EECS 182 Deep Neural Networks
Fall 2023 Anant Sahai

Homework 1

This homework is due on Sunday, Sep 3, 2023, at 10:59PM.

1. Least Squares and the Min-norm problem from the Perspective of SVD

Consider the equation $X\mathbf{w} = \mathbf{y}$, where $X \in \mathbb{R}^{m \times n}$ is a non-square data matrix, w is a weight vector, and y is vector of labels corresponding to the datapoints in each row of X.

Let's say that $X = U\Sigma V^T$ is the (full) SVD of X. Here, U and V are orthonormal square matrices, and Σ is an $m \times n$ matrix with non-zero singular values (σ_i) on the "diagonal".

For this problem, we define $\Sigma^\dagger$ an $n \times m$ matrix with the reciprocals of the singular values ($\frac{1}{\sigma_i}$) along the "diagonal".

(a) First, consider the case where $m > n$, i.e. our data matrix X has more rows than columns (tall matrix) and the system is overdetermined. **How do we find the weights w that minimizes the error between $X\mathbf{w}$ and y?** In other words, we want to solve $\min_{\mathbf{w}} \|X\mathbf{w} - \mathbf{y}\|^2$.

Solution: Meta: Students may be confused about which form of SVD to be using. Make sure they know it is FULL SVD, U and V are square orthonormal matrices.

This is the classic least squares problem. The solution is given by

$$\hat{w} = (X^TX)^{-1}X^Ty$$

This can be derived from vector calculus, and also has an elegant interpretation in the context of orthogonal projection of $\mathbf{y}$ on the column space of X.

(b) **Plug in the SVD $X = U\Sigma V^T$ and simplify.** Be careful with dimensions!

Solution:

$$(X^TX)^{-1}X^T = (V\Sigma^TU^TU\Sigma V^T)^{-1}V\Sigma^TU^T$$

Since U has orthonormal columns, $U^TU = I$. Notice $\Sigma^T\Sigma$ is a square, $n \times n$ diagonal matrix with squared singular values σ_i^2 along the diagonal.

$$(X^TX)^{-1}X^T = (V\Sigma^T\Sigma V^T)^{-1}V\Sigma^TU^T$$

Apply the fact that $(AB)^{-1} = B^{-1}A^{-1}$, and that $V^{-1} = V^T$ since the matrix is orthonormal.

$$(X^TX)^{-1}X^T = V(\Sigma^T\Sigma)^{-1}V^TV\Sigma^TU^T$$

Simplify since $V^TV = I$.

$$(X^TX)^{-1}X^T = V(\Sigma^T\Sigma)^{-1}\Sigma^TU^T$$

Notice that $(\Sigma^T\Sigma)^{-1}\Sigma^T$ is an $n \times m$ matrix with the reciprocals of the singular values, $\frac{1}{\sigma_i}$, on the "diagonal". We can call this matrix $\Sigma^\dagger$. Note that this isn't a true matrix inverse (since the matrix Σ is not square). So we can write our answer as

$$(X^TX)^{-1}X^T = V\Sigma^\dagger U^T$$

You should draw out the matrix shapes and convince yourself that all the matrix multiplications make sense.

(c) You'll notice that the least-squares solution is in the form $\mathbf{w}^* = A\mathbf{y}$. **What happens if we left-multiply X by our matrix A?** This is why the matrix A of the least-squares solution is called the left-inverse.

Solution: $(X^TX)^{-1}X^TX = I$. We can also see this from our SVD interpretation,

$$V\Sigma^\dagger U^TU\Sigma V^T = V\Sigma^\dagger\Sigma V^T = VV^T = I$$

Students should make sure to understand why $\Sigma^\dagger\Sigma = I$ (What are the dimensions, and what are the entries?)

This is why the least-squares solution is called the left-inverse.

(d) Now, let's consider the case where $m < n$, i.e. the data matrix X has more columns than rows and the system is underdetermined. There exist infinitely many solutions for w, but we seek the minimum-norm solution, ie. we want to solve $\min \|\mathbf{w}\|^2 \text{s.t.} X\mathbf{w} = \mathbf{y}$. **What is the minimum norm solution?**

Solution: The min-norm problem is solved by

$$\mathbf{w} = X^T(XX^T)^{-1}\mathbf{y}$$

We can see this by choosing $\mathbf{w}$ that has a zero component in the nullspace of X, and thus $\mathbf{w}$ is in the range of X^T. Alternatively, one can write the Lagrangian, take the dual, apply the KKT conditions, and solve to get the same answer.

