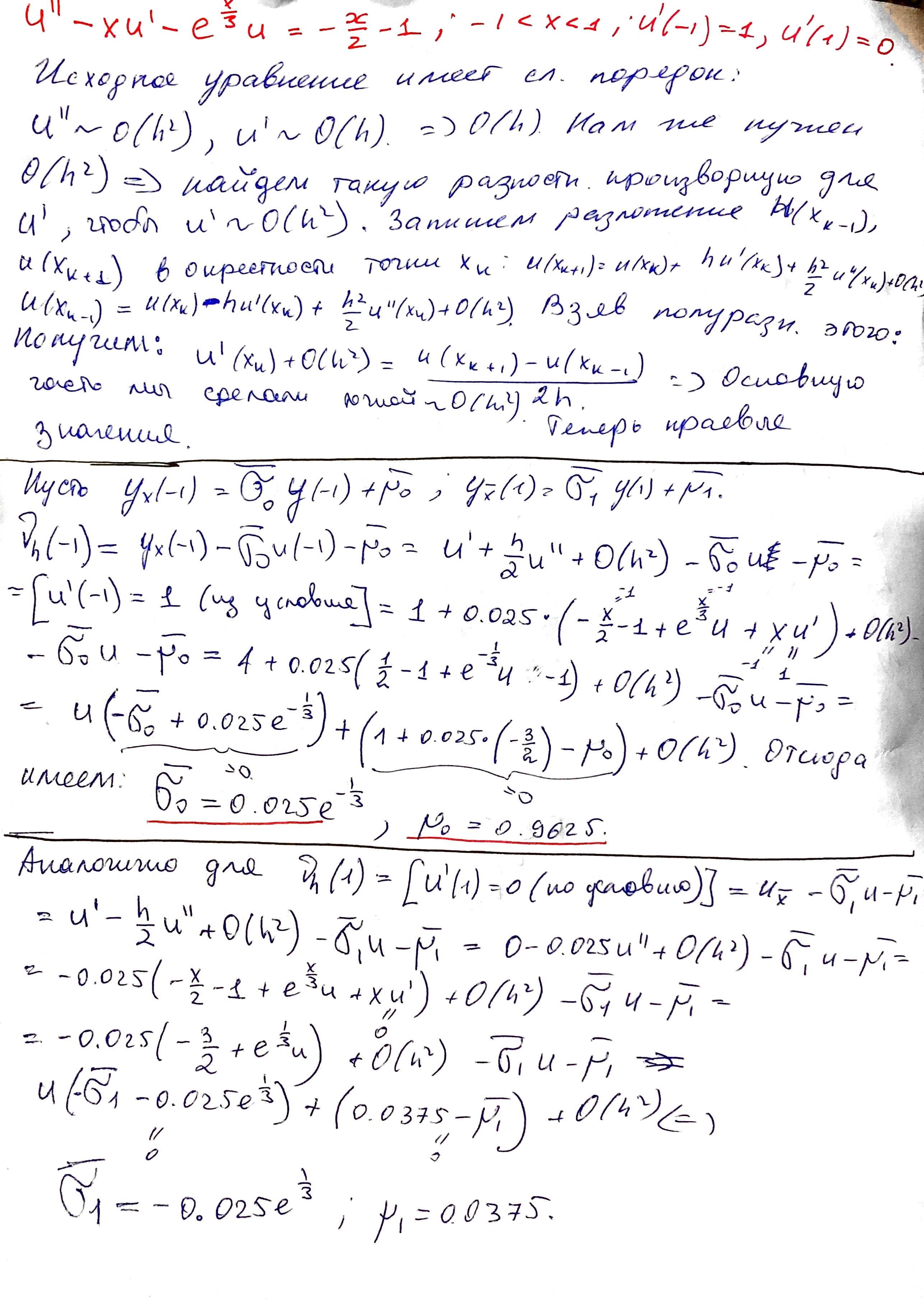
