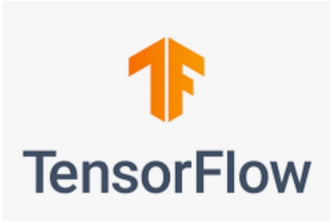

Part I I

Matrix, Matrix Decomposition and Partial Differentiation

Ayokunle Awelewa (Covenant University)

@

The 3rd Google TensorFlow College Outreach Bootcamp and FEDGEN Mini-Workshop

Date: 11th to 13th December, 2023

Sponsored

By

Contents

- Matrix
- Matrix Decomposition
- Partial Differentiation

Matrices

Important properties

Determinant of a matrix:

- The determinant of a square matrix $A \in \mathbb{R}^{n \times n}$ is the factor by which the transformation Ax_i changes volumes in an n-dimensionally space, where x_i is the set of points in the space. For instance, if points x_i form an enclosure with an area of p in a 2-dimensionally space, then the enclosure formed by Ax_i has an area of $p \times \det(A)$.

- The determinant of a square matrix

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}$$

may be computed (recursively) as follows:

$$\det(A) = a_{11}c_{11} + a_{12}c_{12} + \cdots + a_{1n}c_{1n}$$

or

$$\det(A) = a_{11}c_{11} + a_{21}c_{21} + \cdots + a_{n1}c_{n1}$$

or

$$\det(A) = a_{12}c_{12} + a_{22}c_{22} + \cdots + a_{n2}c_{n2}$$

etc.

$$c_{ij} = (-1)^{i+j} \det\left(A^{ij}\right)$$

where c_{ij} are called the cofactors of A, and A^{ij} is the $(n-1) \times (n-1)$ matrix formed by deleting the i-th row and j-th column from A.

- Note that the determinant of a scalar (i.e., 1×1 matrix) is defined as the scalar itself.

- The determinant of a square matrix has the following properties:
  - $det(I) = 1$
  - $det(ABC) = det(A)det(B)det(C)$
  - $det(A^{-1}) = \frac{1}{det(A)}$
  - $det(\alpha A) = \alpha^n det(A)$
  - $det(A^{\mathrm{T}}) = det(A)$

- If U is a diagonal or triangular matrix, then $det(U) = \prod_{i=1}^{n} u_{ii}$
- The determinant of a block-diagonal or block-triangular matrix is the product of the determinants of the diagonal blocks.
- det(A) = 0 if A has two rows that are equal, or a row that is a linear combination of the other rows. The same holds for the columns of A.

- The determinant does not change if the row multiplied with a scalar is subtracted from another row. The same holds for the columns.
- The sign of the determinant changes if two rows are interchanged. The same holds for the columns.

- A square matrix is said to be singular (or non-singular) if its determinant is zero (or non-zero).

Adjoint or adjugate of a matrix:

- The adjoint of a square matrix $\mathrm{A} \in \mathbb{R}^{n \times n}$ is the transpose of the cofactor matrix $\mathrm{C} \in \mathbb{R}^{n \times n}$. That is,

$$\mathrm{adj}(\mathrm{A}) = C^{\mathrm{T}} = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ c_{n1} & c_{n2} & \cdots & c_{nn} \end{bmatrix}^{\mathrm{T}}$$

- An important relation between a matrix and its adjugate is: $\mathrm{adj}(\mathrm{A}) \times \mathrm{A} = \det(\mathrm{A}) \times \mathrm{I} = \mathrm{A} \times \mathrm{adj}(\mathrm{A})$

Inverse of a matrix:

- The inverse of a square matrix $\mathrm{A} \in \mathbb{R}^{n \times n}$ is given as

$$\mathrm{A}^{-1} = \frac{adj(A)}{\det(A)}$$

and

$$\mathrm{A}^{-1}\mathrm{A} = \mathrm{A}\mathrm{A}^{-1} = \mathrm{I}$$

- Note that

$$(\mathrm{ABC})^{-1} = \mathrm{C}^{-1}B^{-1}A^{-1}$$

Generalized or pseudo inverse:

- Consider a non-square matrix $\mathrm{A} \in \mathbb{R}^{m \times n}$.
  - Left generalized or pseudo inverse:

$$A^{+}A = I_{n \times n} \text{ and } A^{+} = \left(A^{\mathrm{T}}A\right)^{-1}A^{\mathrm{T}}$$

