assignment-7

October 31, 2024

[1]:
```
import numpy as np

# Given matrix A
A = np.array([
    [4, -2, 2, 1],
    [1, 1, 0, 1],
    [-2, 1, 3, -1],
    [1, 3, -1, 2]
])
```

[3]:
```
A_inv = np.linalg.inv(A)
print("Inverse of A:\n", A_inv)
```

```
Inverse of A:
 [[ 5.00000000e+00 -3.00000000e+01  1.00000000e+00  1.30000000e+01]
 [ 4.00000000e+00 -2.50000000e+01  1.00000000e+00  1.10000000e+01]
 [-1.00000000e+00  7.00000000e+00  5.82867088e-16 -3.00000000e+00]
 [-9.00000000e+00  5.60000000e+01 -2.00000000e+00 -2.40000000e+01]]
```

[5]:
```
eigenvalues, eigenvectors = np.linalg.eig(A)
print("\nEigenvalues:\n", eigenvalues)
print("\nEigenvectors:\n", eigenvectors)
```

```

Eigenvalues:
 [-0.02231724+0.j          3.6104532 +1.72034086j  3.6104532 -1.72034086j
  2.80141085+0.j        ]

Eigenvectors:
 [[-0.43676159+0.j          0.57598271+0.j          0.57598271-0.j
   0.26748859+0.j        ]
 [-0.36900111+0.j          0.00962337-0.2105225j   0.00962337+0.2105225j
  -0.34893941+0.j        ]
 [ 0.10239683+0.j         -0.00821729+0.55142262j -0.00821729-0.55142262j
  -0.06120796+0.j        ]
 [ 0.81399778+0.j         -0.18869091-0.53300366j -0.18869091+0.53300366j
  -0.89607183+0.j        ]]
```

[7]:
```python
P = eigenvectors
D = np.diag(eigenvalues)
P_inv = np.linalg.inv(P)

print("\nMatrix P (Eigenvectors):\n", P)
print("\nDiagonal Matrix D (Eigenvalues):\n", D)
print("\nInverse of Matrix P:\n", P_inv)
```

```
Matrix P (Eigenvectors):
 [[-0.43676159+0.j          0.57598271+0.j          0.57598271-0.j
   0.26748859+0.j        ]
 [-0.36900111+0.j          0.00962337-0.2105225j   0.00962337+0.2105225j
  -0.34893941+0.j        ]
 [ 0.10239683+0.j         -0.00821729+0.55142262j -0.00821729-0.55142262j
  -0.06120796+0.j        ]
 [ 0.81399778+0.j         -0.18869091-0.53300366j -0.18869091+0.53300366j
  -0.89607183+0.j        ]]

Diagonal Matrix D (Eigenvalues):
 [[-0.02231724+0.j          0.        +0.j          0.        +0.j
   0.        +0.j        ]
 [ 0.        +0.j          3.6104532 +1.72034086j  0.        +0.j
   0.        +0.j        ]
 [ 0.        +0.j          0.        +0.j          3.6104532 -1.72034086j
   0.        +0.j        ]
 [ 0.        +0.j          0.        +0.j          0.        +0.j
   2.80141085+0.j        ]]

Inverse of Matrix P:
 [[ 0.24293955+0.00000000e+00j -1.52243125+6.81957034e-17j
   0.05807612+1.36391407e-16j  0.66140356-6.81957034e-17j]
 [ 1.00207027+1.76334426e-02j -0.26464982-6.27009101e-02j
   0.2698955 -8.46128808e-01j  0.38375201+8.74766701e-02j]
 [ 1.00207027-1.76334426e-02j -0.26464982+6.27009101e-02j
   0.2698955 +8.46128808e-01j  0.38375201-8.74766701e-02j]
 [-0.18035771-0.00000000e+00j -1.34612117-0.00000000e+00j
  -1.06750294+2.75134487e-17j -0.57270992-7.47247360e-17j]]
```

0.0.1 Explanation of Diagonalization Importance

In linear algebra, diagonalizing a matrix simplifies many computations, especially when raising a matrix to a power. A diagonalized matrix represents the system in a simpler form, where each eigenvalue indicates an invariant direction of transformation. This is valuable for understanding the matrix's stability and the long-term behavior of systems.

Diagonalization also reveals insights into the system's properties, such as:

- **Stability**: Eigenvalues can indicate stability in dynamical systems.

- **Matrix powers**: Calculating powers of a diagonal matrix is straightforward, which is useful in various applications.

Thus, diagonalization is a powerful tool in linear algebra and applicable across fields like physics, engineering, and machine learning.

[14]:
```
U, sigma, Vt = np.linalg.svd(A)
Sigma = np.zeros((A.shape[0], A.shape[1]))
np.fill_diagonal(Sigma, sigma)

A_reconstructed = U @ Sigma @ Vt
print("\nReconstructed Matrix A (using SVD):\n", A_reconstructed)
```

```

Reconstructed Matrix A (using SVD):
 [[ 4.00000000e+00 -2.00000000e+00  2.00000000e+00  1.00000000e+00]
 [ 1.00000000e+00  1.00000000e+00 -7.77156117e-16  1.00000000e+00]
 [-2.00000000e+00  1.00000000e+00  3.00000000e+00 -1.00000000e+00]
 [ 1.00000000e+00  3.00000000e+00 -1.00000000e+00  2.00000000e+00]]
```

[ ]: