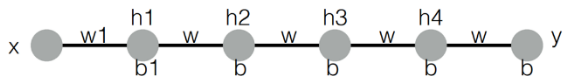
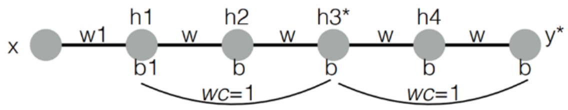
BS6207 Assignment 2

Q1: Vanishing gradient problem

I drive the $\frac{dy}{dw_1}, \frac{dy}{db_1}, \frac{dy^*}{dw_1}, \frac{dy^*}{dw_1}$:





$$\frac{dy}{dw_1} = \frac{\partial y}{\partial h_4}\frac{\partial h_4}{\partial h_3}\frac{\partial h_3}{\partial h_2}\frac{\partial h_2}{\partial h_1}\frac{\partial h_1}{\partial z_1}\frac{\partial z_1}{\partial w_1}$$
$$= \sigma'(z_5)w\,\sigma'(z_4)w\,\sigma'(z_3)w\,\sigma'(z_2)w\,\sigma'(z_1)\cdot x$$
$$= \sigma'(z_5)\ \sigma'(z_4)\ \sigma'(z_3)\ \sigma'(z_2)\ \sigma'(z_t)w^4\cdot x$$

$$\frac{dy}{db_t} = \frac{\partial y}{\partial h_4}\frac{\partial h_4}{\partial h_3}\frac{\partial h_3}{\partial h_2}\frac{\partial h_2}{\partial h_1}\frac{\partial h_1}{\partial z_1}\frac{\partial z_1}{\partial b_1}$$
$$= \sigma'(z_5)w\,\sigma'(z_4)w\,\sigma'(z_3)w\,\sigma'(z_2)w\,\sigma'(z_t)$$
$$= \sigma'(z_5)\ \sigma'(z_4)\ \sigma'(z_3)\ \sigma'(z_2)\ \sigma'(z_t)w^4$$

$$\frac{dy^*}{dw_t} = \frac{\partial y^*}{\partial h_3^*}\frac{\partial h_3^*}{\partial h_1}\frac{\partial h_1}{\partial z_1}\frac{\partial z_1}{\partial w_1}$$
$$= \sigma'(z_5)w_c\sigma'(z_3)\ w_c\sigma'(z_1)x$$
$$= \sigma'(z_5)\ \sigma'(z_3)\ \sigma'(z_1)\ x$$

$$\frac{dy^*}{db_t} = \frac{\partial y^*}{\partial h_3^*}\frac{\partial h_3^*}{\partial h_1}\frac{\partial h_1}{\partial z_1}\frac{\partial z_1}{\partial b_1}$$
$$= \sigma'(z_5)w_c\sigma'(z_3)\ w_c\sigma'(z_1)$$
$$= \sigma'(z_5)\ \sigma'(z_3)\ \sigma'(z_1)$$

Then I divide them, it turns out the result are the same of w and b.

$$\frac{|\frac{dy}{dw_1}|}{|\frac{dy^*}{dw_1}|} = \left|\frac{\sigma'(z_5)\ \sigma'(z_4)\ \sigma'(z_3)\ \sigma'(z_2)\ \sigma'(z_1)w^4\cdot x}{\sigma'(z_5)\ \sigma'(z_3)\ \sigma'(z_1)\cdot x}\right|$$
$$= |w^4\sigma'(z_4)\ \sigma'(z_2)|$$

$$\frac{|\frac{dy}{db_1}|}{|\frac{dy^*}{db_1}|} = \left|\frac{\sigma'(z_5)\ \sigma'(z_4)\ \sigma'(z_3)\ \sigma'(z_2)\ \sigma'(z_1)w^4}{\sigma'(z_5)\ \sigma'(z_3)\ \sigma'(z_1)}\right|$$
$$= |w^4\sigma'(z_4)\ \sigma'(z_2)|$$

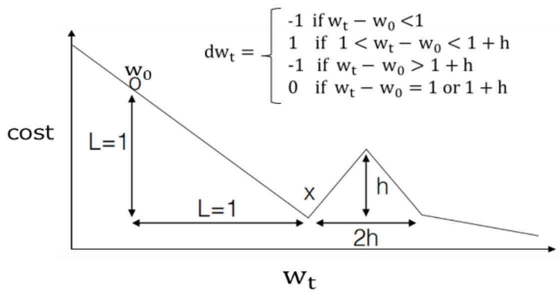
Since w<1, w^4<<1. And $\sigma'<1$, thus $|w^4\sigma'(z_4)\ \sigma'(z_2)|<1$. Thus, $|\frac{dy}{dw_1}| < |\frac{dy^*}{dw_1}|$, $|\frac{dy}{db_1}| < |\frac{dy^*}{db_1}|$.

Residual network with short circuit connections helps to solve vanishing gradient problem.