  - Right generalized or pseudo inverse:

$$AA^{+} = I_{m \times m} \text{ and } A^{+} = A^{\mathrm{T}}\left(AA^{\mathrm{T}}\right)^{-1}$$

Rank of a matrix:

- The rank of a matrix is the number of linearly independent columns or rows.
- The rank is the order of the largest non-singular, square submatrix of the matrix.
- For the rank of a non-square matrix $\mathrm{A} \in \mathbb{R}^{m \times n}$, there exists an upper bound of $\min(m, n)$.
- If the rank of a square matrix $\mathrm{A} \in \mathbb{R}^{n \times n}$ is n, then the matrix is said to be of full rank; otherwise, it is rank-deficient (or not of full rank).

- The following statements are all equivalent for an $n \times n$ rank-deficient matrix A:
  - A is singular
  - A^T is singular
  - A is non-invertible
  - The columns of A are linearly dependent
  - Rows of A are linearly dependent

```
#Elementary matrix operations
import numpy as np
A=np.array([[1,2],[3,4]])
B=np.array([[5,6],[7,8]])

#basic matrix operations
addition=A+B
elementOpe=A*B
matrixMult=np.dot(A,B)
transpose=A.T
inverse_A=np.linalg.inv(A)
matNorm=np.linalg.norm(A)
matDet=np.linalg.det(A)
traceA=np.trace(A)
conjugateTrans=np.conjugate(A.T)
rank_A = np.linalg.matrix_rank(A)
rows, cols = A.shape
```

Special Matrices

Square matrices:

- Any matrix $A \in \mathbb{R}^{n \times n}$ with the same number of rows as columns is termed *square*; n is sometimes referred to as the order of the matrix.

- The principal or main diagonal of a square matrix goes from the top-left corner down to the bottom-right corner.

- The elements of the main diagonal are referred to as the diagonal elements.
- The anti-diagonal starts in the top right-hand corner, and continues diagonally to the left.
- The diagonals over and under the main diagonal are called the super-diagonals and sub-diagonals, respectively

Null matrices:

- If all the elements of a matrix are zero, the matrix is called a null matrix and denoted by **O**.

Identity matrices:

- The identity matrix of order n, denoted by **I** or **I**n is an n x n matrix in which all the diagonal elements are 1 and all the off-diagonal elements are 0.

- A matrix multiplied with the identity matrix remains unchanged; thus, it holds that IA = A and AI = A**.**

Diagonal and tridiagonal matrices:

- A square matrix is said to be a diagonal matrix if all of its off-diagonal elements are zero.
- A square matrix is said to be tridiagonal if non-zero elements only exist on the main diagonal, on the first super-diagonal and on the first sub-diagonal.
- More generally, a matrix is said to be a band matrix if all its non-zero elements are located in a band around the main diagonal.

Triangular matrices:

- A triangular matrix is a square matrix with zero elements below (or above) its principal diagonal.
- A matrix $\mathrm{V} \in \mathbb{R}^{n \times n}$ is upper triangular if $v_{ij} = 0$ for $i > j$; it is lower triangular if $v_{ij} = 0$ for $i < j$.
- In addition, if all the elements on the diagonal are also zero, then the matrix is said to be strictly triangular.

Symmetric matrices:

- For a square matrix $S \in \mathbb{R}^{n \times n}$, if $S = S^T$, then S is symmetric, and its inverse S^{-1} is also symmetric.
- Generally, for a real matrix H, HH^T and H^TH are always symmetric.

Skew-symmetric matrices:

- For a square matrix $S \in \mathbb{R}^{n \times n}$, if $S = -S^T$, then S is anti-symmetric or skew-symmetric.
- All the diagonal elements of a skew-symmetric matrix are zero, which means that the determinant of a skew-symmetric matrix is always zero.

- Any square matrix $A \in \mathbb{R}^{n \times n}$ can be uniquely expressed as the sum of a symmetric matrix and a skew-symmetric matrix. That is,

$$A = \left(\frac{A + A^{T}}{2}\right) + \left(\frac{A - A^{T}}{2}\right)$$

where $\left(\frac{A+A^{T}}{2}\right)$ is symmetric and $\left(\frac{A-A^{T}}{2}\right)$ is skew-symmetric.

