401	0,0298	-0,0138	-0,0120	-0,0014	0,0294
-0,0562	0,0509	-0,0598	0,0401	0,0298	-0,0138	-0,0120	-0,0014	0,0294
-0,2520	0,0118	0,0337	-0,0351	0,0396	-0,0214	-0,0037	0,0120	0,0145
5	-1,53E-14	-1	3,05E-16	-1,1E-14	2,37E-14	-3,4E-15	-1,3E-15	1,24E-14
-0,3495	-0,0201	0,0736	-0,0302	0,1015	-0,0854	-0,0248	-0,0011	0,0838
-0,0562	0,0509	-0,0598	0,0401	0,0298	-0,0138	-0,0120	-0,0014	0,0294
-0,3499	-0,0076	0,08054	-0,072	0,0445	-0,0252	-0,0117	-0,0188	0,0365
-0,0562	0,0509	-0,0598	0,0401	0,0298	-0,0138	-0,0120	-0,0014	0,0294
-0,8020	-0,0168	0,1578	0,0130	0,0436	-0,0933	-0,1278	-0,1014	0,0328
	-0,0509	-0,0598	0,0401	0,0298		-0,0120		

0,0562					0,0138		0,0014	0,0294
	-					-	-	-
1,1387	0,0198	0,0202	-0,0046	0,0077	0,0325	0,1940	0,0187	0,0263
	-	-			-	-	-	
0,7665	0,1017	0,2722	-0,0091	0,0447	0,1390	0,0015	0,0118	0,1610
	-					-	-	
0,2520	0,0118	0,0337	-0,0351	0,0396	0,0214	0,0037	0,0120	0,0145
	-	-			-	-	-	
0,3490	0,0326	0,0666	0,0123	0,2476	0,1961	0,0379	0,0210	0,1311
	-	-			-	-	-	
0,2516	0,0006	0,0268	0,0074	0,1064	0,0892	0,0169	0,0078	0,0618
	-	-			-	-	-	
0,7332	0,1149	0,2908	-0,0158	0,0315	0,1771	0,0281	0,0434	0,1757
	-					-	-	
0,1658	0,0361	-0,0507	0,0322	0,0087	0,0103	0,0743	0,0255	0,0806
	-				-		-	
0,5278	0,0631	-0,0503	-0,0092	0,0473	0,1728	0,0447	0,1190	0,0509
	-					-	-	
0,2520	0,0118	0,0337	-0,0351	0,0396	0,0214	0,0037	0,0120	0,0145
	-					-	-	-
0,0562	0,0509	-0,0598	0,0401	0,0298	0,0138	0,0120	0,0014	0,0294
	-	-			-	-	-	
0,2516	0,0006	0,0268	0,0074	0,1064	0,0892	0,0169	0,0078	0,0618
	-					-	-	-
1,7221	0,0308	0,0892	0,0128	0,0245	0,0959	0,1088	0,0028	0,4482
	-					-	-	
0,2520	0,0118	0,0337	-0,0351	0,0396	0,0214	0,0037	0,0120	0,0145
	-					-	-	
0,2520	0,0118	0,0337	-0,0351	0,0396	0,0214	0,0037	0,0120	0,0145
	-	-			-	-	-	
0,2516	0,0006	0,0268	0,0074	0,1064	0,0892	0,0169	0,0078	0,0618
	-					-	-	
0,2521	0,0119	0,0338	-0,0352	0,0396	0,0215	0,0038	0,0121	0,0146
	-					-	-	-
0,0562	0,0509	-0,0598	0,0402	0,0298	0,0139	0,0121	0,0015	0,0294
	-	0,0119	0,0338	-0,0352	0,0396		-	- 0,0121	0,0146

0,2521					0,0215	0,0038		
-0,2305	-0,3482	0,0614	0,0207	0,0022	-0,2709	0,0027	0,0117	0,0415
-0,2521	0,0119	0,0338	-0,0352	0,0396	-0,0215	0,0038	0,0121	0,0146
-0,3285	-0,3677	0,1082	-0,0170	0,0027	-0,2671	0,0106	0,0185	0,0634
-0,0562	0,0509	-0,0598	0,0402	0,0298	-0,0139	0,0121	-0,0015	-0,0294
-0,2521	0,0119	0,0338	-0,0352	0,0396	-0,0215	0,0038	0,0121	0,0146
-0,2521	0,0119	0,0338	-0,0352	0,0396	-0,0215	0,0038	0,0121	0,0146
-0,0562	0,0509	-0,0598	0,0402	0,0298	-0,0139	0,0121	-0,0015	-0,0294
-2,1391	0,0507	-0,1094	-0,0083	0,0322	-0,1285	0,0852	-0,0158	-0,5254
-0,0562	0,0509	-0,0598	0,0402	0,0298	-0,0139	0,0121	-0,0015	-0,0294

Обернену вагу $1/P_{\varphi}$ знаходимо порядковим множенням =МУМНОЖ(W2:AE2;A46:A54) F2,Ctrl+Shift+Enter , де першою строчкою (W2:AE2) буде перша строчка матриці Q', стовпчиком (A46:A54) ,буде перший стовпчик транспонованої матриці X^T .

Таблиця 0.6. Обернені ваги зрівноваженої функції і середньоквадратичної похибки

1/Py'	√(1/Py')	m(y')
0,487704	0,698358	0,329718
0,08594	0,293155	0,138408
0,08594	0,293155	0,138408
0,075925	0,275546	0,130094
1	1	0,472133
0,487704	0,698358	0,329718

0,13783	0,371254	0,175281
0,08594	0,293155	0,138408
0,183961	0,428907	0,202501
0,08594	0,293155	0,138408
0,62038	0,787642	0,371871
0,08594	0,293155	0,138408
0,945212	0,97222	0,459017
0,263131	0,512963	0,242186
0,075925	0,275546	0,130094
0,469169	0,684959	0,323392
0,115006	0,339126	0,160112
0,318474	0,564335	0,266441
0,148166	0,384923	0,181735
0,569883	0,754906	0,356416
0,075925	0,275546	0,130094
0,08594	0,293155	0,138408
0,115006	0,339126	0,160112
0,550332	0,741844	0,350249
0,075925	0,275546	0,130094
0,075925	0,275546	0,130094
0,115006	0,339126	0,160112
0,075925	0,275546	0,130094
0,08594	0,293155	0,138408
0,075925	0,275546	0,130094
0,383167	0,619005	0,292252
0,075925	0,275546	0,130094
0,379526	0,616057	0,29086
0,08594	0,293155	0,138408
0,075925	0,275546	0,130094
0,075925	0,275546	0,130094
0,08594	0,293155	0,138408
0,579391	0,761177	0,359377
0,08594	0,293155	0,138408

Таблиця 0.7. Матриця коефіцієнтів нормальних рівнянь N

38	182	189	148	178	183	183	172	187
182	878	906	708	854	881	880	833	897
189	906	941	737	886	911	910	856	930
148	708	737	606	695	713	712	674	729
178	854	886	695	844	860	855	803	877
183	881	911	713	860	887	885	838	902
183	880	910	712	855	885	907	855	905
172	833	856	674	803	838	855	848	851
187	897	930	729	877	902	905	851	923

Таблиця 0.8. Обернена матриця Q=N-1

37,60993079	0,17429873	-5,864444295	0,097911
0,17429873	0,399089485	-0,121236141	0,019524
-5,864444295	-0,12123614	1,319023213	-0,04679
0,097910912	0,0195237	-0,046786832	0,037681
0,09744857	0,032038288	-0,03985332	-0,00493
0,307394507	-0,28479067	-0,142807802	0,003788
0,109580929	0,014785251	-0,009081906	0,007933
0,033235048	-0,01321701	0,018636552	-0,00678
-2,488659681	-0,07086638	0,183032862	-0,02198

Продовження матриці $Q=N^{-1}$

0,0974486	0,307394507	0,109580929	0,033235	-2,48866
0,0320383	-0,284790668	0,014785251	-0,01322	-0,07087
-0,039853	-0,142807802	-0,009081906	0,018637	0,183033
-0,004925	0,003787794	0,007932749	-0,00678	-0,02198
0,1412042	-0,106927314	0,021052276	0,013188	-0,06931
-0,106927	0,551223037	-0,003562341	-0,03815	-0,04303
0,0210523	-0,003562341	0,086386633	-0,02698	-0,11003
0,0131881	-0,0381499	-0,02697821	0,031624	0,014732
-0,069307	-0,043027119	-0,110034524	0,014732	0,609304

Таблиця 0.9. Транспонована матриця початкових рівнянь Ntp

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
5	5	5	5	4	5	5	5	5	4	5	4	4	5	5	5	4	5	5	5	5	5	4	5	5	5	5	5	5	4	5	4	5	5	5	5	5	5
5	5	5	5	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
4	5	5	3	3	3	5	2	5	4	5	4	4	3	4	4	4	5	3	3	5	4	4	3	3	4	3	5	3	5	3	4	5	3	3	5	3	5
4	5	5	5	4	4	5	5	5	5	5	5	4	5	3	4	4	5	5	5	5	4	4	5	5	4	5	5	5	5	5	5	5	5	5	4	5
4	5	5	5	4	5	5	5	5	4	5	4	4	5	5	5	4	5	5	5	5	5	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
5	5	5	5	5	5	5	5	5	5	5	0	5	5	5	5	5	4	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
5	5	5	5	4	5	5	5	5	0	5	0	4	5	5	5	5	5	5	0	5	5	5	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5
5	5	5	5	5	5	5	5	5	5	5	4	5	5	5	5	5	5	5	5	5	5	4	5	5	5	5	5	5	5	5	5	5	5	5	5	4	5

Розділ 3. Розробка програмного продукту

3.1. Постановка задачі

Задача полягає в наступному, написати програму для побудови множинної регресії.

Задання факторних і експериментальних ознак (кількість рівнянь), відбувається динамічно, тобто задається користувачем під час роботи. Для початку подивимось на математичну модель розв'язуваного об'єкта. Всі операції в ньому доступні для вирішення на мові програмування C++, також є можливість застосувати до них об'єктно-орієнтований підхід.

Отже, математичний апарат даного досліджуваного обєкта (в нашому випадку множинної регресійного інструменту) — це звичайний набір операцій із матрично-векторними засобами лінійної алгебри. Оскільки в мові C++ відсутні такі операції приступимо до їх написання в підрозділі 3.1.1.

3.2. Написання класу "матриця" та операцій для роботи з ними

Матриця — математичний об'єкт, записаний у вигляді прямокутної таблиці чисел (чи елементів кільця) і допускаючий операції (додавання, віднімання, множення та множення на скаляр). Зазвичай матриці представляються двовимірними (прямокутними) таблицями

матриця $\begin{bmatrix} 1 & 2 & 3 \\ 1 & 2 & 7 \\ 4 & 9 & 2 \\ 6 & 1 & 5 \end{bmatrix}$ є матрицею 4×3. Елемент A[2,3], або a2,3 дорівнює 7.

Додавання

Якщо дано дві матриці m-на-n A і B, можемо означити їх суму A + B як матрицю m-на-n, що утворюється додаванням відповідних елементів, себто,
(A + B)[i, j] = A[i, j] + B[i, j]. Наприклад,

$$\begin{bmatrix} 1 & 3 & 2 \\ 1 & 0 & 0 \\ 1 & 2 & 2 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 5 \\ 7 & 5 & 0 \\ 2 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1+0 & 3+0 & 2+5 \\ 1+7 & 0+5 & 0+0 \\ 1+2 & 2+1 & 2+1 \end{bmatrix} = \begin{bmatrix} 1 & 3 & 7 \\ 8 & 5 & 0 \\ 3 & 3 & 3 \end{bmatrix}$$

Множення на скаляр

Якщо дано матрицю A і число c, можемо означити множення на скаляр cA як (cA)[i, j] = cA[i, j]. Наприклад,

$$2\begin{bmatrix}1 & 8 & -3\\ 4 & -2 & 5\end{bmatrix} = \begin{bmatrix}2\times 1 & 2\times 8 & 2\times -3\\ 2\times 4 & 2\times -2 & 2\times 5\end{bmatrix} = \begin{bmatrix}2 & 16 & -6\\ 8 & -4 & 10\end{bmatrix}$$

З цими двома операціями множина M(m, n, R) усіх матриць m-на-n з дійсними елементами є дійсним векторним простором розмірності mn.

Множення матриць

Множення двох матриць має сенс лише тоді, коли число стовпчиків першої матриці дорівнює числу рядків другої матриці. Якщо A — матриця m-на-n (m рядків, n стовпчиків), а B — матриця n-на-p (n рядків, p стовпчиків), їх добуток AB є матрицею m-на-p (m рядків, p стовпчиків), що розраховується за формулою: (AB)[i, j] = A[i, 1] * B[1, j] + A[i, 2] * B[2, j] + ... + A[i, n] * B[n, j] для кожної пари i та j.

Наприклад,

$$\begin{bmatrix}1 & 0 & 2\\ -1 & 3 & 1\end{bmatrix}\times\begin{bmatrix}3 & 1\\ 2 & 1\\ 1 & 0\end{bmatrix} = \begin{bmatrix}(1\times 3+0\times 2+2\times 1) & (1\times 1+0\times 1+2\times 0)\\ (-1\times 3+3\times 2+1\times 1) & (-1\times 1+3\times 1+1\times 0)\end{bmatrix} = \begin{bmatrix}5 & 1\\ 4 & 2\end{bmatrix}$$

Це множення має такі властивості:
(AB)C = A(BC) для всіх матриць A розмірності k-на-m, B розмірності m-на-n і C розмірності n-на-p (асоціативність).
(A + B)C = AC + BC для всіх матриць A і B розмірності m-на-n і матриць C розмірності n-на-k (дистрибутивність).
C(A + B) = CA + CB для всіх матриць A і B розмірності m-на-n і матриць C розмірності k-на-m (дистрибутивність).

Зауваження: комутативність має місце не завжди: для добутку певних матриць A і B може бути AB ≠ BA.

Матриці називають антикомутативними, якщо AB = −BA. Такі матриці є дуже важливими в представленнях алгебр Лі та в представленнях алгебр Кліффорда.

Отже, з математичної точки зору що таке матриця і які операції над нею виконуються розібрались.