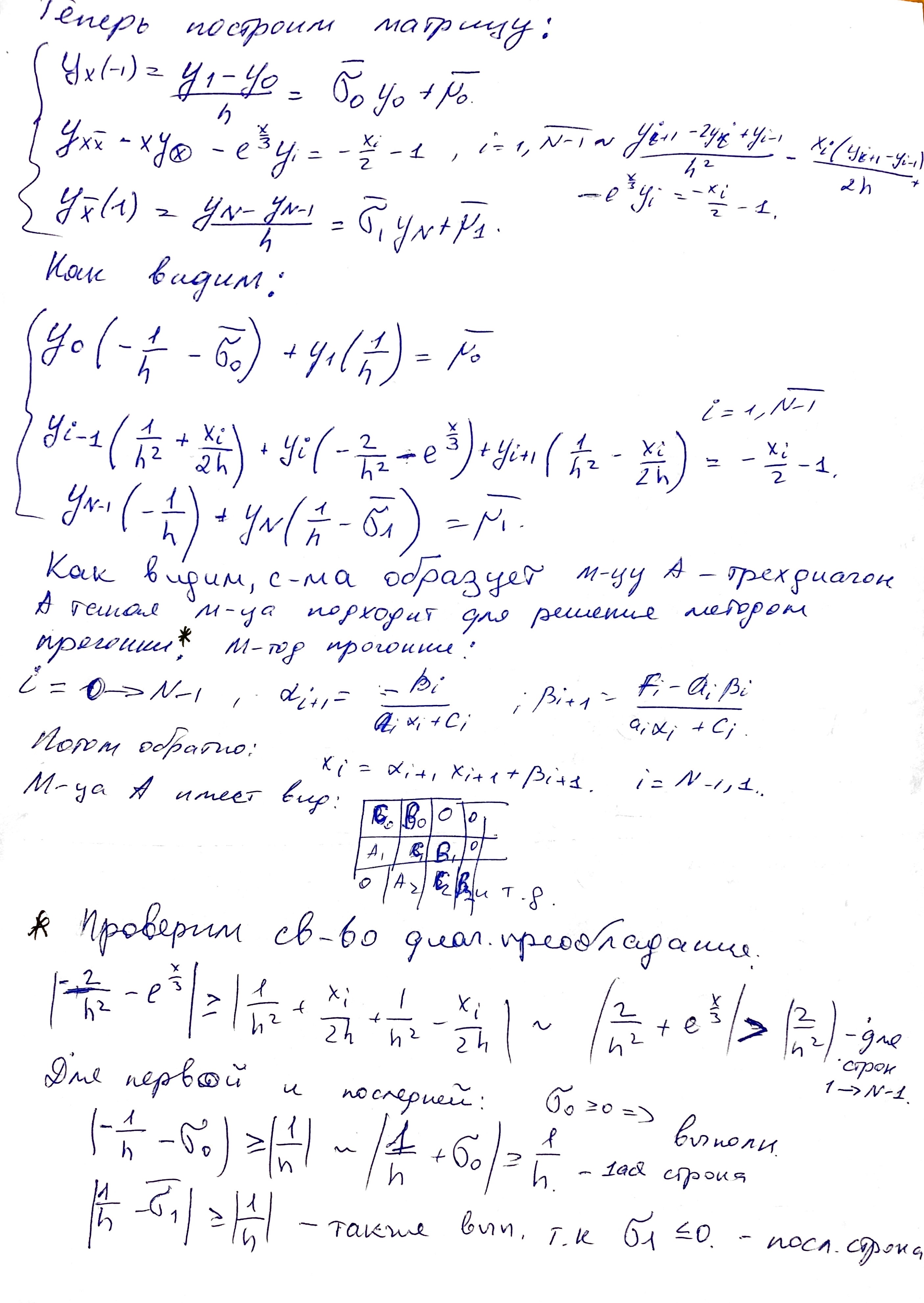
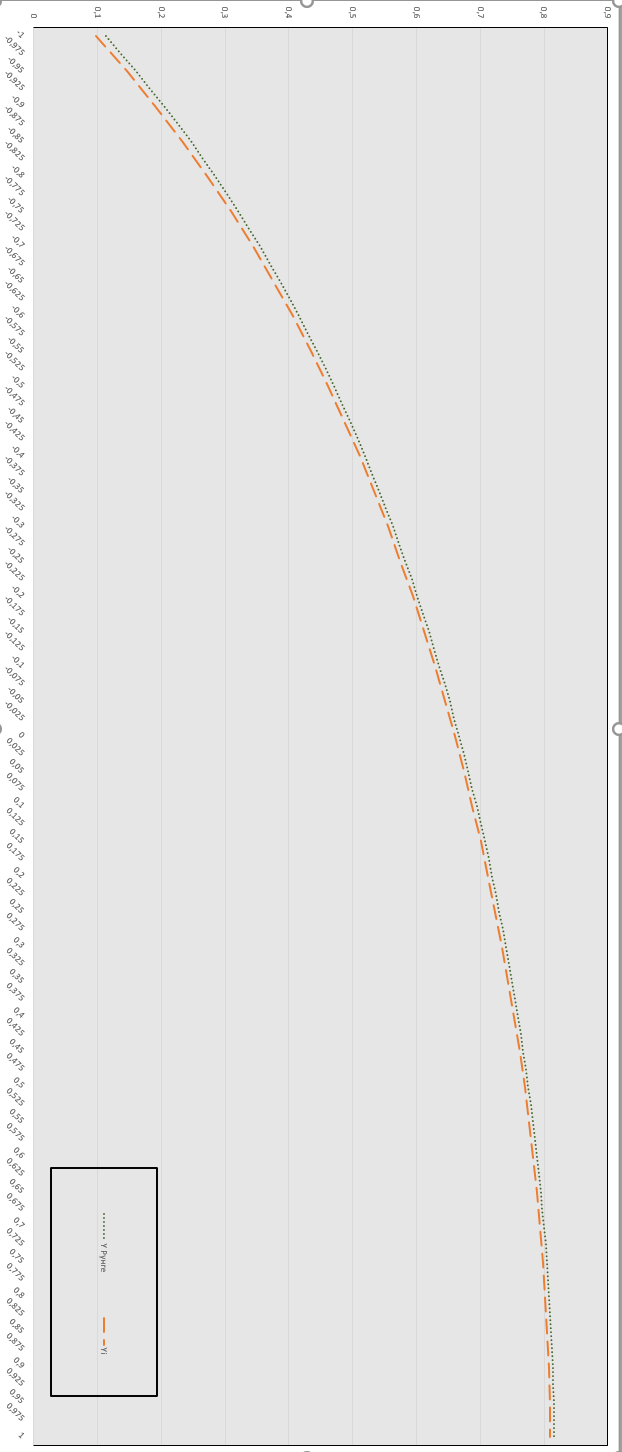
**Лабораторная работа №2 Кулич Святослав Петрович.**

