給定實驗資料

In [1]:

```
import torch
torch.manual_seed(0)

x = torch.randn(10,4, requires_grad=True)
W = torch.randn(4,4, requires_grad=True)
y = torch.randn(10,4, requires_grad=True)

print(x)
print(y)
print(W)
```

```
tensor([[-1.1258, -1.1524, -0.2506, -0.4339],
        [ 0.8487,  0.6920, -0.3160, -2.1152],
        [ 0.3223, -1.2633,  0.3500,  0.3081],
        [ 0.1198,  1.2377,  1.1168, -0.2473],
        [-1.3527, -1.6959,  0.5667,  0.7935],
        [ 0.5988, -1.5551, -0.3414,  1.8530],
        [-0.2159, -0.7425,  0.5627,  0.2596],
        [-0.1740, -0.6787,  0.9383,  0.4889],
        [ 1.2032,  0.0845, -1.2001, -0.0048],
        [-0.5181, -0.3067, -1.5810,  1.7066]], requires_grad=True)
tensor([[ 1.5091,  2.0820,  1.7067,  2.3804],
        [-1.1256, -0.3170, -1.0925, -0.0852],
        [ 0.3276, -0.7607, -1.5991,  0.0185],
        [-0.7504,  0.1854,  0.6211,  0.6382],
        [-0.0033, -0.5344,  1.1687,  0.3945],
        [ 1.9415,  0.7915, -0.0203, -0.4372],
        [-0.2188, -2.4351, -0.0729, -0.0340],
        [ 0.9625,  0.3492, -0.9215, -0.0562],
        [-0.6227, -0.4637,  1.9218, -0.4025],
        [ 0.1239,  1.1648,  0.9234,  1.3873]], requires_grad=True)
tensor([[ 0.2055, -0.4503, -0.5731, -0.5554],
        [ 0.5943,  1.5419,  0.5073, -0.5910],
        [-1.3253,  0.1886, -0.0691, -0.4949],
        [-1.4959, -0.1938,  0.4455,  1.3253]], requires_grad=True)
```

目標函數 $f = ||max(XW, 0) - Y||_F^2$

$f = ||\hat{Y} - Y||_F^2; \hat{Y} = max(Z, 0); Z = XW$

In [2]:

```
# f = (torch.clamp(x.mm(W), 0) - y).pow(2).sum()
f1 = (torch.clamp(x.mm(W), 0) - y).pow(2).sum()
# torch.clamp 讓小於零的值，賦值為零。
print(f1)
```

```
tensor(99.9048, grad_fn=<SumBackward0>)
```

In [3]:

```
# XW 矩陣乘法
z = x.mm(W)
print(z)
# 測試 torch 寫法
# test = torch.mm( x, W)
# print(test)
```

```
tensor([[ 0.0649, -1.2330, -0.1154,  0.8553],
        [ 4.1687,  1.0353, -1.0558, -3.5272],
        [-1.6094, -2.0869, -0.7125,  0.8028],
        [-0.3500,  2.1129,  0.3719, -1.6785],
        [-3.2240, -2.0529,  0.2291,  2.5247],
        [-3.1207, -3.0911, -0.2830,  3.2112],
        [-1.6198, -0.9920, -0.1762,  0.6243],
        [-2.4140, -0.8861, -0.0917,  0.6813],
        [ 1.8953, -0.6369, -0.5659, -0.1305],
        [-0.7464, -0.8685,  1.0108,  3.5132]], grad_fn=<MmBackward>)
```

In [4]:

```
# ReLU
m = torch.nn.ReLU()
tm = m(z)
y_hat = tm
# 建立第二次式
f2 = (y_hat - y).pow(2).sum()
```

