Academicia Globe:
Inderscience Research

ISSN: 2776-1010 Volume 4, Issue 12, December 2023

SOLVING A SYSTEM OF EQUATIONS USING THE GAUSS METHOD IN THE PYTHON PROGRAMMING LANGUAGE

Shamsiddinova Maftunabonu Ulug'bek qizi
Student of Bukhara State University
shamsidinovamaftuna4@gmail.com

Abstract:
This article discusses methods for solving systems of linear algebraic equations. The differences between the methods and their solutions, the Gaussian method of solving systems, the principles of implementing the Gaussian method for solving a system of linear equations in the Python programming language are presented.

Keywords: Python programming language, linear algebra, Gaussian method, Math library, augmented matrix, system of equations, NumPy, Matlab library.

Annotatsiya:
Ushbu maqolada chiziqli algebraik tenglamalar tizimlarini yechish usullarini ko'rib chiqiladi. Usullar orasidagi farqlar va ularning yechimlarini ,sistemalarini yechishning Gauss usuli ,Python dasturlash tilida chiziqli tenglamalar tizimini yechish uchun Gauss usulini amalga oshirish tamoyillari keltirilgan.

Kalit so'zlar: Python dasturlash tili, chiziqli algebra, Gauss usuli, Math kutubxonasi, kengaytirilgan matritsa, tenglamalar tizimi, NumPy, Matlab kutubxonasi.

Аннотация:
В статье рассмотрены методы решения систем линейных алгебраических уравнений. Представлены различия между методами и их решениями, гауссовым методом решения систем, принципами реализации гауссовского метода решения системы линейных уравнений на языке программирования Python.

Ключевые слова: язык программирования Python, линейная алгебра, метод Гаусса, математическая библиотека, расширенная матрица, система уравнений.

Introduction

In mathematics, there are many types of methods for solving systems of equations. Linear algebra is the main branch of mathematics. One of the most important methods for solving a system of equations in linear algebra is the Gaussian method, also known as Gaussian elimination. This method provides a systematic way to solve systems of linear equations. The Gaussian method is a step-by-step process used to convert a system of linear equations into an equivalent system that is easier to solve. The main goal is to reduce the system to its simplest form. That is, the Gaussian method is based on successive elimination of unknowns from the equations, leaving only one unknown in the last equation. Finding the unknown from it and putting it in the penultimate equation, the second unknown is found, and so on, and this process is continued, putting the values of the unknowns found in the first equation, and the first unknown is determined from it. Augmented Matrix A system of linear equations is usually expressed using an augmented matrix. This matrix consists of the coefficients of the variables and the constants on the right-hand side of the equations.

Methods and material

Suppose we are given this system of linear equations:

$$\begin{cases} a_{1,1}x_1 + a_{1,2}x_2 + \ldots + a_{1,n}x_n = b_1, \\ a_{2,1}x_1 + a_{2,2}x_2 + \ldots + a_{2,n}x_n = b_2, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots, \\ a_{n,1}x_1 + a_{n,2}x_2 + \ldots + a_{n,n}x_n = b_n. \end{cases}$$

The matrix form of the system is as follows:

A•X=B

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}, x = \begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_n \end{bmatrix}, b = \begin{bmatrix} b_1 \\ b_2 \\ \cdots \\ b_n \end{bmatrix}$$

Gaussian elimination is based on the idea of sequentially eliminating variables one at a time until only one equation remains with one variable on the left-hand side. This equation is then solved with respect to one variable. Thus, the system of equations is made triangular. For this, a non-zero (or often maximum) element is selected from among the elements of the first column of the matrix and moved to the highest position by reordering the rows. All equations are then normalized by dividing it by the factor a_{n1} . The first row is subtracted from the remaining rows and a new system of equations is obtained, in which the corresponding coefficients are replaced:

$$\begin{cases} x_1 + a'_{12} \cdot x_2 + \cdots + a'_{1n} \cdot x_n = \boldsymbol{b}'_1 \\ 0 + a'_{22} \cdot x_2 + \cdots + a'_{2n} \cdot x_n = \boldsymbol{b}'_2 \\ \cdots \\ 0 + a'_{n2} \cdot x_2 + \cdots + a'_{nn} \cdot x_n = \boldsymbol{b}'_n \end{cases}$$