(e) **Plug in the SVD $X = U\Sigma V^T$ and simplify.** Be careful with dimensions!

Solution:

$$\begin{aligned} X^T(XX^T)^{-1} &= (U\Sigma V^T)^T(U\Sigma V^T(U\Sigma V^T)^T)^{-1} \\ &= V\Sigma^TU^T(U\Sigma V^TV\Sigma^TU^T)^{-1} \\ &= V\Sigma^TU^TU(\Sigma\Sigma^T)^{-1}U^T \\ &= V\Sigma^T(\Sigma\Sigma^T)^{-1}U^T \end{aligned}$$

Here, we have that $\Sigma^T(\Sigma\Sigma^T)^{-1}$ is an $n \times m$ matrix with the reciprocals of the singular values, $\frac{1}{\sigma_i}$, on the "diagonal". We can call this matrix $\Sigma^\dagger$ so that we have

$$= V\Sigma^\dagger U^T$$

(f) You'll notice that the min-norm solution is in the form $\mathbf{w}^* = B\mathbf{y}$. **What happens if we right-multiply X by our matrix B?** This is why the matrix B of the min-norm solution is called the right-inverse.

Solution: Similar to the previous part, $XX^T(XX^T)^{-1} = I$. This can also be seen from the SVD perspective.

This is why the min-norm solution is called the right-inverse.

2. The 5 Interpretations of Ridge Regression

(a) *Perspective 1: Optimization Problem.* Ridge regression can be understood as the unconstrained optimization problem

$$\operatorname*{argmin}_{\mathbf{w}} \|\mathbf{y} - X\mathbf{w}\|_2^2 + \lambda\|\mathbf{w}\|_2^2, \tag{1}$$

where $X \in \mathbb{R}^{n\times d}$ is a data matrix, and $\mathbf{y} \in \mathbb{R}^n$ is the target vector of measurement values. What's new compared to the simple OLS problem is the addition of the $\lambda\|\mathbf{w}\|^2$ term, which can be interpreted as a "penalty" on the weights being too big.

Use vector calculus to expand the objective and solve this optimization problem for w.

Solution: Call our objective f. Expand

$$f(\mathbf{w}) = \mathbf{w}^T X^T X\mathbf{w} - 2\mathbf{w}^T X\mathbf{y} + \mathbf{y}^T\mathbf{y} + \lambda\mathbf{w}^T\mathbf{w}$$

Take gradient wrt $\mathbf{w}$ and set it to zero:

$$\nabla_{\mathbf{w}} f = 2X^T X\mathbf{w} - 2X^T\mathbf{y} + 2\lambda\mathbf{w} = \mathbf{0}$$

Solve for $\mathbf{w}$:

$$\begin{aligned}(X^T X + \lambda I)\mathbf{w} &= X^T\mathbf{y}\\ \mathbf{w} &= (X^T X + \lambda I)^{-1}X^T\mathbf{y}\end{aligned}$$

(b) *Perspective 2: "Hack" of shifting the Singular Values.* In the previous part, you should have found the optimal $\mathbf{w}$ is given by

$$\mathbf{w} = (X^T X + \lambda I)^{-1}X^T\mathbf{y}$$

(If you didn't get this, you should check your work for the previous part).

Let $X = U\Sigma V^T$ be the (full) SVD of the X. Recall that U and V are square orthonormal (norm-preserving) matrices, and Σ is a $n \times d$ matrix with singular values σ_i along the "diagonal". **Plug this into the Ridge Regression solution and simplify. What happens to the singular values of** $(X^T X + \lambda I)^{-1}X^T$ **when** $\sigma_i << \lambda$**? What about when** $\sigma_i >> \lambda$**?**

Solution: We want to plug in $X = U\Sigma V^T$ into $\mathbf{w} = (X^T X + \lambda I)^{-1}X^T\mathbf{y}$. U and V are square (although they may be different sizes). Recall that U and V are orthonormal (real unitary) matrices, so $U^T U = UU^T = I$ and $VV^T = V^T V = I$.

