**Національний Технічний Університет України “КПІ”**

**Навчально-науковий комплекс**

**«Інститут прикладного системного аналізу»**

ЛАБОРАТОРНА РОБОТА № 2

З дисципліни: Основи системного аналізу

# Виконали:

# Барзій Ілля Лєсніков Богдан

Шрам Владислав

(Бригада 1)

# група КА-41

Київ 2017

**Задание**

В каждом из вариантов заданы:

* исходные таблицы данных;
* размерности векторов x1, x2, x3 - n1,n2,n3;
* количество выборок n (для расчета);
* количество целевых функций m;
* веса целевых функций;
* метод решения несовместных систем уравнений, в которых число неизвестных не равно числу уравнений;

Наборы P1, P2, P3 – степени полинома Чебышева (Лежандра, Лагера, Эрмита и др.) для x1, x2, x3 – следует подобрать самостоятельно, исходя из критерия минимума максимального отклонения функций:.

Требуется:

1. сформировать целевые функции и вывести на печать

* значения всех промежуточных коэффициентов (λ, a, c) и функций (Ψ,Φ);
* вид полученных функций  через: 1); 2) полиномы Чебышева;

3) в форме обычных многочленов (целесообразно это предусмотреть в файле результатов) как в нормированном, так и в восстановленном виде;

1. построить графики исходных (по выборкам) и полученных функций;
2. оценить погрешность восстановленных функций  по отношению к исходной заданной выборке.

## 4) сделать письменный отчет о выполненной работе, включив листинг программы.

**Варіант 1**

Размерности векторов X1, X2, Х3 – n1=2, n2=2, n3=3;

количество выборок для расчета = 45;

количество целевых функций m=4;

 принимаются равными нормированным значениям ;

метод решения несовместной системы уравнений: **метод сопряженных направлений.**

**Метод спряженних напрямків розв’язку  
СЛНР**

Ненульові вектори *u* и *v* спряжені (відносно А), якщо:

 \mathbf{u}^\mathrm{T} \mathbf{A} \mathbf{v} = 0. 

Оскільки **A** – симетрична та додатньо визначена, визначимо внутрішній добуток:

 \langle \mathbf{u},\mathbf{v} \rangle_\mathbf{A} := \langle \mathbf{A} \mathbf{u}, \mathbf{v}\rangle = \langle \mathbf{u},  \mathbf{A}^\mathrm{T} \mathbf{v}\rangle = \langle \mathbf{u}, \mathbf{A}\mathbf{v} \rangle = \mathbf{u}^\mathrm{T} \mathbf{A} \mathbf{v}. 

Два вектора спряжені, якщо вони ортогональні за вінутрішнім добутком. Спряженість – симетрична властивість.

Нехай, P=\{\mathbf{p}_k: \forall i\neq k, i, k \in [1,n], \langle\mathbf{p}_i,\mathbf{p}_k\rangle_{A}=0  \} - набір *n* попарно спряжених напрямків. Тоді P - базис в \mathbb{R}^n, так можемо визначити розв’язок \mathbf{x}_* з \mathbf{Ax} = \mathbf{b}:

 \mathbf{x}_* = \sum^{n}_{i=1} \alpha_i \mathbf{p}_i

Видно, що:

 \mathbf{b}=\mathbf{A}\mathbf{x}_* = \sum^{n}_{i=1} \alpha_i  \mathbf{A} \mathbf{p}_i.

Для будь-якого\mathbf{p}_k \in P,

 \mathbf{p}_k^\mathrm{T} \mathbf{b}=\mathbf{p}_k^\mathrm{T} \mathbf{A}\mathbf{x}_* = \sum^{n}_{i=1} \alpha_i\mathbf{p}_k^\mathrm{T} \mathbf{A} \mathbf{p}_i=\alpha_k\mathbf{p}_k^\mathrm{T} \mathbf{A} \mathbf{p}_k.

(так як \forall i\neq k, p_i, p_k попарно спряжені)

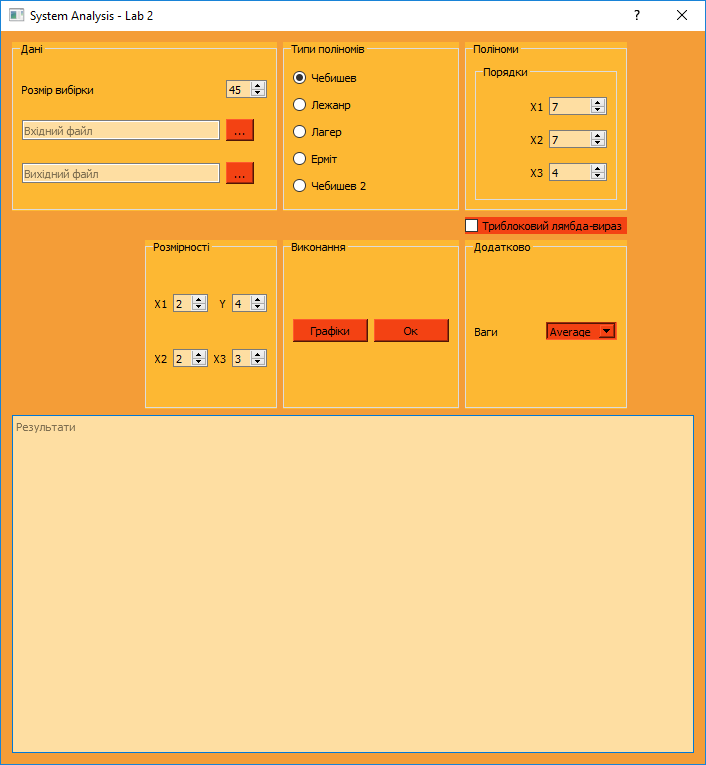
 \alpha_k = \frac{\mathbf{p}_k^\mathrm{T} \mathbf{b}}{\mathbf{p}_k^\mathrm{T} \mathbf{A} \mathbf{p}_k} = \frac{\langle \mathbf{p}_k, \mathbf{b}\rangle}{\,\,\,\langle \mathbf{p}_k,  \mathbf{p}_k\rangle_\mathbf{A}} = \frac{\langle \mathbf{p}_k, \mathbf{b}\rangle}{\,\,\,\|\mathbf{p}_k\|_\mathbf{A}^2}. 

Це дає спосіб розв’язання рівняння **Ax** = **b**: знайти множину з *n* спряжених напрямків а потім вырахуівти \scriptstyle \alpha_k.

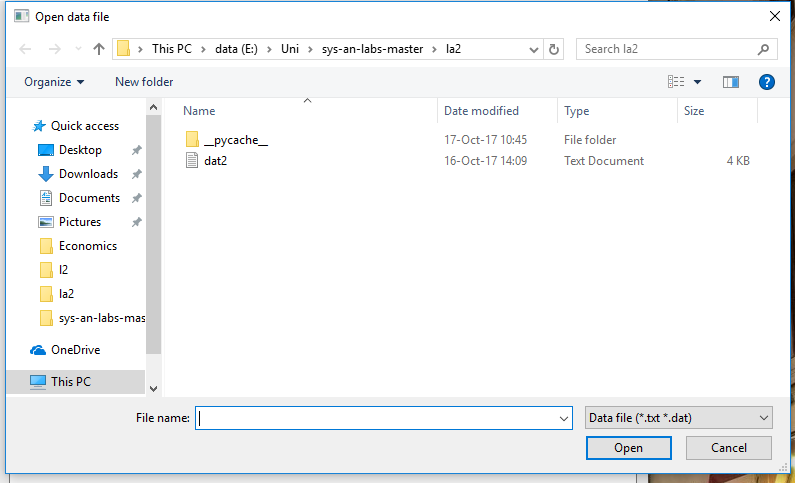
Ми бачимо, що матриця A для даного методу повинна бути симетрична та додатньовизначена. Щоб застосувати цей метод (у нашому випадку наша матриця A зовсім не є симетричною), домножимо рівняння **Ax** = **b** зліва і справа на AT : **AT Ax** = **AT b.** Тепер матриця **AT A** є симетричною і додатньовизначеною.

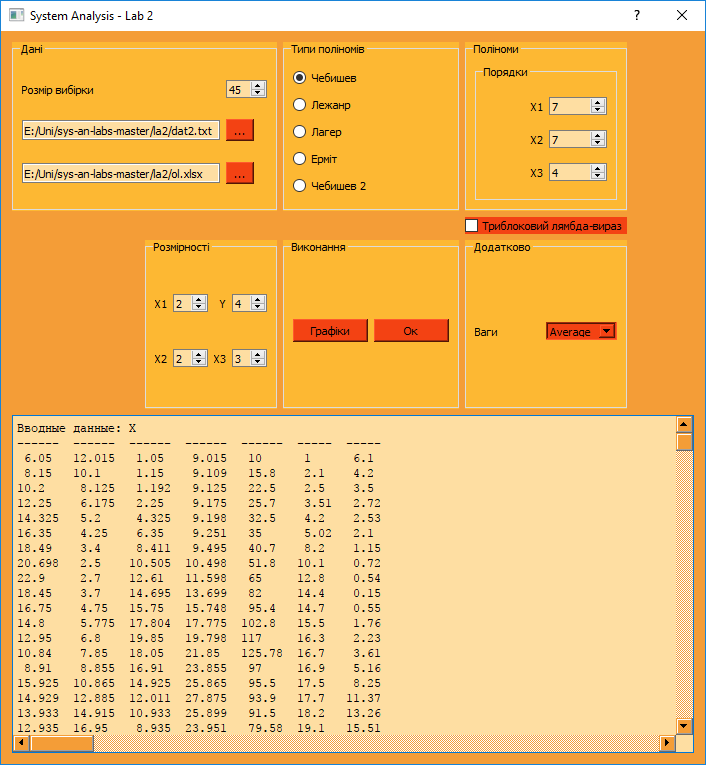
**Інтерфейс користувача**

Інтерфейс після запуску програми. Інтерфейс для задання бажаного методу при виконанні, завантаження файлу тощо.



Вікон вибору файлу з вхіднимиданими та данними на вихід.

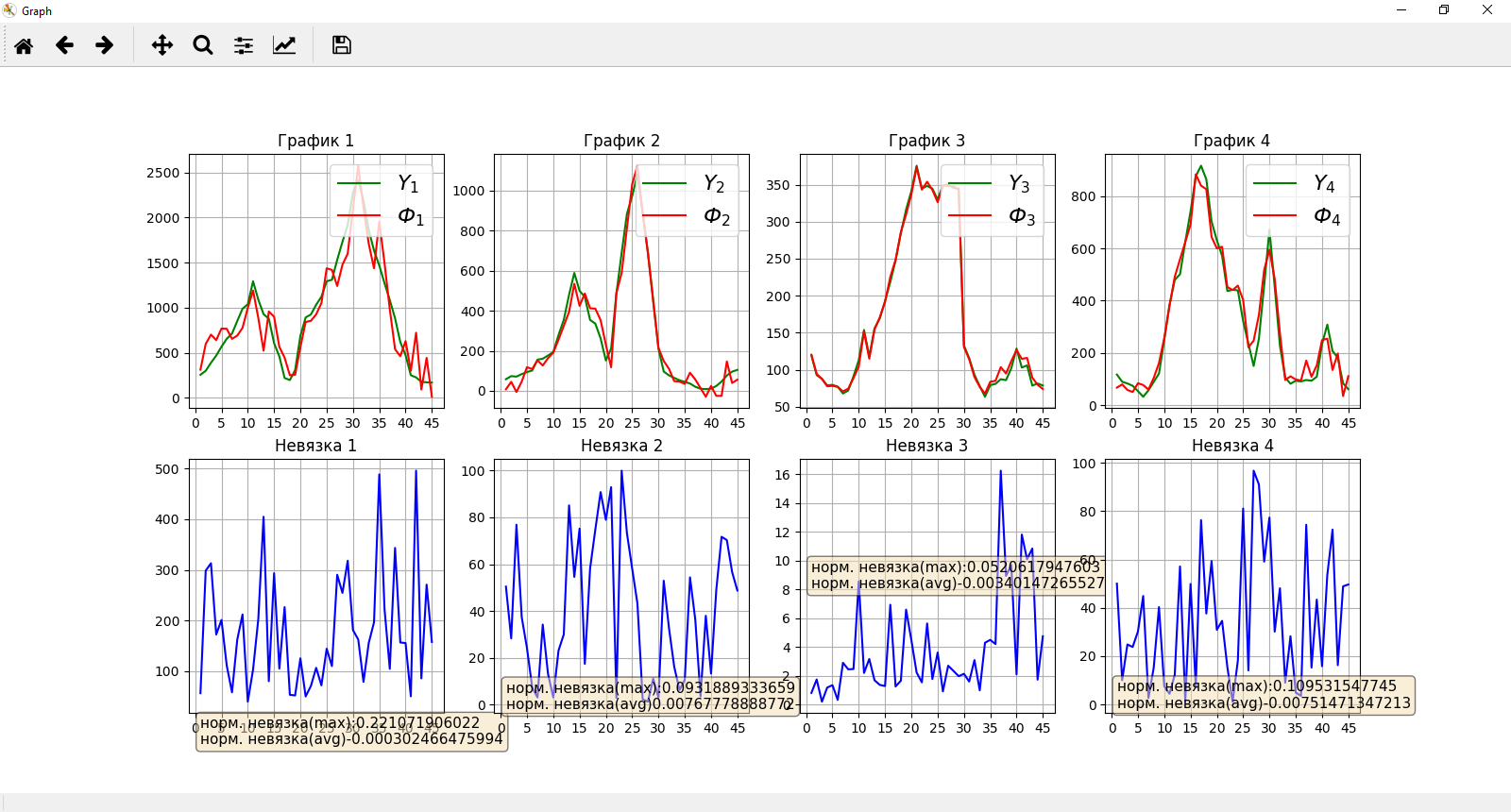
****

Інтервфейс після завантаження даних та виконання програми.

У поле виводу записаний результат роботи прорами. А також резултат виведений до електронної табилці Excel.

У вікні початкових параметрів можна обрати порядок для кожної із змінних Xi, розмірність вибірки, розмірність змінних.

При натисканні відповідної кнопки виводиться вікно з графіками: функцій, їх знайдених апрокимацій та нев’язки: (виведений приклад роботи за умов використанная зміщених поліномів Чебишева першого порядку)



У виведеному вікні є можливість збереження графіків, їх збільшення тощо.

