НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
КАФЕДРА МАТЕМАТИЧНИХ МЕТОДІВ ЗАХИСТУ ІНФОРМАЦІЇ

«До захисту допущено»
Завідувач кафедри
_____ М. М. Савчук
(підпис) *(ініціали, прізвище)*
“____” _____ 2015 р.

Дипломна робота

освітньо-кваліфікаційного рівня “бакалавр”

з напряму підготовки 6.040301 «Прикладна математика»
на тему «Побудова математичної моделі психологічних характеристик людини на основі результатів тестування»
Виконав студент 4 курсу групи ФІ-13
Кригін Валерій Михайлович _____
(підпис)
Керівник д-р ф.-м. наук, професор Дороговцев Андрій Анатолійович _____
(підпис)
Рецензент _____
(підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ — 2015 року

РЕФЕРАТ

Неотъемлемой частью оценки успеваемости студентов является анализ результатов тестирования, однако существующие на данный момент системы тестирования учитывают лишь правильность ответов.

Цель данной работы — разработка математической модели психологических особенностей человека на основе результатов тестирования. Это поможет создать тестирующую систему, способную давать нужные советы людям, которые прошли тестирование.

Для достижения поставленной цели

- был использован метод главных компонент, чтобы извлечь наиболее важные данные из анализируемых выборок;
- был использован критерий Пирсона (χ^2) для проверки гипотез о распределении выборок.

ABSTRACT

Analysis of testing results is a valuable tool in assessing students with their academic standing, however current testing systems focus only in the accuracy of answers.

This study aims to design a mathematical model of individual psychological characteristics on the basis of test results. This will help to implement testing system which is able to give relevant advices to person who has passed the test.

To perform the study

- principal components analysis was used to get most significant data from analyzed samples;
- Pearson's chi-squared test (χ^2) was used to check hypothesis about samples' distribution.

РЕФЕРАТ

Аналіз результатів тестування є невід'ємною частиною перевірки рівня знань студентів, проте існуючі на даний момент системи тестування враховують лише вірність відповідей.

Метою даної роботи є побудова математичної моделі психологічних характеристик людини на основі результатів тестування. Це допоможе створити тестуючу систему, що може надавати доречні поради людям, що пройшли тестування.

Для досягнення мети

- було використано метод головних компонент, щоб отримати найбільш важливі дані з аналізованих вибірок;
- було використано критерій Пірсона (χ^2) для перевірки гіпотез щодо розподілу вибірок.

ЗМІСТ

Вступ 6
1 Теоретичні відомості 8
1.1 Математичне підґрунтя 8
1.1.1 Метод головних компонент 8
1.1.2 Побудова гістограми для критерію Пірсона 12
1.1.3 Критерій узгодженості Пірсона χ^2 13
1.2 Дані з психології 21
1.2.1 Типи вищої нервової діяльності 21
1.2.2 Теппінг-тест (Tapping rate) 22
Висновки до розділу 1. 24
2 Моделювання результатів виконання тестових завдань 26
2.1 Моделювання результатів виконання теппінг-тесту. 26
2.1.1 Процес Пуассона. 26
2.1.2 Квадратична апроксимація методом найменших квадратів 28
2.1.3 Аналіз квадратичної апроксимації 31
2.2 Моделювання результатів виконання тестових завдань 37
2.2.1 Інформаційний метод прогнозування часу розв'язання завдань . . . 37
2.2.2 Моделювання тесту 40
Перелік посилань 47
Додаток А 48
Додаток Б 50
Додаток В 51
Додаток Г 53
Додаток Д 55
Додаток Е 58

ВСТУП

Актуальність роботи полягає в тому, що дистанційне навчання стає дедалі популярнішим, а існуючі на даний момент системи тестування недостатньо гнучкі: вони аналізують лише відповіді на запитання, відносячи їх до вірних або невірних, а на цій базі роблять кінцевий висновок щодо знань студента. З одного боку, це підвищує їх об'єктивність, з іншого, навпаки: всі студенти різні і більшості з них потрібна допомога хоча б у вигляді порад. У викладачів може не вистачати часу на докладне знайомство з кожним студентом, але якщо буде програмно-технічний комплекс, що підкаже викладачам, до яких студентів який підхід краще мати, це безумовно підвищить якість навчання. Стрімкий розвиток комп'ютерної техніки й інформаційних технологій надає можливість визначати ритм складання тесту, а також індивідуальні особливості людини. Дані психологічних досліджень допоможуть правильно трактувати отримані значення, а добре вивчені та перевірені часом математичні методи надають великі можливості для систематизації та обробки результатів вимірювання.

Об'єкти дослідження: студенти, системи тестування.

Предмети дослідження: психологічні особливості студентів, моделі їх поведінки.

Задача: Побудувати математичну модель психологічних характеристик людини на основі результатів тестування.

Метою роботи є покращення якості навчання за допомогою порад студентам і викладачам практичних занять.

Завдання наступні:

1) Вивчити математичні методи та розділи психології, що дозволять розробити математичну модель психологічних характеристик людини, пояснити та обґрунтувати отримані результати
2) Розробити математичну модель людини, що складає тести

3) Розробити та проанализувати модель складання тестів людьми різних типів

1 ТЕОРЕТИЧНІ ВІДОМОСТІ

1.1 Математичне підґрунтя

1.1.1 Метод головних компонент

Метод головних компонент (Principal component analysis) — метод, що дозволяє зменшити розмірність досліджуваної вибірки з мінімальними втратами інформації. [1]

Маємо m об'єктів, з яких треба зняти по n певних властивостей. На вході в нас є виборки $\vec{X}_k$, кожна з яких відповідає сукупності властивостей k-го об'єкту

$$\vec{X}_k = \begin{bmatrix} x_k^1 \\ x_k^2 \\ \vdots \\ x_k^n \end{bmatrix}, \qquad k = \overline{1,m}.$$

Згрупуємо всі вимірювання в одну матрицю X

$$X = \begin{bmatrix} x_1^1 & x_2^1 & \dots & x_m^1 \\ x_1^2 & x_2^2 & \dots & x_m^2 \\ \vdots & \vdots & \ddots & \vdots \\ x_1^n & x_2^n & \dots & x_m^n \end{bmatrix}.$$

Спочатку нам знадобиться знайти вибіркові середні значення для кожної

властивості

$$a_i = \frac{1}{m} \cdot \sum_{k=1}^{m} x_k^i, \qquad i = \overline{1,n}.$$

Маємо вектор вибіркових середніх значень

$$\vec{a} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix}.$$

Центруємо отримані дані, що містяться в матриці X, віднявши від кожного стовбця вектор вибіркових середніх $\vec{a}$

$$\tilde{X} = \begin{bmatrix} \tilde{x}_1^1 & \tilde{x}_2^1 & \dots & \tilde{x}_m^1 \\ \tilde{x}_1^2 & \tilde{x}_2^2 & \dots & \tilde{x}_m^2 \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{x}_1^n & \tilde{x}_2^n & \dots & \tilde{x}_m^n \end{bmatrix} = \begin{bmatrix} x_1^1 - a_1 & x_2^1 - a_1 & \dots & x_m^1 - a_1 \\ x_1^2 - a_2 & x_2^2 - a_2 & \dots & x_m^2 - a_2 \\ \vdots & \vdots & \ddots & \vdots \\ x_1^n - a_n & x_2^n - a_n & \dots & x_m^n - a_n \end{bmatrix}.$$

Обчислюємо вибіркову коваріаційну матрицю властивостей. Вибіркову коваріацію i та j властивості рахуємо за формулою

$$\sigma_i^j = \frac{1}{m} \cdot \sum_{k=1}^{m} \tilde{x}_k^i \cdot \tilde{x}_k^j = \frac{1}{m} \cdot \sum_{k=1}^{m} \left[\left(x_k^i - a_i \right) \cdot \left(x_k^j - a_j \right) \right], \qquad i,j = \overline{1,n}.$$

Маємо вибіркову коваріаційну матрицю

$$K = \begin{bmatrix} \sigma_1^1 & \sigma_2^1 & \dots & \sigma_n^1 \\ \sigma_1^2 & \sigma_2^2 & \dots & \sigma_n^2 \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_1^n & \sigma_2^n & \dots & \sigma_n^n \end{bmatrix}.$$

Щоб отримувати лише потрібну інформацію, ми хочемо знайти таке ортогональне лінійне перетворення L вхідної матриці $\tilde{X}$, щоб отримати матрицю $Y = L \cdot \tilde{X}$, яка має діагональну вибіркову ковариаційну матрицю K' з незростаючими зверху вниз значеннями. Діагональна вибіркова коваріаційна матриця гарантує той факт, що отримані значення Y будуть некорельованими. Рангування значень діагональних елементів матриці K' за величиною дасть більш наочне уявлення про будову досліджуваних об'єктів, адже діагональні елементи — вибіркові дисперсії. Чим більше дисперсія, тим більше відповідна властивість змінюється від об'єкту до об'єкту, і тим більше корисної інформації вона нам надає.

Вибіркова коваріаційна матриця K' для $Y = L \cdot \tilde{X}$ має вигляд

$$K' = L \cdot K \cdot L^* = \begin{bmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda_n \end{bmatrix}.$$

З лінійної алгебри відомо, що матриця L складається з координат власних ве-

кторів матриці K, а елементи λ_k — її власні числа, які існують і є невід'ємними через невід'ємну означеність матриці K. Вважаємо, що числа $\lambda_1, \ldots, \lambda_n$ впорядковані від більшого до меншого для зручності подальших дій. Позначимо власний вектор матриці K, що відповідає власному числу λ_k, як $\vec{l_k}$. Тоді

$$\vec{l_k} = \left[l_k^1, l_k^2, \ldots, l_k^n\right], \qquad k = \overline{1,n}.$$

Матриця L має вигляд

$$L = \begin{bmatrix} l_1^1 & l_1^2 & \ldots & l_1^n \\ l_2^1 & l_2^2 & \ldots & l_2^n \\ \vdots & \vdots & \ddots & \vdots \\ l_n^1 & l_n^2 & \ldots & l_n^n \end{bmatrix}.$$

Треба зменшити розмірність простору досліджуваних параметрів системи з n до $p < n$, але при цьому втратити якомога менше відомостей про досліджувані об'єкти. Введемо міру інформації, що залишається при зменшенні кількості компонент, що розглядаються

$$I = \frac{\lambda_1 + \cdots + \lambda_p}{\lambda_1 + \cdots + \lambda_n}.$$

Будемо вважати, що діємо продуктивно, тому починаємо обирати з перших компонент, адже саме вони є найбільш інформативними. Також бачимо, що інформативність змінюється в межах від 0 (нічого не дізнаємось) до 1 (зберегли усю інформацію).

Надалі буде розглядатися матриця головних компонент Y

$$Y = \begin{bmatrix} y_1^1 & y_2^1 & \dots & y_m^1 \\ y_1^2 & y_2^2 & \dots & y_m^2 \\ \vdots & \vdots & \ddots & \vdots \\ y_1^p & y_2^p & \dots & y_m^p \end{bmatrix}.$$

1.1.2 Побудова гістограми для критерію Пірсона

Для подальшого аналізу потрібно здобути щільність розподілу головних компонент. Оскільки маємо справу з вибіркою і вибірковими характеристиками, потрібно побудувати гістограму, адже це і є вибіркова характеристика, що відповідає щільності. [2]

Побудуємо j-й стовбець гістограми для виборки з k-ї строки матриці Y

$$h_j^k = \frac{1}{m} \cdot \sum_{i=1}^{m} \mathbb{1}\left(y_i^k \in I_j^k\right), \qquad j = \overline{1, N}, \qquad k = \overline{1, p},$$

де I^k — набір напівінтервалів, що розбиває відрізок $\left[\min\limits_{i=\overline{1,m}} y_i^k; \max\limits_{i=\overline{1,m}} y_i^k\right]$ на N рівних частин. Для вибору N можна скористатися досить відомою формулою Стьорджеса (Sturges' formula) [3]

$$N = \lfloor \log_2 m \rfloor + 1.$$

Маємо матрицю гістограм

$$H = \begin{bmatrix} h_1^1 & h_2^1 & \dots & h_N^1 \\ h_1^2 & h_2^2 & \dots & h_N^2 \\ \vdots & \vdots & \ddots & \vdots \\ h_1^p & h_2^p & \dots & h_N^p \end{bmatrix}$$

і напівінтервалів, що відповідають кожному стовбчику кожної гістограми

$$I = \begin{bmatrix} I_1^1 & I_2^1 & \dots & I_N^1 \\ I_1^2 & I_2^2 & \dots & I_N^2 \\ \vdots & \vdots & \ddots & \vdots \\ I_1^p & I_2^p & \dots & I_N^p \end{bmatrix}.$$

1.1.3 Критерій узгодженості Пірсона χ^2

Гістограма може використовуватися не тільки для графічної інтерпретації отриманих даних, але й для віднесення вибірки до якогось відомого розподілу. Відповідь на питання “Чи дійсно вибірка y_1^k, …, y_m^k має розподіл F^k?” може надати критерій узгодженості Пірсона.