$$\begin{aligned}\mathbf{w} &= ((U\Sigma V^T)^T(U\Sigma V^T) + \lambda I)^{-1}(U\Sigma V^T)^T\mathbf{y}\\ &= (V\Sigma^T U^T U\Sigma^T\Sigma V^T + \lambda I)^{-1}(V\Sigma^T U^T)\mathbf{y}\\ &= (V\Sigma^T\Sigma V^T + \lambda V I V^T)^{-1}(V\Sigma^T U^T\mathbf{y})\\ &= (V(\Sigma^T\Sigma + \lambda I)V^T)^{-1}(V\Sigma^T U^T\mathbf{y})\\ &= (V^{-T}(\Sigma^T\Sigma + \lambda I)V^{-1})(V\Sigma^T U^T\mathbf{y})\\ &= V(\Sigma^T\Sigma + \lambda I)^{-1}\Sigma^T U^T\mathbf{y}\end{aligned}$$

Now, let's consider the case when $n > d$. We have

$$\mathbf{w} = V \begin{bmatrix} \sigma_1^2 + \lambda & 0 & \dots & 0 \\ 0 & \sigma_2^2 + \lambda & \dots & 0 \\ \dots & & & \\ 0 & 0 & \dots & \sigma_d^2 + \lambda \end{bmatrix}^{-1} \begin{bmatrix} \sigma_1 & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & \sigma_2 & \dots & 0 & 0 & \dots & 0 \\ \dots & & & & & & \\ 0 & 0 & \dots & \sigma_d & 0 & \dots & 0 \end{bmatrix} U^T \mathbf{y}$$

$$= V \begin{bmatrix} \frac{\sigma_1}{\sigma_1^2+\lambda} & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & \frac{\sigma_2}{\sigma_2^2+\lambda} & \dots & 0 & 0 & \dots & 0 \\ \dots & & & & & & \\ 0 & 0 & \dots & \frac{\sigma_d}{\sigma_d^2+\lambda} & 0 & \dots & 0 \end{bmatrix} U^T \mathbf{y}$$

You can see the "diagonal" terms in the SVD form are $\frac{\sigma_i}{\sigma_i^2+\lambda}$. By adding the λ to the denominator, we prevent an explosion if any σ_i was close to zero. That is, when $\sigma_i << \lambda$, the σ_i^2 term becomes negligible and the effective singular value of the matrix in the solution is $\frac{\sigma_i}{\lambda}$. When $\sigma_i >> \lambda$, the λ term is negligible and the effective singular value of the matrix in the solution is $\frac{1}{\sigma_i}$, the same as it would be without any regularization.

The case when $n = d$ is similar, but without the extra zero-columns since the matrix Σ is square. The case when $n < d$ is also similar, except we would now have additional rows of 0's below the square. You can work out for yourself to see what those look like.

(c) *Perspective 3: Maximum A Posteriori (MAP) estimation.* Ridge Regression can be viewed as finding the MAP estimate when we apply a prior on the (now viewed as random parameters) $\mathbf{W}$. In particular, we can think of the prior for $\mathbf{W}$ as being $\mathcal{N}(\mathbf{0}, I)$ and view the random Y as being generated using $Y = \mathbf{x}^T\mathbf{W} + \sqrt{\lambda}N$ where the noise N is distributed iid (across training samples) as $\mathcal{N}(0,1)$. At the vector level, we have $\mathbf{Y} = X\mathbf{W} + \sqrt{\lambda}\mathbf{N}$. Note that the X matrix whose rows are the n different training points are not random.

Show that (1) is the MAP estimate for W given an observation $\mathbf{Y} = \mathbf{y}$.

Solution:

From how we define MAP estimation,

$$MAP(\mathbf{w}|\mathbf{Y}=\mathbf{y}) = \underset{\mathbf{w}}{\operatorname{argmax}}\, f(\mathbf{w}|\mathbf{Y}=\mathbf{y})$$

$$= \underset{\mathbf{w}}{\operatorname{argmax}}\, \frac{f(\mathbf{w},\mathbf{y})}{f(\mathbf{y})}$$

The denominator doesn't affect the argmax since it doesn't depend on $\mathbf{w}$, so we can omit it. Then we can use the chain rule to expand out the numerator

$$= \underset{\mathbf{w}}{\operatorname{argmax}}\, f(\mathbf{w})f(\mathbf{y}|\mathbf{w})$$

Now, split up the conditional joint density into the product of conditional densities since each element of the $\mathbf{y}$ is independent given $\mathbf{w}$.