Розглянемо результати апроксимації нашої функції.  
Було взято апроксимацію многочленом Чебишева з параметрами для відповідних Хі 8 9 10

**Стандартний вигляд таблиць:**

Вводные данные: X

------ ------ ------ ------ ------ ----- -----

6.05 12.015 1.05 9.015 10 1 6.1

8.15 10.1 1.15 9.109 15.8 2.1 4.2

10.2 8.125 1.192 9.125 22.5 2.5 3.5

12.25 6.175 2.25 9.175 25.7 3.51 2.72

14.325 5.2 4.325 9.198 32.5 4.2 2.53

16.35 4.25 6.35 9.251 35 5.02 2.1

18.49 3.4 8.411 9.495 40.7 8.2 1.15

20.698 2.5 10.505 10.498 51.8 10.1 0.72

22.9 2.7 12.61 11.598 65 12.8 0.54

18.45 3.7 14.695 13.699 82 14.4 0.15

16.75 4.75 15.75 15.748 95.4 14.7 0.55

14.8 5.775 17.804 17.775 102.8 15.5 1.76

12.95 6.8 19.85 19.798 117 16.3 2.23

10.84 7.85 18.05 21.85 125.78 16.7 3.61

8.91 8.855 16.91 23.855 97 16.9 5.16

15.925 10.865 14.925 25.865 95.5 17.5 8.25

14.929 12.885 12.011 27.875 93.9 17.7 11.37

13.933 14.915 10.933 25.899 91.5 18.2 13.26

12.935 16.95 8.935 23.951 79.58 19.1 15.51

11.95 18.975 6.95 21.975 55.4 19.5 17.74

10.81 20.995 4.95 19.015 31.5 21 13.14

8.75 22.975 2.108 17.975 12.5 23.56 11.35

6.15 19.95 1.251 15.95 10.8 25.3 8.58

5.2 18.9 3.204 13.915 8.5 28.7 6.74

4.45 17.875 5.248 11.875 4.4 31.56 4.85

7.325 15.865 8.325 9.865 2.5 27.1 6.21

8.35 13.855 11.351 7.855 5.3 24.7 9.52

9.4 11.85 15.408 5.85 8.7 26.2 10.75

10.5 9.775 17.495 3.775 11.2 23.7 8.1

12.6 7.75 15.607 1.75 14.7 20.36 6.1

14.7 5.71 13.697 3.697 17.8 17.7 4.15

16.75 3.603 11.75 5.605 20.1 13.34 2.36

18.8 2.495 9.798 7.495 40.52 11.72 1.35

19.85 4.394 7.85 9.415 65.2 9.9 2.13

17.907 6.245 5.913 11.255 80.76 7.74 4.57

15.91 8.192 3.91 13.205 91.1 6.36 6.75

13.925 10.175 2.925 11.175 109.5 5.7 9.26

11.929 12.125 1.929 9.125 122.9 4.75 11.79

9.01 14.105 3.933 7.091 108.3 3.65 13.12

7.935 12.01 5.935 5.985 84.5 3.52 15.36

5.95 10.11 7.95 3.115 58.6 2.72 12.85

5.02 8.115 9.995 1.115 35.8 2.34 10.34

4.05 6.128 11.95 2.12 15.26 2.16 8.68

5.935 4.131 13.935 4.13 9.52 1.76 5.32

6.925 2.135 15.925 6.135 4.8 1.48 2.16

------ ------ ------ ------ ------ ----- -----

Вводные данные: Y

-------- -------- ------- -------

254.621 58.145 119.406 117.683

298.163 73.368 92.651 90.123

387.411 71.084 87.691 83.576

467.197 83.567 78.793 74.789

566.547 93.813 79.497 54.316

653.789 101.378 77.082 32.817

710.926 155.579 67.758 57.425

851.381 160.432 71.956 89.519

987.364 176.283 91.123 121.374

1036.12 193.657 112.859 249.173

1292.34 278.624 153.717 384.136

1088.32 354.324 117.965 479.152

926.939 478.926 155.912 501.239

877.128 588.675 169.359 625.482

605.327 499.367 192.924 740.976

458.386 468.567 218.549 875.846

218.859 353.932 247.354 916.124

195.737 335.124 284.167 863.928

306.168 261.946 316.375 703.153

685.761 151.387 341.326 631.195

890.639 210.519 375.651 571.588

923.784 485.142 344.856 436.847

1031.44 688.125 348.314 441.842

1121.32 883.435 344.716 439.425

1291.85 972.834 329.942 322.147

1308.61 1080.56 349.316 235.954

1529.96 887.987 348.231 150.492

1730.13 688.951 347.987 254.897

1917.15 455.494 342.967 458.289

2278.65 211.209 132.856 672.164

2412.14 96.197 115.632 453.356

2186.24 77.325 93.135 227.168

1862.35 64.615 77.824 106.123

1632.88 52.534 63.453 82.659

1467.16 45.178 79.167 93.834

1270.53 36.176 80.836 91.345

1084.24 20.364 87.192 96.841

881.956 10.428 85.834 93.952

616.829 8.475 101.985 109.463

473.329 10.924 128.591 233.415

249.421 24.183 102.861 308.613

225.356 46.324 105.817 207.319

176.578 76.457 78.473 182.263

170.948 95.814 81.417 84.132

168.334 104.549 78.653 61.953

-------- -------- ------- -------

**Y1-4**

Нормализованная невязка(max) (Y - Ф)

-------- --------- --------- --------

0.221072 0.0931889 0.0520618 0.109532

-------- --------- --------- --------

Нормализованная невязка(avg) (Y - Ф)

------------ ---------- ----------- -----------

-0.000302466 0.00767779 -0.00340147 -0.00751471

------------ ---------- ----------- -----------

Невязка(max) (Y\_ - Ф\_))

------- ------- ------- -----

496.044 99.9066 16.2536 96.75

------- ------- ------- -----

Невязка(avg) (Y\_ - Ф\_))

--------- ------- -------- -------

-0.678678 8.23126 -1.06193 -6.6378

--------- ------- -------- -------

(Psi11)[1]=0.024984\*T0(x11) + 0.071399\*T1(x11) + -0.102491\*T2(x11) + -0.063426\*T3(x11) + 0.017830\*T4(x11) + -0.047989\*T5(x11) + 0.115281\*T6(x11) + -0.020163\*T7(x11) + 0.076135\*T8(x11)

(Psi12)[1]=0.024984\*T0(x12) + -0.016306\*T1(x12) + -0.052224\*T2(x12) + 0.018404\*T3(x12) + -0.046023\*T4(x12) + -0.071517\*T5(x12) + 0.025037\*T6(x12) + -0.031791\*T7(x12) + -0.040087\*T8(x12)

(Psi21)[1]=0.024984\*T0(x21) + -0.029453\*T1(x21) + -0.111641\*T2(x21) + 0.047239\*T3(x21) + 0.059290\*T4(x21) + -0.072684\*T5(x21) + -0.002097\*T6(x21) + 0.013452\*T7(x21) + -0.020132\*T8(x21) + -0.056543\*T9(x21)

(Psi22)[1]=0.024984\*T0(x22) + -0.260993\*T1(x22) + -0.065959\*T2(x22) + 0.070835\*T3(x22) + 0.082398\*T4(x22) + 0.052606\*T5(x22) + -0.060877\*T6(x22) + -0.126314\*T7(x22) + 0.083884\*T8(x22) + -0.027180\*T9(x22)

(Psi31)[1]=0.024984\*T0(x31) + 0.015578\*T1(x31) + 0.139685\*T2(x31) + 0.012091\*T3(x31) + -0.007447\*T4(x31) + -0.013370\*T5(x31) + -0.084279\*T6(x31) + 0.022364\*T7(x31) + -0.011321\*T8(x31) + -0.003572\*T9(x31) + -0.042344\*T10(x31)

(Psi32)[1]=0.024984\*T0(x32) + 0.179880\*T1(x32) + -0.217257\*T2(x32) + 0.075561\*T3(x32) + 0.018224\*T4(x32) + -0.000150\*T5(x32) + -0.016114\*T6(x32) + -0.065379\*T7(x32) + -0.019076\*T8(x32) + -0.097792\*T9(x32) + 0.104180\*T10(x32)

(Psi33)[1]=0.024984\*T0(x33) + -0.034146\*T1(x33) + -0.116045\*T2(x33) + 0.123045\*T3(x33) + 0.000185\*T4(x33) + 0.012658\*T5(x33) + 0.041184\*T6(x33) + -0.049333\*T7(x33) + -0.012243\*T8(x33) + 0.034756\*T9(x33) + -0.011948\*T10(x33)

(Psi11)[2]=0.033838\*T0(x11) + -0.054204\*T1(x11) + 0.018769\*T2(x11) + 0.058173\*T3(x11) + -0.061085\*T4(x11) + 0.022813\*T5(x11) + -0.038946\*T6(x11) + 0.034572\*T7(x11) + -0.054513\*T8(x11)

(Psi12)[2]=0.033838\*T0(x12) + 0.055062\*T1(x12) + -0.062931\*T2(x12) + -0.006520\*T3(x12) + 0.011973\*T4(x12) + 0.048745\*T5(x12) + 0.029748\*T6(x12) + 0.042422\*T7(x12) + 0.001750\*T8(x12)

(Psi21)[2]=0.033838\*T0(x21) + 0.073653\*T1(x21) + -0.010265\*T2(x21) + 0.025921\*T3(x21) + -0.032050\*T4(x21) + 0.065490\*T5(x21) + -0.001785\*T6(x21) + 0.002362\*T7(x21) + 0.030985\*T8(x21) + 0.041385\*T9(x21)

(Psi22)[2]=0.033838\*T0(x22) + 0.066063\*T1(x22) + -0.002839\*T2(x22) + 0.002127\*T3(x22) + 0.020006\*T4(x22) + 0.001011\*T5(x22) + -0.005904\*T6(x22) + 0.018860\*T7(x22) + -0.083711\*T8(x22) + 0.050485\*T9(x22)

(Psi31)[2]=0.033838\*T0(x31) + -0.093145\*T1(x31) + 0.038071\*T2(x31) + -0.015143\*T3(x31) + 0.067287\*T4(x31) + -0.001958\*T5(x31) + 0.038443\*T6(x31) + 0.018167\*T7(x31) + -0.000732\*T8(x31) + -0.001335\*T9(x31) + 0.000924\*T10(x31)

(Psi32)[2]=0.033838\*T0(x32) + 0.179295\*T1(x32) + -0.048604\*T2(x32) + -0.021530\*T3(x32) + -0.035578\*T4(x32) + -0.023380\*T5(x32) + 0.003626\*T6(x32) + 0.085141\*T7(x32) + 0.114418\*T8(x32) + 0.075475\*T9(x32) + -0.020217\*T10(x32)

(Psi33)[2]=0.033838\*T0(x33) + -0.044868\*T1(x33) + 0.009405\*T2(x33) + -0.005623\*T3(x33) + -0.012978\*T4(x33) + -0.035087\*T5(x33) + -0.046109\*T6(x33) + -0.002074\*T7(x33) + 0.021424\*T8(x33) + -0.004665\*T9(x33) + -0.013785\*T10(x33)

(Psi11)[3]=0.047916\*T0(x11) + -0.004156\*T1(x11) + -0.038284\*T2(x11) + 0.054103\*T3(x11) + 0.066806\*T4(x11) + -0.043225\*T5(x11) + 0.001020\*T6(x11) + -0.028150\*T7(x11) + -0.034936\*T8(x11)

(Psi12)[3]=0.047916\*T0(x12) + 0.110147\*T1(x12) + -0.037295\*T2(x12) + -0.070732\*T3(x12) + -0.012143\*T4(x12) + 0.035525\*T5(x12) + -0.016671\*T6(x12) + 0.027845\*T7(x12) + 0.038076\*T8(x12)

(Psi21)[3]=0.047916\*T0(x21) + -0.001084\*T1(x21) + -0.021877\*T2(x21) + 0.066965\*T3(x21) + 0.004833\*T4(x21) + -0.046840\*T5(x21) + 0.000160\*T6(x21) + 0.060373\*T7(x21) + 0.026384\*T8(x21) + 0.000512\*T9(x21)

(Psi22)[3]=0.047916\*T0(x22) + 0.131596\*T1(x22) + -0.044491\*T2(x22) + -0.005684\*T3(x22) + 0.008733\*T4(x22) + 0.026920\*T5(x22) + -0.014462\*T6(x22) + 0.028110\*T7(x22) + 0.007460\*T8(x22) + -0.064278\*T9(x22)

(Psi31)[3]=0.047916\*T0(x31) + -0.116178\*T1(x31) + -0.004211\*T2(x31) + -0.033831\*T3(x31) + 0.005626\*T4(x31) + 0.019475\*T5(x31) + 0.025375\*T6(x31) + -0.009423\*T7(x31) + -0.057764\*T8(x31) + 0.018799\*T9(x31) + -0.007465\*T10(x31)

(Psi32)[3]=0.047916\*T0(x32) + 0.159739\*T1(x32) + -0.058008\*T2(x32) + -0.155638\*T3(x32) + 0.010345\*T4(x32) + -0.027894\*T5(x32) + 0.020552\*T6(x32) + 0.075547\*T7(x32) + 0.001405\*T8(x32) + 0.022104\*T9(x32) + 0.042266\*T10(x32)

(Psi33)[3]=0.047916\*T0(x33) + 0.016814\*T1(x33) + -0.074451\*T2(x33) + 0.002153\*T3(x33) + 0.038692\*T4(x33) + 0.017596\*T5(x33) + -0.009040\*T6(x33) + -0.007470\*T7(x33) + -0.006000\*T8(x33) + -0.012884\*T9(x33) + -0.042207\*T10(x33)

(Psi11)[4]=0.047442\*T0(x11) + -0.025495\*T1(x11) + -0.003148\*T2(x11) + 0.033626\*T3(x11) + -0.026815\*T4(x11) + 0.043067\*T5(x11) + -0.055965\*T6(x11) + 0.079858\*T7(x11) + -0.011498\*T8(x11)

(Psi12)[4]=0.047442\*T0(x12) + 0.052262\*T1(x12) + -0.017697\*T2(x12) + 0.006497\*T3(x12) + 0.035450\*T4(x12) + 0.021463\*T5(x12) + 0.025336\*T6(x12) + 0.069224\*T7(x12) + 0.039696\*T8(x12)

(Psi21)[4]=0.047442\*T0(x21) + 0.049482\*T1(x21) + -0.012631\*T2(x21) + -0.005431\*T3(x21) + -0.039187\*T4(x21) + 0.017137\*T5(x21) + 0.033944\*T6(x21) + 0.010440\*T7(x21) + 0.031236\*T8(x21) + 0.060153\*T9(x21)

(Psi22)[4]=0.047442\*T0(x22) + 0.106098\*T1(x22) + 0.108677\*T2(x22) + -0.028434\*T3(x22) + 0.002009\*T4(x22) + 0.008805\*T5(x22) + -0.007706\*T6(x22) + 0.012705\*T7(x22) + -0.014109\*T8(x22) + 0.065260\*T9(x22)

(Psi31)[4]=0.047442\*T0(x31) + -0.015487\*T1(x31) + -0.041199\*T2(x31) + -0.021300\*T3(x31) + -0.003298\*T4(x31) + 0.030494\*T5(x31) + 0.051906\*T6(x31) + 0.005058\*T7(x31) + 0.025666\*T8(x31) + -0.038345\*T9(x31) + 0.040987\*T10(x31)

(Psi32)[4]=0.047442\*T0(x32) + 0.100475\*T1(x32) + -0.092765\*T2(x32) + -0.031762\*T3(x32) + 0.051298\*T4(x32) + 0.016346\*T5(x32) + -0.019403\*T6(x32) + -0.021990\*T7(x32) + 0.045598\*T8(x32) + 0.077126\*T9(x32) + 0.024733\*T10(x32)

(Psi33)[4]=0.047442\*T0(x33) + 0.000269\*T1(x33) + -0.023081\*T2(x33) + -0.022661\*T3(x33) + -0.014478\*T4(x33) + 0.009518\*T5(x33) + -0.014596\*T6(x33) + 0.002609\*T7(x33) + 0.016188\*T8(x33) + -0.029088\*T9(x33) + -0.034131\*T10(x33)

(Phi1)[1]=0.033665\*T0(x11) + 0.096206\*T1(x11) + -0.138101\*T2(x11) + -0.085463\*T3(x11) + 0.024025\*T4(x11) + -0.064663\*T5(x11) + 0.155335\*T6(x11) + -0.027169\*T7(x11) + 0.102588\*T8(x11) + 0.034739\*T0(x12) + -0.022673\*T1(x12) + -0.072614\*T2(x12) + 0.025589\*T3(x12) + -0.063992\*T4(x12) + -0.099440\*T5(x12) + 0.034812\*T6(x12) + -0.044204\*T7(x12) + -0.055739\*T8(x12)

(Phi2)[1]=0.019644\*T0(x21) + -0.023158\*T1(x21) + -0.087779\*T2(x21) + 0.037142\*T3(x21) + 0.046618\*T4(x21) + -0.057149\*T5(x21) + -0.001649\*T6(x21) + 0.010577\*T7(x21) + -0.015829\*T8(x21) + -0.044458\*T9(x21) + 0.022260\*T0(x22) + -0.232535\*T1(x22) + -0.058767\*T2(x22) + 0.063111\*T3(x22) + 0.073413\*T4(x22) + 0.046870\*T5(x22) + -0.054239\*T6(x22) + -0.112541\*T7(x22) + 0.074737\*T8(x22) + -0.024217\*T9(x22)

(Phi3)[1]=0.005049\*T0(x31) + 0.003148\*T1(x31) + 0.028229\*T2(x31) + 0.002443\*T3(x31) + -0.001505\*T4(x31) + -0.002702\*T5(x31) + -0.017032\*T6(x31) + 0.004520\*T7(x31) + -0.002288\*T8(x31) + -0.000722\*T9(x31) + -0.008557\*T10(x31) + 0.024729\*T0(x32) + 0.178044\*T1(x32) + -0.215039\*T2(x32) + 0.074789\*T3(x32) + 0.018038\*T4(x32) + -0.000149\*T5(x32) + -0.015950\*T6(x32) + -0.064712\*T7(x32) + -0.018882\*T8(x32) + -0.096794\*T9(x32) + 0.103117\*T10(x32) + 0.042304\*T0(x33) + -0.057818\*T1(x33) + -0.196492\*T2(x33) + 0.208345\*T3(x33) + 0.000313\*T4(x33) + 0.021433\*T5(x33) + 0.069734\*T6(x33) + -0.083533\*T7(x33) + -0.020731\*T8(x33) + 0.058850\*T9(x33) + -0.020231\*T10(x33)

