Homework 01

1.Exercise from D2L.

Problem 4.

tensor with a shape of (2, 3, 4) is a 3rd-order tensor.

len(X) means length of the first dimension of X. So, len(X)=2.

Problem 5.

Yes, len(X) always correspond to the first dimension of X.

Problem 6.

If I run the code using torch, the result shows a RuntimeError indicating the numerator & denominator don't match.

But I believe "A / A.sum(axis=1)" is to make the sum of every row of A become 1.

2. Matrix & Vector Operations

(a)

$$C = \begin{bmatrix} 2 & -3 & 1 & -1 \\ -4 & 6 & -2 & 2 \\ 6 & -9 & 3 & -3 \\ 4 & -6 & 2 & -2 \end{bmatrix}, \quad d = [9]$$

C is a 2nd-order tensor, d is a 0th-order tensor.

```python
# Homework01 2-(a)
import torch

V_a = torch.tensor([1,-2,3,2]).unsqueeze(0)
V_a_t = torch.t(V_a)    # Transpose into coloum vector

V_b = torch.tensor([2,-3,1,-1]).unsqueeze(0)
V_b_t = torch.t(V_b)

V_c = V_a_t @ V_b

V_d = V_a @ V_b_t

print(V_c)
print(V_c.shape)
print(V_d)
print(V_d.shape)
```

(b) Given that D is a 2×2 diagonal matrix. $AD = \begin{bmatrix} a_{11}d_{11} & a_{12}d_{22} \\ a_{21}d_{11} & a_{22}d_{22} \\ a_{31}d_{11} & a_{32}d_{22} \end{bmatrix} = [a_1 \cdot d_{11} \quad a_2 d_{22}]$

let $AD = C$.

$ADB = CB = [a_1 d_{11} b_{11} + a_2 d_{22} b_{21} \quad \cdots \quad a_1 d_{11} b_{1V} + a_2 d_{22} b_{2V}]$

So, $[e_{ij}] = \sum_{i=1}^{2} a_i d_{ii} b_i^T$

(c)

```python
# Homework01 2-(c)
import torch

V_a = torch.arange(20)

V_a_reshape = V_a.reshape(5,4)

print(V_a)
print(V_a_reshape)
```

output:

```
tensor([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17,
        18, 19])
tensor([[ 0,  1,  2,  3],
        [ 4,  5,  6,  7],
        [ 8,  9, 10, 11],
        [12, 13, 14, 15],
        [16, 17, 18, 19]])
```

(d)

```python
# Homework01 2-(d)
import torch

V_a = torch.arange(20)

V_a_rsh = V_a.reshape(5,4)

HP_V_a = V_a_rsh * V_a_rsh

print(HP_V_a)
```

output:

```
tensor([[  0,   1,   4,   9],
        [ 16,  25,  36,  49],
        [ 64,  81, 100, 121],
        [144, 169, 196, 225],
        [256, 289, 324, 361]])
```

3.Tensor Operations.

```python
# Homework01 3-(a)
import torch

b = torch.arange(24)
B = b.reshape(2,3,4)

print("B = ", B)

# Homework01 3-(b)
sum_B = torch.sum(B)

print("sum of all elements  = ", sum_B)

# Homework01 3-(c)
C = torch.split(B, 1, dim = 0)[0]
D = torch.split(B, 1, dim = 0)[1]

print("C = ", C)
print("D = ", D)
```

```
B =  tensor([[[ 0,  1,  2,  3],
         [ 4,  5,  6,  7],
         [ 8,  9, 10, 11]],

        [[12, 13, 14, 15],
         [16, 17, 18, 19],
         [20, 21, 22, 23]]])
sum of all elements  =  tensor(276)
C =  tensor([[[ 0,  1,  2,  3],
         [ 4,  5,  6,  7],
         [ 8,  9, 10, 11]]])
D =  tensor([[[12, 13, 14, 15],
         [16, 17, 18, 19],
         [20, 21, 22, 23]]])
```

4.Matrix properties.

Homework01_4.py > ...

```python
# Homework01 4
import torch

# 4-(a)
A = torch.tensor([[1,2],[-2,1],[3,-1]],dtype=torch.float32)
rank_A = torch.linalg.  matrix_rank(A)

# print("A = \n", A)
print("\nrank of A = \n", rank_A)

# 4-(b)
U, S, V = torch.linalg.svd(A)
# print("\nU = \n", U)
# print("\nSigma = \n", S)
# print("\nV^T = \n", V)

K = torch.sum(S > 1e-10)    # K = number of Sigmas that are not zero
print("\nK = \n", K)

for i in range(int(K)):
    print(f"\nσ{i+1}:", S[i])
    print(f"u{i+1}:", U[:, i])
    print(f"v{i+1}:", V[i, :])

# 4-(c)
At = A.t()
B = A@At
L, W = torch.linalg.eig(B)

print(L)
print(W)
M = torch.sum(L.real > 1e-10)   # Using L to compare with 1e-10, output indicated "RuntimeError: "gt_cpu" not implemented for 'ComplexFl
print("\nM = \n", M)
for i in range(int(M)):
    print(f"\nλ{i+1}:", L[i])
    print(f"ω{i+1}:", W[:, i])
```

(a)

```
rank of A =
 tensor(2)
```

(b)

```
K =
 tensor(2)
```

```
σ1: tensor(3.8730)
u1: tensor([-0.0816,  0.5715, -0.8165])
v1: tensor([-0.9487,  0.3162])

σ2: tensor(2.2361)
u2: tensor([-9.8995e-01, -1.4142e-01, -2.9802e-08])
v2: tensor([-0.3162, -0.9487])
```

(c)

```
M =
 tensor(2)


λ1: tensor(15.0000+0.j)
ω1: tensor([ 0.0817+0.j, -0.5715+0.j,  0.8165+0.j])

λ2: tensor(5.+0.j)
ω2: tensor([ 9.8995e-01+0.j,  1.4142e-01+0.j, -1.7136e-07+0.j])
```