PROBLEM 1 **(25 points)**

Let $\mathbf{A}$ be the matrix that projects a vector $\mathbf{y} \in \mathbb{R}^3$ onto the plane $\begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = 0$. Provide your reasoning or show the derivation of your answer to the following questions:

(a) What are the eigenvalues of the matrix $\mathbf{A}$? (3 points)

(b) What are the eigenvectors of the matrix $\mathbf{A}$? (3 points)

(c) Find $\det(\mathbf{A})$. (3 points)

(d) Find $\mathbf{A}$. (3 points)

(e) Find $\mathbf{A}^2$. (3 points)

(f) Find $\mathbf{A}^+$ where $^+$ denotes the pseudo-inverse. (3 points)

(g) Find $\|\mathbf{A}\|_F^2$ (3 points)

(h) Find $\mathbf{B}\mathbf{B}^+$, where $\mathbf{B} = \begin{bmatrix} 1 & 0 \\ -1 & 0 \\ 0 & 1 \end{bmatrix}$. (4 points)

PROBLEM 2 **(25 points)**

(a) (15 points)

Consider a set of input vectors in $\mathbb{R}^n$, $\{\mathbf{x}^1, \mathbf{x}^2 \cdots \mathbf{x}^k\}$ and a set of output scalars $\{y^1, y^2 \cdots y^k\}$. We are interested in finding a regression model of the form $\hat{y}(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$.

Assume that, as a method for regularization, you add a noise vector ϵ^i to each input before learning the regression model. ϵ^i are independently drawn from a $\mathcal{N}(\mathbf{0}, \lambda\mathbf{I})$ distribution.

Consider the loss function $J(\mathbf{w}) = \frac{1}{k}\sum_{i=1}^{k}\left[\mathbf{w}^T\left(\mathbf{x}^i + \epsilon^i\right) - \mathbf{y}^i\right]^2$.

Find $\tilde{\mathbf{w}} = \underset{\mathbf{w}}{\operatorname{argmin}}\, \mathbb{E}\left(J(\mathbf{w})\right)$

(b) (10 points)

Consider $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$.
What unit vector ν when added to the input results in the maximum change in the output i.e $\|f(\mathbf{x}+\nu) - f(\mathbf{x})\|_2^2$?

Possibly helpful identities:-

$\nabla_{\mathbf{x}}\mathbf{a}^T\mathbf{x} = \mathbf{a}$

$\nabla_{\mathbf{x}}\mathbf{x}^T\mathbf{A}\mathbf{x} = (\mathbf{A} + \mathbf{A}^T)\mathbf{x}$

$$\underbrace{\mathbf{a}\mathbf{a}^T}_{\text{Matrix}}\ \mathbf{a} = \underbrace{(\mathbf{a}^T\mathbf{a})}_{\text{Scalar}}\ \mathbf{a}$$

PROBLEM 3 **(25 points)**

For each of the following questions, write down T (true) or F (false). Each question is worth 1 point.

1. Like many other machine learning frameworks, the deep learning approach formulates the learning task as an optimization problem minimizing a cost function, and applies the back-propagation algorithm to find the parameters that result in a global minima.
2. Every real-valued symmetric matrix is diagonalizable, and its determinant is nonzero.
3. L^1 regularization induces the sparsity property
4. The rectified linear unit $\phi(a) = \max(0, a)$ is widely used in hidden layers of the modern deep network frameworks. It is a better choice than *tanh* because it is bounded and gives constant derivative for positive input.
5. A model which achieves minimum training error also achieves minimum test error.
6. Unbiased estimators are always more preferable than biased ones.
7. If two random variables are independent, then they are also uncorrelated.
8. Given an abstract formulation of deep network model $y = f(x)$, it can be mathematically shown that minimizing the mean squared error of f yields an estimator of the conditional expectation of the output y given the input x.
9. The *softmax* function is a popular choice for the output layer of deep network because it can be interpreted to generate a probability distribution over a finite set of outcomes.
10. In order to achieve global optimal solution, deep learning approaches are often expressed in terms of convex optimization. One example is to use the squared norm of the error i.e $L(y, \hat{y}) = \|y - \hat{y}\|_2^2$ for the loss function, where y is the ground truth, and $\hat{y}$ is the outcome of the network.
11. Constrained optimization problems can be solved by the Karush-Kuhn-Tucker (KKT) approach, which transforms a constrained problem to an unconstrained one. The KKT approach always succeeds so long as at least one feasible point exists.
12. The Principle Component Analysis(PCA) is a common method for dimensionality reduction.
13. The learning rate of the gradient descent algorithm needs to be set carefully. A good practice is to use the eigenvalues of the Hessian of the cost function to determine the scale of the learning rate.
14. The *no free lunch theorem* implies that there is no universal procedure for examining a training set of specific examples and choosing a function that will generalize to points not in the training set.
15. The state-of-the-art deep network softwares use minibatch to allow parallelization of forward-propagation and back-propagation across examples.
16. In general, deeper models (models with more hidden layers) of deep network tend to perform better than a single hidden layer network having a similar number of parameters.
17. $D_{KL}(P||Q) = D_{KL}(Q||P)$ for all probability distributions $P(x), Q(x)$ over the same random variable

18. The MAP estimator is equivalent to the ML estimator when the prior follows uniform distribution over a bounded parameter space.

19. *Occam's razor* infers that the feed-forward network with at least one hidden layer and a linear output layer can approximate any continuous function from one finite space to another with arbitrary precision, provided that that the network is given enough hidden units.