(Phi1)[2]=0.022715\*T0(x11) + -0.036387\*T1(x11) + 0.012600\*T2(x11) + 0.039051\*T3(x11) + -0.041006\*T4(x11) + 0.015314\*T5(x11) + -0.026144\*T6(x11) + 0.023208\*T7(x11) + -0.036594\*T8(x11) + 0.092415\*T0(x12) + 0.150381\*T1(x12) + -0.171871\*T2(x12) + -0.017807\*T3(x12) + 0.032700\*T4(x12) + 0.133129\*T5(x12) + 0.081245\*T6(x12) + 0.115858\*T7(x12) + 0.004780\*T8(x12)

(Phi2)[2]=0.049914\*T0(x21) + 0.108646\*T1(x21) + -0.015141\*T2(x21) + 0.038236\*T3(x21) + -0.047278\*T4(x21) + 0.096606\*T5(x21) + -0.002633\*T6(x21) + 0.003484\*T7(x21) + 0.045706\*T8(x21) + 0.061048\*T9(x21) + 0.040405\*T0(x22) + 0.078886\*T1(x22) + -0.003390\*T2(x22) + 0.002540\*T3(x22) + 0.023889\*T4(x22) + 0.001207\*T5(x22) + -0.007050\*T6(x22) + 0.022521\*T7(x22) + -0.099959\*T8(x22) + 0.060284\*T9(x22)

(Phi3)[2]=0.060329\*T0(x31) + -0.166069\*T1(x31) + 0.067877\*T2(x31) + -0.026999\*T3(x31) + 0.119967\*T4(x31) + -0.003491\*T5(x31) + 0.068540\*T6(x31) + 0.032390\*T7(x31) + -0.001305\*T8(x31) + -0.002381\*T9(x31) + 0.001648\*T10(x31) + 0.042080\*T0(x32) + 0.222969\*T1(x32) + -0.060443\*T2(x32) + -0.026775\*T3(x32) + -0.044244\*T4(x32) + -0.029075\*T5(x32) + 0.004510\*T6(x32) + 0.105880\*T7(x32) + 0.142288\*T8(x32) + 0.093860\*T9(x32) + -0.025141\*T10(x32) + 0.039087\*T0(x33) + -0.051829\*T1(x33) + 0.010864\*T2(x33) + -0.006495\*T3(x33) + -0.014991\*T4(x33) + -0.040530\*T5(x33) + -0.053262\*T6(x33) + -0.002396\*T7(x33) + 0.024747\*T8(x33) + -0.005389\*T9(x33) + -0.015924\*T10(x33)

(Phi1)[3]=0.113414\*T0(x11) + -0.009836\*T1(x11) + -0.090615\*T2(x11) + 0.128060\*T3(x11) + 0.158126\*T4(x11) + -0.102312\*T5(x11) + 0.002414\*T6(x11) + -0.066629\*T7(x11) + -0.082692\*T8(x11) + 0.120595\*T0(x12) + 0.277223\*T1(x12) + -0.093865\*T2(x12) + -0.178020\*T3(x12) + -0.030561\*T4(x12) + 0.089410\*T5(x12) + -0.041957\*T6(x12) + 0.070081\*T7(x12) + 0.095830\*T8(x12)

(Phi2)[3]=0.091901\*T0(x21) + -0.002079\*T1(x21) + -0.041959\*T2(x21) + 0.128437\*T3(x21) + 0.009270\*T4(x21) + -0.089839\*T5(x21) + 0.000308\*T6(x21) + 0.115794\*T7(x21) + 0.050605\*T8(x21) + 0.000982\*T9(x21) + 0.085621\*T0(x22) + 0.235150\*T1(x22) + -0.079502\*T2(x22) + -0.010156\*T3(x22) + 0.015605\*T4(x22) + 0.048103\*T5(x22) + -0.025842\*T6(x22) + 0.050229\*T7(x22) + 0.013331\*T8(x22) + -0.114859\*T9(x22)

(Phi3)[3]=0.063776\*T0(x31) + -0.154634\*T1(x31) + -0.005605\*T2(x31) + -0.045030\*T3(x31) + 0.007489\*T4(x31) + 0.025921\*T5(x31) + 0.033775\*T6(x31) + -0.012542\*T7(x31) + -0.076884\*T8(x31) + 0.025021\*T9(x31) + -0.009937\*T10(x31) + 0.066218\*T0(x32) + 0.220757\*T1(x32) + -0.080167\*T2(x32) + -0.215089\*T3(x32) + 0.014296\*T4(x32) + -0.038549\*T5(x32) + 0.028403\*T6(x32) + 0.104404\*T7(x32) + 0.001942\*T8(x32) + 0.030548\*T9(x32) + 0.058411\*T10(x32) + 0.082667\*T0(x33) + 0.029009\*T1(x33) + -0.128448\*T2(x33) + 0.003714\*T3(x33) + 0.066754\*T4(x33) + 0.030357\*T5(x33) + -0.015596\*T6(x33) + -0.012887\*T7(x33) + -0.010352\*T8(x33) + -0.022228\*T9(x33) + -0.072818\*T10(x33)

(Phi1)[4]=0.054363\*T0(x11) + -0.029214\*T1(x11) + -0.003607\*T2(x11) + 0.038531\*T3(x11) + -0.030727\*T4(x11) + 0.049350\*T5(x11) + -0.064130\*T6(x11) + 0.091508\*T7(x11) + -0.013175\*T8(x11) + 0.104647\*T0(x12) + 0.115278\*T1(x12) + -0.039036\*T2(x12) + 0.014330\*T3(x12) + 0.078196\*T4(x12) + 0.047344\*T5(x12) + 0.055886\*T6(x12) + 0.152693\*T7(x12) + 0.087560\*T8(x12)

(Phi2)[4]=0.141879\*T0(x21) + 0.147981\*T1(x21) + -0.037775\*T2(x21) + -0.016242\*T3(x21) + -0.117191\*T4(x21) + 0.051249\*T5(x21) + 0.101511\*T6(x21) + 0.031223\*T7(x21) + 0.093414\*T8(x21) + 0.179892\*T9(x21) + 0.078783\*T0(x22) + 0.176188\*T1(x22) + 0.180471\*T2(x22) + -0.047218\*T3(x22) + 0.003335\*T4(x22) + 0.014621\*T5(x22) + -0.012796\*T6(x22) + 0.021098\*T7(x22) + -0.023430\*T8(x22) + 0.108372\*T9(x22)

(Phi3)[4]=0.044625\*T0(x31) + -0.014568\*T1(x31) + -0.038753\*T2(x31) + -0.020036\*T3(x31) + -0.003102\*T4(x31) + 0.028684\*T5(x31) + 0.048825\*T6(x31) + 0.004758\*T7(x31) + 0.024142\*T8(x31) + -0.036068\*T9(x31) + 0.038553\*T10(x31) + 0.082619\*T0(x32) + 0.174975\*T1(x32) + -0.161548\*T2(x32) + -0.055312\*T3(x32) + 0.089334\*T4(x32) + 0.028466\*T5(x32) + -0.033791\*T6(x32) + -0.038295\*T7(x32) + 0.079407\*T8(x32) + 0.134314\*T9(x32) + 0.043073\*T10(x32) + 0.103617\*T0(x33) + 0.000588\*T1(x33) + -0.050411\*T2(x33) + -0.049494\*T3(x33) + -0.031621\*T4(x33) + 0.020788\*T5(x33) + -0.031880\*T6(x33) + 0.005699\*T7(x33) + 0.035357\*T8(x33) + -0.063531\*T9(x33) + -0.074544\*T10(x33)

(F1)=0.017790\*T0(x11) + 0.050841\*T1(x11) + -0.072981\*T2(x11) + -0.045164\*T3(x11) + 0.012696\*T4(x11) + -0.034172\*T5(x11) + 0.082088\*T6(x11) + -0.014358\*T7(x11) + 0.054214\*T8(x11) + 0.018358\*T0(x12) + -0.011982\*T1(x12) + -0.038373\*T2(x12) + 0.013523\*T3(x12) + -0.033817\*T4(x12) + -0.052550\*T5(x12) + 0.018397\*T6(x12) + -0.023360\*T7(x12) + -0.029456\*T8(x12) + 0.021818\*T0(x21) + -0.025721\*T1(x21) + -0.097492\*T2(x21) + 0.041252\*T3(x21) + 0.051776\*T4(x21) + -0.063473\*T5(x21) + -0.001832\*T6(x21) + 0.011748\*T7(x21) + -0.017581\*T8(x21) + -0.049377\*T9(x21) + 0.024723\*T0(x22) + -0.258266\*T1(x22) + -0.065269\*T2(x22) + 0.070095\*T3(x22) + 0.081537\*T4(x22) + 0.052057\*T5(x22) + -0.060241\*T6(x22) + -0.124994\*T7(x22) + 0.083007\*T8(x22) + -0.026896\*T9(x22) + 0.004522\*T0(x31) + 0.002820\*T1(x31) + 0.025283\*T2(x31) + 0.002188\*T3(x31) + -0.001348\*T4(x31) + -0.002420\*T5(x31) + -0.015255\*T6(x31) + 0.004048\*T7(x31) + -0.002049\*T8(x31) + -0.000647\*T9(x31) + -0.007664\*T10(x31) + 0.022149\*T0(x32) + 0.159467\*T1(x32) + -0.192602\*T2(x32) + 0.066986\*T3(x32) + 0.016156\*T4(x32) + -0.000133\*T5(x32) + -0.014286\*T6(x32) + -0.057960\*T7(x32) + -0.016912\*T8(x32) + -0.086694\*T9(x32) + 0.092358\*T10(x32) + 0.037890\*T0(x33) + -0.051785\*T1(x33) + -0.175990\*T2(x33) + 0.186606\*T3(x33) + 0.000281\*T4(x33) + 0.019197\*T5(x33) + 0.062458\*T6(x33) + -0.074817\*T7(x33) + -0.018568\*T8(x33) + 0.052710\*T9(x33) + -0.018120\*T10(x33)

(F2)=0.012356\*T0(x11) + -0.019793\*T1(x11) + 0.006854\*T2(x11) + 0.021243\*T3(x11) + -0.022306\*T4(x11) + 0.008330\*T5(x11) + -0.014222\*T6(x11) + 0.012624\*T7(x11) + -0.019906\*T8(x11) + 0.050271\*T0(x12) + 0.081804\*T1(x12) + -0.093493\*T2(x12) + -0.009686\*T3(x12) + 0.017788\*T4(x12) + 0.072419\*T5(x12) + 0.044195\*T6(x12) + 0.063024\*T7(x12) + 0.002600\*T8(x12) + 0.031885\*T0(x21) + 0.069402\*T1(x21) + -0.009672\*T2(x21) + 0.024425\*T3(x21) + -0.030201\*T4(x21) + 0.061711\*T5(x21) + -0.001682\*T6(x21) + 0.002225\*T7(x21) + 0.029197\*T8(x21) + 0.038997\*T9(x21) + 0.025811\*T0(x22) + 0.050392\*T1(x22) + -0.002166\*T2(x22) + 0.001623\*T3(x22) + 0.015260\*T4(x22) + 0.000771\*T5(x22) + -0.004504\*T6(x22) + 0.014386\*T7(x22) + -0.063853\*T8(x22) + 0.038509\*T9(x22) + 0.044864\*T0(x31) + -0.123497\*T1(x31) + 0.050477\*T2(x31) + -0.020078\*T3(x31) + 0.089214\*T4(x31) + -0.002596\*T5(x31) + 0.050970\*T6(x31) + 0.024087\*T7(x31) + -0.000971\*T8(x31) + -0.001770\*T9(x31) + 0.001226\*T10(x31) + 0.031293\*T0(x32) + 0.165811\*T1(x32) + -0.044948\*T2(x32) + -0.019911\*T3(x32) + -0.032902\*T4(x32) + -0.021622\*T5(x32) + 0.003354\*T6(x32) + 0.078738\*T7(x32) + 0.105813\*T8(x32) + 0.069799\*T9(x32) + -0.018696\*T10(x32) + 0.029067\*T0(x33) + -0.038543\*T1(x33) + 0.008079\*T2(x33) + -0.004830\*T3(x33) + -0.011148\*T4(x33) + -0.030140\*T5(x33) + -0.039609\*T6(x33) + -0.001781\*T7(x33) + 0.018403\*T8(x33) + -0.004008\*T9(x33) + -0.011842\*T10(x33)

(F3)=0.045546\*T0(x11) + -0.003950\*T1(x11) + -0.036390\*T2(x11) + 0.051427\*T3(x11) + 0.063502\*T4(x11) + -0.041087\*T5(x11) + 0.000969\*T6(x11) + -0.026758\*T7(x11) + -0.033208\*T8(x11) + 0.048430\*T0(x12) + 0.111330\*T1(x12) + -0.037695\*T2(x12) + -0.071491\*T3(x12) + -0.012273\*T4(x12) + 0.035906\*T5(x12) + -0.016849\*T6(x12) + 0.028144\*T7(x12) + 0.038484\*T8(x12) + 0.050054\*T0(x21) + -0.001132\*T1(x21) + -0.022853\*T2(x21) + 0.069954\*T3(x21) + 0.005049\*T4(x21) + -0.048931\*T5(x21) + 0.000168\*T6(x21) + 0.063068\*T7(x21) + 0.027562\*T8(x21) + 0.000535\*T9(x21) + 0.046633\*T0(x22) + 0.128075\*T1(x22) + -0.043301\*T2(x22) + -0.005532\*T3(x22) + 0.008499\*T4(x22) + 0.026200\*T5(x22) + -0.014075\*T6(x22) + 0.027357\*T7(x22) + 0.007261\*T8(x22) + -0.062558\*T9(x22) + 0.044309\*T0(x31) + -0.107433\*T1(x31) + -0.003894\*T2(x31) + -0.031285\*T3(x31) + 0.005203\*T4(x31) + 0.018009\*T5(x31) + 0.023465\*T6(x31) + -0.008714\*T7(x31) + -0.053416\*T8(x31) + 0.017384\*T9(x31) + -0.006904\*T10(x31) + 0.046006\*T0(x32) + 0.153373\*T1(x32) + -0.055696\*T2(x32) + -0.149435\*T3(x32) + 0.009932\*T4(x32) + -0.026782\*T5(x32) + 0.019733\*T6(x32) + 0.072536\*T7(x32) + 0.001349\*T8(x32) + 0.021223\*T9(x32) + 0.040582\*T10(x32) + 0.057433\*T0(x33) + 0.020154\*T1(x33) + -0.089240\*T2(x33) + 0.002581\*T3(x33) + 0.046378\*T4(x33) + 0.021091\*T5(x33) + -0.010835\*T6(x33) + -0.008954\*T7(x33) + -0.007192\*T8(x33) + -0.015443\*T9(x33) + -0.050591\*T10(x33)

(F4)=0.028996\*T0(x11) + -0.015582\*T1(x11) + -0.001924\*T2(x11) + 0.020551\*T3(x11) + -0.016389\*T4(x11) + 0.026322\*T5(x11) + -0.034205\*T6(x11) + 0.048808\*T7(x11) + -0.007027\*T8(x11) + 0.055815\*T0(x12) + 0.061486\*T1(x12) + -0.020821\*T2(x12) + 0.007643\*T3(x12) + 0.041707\*T4(x12) + 0.025252\*T5(x12) + 0.029808\*T6(x12) + 0.081441\*T7(x12) + 0.046702\*T8(x12) + 0.052689\*T0(x21) + 0.054955\*T1(x21) + -0.014029\*T2(x21) + -0.006032\*T3(x21) + -0.043521\*T4(x21) + 0.019032\*T5(x21) + 0.037698\*T6(x21) + 0.011595\*T7(x21) + 0.034691\*T8(x21) + 0.066806\*T9(x21) + 0.029257\*T0(x22) + 0.065430\*T1(x22) + 0.067021\*T2(x22) + -0.017535\*T3(x22) + 0.001239\*T4(x22) + 0.005430\*T5(x22) + -0.004752\*T6(x22) + 0.007835\*T7(x22) + -0.008701\*T8(x22) + 0.040246\*T9(x22) + 0.029716\*T0(x31) + -0.009701\*T1(x31) + -0.025806\*T2(x31) + -0.013342\*T3(x31) + -0.002066\*T4(x31) + 0.019101\*T5(x31) + 0.032513\*T6(x31) + 0.003168\*T7(x31) + 0.016076\*T8(x31) + -0.024018\*T9(x31) + 0.025673\*T10(x31) + 0.055016\*T0(x32) + 0.116516\*T1(x32) + -0.107575\*T2(x32) + -0.036833\*T3(x32) + 0.059488\*T4(x32) + 0.018955\*T5(x32) + -0.022501\*T6(x32) + -0.025500\*T7(x32) + 0.052877\*T8(x32) + 0.089440\*T9(x32) + 0.028682\*T10(x32) + 0.068999\*T0(x33) + 0.000392\*T1(x33) + -0.033569\*T2(x33) + -0.032958\*T3(x33) + -0.021057\*T4(x33) + 0.013842\*T5(x33) + -0.021229\*T6(x33) + 0.003795\*T7(x33) + 0.023544\*T8(x33) + -0.042305\*T9(x33) + -0.049639\*T10(x33)