Програмно це буде виглядати так лістинг 3.1.

```
template <class T> class Matrix
{
private:
    T *ptr_adress_matrix;
    unsigned __int32 number_of_rows;
    unsigned __int32 number_of_columns;

    void Destroy();
    void Init(unsigned __int32, unsigned
__int32);
public:
    virtual ~Matrix();

    unsigned __int32 GetNumberOfRows() const;
    unsigned __int32 GetNumberOfColumns()
const;
    unsigned __int32
GetNumberOfMatrixElements() const;

    const T& operator ()(unsigned int id_r,
unsigned int id_c) const
    {
        return *(this->ptr_adress_matrix +
(this->GetNumberOfColumns() *
            ((id_r <= 0u) ? 0u :
                (id_r >= (this-
>GetNumberOfRows() - 1u)) ?
```

```
                    (this->GetNumberOfRows() -
1u) : id_r) +
            ((id_c <= 0u) ? 0u :
                (id_c >= (this-
>GetNumberOfColumns() - 1u)) ?
                    (this->GetNumberOfColumns()
- 1u) : id_c)));
    }

    T& operator ()(unsigned int id_r, unsigned
int id_c)
    {
        return *(this->ptr_adress_matrix +
(this->GetNumberOfColumns() *
            ((id_r <= 0u) ? 0u :
                (id_r >= (this-
>GetNumberOfRows() - 1u)) ?
                    (this->GetNumberOfRows() -
1u) : id_r) +
            ((id_c <= 0u) ? 0u :
                (id_c >= (this-
>GetNumberOfColumns() - 1u)) ?
                    (this->GetNumberOfColumns()
- 1u) : id_c)));
    }

    void ReadFromFile(const __int8 *);
    void WriteToFile(const __int8 *);

    const Matrix<T> &operator  =(const
Matrix<T> &);
    const Matrix<T> &operator +=(const
Matrix<T> &);
    const Matrix<T> &operator *=(const
Matrix<T> &);
    const Matrix<T> &operator *=(const T &);
```

```
    Matrix(const Matrix<T> &);
    Matrix(const __int8 *);
    Matrix(unsigned __int32 = 1u, unsigned
__int32 = 1u);
};
```

В даному прикладі оголошується клас для роботи з матрицями, це своєрідний структурований тип даних, а об'єкт який буде оголошуватись після визначення типу можна назвати зміною.

```
    Matrix<double> X1, X2;
```

Тут Matrix<double> це своєрідний структурований тип даних, X1, X2 — змінні вищезгаданого типу. З ними можна робити тільки ті операції котрі визначені

3.3. Опис та інтерфейс програми

У діалоговому вікні задається константа C, яка характеризує точність нормування генерованих похибок будуємої імітаційної моделі (в нашому випадку C=0,5) і рівень значимості , рівний 0,05 на приведеному Рис.5. головного вікна програми «Множинний регресійний аналіз». Програма створена таким чином, що дає можливість порівняти результати обчислень по програмі з контрольними обчисленнями в MS EXCEL.

Задається число результативних Y_i та факторних показників X_{ij}.

Заповнюється таблиця результатами педагогічного експерименту.

Натиском кнопки «Обчислення» проводиться обчислення за розробленою автором програмою.

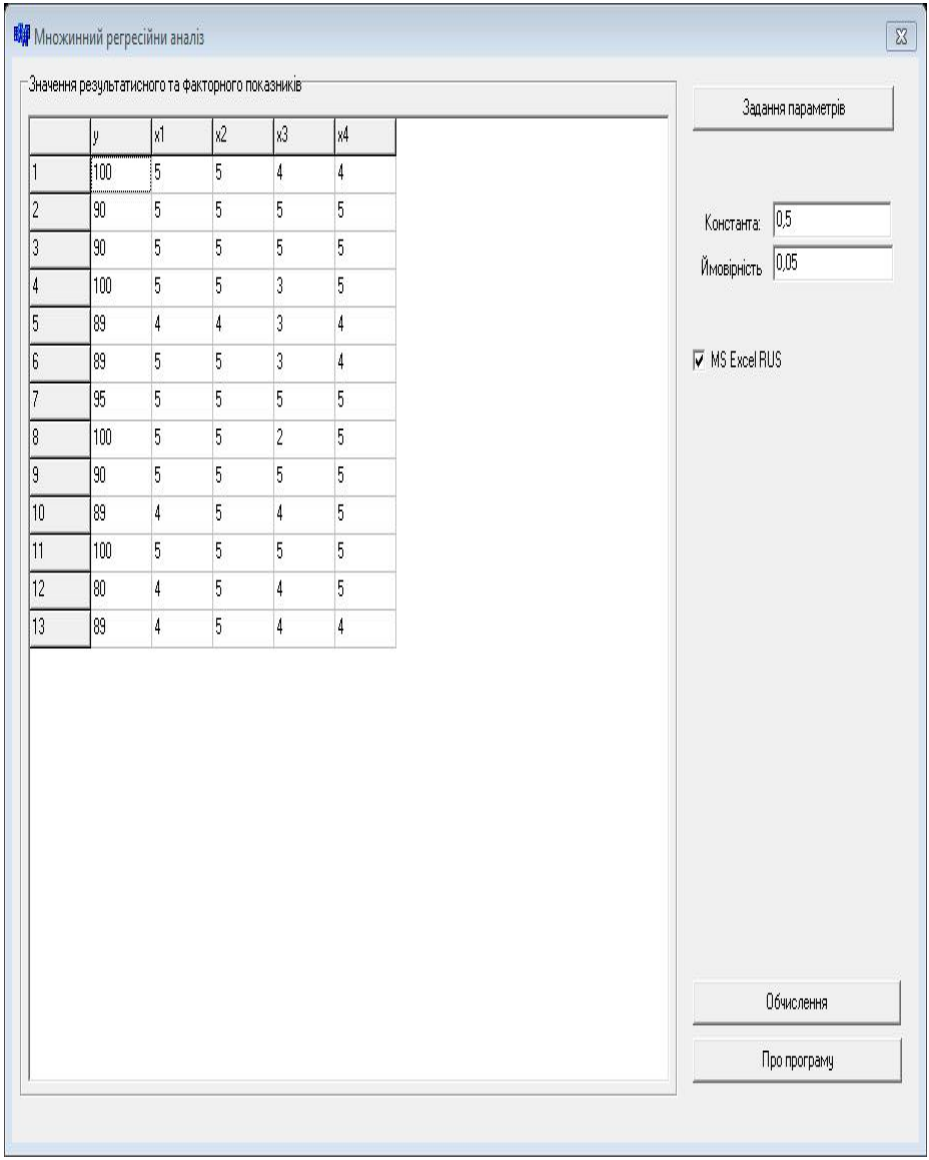
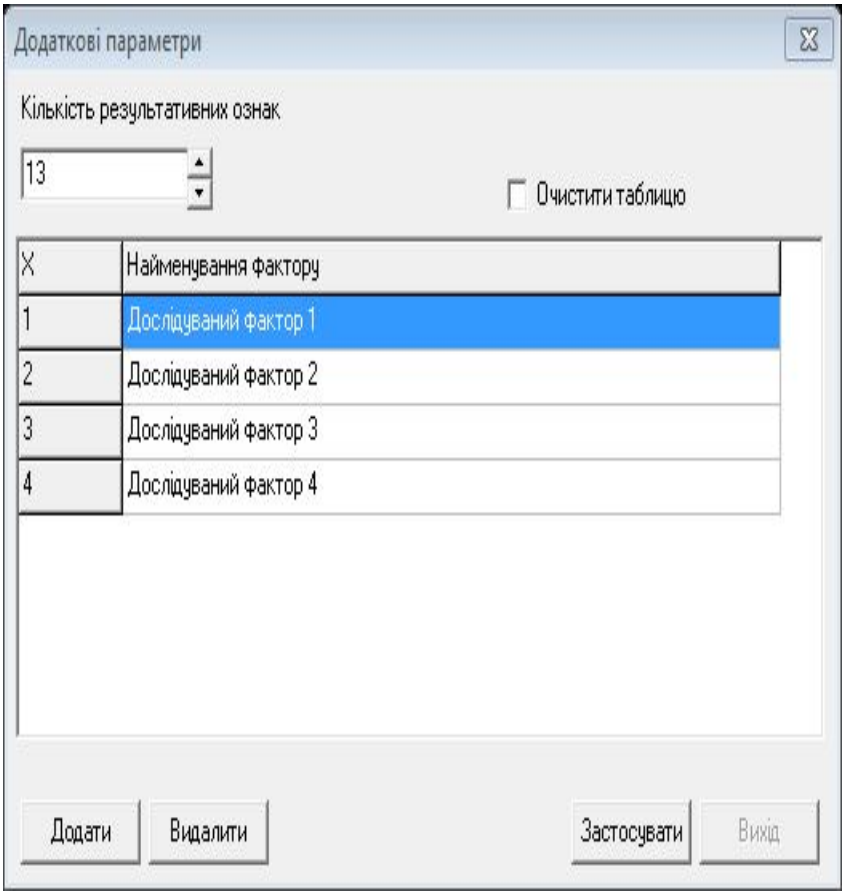


Рис. 5. Головне вікно програми "Множинний регресійний аналіз"

В програмі передбачається діалогове вікно «Додаткові параметри», що дає змогу змінювати кількість результативних ознак з метою проведення детальних досліджень, додаючи або видаляючи деякі ознаки. Результати фіксуються кнопкою «Застосувати». Приводяться найменування досліджуваних факторів. Передбачено виділення галочкою команди «Очистити таблицю».

Рис. 6. Діалогове вікно "Додаткові параметри "



В діалоговому вікні «Результати обчислень» приводиться коефіцієнт пропорційності К, необхідний при нормуванні істинних похибок для створення імітаційної моделі, яку в подальшому зрівноважують за способом найменших квадратів.

Даються повні характеристики нормування істинних похибок: середня квадратична похибка генерованих псевдо-випадкових чисел $m_{\Delta'}$,середня квадратична похибка істинних похибок m_{Δ} , яка повинна дорівнювати наперед заданій константі С, що і буде контролем обчислення істинних похибок .

Приводиться середня квадратична похибка одиниці ваги μ, яка знаходиться в результаті побудови математичної моделі.

Наводяться коефіцієнти істинної математичної моделі із результатів попереднього зрівноваження даних проведеного педагогічного експерименту (лівий стовпчик).

В правому стовпчику даються коефіцієнти зрівноваженої імітаційної моделі, яку отримали на основі введення в істинну модель істинних похибок з подальшим опрацюванням матеріалів за способом найменших квадратів.

Справа від таблиці коефіцієнтів зрівноваженої математичної моделі дається таблиця середніх квадратичних похибок зрівноваженої функції в ході виконання процедури строгого зрівноваження за способом найменших квадратів.

В нижньому ряду зліва приводяться середні квадратичні похибки зрівноважених коефіцієнтів імітованої математичної моделі, а в правому стовпчику дається статистична значимість знайдених коефіцієнтів.

Крім цього, видаються допустимі значення F-критерія Фішера і критерія Стьюдента.

Передбачена кнопка зведення результатів обчислень в окрему таблицю і кнопка побудови графіків.

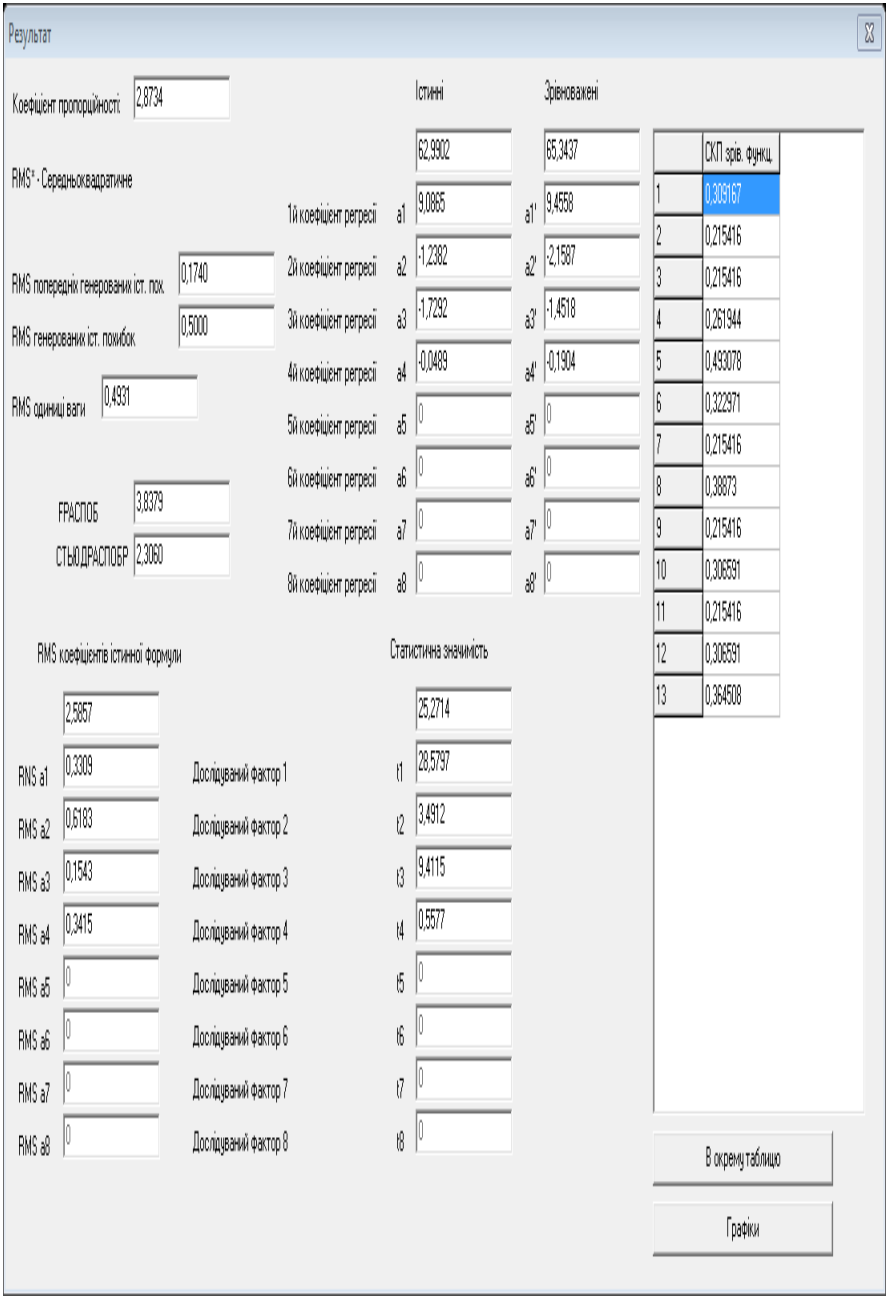
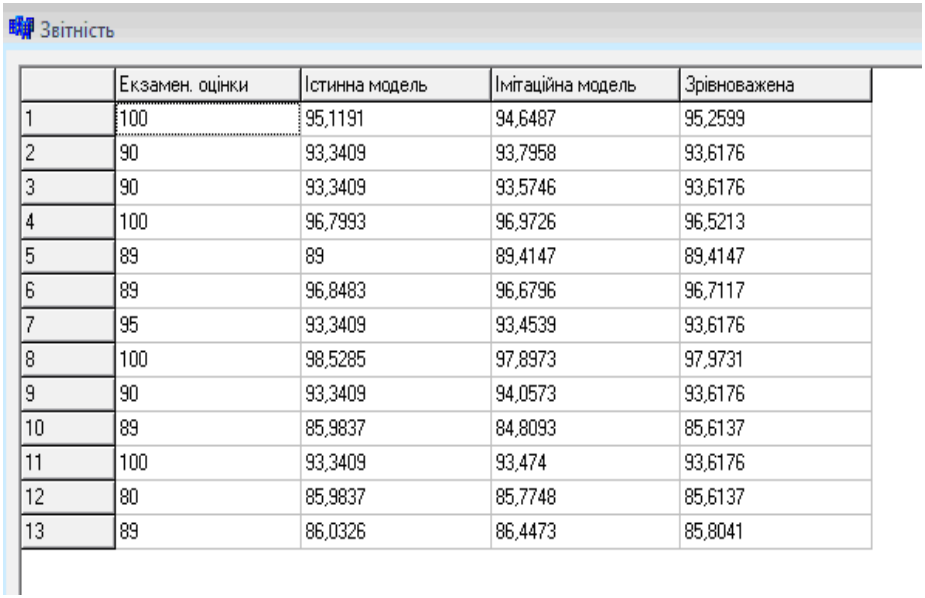


