Міністерство освіти і науки України
Національний університет “Львівська політехніка”
Інститут комп’ютерних наук та інформаційних технологій
Кафедра програмного забезпечення

Звіт
Про виконання лабораторної роботи №9

На тему:

«Наближення функції методом найменших квадратів»
з дисципліни «Чисельні методи»

Лектор:
доцент каф. ПЗ
Мельник Н. Б.
Виконав:
ст. гр. ПЗ-11
Морозов О. Р.
Прийняла:
доцент каф. ПЗ
Мельник Н. Б.

« __ » _____ 2022 р.

∑ = _____ _____ .

Львів – 2022

Тема: наближення функції методом найменших квадратів.
Мета: ознайомлення на практиці з методом найменших квадратів апроксимації (наближення) функцій.

Теоретичні відомості

Метод найменших квадратів - основна ідея цього методу полягає в пошуку полінома фіксованого степеня m:
$P(x) = a_0 + a_1x + a_2x^2 + \ldots + a_mx^m$,

з середньоквадратичною похибкою, $\sigma = \sqrt{\frac{1}{n+1}\sum_{i=0}^{n}(P_m(x_i) - y_i)^2}$.

Так як квадрат кожного відхилення повинен бути мінімальним:

$$\Phi(a_0, a_1, a_2, \ldots a_m) = \sum_{i=0}^{n}\left(P_m(x_i) - y_i\right)^2 = \sum_{i=0}^{n}\left(\sum_{j=0}^{m} a_j x_i^j - y_i\right)^2;$$

Далі виконаємо умову $\frac{\delta\Phi}{\delta a_i}$ = 0, та отримуємо нормальну систему рівнянь $\sum_{j=0}^{m}\left(\sum_{i=0}^{n} x_i^{j+k}\right)a_j = \sum_{k=0}^{n} y_i x_i^k$, де $k = \overline{0,m}$ Отримана таким методом система буде симетричною, а визначник – відмінний від нуля. Розв'язок нормальної системи шукають, використовуючи методи розв'язування СЛАР.