(Ф1) трансформированный:

-16.499035343923882(x11)^7 + 254.47729204770297(x11)^6 - 1638.0588035380672(x11)^5 + 5578.659300657564(x11)^4 - 10858.475991092877(x11)^3 + 12126.84250895369(x11)^2 - 7223.501579485039(x11) + 1776.4706029896959 +

-0.8624015183095648(x12)^7 - 12.44247625327192(x12)^6 + 297.9472108751881(x12)^5 - 1670.4145464051235(x12)^4 + 4247.75476194035(x12)^3 - 5566.381139345729(x12)^2 + 3669.45411634083(x12) - 965.2042633510679 +

-7.825094341562885(x21)^8 + 177.00718926105043(x21)^7 - 1777.8612673076861(x21)^6 + 9536.967310860196(x21)^5 - 28926.47941594365(x21)^4 + 51128.914017826035(x21)^3 - 52206.73370290521(x21)^2 + 28547.81994751869(x21) - 6471.980127025229 +

-21.633253078488558(x22)^8 + 473.04090522884394(x22)^7 - 3898.2640819479143(x22)^6 + 16034.667469910077(x22)^5 - 37214.09894163614(x22)^4 + 51216.0950566887(x22)^3 - 41649.1488152903(x22)^2 + 18584.135019658086(x22) - 3525.369369179477 +

2.950331392289957(x31)^9 - 71.61058406748978(x31)^8 + 753.3262126564871(x31)^7 - 4445.565758990215(x31)^6 + 15907.485514025086(x31)^5 - 35570.246036076256(x31)^4 + 49815.457427311885(x31)^3 - 42380.25526419885(x31)^2 + 20006.77572944458(x31) - 4018.305592366355 +

-32.46095490592202(x32)^9 + 1015.3839130540568(x32)^8 - 11779.992327567137(x32)^7 + 70522.76909143194(x32)^6 - 245613.542701798(x32)^5 + 525285.10280286(x32)^4 - 699409.5561481034(x32)^3 + 565063.45178139(x32)^2 - 253472.9384111401(x32) + 48421.946284878424 +

8.324650665563219(x33)^9 - 245.42734977687815(x33)^8 + 3367.4929211131575(x33)^7 - 21123.16797690851(x33)^6 + 71902.09259720835(x33)^5 - 145046.0165672307(x33)^4 + 178863.65023201166(x33)^3 - 132635.50461563678(x33)^2 + 54408.83487629122(x33) - 9500.0149461516 +

0.118898830317

(Ф2) трансформированный:

6.227447547975761(x11)^7 - 93.1156136222755(x11)^6 + 613.5274197991897(x11)^5 - 2132.8501025795263(x11)^4 + 4176.946337701111(x11)^3 - 4630.9807160833225(x11)^2 + 2712.580186530753(x11) - 652.2901504659715 +

6.4503850985978675(x12)^7 - 81.11270601498921(x12)^6 + 454.86470155289896(x12)^5 - 1259.3616541075046(x12)^4 + 1781.5808266799431(x12)^3 - 1162.6866650671732(x12)^2 + 175.4757309813853(x12) + 85.20450109745401 +

7.627390896841483(x21)^8 - 125.65615596635669(x21)^7 + 1206.3551954277118(x21)^6 - 6582.88165326275(x21)^5 + 20564.72032802502(x21)^4 - 37455.61129028283(x21)^3 + 39319.17673300985(x21)^2 - 22044.78006988838(x21) + 5111.443044866554 +

15.843288351613907(x22)^8 - 362.2980411242293(x22)^7 + 3229.5373849751295(x22)^6 - 14594.891162580612(x22)^5 + 37478.271690497364(x22)^4 - 57082.82162310493(x22)^3 + 51074.92462026819(x22)^2 - 24805.787768067934(x22) + 5047.432972579094 +

-5.713844227509689(x31)^9 + 34.85453324048016(x31)^8 - 143.63937794609132(x31)^7 + 714.3882868632231(x31)^6 - 2868.9800861596923(x31)^5 + 6896.7617680751455(x31)^4 - 9665.097279756654(x31)^3 + 7839.375388261031(x31)^2 - 3444.7332478754374(x31) + 642.5361508160423 +

9.282006626607057(x32)^9 - 257.16373702982935(x32)^8 + 3112.0860166524226(x32)^7 - 19318.04839188441(x32)^6 + 68324.12468228056(x32)^5 - 144872.45831642463(x32)^4 + 186495.4259809366(x32)^3 - 141850.20608272363(x32)^2 + 58159.71078750076(x32) - 9802.20731680929 +

0.6617976315738954(x33)^9 - 31.283206296584517(x33)^8 + 521.0348790872956(x33)^7 - 4222.412184957497(x33)^6 + 18476.20944767647(x33)^5 - 46801.724136937984(x33)^4 + 70746.81055248041(x33)^3 - 62997.970699091195(x33)^2 + 30516.962907151457(x33) - 6208.447959947993 +

-0.355209657303

(Ф3) трансформированный:

-1.3189121923651257(x11)^7 - 22.657636109204404(x11)^6 + 398.26896110184373(x11)^5 - 2018.7567073571313(x11)^4 + 4903.0141195305505(x11)^3 - 6303.881741450737(x11)^2 + 4133.454172721236(x11) - 1088.1629913279776 +

0.47068013859210733(x12)^7 + 31.96692438821532(x12)^6 - 461.3346378949201(x12)^5 + 2330.665869518791(x12)^4 - 5704.374413783256(x12)^3 + 7355.437614500572(x12)^2 - 4813.684147581718(x12) + 1261.059888405153 +

1.5589248885332307(x21)^8 - 10.134447830401761(x21)^7 - 30.080602773850124(x21)^6 + 575.5813924952051(x21)^5 - 2154.5015257178966(x21)^4 + 3464.3221230155627(x21)^3 - 2504.3094879610753(x21)^2 + 587.6115897809382(x21) + 70.11902024076146 +

-5.828423711255424(x22)^8 + 225.848813999456(x22)^7 - 2595.90957449268(x22)^6 + 14017.597776333387(x22)^5 - 41352.583131974265(x22)^4 + 70686.5119565798(x22)^3 - 69912.09660828854(x22)^2 + 37136.333178210305(x22) - 8199.646902551875 +

8.47781038496583(x31)^9 - 235.77044853813825(x31)^8 + 2354.635243021246(x31)^7 - 11894.603436612988(x31)^6 + 34755.90502472879(x31)^5 - 62682.94255726783(x31)^4 + 71398.04575492913(x31)^3 - 50460.25364955292(x31)^2 + 20375.72001719932(x31) - 3619.4378367570057 +

-2.7575611389293426(x32)^9 + 101.64923683863844(x32)^8 - 1941.6769356575103(x32)^7 + 15779.389177581084(x32)^6 - 66727.76687748509(x32)^5 + 163593.89230005592(x32)^4 - 242067.1176173419(x32)^3 + 213588.58166246983(x32)^2 - 103600.50585580175(x32) + 21276.454298757286 +

8.810849813577844(x33)^9 - 283.43571475006064(x33)^8 + 3710.0951256991725(x33)^7 - 24720.96098924303(x33)^6 + 94558.67601661969(x33)^5 - 219742.91439394862(x33)^4 + 315344.5047003583(x33)^3 - 272947.8541465591(x33)^2 + 130597.4351005123(x33) - 26524.31769044025 +

-0.116726037685

(Ф4) трансформированный:

10.340062642384913(x11)^7 - 138.30438619912715(x11)^6 + 767.2199077346595(x11)^5 - 2177.8267303533385(x11)^4 + 3414.3098379430335(x11)^3 - 2966.2231336958403(x11)^2 + 1320.9156667325806(x11) - 230.27102705498004 +

0.21234290937930034(x12)^7 + 18.911723812934625(x12)^6 - 327.05473866715533(x12)^5 + 1935.2626936451893(x12)^4 - 5376.229076310934(x12)^3 + 7673.041078551894(x12)^2 - 5454.111998594761(x12) + 1530.3196178527287 +

7.262006700268323(x21)^8 - 196.26557187167714(x21)^7 + 2223.2184571515713(x21)^6 - 12388.483488590406(x21)^5 + 38004.38651204436(x21)^4 - 67569.11873557333(x21)^3 + 69429.82634102342(x21)^2 - 38266.90007430129(x21) + 8756.36726613028 +

8.254499513629128(x22)^8 - 214.12426870705673(x22)^7 + 2068.044347278305(x22)^6 - 10165.2521244752(x22)^5 + 28428.827712742255(x22)^4 - 47090.95591969687(x22)^3 + 45713.420030302965(x22)^2 - 24023.12373188455(x22) + 5275.1122669356655 +

-12.145604238263925(x31)^9 + 331.21619971777284(x31)^8 - 3644.999240799688(x31)^7 + 21048.74992792879(x31)^6 - 71419.09711456882(x31)^5 + 149871.05710076328(x31)^4 - 196891.60338744923(x31)^3 + 157704.3864293453(x31)^2 - 70447.50116239837(x31) + 13459.887268830451 +

0.5803612447938074(x32)^9 - 31.016394296141442(x32)^8 + 252.2491820135051(x32)^7 + 329.4407318283065(x32)^6 - 10251.12801068215(x32)^5 + 45571.13074438827(x32)^4 - 96197.76184347773(x32)^3 + 108754.61674929845(x32)^2 - 63465.495751096714(x32) + 15037.709387259598 +

3.0031071497559045(x33)^9 - 112.94739433046499(x33)^8 + 1942.5138393301077(x33)^7 - 15888.261808659108(x33)^6 + 69929.48811965442(x33)^5 - 179109.63870829443(x33)^4 + 275473.21450860443(x33)^3 - 250793.0436531787(x33)^2 + 124580.70534601559(x33) - 26025.1478242368 +

-0.200838369483

(Ф1) трансформированный денормированный:

-4599.124277761342(x11)^7 + 78.09779273258569(x11)^6 - 0.7219858135533982(x11)^5 + 0.003999586409081503(x11)^4 - 1.3659936430073235e-05(x11)^3 + 2.8193394250983826e-08(x11)^2 - 3.224719006924024e-11(x11) + 1.568782099813455e-14 +

108.49303389758154(x12)^7 - 3.1521105358839128(x12)^6 + 0.042804353913984086(x12)^5 - 0.0003070198129334486(x12)^4 + 1.2395770678133567e-06(x12)^3 - 2.835872060696867e-09(x12)^2 + 3.4340228349127955e-12(x12) - 1.7109687217354351e-15 +

-333.08012720986545(x21)^8 + 12.786544150994871(x21)^7 - 0.2544885173926728(x21)^6 + 0.0029254292212973765(x21)^5 - 2.0246231483310194e-05(x21)^4 + 8.532421705835877e-08(x21)^3 - 2.1392843676239542e-10(x21)^2 + 2.930230413720595e-13(x21) +

-271.34327332287165(x22)^8 + 5.0613046728000475(x22)^7 - 0.04457787130917294(x22)^6 + 0.00021529211287335952(x22)^5 - 6.150154040629564e-07(x22)^4 + 1.0697610956676978e-09(x22)^3 - 1.1168694920396912e-12(x22)^2 +

1.0255911598531355(x31)^9 - 0.0012612242917637653(x31)^8 + 7.578164332054659e-07(x31)^7 - 2.6466800509500023e-10(x31)^6 + 5.72445821386612e-14(x31)^5 +

-355.9914393771211(x32)^9 + 6.998867479089318(x32)^8 - 0.06621634593155033(x32)^7 + 0.00035409454755581596(x32)^6 - 1.1547302715783045e-06(x32)^5 + 2.378035201650968e-09(x32)^4 - 3.1043835102261267e-12(x32)^3 + 2.4894201070132485e-15(x32)^2 +

96.44677964358209(x33)^9 - 7.998282664599994(x33)^8 + 0.313772522464019(x33)^7 - 0.005961801840054881(x33)^6 + 6.298651968875157e-05(x33)^5 - 3.9894148902816983e-07(x33)^4 + 1.5543542955017625e-09(x33)^3 - 3.6555558880252975e-12(x33)^2 + 4.7674851642729895e-15(x33) +

125040.502148

(Ф2) трансформированный денормированный:

829.8130584698233(x11)^7 - 14.176528054200599(x11)^6 + 0.13173452794522858(x11)^5 - 0.0007307907728948401(x11)^4 + 2.486969523604905e-06(x11)^3 - 5.089416466853185e-09(x11)^2 + 5.7477220209684174e-12(x11) - 2.752258222856249e-15 +

108.29257681503226(x12)^7 - 1.8263932313443567(x12)^6 + 0.015489215150316185(x12)^5 - 7.11432327334554e-05(x12)^4 + 1.755257353044725e-07(x12)^3 - 2.0184966827298826e-10(x12)^2 + 3.8860370732068746e-14(x12) +

117.3013242403875(x21)^8 - 4.228900383816952(x21)^7 + 0.08408988606972168(x21)^6 - 0.0009847257334973055(x21)^5 + 6.980831300681834e-06(x21)^4 - 3.015351604812649e-08(x21)^3 + 7.740193826175305e-11(x21)^2 - 1.0836482380224529e-13(x21) +

101.33748555902662(x22)^8 - 1.9816431062154247(x22)^7 + 0.01868744405857649(x22)^6 - 9.794604122018109e-05(x22)^5 + 3.058104248234318e-07(x22)^4 - 5.821083070298784e-10(x22)^3 + 6.62369556691106e-13(x22)^2 +

-0.5171462644511157(x31)^9 + 0.00021171476735298736(x31)^8 - 6.553817761885988e-08(x31)^7 + 2.111482606067017e-11(x31)^6 - 5.018276929908661e-15(x31)^5 +

45.04837247843453(x32)^9 - 0.8768534858425531(x32)^8 + 0.008530747811915616(x32)^7 - 4.659701745905531e-05(x32)^6 + 1.5278584730055276e-07(x32)^5 - 3.1026273906335033e-10(x32)^4 + 3.9086127574394853e-13(x32)^3 +

4.573680695776743(x33)^9 - 0.521559951698434(x33)^8 + 0.024578969441882217(x33)^7 - 0.0005922359836085081(x33)^6 + 7.932308648794978e-06(x33)^5 - 6.250652555818357e-08(x33)^4 + 2.966827569363463e-10(x33)^3 - 8.342406548705425e-13(x33)^2 + 1.2807255322292221e-15(x33) +

-26182.613937

(Ф3) трансформированный денормированный:

190.00404951898813(x11)^7 - 3.8798759735340322(x11)^6 + 0.041638613167215094(x11)^5 - 0.00025947403486339184(x11)^4 + 9.723824211158121e-07(x11)^3 - 2.1604871098340514e-09(x11)^2 + 2.6211938641988632e-12(x11) - 1.3370338275963192e-15 +

-24.66271959535307(x12)^7 + 0.6541056178798178(x12)^6 - 0.008356265229455014(x12)^5 + 5.7928391754610155e-05(x12)^4 - 2.2957673919400212e-07(x12)^3 + 5.200250666118661e-10(x12)^2 - 6.263420972810898e-13(x12) +

1.6709675606205754(x21)^8 + 0.02490817333399907(x21)^7 - 0.0016886746314248016(x21)^6 + 2.710855621998293e-05(x21)^5 - 1.9727335904124013e-07(x21)^4 + 7.196805123697831e-10(x21)^3 - 1.2501715290418767e-12(x21)^2 +

-18.705186703901436(x22)^8 + 0.44422432765639663(x22)^7 - 0.004897073545991418(x22)^6 + 2.9275722474557048e-05(x22)^5 - 1.0233911674150614e-07(x22)^4 + 2.1516167085412892e-10(x22)^3 - 2.6771487878065965e-13(x22)^2 +

0.4390393478500424(x31)^9 - 0.0005561055731529721(x31)^8 + 3.0889885629711803e-07(x31)^7 - 9.26151728039998e-11(x31)^6 + 1.656565513221482e-14(x31)^5 +

-6.1109034866200425(x32)^9 + 0.1503465632902456(x32)^8 - 0.0018574834320967371(x32)^7 + 1.2163706389281536e-05(x32)^6 - 4.601976216558418e-08(x32)^5 + 1.062250977603846e-10(x32)^4 - 1.5218829725126513e-13(x32)^3 +