Розглянемо вектор

$$\eta^k = \left[\frac{\nu_1^k - m \cdot \rho_1^k}{\sqrt{m \cdot \rho_1^k}}, \dots, \frac{\nu_N^k - m \cdot \rho_N^k}{\sqrt{m \cdot \rho_N^k}} \right].$$

Знайдемо його характеристичну функцію

$$\varphi_{\eta^k}(\lambda) = \mathrm{M}\, e^{i\cdot\left(\lambda,\eta^k\right)}, \qquad \lambda \in \mathbb{R}^N.$$

Для зручності перепозначимо індикатор

$$\mathfrak{I}_{i,j}^k = \mathbb{1}\left\{y_i^k \in I_j^k\right\}.$$

Подивимось, чому дорівнює скалярний добуток в експоненті

$$\begin{aligned}\left(\lambda,\eta^k\right) &= \sum_{j=1}^{N} \lambda_j \cdot \frac{\nu_j^k - m\cdot\rho_j^k}{\sqrt{m\cdot\rho_j^k}} = \sum_{j=1}^{N} \frac{\lambda_j}{\sqrt{m\cdot\rho_j^k}} \cdot \sum_{i=1}^{m}\left(\mathfrak{I}_{i,j}^k - \rho_j^k\right) = \\ &= \sum_{j=1}^{N}\sum_{i=1}^{m} \frac{\lambda_j}{\sqrt{m\cdot\rho_j^k}} \cdot \left(\mathfrak{I}_{i,j}^k - \rho_j^k\right) = \sum_{i=1}^{m}\sum_{j=1}^{N} \lambda_j \cdot \frac{\mathfrak{I}_{i,j}^k - \rho_j^k}{\sqrt{m\cdot\rho_j^k}}.\end{aligned}$$

Бачимо суму m незалежних однаково розподілених випадкових величин. Введемо позначення

$$\mathfrak{I}_j^k = \mathbb{1}\left\{y_1^k \in I_j^k\right\},$$

а також позначимо новий випадковий вектор

$$\zeta^k = \left[\frac{\mathfrak{I}_1^k - \rho_1^k}{\sqrt{m\cdot\rho_1^k}}, \dots \frac{\mathfrak{I}_N^k - \rho_N^k}{\sqrt{m\cdot\rho_N^k}}\right].$$

Скалярний добуток прийняв вигляд

$$\left(\lambda,\eta^k\right) = \sum_{i=1}^{m}\sum_{j=1}^{N} \lambda_j\cdot\zeta_j^k = \sum_{i=1}^{m}\left(\lambda,\zeta^k\right) = m\cdot\left(\lambda,\zeta^k\right).$$

За рахунок незалежності випадкових величин ζ_j^k маємо

$$\varphi_{\eta^k}(\lambda) = \mathrm{M}\, e^{i\cdot(\lambda,\eta^k)} = \mathrm{M}\, e^{m\cdot i\cdot(\lambda,\zeta^k)} = \left(\mathrm{M}\, e^{i\cdot(\lambda,\zeta^k)}\right)^m. \tag{1.1}$$

Розглянемо характеристичну функцію випадкового вектора ζ^k

$$\varphi_{\zeta^k}(\lambda) = \mathrm{M}\left[\exp\left\{i\cdot\sum_{j=1}^{N}\lambda_j\cdot\zeta_j^k\right\}\right]. \tag{1.2}$$

Легко побачити, що

$$\begin{aligned}\left(\lambda,\zeta^k\right) = \sum_{j=1}^{N}\lambda_j\cdot\zeta_j^k = \sum_{j=1}^{N}\lambda_j\cdot\frac{\mathfrak{I}_j^k-\rho_j^k}{\sqrt{m\cdot\rho_j^k}} = \sum_{j=1}^{N}\left(\frac{\lambda_j}{\sqrt{m\cdot\rho_j^k}}\cdot\mathfrak{I}_j^k - \frac{\sqrt{\rho_j^k}\cdot\lambda_j}{\sqrt{m}}\right) = \\ = \sum_{j=1}^{N}\mathfrak{I}_j^k\cdot\left(\frac{\lambda_j}{\sqrt{m\cdot\rho_j^k}} - \sum_{l=1}^{N}\frac{\sqrt{\rho_l^k}\cdot\lambda_l}{\sqrt{m}}\right).\end{aligned}$$

Тобто характеристична функція (1.2) приймає вигляд

$$\varphi_{\zeta^k}(\lambda) = \mathrm{M}\left[\sum_{j=1}^{N}\mathfrak{I}_j^k\cdot\exp\left\{\frac{i}{\sqrt{m}}\left(\frac{\lambda_j}{\sqrt{\rho_j^k}} - \sum_{l=1}^{N}\sqrt{\rho_l^k}\cdot\lambda_l\right)\right\}\right].$$

Перепозначимо вираз в круглих дужках

$$\mathfrak{z}^k = \frac{\lambda_j}{\sqrt{\rho_j^k}} - \sum_{l=1}^{N}\sqrt{\rho_l^k}\cdot\lambda_l.$$

Математичне очікування індикатора — ймовірність події, яку він перевіряє. Отже

$$\varphi_{\zeta^k}(\lambda) = \sum_{j=1}^{N} \rho_j^k \cdot \exp\left\{\frac{i \cdot \mathfrak{z}^k}{\sqrt{m}}\right\}.$$

Якщо розмір вибірки m буде зростати, то характеристична функція η^k (1.1) буде поводитись наступним чином

$$\lim_{m\to\infty} \varphi_{\eta^k}(\lambda) = \lim_{m\to\infty} \left(1 + \sum_{k=1}^{N} p_k \cdot \left[\exp\left\{\frac{i \cdot \mathfrak{z}^k}{\sqrt{m}}\right\} - 1\right] \cdot \frac{m}{m}\right)^m = \\ = \lim_{m\to\infty} \exp\left\{m \cdot \sum_{k=1}^{N} p_k \cdot \left[\exp\left\{\frac{i \cdot \mathfrak{z}^k}{\sqrt{m}}\right\} - 1\right]\right\}.$$

Для $\exp\left\{\frac{i\cdot\mathfrak{z}^k}{\sqrt{m}}\right\}$ використаємо співвідношення

$$e^\alpha - 1 \approx \alpha + \frac{\alpha^2}{2}, \qquad \alpha \ll 1.$$

Маємо

$$\sum_{j=1}^{N} \rho_j^k \cdot \frac{i \cdot \mathfrak{z}^k}{\sqrt{m}} = \sum_{j=1}^{N} \rho_j^k \cdot \frac{i}{\sqrt{m}} \cdot \left(\frac{\lambda_j}{\sqrt{\rho_j^k}} - \sum_{l=1}^{N} \sqrt{\rho_l^k} \cdot \lambda_l\right) = \\ = \frac{i}{\sqrt{m}} \cdot \left(\sum_{j=1}^{N} \sqrt{\rho_j^k} \cdot \lambda_j - \sum_{l=1}^{N} \sqrt{\rho_l^k} \cdot \lambda_l\right) = 0,$$

$$\sum_{j=1}^{N} \rho_j^k \cdot \left(\frac{i \cdot \mathfrak{z}^k}{\sqrt{m}} \right)^2 = -\sum_{j=1}^{N} \frac{\rho_j^k}{m} \cdot \left(\frac{\lambda_j}{\sqrt{\rho_j^k}} - \sum_{l=1}^{N} \sqrt{\rho_l^k} \cdot \lambda_l \right)^2 =$$

$$= -\frac{1}{m} \cdot \left[\sum_{j=1}^{N} \lambda_j - \left(\sum_{l=1}^{N} \sqrt{\rho_l^k} \cdot \lambda_l \right)^2 \right],$$

тому

$$\lim_{m\to\infty} \varphi_{\eta^k}(\lambda) = \lim_{m\to\infty} \exp\left\{ -\frac{m}{m \cdot 2} \cdot \left[\sum_{j=1}^{N} \lambda_j - \left(\sum_{l=1}^{N} \sqrt{\rho_l^k} \cdot \lambda_l \right)^2 \right] \right\} =$$

$$= \exp\left\{ -\frac{1}{2} \cdot \left[\sum_{j=1}^{N} \lambda_j - \left(\sum_{l=1}^{N} \sqrt{\rho_l^k} \cdot \lambda_l \right)^2 \right] \right\} = e^{-\frac{1}{2} \cdot \left(A^k \lambda, \lambda \right)}.$$

Матриця A^k побудована наступним чином

$$A^k = \left\| \delta_{ij} - \sqrt{\rho_i^k} \cdot \sqrt{\rho_j^k} \right\|_{i=1}^{n}.$$

Симетричність мариці очевидна, тому треба довести її невід'ємну визначеність, щоб стверджувати, що вона є коваріаційною. Для цього візьмемо вектор

$$e^k = \left[\sqrt{\rho_1^k}, \ldots, \sqrt{\rho_N^k} \right], \qquad \left\| e^k \right\| = 1.$$

Тоді бачимо, що

$$\left(A^k \lambda, \lambda \right) = \|\lambda\|^2 - \left(\lambda, e^k \right)^2. \tag{1.3}$$

З нерівності Коші маємо

$$\left\| \left(\lambda, e^k \right) \right\| \le \|\lambda\| \cdot \left\| e^k \right\| = \|\lambda\|.$$

Тобто матриця є дійсно невід'ємно визначеною і вектор η^k розподілений за нормальним законом з нульовим середнім і коваріаційною матрицею A^k.

Для подальших розрахунків розглянемо стандартний гаусівський вектор як суму випадкових нормально розподілених випадкових величин в стандартному базисі $\mathbb{R}^N$, який позначимо $[e_1, \ldots, e_N]$

$$\xi = \sum_{j=1}^{N} \xi_j \cdot e_j \sim N\left(\vec{0}, I\right).$$

Згадуємо, що ортогональні перетворення U зберігають відстані, а також справедливо наступне

$$U\xi \sim N\left(0, UIU^{-1}\right) \sim N\left(\vec{0}, I\right).$$

Також ортонормований базис залишається ортонормованим базисом після ортогонального перетворення U. Оберемо такий оператор U, щоб набір $[e_1, \ldots, e_N]$ під його дією перетворився на $[f_1, \ldots, f_N]$, де

$$f_1 = e^k = \left[\sqrt{\rho_1^k}, \ldots, \sqrt{\rho_N^k}\right].$$

Тоді маємо вектор

$$U\xi = \hat{\xi} = \sum_{j=1}^{N} \hat{\xi}_j \cdot f_j \sim N\left(\vec{0}, I\right).$$

Подивимось, який розподіл має наступний вектор

$$\Upsilon = \sum_{j=2}^{N} \hat{\xi}_j \cdot f_j = \hat{\xi} - \hat{\xi}_1 \cdot e^k.$$

Для цього розглянемо квадратичну форму

$$\mathrm{M}\left(\Upsilon,\lambda\right)^2=\sum_{j=2}^{N}\left(\lambda,f_j\right)^2=\sum_{j=1}^{N}\left(\lambda,f_j\right)^2-\left(\lambda,f_1\right)^2=\|\lambda\|^2-\left(\lambda,e^k\right)^2=\left(A^k\lambda,\lambda\right).$$

З рівності (1.3) бачимо, що випадкові вектори η^k та Υ мають однаковий розподіл. Отже, розподіли їх норм теж співпадають. Оскільки сума $N-1$ квадратів незалежних стандартних гаусових випадкових величин має розподіл Пірсона з $N-1$ ступенями вільності

$$\|\Upsilon\|^2=\sum_{j=2}^{N}\xi_j^2\sim\chi^2_{N-1}.$$

Маємо

$$\left\|\eta^k\right\|=\sum_{j=1}^{N}\frac{\left(\nu_j^k-m\cdot\rho_j^k\right)^2}{m\cdot\rho_j^k}=m\cdot\sum_{j=1}^{N}\frac{\left(h_j^k-\rho_j^k\right)^2}{\rho_j^k}\sim\chi^2_{N-1}.$$

Останнє співвідношення дає змогу перевіряти належність виборки $y_1^k,\ldots,y_m^k$ до розподілу F^k. Перевірка виглядає наступним чином.