Рис. 7. Діалогове вікно "Результати обчислень"

В діалоговому вікні «Звітність» приводяться результуючі ознаки - екзаменаційні оцінки, виставлені викладачем за результатами екзамену Y, оцінки істинної моделі, виставлені комп'ютером $Y_{іст.}$, оцінки побудованої імітаційної моделі $Y_{імітац.}$, і оцінки зрівноваженої моделі Y'.


Звітність

	Екзамен. оцінки	Істинна модель	Імітаційна модель	Зрівноважена
1	100	95,1191	94,6487	95,2599
2	90	93,3409	93,7958	93,6176
3	90	93,3409	93,5746	93,6176
4	100	96,7993	96,9726	96,5213
5	89	89	89,4147	89,4147
6	89	96,8483	96,6796	96,7117
7	95	93,3409	93,4539	93,6176
8	100	98,5285	97,8973	97,9731
9	90	93,3409	94,0573	93,6176
10	89	85,9837	84,8093	85,6137
11	100	93,3409	93,474	93,6176
12	80	85,9837	85,7748	85,6137
13	89	86,0326	86,4473	85,8041

Рис. 8. Діалогове вікно "Звітність"

В діалоговому вікні «Графіки» приведені експериментальні значення (оцінки, виставлені викладачем) і оцінки, виставлені комп'ютером на основі опрацювання анкет-відповідей студентів після здачі екзамену.

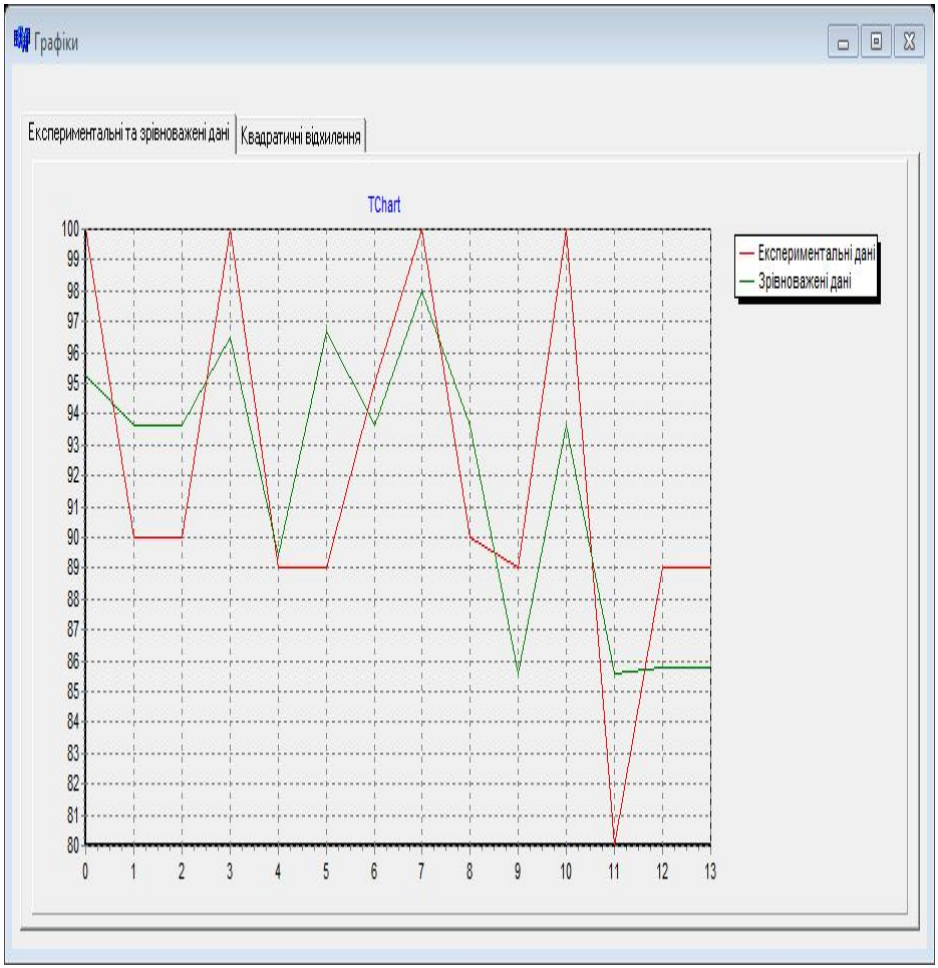


Рис. 9. Діалогове вікно "Графіки"

Висновки

Основні результати дослідження:

На основі проведених досліджень в даній роботі:

1. Генеровані випадкові числа, які приведено до нормованої досліджуваної точності.
2. На основі істинної моделі і генерованих істинних похибок побудована спотворена модель залежності екзаменаційних оцінок і факторних ознак результатів анкетування студентів, які отримали ту чи іншу оцінку.
3. Математична модель апроксимована по способу найменших квадратів поліномом першого степеня.
4. Отримана формула

$$Y_{моделі}' = 53.933095X_0 + 5.379875X_1 + 5.14170X_2 - \\ -0.063645X_3 - 1.049493X_4 - 6.503593X_5 - \\ 0.1142141X_6 + 2.433299X_7 + 2.890344X_8.$$

залежності екзаменаційних оцінок Y' і факторних ознак X_i.

5. Встановлено, що середня квадратична похибка одиниці ваги за результатами зрівноваження складає μ =0,472133 бала.

Середні квадратичні похибки виведених нами коефіцієнтів

2,895444	ma0
0,298263	ma1
0,542238	ma2
0,091649	ma3
0,177414	ma4
0,350532	ma5
0,138767	ma6
0,08396	ma7
0,368537	ma8

Статистична значимість встановлених нами коефіцієнтів

t=a/ma	
18,62688	
18,03737	Інтерес
9,482368	Роб.викл.
0,694451	Трудність
5,915502	Наук.пош.
18,55349	Зв'яз.спец
1,024309	Моногр.1
28,98166	Моногр.2
7,842755	Наук.школ

Наукова значимість дослідження:

6. Встановлені середні квадратичні похибки зрівноваженої функції m_{φ}.

7. Розроблена методика підготовки істинних похибок наперед заданої точності.

8. Дана робота відкриває дорогу для проведення досліджень методом статистичних випробувань Монте -Карло. Вона дає можливість охопити велику аудиторію, тому що генеруються похибки індивідуально і вони не повторюються в других моделях.

9. Результатом даної магістерської дисертації є розробка програмного продукту. Розроблена програма дає можливість виконати необхідні розрахунки, що виникають не тільки при побудові педагогіко-математичної моделі, а і взагалі при апроксимації функції методом множинної регресії. Програма відповідає вимогам простоти, зручності і дружності стосовно користувача, проста в освоєнні і не потребує спеціального навчання.

Літературні джерела

1. Максименко С.Д., Носенко Е.Л. Експериментальна психологія (дидактичний тезаурус). Навчальний посібник –К.: МАУП, **2004, -128 с.**
2. Літнарович Р.М. Теоретико-методологічні аспекти і базові принципи функціонування наукової школи в рамках професійної освіти. Монографія.МЕГУ, Рівне,- 383 с.
3. Літнарович Р.М. Побудова і дослідження істинної моделі якості засвоєння базової дисципліни. Апроксимація поліномом першого степеня.. МЕГУ, Рівне, 2009, –32с.
4. Літнарович Р.М. Основи математики. Дослідження результатів психолого-педагогічного експерименту експоненціальною функцією. Частина 4. МЕГУ, Рівне, 2006, – 17с.
5. Літнарович Р.М. Основи математики. Дослідження результатів психолого-педагогічного експерименту степеневою функцією. Частина 5. МЕГУ, Рівне, 2006, - 17с.
6. Літнарович Р.М. Дослідження точності апроксимації результатів психолого-педагогічного експерименту методом статистичних випробувань Монте Карло.Ч.1.МЕГУ, Рівне,2006,-45с.
7. Ермаков С. Метод Монте-Карло в вычислительной математике. Вводный курс // Невский Диалект, Бином. Лаборатория знаний, 2009 .- 192 с.
8. Соболь И.М. Метод Монте-Карло// Наука, 1978,- 64с.
9. Михайлов Г.А. Оптимизация весовых методов Монте-Карло // Наука, 1987. – 240с.
10. Сабельфельд К. К. Отв. ред. Г. А. Михайлов Методы Монте-Карло в краевых задачах //Новосибирск Наука Сиб. 1989.- 280 с.
11. Биндер К., Хеерман Д.В.Моделирование методом Монте-Карло в статистической физике: Введение // Наука. Физматлит, - 1995. – 144 с.

12. Б. Л. Грановский, С. М. Ермаков, Метод Монте-Карло // Итоги науки и техн. Сер. Теор. вероятн. Мат. стат. Теор. кибернет., - 1976, 59–108 с.
1. В.Ф.Ситник. Н.С.Орленко. Імітаційне моделювання: Навч.-метод. посібник для самост.вивч.дисц.-К.:КНЕУ, 1999.-208с.
2. П.Е.Данко. А.Г.Попов. Высшая математика в упражнениях и задачах. Ч.2. Изд.2-е. Учеб.пособие для втузов. М., Высшая школа», 1974.-464с.
3. В.А.Кудрявцев. Б.П.Демидович. Кратний курс высшей математики: Учебное пособие для вузов.-7-е изд., испр.- М.: Наука. Гл.ред.физ.-мат.лит., 1989.-656с.
4. Р.М.Літнарович. Основи наукових досліджень. Частина 1. Курс лекцій. МЕГУ, 2008.-75с.
5. Р.М.Літнарович. Алгебра матриць. Курс лекцій. МЕГУ,Рівне, 2007.-109 с.
6. Р.М.Літнарович. Конструювання і дослідження математичних моделей. Множинний аналіз.Частина 1. МЕГУ, Рівне, 2009.-127 с.
7. Р.М.Літнарович. Конструювання і дослідження математичних моделей. Поліноміальна апроксимація. Частина 2. МЕГУ, Рівне, 2009.-36 с.
8. Р.М.Літнарович Р.М. Конструювання і дослідження математичних моделей. Онтодидактика поліноміальної апроксимації. Частина 3. МЕГУ, Рівне, 2009.-32 с.
9. С.В. Глушаков, А.Л. Клєвцов. Програмування в середовищі Delphi 7.0. –Харків: Фоліо, 2003.- 528с.
10. В.Є. Гофман, А.Д. Хомоненко. Delphi 6.-2001.-1152с.