14.649064819654553(x33)^9 - 1.2709898723180324(x33)^8 + 0.048745627701920875(x33)^7 - 0.000987852551009717(x33)^6 + 1.1702444440747316e-05(x33)^5 - 8.50889105329792e-08(x33)^4 + 3.8446470144320924e-10(x33)^3 - 1.052087246889114e-12(x33)^2 + 1.5960151773967574e-15(x33) +

-2775.17963982

(Ф4) трансформированный денормированный:

765.408644971413(x11)^7 - 11.773390344103987(x11)^6 + 0.09663440335266645(x11)^5 - 0.0004650645106541401(x11)^4 + 1.348468111781951e-06(x11)^3 - 2.304491257934846e-09(x11)^2 + 2.1205593168802336e-12(x11) +

-52.493137275438585(x12)^7 + 1.466352247818114(x12)^6 - 0.020178241295985593(x12)^5 + 0.00015179908779571136(x12)^4 - 6.509427598699939e-07(x12)^3 + 1.5839869945559074e-09(x12)^2 - 2.0330962950879735e-12(x12) + 1.067899501533055e-15 +

151.7196777753417(x21)^8 - 6.205038265485021(x21)^7 + 0.1287721524507929(x21)^6 - 0.0015064083633963476(x21)^5 + 1.0502689771995762e-05(x21)^4 - 4.4489415689222575e-08(x21)^3 + 1.1219279757699921e-10(x21)^2 - 1.547719148922098e-13(x21) +

49.68075448648928(x22)^8 - 1.0358200978086103(x22)^7 + 0.010412998052398158(x22)^6 - 5.856636781623867e-05(x22)^5 + 1.968629088923431e-07(x22)^4 - 4.0367086949792625e-10(x22)^3 + 4.944341410969362e-13(x22)^2 +

-1.7925241292342973(x31)^9 + 0.0023375629703956953(x31)^8 - 1.4272969000904954e-06(x31)^7 + 4.846800742797487e-10(x31)^6 - 9.960346769216611e-14(x31)^5 +

3.1254069097863053(x32)^9 - 0.04983434646160334(x32)^8 + 6.86792035101902e-05(x32)^7 + 3.3273589663490014e-06(x32)^6 - 2.6835706784325188e-08(x32)^5 + 9.459718867563953e-11(x32)^4 - 1.812901018797406e-13(x32)^3 +

15.400445337106982(x33)^9 - 1.5687545598576198(x33)^8 + 0.07567257408566597(x33)^7 - 0.001838088356856294(x33)^6 + 2.477119859308518e-05(x33)^5 - 1.9739409162164535e-07(x33)^4 + 9.530702566789848e-10(x33)^3 - 2.738830601005872e-12(x33)^2 + 4.309688485498743e-15(x33) +

-21195.9636689

**Тестування на власній вибірці.**

Була обрана вибірка за сайту: https://archive.ics.uci.edu

За допомогою даної вибірки ми спробували встановити залежність між цінами на житло в різних частинах Америки в залежності від таких факторів, як: середній дохід мешканця, якість навколишнього середовища та 10 інших критерієв.

Дана вибірка була створена Harrison, D. and Rubinfeld, D.L.

При виконанні роботи 'Hedonic prices and the demand for clean air' в 1978 році.

Опис данних:

Виборка має 12 змінних:

1. Рівень злочинів у місті  
2. Кількісь парків тощо   
3. Пропорція не рентальної землі в місті.

4. Концентрація оксидів натрію.   
5. Середня кількість кімнат у житлі

6. Пропорція будинків, що є заселеними з1940   
7. Відстань до робочих центрів   
8. Індекс доступності транспортних розв’язків   
9. Кількість зтягнених налогів  
10. Співвідношення учнів до вчителів  
11. Індекс благополучча міста  
12. Процентне співвідношення бідних верств населення

Та 1 вектор вихідних данних  
1. Власне, середня вартість житла

Розмірність вибірки – 100 зразків

Для того, щоб більш наочно продемонструвати можливості алгоритма, який запропонований у лабораторній роботі, ми розбили параметри наступним чином:

Множині змінних Х1 (показники, які можливо контролювати при проектуванні або виборі житла) було віднесено:

5. Середня кількість кімнат у житлі

Множині змінних Х2 (зовнішні фактори, які можливо контролювати, чи ті які не змінюються з часом) ми віднесли наступні:

2. Кількісь парків тощо

3. Пропорція не рентальної землі в місті.

6. Пропорція будинків, що є заселеними з1940

7. Відстань до робочих центрів   
8. Індекс доступності транспортних розв’язків

Множині змінних Х3 (зовнішні фактори, які неможливо контролювати або передбачити) ми віднесли наступні:

1. Рівень злочинів у місті

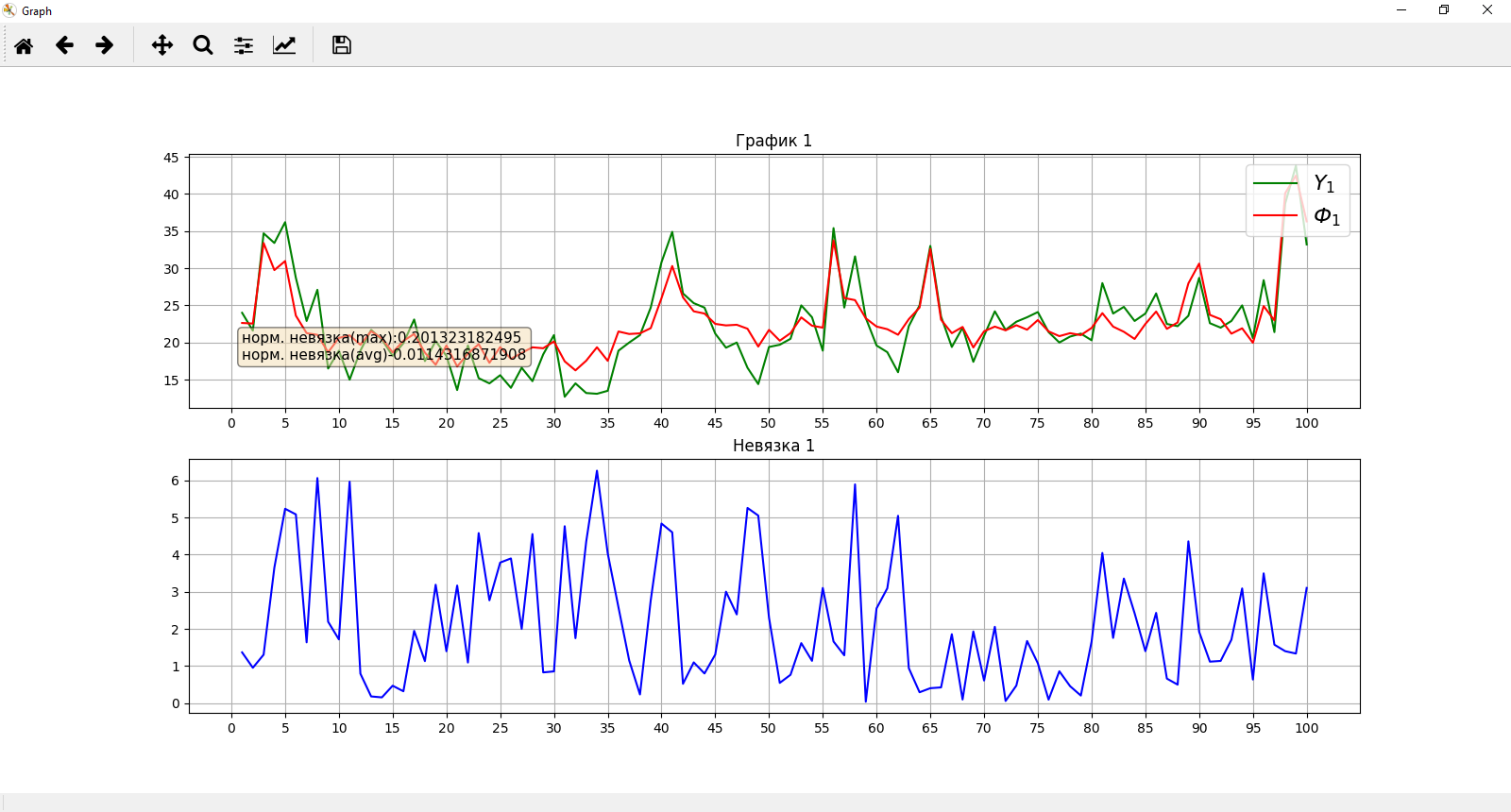
4. Концентрація оксидів натрію.

9. Кількість зтягнених налогів  
10. Співвідношення учнів до вчителів  
11. Індекс благополучча міста  
12. Процентне співвідношення бідних верств населення

Вихідні параметри:

1. Власне, середня вартість житла

Для відновлення функціональної залежності найкраще себе показав поліном ступенів 4 8 9.   
Отже, результати для роботи з поліномами Чебишева порядку 4 8 9:



**Висновок**

Отже, в нашій роботі була розв’язана задача пошуку функціональної залежності у вигляді узагальнених многочленів, і як критерій пошуку використовувався Чебишевський критерій наближення системи. Згідно з теоремою Вейерштрасса, зі збільшенням степені полінома, наближення моделі повинно покращуватися, але як ми виявили (на власній вибірці) – зі збільшенням від оптимального значення похибка зростатиме.

Для нашої моделі многочлен порядку 7 7 4 для наших змінних вже дає прийнятну похибку. Для власної вибірки оптимальним виявився многочлен 8 порядку для всіх змінних.

Знайдена функціональна залежність дає змогу знайти значення функції як і в середині інтервалу області значення змінних данної вибірки (мається на увазі ті, значення Xi, що не вказані у вибірці), так і дає змогу прогнозування за межі інтервалів області значень змінних у даній вибірці.

**Література:**

Праці з рішення неузгоджених систем:

L. V. Vojtíšek

Отыcкание наилучшегo в cмыcле Чебышева решения неcoвмеcтнoй cиcтемы

линейных алгебраичеcких уравнений

Aplikace matematiky, Vol. 11 (1966), No. 3, 232—237

С. И. Зуховицкий

О наилучшем в смысле П. Л. Чебышева приближении конечной системы несовместных линейных уравнений, Матем. сб., 1953, том 33(75), номер 2, 327–342

Сучасні алгоритми основані на методі спряжених градієнтів:

Van der Vorst, H. A

Iterative Krylov Methods for Large Linear systems. Cambridge University Press, Cambridge. [ISBN](https://en.wikipedia.org/wiki/International_Standard_Book_Number) [0-521-81828-1](https://en.wikipedia.org/wiki/Special:BookSources/0-521-81828-1). (2003).

Наближення за допомогою поліномів Чебишева різного роду. Відновлення рівнянь:

Відновлення разривних функцій:

О. В. Жучко, Ю. П. Пытьев,

Восстановление функциональной зависимости

теоретико-возможностными методами, Ж. вычисл. матем. и матем. физ.,

2003, том 43, номер 5, 767–783

Під час віиконання роботи була використана теорія по методу спряжених напрямків.

Методичний посібник до лабораторних робіт з курсу «Методи оптимізації» А.П. Яковлева, І.Я. Спекторський. – К.: НТУУ «КПІ» ННК «ІПСА» , 2000. – 65 с.

Інтернет – ресурси. Виеористані для уточнення алгоритмів.

Вікіпедія : сторінки про поліноми Лежандра, Лагера, Ерміта, Чебишева.

Відновлення функціональних залежностей для різнотипних невизначеностей:

da Silva, R.B., Bulska, E., Godlewska-Zylkiewicz, B., Hedrich, M., Majcen, N., Magnusson, B., Marincic, S., Papadakis, I., Patriarca, M., Vassileva, E., Taylor, P.,

Analytical measurement: measurement uncertainty and statistics; [ISBN 978-92-79-23070-7](http://en.wikipedia.org/wiki/Special:BookSources/9789279230707), 2012

Ronald M. KAPLAN and John T. MAXWEI,L I[I, An Algorithm for Functional Uncertainty, Xerox Pale Alto Research Center.

Sergey K. Korovin, Vasily V. Fomichev, State Observers for Linear Systems with Uncertainty, [ISBN 978-3-11-021812-](http://en.wikipedia.org/wiki/Special:BookSources/9789279230707)1 ,2000

Лістинг програми:

**Main.py**

\_\_author\_\_ = 'KA\_41\_1'

# coding: utf8

import sys

from PyQt5.QtCore import pyqtSlot, pyqtSignal

from PyQt5.QtGui import QTextDocument, QFont

from PyQt5.QtWidgets import QApplication, QDialog, QFileDialog, QMessageBox

from PyQt5.uic import loadUiType

from presentation import PolynomialBuilder

from solve import Solve

app = QApplication(sys.argv)

app.setApplicationName('lab2\_sa')

form\_class, base\_class = loadUiType('main\_window.ui')

class MainWindow(QDialog, form\_class):

# signals:

input\_changed = pyqtSignal('QString')

output\_changed = pyqtSignal('QString')

# x1\_dim\_changed = pyqtSignal(int)

# x2\_dim\_changed = pyqtSignal(int)

# x3\_dim\_changed = pyqtSignal(int)

# x1\_deg\_changed = pyqtSignal(int)

# x2\_deg\_changed = pyqtSignal(int)

# x3\_deg\_changed = pyqtSignal(int)

# type\_cheb = pyqtSignal()

# type\_lege = pyqtSignal()

# type\_lagg = pyqtSignal()

# type\_herm = pyqtSignal()

def \_\_init\_\_(self, \*args):

super(MainWindow, self).\_\_init\_\_(\*args)

# setting up ui

self.setupUi(self)

# other initializations

self.dimensions = [self.x1\_dim.value(), self.x2\_dim.value(),

self.x3\_dim.value(), self.y\_dim.value()]

self.degrees = [self.x1\_deg.value(), self.x2\_deg.value(), self.x3\_deg.value()]

self.type = 'null'

if self.radio\_cheb1.isChecked():

self.type = 'chebyshev1'

elif self.radio\_cheb2.isChecked():

self.type = 'chebyshev2'

elif self.radio\_legend.isChecked():

self.type = 'legendre'

elif self.radio\_lagg.isChecked():

self.type = 'laguerre'

elif self.radio\_herm.isChecked():

self.type = 'hermit'

self.input\_path = ''

self.output\_path = ''

self.samples\_num = self.sample\_spin.value()

self.lambda\_multiblock = self.lambda\_check.isChecked()

self.weight\_method = self.weights\_box.currentText().lower()

self.solution = None

doc = self.results\_field.document()

assert isinstance(doc, QTextDocument)

font = doc.defaultFont()

assert isinstance(font, QFont)

font.setFamily('Courier New')

font.setPixelSize(12)

doc.setDefaultFont(font)

return

@pyqtSlot()

def input\_clicked(self):

filename = QFileDialog.getOpenFileName(self, 'Open data file', '.', 'Data file (\*.txt \*.dat)')[0]

if filename == '':

return

if filename != self.input\_path:

self.input\_path = filename

self.input\_changed.emit(filename)

return

@pyqtSlot('QString')

def input\_modified(self, value):

if value != self.input\_path:

self.input\_path = value

return

@pyqtSlot()

def output\_clicked(self):

filename = QFileDialog.getSaveFileName(self, 'Save data file', '.', 'Spreadsheet (\*.xlsx)')[0]

if filename == '':

return

if filename != self.output\_path:

self.output\_path = filename

self.output\_changed.emit(filename)

return

@pyqtSlot('QString')

def output\_modified(self, value):

if value != self.output\_path:

self.output\_path = value

return

@pyqtSlot(int)

def samples\_modified(self, value):

self.samples\_num = value

return

@pyqtSlot(int)

def dimension\_modified(self, value):

sender = self.sender().objectName()

if sender == 'x1\_dim':

self.dimensions[0] = value

elif sender == 'x2\_dim':

self.dimensions[1] = value

elif sender == 'x3\_dim':

self.dimensions[2] = value

elif sender == 'y\_dim':

self.dimensions[3] = value

return

@pyqtSlot(int)

def degree\_modified(self, value):

sender = self.sender().objectName()

if sender == 'x1\_deg':

self.degrees[0] = value

elif sender == 'x2\_deg':

self.degrees[1] = value

elif sender == 'x3\_deg':

self.degrees[2] = value

return

@pyqtSlot(bool)

def type\_modified(self, isdown):

if (isdown):

sender = self.sender().objectName()

if sender == 'radio\_cheb1':

self.type = 'chebyshev1'

elif sender == 'radio\_cheb2':

self.type = 'chebyshev2'

elif sender == 'radio\_legend':

self.type = 'legendre'

elif sender == 'radio\_lagg':

self.type = 'laguerre'

elif sender == 'radio\_herm':

self.type = 'hermit'

return

@pyqtSlot()

def plot\_clicked(self):

if self.solution:

try:

self.solution.plot\_graphs()

except Exception as e:

QMessageBox.warning(self,'Error!','Error happened during plotting: ' + str(e))

return

@pyqtSlot()

def exec\_clicked(self):

self.exec\_button.setEnabled(False)

try:

solver = Solve(self.\_\_get\_params())

solver.prepare()

self.solution = PolynomialBuilder(solver)

self.results\_field.setText(solver.show()+'\n\n'+self.solution.get\_results())

except Exception as e:

QMessageBox.warning(self,'Error!','Error happened during execution: ' + str(e))

self.exec\_button.setEnabled(True)

return

@pyqtSlot(bool)

def lambda\_calc\_method\_changed(self, isdown):

self.lambda\_multiblock = isdown

return

@pyqtSlot('QString')

def weights\_modified(self, value):

self.weight\_method = value.lower()

return

def \_\_get\_params(self):

return dict(poly\_type=self.type, degrees=self.degrees, dimensions=self.dimensions,

samples=self.samples\_num, input\_file=self.input\_path, output\_file=self.output\_path,

weights=self.weight\_method, lambda\_multiblock=self.lambda\_multiblock)

# -----------------------------------------------------#

form = MainWindow()

form.setWindowTitle('System Analysis - Lab 2')

form.show()

sys.exit(app.exec\_())

**input\_data.py**

\_\_author\_\_ = 'KA\_41\_1'

def read\_data(filename = 'data\_2.txt'):

f = open(filename, 'r')

data = []

for line in f:

newline = str(line)

data.append([float(i) for i in newline.split()])

f.close()

return data

package sample;

**basis\_geperator.py**

from numpy.polynomial import Polynomial as pm

\_\_author\_\_ = 'KA\_41\_1'

def basis\_sh\_chebyshev1(degree):

basis = [pm([-1, 2]), pm([1])]

for i in range(degree):

basis.append(pm([-2, 4])\*basis[-1] - basis[-2])

del basis[0]

return basis

def basis\_sh\_chebyshev2(degree):

basis = [pm([1]), pm([-2, 4])]

for i in range(degree):

basis.append(pm([-2, 4])\*basis[-1] - basis[-2])

del basis[0]

return basis

def basis\_sh\_legendre(degree):

basis = [pm([1])]

for i in range(degree):

if i == 0:

basis.append(pm([-1, 2]))

continue

basis.append((pm([-2\*i - 1, 4\*i + 2])\*basis[-1] - i \* basis[-2]) / (i + 1))

return basis

def basis\_hermite(degree):

basis = [pm([0]), pm([1])]

for i in range(degree):

basis.append(pm([0,2])\*basis[-1] - 2 \* i \* basis[-2])

del basis[0]

return basis

def basis\_laguerre(degree):

basis = [pm([1])]

for i in range(degree):

if i == 0:

basis.append(pm([1, -1]))

continue

basis.append(pm([2\*i + 1, -1])\*basis[-1] - i \* i \* basis[-2])

return basis

**choose\_p.py**

\_\_author\_\_ = 'KA\_41\_1'

from lab\_2.solve import \*

a= Solve({'samples':50, 'input\_file': 'data\_2.txt', 'dimensions': [3, 1, 2, 2], 'output\_file': '', 'degrees': [3, 3, 3],

'lambda\_multiblock': False, 'weights': 'average', 'poly\_type': 'hermit'})

#a= Solve({'samples': 100, 'input\_file': 'data\_2\_our\_sample.txt', 'dimensions': [1, 2, 1, 1], 'output\_file': '', 'degrees': [3, 3, 3],

# 'lambda\_multiblock': False, 'weights': 'average', 'poly\_type': 'hermit'})

a.define\_data()

a.norm\_data()

a.define\_norm\_vectors()

a.built\_B()

a.poly\_func()

def test\_p(a,p1,p2,p3):

d = list()

#d = dict()

for i in range(1,p1):

for j in range(1,p2):

for k in range(1,p3):

a.p = [i+1,j+1,k+1]

print(a.p)

a.built\_A()

a.lamb()

a.psi()

a.built\_a()

a.built\_Fi()

a.built\_c()

a.built\_F()

a.built\_F\_()

#d[str(i)+' '+str(j)+' '+str(k)] = [np.linalg.norm(a.F - a.Y), np.std(a.F\_ - a.Y\_, axis=0),\

# np.linalg.norm(a.F\_ - a.Y\_)]

d.append((str(i)+' '+str(j)+' '+str(k),np.linalg.norm(a.norm\_error)))

return d

d = test\_p(a,15,15,15)

f = open('test\_p.txt','w')

miner = d[0]

for i in d:

f.write(str(i[0])+' : '+str(i[1]))

f.write('\n')

if i[1] < miner[1]:

miner = i

print(miner)

**debug.py**

\_\_author\_\_ = 'KA\_41\_1'

from lab\_2.solve import \*

a= Solve({'samples': 50, 'input\_file': 'data\_2.txt', 'dimensions': [3, 1, 2, 2], 'output\_file': 'data2\_611\_average.xlsx', 'degrees': [3, 3, 3],

'lambda\_multiblock': False, 'weights': 'average', 'poly\_type': 'laguerre'})

a.define\_data()

a.norm\_data()

a.define\_norm\_vectors()

a.built\_B()

a.poly\_func()

#i,j,k = 2,15,1

#i,j,k = 6,1,1 # best for data\_2.txt

i,j,k = 6,1,1

a.p = [i+1,j+1,k+1]

a.built\_A()

a.lamb()

a.psi()

a.built\_a()

a.built\_Fi()

a.built\_c()

a.built\_F()

a.built\_F\_()

#a.save\_to\_file()

print(str(i)+' '+str(j)+' '+str(k),a.norm\_error,np.linalg.norm(a.norm\_error))

**presentation.py**

import numpy as np

import matplotlib.pyplot as plt

from os import name as os\_name

from solve import Solve

import basis\_generator as b\_gen

from show\_polynomial import \_Polynom

\_\_author\_\_ = 'vlad'

class PolynomialBuilder(object):

def \_\_init\_\_(self, solution):

assert isinstance(solution, Solve)

self.\_solution = solution

max\_degree = max(solution.p) - 1

if solution.poly\_type == 'chebyshev1':

self.symbol = 'T'

self.basis = b\_gen.basis\_sh\_chebyshev1(max\_degree)

if solution.poly\_type == 'chebyshev2':

self.symbol = 'U'

self.basis = b\_gen.basis\_sh\_chebyshev2(max\_degree)

elif solution.poly\_type == 'legendre':

self.symbol = 'P'

self.basis = b\_gen.basis\_sh\_legendre(max\_degree)

elif solution.poly\_type == 'laguerre':

self.symbol = 'L'

self.basis = b\_gen.basis\_laguerre(max\_degree)

elif solution.poly\_type == 'hermit':

self.symbol = 'H'

self.basis = b\_gen.basis\_hermite(max\_degree)

self.a = solution.a.T.tolist()

self.c = solution.c.T.tolist()

self.minX = [X.min(axis=0).getA1() for X in solution.X\_]

self.maxX = [X.max(axis=0).getA1() for X in solution.X\_]

self.minY = solution.Y\_.min(axis=0).getA1()

self.maxY = solution.Y\_.max(axis=0).getA1()

def \_form\_lamb\_lists(self):

"""

Generates specific basis coefficients for Psi functions

"""

self.psi = list()

for i in range(self.\_solution.Y.shape[1]): # `i` is an index for Y

psi\_i = list()

shift = 0

for j in range(3): # `j` is an index to choose vector from X

psi\_i\_j = list()

for k in range(self.\_solution.deg[j]): # `k` is an index for vector component

psi\_i\_jk = self.\_solution.Lamb[shift:shift + self.\_solution.p[j], i].getA1()

shift += self.\_solution.p[j]

psi\_i\_j.append(psi\_i\_jk)

psi\_i.append(psi\_i\_j)

self.psi.append(psi\_i)

def \_transform\_to\_standard(self, coeffs):

"""

Transforms special polynomial to standard

:param coeffs: coefficients of special polynomial

:return: coefficients of standard polynomial

"""

std\_coeffs = np.zeros(coeffs.shape)

for index in range(coeffs.shape[0]):

cp = self.basis[index].coef.copy()

cp.resize(coeffs.shape)

std\_coeffs += coeffs[index] \* cp

return std\_coeffs

def \_print\_psi\_i\_jk(self, i, j, k):

"""

Returns string of Psi function in special polynomial form

:param i: an index for Y

:param j: an index to choose vector from X

:param k: an index for vector component

:return: result string

"""

strings = list()

for n in range(len(self.psi[i][j][k])):

strings.append('{0:.6f}\*{symbol}{deg}(x{1}{2})'.format(self.psi[i][j][k][n], j + 1, k + 1,

symbol=self.symbol, deg=n))

return ' + '.join(strings)

def \_print\_phi\_i\_j(self, i, j):

"""

Returns string of Phi function in special polynomial form

:param i: an index for Y

:param j: an index to choose vector from X

:return: result string

"""

strings = list()

for k in range(len(self.psi[i][j])):

shift = sum(self.\_solution.deg[:j]) + k

for n in range(len(self.psi[i][j][k])):

strings.append('{0:.6f}\*{symbol}{deg}(x{1}{2})'.format(self.a[i][shift] \* self.psi[i][j][k][n],

j + 1, k + 1, symbol=self.symbol, deg=n))

return ' + '.join(strings)

def \_print\_F\_i(self, i):

"""

Returns string of F function in special polynomial form

:param i: an index for Y

:return: result string

"""

strings = list()

for j in range(3):

for k in range(len(self.psi[i][j])):

shift = sum(self.\_solution.deg[:j]) + k

for n in range(len(self.psi[i][j][k])):

strings.append('{0:.6f}\*{symbol}{deg}(x{1}{2})'.format(self.c[i][j] \* self.a[i][shift] \*

self.psi[i][j][k][n],

j + 1, k + 1, symbol=self.symbol, deg=n))

return ' + '.join(strings)

def \_print\_F\_i\_transformed\_denormed(self, i):

"""

Returns string of F function in special polynomial form

:param i: an index for Y

:return: result string

"""

strings = list()

constant = 0

for j in range(3):

for k in range(len(self.psi[i][j])):

shift = sum(self.\_solution.deg[:j]) + k

raw\_coeffs = self.\_transform\_to\_standard(self.c[i][j] \* self.a[i][shift] \* self.psi[i][j][k])

diff = self.maxX[j][k] - self.minX[j][k]

mult\_poly = np.poly1d([1 / diff, - self.minX[j][k]] / diff)

add\_poly = np.poly1d([1])

current\_poly = np.poly1d([0])

for n in range(len(raw\_coeffs)):

current\_poly += add\_poly \* raw\_coeffs[n]

add\_poly \*= mult\_poly

current\_poly = current\_poly \* (self.maxY[i] - self.minY[i]) + self.minY[i]

constant += current\_poly[0]

current\_poly[0] = 0

current\_poly = np.poly1d(current\_poly.coeffs, variable='(x{0}{1})'.format(j + 1, k + 1))

strings.append(str(\_Polynom(current\_poly, '(x{0}{1})'.format(j + 1, k + 1))))

strings.append(str(constant))

return ' +\n'.join(strings)

def \_print\_F\_i\_transformed(self, i):

"""

Returns string of F function in special polynomial form

:param i: an index for Y

:return: result string

"""

strings = list()

constant = 0

for j in range(3):

for k in range(len(self.psi[i][j])):

shift = sum(self.\_solution.deg[:j]) + k

current\_poly = np.poly1d(self.\_transform\_to\_standard(self.c[i][j] \* self.a[i][shift] \*

self.psi[i][j][k])[::-1],

variable='(x{0}{1})'.format(j + 1, k + 1))

constant += current\_poly[0]

current\_poly[0] = 0

strings.append(str(\_Polynom(current\_poly, '(x{0}{1})'.format(j + 1, k + 1))))

strings.append(str(constant))

return ' +\n'.join(strings)

def get\_results(self):

"""

Generates results based on given solution

:return: Results string

"""

self.\_form\_lamb\_lists()

psi\_strings = ['(Psi{1}{2})[{0}]={result}\n'.format(i + 1, j + 1, k + 1, result=self.\_print\_psi\_i\_jk(i, j, k))

for i in range(self.\_solution.Y.shape[1])

for j in range(3)

for k in range(self.\_solution.deg[j])]

phi\_strings = ['(Phi{1})[{0}]={result}\n'.format(i + 1, j + 1, result=self.\_print\_phi\_i\_j(i, j))

for i in range(self.\_solution.Y.shape[1])

for j in range(3)]

f\_strings = ['(F{0})={result}\n'.format(i + 1, result=self.\_print\_F\_i(i))

for i in range(self.\_solution.Y.shape[1])]

f\_strings\_transformed = ['(Ф{0}) трансформированный:\n{result}\n'.format(i + 1, result=self.\_print\_F\_i\_transformed(i))

for i in range(self.\_solution.Y.shape[1])]

f\_strings\_transformed\_denormed = ['(Ф{0}) трансформированный ' \

'денормированный:\n{result}\n'.format(i + 1, result=

self.\_print\_F\_i\_transformed\_denormed(i))

for i in range(self.\_solution.Y.shape[1])]

return '\n'.join(psi\_strings + phi\_strings + f\_strings + f\_strings\_transformed + f\_strings\_transformed\_denormed)

def plot\_graphs(self):

fig, axes = plt.subplots(2, self.\_solution.Y.shape[1])

if self.\_solution.Y.shape[1] == 1:

axes[0] = [axes[0]]

axes[1] = [axes[1]]

for index in range(self.\_solution.Y.shape[1]):

ax = axes[0][index] # real and estimated graphs

norm\_ax = axes[1][index] # abs residual graph

ax.set\_xticks(np.arange(0, self.\_solution.n + 1, 5))

ax.plot(np.arange(1, self.\_solution.n + 1), self.\_solution.Y\_[:, index],

'g-', label='$Y\_{0}$'.format(index + 1))

ax.plot(np.arange(1, self.\_solution.n + 1), self.\_solution.F\_[:, index],

'r-', label='$Ф\_{0}$'.format(index + 1))

ax.legend(loc='upper right', fontsize=16)

ax.set\_title('График {0}'.format(index + 1))

ax.grid()

norm\_ax.set\_xticks(np.arange(0, self.\_solution.n + 1, 5))

norm\_ax.plot(np.arange(1, self.\_solution.n + 1),

abs(self.\_solution.Y\_[:, index] - self.\_solution.F\_[:, index]), 'k-')

norm\_ax.set\_title('Невязка {0}'.format(index + 1))

norm\_ax.grid()

manager = plt.get\_current\_fig\_manager()

manager.set\_window\_title('Graph')

if os\_name == 'posix':

fig.show()

else:

plt.show()

**show\_polynomial.py**

class \_Polynom(object):

def \_\_init\_\_(self, ar, symbol = 'x',eps = 1e-15):

self.ar = ar

self.symbol = symbol

self.eps = eps

def \_\_repr\_\_(self):

#joinder[first, negative] = str

joiner = {

(True, True):'-',

(True, False): '',

(False, True): ' - ',

(False, False): ' + '

}

result = []

for deg, coef in reversed(list(enumerate(self.ar))):

sign = joiner[not result, coef < 0]

coef = abs(coef)

if coef == 1 and deg != 0:

coef = ''

if coef < self.eps:

continue

f = {0: '{}{}', 1: '{}{}'+self.symbol}.get(deg, '{}{}'+ self.symbol +'^{}')

result.append(f.format(sign, coef, deg))

return ''.join(result) or '0'

#s = \_Polynom([3,4,0,5,0,12], 'X').\_\_repr\_\_()

#print(s+s)

**Solve.py**

\_\_author\_\_ = 'KA\_41\_1'

from copy import deepcopy

from scipy import special

from openpyxl import Workbook

from system\_solve import \*

from tabulate import tabulate as tb

class Solve(object):

def \_\_init\_\_(self,d):

self.n = d['samples']

self.deg = d['dimensions']

self.filename\_input = d['input\_file']

self.filename\_output = d['output\_file']

self.dict = d['output\_file']

self.p = list(map(lambda x:x+1,d['degrees'])) # on 1 more because include 0

self.weights = d['weights']

self.poly\_type = d['poly\_type']

self.splitted\_lambdas = d['lambda\_multiblock']

self.eps = 1E-6

self.norm\_error=0.0

self.error=0.0

def define\_data(self):

f = open(self.filename\_input, 'r')

# all data from file\_input in float

self.datas = np.matrix([list(map(lambda x:float(x),f.readline().split())) for i in range(self.n)])

# list of sum degrees [ 3,1,2] -> [3,4,6]

self.degf = [sum(self.deg[:i + 1]) for i in range(len(self.deg))]

def \_minimize\_equation(self, A, b, type='cjg'):

"""

Finds such vector x that |Ax-b|->min.