**Курс 3, группа №3.**





**Вот** график полученной дискретной функции. Причем Y Рунге – y, найденный с шагом 0,025.



Yi – с шагом 0,05.

Вот полученные данные:

|  |  |  |  |
| --- | --- | --- | --- |
| i | Xi | Y Рунге | Yi |
| 0 | -1 | 0,11321 | 0,09771 |
| 1 | -0,975 | 0,13733 |  |
| 2 | -0,95 | 0,16061 | 0,14592 |
| 3 | -0,925 | 0,18308 |  |
| 4 | -0,9 | 0,2048 | 0,19087 |
| 5 | -0,875 | 0,22579 |  |
| 6 | -0,85 | 0,24608 | 0,23285 |
| 7 | -0,825 | 0,2657 |  |
| 8 | -0,8 | 0,28468 | 0,2721 |
| 9 | -0,775 | 0,30305 |  |
| 10 | -0,75 | 0,32083 | 0,30885 |
| 11 | -0,725 | 0,33805 |  |
| 12 | -0,7 | 0,35474 | 0,34331 |
| 13 | -0,675 | 0,3709 |  |
| 14 | -0,65 | 0,38656 | 0,37565 |
| 15 | -0,625 | 0,40175 |  |
| 16 | -0,6 | 0,41648 | 0,40604 |
| 17 | -0,575 | 0,43077 |  |
| 18 | -0,55 | 0,44463 | 0,43462 |
| 19 | -0,525 | 0,45808 |  |
| 20 | -0,5 | 0,47113 | 0,46154 |
| 21 | -0,475 | 0,48381 |  |
| 22 | -0,45 | 0,49612 | 0,4869 |
| 23 | -0,425 | 0,50807 |  |
| 24 | -0,4 | 0,51969 | 0,51082 |
| 25 | -0,375 | 0,53097 |  |
| 26 | -0,35 | 0,54194 | 0,5334 |
| 27 | -0,325 | 0,55259 |  |
| 28 | -0,3 | 0,56295 | 0,55471 |
| 29 | -0,275 | 0,57301 |  |
| 30 | -0,25 | 0,5828 | 0,57484 |
| 31 | -0,225 | 0,59231 |  |
| 32 | -0,2 | 0,60157 | 0,59386 |
| 33 | -0,175 | 0,61056 |  |
| 34 | -0,15 | 0,6193 | 0,61184 |
| 35 | -0,125 | 0,62781 |  |
| 36 | -0,1 | 0,63607 | 0,62882 |
| 37 | -0,075 | 0,64411 |  |
| 38 | -0,05 | 0,65192 | 0,64487 |
| 39 | -0,025 | 0,65952 |  |
| 40 | 0 | 0,6669 | 0,66003 |
| 41 | 0,025 | 0,67407 |  |
| 42 | 0,05 | 0,68104 | 0,67434 |
| 43 | 0,075 | 0,68782 |  |
| 44 | 0,1 | 0,69439 | 0,68784 |
| 45 | 0,125 | 0,70078 |  |
| 46 | 0,15 | 0,70697 | 0,70055 |
| 47 | 0,175 | 0,71299 |  |
| 48 | 0,2 | 0,71882 | 0,71251 |
| 49 | 0,225 | 0,72447 |  |
| i | Xi | Y Рунге | Yi |
| 50 | 0,25 | 0,72994 | 0,72374 |
| 51 | 0,275 | 0,73525 |  |
| 52 | 0,3 | 0,74038 | 0,73426 |
| 53 | 0,325 | 0,74534 |  |
| 54 | 0,35 | 0,75013 | 0,74409 |
| 55 | 0,375 | 0,75476 |  |
| 56 | 0,4 | 0,75922 | 0,75323 |
| 57 | 0,425 | 0,76352 |  |
| 58 | 0,45 | 0,76766 | 0,7617 |
| 59 | 0,475 | 0,77163 |  |
| 60 | 0,5 | 0,77544 | 0,76951 |
| 61 | 0,525 | 0,77909 |  |
| 62 | 0,55 | 0,78258 | 0,77665 |
| 63 | 0,575 | 0,7859 |  |
| 64 | 0,6 | 0,78906 | 0,78312 |
| 65 | 0,625 | 0,79206 |  |
| 66 | 0,65 | 0,79489 | 0,78892 |
| 67 | 0,675 | 0,79756 |  |
| 68 | 0,7 | 0,80005 | 0,79403 |
| 69 | 0,725 | 0,80238 |  |
| 70 | 0,75 | 0,80454 | 0,79844 |
| 71 | 0,775 | 0,80651 |  |
| 72 | 0,8 | 0,80832 | 0,80213 |
| 73 | 0,825 | 0,80994 |  |
| 74 | 0,85 | 0,81137 | 0,80507 |
| 75 | 0,875 | 0,81262 |  |
| 76 | 0,9 | 0,81367 | 0,80723 |
| 77 | 0,925 | 0,81453 |  |
| 78 | 0,95 | 0,81518 | 0,80856 |
| 79 | 0,975 | 0,81562 |  |
| 80 | 1 | 0,81585 | 0,80902 |

Листинг:

**public class** Main {  
 **private final static double *A*** = -1D;  
 **private final static double *B*** = 1D;  
  
 **private final static double *MU0*** = 0.9625;  
 **private final static double *SIGMA0*** = 0.025 \* Math.*exp*(-1F / 3);  
  
 **private final static double *MU1*** = 0.0375;  
 **private final static double *SIGMA1*** = -0.025 \* Math.*exp*(1F / 3);  
  
 **private final static double *h1*** = 0.05;  
 **private final static int *SIZE1*** = (**int**) ((***B*** - ***A***) / ***h1***) + 1;  
  
 **private final static double *h2*** = 0.025;  
 **private final static int *SIZE2*** = (**int**) ((***B*** - ***A***) / ***h2***) + 1;  
  