This process is continued until there is only one unknown equation for all subsequent equations:

$$\begin{cases} x_1 + a'_{12} \cdot x_2 + a'_{13} \cdot x_3 + \cdots + a'_{1n} \cdot x_n = \boldsymbol{b}'_1 \\ 0 + x_2 + a''_{23} \cdot x_3 + \cdots + a''_{2n} \cdot x_n = \boldsymbol{b}''_2 \\ 0 + 0 + x_3 + \cdots + a'''_{3n} \cdot x_n = \boldsymbol{b}'''_3 \\ \cdots \\ 0 + 0 + 0 + \cdots + x_n = \boldsymbol{b}_n^{n\prime} \end{cases}$$

Once x_n is found, the solutions of the system are found inversely, repeating for all remaining solutions:

$$x_{n-1} = \boldsymbol{b}_{n-1}^{(n-1)'} - a_{(n-1)n}^{(n-1)'} \cdot x_n$$

$$x_{n-2} = \left(\boldsymbol{b}_{n-2}^{(n-2)'} - a_{(n-2)n}^{(n-2)'} \cdot x_n\right) - a_{(n-2)(n-1)}^{(n-2)'} \cdot x_{n-1}$$

etc. the solutions of the system are found.

For example, let us be given a system of equations of the following form:

$$\begin{cases} x_1 - x_2 + 3x_3 = 2, \\ x_1 + 2x_2 + 5x_3 = -9, \\ 2x_1 - 5x_2 - 4x_3 = 23. \end{cases}$$

First, we construct its expanded matrix and start counting by zeroing until there is one unknown left:

$$\left(\begin{array}{ccc|c} 1 & -1 & 3 & 2 \\ 1 & 2 & 5 & -9 \\ 2 & -5 & -4 & 23 \end{array}\right) \Rightarrow \left(\begin{array}{ccc|c} 1 & -1 & 3 & 2 \\ 0 & 3 & 2 & -11 \\ 0 & -3 & -10 & 19 \end{array}\right) \Rightarrow \left(\begin{array}{ccc|c} 1 & -1 & 3 & 2 \\ 0 & 3 & 2 & -11 \\ 0 & 0 & -8 & 8 \end{array}\right)$$

we make the matrix of the system triangular and find the solutions:

$$\begin{cases} x_1 - x_2 + 3x_3 = 2, \\ 3x_2 + 2x_3 = -11, \\ -8x_3 = 8. \end{cases}$$

In this system, moving from the bottom up, we find the only solution:

$$x_3 = -1,\ x_2 = -3,\ x_1 = 2$$

Results and Discussion

Solving a system of equations is a common task in mathematics and in various fields of science and technology. Programming allows you to automate calculations that make it easier to work with large and complex data sets. The method of solving the system of equations using the Gaussian method was shown above. The Python programming language provides several methods for solving these systems with its powerful libraries and numerical capabilities. Solving linear equations in a programming language has several advantages, including that using code to solve the equations allows you to automate the process, which speeds up calculations and simplifies repetitive tasks. Software solving of linear equations gives very accurate results because computer programs can perform long and complex calculations without errors. It also works with large amounts of data. The programs can process a large number of equations simultaneously, which makes it easier to work with complex models and data analysis. In addition, many programming languages have pre-written libraries and functions for solving linear equations, which are convenient for programmers. There are many ready-made libraries for calculating the system of equations using the Gaussian method in a programming language. The most popular and common libraries:

- **NumPy -** NumPy is a library for the Python programming language that supports large multidimensional arrays and matrices with a large number of functions for operations on them, including other calculations.
- **Math -** Math is also a Python programming language library that provides a wide range of numerical methods, including the ability to solve systems of linear equations using Gaussian.
- **Matlab -** Matlab is a high-level language and environment for technical computing that provides many tools and functions for performing various types of mathematical calculations.