ДЖЕРЕЛА МЕРЕЖІ ІНТЕРНЕТ
24. http://www.piter-press.ru
25. http://www.riskglossary.com.monte-karlo
26. http://www.devoid.com.ua/
27. http://www.programmersclub.ru

Додатки. Програма побудови мотематичної моделі

```
//---------------------------------------------------------
// Model.cpp

#include <dos.h>
#include <vcl.h>
#pragma hdrstop

USEFORM("Unit1.cpp", Form1);
USEFORM("Unit2.cpp", Form2);
USEFORM("Unit3.cpp", Form3);
USEFORM("Unit5.cpp", Form5);
USEFORM("Unit6.cpp", Form6);
USEFORM("Unit4.cpp", Form4);
USEFORM("Unit7.cpp", Form7);
//USEFORM("Unit8.cpp", Form8);

#include "Unit8.h"

WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR, int)
{
    try
    {
        Application->Initialize();

        Form8 = new TForm8(static_cast<void *>(NULL));
        Form8->Image1->Picture-
>LoadFromFile("Loader.bmp");
        Form8->ShowModal();

        Application->CreateForm(__classid(TForm1),
&Form1);
                Application-
>CreateForm(__classid(TForm2), &Form2);
                Application-
>CreateForm(__classid(TForm3), &Form3);
                Application-
>CreateForm(__classid(TForm5), &Form5);
                Application-
>CreateForm(__classid(TForm6), &Form6);
                Application-
>CreateForm(__classid(TForm4), &Form4);
```

```
                Application-
>CreateForm(__classid(TForm7), &Form7);
//                Application-
>CreateForm(__classid(TForm8), &Form8);
                Application->Run();
    }
    catch (Exception &exception)
    {
        Application->ShowException(&exception);
    }
    catch (...)
    {
        try
        {
            throw Exception("");
        }
        catch (Exception &exception)
        {
            Application->ShowException(&exception);
        }
    }
    return 0;
}


//---------------------------------------------------
// Unit1.cpp
#include <vcl.h>
#pragma hdrstop

#include <fstream>

#include "Unit1.h"
#include "Unit2.h"
#include "Unit3.h"
#include "Unit5.h"

#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;

__fastcall TForm1::TForm1(TComponent* Owner)
    : TForm(Owner)
{
```

```
    try
    {
        ExcelApplication =
Variant::GetActiveObject("Excel.Application");
        flagConnSer = true;
    } catch(...) {
        try {
            ExcelApplication =
Variant::CreateObject("Excel.Application");
            flagConnSer = true;
        } catch(...) {
            Application->MessageBoxA("Íåìîæåòåî â³äêðèòè
Microsoft Excel!\n\
                ìîæëèâî Microsoft Excel íå âñòàíîâëåíèé
íà Âàøîìó êîìï'þòåð³.", "Ïîìèëêà", MB_OK + MB_ICONERROR);

            flagConnSer = false;
        }
    }
}

void __fastcall TForm1::FormActivate(TObject *Sender)
{
    std::ifstream Fin("data\\name.txt");
    if (Fin)
    {
        char Str[100];

        for (int i = 1; i < Form2->StringGrid1->RowCount;
i++)
        {
            Fin.getline(Str, sizeof(Str), '\n');
            AnsiString aStr = AnsiString(Str);
            Form2->StringGrid1->Cells[1][i] = aStr;
        }


        Fin.close();
    }
    for (int i = 1; i < StringGrid1->RowCount; i++)
        StringGrid1->Cells[0][i] = i;

    StringGrid1->Cells[1][0] = "y";
    for (int i = 2; i < StringGrid1->ColCount; i++)
```

```
        StringGrid1->Cells[i][0] = "x" + IntToStr(i - 1);


    std::ifstream fin[2];

    fin[0].open("Data\\Y1.txt");
    fin[1].open("Data\\X1.txt");

    if (fin[0] && fin[1])
    {
        double fCell;
        int RowCount, ColCount;
        fin[0] >> RowCount >> ColCount;
        fin[1] >> RowCount >> ColCount;

        StringGrid1->RowCount = RowCount + 1;
        StringGrid1->ColCount = ColCount + 2;
        for (int i = 1; i < StringGrid1->RowCount; i++)
        {
            fin[0] >> fCell;
            StringGrid1->Cells[0][i] = IntToStr(i);
            StringGrid1->Cells[1][i] = FloatToStr(fCell);

            for (int j = 2; j < StringGrid1->ColCount;
j++)
            {
                fin[1] >> fCell;
                StringGrid1->Cells[j][i] =
FloatToStr(fCell);
            }
        }

        fin[0].close();
        fin[1].close();
    }
}

void __fastcall TForm1::FormClose(TObject *Sender,
TCloseAction &Action)
{
    std::ofstream fout[2];
    fout[0].open("Data\\Y1.txt");
    fout[1].open("Data\\X1.txt");
```

```
    if (fout[0] && fout[1])
    {
        fout[0] << StringGrid1->RowCount - 1 << ' ' << 1
<< ' ' << std::endl;
        fout[1] << StringGrid1->RowCount - 1 << ' ' <<
StringGrid1->ColCount - 2 << ' ' << std::endl;

        for (int i = 1; i < StringGrid1->RowCount; i++)
        {
            fout[0] << (double)((StringGrid1->Cells[1][i]
!= "") ? StrToFloat(StringGrid1->Cells[1][i]) : 0) << '
';
            fout[0] << std::endl;

            for (int j = 2; j < StringGrid1->ColCount;
j++)
            {
                fout[1] << ((StringGrid1->Cells[j][i] !=
"") ? StrToFloat(StringGrid1->Cells[j][i]) : 0) << ' ';
            }
            fout[1] << std::endl;
        }


        fout[0].close();
        fout[1].close();
    }

    ExcelApplication.OlePropertySet("DisplayAlerts",
false);

    if (flagConnSer)
        ExcelApplication.OleProcedure("Quit");
}

void __fastcall TForm1::Button1Click(TObject *Sender)
{
    Form2->ShowModal();
}

void __fastcall TForm1::Button2Click(TObject *Sender)
{
    std::ofstream fout[2];
    fout[0].open("Data\\Y1.txt");
```

```
    fout[1].open("Data\\X1.txt");

    if (fout[0] && fout[1])
    {
        fout[0] << StringGrid1->RowCount - 1 << ' ' << 1
<< ' ' << std::endl;
        fout[1] << StringGrid1->RowCount - 1 << ' ' <<
StringGrid1->ColCount - 2 << ' ' << std::endl;

        for (int i = 1; i < StringGrid1->RowCount; i++)
        {
            fout[0] << ((StringGrid1->Cells[1][i] != "")
? (double)StrToFloat(StringGrid1->Cells[1][i]) :
(double)0) << ' ';
            fout[0] << std::endl;

            for (int j = 2; j < StringGrid1->ColCount;
j++)
            {
                fout[1] << ((StringGrid1->Cells[j][i] !=
"") ? (double)StrToFloat(StringGrid1->Cells[j][i]) :
(double)0) << ' ';
            }
            fout[1] << std::endl;
        }

        fout[0].close();
        fout[1].close();
    }

    Form3->ShowModal();
}

void __fastcall TForm1::Button3Click(TObject *Sender)
{
    Form5->ShowModal();
}

// ---------------------------------------------------
// Unit1.h

#ifndef Unit1H
#define Unit1H
```

```
#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
#include <Grids.hpp>

class TForm1 : public TForm
{
__published:
    TGroupBox *GroupBox1;
    TStringGrid *StringGrid1;
    TButton *Button1;
    TButton *Button2;
    TEdit *Edit1;
    TLabel *Label1;
    TButton *Button3;
    TCheckBox *CheckBox1;
    TEdit *Edit2;
    TLabel *Label2;
    void __fastcall FormActivate(TObject *Sender);
    void __fastcall Button1Click(TObject *Sender);
    void __fastcall FormClose(TObject *Sender,
TCloseAction &Action);
    void __fastcall Button2Click(TObject *Sender);
    void __fastcall Button3Click(TObject *Sender);
private:
public:
    bool flagConnSer;
    Variant ExcelApplication, WorkBook, WorkSheet;
    __fastcall TForm1(TComponent* Owner);

    Variant __fastcall TForm1::fromExcelValue(int Row,
int Column)
    {
        try {
            Variant Result, Cursor;

            Cursor = this-
>WorkSheet.OlePropertyGet("Cells", Row, Column);
            Result = Cursor.OlePropertyGet("Value");

            return Result;
        } catch(...) {;}
```

```
     return (Variant) 0;
    }
};

extern PACKAGE TForm1 *Form1;

#endif

// ---------------------------------------------------
// Unit2.cpp

#include <vcl.h>
#pragma hdrstop

#include <fstream>

#include "Unit1.h"
#include "Unit2.h"

#pragma package(smart_init)
#pragma resource "*.dfm"
TForm2 *Form2;

__fastcall TForm2::TForm2(TComponent* Owner)
    : TForm(Owner)
{
}

void __fastcall TForm2::FormActivate(TObject *Sender)
{
    for (int i = 1; i < StringGrid1->RowCount; i++)
        StringGrid1->Cells[1][i] = "Äîñë³äóâàííé ôàêòîð "
+ IntToStr(i);

    Button4->Enabled = false;
    Edit1->Text = IntToStr(Form1->StringGrid1->RowCount -
1);
    StringGrid1->RowCount = Form1->StringGrid1->ColCount
- 1;

    StringGrid1->Cells[0][0] = 'X';
    StringGrid1->Cells[1][0] = "Íàéìåíóâàííÿ ôàêòîðó";
    for (int i = 1; i < StringGrid1->RowCount; i++)
        StringGrid1->Cells[0][i] = IntToStr(i);
```

```
    if (StringGrid1->RowCount == 9) Button1->Enabled =
false;
    else
         Button1->Enabled = true;
    if (StringGrid1->RowCount == 2) Button2->Enabled =
false;
    else
         Button2->Enabled = true;

    UpDown1->Position = StrToInt(Edit1->Text);


    std::ifstream fin("data\\name.txt");
    if (fin)
    {
        char Str[100];

        for (int i = 1; i < StringGrid1->RowCount; i++)
        {
            fin.getline(Str, sizeof(Str), '\n');
            AnsiString aStr = AnsiString(Str);
            StringGrid1->Cells[1][i] = aStr;
        }


        fin.close();
    }
}

void __fastcall TForm2::Button1Click(TObject *Sender)
{
    if (StringGrid1->RowCount == 8) Button1->Enabled =
false;
    if (StringGrid1->RowCount >= 2) Button2->Enabled =
true;

    StringGrid1->RowCount++;
    StringGrid1->Cells[0][StringGrid1->RowCount - 1] =
IntToStr(StringGrid1->RowCount - 1);
}

void __fastcall TForm2::Button2Click(TObject *Sender)
{
```

```
    if (StringGrid1->RowCount == 3) Button2->Enabled =
false;
    if (StringGrid1->RowCount <= 9) Button1->Enabled =
true;

    StringGrid1->RowCount--;
    StringGrid1->Rows[StringGrid1->RowCount]->Clear();
}

void __fastcall TForm2::Button3Click(TObject *Sender)
{
    Button4->Enabled = true;
    Form1->StringGrid1->RowCount = StrToInt(Edit1->Text)
+ 1;
    Form1->StringGrid1->Cells[1][0] = "y";

    for (int i = 1; i < StrToInt(Edit1->Text) + 1; i++)
        Form1->StringGrid1->Cells[0][i] = i;

    Form1->StringGrid1->ColCount = StringGrid1->RowCount
+ 1;

    for (int i = 2; i < Form1->StringGrid1->ColCount;
i++)
        Form1->StringGrid1->Cells[i][0] = "x" +
IntToStr(i - 1);

    for (int i = Form1->StringGrid1->RowCount; i < 39;
i++)
        Form1->StringGrid1->Rows[i]->Clear();

    for (int i = Form1->StringGrid1->ColCount; i < 10;
i++)
        Form1->StringGrid1->Cols[i]->Clear();
}

void __fastcall TForm2::Button4Click(TObject *Sender)
{
    Close();
}

void __fastcall TForm2::FormClose(TObject *Sender,
TCloseAction &Action)
{
```

```
    if (CheckBox1->Checked)
    {
        for (int i = 1; i < Form1->StringGrid1->RowCount;
i++)
            Form1->StringGrid1->Rows[i]->Clear();

        for (int i = 1; i < StrToInt(Edit1->Text) + 1;
i++)
            Form1->StringGrid1->Cells[0][i] = i;
    }

    std::ofstream fout("data\\name.txt");
    if (fout)
    {
        //fout << (StringGrid1->RowCount - 1) <<
std::endl;

        for (int i = 1; i < StringGrid1->RowCount; i++)
            fout << StringGrid1->Cells[1][i].c_str() <<
std::endl;
        fout << std::endl;

        fout.close();
    }
}

void __fastcall TForm2::UpDown1Click(TObject *Sender,
TUDBtnType Button)
{
    Edit1->Text = Edit1->Text;
    if ((UpDown1->Position >= 0) && (UpDown1->Position <=
38))
        Edit1->Text = IntToStr(UpDown1->Position);

// ---------------------------------------------------
// Unit3.cpp

#include <vcl.h>
#pragma hdrstop

#include "Matrix.hpp"

#include "Unit1.h"
#include "Unit2.h"
```

```
#include "Unit3.h"
#include "Unit4.h"
#include "Unit6.h"
#include "Unit7.h"

#pragma package(smart_init)
#pragma resource "*.dfm"
TForm3 *Form3;

__fastcall TForm3::TForm3(TComponent* Owner)
    : TForm(Owner)
{
}

void __fastcall TForm3::FormActivate(TObject *Sender)
