



Министерство науки и высшего образования Российской Федерации  
Федеральное государственное автономное 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 № 4**  
**по курсу «Численные методы линейной алгебры»**  
**«Метод Крылова поиска собственных значений и векторов»**

Студент группы ИУ9-72Б Шемякин В.А.

Преподаватель Посевин Д. П.

*Москва 2025*

# 1 Задание

Реализовать метод Крылова для нахождения собственных значений и собственных векторов матрицы, а также сравнить полученные результаты с методом Данилевского.

## 2 Результаты

Исходный код программы представлен в листинге 1.

```
1 using LinearAlgebra
2 using Printf
3
4 function gauss_solve(M::Matrix{Float64}, b::Vector{Float64})
5     n = size(M, 1)
6     A = hcat(M, b)
7
8     for k in 1:n-1
9         pivot = argmax(abs.(A[k:end, k])) + k - 1
10        if pivot != k
11            A[[k, pivot], :] .= A[[pivot, k], :]
12        end
13
14        for i in k+1:n
15            factor = A[i, k] / A[k, k]
16            A[i, k:end] .-= factor .* A[k, k:end]
17        end
18    end
19
20    x = zeros(Float64, n)
21    for i in n:-1:1
22        x[i] = (A[i, end] - sum(A[i, i+1:n] .* x[i+1:n])) / A[i, i]
23    end
24
25    return x
26 end
27
28
```

```

29 function krylov_eigenvalues(A::Matrix{Float64})
30     n = size(A, 1)
31     y0 = zeros(n)
32     y0[1] = 1.0
33
34     Y = zeros(n, n + 1)
35     Y[:, 1] = y0
36     for k in 2:n+1
37         Y[:, k] = A * Y[:, k-1]
38     end
39
40     M = zeros(n, n)
41     for i in 1:n
42         M[:, i] = Y[:, n - i + 1]
43     end
44     b = Y[:, n + 1]
45
46     p = gauss_solve(M, b)
47
48     F = zeros(n, n)
49     F[1, :] = p
50     for i in 2:n
51         F[i, i-1] = 1.0
52     end
53     return eigvals(F), Y, p
54 end
55
56
57 function eigenvectors_krylov(A, eigenvalues, Y, p)
58     n = length(eigenvalues)
59     eigenvectors = Vector{Vector{ComplexF64}}{n}()
60
61     for i in 1:n
62         q = zeros(ComplexF64, n)
63         q[1] = 1.0
64         for j in 2:n
65             q[j] = - (A[j, j-1] + eigenvalues[i]) * q[j-1]
66             if j <= n
67                 q[j] -= p[j-1]

```

```

68         end
69     end
70
71     x = zeros(ComplexF64, n)
72     for j in 1:n
73         x .+= q[j] .* Y[:, n - j + 1]
74     end
75
76     if norm(x) > 0
77         x ./= norm(x)
78     end
79
80     push!(eigenvectors, x)
81 end
82
83 return eigenvectors
84 end
85
86
87 function method_krylov(A::Matrix{Float64})
88     eigenvalues, Y, p = krylov_eigenvalues(A)
89     eigenvectors = eigenvectors_krylov(A, eigenvalues, Y, p)
90     return eigenvalues, hcat(eigenvectors...)
91 end
92
93
94 real_tol(x; tol=1e-10) = x isa Complex ? (abs(imag(x)) < tol ? real(x) : x) : x
95
96 function make_real_eigs(s, X; tol=1e-10)
97     r = [real_tol(; tol=tol) for _ in s]
98     Y = similar(X)
99     for j in 1:size(X,2)
100         v = X[:,j]
101         if maximum(abs.(imag.(ComplexF64.(v)))) > tol
102             w = real.(v)
103             k = argmax(abs.(w))
104             w = w[k] < 0 ? -w : w
105             Y[:,j] = w
106         else

```

```

107         Y[:,j] = v
108     end
109 end
110 r, Y
111 end
112
113 function print_eigs_col(s, X; tol=1e-10, digits=6)
114     println(" :")
115     for (j,) in enumerate(s)
116         @printf("%-2d = %.*f\n", j, digits, real_tol(; tol=tol))
117     end
118     println("\n :")
119     for j in 1:size(X,2)
120         println("x_j =")
121         for i in 1:size(X,1)
122             x = real_tol(X[i,j]; tol=tol)
123             if x isa Complex
124                 @printf(" %.*f %+. *fi\n", digits, real(x), digits, imag(x))
125             else
126                 @printf(" %.*f\n", digits, x)
127             end
128         end
129         println()
130     end
131 end
132
133
134 A = [2.2  1.0  0.5  2.0;
135      1.0  1.3  2.0  1.0;
136      0.5  2.0  0.5  1.6;
137      2.0  1.0  1.6  2.0]
138
139 s, X = method_krylov(A)
140 s, X = make_real_eigs(s, X; tol=1e-10)
141 print_eigs_col(s, X; tol=1e-10, digits=6)

```

Результат запуска представлен на рисунке 1.

```
Собственные значения матрицы:
λ1 = -1.420087
λ2 = 0.222636
λ3 = 1.545418
λ4 = 5.652032

Собственные векторы матрицы:
x1 =
  0.222043
 -0.515910
  0.757274
 -0.333271

x2 =
 -0.521921
 -0.454869
  0.153447
  0.705086

x3 =
  0.628930
 -0.572574
 -0.485654
  0.201858

x4 =
  0.531736
  0.446194
  0.408816
  0.592484
```

Рис. 1 — Результат

### **3 Вывод**

В ходе лабораторной работы мне удалось реализовать метод Крылова для нахождения собственных значений и собственных векторов матрицы, а также сравнить полученные результаты с методом Данилевского.