Розглянемо випадкову величину

$$R^k=m\cdot\sum_{j=1}^{N}\frac{\left(h_j^k-\rho_j^k\right)^2}{\rho_j^k}. \tag{1.4}$$

Обираємо рівень значущості α для функції розподілу χ^2_{N-1} і шукаємо відповідне до кількості ступенів вільності r_α. Рівень значущості — ймовірність помилки першого роду, тобто ймовірність того, що буде відкинуто вірну гіпотезу

$$\mathbb{P}\left(\chi^2_{N-1}\geq r_\alpha\right)=\alpha.$$

Якщо $R^k \leq r_\alpha$, то гіпотеза про те, що вибірка Y^k дійсно має розподіл F^k, приймається.

Розглянемо той випадок, коли ймовірність ρ_i^k відгадана невірно. Повернемося до формули (1.4)

$$R^k = \sum_{j=1}^{N} \frac{\left(\nu_j^k - m \cdot \rho_j^k\right)^2}{m \cdot \rho_j^k}.$$

Всі члени суми є невід'ємними. Якщо хоча б один елемент буде завеликим, то великою буде вся сума. Маємо випадкову величину η

$$\eta = \nu_i^k - m \cdot \rho_i^k = \sum_{j=1}^{m} \left(\xi_j - \rho_i^k\right), \qquad \mathbb{1}\left(y_j^k \in I_i^k\right) = \xi_j.$$

Якщо ρ_i^k вгадано невірно, то воно не дорівнює математичному очікуванню індикатора. Додамо та віднімемо справжнє математичне очікування

$$\eta = \sum_{j=1}^{m} \left(\xi_j - \mathrm{M}\,\xi_1 + \mathrm{M}\,\xi_1 - \rho_i^k\right) = \sum_{j=1}^{m} \left(\xi_j - \mathrm{M}\,\xi_1\right) + \sum_{j=1}^{m} \left(\mathrm{M}\,\xi_1 - \rho_i^k\right).$$

Останній доданок є просто різницею, яку помножено на m

$$\eta = \sum_{j=1}^{m} \left(\xi_j - \mathrm{M}\,\xi_1\right) + m \cdot \left(\mathrm{M}\,\xi_1 - \rho_i^k\right).$$

Поділимо на $\sqrt{m}$, щоб скористатися центральною граничною теоремою

$$\frac{\eta}{\sqrt{m}} = \frac{1}{\sqrt{m}} \cdot \sum_{j=1}^{m} \left(\xi_j - \mathrm{M}\,\xi_1\right) + \frac{1}{\sqrt{m}} \cdot m \cdot \left(\mathrm{M}\,\xi_1 - \rho_i^k\right).$$

Перший доданок асимптотично має розподіл $N\left(0, \sigma^2\right)$, де σ^2 — дисперсія випад-

кової величини ξ_1 для достатньо великих m. Отже вся сума зростає пропорційно до $\sqrt{m}$

$$\frac{\eta}{\sqrt{m}} = \frac{1}{\sqrt{m}} \cdot \sum_{j=1}^{m} (\xi_j - \mathrm{M}\,\xi_1) + \sqrt{m} \cdot \left(\mathrm{M}\,\xi_1 - \rho_i^k\right) \sim \sqrt{m} \cdot \left(\mathrm{M}\,\xi_1 - \rho_i^k\right).$$

Тобто зараз R^k буде зростати пропорційно до величини m, і буде великим у порівнянні з r_α, що призведе до відхилення невірної гіпотези.

1.2 Дані з психології

1.2.1 Типи вищої нервової діяльності

Для визначення того, які показники вимірювати і яким чином, скористуємось відомою класифікацією типів вищої нервової діяльності.

Згідно з Павловим[4] типи вищої нервової діяльності характеризуються трьома показниками: сила нервової системи (сильна або слабка), врівноваженість (врівноважена або неврівноважена) та рухливість (рухлива або інетртна). Павлов розглядає 4 комбінації цих показників з 8 можливих:

1) Слабка
2) Сильна та неврівноважена
3) Сильна, врівноважена та інертна
4) Сильна, врівноважена та рухлива

Далі ці класи (комбінації) будуть називатися відповідно слабкий, неврівноважений, інертний та рухливий.

1.2.2 Теппінг-тест (Tapping rate)

Існують відомі залежності між типом вищої нервової діяльності та зміною максимального темпу рухів кистю руки з часом. Протягом 30 секунд людина намагається притримуватися максимально можливого для себе темпу. Показники темпу фіксуються через кожні 5 секунд, а далі по 6 отриманим точкам будується крива темпа руху. [5]

Для тесту можна використовувати ручку (олівець) і папір, або телеграф. Сучасні технології дозволяють проводити тест за допомогою клавіатури комп'ютера або екрану планшета.

З олівцем і папіром тест поводиться наступним чином:

1) На папері креслиться 6 квадратів
2) Людина починає ставити якомога більше точок в першому квадраті впродовж перших 5 секунд
3) Коли проходить 5 секунд, потрібно перейти до наступного квадрату і ставити точки там
4) Процедура повторюється до тих пір, доки не пройде 30 секунд — в кінці буде заповнено всі 6 квадратів

Далі підраховується кількість точок в кожному квадраті та малюється ламана, де горизонтальна вісь відповідає номеру часового проміжку (номеру квадрата), а вертикальна відповідає кількості точок в квадраті.

Трактуються отримані дані наступним чином:

1) Спадна ламана відповідає слабкому типу (рис. 1.1д). Вона спадає після перших 5 секунд тесту і не повертається до початкового рівня
2) Ламана, що спочатку зростає, а після 10-15 секунд спадає нижче початкового рівня (проміжна між рівною та опуклою) відповідає неврівноваженому типу (рис. 1.1б).

3) Опукла вниз ламана відповідає інертному типу (рис. 1.1г). Вона спочатку спадає, а на 25-30 секундах може зрости до початкового темпу
4) Опукла вгору ламана відповідає рухливому типу (рис. 1.1а). Це така ламана, що зростає в перші 10-15 секунд тесту, а після 25-30 секунд повертається або падає нижче початкового рівня
5) Також темп може залишатися приблизно на одному рівні протягом усього тесту, що є оптимальним для складання іспитів (рис. 1.1в).

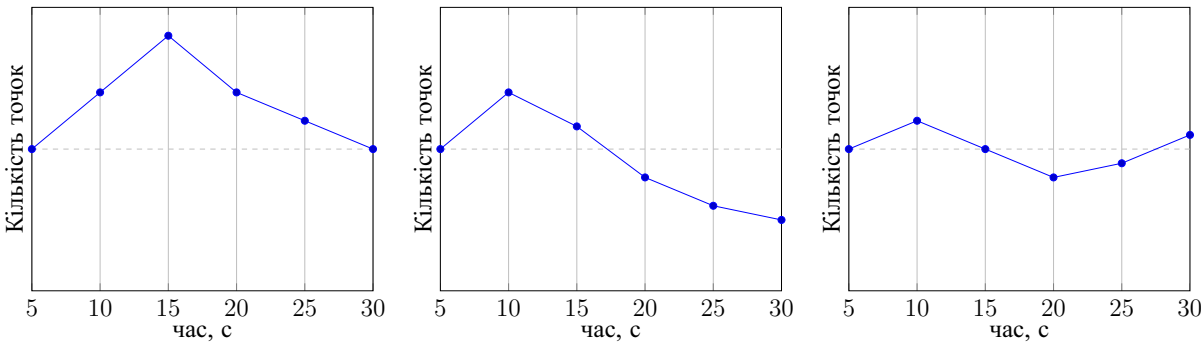
Достовірним відхиленням достатньо вважати різницю в два і більше натиснення між двома сусідніми п’ятисекундними проміжками. [5]

Оскільки цей тест заснований на вимірюванні витривалості нервової системи людини за умови максимального навантаження та перевіряє темп реагування (натиснення) на подразнювачі (внутрішній подразнювач — команда собі “треба тиснути”), було вирішено використовувати відомі вигляди кривих (рис. 1.1) при моделюванні результатів виконання завдань однакової складності.

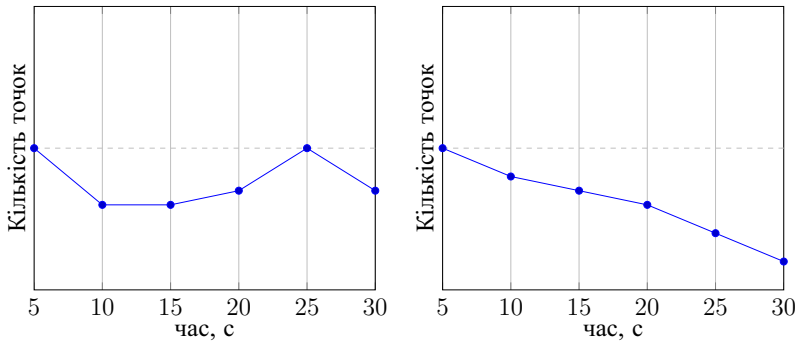
Потрібно зауважити, що швидкість розв’язування задач може змінюватися з досвідом. Тобто, якщо студент зі слабкою нервовою системою буде тренуватися виконувати завдяння, то його показники з часом перейдуть на якісно новий рівень. Щодо студентів з сильною нервовою системою: швидкість не завжди означає якість виконання завдань.

Завдання системи — класифікувати студентів в автоматичному режимі, з метою подальшого надання порад викладачам практичних занять щодо підвищення продуктивності роботи кожного студента. Це полегшує знаходження індивідуального підходу. Наприклад, тим, хто надто швидко втомлюється, потрібно розв’яювати якомога більше базових завдань, що не є складними, але розв’язок яких повинен бути на рівні рефлексів. Студентам, які поспішають, буде корисно ретельно коментувати у письмовій формі хід своїх думок, щоб вгамуватися та підвищити свою уважність.

Для тесту було обрано саме 30-секундний проміжок часу, адже спочатку



(а) Опукла вгору (б) Проміжна (в) Рівна



(г) Опукла вниз (д) Спадна

Рисунок 1.1 — Загальний вигляд залежностей кількості поставлених точок від часу. Пунктирна лінія — кількість точок в перші 5 секунд

виміри виконувалися протягом однієї хвилини і було виявлено, що найважливіша інформація отримується протягом перших 20-25 секунд, а далі лише марно втрачається час та сили тестованого. [5]

Висновки до розділу 1

В даному розділі було досліджено відомі статистичні методи обробки даних та психологічні особливості студентів, що домопожуть створити систему тестування, яка буде більш об’єктивною, ніж класичні аналоги. Також було розпочато моделювання, за допомогою якого будуть виконані перші кроки в створенні та тестуванні системи.

Метод головних компонент буде використано для виявлення найвагоміших

рис (головних компонент) поведінки студентів під час виконання тестів. Для його застосування було вирішено розбити час розв’язання тестових завдань на скінченну кількість відрізків — дискретизувати час.

За допомогою критерію узгодженості Пірсона буде перевірено гіпотези щодо розподілів головних компонент, які були знайдені на попередньому кроці. Оскільки вони будуть лінійними комбінаціями інших елементів виборки, можна припустити, що розподіл буде нормальним — залишиться визначити середнє та дисперсію кожної головної компоненти.

Дані з психології надали матеріал, на основі якого можна розділити людей на кілька категорій, кожна з яких має свої особливості. На основі цього було вирішено, які поради система буде надавати викладачеві практичних занять щодо роботи з різними студентами.

