Федеральное государственное автономное образовательное учреждение высшего образования

Университет ИТМО

**Лабораторная работа №1**

**Курса “Вычислительная математика”**

Вариант 9

**Выполнил:**Кривоносов Егор Дмитриевич  
**Группа:** P3211  
  
**Преподаватель:**Малышева Татьяна Алексеевна

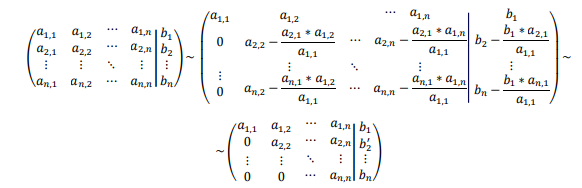
2021 г.

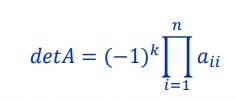
**Цель работы:**

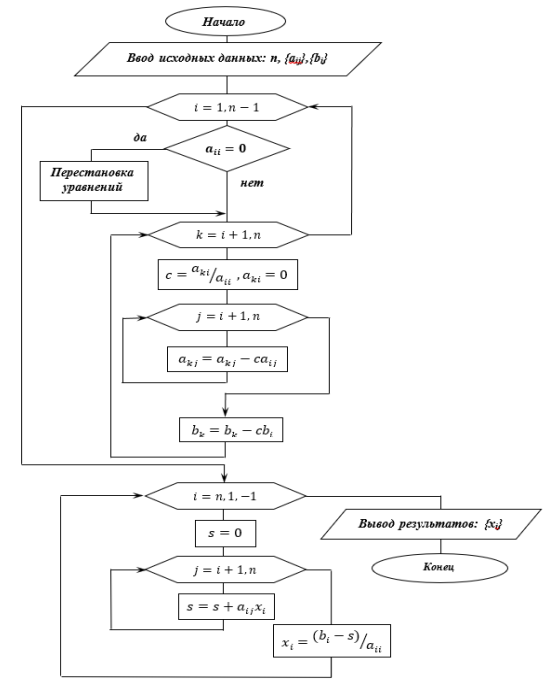
Реализовать и протестировать программу для решения системы линейных уравнений методом Гаусса, размерность до 20 включительно неизвестных.

**Описание метода, расчетные формулы:**

Суть метода заключается в преобразования расширенной СЛАУ к треугольному виду и последующему нахождению всех неизвестных. Если матрица квадратная и она имеет определитель, не равный нулю, то мы имеем единственное решение. Далее мы находим все неизвестные начиная с последней строки. Каждая неизвестная выражается через предыдущие, а последняя известна сразу.

  
(к треугольному виду)

  
(решения)  
  
(определитель)

**Блок-схема для метода Гаусса:  
**

**Код реализации решения на Python 3.\*:  
# Если в процессе вычисления a11, a22, a33, ... = 0 - нужно переставить соответственно коэф.**def \_\_check\_diagonal(self, i):  
 j = i  
 while j < self.n:  
 if (self.system[j][i] != 0) and (self.system[i][j] != 0):  
 swap = self.system[j]  
 self.system[j] = self.system[i]  
 self.system[i] = swap  
 self.swap += 1  
 return  
 j += 1  
 print(color.BOLD + color.RED, 'Нет решений!', color.END)  
 return ArithmeticError

**# Вычисление треугольной матрицы прямым способом по формулам**  
 def \_\_make\_triangle(self):  
 try:  
 i = 0  
 while i < self.n:  
 if self.system[i][i] == 0:  
 self.\_\_check\_diagonal(i)  
 m = i  
 while m < self.n - 1:  
 a = -(self.system[m + 1][i] / self.system[i][i])  
 j = i  
 while j < self.n:  
 self.system[m + 1][j] += a \* self.system[i][j]  
 j += 1  
 self.system[m + 1][-1] += a \* self.system[i][-1]  
 m += 1  
 k = 0  
 line\_sum = 0  
 while k < self.n:  
 line\_sum += self.system[i][k]  
 k += 1  
 if line\_sum == 0:  
 print(color.BOLD + color.RED, 'Данная система не совместима, решений нет!', color.END)  
 return ArithmeticError  
 i += 1  
 except ValueError:  
 print(color.BOLD + color.RED, 'Некорректная работа с данными!', color.END)  
 return

**# Подсчет и вывод определителя на экран** def \_\_get\_determinate(self): i = 0 self.det = 1 while i < self.n: self.det \*= self.system[i][i] i += 1 if self.swap % 2 == 1: self.det \*= -1 print('\n', color.UNDERLINE + color.YELLOW, 'Определитель', color.END, ' = ', self.det, '\n') if self.det == 0: print(color.BOLD + color.RED, 'Система является вырожденной, нет решения.', color.END) return ArithmeticError  
  