When creating a linear equation program in the Python programming language, we need to pay attention to the following important points. It is convenient to use the NumPy library, which provides many useful functions for working with linear algebra, for use in import modules, for working with equations and matrices. Input Validation It is important to check the values entered to ensure that they are within the range of data types and valid values. When calculating equations, the program must correctly calculate given linear equations using mathematical operations such as addition, subtraction, multiplication, and division. It is important to ensure handling of possible errors such as division by zero or inconsistency of the system of equations. These aspects help us to create robust and efficient linear equations program in Python programming language. The first step to solving systems of equations in Python is to understand the structure of the equation. First of all, we import the necessary library of the Python programming language. In order to determine how many times the zeroing process is repeated, we need to determine how many unknowns the system of equations consists of. We open an empty list and insert the expanded matrix into our empty list. We use the Gaussian method to solve the system of linear equations. It iterates through the rows and columns of the matrix, performs calculations to find solutions to the unknowns, and stores them in a non-empty list. That's it, the solution of the equations will be formed in our list, and we will solve the system of equations by printing these solutions on the screen.

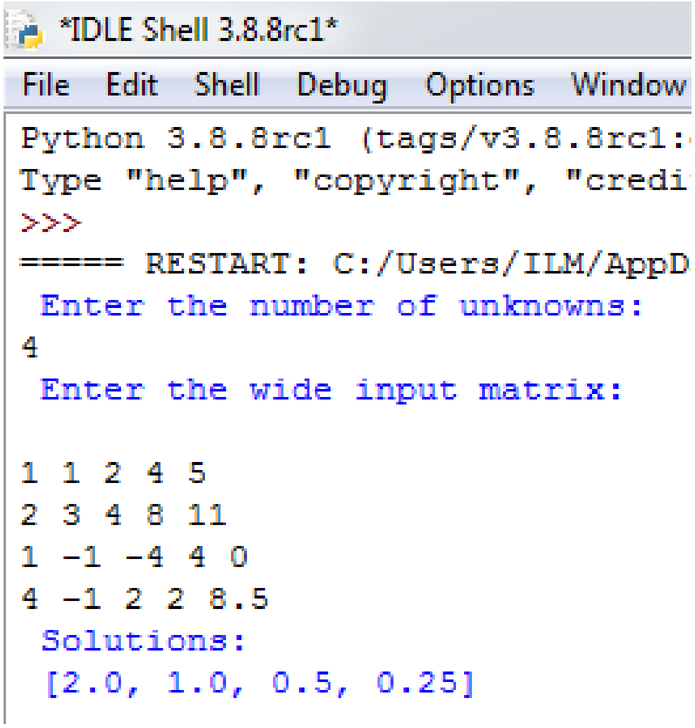
If we want to use the **Math** library, the program code will be as follows:

```
import math
n=int(input(' Enter the number of unknowns:\n'))
a=[]
j=0
print(' Enter the wide input matrix:\n')
```

```
for i in range(n):
   a.append([float(j) for j in input().split()])
k=1
x=[0]*n
while k<n:
   j=k
   while j<n:
      m=a[j][k-1]/a[k-1][k-1]
      i=0
      while i<n+1:
         a[j][i]=a[j][i]-m*a[k-1][i]
         i+=1
      j+=1
   k+=1
i=n-1
if a[i][i]==0:
   print(' Roots are not available')
else:
   while i>=0:
      x[i]=a[i][n]/a[i][i]
      c=n-1
      while c>i:
         x[i]=x[i]-(a[i][c]*x[c]/a[i][i])
         c-=1
      i-=1
   print(' Solutions:\n',x)
input()
```

Program result:

$$\begin{cases} x + y + 2z + 4t = 5 \\ 2x + 3y + 4z + 8t = 11 \\ x - y - 4z + 4t = 0 \\ 4x - y + 2z + 2t = 8{,}5 \end{cases}$$



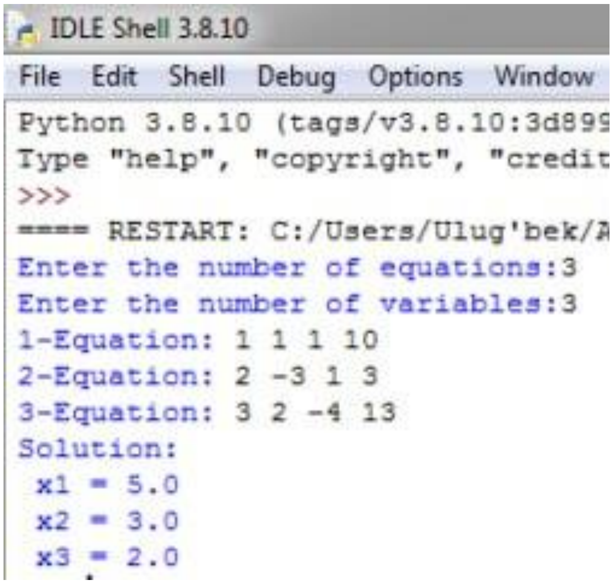
```
Python 3.8.8rc1 (tags/v3.8.8rc1:
Type "help", "copyright", "credi
>>>
===== RESTART: C:/Users/ILM/AppD
 Enter the number of unknowns:
4
 Enter the wide input matrix:

1 1 2 4 5
2 3 4 8 11
1 -1 -4 4 0
4 -1 2 2 8.5
 Solutions:
 [2.0, 1.0, 0.5, 0.25]
```