$$= \underset{\mathbf{w}}{\operatorname{argmax}}\, f(\mathbf{w}) \prod_{i=1}^{n} f(y_i|\mathbf{w})$$

We can now recall the formula for standard normal pdf is $f_Z(z) = \frac{e^{-z^2/2}}{\sqrt{2\pi}}$. To find$f(y_i|\mathbf{w})$, we know that $y_i = \mathbf{x_i}^T\mathbf{w} + \sqrt{\lambda}N_i$, where $N_i \sim \mathcal{N}(0,1)$, so we'd have $\frac{y_i - \mathbf{x_i}^T\mathbf{w}}{\sqrt{\lambda}} \sim \mathcal{N}(0,1)$. Now, plugging in the pdf in the previous expressions we'd have

$$MAP = \underset{\mathbf{w}}{\operatorname{argmax}} \frac{e^{-\|\mathbf{w}\|^2/2}}{\sqrt{2\pi}} \prod_{i=1}^{n} \frac{e^{-(\frac{y_i - \mathbf{x_i}^T\mathbf{w}}{\sqrt{\lambda}})^2/2}}{\sqrt{2\pi}}$$

We can ignore multiplicative scaling constants (since they don't affect the value of the argmax). Also, since log is a monotonically increasing function, we can take log of both sides without affecting the argmax. Taking logs is useful since it allows us to turn products into sums. So we now have:

$$MAP = \underset{\mathbf{w}}{\operatorname{argmax}} -\frac{\|\mathbf{w}\|^2}{2} - \frac{1}{\lambda}\sum_i \frac{(y_i - \mathbf{x}_i^T\mathbf{w})^2}{2}$$

We can ignore scaling factors again, and arrange all the summation terms in a vector and taking the norm-squared. We can also turn argmax into argmin by negating the objective function:

$$MAP = \underset{\mathbf{w}}{\operatorname{argmin}} \|\mathbf{w}\|^2 + \frac{1}{\lambda}\|\mathbf{y} - X\mathbf{w}\|^2$$

Finally, we can multiply through by λ without changing the argmin since it is a positive constant:

$$MAP = \underset{\mathbf{w}}{\operatorname{argmin}} \lambda\|\mathbf{w}\|^2 + \|\mathbf{y} - X\mathbf{w}\|^2$$

And now it is the same form as the original Ridge Regression optimization problem.

(d) *Perspective 4: Fake Data.* Another way to interpret "ridge regression" is as the ordinary least squares for an augmented data set — i.e. adding a bunch of fake data points to our data. Consider the following augmented measurement vector $\hat{\mathbf{y}}$ and data matrix $\hat{\mathbf{X}}$:

$$\hat{\mathbf{y}} = \begin{bmatrix} \mathbf{y} \\ \mathbf{0}_d \end{bmatrix} \qquad \hat{\mathbf{X}} = \begin{bmatrix} X \\ \sqrt{\lambda}\mathbf{I}_d \end{bmatrix},$$

where $\mathbf{0}_d$ is the zero vector in $\mathbb{R}^d$ and $\mathbf{I}_d \in \mathbb{R}^{d\times d}$ is the identity matrix. **Show that the classical OLS optimization problem** $\operatorname{argmin}_{\mathbf{w}} \|\hat{\mathbf{y}} - \hat{\mathbf{X}}\mathbf{w}\|_2^2$ **has the same minimizer as (1).**

Solution: There are two easy ways of seeing the answer. The first is to look at the optimization problem itself and expand out the terms.

Recall that $\|\hat{\mathbf{y}} - \hat{\mathbf{X}}\mathbf{w}\|_2^2 = \sum_{i=1}^{n+d}(\hat{y}_i - \hat{\mathbf{x}}_i\mathbf{w})^2$ where $\hat{\mathbf{x}}_i$ are rows of $\hat{\mathbf{X}}$: the squared norm of the error is the sum of squared errors in individual coordinates. Our augmentation adds d more terms to that sum, which exactly give the ridge regularization. To see that we can write

$$\sum_{i=1}^{n}(\hat{y}_i - \hat{\mathbf{x}}_i\mathbf{w})^2 = \sum_{i=1}^{n}(y_i - \mathbf{x}_i\mathbf{w})^2 = \|\mathbf{y} - X\mathbf{w}\|_2^2$$

$$\sum_{i=n+1}^{d}(\hat{y}_i - \hat{\mathbf{x}}_i\mathbf{w})^2 = \sum_{i=n+1}^{d}(\sqrt{\lambda}w_i)^2 = \lambda\|\mathbf{w}\|_2^2$$

$$\sum_{i=1}^{n+d}(\hat{y}_i - \hat{\mathbf{x}}_i\mathbf{w})^2 = \|\mathbf{y} - X\mathbf{w}\|_2^2 + \lambda\|\mathbf{w}\|_2^2.$$