Індивідуальне завдання

Варіант 2

x	0,15	0,20	0,25	0,30	0,40	0,50
y	4,48	5,47	6,05	7,39	8,11	9,93

Хід роботи

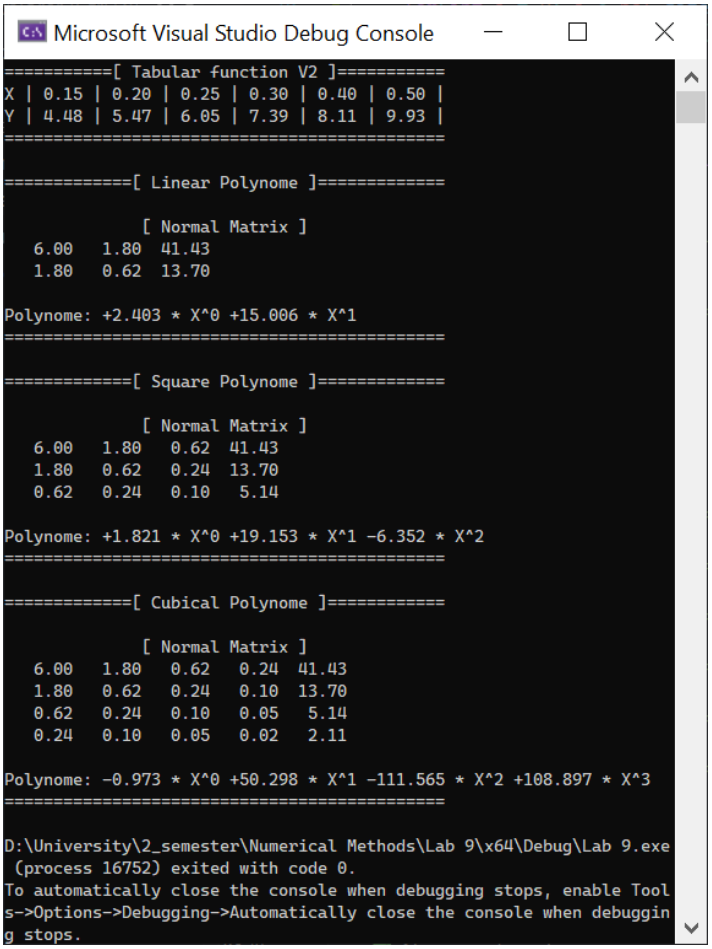
Код програми:

```
#include <iostream>
#include <iomanip>
#include <cmath>
#include <vector>

using namespace std;

vector<vector<double>> TMatrix(vector<vector<double>> A) {
    vector<vector<double>> At(A.size(), vector<double>(A.size()));

    for(int i = 0; i < A.size(); i++) {
        for(int j = 0; j < A[i].size(); j++) {
            At[j][i] = A[i][j];
        }
    }
    return At;
}

vector<double> SqrtRootMethod(vector<vector<double>> N) {
	const int size = N.size();

	vector<vector<double>> l(size, vector<double>(size));
	vector<vector<double>> lt(size, vector<double>(size));

	for(int i = 0; i < size; i++) {
		double tmp = 0;

		for(int k = 0; k < i; k++) {
			tmp += lt[k][i] * lt[k][i];
		}

		lt[i][i] = sqrt(N[i][i] - tmp);

		for(int j = i; j < size; j++) {
			tmp = 0;

			for(int k = 0; k < i; k++)
				tmp += lt[k][i] * lt[k][j];

			lt[i][j] = (N[i][j] - tmp) / lt[i][i];
		}
	}

	l = TMatrix(lt);
```

```
	vector<double> Y(size);                                 //search Y

	for(int i = 0; i < size; i++) {
		double sum = 0;
		for(unsigned j = 0; j < i; j++) {
			sum += Y[j] * l[i][j];
		}
		Y[i] = (N[i].back() - sum) / l[i][i];
	}

	vector<double> X(size);                                 //search X

	for(int i = 0; i < size; i++)
		X[i] = 0;

	for(int i = size-1; i >= 0; i--) {
		double sum = 0;

		for(int j = i + 1; j < size; j++) {
			sum += X[j] * l[j][i];
		}

		X[i] = (Y[i] - sum) / l[i][i];
	}

	return X;
}

void PrintMatrix(vector<vector<double>> E) {
	cout << noshowpos;
	for(int i = 0; i < E.size(); ++i) {
		for(int j = 0; j < E[i].size(); ++j)
			cout << fixed << setprecision(2) << setw(7) << E[i][j];
		cout << endl;
	}
}

void PrintPolinome(vector<double> pol) {
	cout << endl << fixed << setprecision(4) << setw(6) << "Polynome: ";

	for(int i = 0; i < pol.size(); i++)
		cout << showpos << pol[i] << " * X^" << noshowpos << i << " ";

	cout << noshowpos << endl;
}

void MinSqrtMethod(vector<double> X, vector<double> Y, int m) {

	int n = X.size();

    vector<vector<double>> slarNormal(m + 1, vector<double>(m+2));
```

```
	for(int k = 0; k < m + 1; k++) {
		for(int j = 0; j < m + 1; j++) {
			for(int i = 0; i < n; i++) {
				slarNormal[k][j] += pow(X[i], j + k);
			}
		}
		for(int i = 0; i < n; i++) {
			slarNormal[k][m + 1] += Y[i] * pow(X[i], k);
		}
	}
	cout << endl << "\t[ Normal Matrix ]" << endl;
	PrintMatrix(slarNormal);

	vector<double> pol = SqrtRootMethod(slarNormal);
	PrintPolinome(pol);
}

int main() {
    cout << "===========[ Tabular function V2 ]==========" << endl <<
            "X | 0.15 | 0.20 | 0.25 | 0.30 | 0.40 | 0.50 |" << endl <<
            "Y | 4.48 | 5.47 | 6.05 | 7.39 | 8.11 | 9.93 |" << endl <<
            "============================================" << endl;

    vector<double> X = { 0.15, 0.20, 0.25, 0.30, 0.40, 0.50 };
    vector<double> Y = { 4.48, 5.47, 6.05, 7.39, 8.11, 9.93 };

	cout << endl << "=============[ Linear Polynome ]=============" << endl;
	MinSqrtMethod(X, Y, 1);
	cout << "============================================" << endl;

	cout << endl << "=============[ Square Polynome ]=============" << endl;
	MinSqrtMethod(X, Y, 2);
	cout << "============================================" << endl;

	cout << endl << "=============[ Cubical Polynome ]============" << endl;
	MinSqrtMethod(X, Y, 3);
	cout << "============================================" << endl;

    return 0;
}
```

Результат:



Висновок

Виконуючи лабораторну роботу №9, я ознайомився на практиці з методом найменших квадратів апроксимації (наближення) функцій, та склав програму для побудови лінійного, квадратичного і кубічного апроксимаційних поліномів.