In [5]:

```
print(f2)
```

```
tensor(99.9048, grad_fn=<SumBackward0>)
```

In [6]:

```
# W.grad.zero_()
print(W.grad)
```

```
None
```

In [7]:

```
# f.backward()
f2.backward()
```

直接微分求導

In [8]:

```
print(W.grad)
print(y.grad)
print(x.grad)
```

```
tensor([[ 18.2980,   2.7573,   2.3914,  -0.1974],
        [ 11.0817,   6.6428,   2.5163, -20.3225],
        [ -8.6662,   3.4506,  -1.8979,  -3.3608],
        [-21.1681,  -6.6739,  -1.0693,  27.0278]])
tensor([[ 2.8885e+00,  4.1639e+00,  3.4134e+00,  3.0501e+00],
        [-1.0589e+01, -2.7045e+00, -2.1849e+00, -1.7039e-01],
        [ 6.5523e-01, -1.5214e+00, -3.1982e+00, -1.5687e+00],
        [-1.5009e+00, -3.8551e+00,  4.9843e-01,  1.2764e+00],
        [-6.6077e-03, -1.0689e+00,  1.8791e+00, -4.2604e+00],
        [ 3.8829e+00,  1.5830e+00, -4.0504e-02, -7.2968e+00],
        [-4.3767e-01, -4.8701e+00, -1.4583e-01, -1.3166e+00],
        [ 1.9250e+00,  6.9834e-01, -1.8429e+00, -1.4750e+00],
        [-5.0359e+00, -9.2744e-01,  3.8436e+00, -8.0509e-01],
        [ 2.4780e-01,  2.3296e+00, -1.7491e-01, -4.2519e+00]])
tensor([[  1.1002,   0.0860,   5.3377,   0.2788],
        [  0.9583,  10.4633, -13.5234, -16.3639],
        [ -0.8712,  -0.9272,  -0.7764,   2.0790],
        [ -1.4504,   5.6914,   0.7613,  -0.9693],
        [ -1.2892,  -3.4714,  -1.9788,   4.8091],
        [ -4.0523,  -4.3127,  -3.6114,   9.6703],
        [ -0.7312,  -0.7782,  -0.6516,   1.7449],
        [ -0.8191,  -0.8718,  -0.7300,   1.9547],
        [  1.0350,   2.9930,  -6.6743,  -7.5333],
        [ -2.4616,  -2.4243,  -2.1164,   5.7128]])
```

公式推導求導

```
In [9]:
y_grad = -2*(y_hat-y)
print(y_grad)
```

```
tensor([[ 2.8885e+00,  4.1639e+00,  3.4134e+00,  3.0501e+00],
        [-1.0589e+01, -2.7045e+00, -2.1849e+00, -1.7039e-01],
        [ 6.5523e-01, -1.5214e+00, -3.1982e+00, -1.5687e+00],
        [-1.5009e+00, -3.8551e+00,  4.9843e-01,  1.2764e+00],
        [-6.6077e-03, -1.0689e+00,  1.8791e+00, -4.2604e+00],
        [ 3.8829e+00,  1.5830e+00, -4.0504e-02, -7.2968e+00],
        [-4.3767e-01, -4.8701e+00, -1.4583e-01, -1.3166e+00],
        [ 1.9250e+00,  6.9834e-01, -1.8429e+00, -1.4750e+00],
        [-5.0359e+00, -9.2744e-01,  3.8436e+00, -8.0509e-01],
        [ 2.4780e-01,  2.3296e+00, -1.7491e-01, -4.2519e+00]],
       grad_fn=<MulBackward0>)
```

```
In [10]:
v = abs(x.mm(W)* 0)
g = torch.heaviside(input =x.mm(W), values = v)
x_grad = 2*(torch.mul((y_hat-y),g)).mm(torch.t(W))
print(x_grad)
```

```
tensor([[  1.1002,   0.0860,   5.3377,   0.2788],
        [  0.9583,  10.4633, -13.5234, -16.3639],
        [ -0.8712,  -0.9272,  -0.7764,   2.0790],
        [ -1.4504,   5.6914,   0.7613,  -0.9693],
        [ -1.2892,  -3.4714,  -1.9788,   4.8091],
        [ -4.0523,  -4.3127,  -3.6114,   9.6703],
        [ -0.7312,  -0.7782,  -0.6516,   1.7449],
        [ -0.8191,  -0.8718,  -0.7300,   1.9547],
        [  1.0350,   2.9930,  -6.6743,  -7.5333],
        [ -2.4616,  -2.4243,  -2.1164,   5.7128]], grad_fn=<MulBackward0>)
```

```
In [11]:
W_grad = 2*torch.t(x).mm(torch.mul((y_hat-y),g))
print(W_grad)
```

```
tensor([[ 18.2980,   2.7573,   2.3914,  -0.1974],
        [ 11.0817,   6.6428,   2.5163, -20.3225],
        [ -8.6662,   3.4506,  -1.8979,  -3.3608],
        [-21.1681,  -6.6739,  -1.0693,  27.0278]], grad_fn=<MulBackward0>)
```

```
In [ ]:
```