Alternatively, we can look at the solution and simplify it. We know that the solution to ordinary least squares for the augmented data is just

$$
\begin{aligned}
(\hat{X}^\top \hat{X})^{-1}\check{X}^\top \hat{\mathbf{y}} &= (\begin{bmatrix} X \\ \sqrt{\lambda}\mathbf{I}_d \end{bmatrix}^\top \begin{bmatrix} X \\ \sqrt{\lambda}\mathbf{I}_d \end{bmatrix})^{-1} \begin{bmatrix} X \\ \sqrt{\lambda}\mathbf{I}_d \end{bmatrix}^\top \begin{bmatrix} \mathbf{y} \\ \mathbf{0}_d \end{bmatrix} \\
&= (\begin{bmatrix} X^\top & \sqrt{\lambda}\mathbf{I}_d \end{bmatrix} \begin{bmatrix} X \\ \sqrt{\lambda}\mathbf{I}_d \end{bmatrix})^{-1} \begin{bmatrix} X^\top & \sqrt{\lambda}\mathbf{I}_d \end{bmatrix} \begin{bmatrix} \mathbf{y} \\ \mathbf{0}_d \end{bmatrix} \\
&= (X^\top X + \lambda \mathbf{I}_d)^{-1} X^\top \mathbf{y}
\end{aligned}
$$

Notice that this is the same as the solution for ridge regression. Either way, we get the desired result.

(e) *Perspective 5: Fake Features*. For this last interpretation, let's instead construct an augmented design matrix in the following way:

$$\check{\mathbf{X}} = [X \ \sqrt{\lambda}\mathbf{I_n}]$$

i.e. we stack X with $\sqrt{\lambda}\mathbf{I}_n$ horizontally. Now our problem is underdetermined: the new dimension $d+n$ is larger than the number of points n. Therefore, there are infinitely many values $\boldsymbol{\eta} \in \mathbb{R}^{d+n}$ for which $\check{\mathbf{X}}\boldsymbol{\eta} = \mathbf{y}$. We are interested in the **min-norm** solution, ie. the solution to

$$\underset{\boldsymbol{\eta}}{\operatorname{argmin}} \|\boldsymbol{\eta}\|_2^2 \text{ s.t. } \check{\mathbf{X}}\boldsymbol{\eta} = \mathbf{y}. \tag{2}$$

Show that this is yet another form of ridge regression and that the first d coordinates of $\boldsymbol{\eta}^*$ form the minimizer of (1).

Solution: Let's look inside the $d+n$ dimensional vector $\boldsymbol{\eta}$ by writing it as $\boldsymbol{\eta} = \begin{bmatrix} \mathbf{w} \\ \boldsymbol{\xi} \end{bmatrix}$. Here, $\mathbf{w}$ is d-dimensional and $\boldsymbol{\xi}$ is n-dimensional. Then (2) expands to

$$\underset{\mathbf{w},\boldsymbol{\xi}}{\operatorname{argmin}} \|\mathbf{w}\|_2^2 + \|\boldsymbol{\xi}\|_2^2 \text{ s.t. } X\mathbf{w} + \sqrt{\lambda}\boldsymbol{\xi} = \mathbf{y}.$$

The constraint just says that $\sqrt{\lambda}\boldsymbol{\xi} = \mathbf{y} - X\mathbf{w}$. In other words, $\sqrt{\lambda}\boldsymbol{\xi}$ is the classic residual. This yields $\boldsymbol{\xi} = \frac{1}{\sqrt{\lambda}}(\mathbf{y} - X\mathbf{w})$ and plugging that into the first part we get

$$\underset{\mathbf{w},\boldsymbol{\xi}}{\operatorname{argmin}} \|\mathbf{w}\|_2^2 + \frac{1}{\lambda}\|\mathbf{y} - X\mathbf{w}\|_2^2.$$