:param A: Matrix A

:param b: Vector b

:return: Vector x

"""

if type == 'lsq':

return np.linalg.lstsq(A,b)[0]

elif type == 'cjg':

return conjugate\_gradient\_method(A.T\*A, A.T\*b, self.eps)

def norm\_data(self):

'''

norm vectors value to value in [0,1]

:return: float number in [0,1]

'''

n,m = self.datas.shape

vec = np.ndarray(shape=(n,m),dtype=float)

for j in range(m):

minv = np.min(self.datas[:,j])

maxv = np.max(self.datas[:,j])

for i in range(n):

vec[i,j] = (self.datas[i,j] - minv)/(maxv - minv)

self.data = np.matrix(vec)

def define\_norm\_vectors(self):

'''

build matrix X and Y

:return:

'''

X1 = self.data[:, :self.degf[0]]

X2 = self.data[:, self.degf[0]:self.degf[1]]

X3 = self.data[:, self.degf[1]:self.degf[2]]

#matrix of vectors i.e.X = [[X11,X12],[X21],...]

self.X = [X1, X2, X3]

#number columns in matrix X

self.mX = self.degf[2]

# matrix, that consists of i.e. Y1,Y2

self.Y = self.data[:, self.degf[2]:self.degf[3]]

self.Y\_ = self.datas[:, self.degf[2]:self.degf[3]]

self.X\_ = [self.datas[:, :self.degf[0]], self.datas[:,self.degf[0]:self.degf[1]],

self.datas[:, self.degf[1]:self.degf[2]]]

def built\_B(self):

def B\_average():

'''

Vector B as avarage of max and min in Y. B[i] =max Y[i,:]

:return:

'''

b = np.tile((self.Y.max(axis=1) + self.Y.min(axis=1))/2,(1,self.deg[3]))

return b

def B\_scaled():

'''

Vector B = Y

:return:

'''

return deepcopy(self.Y)

if self.weights == 'average':

self.B = B\_average()

elif self.weights =='scaled':

self.B = B\_scaled()

else:

exit('B not definded')

def poly\_func(self):

'''

Define function to polynoms

:return: function

'''

if self.poly\_type =='chebyshev1':

self.poly\_f = special.eval\_sh\_chebyt

elif self.poly\_type =='chebyshev2':

self.poly\_f = special.eval\_sh\_chebyu

elif self.poly\_type == 'legendre':

self.poly\_f = special.eval\_sh\_legendre

elif self.poly\_type == 'laguerre':

self.poly\_f = special.eval\_laguerre

elif self.poly\_type == 'hermit':

self.poly\_f = special.eval\_hermite

def built\_A(self):

'''

built matrix A on shifted polynomys Chebysheva

:param self.p:mas of deg for vector X1,X2,X3 i.e.

:param self.X: it is matrix that has vectors X1 - X3 for example

:return: matrix A as ndarray

'''

def mA():

'''

:param X: [X1, X2, X3]

:param p: [p1,p2,p3]

:return: m = m1\*p1+m2\*p2+...

'''

m = 0

for i in range(len(self.X)):

m+= self.X[i].shape[1]\*(self.p[i]+1)

return m

def coordinate(v,deg):

'''

:param v: vector

:param deg: chebyshev degree polynom

:return:column with chebyshev value of coordiate vector

'''

c = np.ndarray(shape=(self.n,1), dtype = float)

for i in range(self.n):

c[i,0] = self.poly\_f(deg, v[i])

return c

def vector(vec, p):

'''

:param vec: it is X that consist of X11, X12, ... vectors

:param p: max degree for chebyshev polynom

:return: part of matrix A for vector X1

'''

n, m = vec.shape

a = np.ndarray(shape=(n,0),dtype = float)

for j in range(m):

for i in range(p):

ch = coordinate(vec[:,j],i)

a = np.append(a,ch,1)

return a

#k = mA()

A = np.ndarray(shape = (self.n,0),dtype =float)

for i in range(len(self.X)):

vec = vector(self.X[i],self.p[i])

A = np.append(A, vec,1)

self.A = np.matrix(A)

def lamb(self):

lamb = np.ndarray(shape = (self.A.shape[1],0), dtype = float)

for i in range(self.deg[3]):

if self.splitted\_lambdas:

boundary\_1 = self.p[0] \* self.deg[0]

boundary\_2 = self.p[1] \* self.deg[1] + boundary\_1

lamb1 = self.\_minimize\_equation(self.A[:, :boundary\_1], self.B[:, i])

lamb2 = self.\_minimize\_equation(self.A[:, boundary\_1:boundary\_2], self.B[:, i])

lamb3 = self.\_minimize\_equation(self.A[:, boundary\_2:], self.B[:, i])

lamb = np.append(lamb, np.concatenate((lamb1, lamb2, lamb3)), axis=1)

else:

lamb = np.append(lamb, self.\_minimize\_equation(self.A, self.B[:, i]), axis=1)

self.Lamb = np.matrix(lamb) #Lamb in full events

def psi(self):

def built\_psi(lamb):

'''

return matrix xi1 for b1 as matrix

:param A:

:param lamb:

:param p:

:return: matrix psi, for each Y

'''

psi = np.ndarray(shape=(self.n, self.mX), dtype = float)

q = 0 #iterator in lamb and A

l = 0 #iterator in columns psi

for k in range(len(self.X)): # choose X1 or X2 or X3

for s in range(self.X[k].shape[1]):# choose X11 or X12 or X13

for i in range(self.X[k].shape[0]):

psi[i,l] = self.A[i,q:q+self.p[k]]\*lamb[q:q+self.p[k], 0]

q+=self.p[k]

l+=1

return np.matrix(psi)

self.Psi = [] #as list because psi[i] is matrix(not vector)

for i in range(self.deg[3]):

self.Psi.append(built\_psi(self.Lamb[:,i]))

def built\_a(self):

self.a = np.ndarray(shape=(self.mX,0), dtype=float)

for i in range(self.deg[3]):

a1 = self.\_minimize\_equation(self.Psi[i][:, :self.degf[0]], self.Y[:, i])

a2 = self.\_minimize\_equation(self.Psi[i][:, self.degf[0]:self.degf[1]], self.Y[:, i])

a3 = self.\_minimize\_equation(self.Psi[i][:, self.degf[1]:], self.Y[:, i])

# temp = self.\_minimize\_equation(self.Psi[i], self.Y[:, i])

# self.a = np.append(self.a, temp, axis=1)

self.a = np.append(self.a, np.vstack((a1, a2, a3)),axis = 1)

def built\_F1i(self, psi, a):

'''

not use; it used in next function

:param psi: matrix psi (only one

:param a: vector with shape = (6,1)

:param degf: = [3,4,6]//fibonachi of deg

:return: matrix of (three) components with F1 F2 and F3

'''

m = len(self.X) # m = 3

F1i = np.ndarray(shape = (self.n,m),dtype = float)

k = 0 #point of begining columnt to multipy

for j in range(m): # 0 - 2

for i in range(self.n): # 0 - 49

F1i[i,j] = psi[i,k:self.degf[j]]\*a[k:self.degf[j],0]

k = self.degf[j]

return np.matrix(F1i)

def built\_Fi(self):

self.Fi = []

for i in range(self.deg[3]):

self.Fi.append(self.built\_F1i(self.Psi[i],self.a[:,i]))

def built\_c(self):

self.c = np.ndarray(shape = (len(self.X),0),dtype = float)

for i in range(self.deg[3]):

self.c = np.append(self.c, conjugate\_gradient\_method(self.Fi[i].T\*self.Fi[i], self.Fi[i].T\*self.Y[:,i],self.eps),\

axis = 1)

def built\_F(self):

F = np.ndarray(self.Y.shape, dtype = float)

for j in range(F.shape[1]):#2

for i in range(F.shape[0]): #50

F[i,j] = self.Fi[j][i,:]\*self.c[:,j]

self.F = np.matrix(F)

self.norm\_error = []

for i in range(self.Y.shape[1]):

self.norm\_error.append(np.linalg.norm(self.Y[:,i] - self.F[:,i],np.inf))

def built\_F\_(self):

minY = self.Y\_.min(axis=0)

maxY = self.Y\_.max(axis=0)

self.F\_ = np.multiply(self.F,maxY - minY) + minY

self.error = []

for i in range(self.Y\_.shape[1]):

self.error.append(np.linalg.norm(self.Y\_[:,i] - self.F\_[:,i],np.inf))

def save\_to\_file(self):

wb = Workbook()

#get active worksheet

ws = wb.active

l = [None]

ws.append(['Введенные данные: X'])

for i in range(self.n):

ws.append(l+self.datas[i,:self.degf[3]].tolist()[0])

ws.append([])

ws.append(['Введенные данные: Y'])

for i in range(self.n):

ws.append(l+self.datas[i,self.degf[2]:self.degf[3]].tolist()[0])

ws.append([])

ws.append(['X нормализованные:'])

for i in range(self.n):

ws.append(l+self.data[i,:self.degf[2]].tolist()[0])

ws.append([])

ws.append(['Y нормализованные:'])

for i in range(self.n):

ws.append(l+self.data[i,self.degf[2]:self.degf[3]].tolist()[0])

ws.append([])

ws.append(['матр B:'])

for i in range(self.n):

ws.append(l+self.B[i].tolist()[0])

ws.append([])

ws.append(['матр A:'])

for i in range(self.A.shape[0]):

ws.append(l+self.A[i].tolist()[0])

ws.append([])

ws.append(['матр Lambda:'])

for i in range(self.Lamb.shape[0]):

ws.append(l+self.Lamb[i].tolist()[0])

ws.append([])

for j in range(len(self.Psi)):

s = 'матр Psi%i:' %(j+1)

ws.append([s])

for i in range(self.n):

ws.append(l+self.Psi[j][i].tolist()[0])

ws.append([])

ws.append(['матр a:'])

for i in range(self.mX):

ws.append(l+self.a[i].tolist()[0])

ws.append([])

for j in range(len(self.Fi)):

s = 'Матр Ф%i:' %(j+1)

ws.append([s])

for i in range(self.Fi[j].shape[0]):

ws.append(l+self.Fi[j][i].tolist()[0])

ws.append([])

ws.append(['матр c:'])

for i in range(len(self.X)):

ws.append(l+self.c[i].tolist()[0])

ws.append([])

ws.append(['Y перестроенное нормализованное :'])

for i in range(self.n):

ws.append(l+self.F[i].tolist()[0])

ws.append([])

ws.append(['Y перестроенное :'])

for i in range(self.n):

ws.append(l+self.F\_[i].tolist()[0])

ws.append([])

ws.append(['Нормализовная невязка (Y -Ф)'])

ws.append(l + self.norm\_error)

ws.append(['невязка (Y\_ - Ф\_))'])

ws.append(l+self.error)

wb.save(self.filename\_output)

def show(self):

text = []

text.append('Вводные данные: X')

text.append(tb(np.array(self.datas[:, :self.degf[2]])))

text.append('\nВводные данные: Y')

text.append(tb(np.array(self.datas[:,self.degf[2]:self.degf[3]])))

text.append('\nX нормализованный:')

text.append(tb(np.array(self.data[:,:self.degf[2]])))

text.append('\nY нормализованный:')

text.append(tb(np.array(self.data[:,self.degf[2]:self.degf[3]])))

text.append('\nматрица B:')

text.append(tb(np.array(self.B)))

text.append('\nматрица A:')

text.append(tb(np.array(self.A)))

text.append('\nматрица Lambda:')

text.append(tb(np.array(self.Lamb)))

for j in range(len(self.Psi)):

s = '\nматрица Psi%i:' %(j+1)

text.append(s)

text.append(tb(np.array(self.Psi[j])))

text.append('\nматрица a:')

text.append(tb(self.a.tolist()))

for j in range(len(self.Fi)):

s = '\nматрица Ф%i:' %(j+1)

text.append(s)

text.append(tb(np.array(self.Fi[j])))

text.append('\nматрица c:')

text.append(tb(np.array(self.c)))

text.append('\nY перестроенное нормализованное :')

text.append(tb(np.array(self.F)))

text.append('\nY перестроенное :')

text.append(tb(self.F\_.tolist()))

text.append('\nНормализованная невязка (Y - Ф)')

text.append(tb([self.norm\_error]))

text.append('\nНевязка (Y\_ - Ф\_))')

text.append(tb([self.error]))

return '\n'.join(text)

def prepare(self):

self.define\_data()

self.norm\_data()

self.define\_norm\_vectors()

self.built\_B()

self.poly\_func()

self.built\_A()

self.lamb()

self.psi()

self.built\_a()

self.built\_Fi()

self.built\_c()

self.built\_F()

self.built\_F\_()

self.save\_to\_file()

**system\_solve.py**

\_\_author\_\_ = 'KA\_41\_1'

import numpy as np

def conjugate\_gradient\_method(A, b, eps):

'''

Conjugate Gradient Method that solve equation Ax = b with given accuracy

:param A:matrix A

:param b:vector b

:param eps: accuracy

:return: solution x

'''

n = len(A.T) # number column

xi1 = xi = np.zeros(shape=(n,1), dtype = float)

vi = ri = b # start condition

i = 0 #loop for number iteration

while True:

try:

i+= 1

ai = float(vi.T\*ri)/float(vi.T\*A\*vi) # alpha i

xi1 = xi+ai\*vi # x i+1

ri1 = ri-ai\*A\*vi # r i+1

betai = -float(vi.T\*A\*ri1)/float(vi.T\*A\*vi) # beta i

vi1 = ri1+betai\*vi

if (np.linalg.norm(ri1)<eps) or i > 10 \* n:

break

else:

xi,vi,ri = xi1,vi1,ri1

except Exception:

print("problem with minimization")

return np.matrix(xi1)

**main\_window.ui**

<?xml version="1.0" encoding="UTF-8"?>

<ui version="4.0">

<class>Form</class>

<widget class="QWidget" name="Form">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>650</width>

<height>733</height>

</rect>

</property>

<property name="windowTitle">

<string>Form1</string>

</property>

<layout class="QGridLayout" name="gridLayout\_2">

<item row="2" column="2">

<widget class="QGroupBox" name="groupBox\_2">

<property name="sizePolicy">

<sizepolicy hsizetype="Fixed" vsizetype="Preferred">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="title">

<string>Dimensions</string>

</property>

<layout class="QGridLayout" name="gridLayout\_3">

<item row="1" column="1">

<widget class="QSpinBox" name="x1\_dim">

<property name="value">

<number>2</number>

</property>

</widget>

</item>

<item row="1" column="0">

<widget class="QLabel" name="label\_2">

<property name="text">

<string>X1</string>

</property>

<property name="alignment">

<set>Qt::AlignRight|Qt::AlignTrailing|Qt::AlignVCenter</set>

</property>

</widget>

</item>

<item row="2" column="1">

<widget class="QSpinBox" name="x2\_dim">

<property name="value">

<number>2</number>

</property>

</widget>

</item>

<item row="2" column="0">

<widget class="QLabel" name="label\_3">

<property name="text">

<string>X2</string>

</property>

<property name="alignment">

<set>Qt::AlignRight|Qt::AlignTrailing|Qt::AlignVCenter</set>

</property>

</widget>

</item>

<item row="2" column="2">

<widget class="QLabel" name="label\_5">

<property name="text">

<string>X3</string>

</property>

<property name="alignment">

<set>Qt::AlignRight|Qt::AlignTrailing|Qt::AlignVCenter</set>

</property>

</widget>

</item>

<item row="1" column="2">

<widget class="QLabel" name="label\_4">

<property name="text">

<string>Y</string>

</property>

<property name="alignment">

<set>Qt::AlignRight|Qt::AlignTrailing|Qt::AlignVCenter</set>

</property>

</widget>

</item>

<item row="2" column="3">

<widget class="QSpinBox" name="x3\_dim">

<property name="value">

<number>3</number>

</property>

</widget>

</item>

<item row="1" column="3">

<widget class="QSpinBox" name="y\_dim">

<property name="value">

<number>4</number>

</property>

</widget>

</item>

</layout>

</widget>

</item>

<item row="2" column="3">

<widget class="QGroupBox" name="groupBox\_6">

<property name="title">

<string>Processing</string>

</property>

<layout class="QHBoxLayout" name="horizontalLayout\_2">

<item>

<widget class="QPushButton" name="plot\_button">

<property name="text">

<string>Plot</string>

</property>

</widget>

</item>

<item>

<widget class="QPushButton" name="exec\_button">

<property name="text">

<string>Execute</string>

</property>

<property name="default">

<bool>true</bool>

</property>

<property name="flat">

<bool>false</bool>

</property>

</widget>

</item>

</layout>

</widget>

</item>

<item row="6" column="0" rowspan="2" colspan="6">

<widget class="QTextBrowser" name="results\_field">

<property name="lineWrapMode">

<enum>QTextEdit::NoWrap</enum>

</property>

<property name="readOnly">

<bool>true</bool>

</property>

<property name="html">

<string>&lt;!DOCTYPE HTML PUBLIC &quot;-//W3C//DTD HTML 4.0//EN&quot; &quot;http://www.w3.org/TR/REC-html40/strict.dtd&quot;&gt;