 **public static void** main(String[] args) {  
 **double**[][] a = *matrixGenerator*(***h1***, ***SIZE1***);  
  
 **double**[][] b = *matrixGenerator*(***h2***, ***SIZE2***);  
  
 **double**[] answer1 = *doProgonka*(a, ***SIZE1***);  
  
 **double**[] answer2 = *doProgonka*(b, ***SIZE2***);  
  
 **for** (**int** i = 0; i < ***SIZE1***; i++) {  
 System.***out***.println(**"Y"** + i + **" = "** + String.*format*(**"%.5f"**, answer1[i]));  
 System.***out***.println();  
 }  
  
*// System.out.println();  
//  
// for (int i = 0; i < SIZE2; i++) {  
// System.out.println(/\*"Y" + i + " = " + \*/String.format("%.5f", answer2[i]));  
// }  
  
 rungeError*(answer2, ***SIZE2***, answer1, ***SIZE1***);  
 }  
  
 **public static double**[] doProgonka(**double**[][] matrix, **int** size) {  
 */\* Прямая прогонка \*/* **double**[] y = **new double**[size];  
 **double**[] alpha = **new double**[size];  
 **double**[] betta = **new double**[size];  
  
 y[0] = matrix[0][0];  
 alpha[1] = -matrix[0][1] / y[0];  
 betta[1] = matrix[0][size] / y[0];  
  
 **for** (**int** i = 1; i < size - 1; i++) {  
 y[i] = matrix[i][i] + matrix[i][i - 1] \* alpha[i];  
 alpha[i + 1] = -matrix[i][i + 1] / y[i];  
 betta[i + 1] = (matrix[i][size] - matrix[i][i - 1] \* betta[i]) / y[i];  
 }  
  
 */\*Обратный ход\*/* **double**[] x = **new double**[size];  
 x[size - 1] = (matrix[size - 1][size] - matrix[size - 1][size - 2] \* betta[size - 1])  
 / (matrix[size - 1][size - 1] + matrix[size - 1][size - 2] \* alpha[size - 1]);  
 **for** (**int** i = size - 2; i >= 0; i--) {  
 x[i] = alpha[i + 1] \* x[i + 1] + betta[i + 1];  
 }  
  
 **return** x;  
 }  
  
 **public static void** rungeError(**double**[] answer1, **int** size1, **double**[] answer2, **int** size2) {  
 **double** max = 0;  
 **if** (size2 < size1) {  
 **for** (**int** i = 0; i < size2; i++) {  
 max = Math.*max*(max, Math.*abs*(answer2[i] - answer1[2 \* i]) / 3);  
 }  
 }  
  
 **if** (size2 > size1) {  
 **for** (**int** i = 0; i < size1; i++) {  
 max = Math.*max*(max, Math.*abs*(answer2[i \* 2] - answer1[i]) / 3);  
 }  
 }  
 System.***out***.println(**"RungeError max(|answer1(xi)-answer2(xi)|/3) = "** + max);  
 }  
  
 **public static double**[][] matrixGenerator(Double h, **int** size) {  
  
 **double**[][] matrix = **new double**[size][size + 1];  
  
 matrix[0][0] = -***SIGMA0*** - 1 / h;  
 matrix[0][1] = 1 / h;  
  
 matrix[0][size] = ***MU0***;  
  
 **double** xi = -1D + h;  
 **int** i;  
 **for** (i = 1; i < size - 1; i++) {  
 matrix[i][i - 1] = 1 / (h \* h) + xi / (2 \* h);  
 matrix[i][i] = -2 / (h \* h) - Math.*exp*(xi / 3);  
 matrix[i][i + 1] = 1 / (h \* h) - xi / (2 \* h);  
  
 matrix[i][size] = -xi / 2 - 1;  
  
 xi += h;  
 }  
  
 matrix[size - 1][size - 2] = -1 / h;  
 matrix[size - 1][size - 1] = -***SIGMA1*** + 1 / h;  
  
 matrix[size - 1][size] = ***MU1***;  
  
 **return** matrix;  
 }  
}

**Вывод:**

**В ходе лабораторной работы были успешно реализованы такие методы численного решения ДУ второго порядка как:**

* **Метод сеток решения краевой задачи для ОДУ**

**Была оценена погрешность по правилу Рунге:**

* **Погрешность, посчитанная методом Рунге, примерно равняется ~ 10^-3 и совпадает с точностью метода O(h^2), которая в нашем случае: h^2 = 0.0025 ~ 10^-3;**