Q2: Local minimum problem

From the plot a 1D function, I can drive the differential equation shown in the right side. I first applied standard gradient descend to minimise this function at the point 'o'. Note that since h > a = 0.3, when w_4 = 1.2 away from w_0, the dw changes to +1. Thus, the w_5 is eques to w_3, which means the w is stuck at the point 'x'.

$$w_1 = w_0 - 0.3 * -1 = w_0 + 0.3$$
$$w_2 = w_t - 0.3 * -1 = w_0 + 0.6$$
$$w_3 = w_2 - 0.3 * -1 = w_0 + 0.9$$
$$w_4 = w_3 - 0.3 * -1 = w_0 + 1.2$$
$$w_5 = w_4 - 0.3 * 1 = w_0 - 0.9 = w_3$$



Then I applied Adam optimizer to minimise this function. For the convience of calculation, I wrote the script to find the maximum h. I first convert the algorithm of Adam optimizer into codes.

$$g_t = dw_{t-1}$$
$$m_t = \beta_1 m_{t\text{-}1} + (1\text{-}\ \beta_1)\ g_t$$
$$v_t = \beta_2 v_{t\text{-}1} + (1\text{-}\ \beta_2)\ {g_t}^2$$
$$\widehat{m}_t = \ m_t/(1\text{-}\beta_1^t)$$
$$\hat{v}_t = \ v_t/(1\text{-}\beta_2^t)$$
$$w_t \ = w_{t\text{-}1} \ - a\ \widehat{m}_t/(\sqrt{\hat{v}_t} + \epsilon)$$

```python
def AdamOptim(t, dw, m_dw, v_dw):
    # update the first moment estimate
    m_dw = beta1 * m_dw + (1 - beta1) * dw

    # update the second raw moment estimate
    v_dw = beta2 * v_dw + (1 - beta2) * (dw ** 2)

    # bias correction
    m_dw_corr = m_dw / (1 - beta1 ** t)
    v_dw_corr = v_dw / (1 - beta2 ** t)
    res = eta * (m_dw_corr / (np.sqrt(v_dw_corr) + epsilon))

    # update weights and biases
    return m_dw, v_dw, res
```

From the result we can see the max height 'h' of the bump in which the adam optimiser will escape the local min at 'x' is around 0.4101.

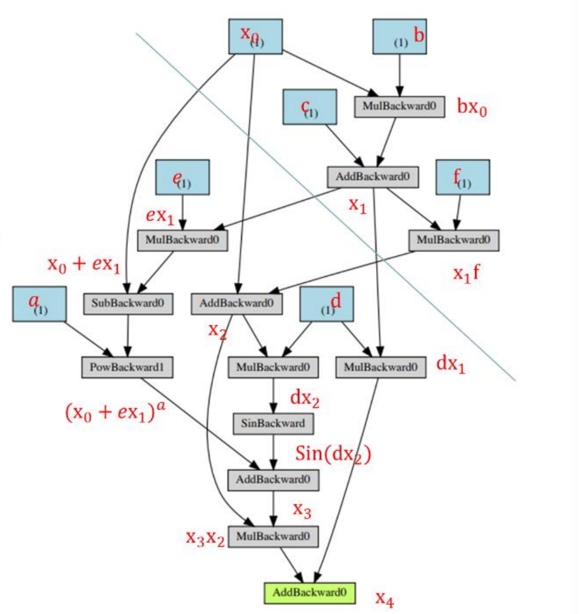
```
h = 0.4101----------Pass X!--------------
Updated w [0.3, 0.6, 0.9, 1.2, 1.353, 1.41, 1.514, 1.651, 1.816, 2.001]
h = 0.4102--------stuck at X!---------------
Updated w [0.3, 0.6, 0.9, 1.2, 1.353, 1.41, 1.399, 1.336, 1.235, 1.103, 0.947]
```

Besides, comparing to the change of w in standard gradient descend([0.3,0.6,0.9,1.2,0.9]), Adam optimizer converge slower and thus performs better. Because It takes advantage of momentum by using moving average of the gradient instead of gradient itself.

Q3: Label nodes of compute tree

we can split this tree into two parts (bule line) and solve it form the bottom to the top.

$$x_1 = b\,x_0 + c$$
$$x_2 = x_0 + x_1 f$$
$$x_3 \ = (x_0 + ex_1)^a \ + \mathrm{Sin}(dx_2)$$
$$x_4 \ = \ x_3 x_2 \ + \ dx_1$$



1. $x_4 \ = \ x_3 x_2 \ + \ dx_{t1}$

   Within it, x_3x_2 should be the parts containing sin, thus the right the parts should be dx_1. The d and x_1 can be solved.
2. $x_1 = b\,x_0 + c$

   Given the x_1, it is not hard the solve the top right part. The constant value should be c and the mul_node should be b x_0.The last node unsolved in the first part could only be f, which is only parameter left multiplying x_1.
3. $x_3 = (x_0 + ex_1)^a + \mathrm{Sin}(dx_2)$;

   Back to the x_3x_2 part, the one containing sin node should be x_3.Thus we could solve Sin(dx_2). $(x_0 + ex_1)^a$ is the one add with Sin(dx_2), which can also be solved. Parameter a is powered with $x_0 + ex_1$, should be the input of pow node. Given x_0 *and* x_1, we can settle down e.
4. $x_2 = x_0 + x_1 f$

   The last part should be x_2,which is natually solved by the previous deduction.