2 МОДЕЛЮВАННЯ РЕЗУЛЬТАТІВ ВИКОНАННЯ ТЕСТОВИХ ЗАВДАНЬ

2.1 Моделювання результатів виконання теппінг-тесту

2.1.1 Процес Пуассона

Щоб моделювати темп виконання студентами контрольних задач, використаємо відомі дані з психології щодо ритму виконання теппінг-тесту для людей з різними типами вищої нервової діяльності. Оскільки постукування є потоком однорідних випадкових подій, змоделюємо їх як реалізації процесу Пуассона.

Нехай $\{N(t)\,|t \geq 0\}$ — процес Пуассона з інтенсивністю $\lambda(t)$. Маємо 7 контрольних точок на часовій вісі $t_0 = 0, t_1 = 5, \ldots, t_6 = 30$. Введемо відрізок часу T, за який проводиться вимірювання, та проміжки T_i, на які його розбито в експерименті

$$T = (0, 30], \qquad T_i = (t_{i-1}, t_i], \qquad i = \overline{1,6}.$$

Нехай ν_i визначається наступним чином

$$\nu_i = N(t_i) - N(t_{i-1}), \qquad i = \overline{1,6},$$

тоді ν_i — випадкова величина, що має розподіл Пуассона з параметром λ_i, де [6]

$$\lambda_i = m(T_i) = \int_{t_{i-1}}^{t_i} \lambda(\tau)\ d\tau, \qquad i = \overline{1,6}.$$

Функція m — міра інтенсивності. [7]

Проміжки T_i утворюють розбиття множини T, а отже не перетинаються. Це означає, що випадкові величини ν_i незалежні між собою, і є можливість згенерувати процес натискання за допомогою 6 незалежних випадкових величин, що мають розподіл Пуассона.

Інтенсивність (параметр λ) є математичним очікуванням випадкової величини, що має розподіл Пуассона. Маючи результати реальних експериментів, можна обчислити середні значення кількості постукувань для кожного часового проміжку T_i, а отримані результати використовувати в якості параметрів λ_i.

Є наступні відомості щодо розподілу типів:

- Слабкий — 29%,
- Неврівноважений — 38%,
- Рухливий — 23%,
- Інертний — 10%.

Спростимо ці результати та додамо змішаний тип

- Змішаний — 3%,
- Слабкий — 25%,
- Неврівноважений — 35%,
- Рухливий — 25%,
- Інертний — 12%.

На основі цих спостережень за допомогою крітерію Пірсона було підібрано необхідну інтенсивність процесу Пуассона. На рис. 2.2а зображено різні реалізації послідовності випадкових величин, що розподілені за законом Пуассона з параметрами

$$[\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5, \lambda_6] = [40, 41, 46, 38, 35, 34]\,. \tag{2.1}$$

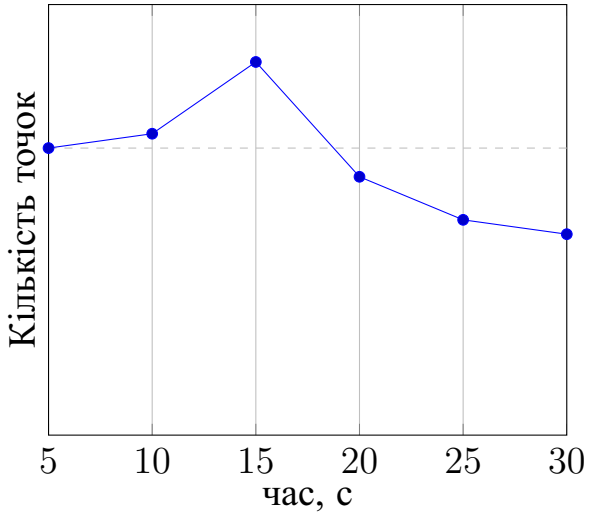


Рисунок 2.1 — Опукла ламана

2.1.2 Квадратична апроксимація методом найменших квадратів

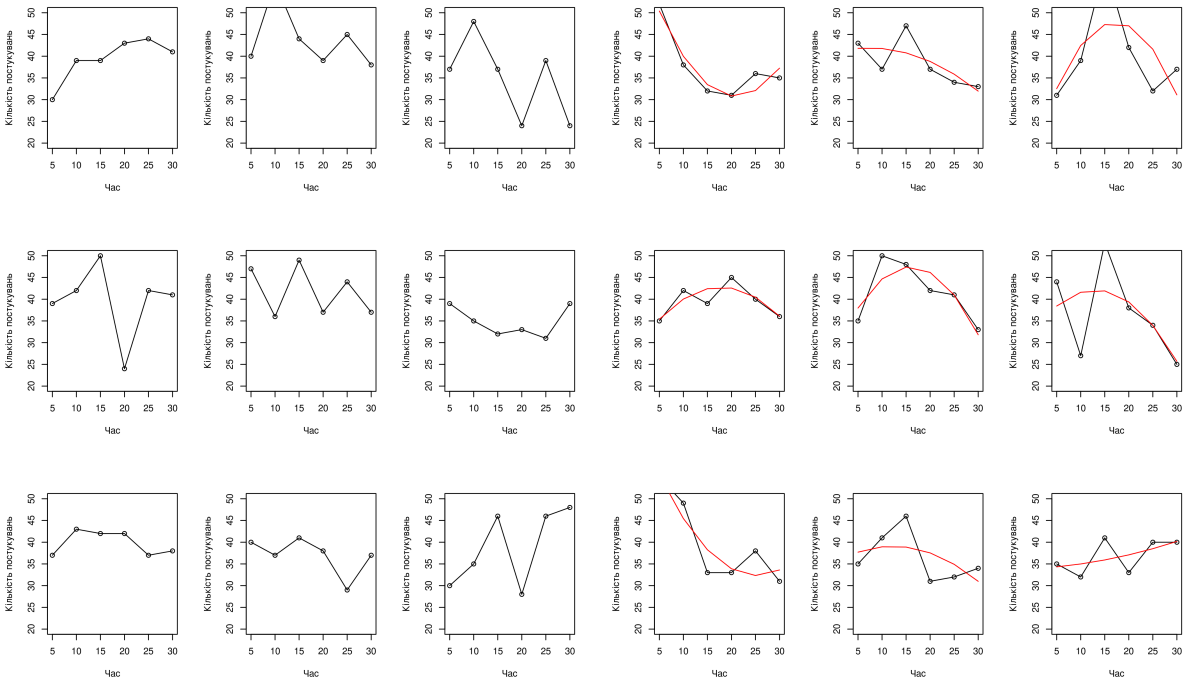
Види результуючих ламаних (опукла вгору, опукла вниз, рівна, спадна) можна представити квадратичними функціями виду

$$y\left(t\right) = a \cdot t^2 + b \cdot t + c.$$

На рис. 2.2б зображено реалізації послідовності випадкових величин, що розподілені за законом Пуассона, та криві, що є результатом апроксимації цих реалізацій.

Щоб представити результати у вигляді квадратичних функцій, використаємо метод найменших квадратів. Нехай $\left\{y_i \mid i = \overline{1,6}\right\}$ — результати поточного теппінг-тесту. Суть методу полягає в знаходженні таких коефіцієнтів a, b і c, з якими наступна величина приймає мінімальне значення

$$R^2\left(a,b,c\right) = \sum_{i=1}^{6} \left(a \cdot t_i^2 + b \cdot t_i + c - y_i\right)^2.$$



(а) Без апроксимації (б) З апроксимацією

Рисунок 2.2 — Приклади результатів моделювання

Для цього треба взяти часткові похідні по кожному коефіцієнту, прирівняти ці похідні до нуля та розв’язати отриману систему

$$\begin{cases} \sum_{i=1}^{6} 2 \cdot \left(a \cdot t_i^2 + b \cdot t_i + c - y_i\right) & = 0 \\ \sum_{i=1}^{6} 2 \cdot t_i \cdot \left(a \cdot t_i^2 + b \cdot t_i + c - y_i\right) & = 0 \\ \sum_{i=1}^{6} 2 \cdot t_i^2 \cdot \left(a \cdot t_i^2 + b \cdot t_i + c - y_i\right) & = 0. \end{cases}$$

Цю систему можна переписати в матричному вигляді наступним чином

$$\begin{bmatrix} \sum_{i=1}^{6} t_i^2 & \sum_{i=1}^{6} t_i & \sum_{i=1}^{6} 1 \\ \sum_{i=1}^{6} t_i^3 & \sum_{i=1}^{6} t_i^2 & \sum_{i=1}^{6} t_i \\ \sum_{i=1}^{6} t_i^4 & \sum_{i=1}^{6} t_i^3 & \sum_{i=1}^{6} t_i^2 \end{bmatrix} \cdot \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{6} y_i \\ \sum_{i=1}^{6} y_i \cdot t_i \\ \sum_{i=1}^{6} y_i \cdot t_i^2 \end{bmatrix}.$$

Введемо позначення:

$$\sum_{i=1}^{6} 1 = N,$$
$$\sum_{i=1}^{6} t_i^k = X^k, \qquad k = \overline{1,4},$$
$$\sum_{i=1}^{6} y_i \cdot t_i^k = YX^k, \qquad k = \overline{0,2}.$$

Рівняння прийняло наступний вигляд

$$\begin{bmatrix} X^2 & X & N \\ X^3 & X^2 & X \\ X^4 & X^3 & X^2 \end{bmatrix} \cdot \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} Y \\ YX \\ YX^2 \end{bmatrix}.$$

Використаємо метод Крамера для розв'язання системи лінійних рівнянь. Визначник Δ

$$\Delta = X^2 \cdot \left(X^2 \cdot X^2 - X \cdot X^3\right) - X \cdot \left(X^3 \cdot X^2 - X \cdot X^4\right) + N \cdot \left(X^3 \cdot X^3 - X^2 \cdot X^4\right).$$

Визначник Δ_a

$$\Delta_a = Y \cdot \left(X^2 \cdot X^2 - X \cdot X^3\right) - YX \cdot \left(X \cdot X^2 - N \cdot X^3\right) + YX^2 \cdot \left(X \cdot X - N \cdot X^2\right).$$

Визначник Δ_b

$$\Delta_b = -Y \cdot \left(X^3 \cdot X^2 - X \cdot X^4\right) + YX \cdot \left(X^2 \cdot X^2 - N \cdot X^4\right) - YX^2 \cdot \left(X^2 \cdot X - N \cdot X^3\right).$$

Визначник Δ_c

$$\Delta_c = Y \cdot \left(X^3 \cdot X^3 - X^2 \cdot X^4\right) - YX \cdot \left(X^2 \cdot X^3 - X \cdot X^4\right) + YX^2 \cdot \left(X^2 \cdot X^2 - X \cdot X^3\right).$$

Відомо, що розв'язками є наступні вирази

$$a = \frac{\Delta_a}{\Delta}, \qquad b = \frac{\Delta_b}{\Delta}, \qquad c = \frac{\Delta_c}{\Delta}.$$

Бачимо, що параметри — лінійні комбінації y_i, а коефіцієнти при y_i — раціональні дроби. Тобто можна записати розв'язки наступним чином

$$a = \sum_{i=1}^{6} y_i \cdot \frac{a_i}{\Delta}, \qquad b = \sum_{i=1}^{6} y_i \cdot \frac{b_i}{\Delta}, \qquad c = \sum_{i=1}^{6} y_i \cdot \frac{c_i}{\Delta}.$$

2.1.3 Аналіз квадратичної апроксимації

2.1.3.1 Задача

Запишемо умови, що накладаються на a, b і c, за яких парабола

$$y(t) = a \cdot t^2 + b \cdot t + c$$

відповідає раніше визначеним типам вищої нервової діяльності. Введемо такі позначення: t_{max} і t_{min} — точки, в яких парабола досягає локального максимуму та мінімуму на відрізку $[t_1, t_6]$ відповідно (t_1 і t_6 є точками першого та останнього спостереження).

