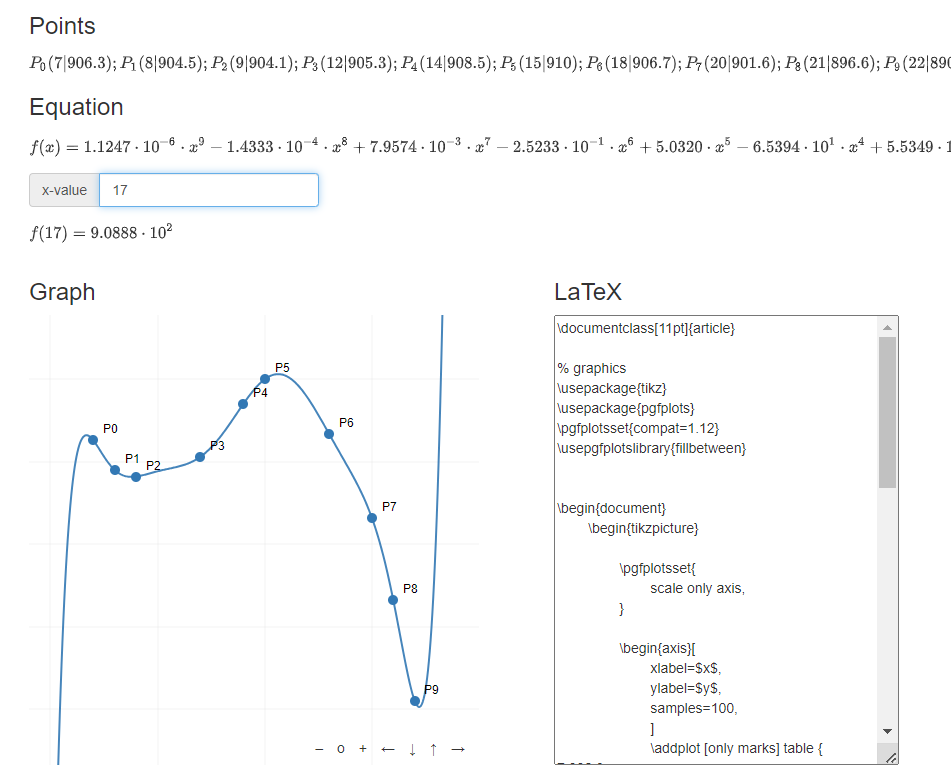


Власна умова:

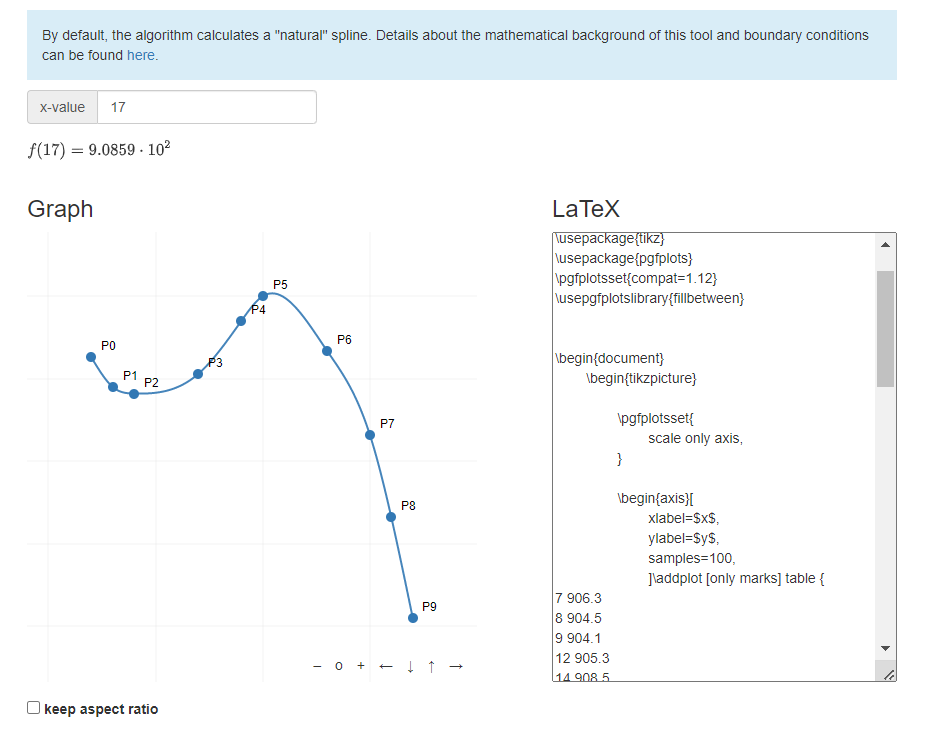
Чисельність населення Чернівецької області протягом 2007-2022 років:

|  |  |  |  |
| --- | --- | --- | --- |
| на 1.01.2007 | 906,3 |  |  |
| на 1.01.2008 | 904,5 |  |  |
| на 1.01.2009 | 904,1 |  |  |
| на 1.01.2012 | 905,3 |  |  |
| на 1.01.2014 | 908,5 |  |  |
| на 1.01.2015 | 910,0 |  |  |
| на 1.01.2018 | 906,7 |  |  |
| на 1.01.2020 | 901,6 |  |  |
| на 1.01.2021 | 896,6 |  |  |
| на 1.01.2022 | 890,5 |

