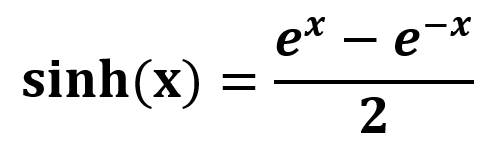
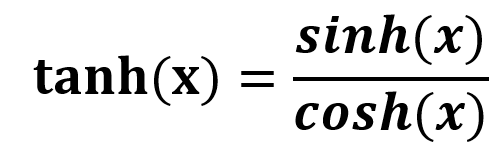
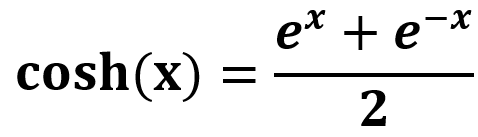
Tanh，又稱為雙曲正切函數，基本的定義如下：

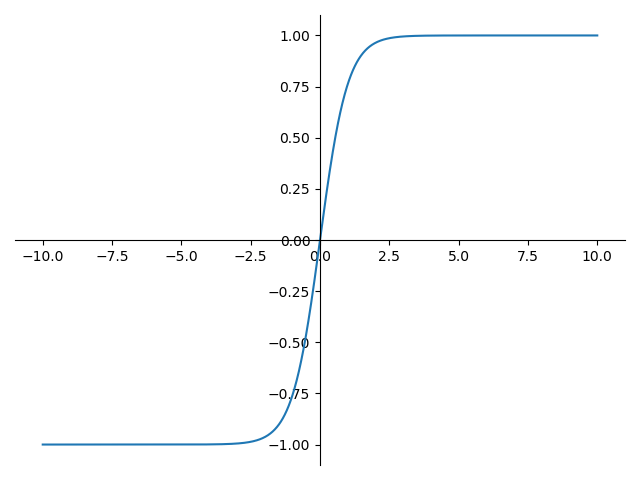
撰寫說明…

基本上便是 sinh(x) / cosh(x)，繪製出的圖案其實便是將我們輸入的 x 值映射在 [-1, 1] 之間。

以下我寫了一個簡單的 sample code 來顯示 tanh(x) 的圖形：

*# -\*- coding: utf-8 -\*-*  
**import** **matplotlib.pyplot** **as** **plt**  
**import** **math**  
  
value = -10  
x = [value+n\*0.01 **for** n **in** range(2001)]  
y = [math.tanh(v) **for** v **in** x]  
  
  
plt.plot(x, y)  
ax = plt.gca()  
ax.spines['right'].set\_color('none')  
ax.spines['top'].set\_color('none')  
ax.xaxis.set\_ticks\_position('bottom')  
ax.spines['bottom'].set\_position(('data',0))  
ax.yaxis.set\_ticks\_position('left')  
ax.spines['left'].set\_position(('data',0))  
plt.show()

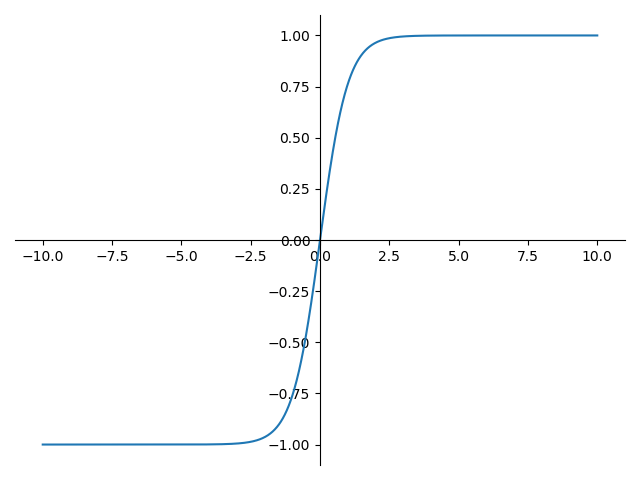
Output:



若是不用 python 內建的 math.tanh，而是改用定義的公式來計算：

**def** sinh(x):  
    **return** (math.exp(x)-math.exp(-1\*x))/2  
  
  
**def** cosh(x):  
    **return** (math.exp(x)+math.exp(-1\*x))/2  
  
  
**def** tanh(x):  
    **return** sinh(x)/cosh(x)  
  
value = -10  
x = [value+n\*0.01 **for** n **in** range(2001)]  
y = [tanh(v) **for** v **in** x]  
  
  
plt.plot(x, y)  
ax = plt.gca()  
ax.spines['right'].set\_color('none')  
ax.spines['top'].set\_color('none')  
ax.xaxis.set\_ticks\_position('bottom')  
ax.spines['bottom'].set\_position(('data',0))  
ax.yaxis.set\_ticks\_position('left')  
ax.spines['left'].set\_position(('data',0))  
plt.show()

Output:



我們也可以得到一樣的結果。

**應用**

* tanh 函數也存在著梯度飽和的問題，跟 sigmoid 函數一樣
* 收斂比 ReLU 慢