Позначимо характеристики параболи, на основі яких буде відбуватися класифікація:

1) сумарне відхилення кривої на проміжку $\tau = \left[t^1; t^2\right]$ від положення $y\left(t^1\right)$ рахуємо за формулою
$$\sigma(\tau) = \int_{t^1}^{t^2} |\dot{y}(t)|\, dt;$$
2) від знаку a залежить те, чи t_g максимум або мінімум
$$M = sign(a);$$
3) якщо знаки похідних на кінцях відрізку $[t_1; t_6]$ різні, то функція досягає екстремуму саме на цьому відрізку
$$V = sign(\dot{y}(t_1) \cdot \dot{y}(t_6));$$
4) глобального екстремуму функція y досягає в точці
$$t_g = -\frac{b}{2 \cdot a};$$
5) позначимо площу трикутника, що обмежується прямими $t = t_1$, $t = t_g$, $y = \dot{y}(t)$ та $y = 0$
$$\sigma_S = \sigma([t_1; t_g]) = \frac{|(t_g - t_1) \cdot \dot{y}(t_1)|}{2},$$

а також площу трикутника, що обмежується прямими $t = t_6$, $t = t_g$, $y = \dot{y}(t)$ та $y = 0$

$$\sigma_F = \sigma([t_6; t_g]) = \frac{|(t_g - t_6) \cdot \dot{y}(t_6)|}{2};$$

6) найбільше відхилення від початкового положення за тієї умови, що глобальний екстремум знаходиться на відрізку $[t_1; t_6]$ ($V = -1$), рахується за формулою

$$\sigma_M = \max\{\sigma_S, \sigma_F\};$$

7) якщо глобальний екстремум досягається за межами відрізку $[t_1; t_6]$ ($V = 1$), відхилення буде наступним

$$\sigma_P = \max\{\sigma_S, \sigma_F\} - \min\{\sigma_S, \sigma_F\},$$

адже геометрично це буде різниця площ більшого і меншого трикутників, що були утворені прямими $t = t_1$, $t = t_6$, $t = t_g$, $y = \dot{y}(t)$ та $y = 0$;

2.1.3.2 Змішаний тип

Крива, що відповідає змішаному типу, весь час тримається приблизно на початковому рівні. Тобто, для певного ε, яке зазвичай беруть рівним 2, маємо такі випадки:

1) Функція y не монотонна на відрізку T

$$\begin{cases} V = -1, \\ \sigma_M < \varepsilon; \end{cases}$$

2) Функція y монотонна на відрізку T

$$\begin{cases} V = 1, \\ \sigma_P < \varepsilon. \end{cases}$$

2.1.3.3 Слабкий тип

Крива, що відповідає слабкому типу, спадає після моменту t_1 і не повертається до початкового положення.

Можливі наступні випадки:

1) t_g — точка глобального мінімуму

$$\begin{cases} V = 1, \\ t_g > t_6, \\ M = 1, \\ \sigma_P > \varepsilon; \end{cases}$$

2) t_g — точка глобального максимуму

$$\begin{cases} V = 1, \\ t_g < t_6, \\ M = -1, \\ \sigma_P > \varepsilon. \end{cases}$$

2.1.3.4 Неврівноважений тип

Крива, що відповідає неврівноваженому типу, спочатку зростає, потім не пізніше моменту часу t_3 починає спадати нижче початкового рівня

$$\begin{cases} M = -1, \\ V = -1, \\ t_g < t_3, \\ \sigma_F > \varepsilon. \end{cases}$$

2.1.3.5 Рухливий тип

Крива, що відповідає рухливому типу, спочатку зростає, не пізніше моменту t_3 починає спадати, а до моменту t_6 повертається до початкового рівня або спадає нижче нього

$$\begin{cases} M = -1, \\ V = -1, \\ t_g \geq t_3, \\ \sigma_F - \sigma_S > 0. \end{cases}$$

2.1.3.6 Інертний тип

Крива, що відповідає інертному типу, спочатку спадає, а під кінець тесту зростає до початкового рівня

$$\begin{cases} M = 1, \\ V = -1, \\ 0 \leq \sigma_S - \sigma_F \geq -\varepsilon. \end{cases}$$

2.1.3.7 Результати класифікації

Було згенеровано 6 вибірок по 100 реалізацій процесу Пуассона з інтенсивністю, яку було вказано у формулі (2.1). Ці реалізації відповідають проходженню теппінг-тесту шістьма групами по сто студентів. Результати класифікації можна побачити на рис. 2.3.

Опис:

- Не вдалося віднести до того чи іншого типу 11% студентів.
- 2% студентів мають змішаний тип вищої нервової діяльності;
- Слабкий тип зустрівся у 18% випадків;
- 34% студентів відносяться до неврівноваженого типу;
- 20% — студенти рухливого типу;
- Студенти інертного типу зустрілися у 14% випадків;

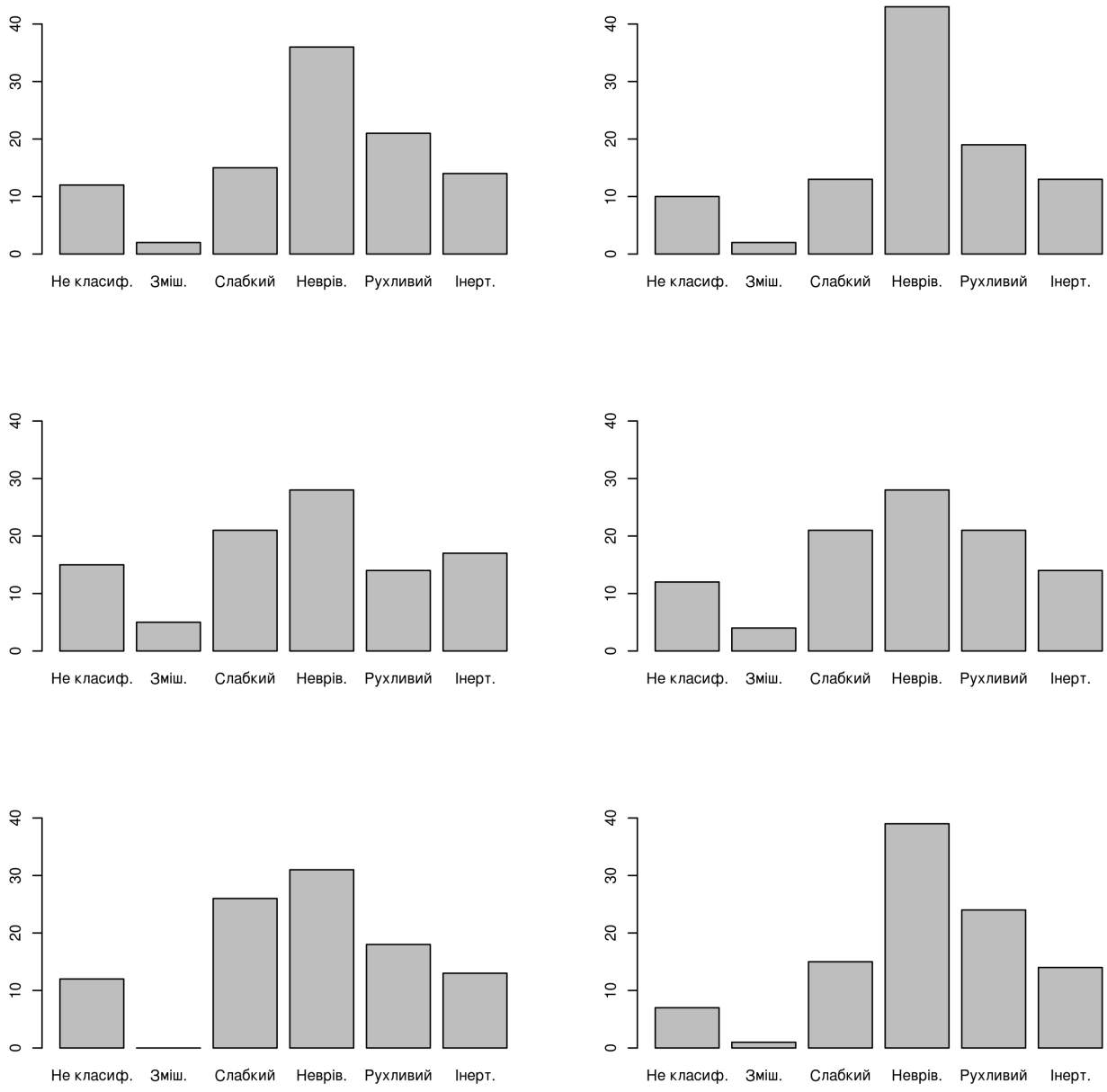


Рисунок 2.3 — Результати класифікації

2.2 Моделювання результатів виконання тестових завдань

2.2.1 Інформаційний метод прогнозування часу розв'язання завдань

Один з методів прогнозування часу розв'язання завдання людиною, що використовується в інженерній психології, є інформаційний метод. Вважаємо, що

студент виконує завдання одне за одним. Кожне завдання має певну складність H_i — кількість інформації, що потрібно обробити для розв'язку завдання. Студент виконує кожне завдання з певною швидкістю V_i. Тоді час виконання тесту τ буде рахуватися за формулою

$$\tau = \sum_{i=1}^{l} \frac{H_i}{V_i},$$

де l — кількість задач. [8]

Вважаємо, що швидкість виконання завдання i рахується за формулою

$$V_i(t) = \frac{1}{\tau_i},$$

де τ_i — випадкова величина, що має показниковий розподіл з параметром λ_i, який залежить від типу вищої нервової діяльності людини та часу, що затрачено на виконання попередніх завдань:

$$\tau_i \sim Exp(\lambda_i).$$

Тоді τ_i буде часом, що витрачається на одиницю інформації завдання

$$\tau = \sum_{i=1}^{l} H_i \cdot \tau_i.$$

Якщо всі завдання мають однакову складність $H_i = H$, рівність приймає вигляд

$$\tau = H \cdot \sum_{i=1}^{l} \tau_i.$$

Щоб застосувати отримані з моделювання теппінг-тесту результаті, потрібно горизонтально масштабувати параболи, до яких було аппроксимовано ламані. Початкова парабола y має вигляд

$$y(t) = a^2 \cdot t + b \cdot t + c.$$

Потрібно отримати параболу λ

$$\lambda(t) = A^2 \cdot t + B \cdot t + C$$

таку, для якої виконується

$$\begin{aligned} \lambda(0) &= y(0), \\ \lambda(T) &= y(t_t), \\ \lambda\left(\frac{T}{2}\right) &= y\left(\frac{t_t}{2}\right) \end{aligned}$$

t_t — час виконання теппінг-тесту (30 секунд), T — час виконання тестових завдань (40 або 90 хвилин). Маємо розв'язок

$$\begin{aligned} A &= a \cdot \left(\frac{t_t}{T}\right)^2, \\ B &= b \cdot \frac{t_t}{T}, \\ C &= c, \end{aligned}$$

тобто

$$\lambda(t) = a \cdot \left(\frac{t_t}{T}\right)^2 \cdot t^2 + b \cdot \frac{t_t}{T} \cdot t + c.$$

Отже маємо

$$\tau_1 \sim Exp\left\{\lambda\left(0\right)\right\},$$
$$\tau_i \sim Exp\left\{\lambda\left(\sum_{j=1}^{i} H_j \cdot \tau_j\right)\right\}, \qquad i > 1.$$

2.2.2 Моделювання тесту

Було змодельовано тестування на півтори години з 10 задачами, кожна з яких має складність $H = 0.5$. Номер задачі, яку виконує студент на даний момент, фиксується кожні 10 секунд. Щоб отримати достатньо інформації для головних компонент, було змодельовано проходження тестування 600 студентами.

На рис. 2.4 можна побачити дисперсії перших десяти головних компонент. Майже половину інформації (44.6%) можна отримати з першої компоненти. На рис. 2.5 наведено гістограми перших чотирьох головних компонент. Але з рис. 2.6 видно, що доцільно використовувати лише першу компоненту, а також не вдасться відрізнити від інших класів змішаний та неврівноважений тип. На рис. 2.7 зображено склад першої головної компоненти з усіма класами та без змішаного і неврівноваженого типів. За допомогою алгоритму CART (Classification and Regression Tree) було класифіковано дві групи студентів:

1) Ті, кому треба більше часу думати і не поспішати при виконанні завдань, щоб запобігти помилок — рухливий клас, а також не класифіковані студенти, адже в основному не вдалося віднести до того чи іншого типу тих студентів, активність яких не спадає з часом.
2) Ті, кому треба більше уваги приділяти елементарним задачам, щоб довести до автоматизму їх розв'язання і не витрачати час на зайві дії — це студенти, що відносяться до слабкого та інертного типу.