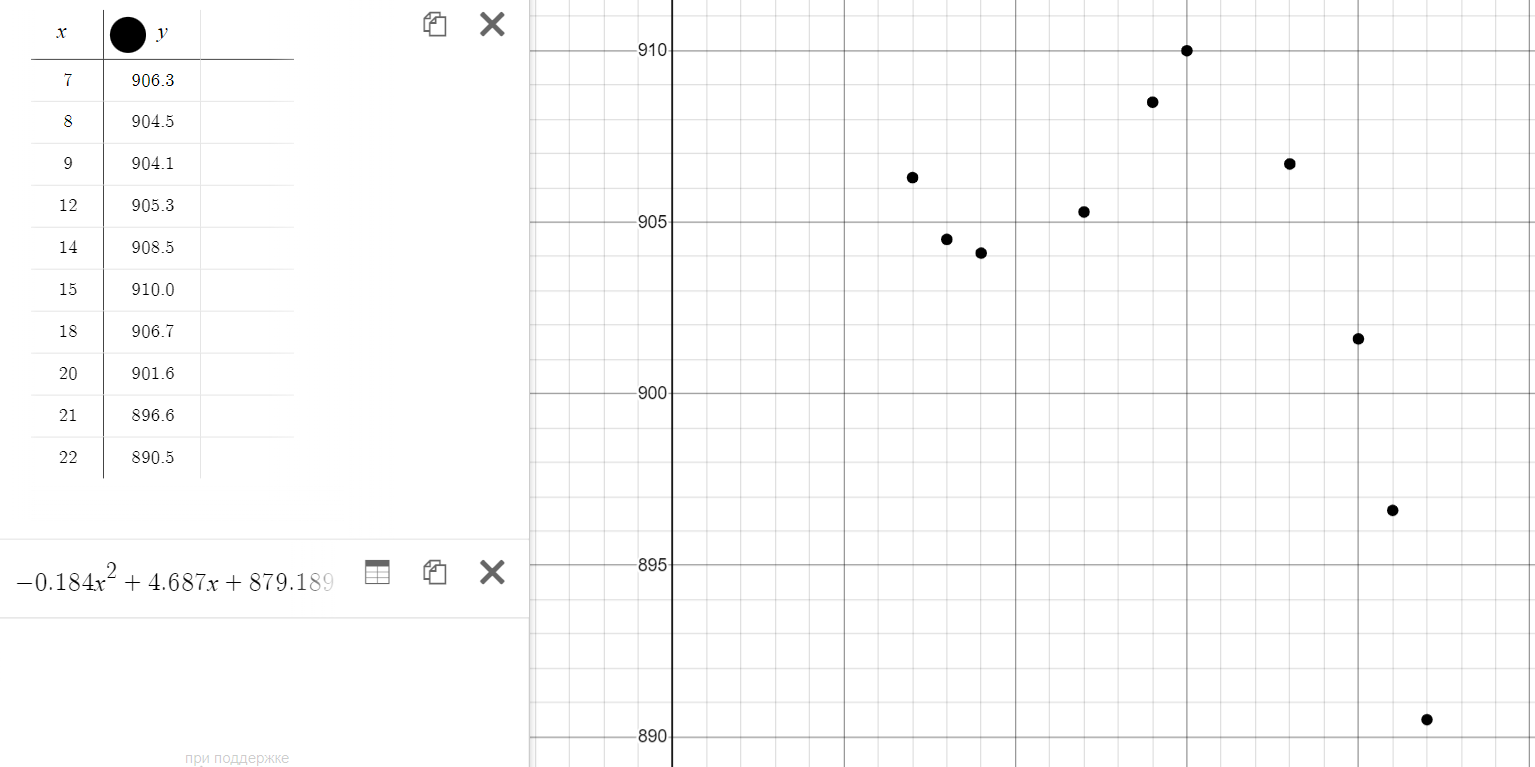
1) Проінтерполювати функцію в деякій точці 𝑥 ≠ 𝑥[𝑖]. Поліноміальну інтерполяцію можна виконати в деякій комп’ютерній системі.



2) За допомогою кубічної сплайн-інтерполяції апроксимувати функцію в деякій комп’ютерній системі.

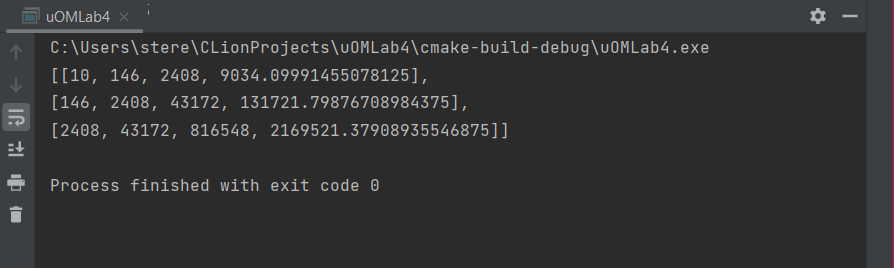


3) Методом найменших квадратів побудувати найкраще середньоквадратичне наближення підібравши відповідну функцію.



Код програми, що знаходить матрицю для параболи:

#include <iostream>  
#include <cmath>  
#include <iomanip>  
  
using namespace std;  
  
void output\_Matrix(int n, int m, double matrix[3][4]){  
 cout << "[";  
 for(int i = 0; i < n - 1; ++i){  
 cout << "[";  
 for(int j = 0; j < m - 1; ++j){  
 cout << matrix[i][j] << ", ";  
 }  
 cout << matrix[i][m-1] << "]," << endl;  
 }  
 cout << "[";  
 for(int j = 0; j < m-1; ++j){  
 cout << matrix[n-1][j] << ", ";  
 }  
 cout << matrix[n-1][m-1];  
 cout << "]]" << endl;  
}  
  
int main() {  
 cout << setprecision(50);  
 float matrix[20][2] = {{7, 906.3},  
 {8, 904.5},  
 {9, 904.1},  
 {12, 905.3},  
 {14, 908.5},  
 {15, 910.0},  
 {18, 906.7},  
 {20, 901.6},  
 {21, 896.6},  
 {22, 890.5}};  
 int n = 10;  
 double summary\_x = 0, summary\_x2 = 0, summary\_y = 0, summary\_xy = 0, summary\_x3 = 0, summary\_x4 = 0, summary\_x2y = 0;  
 for (int i = 0; i <= n; ++i){  
 summary\_y += matrix[i][1];  
 summary\_x2 += pow(matrix[i][0], 2);  
 summary\_x3 += pow(matrix[i][0], 3);  
 summary\_x4 += pow(matrix[i][0], 4);  
 summary\_x += matrix[i][0];  
 summary\_xy += matrix[i][1] \* matrix[i][0];  
 summary\_x2y += pow(matrix[i][0], 2) \* matrix[i][1];  
 }  
 double answer[3][4];  
 answer[0][0] = n;  
 answer[0][1] = answer[1][0] = summary\_x;  
 answer[0][2] = answer[2][0] = answer[1][1] = summary\_x2;  
 answer[1][2] = answer[2][1] = summary\_x3;  
 answer[2][2] = summary\_x4;  
 answer[0][3] = summary\_y;  
 answer[1][3] = summary\_xy;  
 answer[2][3] = summary\_x2y;  
 output\_Matrix(3, 4, answer);  
 return 0;  
}



Код програми, що обчислює невідомі з матриці та похибки:

def gauss\_jordan(m, eps=0.00000000000001):

(h, w) = (len(m), len(m[0]))

for y in range(0, h):

maxrow = y

for y2 in range(y + 1, h):

if abs(m[y2][y]) > abs(m[maxrow][y]):

maxrow = y2

(m[y], m[maxrow]) = (m[maxrow], m[y])

if abs(m[y][y]) <= eps: # Перевіряє чи одинична матриця

return False

for y2 in range(y + 1, h):

c = m[y2][y] / m[y][y]

for x in range(y, w):

m[y2][x] -= m[y][x] \* c

for y in range(h - 1, 0 - 1, -1):

c = m[y][y]

for y2 in range(0, y):

for x in range(w - 1, y - 1, -1):

m[y2][x] -= m[y][x] \* m[y2][y] / c

m[y][y] /= c

for x in range(h, w):

m[y][x] /= c

return True

mtx = [[10, 146, 2408, 9034.09991455078125],

[146, 2408, 43172, 131721.79876708984375],

[2408, 43172, 816548, 2169521.37908935546875]]

def gauss\_print(m):