When considering whether the optimization problem is equivalent, we need to think about the minimizer and not the minimum itself. For this, we simply notice that scaling the objective by a constant factor doesn't change the minimizers and so:

$$\underset{\mathbf{w},\boldsymbol{\xi}}{\operatorname{argmin}} \|\mathbf{w}\|_2^2 + \frac{1}{\lambda}\|\mathbf{y} - X\mathbf{w}\|_2^2 = \underset{\mathbf{w},\boldsymbol{\xi}}{\operatorname{argmin}} \lambda\|\mathbf{w}\|_2^2 + \|\mathbf{y} - X\mathbf{w}\|_2^2$$

which is equivalent to (1).

(f) We know that the Moore-Penrose pseudo-inverse for an underdetermined system (wide matrix) is given by $A^\dagger = A^T(AA^T)^{-1}$, which corresponds to the min-norm solution for $A\boldsymbol{\eta} = \mathbf{z}$. That is, the optimization problem

$$\text{argmin}\,\|\boldsymbol{\eta}\|^2 \text{s.t.} A\boldsymbol{\eta} = \mathbf{z}$$

is solved by $\boldsymbol{\eta} = A^\dagger \mathbf{z}$. Let $\hat{\mathbf{w}}$ be the minimizer of (1).

Use the pseudo-inverse to show that solving to the optimization problem in (2) yields

$$\hat{\mathbf{w}} = X^T(XX^T + \lambda\mathbf{I})^{-1}\mathbf{y}$$

Then, show that this is equivalent to the standard formula for Ridge Regression

$$\hat{\mathbf{w}} = (X^TX + \lambda\mathbf{I})^{-1}X^T\mathbf{y}$$

Hint: It may be helpful to review Kernel Ridge Form.

Solution: First, we simply need to plug in our matrix into the pseudo-inverse formula provided and simplify

$$\hat{X}^\top(\hat{X}\hat{X}^\top)^{-1}\hat{\mathbf{y}} = \begin{bmatrix} X & \sqrt{\lambda}\mathbf{I}_n \end{bmatrix}^\top (\begin{bmatrix} X & \sqrt{\lambda}\mathbf{I}_n \end{bmatrix} \begin{bmatrix} X & \sqrt{\lambda}\mathbf{I}_n \end{bmatrix}^\top)^{-1}\mathbf{y}$$

$$\begin{bmatrix} \hat{\mathbf{w}} \\ \boldsymbol{\xi} \end{bmatrix} = \begin{bmatrix} X^\top \\ \sqrt{\lambda}\mathbf{I}_n \end{bmatrix} (\begin{bmatrix} X & \sqrt{\lambda}\mathbf{I}_n \end{bmatrix} \begin{bmatrix} X^\top \\ \sqrt{\lambda}\mathbf{I}_n \end{bmatrix})^{-1}\mathbf{y}$$

$$= \begin{bmatrix} X^\top \\ \sqrt{\lambda}\mathbf{I}_n \end{bmatrix} (XX^\top + \sqrt{\lambda}^2\mathbf{I})^{-1}\mathbf{y}$$

Looking at just the top d terms, we see $\hat{\mathbf{w}} = X^T(XX^T + \lambda\mathbf{I})^{-1}\mathbf{y}$ as desired.

Now, let's show the two forms of Ridge Regression are equivalent. That is, let's show

$$(X^TX + \lambda\mathbf{I})^{-1}X^T = X^T(XX^T + \lambda\mathbf{I})^{-1}$$

With the expression above, we can left-multiply both sides by $(X^TX + \lambda\mathbf{I})$ and right-multiply both sides by $(XX^T + \lambda\mathbf{I})$ to get

$$X^T(XX^T + \lambda\mathbf{I}) = (X^TX + \lambda\mathbf{I})X^T$$

And distributing the matrix multiplication we have

$$X^TXX^T + \lambda X^T = X^TXX^T + \lambda X^T$$

which we can see is always true as desired.

(g) We know that the solution to ridge regression (1) is given by $\hat{\mathbf{w}}_\mathbf{r} = (X^\top X + \lambda\mathbf{I})^{-1}X^\top\mathbf{y}$. **What happens when $\lambda \to \infty$?** It is for this reason that sometimes ridge regularization is referred to as "shrinkage."