Дерево класифікації зображено на рис. 2.8. Як можна побачити, критерій досить простий:

- ті студенти, перший головний крітерій яких перевищує поріг 17.155, відносяться з ймовірністю біля 90% до тих, яким треба більше тренуватися;
- ті студенти, перший головний крітерій яких не перевищує поріг 17.155, відносяться з ймовірністю біля 90% до тих, яким треба більше думати.

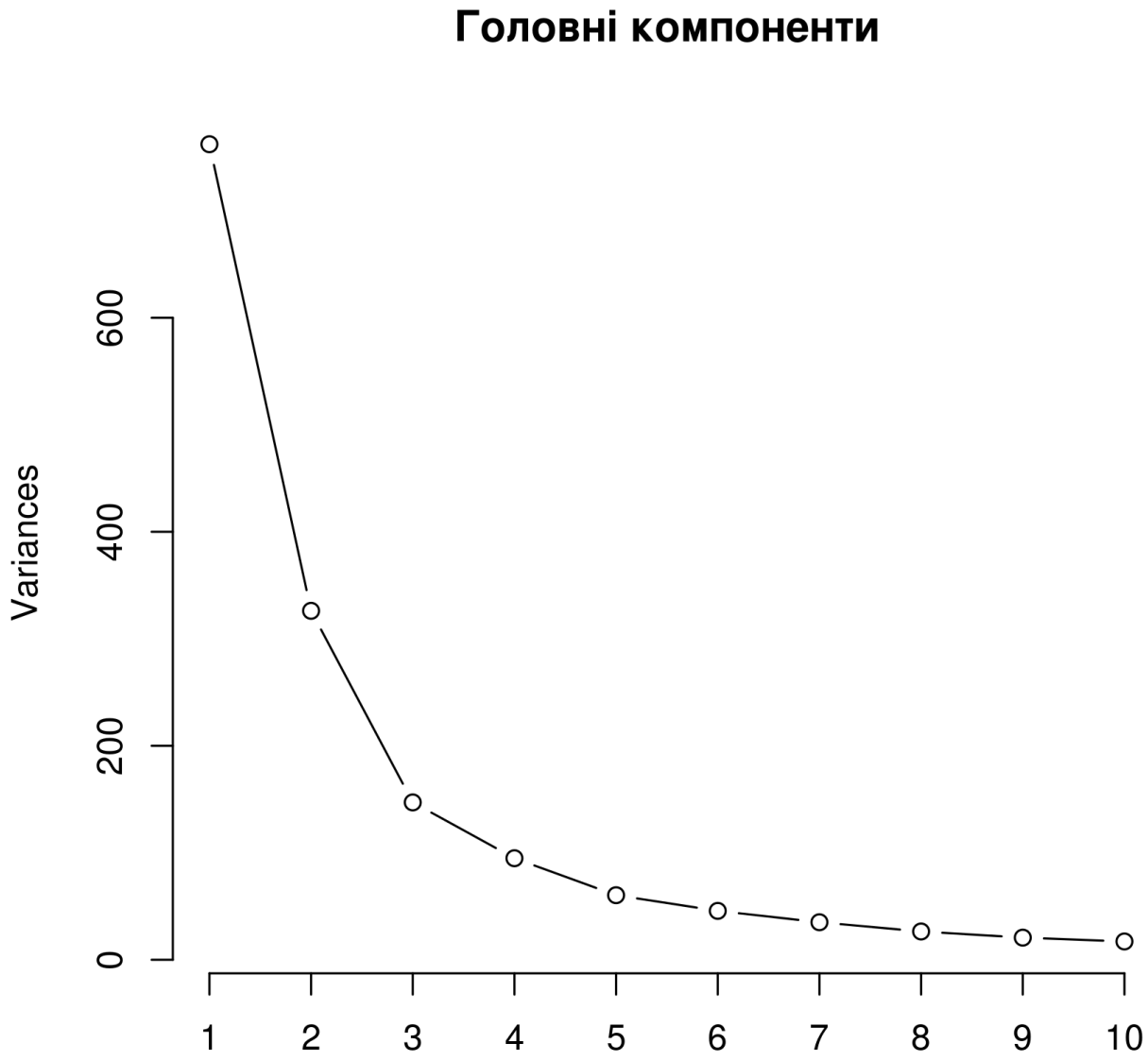


Рисунок 2.4 — Дисперсії першого десятку головних компонент

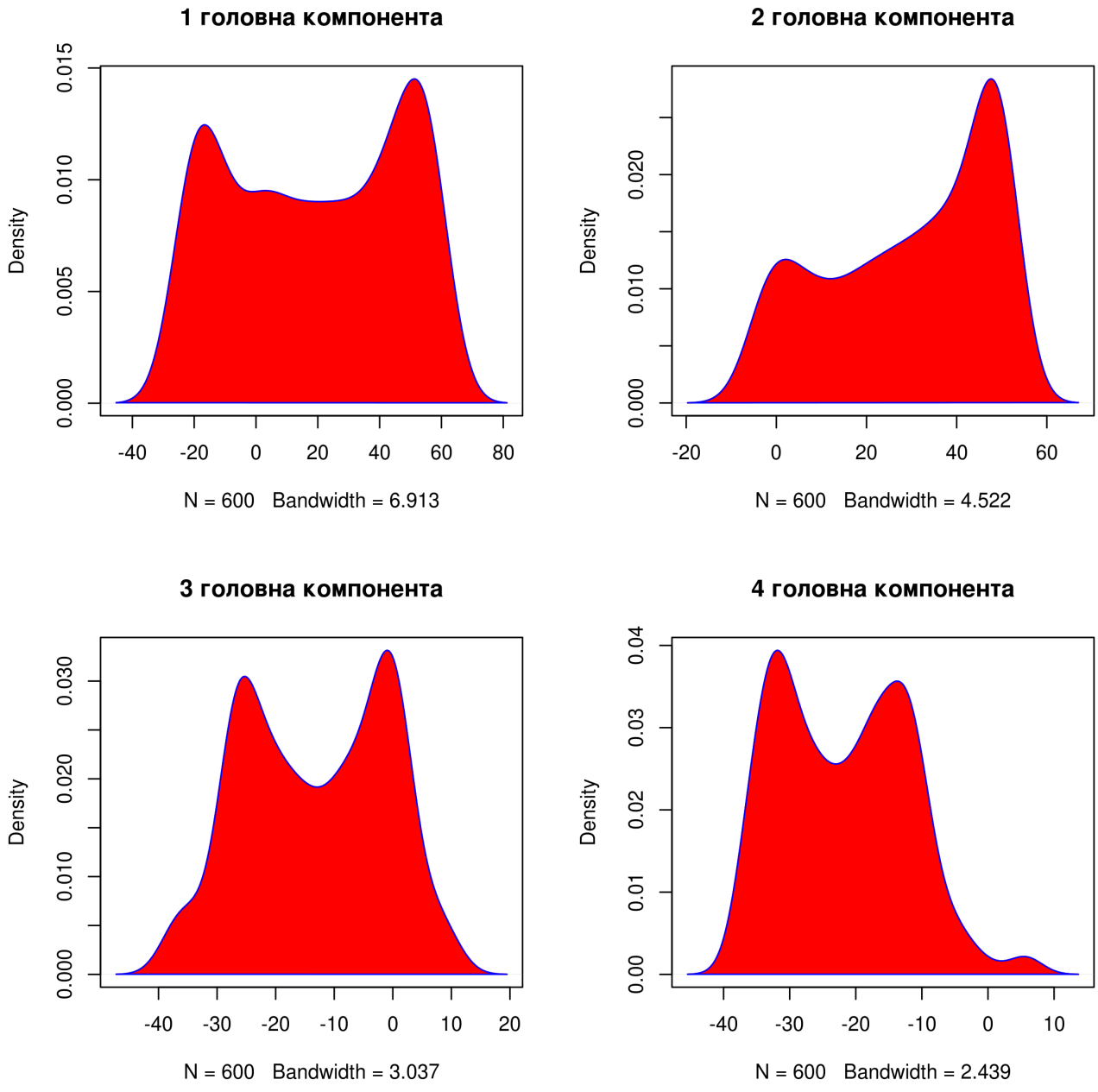


Рисунок 2.5 — Гістограми першої четвірки головних компонент

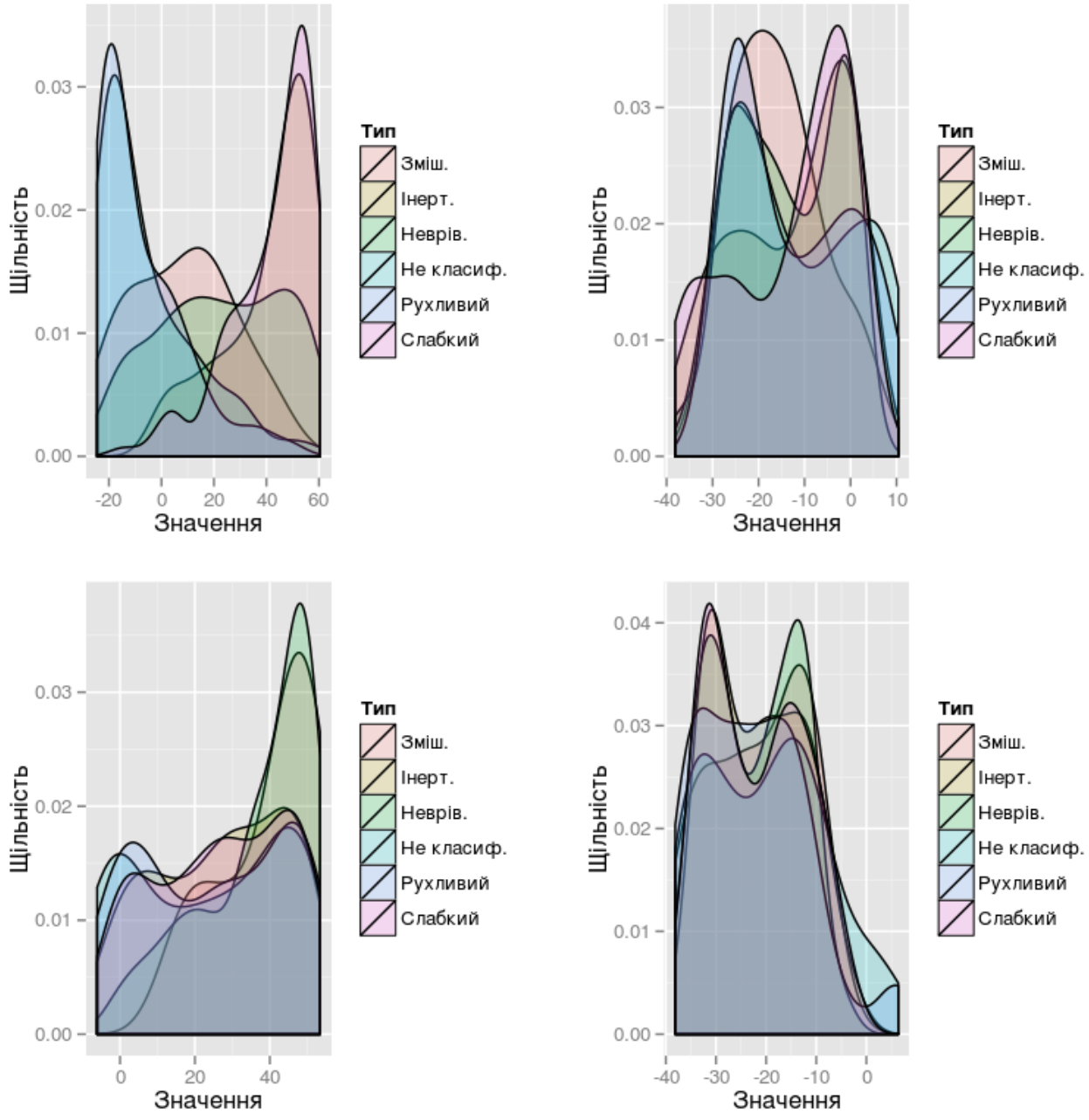


Рисунок 2.6 — Детализовані гістограми першої четвірки головних компонент

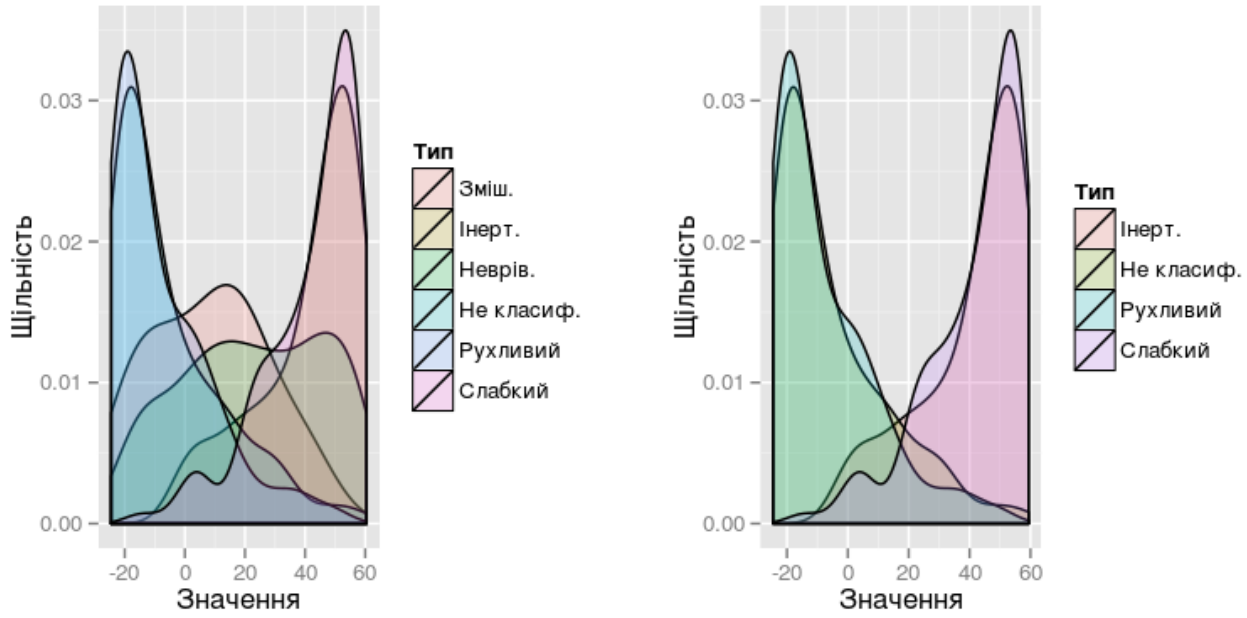


Рисунок 2.7 — Детализовані гістограми першої головної компоненти

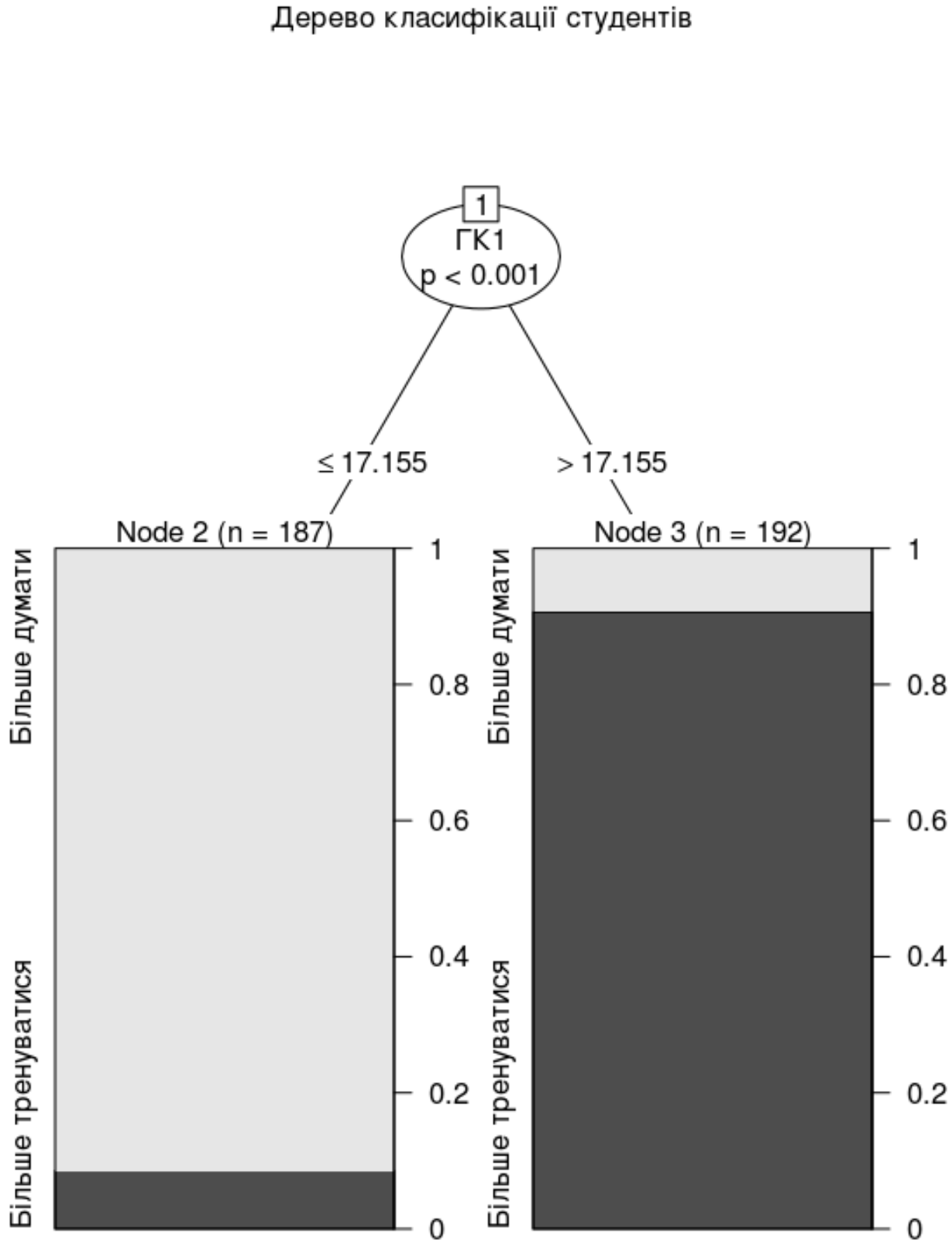


Рисунок 2.8 — Дерево класифікації

ПЕРЕЛІК ПОСИЛАНЬ

1. Айвазян, С.А. Прикладная статистика: Классификация и снижение размерности : Справочное издание / С.А. Айвазян. — М.: Финансы и статистика, 1989. — 606 с.

2. Боровков, А. А. Математическая статистика / А. А. Боровков // Учебники для ВУЗов. Специальная литература. — СПб.: Лань, 2010. — 705 с.

3. Sturges, Herbert A. The Choice of a Class Interval / Herbert A. Sturges // *j-J-AM-STAT-ASSOC*. — 1926. — March. — Vol. 21, no. 153. — Pp. 65–66.

4. Павлов, И.П. Двадцатилетний опыт объективного изучения высшей деятельности (поведения) животных / И.П. Павлов. — М.: Рипол Классик, 1973. — 659 с.

5. Ильин, Е.П. Дифференциальная психофизиология / Е.П. Ильин // Серия Учебник нового века. — СПб.: Питер, 2001. — 464 с.

6. Булинский, А.В. Теория случайных процессов / А.В. Булинский, А.Н. Ширяев // Теория вероятностей. Математическая статистика : ТВМС. — М.: Физматлит, 2003. — 399 с.

7. Kingman, J.F.C. Poisson Processes / J.F.C. Kingman // Oxford studies in probability. — Oxford: Clarendon Press, 1992. — 112 pp.

8. Ломов, Б. Ф. Справочник по инженерной психологии / Б. Ф. Ломов. — Москва: Машиностроение, 1982. — 368 с.

ДОДАТОК А

Лістинг коду програми, яка моделює результати теппінг-тесту

```
source("generate_sequence.r")
source("analyze_approximation.r")
wideScreen()
# Read command line arguments
args <- commandArgs(TRUE)
display_approximation <- FALSE
if (length(args) > 0 && args[1] == "approximation") {
