

1. Why Learning Rates Cannot be Too Big

Gradient descent has a speed limit, and the data sets it. The largest singular value of the problem decides how big a step you can take before the iterates blow up. One learning rate has to serve every direction at once, though, and the directions with small singular values want a much bigger step than the large ones can tolerate. By the end of this problem you will be able to write down that limit, choose the step size that balances the two, and see what the ratio of the largest to the smallest singular value costs you.

To understand the role of the learning rate, it is useful to understand it in the context of the simplest possible problem first.

Suppose that we want to solve the scalar equation

$$\sigma w = y \tag{1}$$

where we know that $\sigma > 0$. We proceed with an initial condition $w_0 = 0$ by using gradient descent to minimize the squared loss

$$L(w) = (y - \sigma w)^2 \tag{2}$$

which has a derivative with respect to the parameter w of $-2\sigma(y - \sigma w)$.

Gradient descent with a learning rate of η follows the recurrence-relation or discrete-time state evolution of:

$$\begin{aligned} w_{t+1} &= w_t + 2\eta\sigma(y - \sigma w_t) \\ &= (1 - 2\eta\sigma^2)w_t + 2\eta\sigma y. \end{aligned} \tag{3}$$

(a) **For what values of learning rate $\eta > 0$ is the recurrence (3) stable?**

(HINT: Remember the role of the unit circle in determining the stability or instability of such recurrences. If you keep taking higher and higher positive integer powers of a number, what does that number have to be like for this to converge?)

Solution: We can rewrite the update rule as

$$\begin{aligned} w_{t+1} - \frac{y}{\sigma} &= (1 - 2\eta\sigma^2)(w_t - \frac{y}{\sigma}) \\ w_{t+1} &= \frac{y}{\sigma} + (1 - 2\eta\sigma^2)^{t+1}(w_0 - \frac{y}{\sigma}) \end{aligned} \tag{4}$$

To make the recurrence (3) stable with $\eta > 0$, we need $|1 - 2\eta\sigma^2| < 1$. This gives $\eta < \frac{1}{\sigma^2}$.

(b) The previous part gives you an upper bound for the learning rate η that depends on σ beyond which we cannot safely go. **If η is below that upper bound, how fast does w_t converge to its final solution $w^* = \frac{y}{\sigma}$? i.e. if we wanted to get within a factor $(1 - \epsilon)$ of w^* , how many iterations t would we need?**

(HINT: The absolute value of the error of current w to the optimality might help.)

Solution:

$$|w_T - w^*| < \epsilon |w^*| \quad (5)$$

Use the derived update rule in (4). We have

$$\begin{aligned} |w_T - \frac{y}{\sigma}| &< \epsilon \left| \frac{y}{\sigma} \right| \\ |(1 - 2\eta\sigma^2)^T| &< \epsilon \\ T &> \frac{\log(\epsilon)}{\log(|1 - 2\eta\sigma^2|)}, \end{aligned} \quad (6)$$

- (c) Suppose that we now have a vector problem where we have two parameters $w[1], w[2]$. One with a large σ_ℓ and the other with a tiny σ_s . i.e. $\sigma_\ell \gg \sigma_s$ and we have the vector equation we want to solve:

$$\begin{bmatrix} \sigma_\ell & 0 \\ 0 & \sigma_s \end{bmatrix} \begin{bmatrix} w[1] \\ w[2] \end{bmatrix} = \begin{bmatrix} y[1] \\ y[2] \end{bmatrix}. \quad (7)$$

We use gradient descent with a single learning rate η to solve this problem starting from an initial condition of $\mathbf{w} = \mathbf{0}$.

For what learning rates $\eta > 0$ will we converge? Which of the two σ_i is limiting our learning rate?

Solution: Similarly, we can rewrite the loss function and update rule w.r.t the vector form.

$$\begin{aligned} L(\mathbf{w}) &= \|\mathbf{y} - \Sigma\mathbf{w}\|^2, \Sigma = \begin{bmatrix} \sigma_\ell & 0 \\ 0 & \sigma_s \end{bmatrix} \\ \nabla_{\mathbf{w}} L(\mathbf{w}) &= 2(\Sigma^2\mathbf{w} - \Sigma\mathbf{y}) \\ \mathbf{w}_{t+1} &= (I - 2\eta\Sigma^2)\mathbf{w}_t + 2\eta\Sigma\mathbf{y} \end{aligned} \quad (8)$$

To ensure the convergence, we need

$$\begin{cases} |1 - 2\eta\sigma_\ell^2| < 1 \\ |1 - 2\eta\sigma_s^2| < 1 \end{cases} \quad (9)$$

$$\eta < \min\left(\frac{1}{\sigma_\ell^2}, \frac{1}{\sigma_s^2}\right) = \frac{1}{\sigma_\ell^2} \quad (10)$$

- (d) **For the previous problem, depending on $\eta, \sigma_\ell, \sigma_s$, which of the two dimensions is converging faster and which is converging slower?**

Solution: We can rewrite the update rule w.r.t each dimension, this gives

$$\begin{aligned} w[1]_t &= \frac{y[1]}{\sigma_\ell} + (1 - 2\eta\sigma_\ell^2)^t \left(-\frac{y[1]}{\sigma_\ell}\right) \\ w[2]_t &= \frac{y[2]}{\sigma_s} + (1 - 2\eta\sigma_s^2)^t \left(-\frac{y[2]}{\sigma_s}\right) \end{aligned}$$