20. Modern deep learning softwares usually implement the stochastic gradient descent rather than the classic gradient decent. One advantage of stochastic gradient decent is that it uses a predetermined stochastic process to generate a value for learning rate at each iteration.

21. Regularization methods in deep learning always involve adding a penalty function to the objective function.

22. The back-propagation algorithm is a systematic method based on the chain rule for computing the derivative of the cost function with respect to the parameters, starting from the last layer and going back to the first layer.

23. The deep learning library, Theano, handles automatic differentiation by using a symbolic representation of user-defined cost function to compute a symbolic representation of that function's gradient. This feature of Theano allows user to focus on design of forward path, whereas, historically, researchers used to spend great amount of time on deriving the gradient of complex networks.

24. Unsupervised learning algorithms experience a dataset containing features, but each example is also associated with a label or target.

25. Regularization is any component of model training or learning that is introduced to overcome limitations that arise with limited training datasets, such as overfitting.

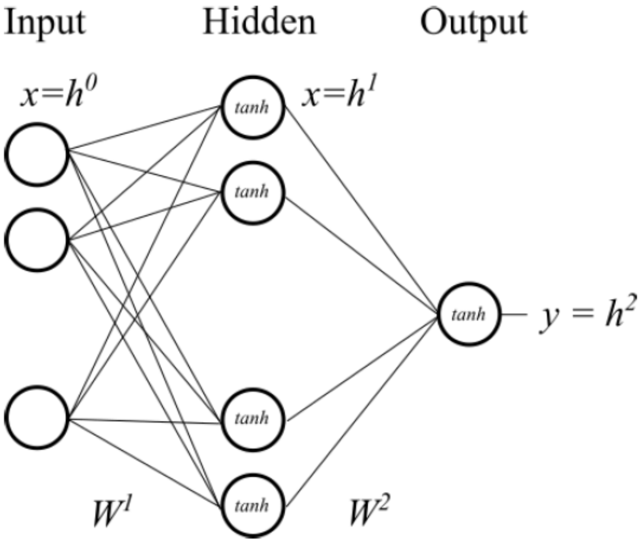
PROBLEM 4 **(25 points)**

Consider a shallow neural network with one input layer, one hidden layer, and one output layer. The input and hidden layers have N and M neurons, respectively, while the output layer has a single neuron.

In this model, we choose to use the hyperbolic tangent activation function:

$$h^2 = \tanh\left(b^2 + (\mathbf{h}^1)^T \mathbf{w}^2\right)$$
$$\mathbf{h}^1 = \tanh\left(\mathbf{b}^1 + (\mathbf{h}^0)^T \mathbf{W}^1\right)$$

$\mathbf{h}^0$ denotes the input, and $\mathbf{h}^1$ and h^2 are the output of the hidden layer and the output layer respectively. Consequently, $\mathbf{h}^0 \in \mathbb{R}^N$, $\mathbf{h}^1 \in \mathbb{R}^M$, and $h^2 \in \mathbb{R}$. $\mathbf{W}^1$ is the weight matrix from input to hidden layer and $\mathbf{w}^2$ is a weight vector from hidden layer to output layer. $\mathbf{b}^1$ and b^2 are the biases for the hidden and output layer respectively. The entire model is expressed as $y = f(\mathbf{x})$



For a given set of input $\{\mathbf{x}^1, ..., \mathbf{x}^k\}$ with their associated output $\{y^1, ..., y^k\}$, we use *mean squared error* as the cost function:

$$L = \frac{1}{k}\sum_{i=1}^{k}(y^i - f(\mathbf{x}^i))^2$$

In this problem, you are asked to derive the back-propagation algorithm.

(a) (10 points)

Derive $\dfrac{\partial L}{\partial \mathbf{w}^2}$ and $\dfrac{\partial L}{\partial \mathbf{h}^1}$. You might want to start with applying *chain rule*. $\big($Hint: $\frac{d}{dx}\tanh(x) = \mathrm{sech}^2(x)\big)$

(b) (10 points)

Derive $\dfrac{\partial L}{\partial \mathbf{W}^1}$. For this part, you might want to express $\mathbf{W}^1$ as

$$\mathbf{W}^1 = \begin{bmatrix} | & & | \\ \mathbf{w}_1^1 & \cdots & \mathbf{w}_M^1 \\ | & & | \end{bmatrix}$$

where $\mathbf{w}_i^1$, for $i = 1, ..., M$, is a column vector of $\mathbf{W}^1$. Then, derive $\dfrac{\partial L}{\partial \mathbf{w}_i^1}$ in terms of $\dfrac{\partial L}{\partial \mathbf{h}_i^1}$.

(c) (5 points)

If the loss function was instead defined as

$$L_r = \frac{1}{k}\sum_{i=1}^{k}(y^i - f(\mathbf{x}^i))^2 + \|\mathbf{W}^1\|_F^2 + \|\mathbf{w}^2\|_2^2$$

what will be the expressions for $\dfrac{\partial L}{\partial \mathbf{w}^2}$ and $\dfrac{\partial L}{\partial \mathbf{W}^1}$?

Possibly helpful identities:-

$\nabla_{\mathbf{x}}\mathbf{w}^T\mathbf{x} = \mathbf{w}$

$\|\mathbf{W}\|_F^2 = Tr(\mathbf{W}^T\mathbf{W}) = \sum_{i=1}^{k}(\mathbf{w}^i)^T\mathbf{w}^i$, where $\mathbf{w}^i$, for $i = 1, ..., k$, is a column vector of $\mathbf{W}$.