    display_approximation <- TRUE
}
# Init cells for charts
n <- 3
m <- 3
png('output.png', width=18, height=22, units="cm", res=300)
par(mfrow=c(n,m))
chart.xlab <- "Час"
chart.ylab <- "Кількість постукувань"
chart.ylim <- c(20,50)
# Parameters
sample.time <- seq(5, 30, 5)
sample.size = length(sample.time)
sample.prediction_degree = 2
sample.poly = poly(sample.time, sample.prediction_degree, raw=TRUE)
lambda.default = c(40, 41, 46, 38, 35, 34)
# Fill every cell
for (i in 1:(n*m)) {
    # Generate non-homogeneous Poisson process trajectory
    sample.current <- generate_trajectory(lambda.default)
    # Plot
    plot(sample.time, sample.current, xlab=chart.xlab, ylab=chart.ylab,
         ylim=chart.ylim)
    lines(sample.time, sample.current)
    if (display_approximation) {
        # Calculate approximation
        abc <- find_abc(sample.current)
        f <- function(t) {
            return(abc[1]*t^2+abc[2]*t+abc[3])
        }
        # Draw approximation
        lines(sample.time, Map(f, sample.time), xlab=chart.xlab, ylab=chart.ylab,
         ylim=chart.ylim, col='red')
        # Draw start rhythm
        #abline(h=sample.model$coefficients[3]*sample.time[1]^2 +
        #         sample.model$coefficients[2]*sample.time[1] +
        #         sample.model$coefficients[1], col='green', lty=2, lwd=1)
        # Draw minumum and maximum
```

```
46          #abline(v=extrema[['min']], col='gray', lty=3, lwd=1)
47          #abline(v=extrema[['max']], col='blue', lty=3, lwd=1)
48          # Display group
49          #print(get_group(sample.model, sample.time,
50          #                extrema[['min']], extrema[['max']]))
51      }
52  }
53  dev.off()
```

ДОДАТОК Б

Лістинг коду модуля, що генерує процес Пуассона

```
generate_trajectory <- function(lambda.default) {
    # Generate non-homogeneous Poisson process
    return(rpois(length(lambda.default), lambda.default))
}
```

Лістинг коду модуля з функцією, що розтягує параболу

```
t.stretch.start  <- 0
t.stretch.finish <- 30

parabola.stretch <- function (a, b, c, t.finish.new) {
    t.k <- t.stretch.finish/t.finish.new
    function (t) {
        x <- t.k * t
        a*x^2+b*x+c
    }
}
```

Лістинг коду модуля з функцією, що рахує відстань для крітерію Пірсона χ^2

```
chi_squared.distance <- function(m, h, p) {
    return(m * sum((h - p)^2 / p))
}
```