{
    for (int i = 51; i <= 58; i++)
        dynamic_cast<TLabel *>(FindComponent("Label" +
IntToStr(i)))->Caption = Form2->StringGrid1->Cells[1][i -
50];

    Form4->ProgressBar1->Position = int();
    Form4->ShowModal();

    Matrix<double> a1("data\\a1.txt"),
a2("data\\a2.txt"),
        rms("data\\rms.txt"), t("data\\t.txt"),
C("data\\c.txt");


    Form1->ExcelApplication.OlePropertySet("Visible",
false);
    Form1-
>ExcelApplication.OlePropertySet("SheetsInNewWorkbook",
1);
    Form1-
>ExcelApplication.OlePropertyGet("Workbooks").OleProcedur
e("Add");

    AnsiString F[2], T[2];

    F[0] = "=FÐÀÑÏ(" + Form1->Edit2->Text + ";" +
IntToStr((int)m) + ";" + IntToStr((int)n) + ")";
```

```
    F[1] = "=FINV(" + Form1->Edit2->Text + ";" +
IntToStr((int)m) + ";" + IntToStr((int)n) + ")";

    T[0] = "=ÑÒÜÞÄÐÀÑÏÎÁÐ(" + Form1->Edit2->Text + ";" +
IntToStr((int)n) + ")";
    T[1] = "=TINV(" + Form1->Edit2->Text + ";" +
IntToStr((int)n) + ")";

    Form1->WorkSheet = Form1-
>ExcelApplication.OlePropertyGet("Worksheets").OlePropert
yGet("Item", 1);

    if (Form1->CheckBox1->Checked)
    {
        Form1-
>WorkSheet.OlePropertyGet("Cells").OlePropertyGet("Item",
1, 1).OlePropertySet("Value", F[0].c_str());
        Form1-
>WorkSheet.OlePropertyGet("Cells").OlePropertyGet("Item",
1, 2).OlePropertySet("Value", T[0].c_str());
    }
    else
    {
        Form1-
>WorkSheet.OlePropertyGet("Cells").OlePropertyGet("Item",
1, 1).OlePropertySet("Value", F[1].c_str());
        Form1-
>WorkSheet.OlePropertyGet("Cells").OlePropertyGet("Item",
1, 2).OlePropertySet("Value", T[1].c_str());
    }

    Edit41->Text = FloatToStrF(StrToFloat(Form1-
>fromExcelValue(1, 1)), ffFixed, 10, 4);
    Edit42->Text = FloatToStrF(StrToFloat(Form1-
>fromExcelValue(1, 2)), ffFixed, 10, 4);

    StringGrid1->RowCount = C.GetNumberOfRows() + 1;

    StringGrid1->Cells[1][0] = "ÑÊÏ çð³â. ôóíêö.";
    for (int i = 1; i < StringGrid1->RowCount; i++)
        StringGrid1->Cells[0][i] = i;

    for (int i = 1; i <= StringGrid1->RowCount; i++)
        StringGrid1->Cells[1][i] = C(i-1, 0u);
```

```
    switch(a1.GetNumberOfRows() - 1u)
    {
        case 1u:
        {
            Edit1->Text = FloatToStrF(a1(0u, 0u),
ffFixed, 10, 4);
            Edit2->Text = FloatToStrF(a1(1u, 0u),
ffFixed, 10, 4);
            Edit3->Enabled = false;
            Edit4->Enabled = false;
            Edit5->Enabled = false;
            Edit6->Enabled = false;
            Edit7->Enabled = false;
            Edit8->Enabled = false;
            Edit9->Enabled = false;

            Edit10->Text = FloatToStrF(a2(0u, 0u),
ffFixed, 10, 4);
            Edit11->Text = FloatToStrF(a2(1u, 0u),
ffFixed, 10, 4);
            Edit12->Enabled = false;
            Edit13->Enabled = false;
            Edit14->Enabled = false;
            Edit15->Enabled = false;
            Edit16->Enabled = false;
            Edit17->Enabled = false;
            Edit18->Enabled = false;

            Edit23->Text = FloatToStrF(rms(0u, 0u),
ffFixed, 10, 4);
            Edit24->Text = FloatToStrF(rms(1u, 0u),
ffFixed, 10, 4);
            Edit25->Enabled = false;
            Edit26->Enabled = false;
            Edit27->Enabled = false;
            Edit28->Enabled = false;
            Edit29->Enabled = false;
            Edit30->Enabled = false;
            Edit31->Enabled = false;

            Edit32->Text = FloatToStrF(t(0u, 0u),
ffFixed, 10, 4);
```

```
            Edit33->Text = FloatToStrF(t(1u, 0u),
ffFixed, 10, 4);
            Edit34->Enabled = false;
            Edit35->Enabled = false;
            Edit36->Enabled = false;
            Edit37->Enabled = false;
            Edit38->Enabled = false;
            Edit39->Enabled = false;
            Edit40->Enabled = false;
        }

        break;
        case 2u:
        {
            Edit1->Text = FloatToStrF(a1(0u, 0u),
ffFixed, 10, 4);
            Edit2->Text = FloatToStrF(a1(1u, 0u),
ffFixed, 10, 4);
            Edit3->Text = FloatToStrF(a1(2u, 0u),
ffFixed, 10, 4);
            Edit4->Enabled = false;
            Edit5->Enabled = false;
            Edit6->Enabled = false;
            Edit7->Enabled = false;
            Edit8->Enabled = false;
            Edit9->Enabled = false;

            Edit10->Text = FloatToStrF(a2(0u, 0u),
ffFixed, 10, 4);
            Edit11->Text = FloatToStrF(a2(1u, 0u),
ffFixed, 10, 4);
            Edit12->Text = FloatToStrF(a2(2u, 0u),
ffFixed, 10, 4);
            Edit13->Enabled = false;
            Edit14->Enabled = false;
            Edit15->Enabled = false;
            Edit16->Enabled = false;
            Edit17->Enabled = false;
            Edit18->Enabled = false;

            Edit23->Text = FloatToStrF(rms(0u, 0u),
ffFixed, 10, 4);
            Edit24->Text = FloatToStrF(rms(1u, 0u),
ffFixed, 10, 4);
```

```
            Edit25->Text = FloatToStrF(rms(2u, 0u),
ffFixed, 10, 4);
            Edit26->Enabled = false;
            Edit27->Enabled = false;
            Edit28->Enabled = false;
            Edit29->Enabled = false;
            Edit30->Enabled = false;
            Edit31->Enabled = false;

            Edit32->Text = FloatToStrF(t(0u, 0u),
ffFixed, 10, 4);
            Edit33->Text = FloatToStrF(t(1u, 0u),
ffFixed, 10, 4);
            Edit34->Text = FloatToStrF(t(2u, 0u),
ffFixed, 10, 4);
            Edit35->Enabled = false;
            Edit36->Enabled = false;
            Edit37->Enabled = false;
            Edit38->Enabled = false;
            Edit39->Enabled = false;
            Edit40->Enabled = false;
        }

        break;
        case 3u:
        {
            Edit1->Text = FloatToStrF(a1(0u, 0u),
ffFixed, 10, 4);
            Edit2->Text = FloatToStrF(a1(1u, 0u),
ffFixed, 10, 4);
            Edit3->Text = FloatToStrF(a1(2u, 0u),
ffFixed, 10, 4);
            Edit4->Text = FloatToStrF(a1(3u, 0u),
ffFixed, 10, 4);
            Edit5->Enabled = false;
            Edit6->Enabled = false;
            Edit7->Enabled = false;
            Edit8->Enabled = false;
            Edit9->Enabled = false;

            Edit10->Text = FloatToStrF(a2(0u, 0u),
ffFixed, 10, 4);
            Edit11->Text = FloatToStrF(a2(1u, 0u),
ffFixed, 10, 4);
```

```
            Edit12->Text = FloatToStrF(a2(2u, 0u),
ffFixed, 10, 4);
            Edit13->Text = FloatToStrF(a2(3u, 0u),
ffFixed, 10, 4);
            Edit14->Enabled = false;
            Edit15->Enabled = false;
            Edit16->Enabled = false;
            Edit17->Enabled = false;
            Edit18->Enabled = false;

            Edit23->Text = FloatToStrF(rms(0u, 0u),
ffFixed, 10, 4);
            Edit24->Text = FloatToStrF(rms(1u, 0u),
ffFixed, 10, 4);
            Edit25->Text = FloatToStrF(rms(2u, 0u),
ffFixed, 10, 4);
            Edit26->Text = FloatToStrF(rms(3u, 0u),
ffFixed, 10, 4);
            Edit27->Enabled = false;
            Edit28->Enabled = false;
            Edit29->Enabled = false;
            Edit30->Enabled = false;
            Edit31->Enabled = false;

            Edit32->Text = FloatToStrF(t(0u, 0u),
ffFixed, 10, 4);
            Edit33->Text = FloatToStrF(t(1u, 0u),
ffFixed, 10, 4);
            Edit34->Text = FloatToStrF(t(2u, 0u),
ffFixed, 10, 4);
            Edit35->Text = FloatToStrF(t(3u, 0u),
ffFixed, 10, 4);
            Edit36->Enabled = false;
            Edit37->Enabled = false;
            Edit38->Enabled = false;
            Edit39->Enabled = false;
            Edit40->Enabled = false;
        }

        break;
        case 4u:
        {
            Edit1->Text = FloatToStrF(a1(0u, 0u),
ffFixed, 10, 4);
```

```
            Edit2->Text = FloatToStrF(a1(1u, 0u),
ffFixed, 10, 4);
            Edit3->Text = FloatToStrF(a1(2u, 0u),
ffFixed, 10, 4);
            Edit4->Text = FloatToStrF(a1(3u, 0u),
ffFixed, 10, 4);
            Edit5->Text = FloatToStrF(a1(4u, 0u),
ffFixed, 10, 4);
            Edit6->Enabled = false;
            Edit7->Enabled = false;
            Edit8->Enabled = false;
            Edit9->Enabled = false;

            Edit10->Text = FloatToStrF(a2(0u, 0u),
ffFixed, 10, 4);
            Edit11->Text = FloatToStrF(a2(1u, 0u),
ffFixed, 10, 4);
            Edit12->Text = FloatToStrF(a2(2u, 0u),
ffFixed, 10, 4);
            Edit13->Text = FloatToStrF(a2(3u, 0u),
ffFixed, 10, 4);
            Edit14->Text = FloatToStrF(a2(4u, 0u),
ffFixed, 10, 4);
            Edit15->Enabled = false;
            Edit16->Enabled = false;
            Edit17->Enabled = false;
            Edit18->Enabled = false;

            Edit23->Text = FloatToStrF(rms(0u, 0u),
ffFixed, 10, 4);
            Edit24->Text = FloatToStrF(rms(1u, 0u),
ffFixed, 10, 4);
            Edit25->Text = FloatToStrF(rms(2u, 0u),
ffFixed, 10, 4);
            Edit26->Text = FloatToStrF(rms(3u, 0u),
ffFixed, 10, 4);
            Edit27->Text = FloatToStrF(rms(4u, 0u),
ffFixed, 10, 4);
            Edit28->Enabled = false;
            Edit29->Enabled = false;
            Edit30->Enabled = false;
            Edit31->Enabled = false;
```

```
            Edit32->Text = FloatToStrF(t(0u, 0u),
ffFixed, 10, 4);
            Edit33->Text = FloatToStrF(t(1u, 0u),
ffFixed, 10, 4);
            Edit34->Text = FloatToStrF(t(2u, 0u),
ffFixed, 10, 4);
            Edit35->Text = FloatToStrF(t(3u, 0u),
ffFixed, 10, 4);
            Edit36->Text = FloatToStrF(t(4u, 0u),
ffFixed, 10, 4);
            Edit37->Enabled = false;
            Edit38->Enabled = false;
            Edit39->Enabled = false;
            Edit40->Enabled = false;
        }

        break;
        case 5u:
        {
            Edit1->Text = FloatToStrF(a1(0u, 0u),
ffFixed, 10, 4);
            Edit2->Text = FloatToStrF(a1(1u, 0u),
ffFixed, 10, 4);
            Edit3->Text = FloatToStrF(a1(2u, 0u),
ffFixed, 10, 4);
            Edit4->Text = FloatToStrF(a1(3u, 0u),
ffFixed, 10, 4);
            Edit5->Text = FloatToStrF(a1(4u, 0u),
ffFixed, 10, 4);
            Edit6->Text = FloatToStrF(a1(5u, 0u),
ffFixed, 10, 4);
            Edit7->Enabled = false;
            Edit8->Enabled = false;
            Edit9->Enabled = false;

            Edit10->Text = FloatToStrF(a2(0u, 0u),
ffFixed, 10, 4);
            Edit11->Text = FloatToStrF(a2(1u, 0u),
ffFixed, 10, 4);
            Edit12->Text = FloatToStrF(a2(2u, 0u),
ffFixed, 10, 4);
            Edit13->Text = FloatToStrF(a2(3u, 0u),
ffFixed, 10, 4);
```

```
            Edit14->Text = FloatToStrF(a2(4u, 0u),
ffFixed, 10, 4);
            Edit15->Text = FloatToStrF(a2(5u, 0u),
ffFixed, 10, 4);
            Edit16->Enabled = false;
            Edit17->Enabled = false;
            Edit18->Enabled = false;

            Edit23->Text = FloatToStrF(rms(0u, 0u),
ffFixed, 10, 4);
            Edit24->Text = FloatToStrF(rms(1u, 0u),
ffFixed, 10, 4);
            Edit25->Text = FloatToStrF(rms(2u, 0u),
ffFixed, 10, 4);
            Edit26->Text = FloatToStrF(rms(3u, 0u),
ffFixed, 10, 4);
            Edit27->Text = FloatToStrF(rms(4u, 0u),
ffFixed, 10, 4);
            Edit28->Text = FloatToStrF(rms(5u, 0u),
ffFixed, 10, 4);
            Edit29->Enabled = false;
            Edit30->Enabled = false;
            Edit31->Enabled = false;

            Edit32->Text = FloatToStrF(t(0u, 0u),
ffFixed, 10, 4);
            Edit33->Text = FloatToStrF(t(1u, 0u),
ffFixed, 10, 4);
            Edit34->Text = FloatToStrF(t(2u, 0u),
ffFixed, 10, 4);
            Edit35->Text = FloatToStrF(t(3u, 0u),
ffFixed, 10, 4);
            Edit36->Text = FloatToStrF(t(4u, 0u),
ffFixed, 10, 4);
            Edit37->Text = FloatToStrF(t(5u, 0u),
ffFixed, 10, 4);
            Edit38->Enabled = false;
            Edit39->Enabled = false;
            Edit40->Enabled = false;
        }

        break;
        case 6u:
        {
```

```
            Edit1->Text = FloatToStrF(a1(0u, 0u),
ffFixed, 10, 4);
            Edit2->Text = FloatToStrF(a1(1u, 0u),
ffFixed, 10, 4);
            Edit3->Text = FloatToStrF(a1(2u, 0u),
ffFixed, 10, 4);
            Edit4->Text = FloatToStrF(a1(3u, 0u),
ffFixed, 10, 4);
            Edit5->Text = FloatToStrF(a1(4u, 0u),
ffFixed, 10, 4);
            Edit6->Text = FloatToStrF(a1(5u, 0u),
ffFixed, 10, 4);
            Edit7->Text = FloatToStrF(a1(6u, 0u),
ffFixed, 10, 4);
            Edit8->Enabled = false;
            Edit9->Enabled = false;


            Edit10->Text = FloatToStrF(a2(0u, 0u),
ffFixed, 10, 4);
            Edit11->Text = FloatToStrF(a2(1u, 0u),
ffFixed, 10, 4);
            Edit12->Text = FloatToStrF(a2(2u, 0u),
ffFixed, 10, 4);
            Edit13->Text = FloatToStrF(a2(3u, 0u),
ffFixed, 10, 4);
            Edit14->Text = FloatToStrF(a2(4u, 0u),
ffFixed, 10, 4);
            Edit15->Text = FloatToStrF(a2(5u, 0u),
ffFixed, 10, 4);
            Edit16->Text = FloatToStrF(a2(6u, 0u),
ffFixed, 10, 4);
            Edit17->Enabled = false;
            Edit18->Enabled = false;

            Edit23->Text = FloatToStrF(rms(0u, 0u),
ffFixed, 10, 4);
            Edit24->Text = FloatToStrF(rms(1u, 0u),
ffFixed, 10, 4);
            Edit25->Text = FloatToStrF(rms(2u, 0u),
ffFixed, 10, 4);
            Edit26->Text = FloatToStrF(rms(3u, 0u),
ffFixed, 10, 4);
```

```
            Edit27->Text = FloatToStrF(rms(4u, 0u),
ffFixed, 10, 4);
            Edit28->Text = FloatToStrF(rms(5u, 0u),
ffFixed, 10, 4);
            Edit29->Text = FloatToStrF(rms(6u, 0u),
ffFixed, 10, 4);
            Edit30->Enabled = false;
            Edit31->Enabled = false;

            Edit32->Text = FloatToStrF(t(0u, 0u),
ffFixed, 10, 4);
            Edit33->Text = FloatToStrF(t(1u, 0u),
ffFixed, 10, 4);
            Edit34->Text = FloatToStrF(t(2u, 0u),
ffFixed, 10, 4);
            Edit35->Text = FloatToStrF(t(3u, 0u),
ffFixed, 10, 4);
            Edit36->Text = FloatToStrF(t(4u, 0u),
ffFixed, 10, 4);
            Edit37->Text = FloatToStrF(t(5u, 0u),
ffFixed, 10, 4);
            Edit38->Text = FloatToStrF(t(6u, 0u),
ffFixed, 10, 4);
            Edit39->Enabled = false;
            Edit40->Enabled = false;
        }

        break;
        case 7u:
        {
            Edit1->Text = FloatToStrF(a1(0u, 0u),
ffFixed, 10, 4);
            Edit2->Text = FloatToStrF(a1(1u, 0u),
ffFixed, 10, 4);
            Edit3->Text = FloatToStrF(a1(2u, 0u),
ffFixed, 10, 4);
            Edit4->Text = FloatToStrF(a1(3u, 0u),
ffFixed, 10, 4);
            Edit5->Text = FloatToStrF(a1(4u, 0u),
ffFixed, 10, 4);
            Edit6->Text = FloatToStrF(a1(5u, 0u),
ffFixed, 10, 4);
            Edit7->Text = FloatToStrF(a1(6u, 0u),
ffFixed, 10, 4);
```

```
            Edit8->Text = FloatToStrF(a1(7u, 0u),
ffFixed, 10, 4);
            Edit9->Enabled = false;


            Edit10->Text = FloatToStrF(a2(0u, 0u),
ffFixed, 10, 4);
            Edit11->Text = FloatToStrF(a2(1u, 0u),
ffFixed, 10, 4);
            Edit12->Text = FloatToStrF(a2(2u, 0u),
ffFixed, 10, 4);
            Edit13->Text = FloatToStrF(a2(3u, 0u),
ffFixed, 10, 4);
            Edit14->Text = FloatToStrF(a2(4u, 0u),
ffFixed, 10, 4);
            Edit15->Text = FloatToStrF(a2(5u, 0u),
ffFixed, 10, 4);
            Edit16->Text = FloatToStrF(a2(6u, 0u),
ffFixed, 10, 4);
            Edit17->Text = FloatToStrF(a2(7u, 0u),
ffFixed, 10, 4);
            Edit18->Enabled = false;

            Edit23->Text = FloatToStrF(rms(0u, 0u),
ffFixed, 10, 4);
            Edit24->Text = FloatToStrF(rms(1u, 0u),
ffFixed, 10, 4);
            Edit25->Text = FloatToStrF(rms(2u, 0u),
ffFixed, 10, 4);
            Edit26->Text = FloatToStrF(rms(3u, 0u),
ffFixed, 10, 4);
            Edit27->Text = FloatToStrF(rms(4u, 0u),
ffFixed, 10, 4);
            Edit28->Text = FloatToStrF(rms(5u, 0u),
ffFixed, 10, 4);
            Edit29->Text = FloatToStrF(rms(6u, 0u),
ffFixed, 10, 4);
            Edit30->Text = FloatToStrF(rms(7u, 0u),
ffFixed, 10, 4);
            Edit31->Enabled = false;

            Edit32->Text = FloatToStrF(t(0u, 0u),
ffFixed, 10, 4);
```

```
            Edit33->Text = FloatToStrF(t(1u, 0u),
ffFixed, 10, 4);
            Edit34->Text = FloatToStrF(t(2u, 0u),
ffFixed, 10, 4);
            Edit35->Text = FloatToStrF(t(3u, 0u),
ffFixed, 10, 4);
            Edit36->Text = FloatToStrF(t(4u, 0u),
ffFixed, 10, 4);
            Edit37->Text = FloatToStrF(t(5u, 0u),
ffFixed, 10, 4);
            Edit38->Text = FloatToStrF(t(6u, 0u),
ffFixed, 10, 4);
            Edit39->Text = FloatToStrF(t(7u, 0u),
ffFixed, 10, 4);
            Edit40->Enabled = false;
        }

        break;
        case 8u:
        {
            Edit1->Text = FloatToStrF(a1(0u, 0u),
ffFixed, 10, 4);
            Edit2->Text = FloatToStrF(a1(1u, 0u),
ffFixed, 10, 4);
            Edit3->Text = FloatToStrF(a1(2u, 0u),
ffFixed, 10, 4);
            Edit4->Text = FloatToStrF(a1(3u, 0u),
ffFixed, 10, 4);
            Edit5->Text = FloatToStrF(a1(4u, 0u),
ffFixed, 10, 4);
            Edit6->Text = FloatToStrF(a1(5u, 0u),
ffFixed, 10, 4);
            Edit7->Text = FloatToStrF(a1(6u, 0u),
ffFixed, 10, 4);
            Edit8->Text = FloatToStrF(a1(7u, 0u),
ffFixed, 10, 4);
            Edit9->Text = FloatToStrF(a1(8u, 0u),
ffFixed, 10, 4);

            Edit10->Text = FloatToStrF(a2(0u, 0u),
ffFixed, 10, 4);
            Edit11->Text = FloatToStrF(a2(1u, 0u),
ffFixed, 10, 4);
```

```
            Edit12->Text = FloatToStrF(a2(2u, 0u),
ffFixed, 10, 4);
            Edit13->Text = FloatToStrF(a2(3u, 0u),
ffFixed, 10, 4);
            Edit14->Text = FloatToStrF(a2(4u, 0u),
ffFixed, 10, 4);
            Edit15->Text = FloatToStrF(a2(5u, 0u),
ffFixed, 10, 4);
            Edit16->Text = FloatToStrF(a2(6u, 0u),
ffFixed, 10, 4);
            Edit17->Text = FloatToStrF(a2(7u, 0u),
ffFixed, 10, 4);
            Edit18->Text = FloatToStrF(a2(8u, 0u),
ffFixed, 10, 4);

            Edit23->Text = FloatToStrF(rms(0u, 0u),
ffFixed, 10, 4);
            Edit24->Text = FloatToStrF(rms(1u, 0u),
ffFixed, 10, 4);
            Edit25->Text = FloatToStrF(rms(2u, 0u),
ffFixed, 10, 4);
            Edit26->Text = FloatToStrF(rms(3u, 0u),
ffFixed, 10, 4);
            Edit27->Text = FloatToStrF(rms(4u, 0u),
ffFixed, 10, 4);
            Edit28->Text = FloatToStrF(rms(5u, 0u),
ffFixed, 10, 4);
            Edit29->Text = FloatToStrF(rms(6u, 0u),
ffFixed, 10, 4);
            Edit30->Text = FloatToStrF(rms(7u, 0u),
ffFixed, 10, 4);
            Edit31->Text = FloatToStrF(rms(8u, 0u),
ffFixed, 10, 4);

            Edit32->Text = FloatToStrF(t(0u, 0u),
ffFixed, 10, 4);
            Edit33->Text = FloatToStrF(t(1u, 0u),
ffFixed, 10, 4);
            Edit34->Text = FloatToStrF(t(2u, 0u),
ffFixed, 10, 4);
            Edit35->Text = FloatToStrF(t(3u, 0u),
ffFixed, 10, 4);
            Edit36->Text = FloatToStrF(t(4u, 0u),
ffFixed, 10, 4);
```

```
            Edit37->Text = FloatToStrF(t(5u, 0u),
ffFixed, 10, 4);
            Edit38->Text = FloatToStrF(t(6u, 0u),
ffFixed, 10, 4);
            Edit39->Text = FloatToStrF(t(7u, 0u),
ffFixed, 10, 4);
            Edit40->Text = FloatToStrF(t(8u, 0u),
ffFixed, 10, 4);
        }

        break;
    }

    Matrix<double> Delta1("data\\Delta1.txt"),
Delta2("data\\Delta2.txt");
    Edit19->Text = FloatToStrF(Form4->K, ffFixed, 10, 4);
    Edit20->Text = FloatToStrF(RMS(Delta1), ffFixed, 10,
4);
    Edit21->Text = FloatToStrF(RMS(Delta2), ffFixed, 10,
4);
    Edit22->Text = FloatToStrF(Form4->mu, ffFixed, 10,
4);
}

void __fastcall TForm3::FormClose(TObject *Sender,
TCloseAction &Action)
{
    for (int i = 1; i <= 40; i++)
    {
        dynamic_cast<TEdit *>(FindComponent("Edit" +
IntToStr(i)))->Text = IntToStr(0);
        dynamic_cast<TEdit *>(FindComponent("Edit" +
IntToStr(i)))->Enabled = true;
    }

    for (int i = 0; i < StringGrid1->RowCount; i++)
        StringGrid1->Rows[i]->Clear();

    DeleteFile((ExtractFilePath(Application->ExeName) +
"Data\\a1.txt").c_str());
    DeleteFile((ExtractFilePath(Application->ExeName) +
"Data\\a2.txt").c_str());
    DeleteFile((ExtractFilePath(Application->ExeName) +
"Data\\Y2.txt").c_str());
```

```
    DeleteFile((ExtractFilePath(Application->ExeName) +
"Data\\Y3.txt").c_str());
    DeleteFile((ExtractFilePath(Application->ExeName) +
"Data\\Y4.txt").c_str());
    DeleteFile((ExtractFilePath(Application->ExeName) +
"Data\\Delta1.txt").c_str());
    DeleteFile((ExtractFilePath(Application->ExeName) +
"Data\\Delta2.txt").c_str());
    DeleteFile((ExtractFilePath(Application->ExeName) +
"Data\\rms.txt").c_str());
    DeleteFile((ExtractFilePath(Application->ExeName) +
"Data\\C.txt").c_str());
    DeleteFile((ExtractFilePath(Application->ExeName) +
"Data\\V.txt").c_str());
    DeleteFile((ExtractFilePath(Application->ExeName) +
"Data\\XT.txt").c_str());
    DeleteFile((ExtractFilePath(Application->ExeName) +
"Data\\t.txt").c_str());
}

void __fastcall TForm3::Button1Click(TObject *Sender)
{
    Form6->ShowModal();
}

void __fastcall TForm3::Button2Click(TObject *Sender)
{
    Form7->ShowModal();
}

// ---------------------------------------------------
// Unit3.h

#ifndef Unit3H
#define Unit3H

#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
#include <Grids.hpp>

class TForm3 : public TForm
{
```

```
__published:
    TEdit *Edit1;
    TEdit *Edit2;
    TEdit *Edit3;
    TEdit *Edit4;
    TEdit *Edit5;
    TEdit *Edit6;
    TEdit *Edit7;
    TEdit *Edit8;
    TEdit *Edit9;
    TEdit *Edit10;
    TEdit *Edit11;
    TEdit *Edit12;
    TEdit *Edit13;
    TEdit *Edit14;
    TEdit *Edit15;
    TEdit *Edit16;
    TEdit *Edit17;
    TEdit *Edit18;
    TEdit *Edit19;
    TEdit *Edit20;
    TEdit *Edit21;
    TEdit *Edit22;
    TEdit *Edit23;
    TEdit *Edit24;
    TEdit *Edit25;
    TEdit *Edit26;
    TEdit *Edit27;
    TEdit *Edit28;
    TEdit *Edit29;
    TEdit *Edit30;
    TEdit *Edit31;
    TEdit *Edit32;
    TEdit *Edit33;
    TEdit *Edit34;
    TEdit *Edit35;
    TEdit *Edit36;
    TEdit *Edit37;
    TEdit *Edit38;
    TEdit *Edit39;
    TEdit *Edit40;
    TLabel *Label1;
    TLabel *Label2;
    TLabel *Label3;
```

```
TLabel *Label4;
TLabel *Label5;
TLabel *Label6;
TLabel *Label7;
TLabel *Label8;
TLabel *Label9;
TLabel *Label10;
TLabel *Label11;
TLabel *Label12;
TLabel *Label13;
TLabel *Label14;
TLabel *Label15;
TLabel *Label16;
TLabel *Label17;
TLabel *Label18;
TLabel *Label19;
TLabel *Label20;
TLabel *Label21;
TLabel *Label22;
TLabel *Label23;
TLabel *Label24;
TLabel *Label25;
TLabel *Label26;
TLabel *Label27;
TLabel *Label28;
TLabel *Label29;
TLabel *Label30;
TLabel *Label31;
TLabel *Label32;
TLabel *Label33;
TLabel *Label34;
TLabel *Label35;
TLabel *Label36;
TLabel *Label37;
TLabel *Label38;
TLabel *Label39;
TLabel *Label40;
TLabel *Label41;
TLabel *Label42;
TLabel *Label43;
TLabel *Label44;
TLabel *Label45;
TLabel *Label46;
TLabel *Label47;
```

```
    TEdit *Edit41;
    TEdit *Edit42;
    TStringGrid *StringGrid1;
    TButton *Button1;
    TLabel *Label48;
    TLabel *Label49;
    TLabel *Label50;
        TButton *Button2;
        TLabel *Label51;
        TLabel *Label52;
        TLabel *Label53;
        TLabel *Label54;
        TLabel *Label55;
        TLabel *Label56;
        TLabel *Label57;
        TLabel *Label58;
        TLabel *Label59;

    void __fastcall FormActivate(TObject *Sender);
    void __fastcall FormClose(TObject *Sender,
TCloseAction &Action);
    void __fastcall Button1Click(TObject *Sender);
        void __fastcall Button2Click(TObject *Sender);
private:
public:
    double n, m;
    __fastcall TForm3(TComponent* Owner);
};

extern PACKAGE TForm3 *Form3;