&lt;html&gt;&lt;head&gt;&lt;meta name=&quot;qrichtext&quot; content=&quot;1&quot; /&gt;&lt;style type=&quot;text/css&quot;&gt;

p, li { white-space: pre-wrap; }

&lt;/style&gt;&lt;/head&gt;&lt;body style=&quot; font-family:'Ubuntu'; font-size:11pt; font-weight:400; font-style:normal;&quot;&gt;

&lt;p style=&quot;-qt-paragraph-type:empty; margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; -qt-block-indent:0; text-indent:0px; font-family:'.Menio Regular'; font-size:10pt;&quot;&gt;&lt;br /&gt;&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>

</property>

<property name="placeholderText">

<string>Results are shown here</string>

</property>

</widget>

</item>

<item row="0" column="4">

<widget class="QGroupBox" name="groupBox\_5">

<property name="title">

<string>Polynomials</string>

</property>

<layout class="QHBoxLayout" name="horizontalLayout">

<item>

<widget class="QGroupBox" name="groupBox\_4">

<property name="title">

<string>Degreees</string>

</property>

<layout class="QGridLayout" name="gridLayout\_4">

<item row="0" column="0">

<widget class="QLabel" name="label\_6">

<property name="text">

<string>X1</string>

</property>

<property name="alignment">

<set>Qt::AlignRight|Qt::AlignTrailing|Qt::AlignVCenter</set>

</property>

</widget>

</item>

<item row="2" column="1">

<widget class="QSpinBox" name="x3\_deg">

<property name="value">

<number>4</number>

</property>

</widget>

</item>

<item row="1" column="0">

<widget class="QLabel" name="label\_8">

<property name="text">

<string>X2</string>

</property>

<property name="alignment">

<set>Qt::AlignRight|Qt::AlignTrailing|Qt::AlignVCenter</set>

</property>

</widget>

</item>

<item row="0" column="1">

<widget class="QSpinBox" name="x1\_deg">

<property name="value">

<number>7</number>

</property>

</widget>

</item>

<item row="1" column="1">

<widget class="QSpinBox" name="x2\_deg">

<property name="value">

<number>7</number>

</property>

</widget>

</item>

<item row="2" column="0">

<widget class="QLabel" name="label\_7">

<property name="text">

<string>X3</string>

</property>

<property name="alignment">

<set>Qt::AlignRight|Qt::AlignTrailing|Qt::AlignVCenter</set>

</property>

</widget>

</item>

</layout>

</widget>

</item>

</layout>

</widget>

</item>

<item row="0" column="0" colspan="3">

<widget class="QGroupBox" name="groupBox">

<property name="autoFillBackground">

<bool>false</bool>

</property>

<property name="title">

<string>Data</string>

</property>

<layout class="QGridLayout" name="gridLayout\_5">

<item row="2" column="1">

<widget class="QToolButton" name="select\_output">

<property name="text">

<string>...</string>

</property>

</widget>

</item>

<item row="0" column="1">

<widget class="QSpinBox" name="sample\_spin">

<property name="maximum">

<number>999</number>

</property>

<property name="value">

<number>45</number>

</property>

</widget>

</item>

<item row="1" column="1">

<widget class="QToolButton" name="select\_input">

<property name="text">

<string>...</string>

</property>

</widget>

</item>

<item row="1" column="0">

<widget class="QLineEdit" name="line\_input">

<property name="placeholderText">

<string>Input file</string>

</property>

</widget>

</item>

<item row="0" column="0">

<widget class="QLabel" name="label">

<property name="text">

<string>Number of samples:</string>

</property>

</widget>

</item>

<item row="2" column="0">

<widget class="QLineEdit" name="line\_output">

<property name="placeholderText">

<string>Output file</string>

</property>

</widget>

</item>

</layout>

</widget>

</item>

<item row="0" column="3">

<widget class="QGroupBox" name="groupBox\_3">

<property name="title">

<string>Types</string>

</property>

<layout class="QVBoxLayout" name="verticalLayout">

<item>

<widget class="QRadioButton" name="radio\_cheb1">

<property name="text">

<string>Chebyshev</string>

</property>

<property name="checked">

<bool>true</bool>

</property>

</widget>

</item>

<item>

<widget class="QRadioButton" name="radio\_legend">

<property name="text">

<string>Legendre</string>

</property>

</widget>

</item>

<item>

<widget class="QRadioButton" name="radio\_lagg">

<property name="text">

<string>Laguerre</string>

</property>

</widget>

</item>

<item>

<widget class="QRadioButton" name="radio\_herm">

<property name="text">

<string>Hermit</string>

</property>

</widget>

</item>

<item>

<widget class="QRadioButton" name="radio\_cheb2">

<property name="text">

<string>Chebyshev II</string>

</property>

</widget>

</item>

</layout>

</widget>

</item>

<item row="2" column="4">

<widget class="QGroupBox" name="groupBox\_7">

<property name="title">

<string>Additional</string>

</property>

<layout class="QHBoxLayout" name="horizontalLayout\_3">

<item>

<widget class="QLabel" name="label\_9">

<property name="text">

<string>Weights:</string>

</property>

</widget>

</item>

<item>

<widget class="QComboBox" name="weights\_box">

<item>

<property name="text">

<string>Average</string>

</property>

</item>

<item>

<property name="text">

<string>Scaled</string>

</property>

</item>

</widget>

</item>

</layout>

</widget>

</item>

<item row="1" column="4">

<widget class="QCheckBox" name="lambda\_check">

<property name="text">

<string>Use 3-block lambda calculation</string>

</property>

</widget>

</item>

</layout>

</widget>

<tabstops>

<tabstop>radio\_cheb1</tabstop>

<tabstop>radio\_cheb2</tabstop>

<tabstop>radio\_legend</tabstop>

<tabstop>radio\_lagg</tabstop>

<tabstop>radio\_herm</tabstop>

<tabstop>x1\_deg</tabstop>

<tabstop>x2\_deg</tabstop>

<tabstop>x3\_deg</tabstop>

<tabstop>x1\_dim</tabstop>

<tabstop>x2\_dim</tabstop>

<tabstop>sample\_spin</tabstop>

<tabstop>line\_input</tabstop>

<tabstop>select\_input</tabstop>

<tabstop>line\_output</tabstop>

<tabstop>select\_output</tabstop>

<tabstop>exec\_button</tabstop>

<tabstop>results\_field</tabstop>

</tabstops>

<resources/>

<connections>

<connection>

<sender>exec\_button</sender>

<signal>clicked()</signal>

<receiver>Form</receiver>

<slot>exec\_clicked()</slot>

<hints>

<hint type="sourcelabel">

<x>379</x>

<y>219</y>

</hint>

<hint type="destinationlabel">

<x>309</x>

<y>199</y>

</hint>

</hints>

</connection>

<connection>

<sender>plot\_button</sender>

<signal>clicked()</signal>

<receiver>Form</receiver>

<slot>plot\_clicked()</slot>

<hints>

<hint type="sourcelabel">

<x>526</x>

<y>219</y>

</hint>

<hint type="destinationlabel">

<x>309</x>

<y>199</y>

</hint>

</hints>

</connection>

<connection>

<sender>sample\_spin</sender>

<signal>valueChanged(int)</signal>

<receiver>Form</receiver>

<slot>samples\_modified(int)</slot>

<hints>

<hint type="sourcelabel">

<x>568</x>

<y>53</y>

</hint>

<hint type="destinationlabel">

<x>309</x>

<y>199</y>

</hint>

</hints>

</connection>

<connection>

<sender>select\_input</sender>

<signal>clicked()</signal>

<receiver>Form</receiver>

<slot>input\_clicked()</slot>

<hints>

<hint type="sourcelabel">

<x>557</x>

<y>95</y>

</hint>

<hint type="destinationlabel">

<x>309</x>

<y>199</y>

</hint>

</hints>

</connection>

<connection>

<sender>select\_output</sender>

<signal>clicked()</signal>

<receiver>Form</receiver>

<slot>output\_clicked()</slot>

<hints>

<hint type="sourcelabel">

<x>557</x>

<y>136</y>

</hint>

<hint type="destinationlabel">

<x>309</x>

<y>199</y>

</hint>

</hints>

</connection>

<connection>

<sender>x1\_dim</sender>

<signal>valueChanged(int)</signal>

<receiver>Form</receiver>

<slot>dimension\_modified(int)</slot>

<hints>

<hint type="sourcelabel">

<x>361</x>

<y>44</y>

</hint>

<hint type="destinationlabel">

<x>309</x>

<y>199</y>

</hint>

</hints>

</connection>

<connection>

<sender>x2\_dim</sender>

<signal>valueChanged(int)</signal>

<receiver>Form</receiver>

<slot>dimension\_modified(int)</slot>

<hints>

<hint type="sourcelabel">

<x>361</x>

<y>78</y>

</hint>

<hint type="destinationlabel">

<x>309</x>

<y>199</y>

</hint>

</hints>

</connection>

<connection>

<sender>x3\_dim</sender>

<signal>valueChanged(int)</signal>

<receiver>Form</receiver>

<slot>dimension\_modified(int)</slot>

<hints>

<hint type="sourcelabel">

<x>361</x>

<y>112</y>

</hint>

<hint type="destinationlabel">

<x>309</x>

<y>199</y>

</hint>

</hints>

</connection>

<connection>

<sender>x1\_deg</sender>

<signal>valueChanged(int)</signal>

<receiver>Form</receiver>

<slot>degree\_modified(int)</slot>

<hints>

<hint type="sourcelabel">

<x>227</x>

<y>81</y>

</hint>

<hint type="destinationlabel">

<x>309</x>

<y>199</y>

</hint>

</hints>

</connection>

<connection>

<sender>x2\_deg</sender>

<signal>valueChanged(int)</signal>

<receiver>Form</receiver>

<slot>degree\_modified(int)</slot>

<hints>

<hint type="sourcelabel">

<x>227</x>

<y>131</y>

</hint>

<hint type="destinationlabel">

<x>309</x>

<y>199</y>

</hint>

</hints>

</connection>

<connection>

<sender>x3\_deg</sender>

<signal>valueChanged(int)</signal>

<receiver>Form</receiver>

<slot>degree\_modified(int)</slot>

<hints>

<hint type="sourcelabel">

<x>244</x>

<y>133</y>

</hint>

<hint type="destinationlabel">

<x>309</x>

<y>199</y>

</hint>

</hints>

</connection>

<connection>

<sender>radio\_cheb1</sender>

<signal>toggled(bool)</signal>

<receiver>Form</receiver>

<slot>type\_modified(bool)</slot>

<hints>

<hint type="sourcelabel">

<x>86</x>

<y>78</y>

</hint>

<hint type="destinationlabel">

<x>309</x>

<y>199</y>

</hint>

</hints>

</connection>

<connection>

<sender>radio\_cheb2</sender>

<signal>toggled(bool)</signal>

<receiver>Form</receiver>

<slot>type\_modified(bool)</slot>

<hints>

<hint type="sourcelabel">

<x>86</x>

<y>78</y>

</hint>

<hint type="destinationlabel">

<x>309</x>

<y>199</y>

</hint>

</hints>

</connection>

<connection>

<sender>radio\_herm</sender>

<signal>toggled(bool)</signal>

<receiver>Form</receiver>

<slot>type\_modified(bool)</slot>

<hints>

<hint type="sourcelabel">

<x>86</x>

<y>186</y>

</hint>

<hint type="destinationlabel">

<x>309</x>

<y>199</y>

</hint>

</hints>

</connection>

<connection>

<sender>radio\_lagg</sender>

<signal>toggled(bool)</signal>

<receiver>Form</receiver>

<slot>type\_modified(bool)</slot>

<hints>

<hint type="sourcelabel">

<x>86</x>

<y>150</y>

</hint>

<hint type="destinationlabel">

<x>309</x>

<y>199</y>

</hint>

</hints>

</connection>

<connection>

<sender>radio\_legend</sender>

<signal>toggled(bool)</signal>

<receiver>Form</receiver>

<slot>type\_modified(bool)</slot>

<hints>

<hint type="sourcelabel">

<x>86</x>

<y>114</y>

</hint>

<hint type="destinationlabel">

<x>309</x>

<y>199</y>

</hint>

</hints>

</connection>

<connection>

<sender>Form</sender>

<signal>output\_changed(QString)</signal>

<receiver>line\_output</receiver>

<slot>setText(QString)</slot>

<hints>

<hint type="sourcelabel">

<x>309</x>

<y>199</y>

</hint>

<hint type="destinationlabel">

<x>483</x>

<y>136</y>

</hint>

</hints>

</connection>

<connection>

<sender>Form</sender>

<signal>input\_changed(QString)</signal>

<receiver>line\_input</receiver>

<slot>setText(QString)</slot>

<hints>

<hint type="sourcelabel">

<x>309</x>

<y>199</y>

</hint>

<hint type="destinationlabel">

<x>483</x>

<y>95</y>

</hint>

</hints>

</connection>

<connection>

<sender>line\_input</sender>

<signal>textChanged(QString)</signal>

<receiver>Form</receiver>

<slot>input\_modified(QString)</slot>

<hints>

<hint type="sourcelabel">

<x>483</x>

<y>95</y>

</hint>

<hint type="destinationlabel">

<x>309</x>

<y>199</y>

</hint>

</hints>

</connection>

<connection>

<sender>line\_output</sender>

<signal>textChanged(QString)</signal>

<receiver>Form</receiver>

<slot>output\_modified(QString)</slot>

<hints>

<hint type="sourcelabel">

<x>483</x>

<y>136</y>

</hint>

<hint type="destinationlabel">

<x>309</x>

<y>199</y>

</hint>

</hints>

</connection>

<connection>

<sender>lambda\_check</sender>

<signal>toggled(bool)</signal>

<receiver>Form</receiver>

<slot>lambda\_calc\_method\_changed(bool)</slot>

<hints>

<hint type="sourcelabel">

<x>301</x>

<y>219</y>

</hint>

<hint type="destinationlabel">

<x>309</x>

<y>199</y>

</hint>

</hints>

</connection>

<connection>

<sender>weights\_box</sender>

<signal>currentIndexChanged(QString)</signal>

<receiver>Form</receiver>

<slot>weights\_modified(QString)</slot>

<hints>

<hint type="sourcelabel">

<x>136</x>

<y>220</y>

</hint>

<hint type="destinationlabel">

<x>309</x>

<y>199</y>

</hint>

</hints>

</connection>

<connection>

<sender>y\_dim</sender>

<signal>valueChanged(int)</signal>

<receiver>Form</receiver>

<slot>dimension\_modified(int)</slot>

<hints>

<hint type="sourcelabel">

<x>378</x>

<y>146</y>

</hint>

<hint type="destinationlabel">

<x>311</x>

<y>299</y>

</hint>

</hints>

</connection>

</connections>

<slots>

<signal>input\_changed(QString)</signal>

<signal>output\_changed(QString)</signal>

<signal>x1\_dim\_changed(int)</signal>

<signal>x2\_dim\_changed(int)</signal>

<signal>x3\_dim\_changed(int)</signal>

<signal>x1\_deg\_changed(int)</signal>

<signal>x2\_deg\_changed(int)</signal>

<signal>x3\_deg\_changed(int)</signal>

<signal>type\_cheb()</signal>

<signal>type\_lege()</signal>

<signal>type\_lagg()</signal>

<signal>type\_herm()</signal>

<slot>input\_clicked()</slot>

<slot>output\_clicked()</slot>

<slot>samples\_modified(int)</slot>

<slot>dimension\_modified(int)</slot>

<slot>degree\_modified(int)</slot>

<slot>type\_modified(bool)</slot>

<slot>plot\_clicked()</slot>

<slot>exec\_clicked()</slot>

<slot>input\_modified(QString)</slot>

<slot>output\_modified(QString)</slot>

<slot>weights\_modified(QString)</slot>

<slot>lambda\_calc\_method\_changed(bool)</slot>

</slots>

</ui>