



Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ _____ «Информатика и системы управления»

КАФЕДРА _____ «Теоретическая информатика и компьютерные технологии»

Лабораторная работа № 3
по курсу «Численные методы линейной алгебры»
«Оценки источников погрешностей решения СЛАУ методом Гаусса»

Студент группы ИУ9-72Б Виленский С. Д.

Преподаватель Посевин Д. П.

Москва 2024

1 Задание

Реализовать формулы оценок ошибок округления и исходных данных.

2 Результаты

Исходный код программы представлен в листингах 1- ??.

Листинг 1 — Реализация алгоритмов оценки источников погрешностей

```
1 using LinearAlgebra
2 using Random
3
4 ## Generating input data
5
6 function generate_matrix(n::Int64)::Matrix{Float64}
7     return rand(-100:0.01:100, n, n)
8 end
9 function generate_matrix_delta(n::Int64)::Matrix{Float64}
10    return rand(-10:0.01:10, n, n)
11 end
12
13 function generate_vector(n::Int64)::Vector{Float64}
14    return rand(-100:0.01:100, n)
15 end
16 function generate_vector_delta(n::Int64)::Vector{Float64}
17    return rand(-10:0.01:10, n)
18 end
19
20 ## Norms
21
22 function euclidean_norm(A)::Float64
23    return sqrt(sum(abs2, A))
24 end
25
26 function uniform_norm(x::Vector{Float64})::Float64
27    return maximum(abs.(x))
28 end
29
30 function uniform_norm(x::Matrix{Float64})::Float64
31    return maximum(sum(abs, x, dims=2))
32 end
33
34 ## Condition number of a matrix
35
36 function get_condition_number(A::Matrix{Float64}, _norm::Function)::
    Float64
37    return _norm(inv(A)) * _norm(A)
38 end
39
40 ## Growth rate of matrix elements
41
42 function gauss_growth_factor(A::Matrix{Float64})::Float64
43    n = size(A, 1)
```

Листинг 2 — Реализация алгоритмов оценки источников погрешностей

```

1  max_initial = maximum(abs.(A))
2  max_during = max_initial
3
4  A_work = copy(A)
5
6  for k in 1:n-1
7      for i in k+1:n
8          if A_work[k, k] != 0
9              factor = A_work[i, k] / A_work[k, k]
10             for j in k:n
11                 A_work[i, j] -= factor * A_work[k, j]
12             end
13         end
14     end
15     max_during = max(max_during, maximum(abs.(A_work)))
16 end
17
18 return max_during / max_initial
19 end
20
21 ## Error estimates
22
23 function error_rounding(
24     A::Matrix{Float64},
25     p::Int64,
26     t::Int64,
27     _norm::Function
28 )::Float64
29     nu_A = get_condition_number(A, _norm)
30     n = size(A, 1)
31     g_A = gauss_growth_factor(A)
32     return nu_A * n * g_A / p ^ t
33 end
34
35 function error_input_data(
36     A::Matrix{Float64}, delta_A::Matrix{Float64},
37     f::Vector{Float64}, delta_f::Vector{Float64},
38     _norm::Function
39 )::Float64
40     nu_A = get_condition_number(A, _norm)
41     error_A = _norm(delta_A) / _norm(A)
42     error_f = _norm(delta_f) / _norm(f)
43     return nu_A * (error_A + error_f)
44 end
45
46 ## Solution error
47
48 function error_result(
49     A::Matrix{Float64}, delta_A::Matrix{Float64},
50     f::Vector{Float64}, delta_f::Vector{Float64},
51     _norm::Function
52 )::Float64
53     x = A \ f
54     delta_x = (A + delta_A) \ (f + delta_f) - x
55     return _norm(delta_x) / _norm(x)
56 end

```

Листинг 3 — Реализация алгоритмов оценки источников погрешностей

```
1 ## Testing
2
3 A = [
4     100. 99.;
5     99. 98.
6 ]
7 delta_A = [
8     0. 0.;
9     0. 0.
10 ]
11 f = [
12     199.,
13     197.
14 ]
15 delta_f = [
16     -.01,
17     .01
18 ]
19
20 print("Relative error of input data: ", error_input_data(A, delta_A, f,
21     delta_f, uniform_norm))
22
23 A = generate_matrix(5)
24 delta_A = generate_matrix_delta(5)
25 f = generate_vector(5)
26 delta_f = generate_vector_delta(5)
27
28 _norm = euclidean_norm
29
30 print("Relative error of rounding: ", error_rounding(A, 2, 11, _norm))
31 print("\nRelative error of input data: ", error_input_data(A, delta_A, f,
32     delta_f, _norm))
33 print("\nRelative error of result: ", error_result(A, delta_A, f,
34     delta_f, _norm))
```

Результат работы программы представлен в листинге 4.

Листинг 4 — Результат работы программы

```
1 Relative error of input data: 1.9899999999989821
2 Relative error of result: 1.9899999999940312
3
4 Relative error of rounding: 0.11516827877965007
5 Relative error of input data: 1.32573555646218
6 Relative error of result: 0.1490879688109449
```

3 Выводы

Проанализировав графики зависимостей ошибок разных методов от диагонального доминирования матриц можно сделать вывод о том, что самым оптимальным является модификация метода Гаусса перестановкой по строкам и столбцам, модификации метода Гаусса перестановкой по строкам или по столбцам относительно имеет близкую погрешность и классический метод Гаусса среди прочих имеет наибольшую относительную ошибку.