10월 9일까지 할 일

1. lab 04 슬라이드 7 페이지 example 실행해보기

- 실행하면서 각 코드들이 의미하는 바 이해하기

```
import tensorflow as tf
x1_data = [73., 93., 89., 96., 73.]
x2_data = [80., 88., 91., 98., 66.]
x3_data = [75., 93., 90., 100., 70.]
y_data = [152., 185., 180., 196., 142.]

# placeholders for a tensor that will be always fed.
x1 = tf.placeholder(tf.float32)
x2 = tf.placeholder(tf.float32)
x3 = tf.placeholder(tf.float32)
Y = tf.placeholder(tf.float32)

w1 = tf.Variable(tf.random_normal([1]), name='weight1')
w2 = tf.Variable(tf.random_normal([1]), name='weight2')
w3 = tf.Variable(tf.random_normal([1]), name='weight3')
b = tf.Variable(tf.random_normal([1]), name='bias')
hypothesis = x1 * w1 + x2 * w2 + x3 * w3 + b

# cost/loss function
cost = tf.reduce_mean(tf.square(hypothesis - Y))
# Minimize. Need a very small learning rate for this data set
optimizer = tf.train.GradientDescentOptimizer(learning_rate=1e-5)
train = optimizer.minimize(cost)

# Launch the graph in a session.
sess = tf.Session()
# Initializes global variables in the graph.
sess.run(tf.global_variables_initializer())
for step in range(2001):
    cost_val, hy_val, _ = sess.run([cost, hypothesis, train],
                         feed_dict={x1: x1_data, x2: x2_data, x3: x3_data, Y: y_data})
    if step % 10 == 0:
        print(step, "Cost: ", cost_val, "\nPrediction:\n", hy_val)
```

```
0 Cost:  19614.8
Prediction:
[ 21.69748688
39.10213089  31.82624626
35.14236832
32.55316544]
10 Cost:  14.0682
Prediction:
[ 145.56100464
187.94958496
178.50236511
194.86721802
146.08096313]

...

1990 Cost:  4.9197
Prediction:
[ 148.15084839
186.88632202
179.6293335
195.81796265
144.46044922]
2000 Cost:  4.89449
Prediction:
[ 148.15931702
186.8805542
179.63194275
195.81971741
144.45298767]
```

https://github.com/hunkim/DeepLearningZeroToAll/blob/master/lab-04-1-multi_variable_linear_regression.p

2. tmm package를 이용한 DBR (Distributed Bragg Relfector) simulation

- Tmm package 사용해보기
- Python 적응 (package 사용, 문법 등등)
- Documentation: https://pythonhosted.org/tmm/tmm.html

`tmm.tmm_core.`**coh_tmm**(*pol, n_list, d_list, th_0, lam_vac*)

Main "coherent transfer matrix method" calc. Given parameters of a stack, calculates everything you could ev calculations without affecting the rest.)

pol is light polarization, "s" or "p".

n_list is the list of refractive indices, in the order that the light would pass through them. The 0'th element (semi- infinite medium to which the light exits (if any exits).

th_0 is the angle of incidence: 0 for normal, pi/2 for glancing. Remember, for a dissipative incoming medium (lateral position).

d_list is the list of layer thicknesses (front to back). Should correspond one-to-one with elements of n_list. First

lam_vac is vacuum wavelength of the light.

Outputs the following as a dictionary (see manual for details)

- r–reflection amplitude
- t–transmission amplitude
- R–reflected wave power (as fraction of incident)
- T–transmitted wave power (as fraction of incident)
- power_entering–Power entering the first layer, usually (but not always) equal to 1-R (see manual).
- vw_list– n'th element is [v_n,w_n], the forward- and backward-traveling amplitudes, respectively, in the n't
- kz_list–normal component of complex angular wavenumber for forward-traveling wave in each layer.
- th_list–(complex) propagation angle (in radians) in each layer
- pol, n_list, d_list, th_0, lam_vac–same as input

DBR simulation 조건
- target: 1 THz (=300 um)
- pol: 's'
- n_h = 2.092 (SiO_2), n_l = 1 (Air) at 1 THz
- th_0: 0 (normal incidence)
- Wavelength: 150 um ~ 3000 um
- 10.5 pair
- 예제 그림과 같이 graph 출력