ДОДАТОК В

Лістинг коду програми, яка класифікує моделі студентів за типами вищої нервової діяльності

```
t.start <- 5
t.middle <- 15
t.finish <- 30
t.diff <- t.finish - t.start

X    <- seq(5, 30, 5)
X2   <- X^2
X3   <- X^3
X4   <- X^4
N    <- 6

SX   <- sum(X)
SX2  <- sum(X2)
SX3  <- sum(X3)
SX4  <- sum(X4)
SN   <- N

delta <- SX2 * (SX2^2 - SX * SX3) - SX * (SX3 * SX2 - SX * SX4) +
         SN * (SX3^2 - SX2 * SX4)

delta.a.k <- c(SX2^2 -SX * SX3, -(SX * SX2 - SN * SX3), SX * SX - SN * SX2)/delta

delta.b.k <- c(-(SX3 * SX2 -SX * SX4), SX2^2 - SN * SX4, -(SX2 * SX - SN * SX3))/delta

delta.c.k <- c(SX3^2 - SX2 * SX4, -(SX2 * SX3 - SX * SX4), SX2 * SX2 - SX * SX3)/delta

#MX <- matrix(c(SX2, SX3, SX4, SX, SX2, SX3, SN, SX, SX2), nrow=3, ncol=3)

find_abc <- function(sample) {
    SY.k <- c(sum(sample), sample %*% X, sample %*% X2)
    return(c(SY.k %*% delta.a.k, SY.k %*% delta.b.k, SY.k %*% delta.c.k))
}

get_group <- function(a, b, c) {
    EPSILON <- 2
    if (a == 0) {
        diff <- - b * t.diff
        if (diff < -EPSILON) {
            return(1)
        } else if (diff < EPSILON) {
            return(0)
        }
        return(-1)
```

```
44      }
45
46      d1 <- (2*a*t.start + b)
47      d6 <- (2*a*t.finish + b)
48      M <- sign(a)
49      V <- sign(d1*d6)
50      t.g <- - b / (2*a)
51      sigma.s <- abs((t.g-t.start) * d1)/2
52      sigma.f <- abs((t.g-t.finish) * d6)/2
53      sigma.m <- max(sigma.s, sigma.f)
54      sigma.p <- sigma.m - min(sigma.s, sigma.f)
55
56      if ((V == -1 && sigma.m < EPSILON) ||
57          (V == 1 && sigma.p < EPSILON)) {
58          return(0)
59      } else if (V == 1 && sigma.p > EPSILON && (
60                  (t.g < t.start && M == -1) ||
61                  (t.g > t.finish && M == 1)  ||
62                  (a == 0 && b < 0))) {
63          return(1)
64      } else if (M == -1 && V == -1 && t.g < t.middle &&
65                  sigma.f > EPSILON) {
66          return(2)
67      } else if (M == -1 && V == -1 && t.g >= t.middle &&
68                  sigma.f - sigma.s > 0) {
69          return(3)
70      } else if (M == 1 && V == -1 && (sigma.s - sigma.f >= -EPSILON)) {
71          return(4)
72      } else {
73          return(-1)
74      }
75  }
```

ДОДАТОК Г

Лістинг коду програми, яка генерує студентів та будує гістограми

```
source("generate_sequence.r")
source("analyze_approximation.r")
source("chi_squared.r")

n       <- 100
rows    <- 3
columns <- 2
sample.time <- seq(5, 30, 5)
sample.size = length(sample.time)
sample.prediction_degree = 2
sample.poly = poly(sample.time, sample.prediction_degree, raw=TRUE)
lambda.default = c(40, 41, 46, 38, 35, 34)
needed.percentage = c(.1, .03, .25, .35, .25, .12)

groups.number = 4
groups.names = c("Не класиф.", "Зміш.", "Слабкий", "Неврів.", "Рухливий", "Інерт.")

global.groups = rep(0, length(groups.number))
global.percentage = rep(0, length(groups.number))
global.distances = rep(0, length(groups.number))

png('output.png', width=22, height=22, units="cm", res=300)
par(mfrow=c(rows,columns))

result = rep(-1, n)
res <- rep(0, groups.number+2)
for (j in 2:(rows*columns+1)) {
    for (i in 1:n) {
        ## Generate non-homogeneous Poisson process trajectory
        sample.current <- generate_trajectory(lambda.default)
        # Calculate approximation
        abc <- find_abc(sample.current)
        # Get group
        result[i] <- get_group(abc[1], abc[2], abc[3])
    }
    elements_in_group <- function(x) length(result[result==x])
    groups <- unlist(Map(elements_in_group, -1:groups.number))
    print(sprintf("%.2f", groups/n))
    print(sprintf("%.2f", chi_squared.distance(n, tail(groups/n, 5), tail(needed.percentage,5))))
    barplot(groups, ylim=c(0, 40), names.arg=groups.names)
    global.groups <- global.groups + groups
}
print('Result:')
global.percentage = global.groups / (n*rows*columns)
print(sprintf("%.2f", global.percentage))
```

```
46  print(sprintf("%.2f", chi_squared.distance(n, tail(global.percentage, 5), tail(needed.percentage,5))))
```

ДОДАТОК Д

Лістинг коду програми, яка синтезує параметр λ

```
library("iterators")
library("parallel")
library("foreach")
library("doParallel")

source("generate_sequence.r")
source("analyze_approximation.r")
source("chi_squared.r")

n       <- 1000
rows    <- 3
columns <- 2
sample.time <- seq(5, 30, 5)
sample.size = length(sample.time)
sample.prediction_degree = 2
sample.poly = poly(sample.time, sample.prediction_degree, raw=TRUE)
lambda.default = rep(40, 6)

groups.number <- 4
needed.percentage = c(.03, .25, .35, .25, .12)

dist.min <- 20

calculate.lambda <- function(lambda.current) {
    result = rep(-1, n)
    for (i in 1:n) {
        ## Generate non-homogeneous Poisson process trajectory
        sample.current <- generate_trajectory(lambda.current)
        # Calculate approximation
        abc <- find_abc(sample.current)
        # Get group
        result[i] = get_group(abc[1], abc[2], abc[3])
    }
    elements_in_group <- function(x) length(result[result==x])
    groups <- unlist(Map(elements_in_group, 0:groups.number))
    distance <- chi_squared.distance(n, groups/n, needed.percentage)
    if (distance < dist.min) {
        cat("Lambda:", sprintf("%d,", lambda.current),
            sprintf("Distance: %.2f,", distance),
            "Groups:", sprintf("%.2f,", groups/n),
            "\n")
        dist.min <<- distance
    }
    return()
}
```

```
46
47 iterate.lambdas.seq <- function(lambda.current, lambda.pos) {
48     if (lambda.pos == 2) {
49         lambda.min <- 0
50         lambda.max <- 5
51     } else if (lambda.pos == 3) {
52         lambda.min <- -5
53         lambda.max <- 10
54     } else if (lambda.pos == 4) {
55         lambda.min <- -15
56         lambda.max <- -5
57     } else if (lambda.pos == 5) {
58         lambda.min <- -5
59         lambda.max <- 0
60     } else if (lambda.pos == 6) {
61         lambda.min <- -5
62         lambda.max <- 0
63     }
64
65     get_lambda <- function (i) {
66         lambda.current[lambda.pos] <- lambda.current[lambda.pos-1] + i
67         return(lambda.current)
68     }
69     lambdas <- Map(get_lambda, lambda.min:lambda.max)
70     for (l in lambdas) {
71         iterate.lambdas(l, lambda.pos+1)
72     }
73     return()
74 }
75
76 iterate.lambdas <- function(lambda.current, lambda.pos) {
77     if (lambda.pos == 7) {
78         calculate.lambda(lambda.current)
79     } else {
80         iterate.lambdas.seq(lambda.current, lambda.pos)
81     }
82     return()
83 }
84
85 lambda.current <- lambda.default
86 lambda.min <- 0
87 lambda.max <- 5
88 get_lambda <- function (i) {
89     lambda.current[2] <- lambda.current[1] + i
90     return(lambda.current)
91 }
92 lambdas <- Map(get_lambda, lambda.min:lambda.max)
93 cl <- makeCluster(detectCores(), outfile="")
94 registerDoParallel(cl, cores = detectCores() - 1)
```

```
95  foreach(i=lambdas, .options.multicore=c(silent=FALSE)) %dopar% {
96      source("generate_sequence.r")
97      source("analyze_approximation.r")
98      source("chi_squared.r")
99      options(warn=1)
100     iterate.lambdas(i, 3)
101 }
102 stopCluster(cl)
```

ДОДАТОК Е

Лістинг коду програми, яка моделює та аналізує результати тестування

```
library('ggplot2')
library('party')

source("generate_sequence.r")
source("analyze_approximation.r")
source("modify_parabola.r")
options(warn=1)

lambda.default = c(40, 41, 46, 38, 35, 34)

exam.cpm     <- 6
exam.minutes <- 90
exam.length  <- exam.cpm * exam.minutes

task.N <- 10
#task.H <- exam.length / (task.N*mean(lambda.default))
task.H <- exam.length / (task.N*mean(lambda.default)*3)
tasks  <- rep(task.H, task.N)

exam.pass <- function (intensity) {
    get.tau <- function (t) {
        result <- rexp(1, intensity(t))
        if (is.nan(result)) print(t)
        result
    }
    exam.H.passed <- rep(NA, exam.length)
    exam.H.passed[1] <- get.tau(1)
    for (i in 2:exam.length) {
        exam.H.passed[i] <- get.tau(i) + exam.H.passed[i-1]
    }
    (function () {
        is.positive <- function (x) {
            x > 0
        }
        .exam.row <- function (accumulator, task.n.current, exam.H.left) {
            tmp <- exam.H.left - task.H
            positive <- tmp[is.positive(tmp)]
            negative <- tmp[Negate(is.positive)(tmp)]
            if (length(positive) == 0 || task.n.current == task.N) {
                c(accumulator, rep(task.n.current, length(negative)), rep(0, length(positive)))
            } else {
                .exam.row(c(accumulator, rep(task.n.current, length(negative))), task.n.current+1, positive)
            }
        }
        .exam.row(c(), 1, exam.H.passed)
```

```
    })()
}

get.some <- function (n) {
    result <- c()
    groups <- rep(-1, n)
    values <- matrix(nrow=n, ncol=exam.length)
    abc <- NA
    for (i in 1:n) {
        repeat {
            abc <- find_abc(generate_trajectory(lambda.default))
            if (abc[3] > 0) break
        }
        new.parabola <- parabola.stretch(abc[1], abc[2], abc[3], exam.length)
        groups[i] <- get_group(abc[1], abc[2], abc[3])
        values[i,] <- exam.pass(new.parabola)
    }
    list(values=values, groups=groups)
}

draw.some <- function(i) {
    plot(density(result$values %*% students.pca$rotation[,i]), col='red', main=paste(i, "головна компонента"))
    polygon(density(result$values %*% students.pca$rotation[,i]), col='red', border='blue')
}

get.group <- function (sample, group.number) {
    sample$values[sample$groups == group.number,]
}

groups.names <- c("Не класиф.", "Зміш.", "Слабкий",
                  "Неврів.", "Рухливий", "Інерт.")

get.chart.g <- function(sample, i, groups) {
     pca.distribution <- Reduce(function (result, g) {
        rbind(result, data.frame(Значення
              =c(get.group(sample, g) %*% students.pca$rotation[,i]),Тип
              =groups.names[g+2], stringsAsFactors=FALSE))
    }, groups, c())
    ggplot(pca.distribution, aesЗначення(, fillТип=)) +
           ylab("Щільність") +
           geom_density(alpha = 0.2)
}

get.chart <- function(sample, i) {
    get.chart.g(sample, i, (seq(length(groups.names))-2))
}

get.amount <- function(sample, pc, group.number, f) {
    fpc <- get.group(sample, group.number) %*% students.pca$rotation[,pc]
```

```
    fpc[f(fpc)]
}

result <- get.some(600)

students.pca <- prcomp(result$values, center = TRUE)
plot(students.pca, type = "l", title="Головні компоненти")
print(summary(students.pca))

get.strict.pc <- function (sample, pc) {
    sample$values[is.element(sample$groups, c(-1, 1, 3, 4)),]%*%students.pca$rotation[,1]
}

get.students <- function (sample) {
    data.frame(group=unlist(Map(
    function(x) {
        if (x == -1 || x == 3) {
            'Більше думати'
        } else if (x == 1 || x == 4) {
            'Більше тренуватися'
        } else {
            NA
        }
    },
    sample$groups[is.element(sample$groups, c(-1, 1, 3, 4))])),ГК
    1=get.strict.pc(sample, 1))
}

students <- data.frame(
    group=unlist(Map(function(x) {
        if (x == -1 || x == 3) {
            'Більше думати'
        } else if (x == 1 || x == 4) {
            'Більше тренуватися'
        } else {
            NA
        }
    },
    result$groups[is.element(result$groups, c(-1, 1, 3, 4))])),ГК
    1=get.strict.pc(result, 1))


fit.visual <- ctree(group ~ ГК1, data=students, controls=ctree_control(maxdepth=1))
fit.good <- ctree(group ~ ГК1, data=students)

plot(fit.visual, main="Дерево класифікації студентів")

#treeresponse(fit.good, newdata=students[1:10,])
```