**# Подсчет неизвестных x1, x2, x3, ..., xn**  
 def \_\_calc\_vector\_x(self):  
 i = self.n – 2  
 self.x.append(self.system[self.n - 1][-1] / self.system[self.n - 1][self.n - 1])  
 while i > -1:  
 k = self.n – 1  
 val = self.system[i][-1]  
 while k > i:  
 val -= self.x[self.n - 1 - k] \* self.system[i][k]  
 k -= 1  
 self.x.append(val / self.system[i][i])  
 i -= 1  
  
**# Подсчет коэф. невязки r1, r2, r3, ..., rn и вывод на экран**def \_\_print\_vector\_residuals(self):  
 i = 0  
 print(color.UNDERLINE + color.YELLOW, 'Невязки (величина ошибки):', color.END)  
 while i < self.n:  
 res = 0  
 j = 0  
 while j < self.n:  
 res += self.system[i][j] \* self.x[j]  
 j += 1  
 res -= self.system[i][-1]  
 i += 1  
 print('\t', 'Невязка для', i, 'строки:', abs(res))  
 print('')

**Примеры и результаты работы программы:  
Решатель Системы Уравнений методом Гаусса!**

**Доступные функции программы:**  
 1: Считывание линейной системы из файла.  
 2: Ввод линейной системы.  
 3: Выход.   
Введите число функции = 2  
  
Выбран способ ввода вручную.   
Введите кол-во уравнений не более 20: 2  
  
Введите коэффициенты уравнения в формает:  
 ai1 ai2 ... aij | bi   
1: 1 5 | 7  
2: 3 -2 | 4

Наша система:

1.0 x[0] 5.0 x[1] | 7.0  
3.0 x[0] -2.0 x[1] | 4.0

Треугольная матрица:

1.0 x[0] 5.0 x[1] | 7.0  
0.0 x[0] -17.0 x[1] | -17.0

Определитель = -17.0

Решение системы:   
 x[0]: 2.0  
 x[1]: 1.0

Невязки (величина ошибки):   
 Невязка для 1 строки: 0.0  
 Невязка для 2 строки: 0.0

**Доступные функции программы:**  
 1: Считывание линейной системы из файла.  
 2: Ввод линейной системы.  
 3: Выход.   
Введите число функции = 1

Выбран способ считывание с файла.   
Файл должен содержать линейную систему вида (Размерность не более n = 20):   
 a11 a12 ... a1n | b1   
 a21 a22 ... a2n | b2   
 ... ... ... ... | ..   
 an1 an2 ... ann | bn

Введите путь к фалу: test5x5.txt

Наша система:   
1.0 x[0] -1.0 x[1] 3.0 x[2] 1.0 x[3] 3.0 x[4] | 5.0  
4.0 x[0] -1.0 x[1] 5.0 x[2] 4.0 x[3] -2.0 x[4] | 4.0  
2.0 x[0] -2.0 x[1] 4.0 x[2] 1.0 x[3] 3.0 x[4] | 6.0  
1.0 x[0] -4.0 x[1] 5.0 x[2] -1.0 x[3] -4.0 x[4] | 3.0  
2.0 x[0] -4.0 x[1] 5.0 x[2] -1.0 x[3] 5.0 x[4] | 3.0

Треугольная матрица:   
1.0 x[0] -1.0 x[1] 3.0 x[2] 1.0 x[3] 3.0 x[4] | 5.0  
0.0 x[0] 3.0 x[1] -7.0 x[2] 0.0 x[3] -14.0 x[4] | -16.0  
0.0 x[0] 0.0 x[1] -2.0 x[2] -1.0 x[3] -3.0 x[4] | -4.0  
0.0 x[0] 0.0 x[1] 0.0 x[2] 0.5 x[3] -13.5 x[4] | -8.0  
0.0 x[0] 0.0 x[1] 0.0 x[2] 0.0 x[3] -6.33333333333334 x[4] | -9.000000000000004

Определитель = 19.00000000000002

Решение системы:   
 x[0]: -12.789473684210517  
 x[1]: -25.10526315789471  
 x[2]: -11.315789473684196  
 x[3]: 22.368421052631554  
 x[4]: 1.4210526315789465

Невязки (величина ошибки):   
 Невязка для 1 строки: 0.0  
 Невязка для 2 строки: 3.552713678800501e-15  
 Невязка для 3 строки: 0.0  
 Невязка для 4 строки: 0.0  
 Невязка для 5 строки: 0.0

**Доступные функции программы:** 1: Считывание линейной системы из файла. 2: Ввод линейной системы. 3: Выход. Введите число функции = 3 Удачи!

**Вывод:**

Метод Гаусса подходит для простого вычисления СЛАУ, так как он более универсален и прост в реализации, а также работает за конечное число арифметических операций. Но и у него есть недостатки так как весь массив приходится хранить в оперативной памяти компьютера. Происходит накопление погрешности в процессе решения, поскольку на любом этапе используют результаты предыдущих операций (решить проблему можно с помощью округлений, но и у этого есть свои недостатки).