(h, w) = (len(m), len(m[0]))

v = 1

while v != h + 1:

print("x", v, "=", m[v - 1][w - 1])

v += 1

if gauss\_jordan(mtx):

gauss\_print(mtx)

date = [[7, 906.3],

[8, 904.5],

[9, 904.1],

[12, 905.3],

[14, 908.5],

[15, 910.0],

[18, 906.7],

[20, 901.6],

[21, 896.6],

[22, 890.5]]

def funk(x):

return mtx[0][3] + mtx[1][3] \* x + mtx[2][3] \* (x \*\* 2)

fk\_fik = []

fk\_fik\_2 = []

for i in range(0, 10):

fk\_fik.append(abs(date[i][1] - funk(date[i][0])))

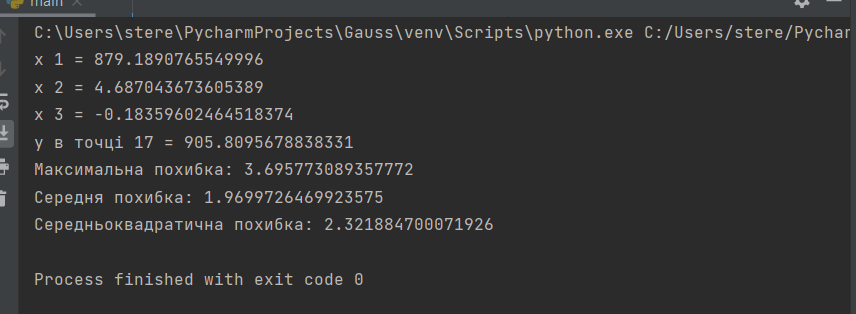
fk\_fik\_2.append(abs(date[i][1] - funk(date[i][0])) \*\* 2)

print("y в точці 17 =", funk(17))

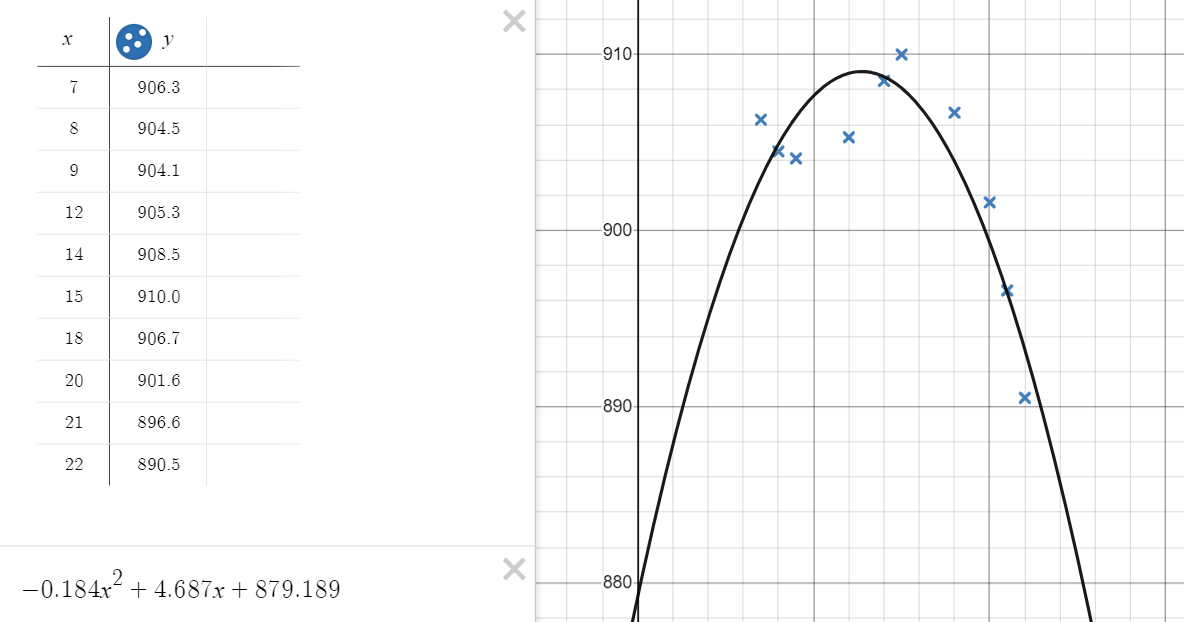
print("Максимальна похибка:", max(fk\_fik))

print("Середня похибка:", sum(fk\_fik)/10)

print("Середньоквадратична похибка:", (sum(fk\_fik\_2) / 10) \*\* (1/2))



4) Зобразити результати в графічному вигляді.



Висновки: Перевіривши правильність результатів третього пункту в онлайн калькуляторі, результати збіглись. Зрівнявши три методи обчислення невідомого числа з вибірки, можна сказати, що вони всі дають різні результати, однак перші два способи більш схожі одне до одного, ніж третій. Тому варто зауважити, що  до точного значення, що також відоме нам і це 908.1, найбільш наближеним буде результат кубічної-сплайн інтерполяції, тому її найдоречніше використати в цьому випадку.