Solution:

As $\lambda \to \infty$ the matrix $(X^\top X + \lambda\mathbf{I})^{-1}$ converges to the zero matrix, and so we have $\mathbf{w} = \mathbf{0}$.

(h) **What happens to the solution of ridge regression when you take the limit $\lambda \to 0$?** Consider both the cases when X is wide (underdetermined system) and X is tall (overdetermined system).

Solution:

When X is wide (underdetermined), we converge to the min-norm solution.

$$\mathbf{w}^* = X^T(XX^T)^{-1}\mathbf{y}$$

When X is tall (overdetermined), we converge to the OLS solution.

$$\mathbf{w}^* = (X^TX)^{-1}X^T\mathbf{y}$$

Both of these can be seen by using the relevant form of Ridge Regression and plugging in $\lambda = 0$ directly, or by using the SVD perspective.

3. General Case Tikhonov Regularization

Consider the optimization problem:

$$\min_{\mathbf{x}} ||W_1(A\mathbf{x} - \mathbf{b})||_2^2 + ||W_2(\mathbf{x} - \mathbf{c})||_2^2$$

Where W_1, A, and W_2 are matrices and $\mathbf{x}$, $\mathbf{b}$ and $\mathbf{c}$ are vectors. W_1 can be viewed as a generic weighting of the residuals and W_2 along with c can be viewed as a generic weighting of the parameters.

(a) **Solve this optimization problem manually** by expanding it out as matrix-vector products, setting the gradient to $\mathbf{0}$, and solving for $\mathbf{x}$.

Solution: Expand our objective function:

$$f(\mathbf{x}) = (A\mathbf{x} - \mathbf{b})^T W_1^T W_1(A\mathbf{x} - \mathbf{b}) + (\mathbf{x} - \mathbf{c})^T W_2^T W_2(\mathbf{x} - \mathbf{c})$$

$$f(\mathbf{x}) = \mathbf{x}^T A^T W_1^T W_1 A\mathbf{x} - 2\mathbf{b}^T W_1^T W_1 A\mathbf{x} + \mathbf{b}^T W_1^T W_1 \mathbf{b} + \mathbf{x}^T W_2^T W_2 \mathbf{x} - 2\mathbf{c}^T W_2^T W_2 \mathbf{x} + \mathbf{c}^T W_2^T W_2 \mathbf{c}$$

Now take gradients and set it to $\mathbf{0}$:

$$\nabla f = 2A^T W_1^T W_1 A\mathbf{x} - 2A^T W_1^T W_1 \mathbf{b} + 2W_2^T W_2 \mathbf{x} - 2W_2^T W_2 \mathbf{c} = 0$$

Isolating the $\mathbf{x}$ terms on one side, we have:

$$(A^T W_1^T W_1 A + W_2^T W_2)\mathbf{x} = A^T W_1^T W_1 \mathbf{b} + W_2^T W_2 \mathbf{c}$$

So we can solve to get

$$\mathbf{x} = (A^T W_1^T W_1 A + W_2^T W_2)^{-1}(A^T W_1^T W_1 \mathbf{b} + W_2^T W_2 \mathbf{c})$$

(b) **Construct an appropriate matrix C and vector d that allows you to rewrite this problem as**

$$\min_{x} \|C\mathbf{x} - \mathbf{d}\|^2$$

and use the OLS solution ($\mathbf{x}^* = (C^TC)^{-1}C^T\mathbf{d}$) to solve. Confirm your answer is in agreement with the previous part.

Solution: We can rewrite our problem in least-squares form using

$$C = \begin{bmatrix} W_1 A \\ W_2 \end{bmatrix}, \text{ and } \mathbf{d} = \begin{bmatrix} W_1 \mathbf{b} \\ W_2 \mathbf{c} \end{bmatrix}$$

Now, using least squares solution and solving, we get

$$\mathbf{x}^* = (C^TC)^{-1}C^T\mathbf{d} = (\begin{bmatrix} W_1A \\ W_2 \end{bmatrix}^\top \begin{bmatrix} W_1A \\ W_2 \end{bmatrix})^{-1} \begin{bmatrix} W_1A \\ W_2 \end{bmatrix}^\top \begin{bmatrix} W_1\mathbf{b} \\ W_2\mathbf{c} \end{bmatrix}$$

$$= (\begin{bmatrix} A^TW_1^T & W_2^T \end{bmatrix} \begin{bmatrix} W_1A \\ W_2 \end{bmatrix})^{-1} \begin{bmatrix} A^TW_1^T & W_2^T \end{bmatrix} \begin{bmatrix} W_1\mathbf{b} \\ W_2\mathbf{c} \end{bmatrix}$$

$$\mathbf{x}^* = (A^TW_1^TW_1A + W_2^TW_2)^{-1}(A^TW_1^TW_1\mathbf{b} + W_2^TW_2\mathbf{c})$$

Which is the same as the previous part, as desired.

