Berechne Inv(A) einer symmetrischen (3x3)-Matrix mit reelen Zahlen

Du kannst dieses Code-Snippet in einem Jupyter Notebook ausführen. Importiere die benötigte NumPy-Bibliothek. Das Notebook erstellt eine symmetrische 3x3-Matrix A, berechnet Det(A) und Adj(A), woraus sich Inv(A) ergibt.

Insgesamt besteht das Programm aus 4 Schritten.

Autor: Hermann Völlinger, DHBW Stuttgart; Date: 7.10.2023

1. Schritt: Vorbereitungen und Berechnung+ Ausgabe von Det(A):

Importiere NumPy-Bibliothek. Definiere symmetrische (3x3)-Matrix. Berechne Det(A) und zeige das Ergebniss.

In [1]:
```
# Importiere die NumPy-Bibliothek
import numpy as np

# Importiere das sympy-Modul
import sympy as sp

# Import libary time to check execution date+time
import time

# Definiere die reellen Variablen als Symbole
a, b, c, d, e, f = sp.symbols('a b c d e f')

# Definiere die 3x3-Matrix A mit den Symbolen
A=sp.Matrix([[a,b,c],
             [b,d,e],
             [c,e,f]])

# Berechne die Determinante von A und klammere sie aus
det_A=sp.det(A)

print("Die Determinante von A = Det(A) ist gegeben durch:")
det_A
```

```
Die Determinante von A = Det(A) ist gegeben durch:
```

Out[1]: $adf - ae^2 - b^2f + 2bce - c^2d$

2. Schritt: Berechne Kofaktoren als einzelne Zeilen und Adj(A):

Adj(A) = symmetrisch (C_12=C_21; C_13=C_31) mit 6 Kofaktoren: C_11, -C_12, C_13, C_22, -C_23 und C_33.

In [2]:
```
# Berechne die 6 Kofaktoren
C_11 = (A[1, 1] * A[2, 2] - A[2, 1] * A[1, 2])
C_12 = -(A[1, 0] * A[2, 2] - A[2, 0] * A[1, 2])
C_13 = (A[1, 0] * A[2, 1] - A[2, 0] * A[1, 1])
C_22 = (A[0, 0] * A[2, 2] - A[2, 0] * A[0, 2])
C_23 = -(A[0, 0] * A[2, 1] - A[2, 0] * A[0, 1])
C_33 = (A[0, 0] * A[1, 1] - A[1, 0] * A[0, 1])

# Berechne die adjungierte Matrix Adj(A)
Adj_A = sp.Matrix([[C_11, -C_12, C_13],
                   [-C_12, C_22, -C_23],
                   [C_13, -C_23, C_33]])
```

3. Schritt: Ausgabe Kofaktoren als einzelne Zeilen und Adj(A):

In [3]:
```
# Ausgabe der Kofaktoren als einzelne Zeilen und adjungierten Matrix
print("Die Kofaktoren sind:")
print("C_11 =", C_11)
print("C_12 =", C_12)
print("C_13 =", C_13)
print("C_22 =", C_22)
print("C_23 =", C_23)
print("C_33 =", C_33)
print(" ")
# Darstellung der adjungierten Matrix Adj(A) in Matrixschreibweise einer 3x3-Matrix
print("Die adjungierte Matrix ist somit Adj(A):")
Adj_A
```

```
Die Kofaktoren sind:
C_11 = d*f - e**2
C_12 = -b*f + c*e
C_13 = b*e - c*d
C_22 = a*f - c**2
C_23 = -a*e + b*c
C_33 = a*d - b**2

Die adjungierte Matrix ist somit Adj(A):
```

Out[3]:
$$\begin{bmatrix} df - e^2 & bf - ce & be - cd \\ bf - ce & af - c^2 & ae - bc \\ be - cd & ae - bc & ad - b^2 \end{bmatrix}$$

4. Schritt: Berechnung und Ausgabe von Inv(A):

Inverse(A) = 1/det(A)*Adj(A), wobei det(A) im 1. und Adj(A) im 2. Schritt berechnet wurde.

In [4]:
```
# Berechne die inverse Matrix von A
A_inv = (1 / det_A) * Adj_A

# Beschreibung der Formel für Inverse(A)
print("Inv(A) = 1/det(A)*Adj(A) ist somit gegeben durch:")
A_inv
```

```
Inv(A) = 1/det(A)*Adj(A) ist somit gegeben durch:
```

Out[4]:
$$\begin{bmatrix} \frac{df-e^2}{adf-ae^2-b^2f+2bce-c^2d} & \frac{bf-ce}{adf-ae^2-b^2f+2bce-c^2d} & \frac{be-cd}{adf-ae^2-b^2f+2bce-c^2d} \\ \frac{bf-ce}{adf-ae^2-b^2f+2bce-c^2d} & \frac{af-c^2}{adf-ae^2-b^2f+2bce-c^2d} & \frac{ae-bc}{adf-ae^2-b^2f+2bce-c^2d} \\ \frac{be-cd}{adf-ae^2-b^2f+2bce-c^2d} & \frac{ae-bc}{adf-ae^2-b^2f+2bce-c^2d} & \frac{ad-b^2}{adf-ae^2-b^2f+2bce-c^2d} \end{bmatrix}$$

In [5]:
```
# print the date & time of the notebook
print('************************************************************************')
print("Actual date & time of the notebook:",time.strftime("%d.%m.%Y  %H:%M:%S"))
print('************************************************************************')
```

```
************************************************************************
Actual date & time of the notebook: 07.10.2023  10:43:04
************************************************************************
```