5.5.4

RAVULA SHASHANK REDDY - EE25BTECH11047

October 2, 2025

Question

Find the value of k for which the matrix is singular.

$$\begin{pmatrix} k & 8 \\ 1 & 2k \end{pmatrix}$$

Solution

Singular matrix:

$$\det\begin{pmatrix} k & 8 \\ 1 & 2k \end{pmatrix} = 0 \tag{1}$$

$$k \cdot 2k - 1 \cdot 8 = 0 \tag{2}$$

$$2k^2 - 8 = 0 \tag{3}$$

$$2k^2 = 8 \tag{4}$$

$$k^2 = 4 \tag{5}$$

$$\boxed{k = \pm 2} \tag{6}$$

C Code

```c
        #include <stdio.h>
#include <math.h>

int main() {
    double A[2][2];
    double a, b, c; // coefficients for quadratic
    double D, k1, k2;

    // Input matrix
    printf(Enter the 2x2 matrix elements:\n);
    for(int i=0; i<2; i++) {
        for(int j=0; j<2; j++) {
            scanf(%lf, &A[i][j]);
        }
    }

    // Expecting matrix of form [[k, 8], [1, 2k]]
    // General quadratic from determinant:
```

C Code

```
// det = A[0][0]*A[1][1] - A[0][1]*A[1][0]
// => k*(2k) - 8*1 = 2k^2 - 8 = 0

a = 2;
b = 0;
c = -8;

// Discriminant
D = b*b - 4*a*c;

if (D < 0) {
    printf(No real values of k\n);
}
```

C Code

```
        k1 = (-b + sqrt(D)) / (2*a);
        k2 = (-b - sqrt(D)) / (2*a);

        printf(Equation: 2k^2 - 8 = 0\n);
        printf(Values of k for which matrix is singular:\n);
        printf(k = %.2f\n, k1);
        printf(k = %.2f\n, k2);
    }

    return 0;
}
```

Python Direct

```
        import numpy as np
from libs.matrix.funcs import det # your custom determinant
    function

# Define matrix with k
def det_val(k):
    A = np.array([[k, 8],
                  [1, 2*k]], dtype=float)
    return det(A)

solutions = []
for k in range(-10, 11): # checking a range
    if abs(det_val(k)) < 1e-9:
        solutions.append(k)

print(Values of k for which matrix is singular:, solutions)
```

Python Shared

```
        import ctypes

# Load shared library
lib = ctypes.CDLL(./libmatrix.so)

# Define argument and return types
lib.find_k.argtypes = [ctypes.POINTER(ctypes.c_double), ctypes.
    POINTER(ctypes.c_double)]
lib.find_k.restype = ctypes.c_int

# Prepare variables
k1 = ctypes.c_double()
k2 = ctypes.c_double()
```

Python Shared

```
# Call function
n = lib.find_k(ctypes.byref(k1), ctypes.byref(k2))

# Print results
if n == 0:
    print(No real values of k)
elif n == 2:
    print(Values of k for which matrix is singular:)
    print(k =, k1.value)
    print(k =, k2.value)
```