#endif

// ---------------------------------------------------
// Unit4.cpp

#include <vcl.h>
#pragma hdrstop

#include <cstdlib>
#include <ctime>

#include "Unit1.h"
#include "Unit3.h"
```

```
#include "Unit4.h"
#include "Matrix.hpp"

#pragma package(smart_init)
#pragma resource "*.dfm"
TForm4 *Form4;

__fastcall TForm4::TForm4(TComponent* Owner)
    : TForm(Owner)
{
}

void __fastcall TForm4::FormActivate(TObject *Sender)
{
    Timer1->Enabled = false;

    Matrix<double> X1, X2;
    Matrix<double> Y1, Y2, Y3, Y4;

    X1.ReadFromFile("data\\X1.txt");
    Y1.ReadFromFile("data\\Y1.txt");

    X2 = Matrix<double>(X1.GetNumberOfRows(),
X1.GetNumberOfColumns() + 1u);

    for (unsigned __int32 i = __int32(); i <
X2.GetNumberOfRows(); i++)
        for (unsigned __int32 j = __int32(); j <
X2.GetNumberOfColumns(); j++)

            X2(i, j) = (j == __int32()) ? double(1) :
X1(i, j - __int32(1));

    X1 = Transpose(X2);
    X1.WriteToFile("data\\XT.txt");

    Matrix<double> N(X1 * X2);

    double Det = Determinant(N);

    Form4->ProgressBar1->Position += 5;

    if (Det != double())
    {
```

```
        Matrix<double> Q(Inverse(N));
        Matrix<double> l1(X1 * Y1), l2;
        Matrix<double> a1(Q * l1), a2;

        Y2 = X2 * a1;

        Matrix<double> Xi(Form1->StringGrid1->RowCount -
1);

        std::srand((unsigned int)std::time(NULL));
        for (unsigned int i = 0u; i <
Xi.GetNumberOfRows(); i++)
            Xi(i, 0u) = (1 + std::rand() % 98) * 0.007;

        Xi.WriteToFile("Data\\Xi.txt");

        //Matrix<double> Xi("data\\Xi.txt");
        Matrix<double> Delta1(Form1->StringGrid1-
>RowCount - 1), Delta2;

        for (unsigned int i = 0u; i <
Delta1.GetNumberOfRows(); i++)
            Delta1(i, 0u) = Xi(i, 0u) - AM(Xi);

        K = StrToFloat(Form1->Edit1->Text)/RMS(Delta1);
        Delta2 = K * Delta1;

        Delta1.WriteToFile("data\\Delta1.txt");
        Delta2.WriteToFile("data\\Delta2.txt");

        Y3 = Y2;
        for (unsigned int i = 0u; i <
Y3.GetNumberOfRows(); i++)
            Y3(i, 0u) = Y2(i, 0u) + Delta2(i, 0u);

        l2 = X1 * Y3;
        a2 = Q * l2;

        a1.WriteToFile("data\\a1.txt");
        a2.WriteToFile("data\\a2.txt");

        Y4 = X2 * a2;

        Y2.WriteToFile("data\\Y2.txt");
```

```
        Y3.WriteToFile("data\\Y3.txt");
        Y4.WriteToFile("data\\Y4.txt");

        Matrix<double> V(Form1->StringGrid1->RowCount -
1);
        for (unsigned int i = 0u; i <
V.GetNumberOfRows(); i++)
            V(i, 0u) = Y4(i, 0u) - Y3(i, 0u);

        V.WriteToFile("data\\V.txt");

        mu = std::sqrt(SumProductM(V,
V)/(X2.GetNumberOfRows() - (X2.GetNumberOfColumns() - 1u)
- 1u));

        Matrix<double> rms(a1.GetNumberOfRows());

        for (unsigned int i = 0u; i <
rms.GetNumberOfRows(); i++)
            rms(i, 0u) = mu * std::sqrt(Q(i, i));

        rms.WriteToFile("data\\rms.txt");

        Matrix<double> t(a2.GetNumberOfRows());
        for (unsigned int i = 0u; i <
t.GetNumberOfRows(); i++)
            t(i, 0u) = std::fabs(a2(i, 0u)/rms(i, 0u));

        t.WriteToFile("data\\t.txt");

        Matrix< double > Q_zr;
        Q_zr = X2 * Q;

        Matrix< double > A1( 1u, X2.GetNumberOfColumns()
);
        Matrix< double > B1( X2.GetNumberOfColumns(), 1u
);
        Matrix< double > C1, C( X2.GetNumberOfRows() );

        for ( unsigned int i = 0u; i <
X2.GetNumberOfRows(); i++ )
        {
            for ( unsigned int a = 0u; a <
X2.GetNumberOfColumns(); a++ )
```

```
                A1(0u, a) = Q_zr(i, a);

            for ( unsigned int b = 0u; b <
X2.GetNumberOfColumns(); b++ )
                B1(b, 0u) = X1(b, i);

            C1 = A1 * B1;
            C(i, 0u) = mu * std::sqrt( C1(0u, 0u) );
        }

        C.WriteToFile("data\\C.txt");

    Form3->n = (double)(X2.GetNumberOfRows() -
(X2.GetNumberOfColumns() - 1u) - 1u);
    Form3->m = (double)(X2.GetNumberOfColumns() - 1u);
    }

    X1.~Matrix<double>();
    X2.~Matrix<double>();
    Y1.~Matrix<double>();
    Y2.~Matrix<double>();
    N .~Matrix<double>();

    Timer1->Enabled = true;
}

void __fastcall TForm4::Timer1Timer(TObject *Sender)
{
    ProgressBar1->Position++;

    if (ProgressBar1->Position == 100)
        Close();
}

//---------------------------------------------------
// Unit4.h

#ifndef Unit4H
#define Unit4H

#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
```

```
#include <ComCtrls.hpp>
#include <ExtCtrls.hpp>
#include "Matrix.hpp"
class TForm4 : public TForm
{
__published:
    TProgressBar *ProgressBar1;
    TTimer *Timer1;
    void __fastcall FormActivate(TObject *Sender);
    void __fastcall Timer1Timer(TObject *Sender);
private:
public:

    double K, mu;

    __fastcall TForm4(TComponent* Owner);
};

extern PACKAGE TForm4 *Form4;

#endif
// ---------------------------------------------------
// Unit5.h

#include <vcl.h>
#pragma hdrstop

#include "Unit5.h"

#pragma package(smart_init)
#pragma resource "*.dfm"
TForm5 *Form5;

__fastcall TForm5::TForm5(TComponent* Owner)
    : TForm(Owner)
{
}

void __fastcall TForm5::Button1Click(TObject *Sender)
{
    Close();
}
//---------------------------------------------------
// Unit5.h
```

```
#ifndef Unit5H
#define Unit5H

#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>

class TForm5 : public TForm
{
__published:
    TLabel *Label1;
    TLabel *Label2;
    TLabel *Label3;
    TButton *Button1;
    void __fastcall Button1Click(TObject *Sender);
private:
public:
    __fastcall TForm5(TComponent* Owner);
};

extern PACKAGE TForm5 *Form5;

#endif

// ---------------------------------------------------
// Unit6.cpp

#include <vcl.h>
#pragma hdrstop
#include "Matrix.hpp"

#include "Unit6.h"

#pragma package(smart_init)
#pragma resource "*.dfm"
TForm6 *Form6;

__fastcall TForm6::TForm6(TComponent* Owner)
    : TForm(Owner)
{
}
```

```
void __fastcall TForm6::FormActivate(TObject *Sender)
{
    Matrix<double> Y1("data\\Y1.txt");
    Matrix<double> Y2("data\\Y2.txt");
    Matrix<double> Y3("data\\Y3.txt");
    Matrix<double> Y4("data\\Y4.txt");

    StringGrid1->RowCount = Y1.GetNumberOfRows() + 1;
    StringGrid1->ColCount = 5;

    StringGrid1->Cells[1][0] = "Åêçàìåí. îö³íêè";
    StringGrid1->Cells[2][0] = "²ñòèíà ìîäåëü";
    StringGrid1->Cells[3][0] = "²ì³òàö³éíà ìîäåëü";
    StringGrid1->Cells[4][0] = "Çð³âíîâàæåíà";

    for (int i = 1; i < StringGrid1->RowCount; i++)
        StringGrid1->Cells[0][i] = i;

    for (int i = 1; i < StringGrid1->RowCount; i++)
    {
        StringGrid1->Cells[1][i] = Y1(i-1, 0);
        StringGrid1->Cells[2][i] = Y2(i-1, 0);
        StringGrid1->Cells[3][i] = Y3(i-1, 0);
        StringGrid1->Cells[4][i] = Y4(i-1, 0);
    }
}

void __fastcall TForm6::FormClose(TObject *Sender,
TCloseAction &Action)
{
    for (int i = 0; i < StringGrid1->RowCount; i++)
        StringGrid1->Rows[i]->Clear();
}
//---------------------------------------------------------
// Unit6.h

#ifndef Unit6H
#define Unit6H

#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
#include <Grids.hpp>
```

```
class TForm6 : public TForm
{
__published:
    TStringGrid *StringGrid1;
    void __fastcall FormActivate(TObject *Sender);
    void __fastcall FormClose(TObject *Sender,
TCloseAction &Action);
private:
public:
    __fastcall TForm6(TComponent* Owner);
};

extern PACKAGE TForm6 *Form6;

#endif

//---------------------------------------------------------
// Unit7.cpp

#include <vcl.h>
#pragma hdrstop

#include <cmath>

#include "Matrix.hpp"
#include "Unit7.h"

#pragma package(smart_init)
#pragma resource "*.dfm"
TForm7 *Form7;

__fastcall TForm7::TForm7(TComponent* Owner)
    : TForm(Owner)
{
}

void __fastcall TForm7::FormActivate(TObject *Sender)
{
    Matrix<double> Y1("data\\Y1.txt");
    Matrix<double> V("data\\V.txt");
    Matrix<double> Y3("data\\Y3.txt");
    Matrix<double> Y4("data\\Y4.txt");
```

```
    for (unsigned int i = 0u; i <= Y1.GetNumberOfRows();
i++)
    {
        Series1->AddXY(i, Y1(i, 0u), "", clRed);
        Series2->AddXY(i, Y4(i, 0u), "", clGreen);
        Series3->AddXY(i, std::pow(V(i, 0u), (double)2),
"", clBlue);
    }


}

// ---------------------------------------------------
// Unit7.h

#include <vcl.h>
#pragma hdrstop

#include <cmath>

#include "Matrix.hpp"
#include "Unit7.h"

#pragma package(smart_init)
#pragma resource "*.dfm"
TForm7 *Form7;

__fastcall TForm7::TForm7(TComponent* Owner)
    : TForm(Owner)
{
}

void __fastcall TForm7::FormActivate(TObject *Sender)
{
    Matrix<double> Y1("data\\Y1.txt");
    Matrix<double> V("data\\V.txt");
    Matrix<double> Y3("data\\Y3.txt");
    Matrix<double> Y4("data\\Y4.txt");

    for (unsigned int i = 0u; i <= Y1.GetNumberOfRows();
i++)
    {
        Series1->AddXY(i, Y1(i, 0u), "", clRed);
        Series2->AddXY(i, Y4(i, 0u), "", clGreen);
```

```
        Series3->AddXY(i, std::pow(V(i, 0u), (double)2),
"", clBlue);
    }


}

//---------------------------------------------------------
// Unit8.cpp

#include <vcl.h>
#pragma hdrstop

#include "Unit8.h"

#pragma package(smart_init)
#pragma resource "*.dfm"
TForm8 *Form8;

__fastcall TForm8::TForm8(TComponent* Owner)
    : TForm(Owner)
{
}

void __fastcall TForm8::Timer1Timer(TObject *Sender)
{
    Close();
}

// ---------------------------------------------------------
// Unit8.h

#ifndef Unit8H
#define Unit8H

#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
#include <ExtCtrls.hpp>
#include <Graphics.hpp>

class TForm8 : public TForm
{
```

```
__published:
        TImage *Image1;
        TLabel *Label1;
        TLabel *Label2;
        TLabel *Label3;
        TTimer *Timer1;
        void __fastcall Timer1Timer(TObject *Sender);
private:
public:
    __fastcall TForm8(TComponent* Owner);
};

extern PACKAGE TForm8 *Form8;

#endif


//////////////////////////////////////////////////////
// Matrix.hpp

#if !defined( MATRIX )
#define MATRIX

#include <cmath>
#include <fstream>

template <class T> class Matrix;

template <class T> const Matrix<T> &operator +(const
Matrix<T> &, const Matrix<T> &);
template <class T> const Matrix<T> &operator *(const
Matrix<T> &, const Matrix<T> &);
template <class T> const Matrix<T> &operator *(const T &,
const Matrix<T> &);

template <class T> T Determinant(const Matrix<T> &);
template <class T> Matrix<T> &Inverse(const Matrix<T> &);
template <class T> T Minor(const Matrix<T> &, unsigned
__int32, unsigned __int32);
template <class T> Matrix<T> &Transpose(const Matrix<T>
&);


template <class T> T AM(const Matrix<T> &);
```

```
template <class T> T RMS(const Matrix<T> &);
template <class T> T SumM(const Matrix<T> &);
template <class T> T SumProductM(const Matrix<T> &, const
Matrix<T> &);

template <class T> class Matrix
{
private:
    T *ptr_adress_matrix;
    unsigned __int32 number_of_rows;
    unsigned __int32 number_of_columns;

    void Destroy();
    void Init(unsigned __int32, unsigned __int32);
public:
    virtual ~Matrix();

    unsigned __int32 GetNumberOfRows() const;
    unsigned __int32 GetNumberOfColumns() const;
    unsigned __int32 GetNumberOfMatrixElements() const;

    const T& operator ()(unsigned int id_r, unsigned int
id_c) const
    {
        return *(this->ptr_adress_matrix + (this-
>GetNumberOfColumns() *
            ((id_r <= 0u) ? 0u :
                (id_r >= (this->GetNumberOfRows() - 1u))
?
                    (this->GetNumberOfRows() - 1u) :
id_r) +
            ((id_c <= 0u) ? 0u :
                (id_c >= (this->GetNumberOfColumns() -
1u)) ?
                    (this->GetNumberOfColumns() - 1u) :
id_c)));
    }

    T& operator ()(unsigned int id_r, unsigned int id_c)