Above, we showed how to solve systems of equations using the Math library. Let's run our program using the **NumPy** library. Program code:

```
import numpy as np
a= int(input("Enter the number of equations:"))
b= int(input("Enter the number of variables:"))
m= np.zeros((a, b+1))
for i in range(a):
   c= input(f"{i+1}-Equation: ").split()
   for j in range(b+1):
      m[i][j] = float(c[j])
for i in range(a):
   if m[i][i] == 0.0:
      sys.exit("Divide by zero detected!")
   for j in range(i+1, a):
      r= m[j][i] / m[i][i]
      for k in range(b+1):
         m[j][k]=m[j][k]-r* m[i][k]
s = np.zeros(b)
s[b-1] = m[a-1][b] / m[a-1][b-1]
for i in range(a-2, -1, -1):
   s[i] = m[i][b]
   for j in range(i+1, b):
        s[i] = s[i] - m[i][j] * s[j]
    s[i] = s[i] / m[i][i]
print("Solution:")
for i in range(b):
    print(f" x{i+1} = {s[i]}")
```

Program result:

$$\begin{cases} x + y + z = 10 \\ 2x - 3y + z = 3 \\ 3x + 2y - 4z = 13 \end{cases}$$



```
Python 3.8.10 (tags/v3.8.10:3d899
Type "help", "copyright", "credit
>>>
==== RESTART: C:/Users/Ulug'bek/A
Enter the number of equations:3
Enter the number of variables:3
1-Equation: 1 1 1 10
2-Equation: 2 -3 1 3
3-Equation: 3 2 -4 13
Solution:
 x1 = 5.0
 x2 = 3.0
 x3 = 2.0
```

The software provided the solutions to our equation accurately and quickly.

Methodology

Using Python to perform numerical methods has many advantages, including understandable and easy-to-develop code, a rich feature library, an interactive development environment, and support from a large developer community. This makes Python one of the most popular programming languages for digital computing. Many scientists have studied the application of the Python programming language in performing numerical methods. Including S.K. In his manual, Buychanov detailed the implementation of numerical methods in the Python programming language [3]. The application of Python in solving mathematical problems was developed by N.I. Kamalova showed it. U - Through programming, mathematicians can solve various mathematical analysis problems, such as finding derivatives and integrals of functions, numerical integration, solving differential equations, etc. In addition, he expressed the opinion that programming allows conducting a series of experiments and research to test and develop new mathematical theories and models [2].

In general, convenient integration with other languages such as C++ and Fortran allows the use of optimized numerical libraries and acceleration of calculations.

Conclusion

The Gaussian method is one of the efficient methods for solving systems of linear equations, and the flexibility and extensive libraries of the Python programming language make it a suitable language for implementing this method. By following the above steps, we can use the power of the Python programming language to solve complex systems of equations and obtain their solutions. For those working on mathematical problems, scientific simulations, or engineering applications, Gaussian computation in the Python programming language can be a valuable tool. It allows to find the solutions of the system of complex equations in a short time.

References

1. Ernazar Xolmurodov, Abdulaziz Yusupov ,,Oliy Matematika “ o’quv qo’llanma.
2. Камалова Н. И., Бадриддинова Г. М. Симбиоз цифрового и аналитического мышления: взаимная интеграция программирования и математического анализа в современном мире //Proceedings of International Conference on Scientific Research in Natural and Social Sciences. – 2023. – Т. 2. – №. 12. – С. 167-171.
3. Буйначев С. К., Боклаг Н. Ю. Основы программирования на языке Python: учебное пособие. – 2014.