Orthogonal matrices:

- For a square matrix $A \in \mathbb{R}^{n \times n}$, if $A^{-1} = A^T$, then A is orthogonal.
- Since the columns (and rows) vectors of the matrix are of unit length and orthogonal to each other, the matrix is also called orthonormal.

Normal matrices:

- For a square matrix $\mathrm{A} \in \mathbb{R}^{n \times n}$, if $\mathrm{A}^{\mathrm{T}}\mathrm{A} = \mathrm{A}\mathrm{A}^{\mathrm{T}}$, then A is normal.
- All symmetric, skew-symmetric, and orthogonal matrices are normal matrices.

Block matrices:

- The elements of a given matrix can be grouped into blocks to form elements that are submatrices. This process is called partitioning, and the resulting matrix is called a block matrix.

- The matrix elements in a block matrix are grouped horizontally and vertically in an array.

- All block elements along the same horizontal line must have the same number of rows, while all block elements along the same vertical line must have the same number of columns.

- If two or more block matrices can be added if they are of the same dimension and identically partitioned.

- Similarly, if the two matrices A and B are conformable for multiplication, then they can be conformably partitioned for multiplication as well:
  - To each column partition line in A, there corresponds a row partition line in B such that the number of columns of A between two adjacent partition lines is equal to the number of rows of B between the corresponding adjacent lines.

- A block matrix with element matrices $A_{ij} \in \mathbb{R}^{m_i \times n_j}$, $i = 1, \cdots, m,\ j = 1, \cdots, n$ is represented by

$$A = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1n} \\ A_{21} & A_{22} & \cdots & A_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{m1} & A_{m2} & \cdots & A_{mn} \end{bmatrix} \in \mathbb{R}^{\sum_{i=1}^{m} m_i \times \sum_{j=1}^{n} n_j}$$

Matrix Norm

Matrix norm:

- Recall the linear transformation $y = Ax$ from $\mathbb{R}^n$ to $\mathbb{R}^m$ by an $m \times n$ matrix A. The induced 2-norm of A is defined by

$$\|A\| = \max_{\|x\|=1} \|Ax\| = \left[\lambda_m\left(A^T A\right)\right]^{1/2}$$

  where $\lambda_m\left(A^T A\right)$ is the maximum eigenvalue of $A^T A$.

- For real matrices $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{n \times r}$,

$$\|AB\| \leq \|A\| \times \|B\|.$$

- Other matrix norms are:
  - One-norm of $A \in \mathbb{R}^{m \times n}$: the maximum of the sums of absolute values of the columns of A, given by

$$\|\mathrm{A}\|_1 = \max_{1 \le j \le n} \sum_{i=1}^{m} |a_{ij}|.$$

  - Infinity norm of $A \in \mathbb{R}^{m \times n}$: the maximum of the sums of absolute values of the rows of A, given by

$$\|\mathrm{A}\|_\infty = \max_{1 \le i \le m} \sum_{j=1}^{n} |a_{ij}|.$$

Matrix Decomposition

Singular value decomposition:

- Similar to eigenvalue decomposition for a square matrix, a non-square matrix can be decomposed as well.
- Consider a matrix $\mathrm{A} \in \mathbb{R}^{m \times n}$. Then

$$\mathrm{A}v_i = \sigma_i u_i$$

where $u_i \in \mathbb{R}^m$ and $v_i \in \mathbb{R}^n$ are called the right and left singular vectors of A, respectively;

σ_i are called the singular values of A.

- The right singular vectors $u_i \in \mathbb{R}^m$ represent the eigenvectors of AA^T.

- The left singular vectors and $v_i \in \mathbb{R}^n$ represent the eigenvectors of A^TA.

- The singular values σ_i denote the positive square roots of the eigenvalues of AA^T (they are nonnegative real numbers).