    {
        return *(this->ptr_adress_matrix + (this-
>GetNumberOfColumns() *
            ((id_r <= 0u) ? 0u :
```

```
                (id_r >= (this->GetNumberOfRows() - 1u))
?
                    (this->GetNumberOfRows() - 1u) :
id_r) +
            ((id_c <= 0u) ? 0u :
                (id_c >= (this->GetNumberOfColumns() -
1u)) ?
                    (this->GetNumberOfColumns() - 1u) :
id_c)));
    }

    void ReadFromFile(const __int8 *);
    void WriteToFile(const __int8 *);

    const Matrix<T> &operator  =(const Matrix<T> &);
    const Matrix<T> &operator +=(const Matrix<T> &);
    const Matrix<T> &operator *=(const Matrix<T> &);
    const Matrix<T> &operator *=(const T &);


    Matrix(const Matrix<T> &);
    Matrix(const __int8 *);
    Matrix(unsigned __int32 = 1u, unsigned __int32 = 1u);
};

template <class T> inline void Matrix<T>::Destroy()
{
    if (this->ptr_adress_matrix)
        delete[] this->ptr_adress_matrix;

    this->ptr_adress_matrix = 0x00;
    this->number_of_rows    = __int32();
    this->number_of_columns = __int32();
}

template <class T> void Matrix<T>::Init(unsigned __int32
number_of_rows,
    unsigned __int32 number_of_columns)
{
    this->Destroy();

    this->ptr_adress_matrix = 0x00;
    this->number_of_rows    = __int32(number_of_rows);
    this->number_of_columns = __int32(number_of_columns);
```

```
    this->ptr_adress_matrix = new T [this-
>GetNumberOfMatrixElements()];

    for (unsigned __int32 i = __int32(); i < this-
>GetNumberOfMatrixElements(); i++)
        *(this->ptr_adress_matrix + i) = T();
}

template <class T> Matrix<T>::~Matrix() { this-
>Destroy(); }

template <class T> inline unsigned __int32
Matrix<T>::GetNumberOfRows() const { return this-
>number_of_rows; }
template <class T> inline unsigned __int32
Matrix<T>::GetNumberOfColumns() const { return this-
>number_of_columns; }
template <class T> inline unsigned __int32
Matrix<T>::GetNumberOfMatrixElements() const { return
this->GetNumberOfRows() * this->GetNumberOfColumns(); }

template <class T> void Matrix<T>::ReadFromFile(const
__int8 *pFileName)
{
    std::ifstream fin(pFileName);

    if (fin)
    {
        unsigned __int32 number_of_rows    = __int32();
        unsigned __int32 number_of_columns = __int32();

        fin >> number_of_rows >> number_of_columns;

        this->Init(
            (number_of_rows <= 1u) ? 1u :
                (number_of_rows >= 128u) ? 128u :
number_of_rows,
            (number_of_columns <= 1u) ? 1u :
                (number_of_columns >= 128u) ? 128u :
number_of_columns);

        for (unsigned __int32 i = __int32(); i < this-
>GetNumberOfRows(); i++)
```

```
            for (unsigned __int32 j = __int32(); j <
this->GetNumberOfColumns(); j++)

                fin >> (*this)(i, j);

        fin.close();
    }
}

template <class T> void Matrix<T>::WriteToFile(const
__int8 *pFileName)
{
    std::ofstream fout(pFileName);

    if (fout)
    {
        fout << this->GetNumberOfRows() << ' ' << this-
>GetNumberOfColumns()
            << ' ' << std::endl;

        for (unsigned __int32 i = __int32(); i < this-
>GetNumberOfRows(); i++)
        {
            for (unsigned __int32 j = __int32(); j <
this->GetNumberOfColumns(); j++)
                fout <<  (*this)(i, j) << ' ';

            fout << std::endl;
        }

        fout.close();
    }
}

template <class T> const Matrix<T> &Matrix<T>::operator
=(const Matrix<T> &rhs)
{
    if ((this != &rhs) && &rhs)
    {
        if ((this->GetNumberOfRows() !=
rhs.GetNumberOfRows()) ||
                (this->GetNumberOfColumns() !=
rhs.GetNumberOfColumns()))
```

```
            this->Init(rhs.GetNumberOfRows(),
rhs.GetNumberOfColumns());

        for (unsigned __int32 i = __int32(); i < this-
>GetNumberOfRows(); i++)
            for (unsigned __int32 j = __int32(); j <
this->GetNumberOfColumns(); j++)

                (*this)(i, j) = rhs(i, j);
    }

    return *this;
}

template <class T> const Matrix<T> &Matrix<T>::operator
+=(const Matrix<T> &rhs)
{
    if ((this->GetNumberOfRows() ==
rhs.GetNumberOfRows()) &&
        (this->GetNumberOfColumns() ==
rhs.GetNumberOfColumns()))

        for (unsigned __int32 i = __int32(); i < this-
>GetNumberOfRows(); i++)
            for (unsigned __int32 j = __int32(); j <
this->GetNumberOfColumns(); j++)

                (*this)(i, j) += rhs(i, j);

    return *this;
}

template <class T> const Matrix<T> &Matrix<T>::operator
*=(const Matrix<T> &rhs)
{
    if (this->GetNumberOfColumns() ==
rhs.GetNumberOfRows())
    {
        Matrix<T> M;
        M.Init(this->GetNumberOfRows(),
rhs.GetNumberOfColumns());

        for (unsigned int i = 0u; i <
M.GetNumberOfRows(); i++)
```

```
            for (unsigned int j = 0u; j <
M.GetNumberOfColumns(); j++)
                for (unsigned int k = 0u; k < this-
>GetNumberOfColumns(); k++)

                    M(i, j) += (*this)(i, k) * rhs(k, j);

        this->Destroy();
        *this = M;
    }

    return *this;
}

template <class T> Matrix<T>::Matrix(const Matrix<T>
&other)
{
    if ((this != &other) && &other)
    {
        this->ptr_adress_matrix = 0x00;
        this->number_of_rows    = __int32();
        this->number_of_columns = __int32();

        this->Init(other.GetNumberOfRows(),
other.GetNumberOfColumns());

        for (unsigned __int32 i = __int32(); i < this-
>GetNumberOfRows(); i++)
            for (unsigned __int32 j = __int32(); j <
this->GetNumberOfColumns(); j++)

                (*this)(i, j) = other(i, j);
    }
}

template <class T> Matrix<T>::Matrix(const __int8
*pFileName)
{
    this->ptr_adress_matrix = 0x00;
    this->number_of_rows    = __int32();
    this->number_of_columns = __int32();

    this->ReadFromFile(pFileName);
}
```

```
template <class T> Matrix<T>::Matrix(unsigned __int32
number_of_rows,
    unsigned __int32 number_of_columns)
{
    this->ptr_adress_matrix = 0x00;
    this->number_of_rows    = __int32();
    this->number_of_columns = __int32();

    this->Init(
        (number_of_rows <= 1u) ? 1u :
            (number_of_rows >= 128u) ? 128u :
number_of_rows,
        (number_of_columns <= 1u) ? 1u :
            (number_of_columns >= 128u) ? 128u :
number_of_columns);
}

template <class T> const Matrix<T> &operator +(const
Matrix<T> &lhs,
    const Matrix<T> &rhs)
{
    Matrix<T> *pM = 0x00;

    if ((lhs.getNumberOfRows() == rhs.GetNumberOfRows())
&&
        (lhs.GetNumberOfColumns() ==
rhs.GetNumberOfColumns()))
    {
        pM = new Matrix<T>(lhs.GetNumberOfRows(),
rhs.GetNumberOfColumns());

        for (unsigned __int32 i = __int32(); i < pM-
>GetNumberOfRows(); i++)
            for (unsigned __int32 j = __int32(); j < pM-
>GetNumberOfColumns(); j++)

                (*pM)(i, j) = lhs(i, j) + rhs(i, j);
    }
    else pM = new Matrix<T>(1u, 1u);

    return *pM;
}
```

```
template <class T> const Matrix<T> &operator *(const
Matrix<T> &lhs,
    const Matrix<T> &rhs)
{
    Matrix<T> *pM = 0x00;

    if (lhs.GetNumberOfColumns() ==
rhs.GetNumberOfRows())
    {
        pM = new Matrix<T>(lhs.GetNumberOfRows(),
rhs.GetNumberOfColumns());

        for (unsigned __int32 i = __int32(); i < pM-
>GetNumberOfRows(); i++)
            for (unsigned __int32 j = __int32(); j < pM-
>GetNumberOfColumns(); j++)
                for (unsigned __int32 k = __int32(); k <
rhs.GetNumberOfRows(); k++)

                    (*pM)(i, j) += lhs(i, k) * rhs(k, j);
    }

    return *pM;
}

template <class T> const Matrix<T> &operator *(const T
&lhs,
    const Matrix<T> &rhs)
{
    Matrix<T> *pM = new Matrix<T>(rhs.GetNumberOfRows(),
rhs.GetNumberOfColumns());

    for (unsigned __int32 i = __int32(); i < pM-
>GetNumberOfRows(); i++)
        for (unsigned __int32 j = __int32(); j < pM-
>GetNumberOfColumns(); j++)

            (*pM)(i, j) = lhs * rhs(i, j);

    return *pM;
}

template <class T> T Determinant(const Matrix<T> &M)
{
```

```
    T Result = T();

    if (M.GetNumberOfRows() == M.GetNumberOfColumns())
    {
        switch (M.GetNumberOfMatrixElements())
        {
            case 1u: Result = M(0u, 0u); break;
            case 4u: Result = M(0u, 0u) * M(1u, 1u) -
M(0u, 1u) * M(1u, 0u); break;

            default:
                for (unsigned __int32 j = __int32(); j <
M.GetNumberOfColumns(); j++)
                    Result += M(0u, j) * std::pow((-
1.0f), (__int32)(2u + j)) * Minor(M, 0u, j);
            break;
        }
    }

    return Result;
}

template <class T> Matrix<T> &Inverse(const Matrix<T> &M)
{
    Matrix<T> *pM = 0x00000000;

    if (M.GetNumberOfRows() == M.GetNumberOfColumns())
    {
        pM = new Matrix<T>(M.GetNumberOfRows(),
M.GetNumberOfColumns());

        for (unsigned __int32 i = __int32(); i < pM-
>GetNumberOfRows(); i++)
            for (unsigned __int32 j = __int32(); j < pM-
>GetNumberOfColumns(); j++)

                (*pM)(i, j) = std::pow(-1.0f, (int)(2u +
i + j)) * Minor(M, i, j);

        (*pM) = (1.0/Determinant(M)) * Transpose(*pM);
    }

    return *pM;
}
```

```
template <class T> T Minor(const Matrix<T> &M, unsigned
__int32 fix_r,
    unsigned __int32 fix_c)
{
    Matrix<T> M1((M.GetNumberOfRows() - 1u),
(M.GetNumberOfColumns() - 1u));

    for (unsigned __int32 i = __int32(), l = __int32(); i
< M1.GetNumberOfRows(); i++, l++)
        for (unsigned __int32 j = __int32(), k =
__int32(); j < M1.GetNumberOfColumns(); j++, k++)
        {
            if (l == fix_r) l++;
            if (k == fix_c) k++;

            M1(i, j) = M(l, k);
        }

    return Determinant(M1);
}

template <class T> Matrix<T> &Transpose(const Matrix<T>
&M)
{
    Matrix<T> *pM = 0x00000000;

    if (&M)
    {
        pM = new Matrix<T>(M.GetNumberOfColumns(),
M.GetNumberOfRows());

        for (unsigned __int32 i = __int32(); i < pM-
>GetNumberOfRows(); i++)
            for (unsigned __int32 j = __int32(); j < pM-
>GetNumberOfColumns(); j++)

                (*pM)(i, j) = M(j, i);
    }

    return *pM;
}

template <class T> T AM(const Matrix<T> &M)
```

```
{
    return SumM(M)/M.GetNumberOfMatrixElements();
}

template <class T> T RMS(const Matrix<T> &M)
{
    return std::sqrt(SumProductM(M,
M)/M.GetNumberOfMatrixElements());
}

template <class T> T SumM(const Matrix<T> &M)
{
    T sum = T();

    for ( unsigned int i = 0u; i < M.GetNumberOfRows();
i++ )
        for ( unsigned int j = 0u; j <
M.GetNumberOfColumns(); j++ )

            sum += M(i, j);

    return( sum );
}

template <class T> T SumProductM(const Matrix<T> &M1,
const Matrix<T> &M2)
{
    Matrix<T> product(M1.GetNumberOfRows(),
M1.GetNumberOfColumns());

    for ( unsigned int i = 0u; i <
product.GetNumberOfRows(); i++ )
        for ( unsigned int j = 0u; j <
product.GetNumberOfColumns(); j++ )

            product(i, j) = M1(i, j) * M2(i, j);

    return SumM(product);
}

#endif
```

Костючок Сергій Васильович
спеціаліст системотехнік, магістрант інформаційних технологій

Побудова моделі вивчення базової дисципліни в середовищі С++ і її використання в курсі «Педагогіка вищої школи»

8.080201 – „Інформатика”

М О Н О Г Р А Ф І Я

магістерської дисертації на здобуття академічного ступеня магістра з інформатики

Комп’ютерний набір в редакторі Microsoft®Office® Word 2007 О.І.Чернявський
Редагування, верстка, макетування та дизайн Р.М.Літнарович.
Науковий керівник Р. М. Літнарович, доцент, кандидат технічних наук
Міжнародний Економіко-Гуманітарний Університет ім. акад. Степана Дем’янчука
Кафедра математичного моделювання
33027,м.Рівне,Україна
Вул.акад. С.Дем’янчука,4, корпус 1
Телефон:(+00380) 362 23-73-09
Факс:(+00380) 362 23-01-86
E-mail:mail@regi.rovno.ua
kostyuchok87@gmail.com