(c) **Choose a W_1, W_2, and $\mathbf{c}$** such that this reduces to the simple case of ridge regression that you've seen in the previous problem, $\mathbf{x}^* = (A^TA + \lambda I)^{-1}A^T\mathbf{b}$.

Solution: This reduces to ridge regression when $W_1 = I$, $W_2 = \sqrt{\lambda}I$, and $\mathbf{c} = \mathbf{0}$. You can see this in both the optimization problem and the result.

4. Coding Fully Connected Networks

In this coding assignment, you will be building a fully-connected neural network from scratch using NumPy. You will have the choice between two options:

Use Google Colab (Recommended). Open this url and follow the instructions in the notebook.

Use a local Conda environment. Clone `https://github.com/Berkeley-CS182/cs182fa23_public`, `cd` to `hw/hw1/code/networks/problem`, and refer to `README.md` for further instructions.

Please submit the `.pdf` export of only the jupyter notebook when it is completed. In addition, please answer the following question:

(a) **Did you notice anything about the comparative difficulty of training the three-layer net vs training the five layer net?**

Solution: Training a five-layer neural network is more difficult and more sensitive to hyperparameters (initialization scale and learning rate).

5. Visualizing features from local linearization of neural nets

This problem expects you to modify the Jupyter Notebook you were given in the first discussion section for the course to allow the visualization of the effective "features" that correspond to the local linearization of the network in the neighborhood of the parameters.

We provide you with some starter code in the course repo, or you can use Google Colab. For this question, please submit the `.pdf` export of the jupyter notebook when it is completed. In addition, answer the questions below, including plots from the notebook where relevant.

(a) **Visualize the features corresponding to $\frac{\partial}{\partial w_i^{(1)}}y(x)$ and $\frac{\partial}{\partial b_i^{(1)}}y(x)$ where $w_i^{(1)}$ are the first hidden layer's weights and the $b_i^{(1)}$ are the first hidden layer's biases.** These derivatives should be evaluated at at least both the random initialization and the final trained network. When visualizing these features, plot them as a function of the scalar input x, the same way that the notebook plots the constituent "elbow" features that are the outputs of the penultimate layer.

Solution: See notebook.

(b) During training, we can imagine that we have a generalized linear model with a feature matrix corresponding to the linearized features corresponding to each learnable parameter. We know from our analysis of gradient descent, that the singular values and singular vectors corresponding to this feature matrix are important.

Use the SVD of this feature matrix to plot both the singular values and visualize the "principle features" that correspond to the d-dimensional singular vectors multiplied by all the features corresponding to the parameters.

(HINT: Remember that the feature matrix whose SVD you are taking has n rows where each row corresponds to one training point and d columns where each column corresponds to each of the learnable features. Meanwhile, you are going to be plotting/visualizing the "principle features" as functions of x even at places where you don't have training points.)

Solution: See notebook.

(c) Augment the jupyter notebook to add a second hidden layer of the same size as the first hidden layer, fully connected to the first hidden layer. **Allow the visualization of the features corresponding to the parameters in both hidden layers, as well as the "principle features" and the singular values.**

Solution: See notebook.

6. Homework Process and Study Group

Citing sources and collaborators are an important part of life, including being a student!
We also want to understand what resources you find helpful and how much time homework is taking, so we can change things in the future if possible.

(a) **What sources (if any) did you use as you worked through the homework?**

(b) **If you worked with someone on this homework, who did you work with?**
List names and student ID's. (In case of homework party, you can also just describe the group.)

(c) **Roughly how many total hours did you work on this homework?**

Contributors:

- Saagar Sanghavi.
- Alexander Tsigler.
- Anant Sahai.
- Jane Yu.
- Philipp Moritz.
- Soroush Nasiriany.
- Linyuan Gong.
- Luke Jaffe.
- Sheng Shen.