- The singular value decomposition (SVD) of A can be written as

$$\mathrm{A} = \mathrm{U}\Sigma\mathrm{V}^{\mathrm{T}}$$

$\mathrm{U} \in \mathbb{R}^{m \times m}$: matrix of right singular values u_i

$\mathrm{V} \in \mathbb{R}^{n \times n}$: matrix of left singular values v_i

$\Sigma \in \mathbb{R}^{m \times n}$: diagonal matrix of the singular values σ_i

- Note that, if $m < n$, the singular values σ_i are placed on the main diagonal of the $m \times m$ submatrix of Σ; otherwise ($n < m$), they are placed on the main diagonal of the $n \times n$ submatrix of Σ.

- The singular values are ordered so that $\sigma_1 \geq \sigma_2 \geq \ldots. \geq \sigma_\rho \geq 0$, where $\rho \leq \min(m, n)$. All other σ_k, if any, are zero. The structure of Σ is

$$\begin{bmatrix} \sigma_1 & & 0 \\ & \ddots & \\ & & \sigma_p \\ & & \\ 0 & & 0 \end{bmatrix}$$

- There is a relationship between the pseudo inverse of A and its SVD.
- An $m \times n$ matrix A with the SVD

$$\mathrm{A} = \mathrm{U}\Sigma\mathrm{V}^{\mathrm{T}}$$

has a pseudo inverse A^{+} given as

$$\mathrm{A}^{+} = \mathrm{V}\Sigma^{+}\mathrm{U}^{\mathrm{T}}$$

where Σ^{+} is the pseudo inverse of Σ obtained by finding the inverse of the non-zero singular values and transposing the resulting matrix.

- The pseudo inverse is useful in situations in which the matrix is non-square or non-invertible.

- The singular values are useful in finding the condition number of a matrix.
- The condition number is the ratio of the maximum singular value to the minimum singular value or
- $\mathrm{cond}(A) = \|A\|.\|A^{-1}\|$
- The condition number measures the sensitivity of the solution of a linear system of equations to changes in the input data (or coefficients) of the system.

- If the condition number is high, it means the matrix is close to being singular and is said to be poorly conditioned.
- A matrix is well-conditioned if its condition number is close to 1, and any small changes input data will, in the same proportion, give small changes in the output.

Python code for singular decomposition

```
import numpy as np

# Create a matrix
A = np.array([[4, 2, 1], [1, 3, 4]])

# Perform SVD
U, S, Vt = np.linalg.svd(A)

# Compute pseudoinverse
SI = np.linalg.pinv(A)

print(“SVD U matrix:")
print(U)
print("\nSingular values S:")
print(np.diag(S))
print("\nSVD V-transpose matrix:")
print(Vt)
print("\nPseudoinverse SI:")
print(SI)
```

LU factorization or decomposition:

- The LU decomposition decomposes a matrix into the product of a lower triangular matrix (L) and an upper triangular matrix (U) as follows:

  PA = LU

  P: permutation matrix

  L: lower triangular matrix with ones on the diagonal

  U: upper triangular matrix

- A permutation matrix has the same columns as the identity matrix, but in different order. There is exactly one unit entry in each row and in each column.
- Swapping two rows or columns of an identity matrix results in a permutation matrix.

Python code for LU decomposition

```
import scipy as sc

# Create a matrix
A = sc.array([[1, 2], [3, 4]])

# Perform LU decomposition
P, L, U = sc.linalg.lu(A)

print("Permutation matrix P:")
print(P)
print("\nLower triangular matrix L:")
print(L)
print("\nUpper triangular matrix U:")
print(U)
```

Cholesky decomposition:

- A symmetric and positive definite matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ can be decomposed as

$$\mathbf{A} = \mathbf{G}\mathbf{G}^{\mathbf{T}}$$

$\mathbf{G} \in \mathbb{R}^{n \times n}$: lower triangular

- Note that a matrix is positive definite if all its eigenvalues are greater than zero (that is, positive)
- Similarly, a matrix is positive semi-definite if all its eigenvalues are greater than or equal to zero (that is, nonnegative).
- Negative definiteness and semi-definiteness can be similarly defined.

Python code for Cholesky factorization

```
import numpy as np

# Create a positive-definite matrix
A = np.array([[4, -1, 0], [-1, 5, 2], [0, 2, 9]])

# Perform Cholesky decomposition
L = np.linalg.cholesky(A)

print("Lower triangular matrix L:")
print(L)
```

QR decomposition:

- An $\boldsymbol{m \times n}$ (or $\boldsymbol{n \times n}$) matrix $\mathbf{A}$ can be decomposed as

$$\mathbf{A = QR}$$

$\mathbf{Q} \in \mathbb{R}^{\boldsymbol{m \times n}}$**:** orthogonal matrix

$\mathbf{R} \in \mathbb{R}^{\boldsymbol{n \times n}}$: upper triangular

or

$\mathbf{Q} \in \mathbb{R}^{\boldsymbol{n \times n}}$**:** orthogonal matrix

$\mathbf{R} \in \mathbb{R}^{\boldsymbol{n \times n}}$: upper triangular

Python code for QR decomposition
A = np.array([[1, 3], [4, 5],[0, 2]]) # Perform QR decomposition Q, R = np.linalg.qr(A) print(“Orthogonal matrix Q:") print(Q) print("\nUpper triangular matrix R:") print(R)

Partial Differentiation

- Our engineered world is dynamic, and things change all the time.
- Generally, differentiation is a way of determining the degree of change of a function as the independent variable of the function changes.
- In ordinary differentiation, a function with one independent variable is considered.
  - For instance, the derivative of $f(x)$ with respect to x is $\dot{f}(x) = \frac{dy}{dx} = f'(x)$

- In partial differentiation, a function changes with respect to two or more variables, and each derivative of the function with respect to any of the variables is called a partial derivative.
  - For instance, the derivatives of $f(x, y, z)$ with respect to x, y, and z are $\frac{\partial f}{\partial x}$, $\frac{\partial f}{\partial y}$, and $\frac{\partial f}{\partial z}$, respectively.
  - Note that for each partial derivative with respect to a variable, the other variables are assumed to be constant.

- Bear in mind that all the rules of ordinary differentiation (i.e., the product rule, the quotient rule, etc.) are similarly applicable to partial differentiation.

Basic Approach to Partial Differentiation

- To carry out partial differentiation, perform normal differentiation with respect to the independent variable of interest, and treat other variables as constants.
- Python has a symbolic math library for various mathematical operations using symbols. The library has to be imported as:
  - import sympy as sp

    to import the entire library, or
  - from sympy import symbols, diff, sin, cos

    to import functions *symbols, diff, sin,* and *cos*

Exercise 1

- For the functions
  - $v = 10w^3r - 5r^2x + wrx^3$, and
  - $v = 10sin50xr - 5coswx$,
  - $v = \frac{x+5r}{x-3w+r}$,

  find the partial derivatives $\frac{\partial v}{\partial x}$, $\frac{\partial v}{\partial r}$, $\frac{\partial v}{\partial w}$, and $\frac{\partial^2 v}{\partial x \partial r}$.

  As well, write a Python script to verify your answers.

Analytical Solution to Exercise 1

$$v = 10w^3r - 5r^2x + wrx^3$$

- $\frac{\partial v}{\partial x} = -5r^2 + 3wrx^2$
- $\frac{\partial v}{\partial r} = 10w^3 - 10rx + wx^3$
- $\frac{\partial v}{\partial w} = 30w^2r + rx^3$
- $\frac{\partial^2 v}{\partial x \partial r} = -10r + 3wx^2$

Python Code for Exercise 1

```
#Import the symbols and diff functions from the
#symbolic math library

from sympy import symbols, diff

#Define the variables
r, w, x = symbols('r w x')

#Define the function
f = 10*w**3*r - 5*r**2*x + w*r*x**3
```

```
#Compute the partial derivatives
df_dx = diff(f, x)  # Partial derivative with respect to x
df_dr = diff(f, r)  # Partial derivative with respect to r
df_dw = diff(f, w)  #  Partial derivative with respect to w
df_dxdr = diff(f, x, r)  # Mixed partial derivative:
                         # d/dx(df/dr) or d/dr(df/dx)
```

```
print("Original function:", f)
print("Partial derivative with respect to x:", df_dx)
print("Partial derivative with respect to r:", df_dr)
print("Partial derivative with respect to w:", df_dw)
print("Mixed partial derivative:", df_dxdr)
```

Small Changes in Independent Variables

- The partial differentiation technique can be employed to find an approximate change in a function as its independent variables change by small amounts.
- Consider the energy E stored by a variable-capacitance capacitor, which is given as

$$E = \frac{1}{2} CV^2,$$

where C is the variable capacitance and V the voltage across the capacitor.

- $E = \frac{1}{2}CV^2$

- If C and V change by small amounts δC and δV, respectively, the approximate change in E, denoted as δE, can be determined.
- Considering changes in C and V, the energy equation above can be rewritten as

  $\mathrm{E} + \delta E = \frac{1}{2}(C + \delta C)(V + \delta V)^2$.
- The above equation can be expanded into

  $\mathrm{E} + \delta E = \frac{1}{2}(C + \delta C)(V + \delta V)^2$

  $\frac{1}{2}(C + \delta C)(V^2 + 2V\delta V + (\delta V)^2)$

  $\left(\frac{1}{2}CV^2 + CV\delta V + \frac{1}{2}V^2\delta C\right) + \left(\frac{1}{2}C(\delta V)^2 + \frac{1}{2}\delta C(\delta V)^2 + V\delta C\delta V\right)$

- $E = \frac{1}{2}CV^2$

- Since δC and δV are small, the last equation above becomes

$$\mathrm{E} + \delta E = \left(\frac{1}{2}CV^2 + CV\delta V + \frac{1}{2}V^2\delta C\right)$$

or

$$\delta E \approx CV\delta V + \frac{1}{2}V^2\delta C$$

$$\boxed{\delta E = \frac{\partial E}{\partial V}\delta V + \frac{\partial E}{\partial C}\delta C}$$

- The last equation can be extended to a function with three or more independent variables.
- For instance, if $v = f(r, x, w)$, then

$$\delta v = \frac{\partial f}{\partial r}\delta r + \frac{\partial f}{\partial x}\delta x + \frac{\partial f}{\partial w}\delta w$$

Exercise 2

- Consider that $E = \frac{1}{2}CV^2$, where V = 50 volts and C = 10 μF. Find the change in E due to an increase of 0.5 volt in V and an increase of 0.1 μF in C.

  As well, write a Python script to verify your answer.

Analytical Solution to Exercise 2

$$E = \frac{1}{2}CV^2;\ \delta E = \frac{\partial E}{\partial V}\delta V + \frac{\partial E}{\partial C}\delta C$$

- V=50; $\delta V = 0.5$
- $C = 10\times10^{-6};\ \delta C = 0.1\times10^{-6}$
- $\frac{\partial E}{\partial V} = CV$
- $\frac{\partial E}{\partial C} = \frac{1}{2}V^2$
- $\delta E = (10\times10^{-6}\times50\times0.5) + \left(\frac{1}{2}\times50^2\times0.1\times 10^{-6}\right) = 375\times10^{-6}$ Joule

Python Code for Exercise 2

```
from sympy import symbols, diff
# Define the variables
C,V = symbols('C V')

# Define the function
E =0.5*C*V**2

# Compute the partial derivatives
dE_dC = diff(E, C)
dE_dV = diff(E, V)
```

```
# Substitute numerical values
C_value = 10E-6
V_value = 50

# Substitute values into the partial derivatives
dE_dC_value = dE_dC.subs([(C, C_value), (V, V_value)])
dE_dV_value = dE_dV.subs([(C, C_value), (V, V_value)])
```

```
# Calculate the change in E due to dC=0.1E-6 and dV=0.5
dC=0.1E-6
dV=0.5
dE=dE_dC_value*dC+ dE_dV_value*dV
```

```
#Output the results
print("Original function:", E)
print("Partial derivative with respect to C:", dE_dC)
print("Partial derivative with respect to V:", dE_dV)
print("Substitute C =", C_value, "and V =", V_value,
"into dE/dC:", dE_dC_value)
print("Substitute C =", C_value, "and V =", V_value,
"into dE/dV:", dE_dV_value)
```

```
print("Change in E due to a positive change in C and V is dE = ", dE)

if dE>0:
    print("E increases by", dE)
else:
    print("E decreases by", dE)
```

Rate of Change of Independent Variables

- Recall the energy function

  $E = \frac{1}{2} C V^2$,

  and its change δE (due to δV and δC) given as

  $\delta E = \frac{\partial E}{\partial V} \delta V + \frac{\partial E}{\partial C} \delta C$.

- If the rate at which C and V are changing is known, then the rate of change of E can be determined.

- If the rate of change of C is $\frac{\delta C}{\delta t}$, and that of V is $\frac{\delta V}{\delta t}$, then the rate of change of E can be expressed as $\frac{\delta E}{\delta t} = \frac{\partial E}{\partial V}\frac{\delta V}{\delta t} + \frac{\partial E}{\partial C}\frac{\delta C}{\delta t}$.
- As $\delta t \to 0$, $\frac{\delta E}{\delta t} \to \frac{dE}{dt}$, $\frac{\delta V}{\delta t} \to \frac{dV}{dt}$, and $\frac{\delta C}{\delta t} \to \frac{dC}{dt}$.
- Therefore,

$$\frac{dE}{dt} = \frac{\partial E}{\partial V}\frac{dV}{dt} + \frac{\partial E}{\partial C}\frac{dC}{dt}$$

Exercise 3

- The position of an object in a two-dimensional space (i.e., a plane) is represented by coordinates x and y and defined as

$$x = L_1 cos\theta_1 + L_2 \cos(\theta_1 + \theta_2)$$

$$y = L_1 sin\theta_1 + L_2 \sin(\theta_1 + \theta_2)$$

If $\theta_1 = \frac{44}{63}$ rad and changes at 0.5 rad/s, while $\theta_2 = \frac{11}{21}$ rad and changes at 0.75 rad/s, find the rate at which x and y change. Also, write a Python script to verify your answers. [$L_1 = 10$ units; $L_2 = 5$ units.]

Analytical Solution to Exercise 3

$$x = L_1 cos\theta_1 + L_2 \cos(\theta_1 + \theta_2)$$

$$y = L_1 sin\theta_1 + L_2 \sin(\theta_1 + \theta_2)$$

- The rate at which x changes is

$$\frac{dx}{dt} = \frac{\partial x}{\partial \theta_1}\frac{d\theta_1}{dt} + \frac{\partial x}{\partial \theta_2}\frac{d\theta_2}{dt}$$

$$\frac{dx}{dt} = (-L_1 sin\theta_1 - L_2 \sin(\theta_1 + \theta_2)\,)\frac{d\theta_1}{dt} + (-L_2 \sin(\theta_1 + \theta_2))\frac{d\theta_2}{dt}$$

Inserting the given values results in

$$\frac{dx}{dt} = -9.089$$

- The rate at which y changes is

$$\frac{dy}{dt} = \frac{\partial y}{\partial \theta_1}\frac{d\theta_1}{dt} + \frac{\partial y}{\partial \theta_2}\frac{d\theta_2}{dt}$$

$$\frac{dy}{dt} = (L_1 cos\theta_1 + L_2 \cos(\theta_1 + \theta_2)\,)\frac{d\theta_1}{dt} + (L_2 \cos(\theta_1 + \theta_2))\frac{d\theta_2}{dt}$$

Inserting the given values results in

$$\frac{dy}{dt} = 5.964$$

Python Code for Exercise 3

```
from sympy import symbols, diff, sin, cos

# Define the variables
theta1,theta2 = symbols('theta1 theta2')

# Define the function
x=10*cos(theta1)+5*cos(theta1+theta2)
y=10*sin(theta1)+5*sin(theta1+theta2)
```

```
# Compute the partial derivatives
dx_dtheta1 = diff(x, theta1)
dx_dtheta2 = diff(x, theta2)
dy_dtheta1 = diff(y, theta1)
dy_dtheta2 = diff(y, theta2)

# Substitute numerical values
theta1_value = 44/63
theta2_value = 11/21
```

```
# Substitute values into the partial derivatives
dx_dtheta1_value = dx_dtheta1.subs([(theta1,
theta1_value), (theta2, theta2_value)])
dx_dtheta2_value = dx_dtheta2.subs([(theta1,
theta1_value), (theta2, theta2_value)])
dy_dtheta1_value = dy_dtheta1.subs([(theta1,
theta1_value), (theta2, theta2_value)])
dy_dtheta2_value = dy_dtheta2.subs([(theta1,
theta1_value), (theta2, theta2_value)])
```

```
# Calculate the change in x and y due to
#dtheta1_dt=0.5 and dtheta2_dt=0.75
dtheta1_dt=0.5
dtheta2_dt=0.75
dx_dt=dx_dtheta1_value*dtheta1_dt+
dx_dtheta2_value*dtheta2_dt
dy_dt=dy_dtheta1_value*dtheta1_dt+
dy_dtheta2_value*dtheta2_dt
```

```
#Output the results
print("Original function:", x)
print("Original function:", y)
print("Partial derivative of x with respect to
theta1:", dx_dtheta1)
print("Partial derivative of x with respect to
theta2:", dx_dtheta2)
print("Partial derivative of y with respect to
theta1:", dy_dtheta1)
print("Partial derivative of y with respect to
theta2:", dy_dtheta2)
```

```
print("Substitute theta1 =", theta1_value, "and theta2 =", theta2_value, "into dx/dtheta1:", dx_dtheta1_value)

print("Substitute theta1 =", theta1_value, "and theta2 =", theta2_value, "into dx/dtheta2:", dx_dtheta2_value)

print("Substitute theta1 =", theta1_value, "and theta2 =", theta2_value, "into dy/dtheta1:", dy_dtheta1_value)

print("Substitute theta1 =", theta1_value, "and theta2 =", theta2_value, "into dy/dtheta2:", dy_dtheta2_value)
```

```
if dx_dt>0:
    print("x increases by", dx_dt)
else:
    print("x decreases by", dx_dt)

if dy_dt>0:
    print("y increases by", dy_dt)
else:
    print("y decreases by", dy_dt)
```

Changes in Independent Variables as Functions of other Variables

- If v is a function of two variables p and q, i.e., $v = f(p, q)$, and p and q are respectively functions of variables r and w, i.e., $p = g(r, w)$ and $q = h(r, w)$, then v changes with respect to r and w as follows:

$$\frac{\partial v}{\partial r} = \frac{\partial f}{\partial p}\frac{\partial p}{\partial r} + \frac{\partial f}{\partial q}\frac{\partial q}{\partial r}$$

$$\frac{\partial v}{\partial w} = \frac{\partial f}{\partial p}\frac{\partial p}{\partial w} + \frac{\partial f}{\partial q}\frac{\partial q}{\partial w}$$

- Like a chain rule, the partial derivatives in the above equations must be followed in an orderly fashion.

- Assuming that v is a function of one variable p, i.e., $v = f(p)$, and p is itself a function of many other variables w_i, $i = 1, 2, 3, 4, \cdots, n$, i.e., $p = g(w_1, w_2, \cdots, w_n)$, then v changes with respect to w_i as follows:

$$\frac{\partial v}{\partial w_1} = \frac{\partial v}{\partial p}\frac{\partial p}{\partial w_1}$$

$$\frac{\partial v}{\partial w_2} = \frac{\partial v}{\partial p}\frac{\partial p}{\partial w_2}$$

$$\vdots$$

$$\frac{\partial v}{\partial w_n} = \frac{\partial v}{\partial p}\frac{\partial p}{\partial w_n}$$

Exercise 4

- Consider a hypothetical ANN which has two neurons in the input layer, one neuron in the hidden layer, and one in the output layer. Assume that the inputs are scalar and the hidden neuron uses a sigmoid activation function with bias b_h. Let the hidden neuron output be given as $o_h = f(x) = \frac{1}{1+e^{-x}}$, where $x = f(w_1, w_2) = u_1 w_1 + u_2 w_2 + b_h$. u_1 and u_2 are the inputs, while w_1 and w_2 are the input layer weights.

Exercise 4 Cont.

Find the partial derivatives of the hidden layer output with respect to the weights.

Solution to Exercise 4

- The partial derivatives are

$$\frac{\partial o_h}{\partial w_1} = \frac{\partial o_h}{\partial x}\frac{\partial x}{\partial w_1} = \frac{e^{-x}u_1}{(1+e^{-x})^2} = u_1 o_h(1-o_h)$$

$$\frac{\partial o_h}{\partial w_2} = \frac{\partial o_h}{\partial x}\frac{\partial x}{\partial w_2} = \frac{e^{-x}u_2}{(1+e^{-x})^2} = u_2 o_h(1-o_h)$$

END OF PART II

THANK YOU