This faster convergence dimension is $\min(|1 - 2\eta\sigma_l^2|, |1 - 2\eta\sigma_s^2|)$

- (e) The speed of convergence overall will be dominated by the slower of the two. **For what value of η will we get the fastest overall convergence to the solution?**

Solution: Recall that the minimum of the pointwise maximum of two functions occurs at a point where both functions are equal. Thus, the fastest convergence is achieved when $|1 - 2\eta\sigma_l^2| = |1 - 2\eta\sigma_s^2|$, and since $\sigma_l \geq \sigma_s$, there are two solutions, one is the trivial $\eta = 0$, and the other is

$$\eta = \frac{1}{\sigma_l^2 + \sigma_s^2}$$

See Figure 1 for a picture.

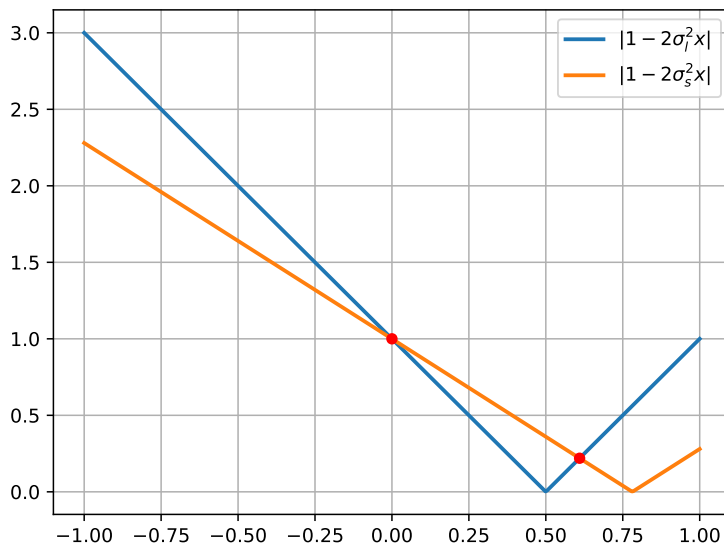


Figure 1: Plot of $|1 - 2\eta\sigma_l^2|, |1 - 2\eta\sigma_s^2|$, showing their two intersections. One is the trivial $\eta = 0$, and the other is the nontrivial $\eta = \frac{1}{\sigma_l^2 + \sigma_s^2}$.

- (f) Comment on what would happen if we had more parallel problems with σ_i that all were in between σ_l and σ_s ? **Would they influence the choice of possible learning rates or the learning rate with the fastest convergence?**

Solution: The bounds on the learning rate should still be the same. For maximal convergence, the largest and smallest values of sigma are the two that cause gradient descent to take the longest if we poorly choose a learning rate.

- (g) Using what you know about the SVD, **how is the simple scalar and parallel scalar problem analysis above relevant to solving general least-squares problems of the form $Xw \approx y$ using gradient descent?**

Solution: We can think of SVD as a change of bases into and then back from a coordinate system where X instead is just diagonal, this will directly connect to the same problem with lots of parallel scalar problems.

2. Where Gradient Descent Cannot Go

When a model has more parameters than training points, infinitely many weight vectors fit the data perfectly, and the loss is the same at all of them. Gradient descent still ends up at one of them. This problem works out which one. Each step is a combination of the training inputs, so the iterates never leave $\mathbf{w}_0$ plus the row space of X , and whatever $\mathbf{w}_0$ had in the orthogonal directions stays there. The limit is the interpolating solution closest to $\mathbf{w}_0$. From the origin that is the minimum-norm solution. From anywhere else it is not.

Let $X \in \mathbb{R}^{n \times d}$ with $n < d$ and full row rank n , so $X\mathbf{w} = \mathbf{y}$ has infinitely many solutions. Run gradient descent on $L(\mathbf{w}) = \|\mathbf{y} - X\mathbf{w}\|_2^2$ from a starting point $\mathbf{w}_0$:

$$\mathbf{w}_{t+1} = \mathbf{w}_t + 2\eta X^\top \mathbf{r}_t, \quad \mathbf{r}_t := \mathbf{y} - X\mathbf{w}_t. \quad (11)$$

Recall that $\mathbb{R}^d$ is the orthogonal direct sum of the row space $\text{range}(X^\top)$ and the null space $\text{null}(X)$, so every $\mathbf{w}$ splits uniquely as $\mathbf{w} = \mathbf{w}^\parallel + \mathbf{w}^\perp$ with $\mathbf{w}^\parallel \in \text{range}(X^\top)$ and $\mathbf{w}^\perp \in \text{null}(X)$.

- (a) **Show that every gradient descent step lies in $\text{range}(X^\top)$, so that $\mathbf{w}_t \in \mathbf{w}_0 + \text{range}(X^\top)$ for all t . Conclude that the null-space component never moves: $\mathbf{w}_t^\perp = \mathbf{w}_0^\perp$ for all t .**

Solution: The increment in (11) is $\mathbf{w}_{t+1} - \mathbf{w}_t = 2\eta X^\top \mathbf{r}_t$, which is $X^\top$ applied to a vector and so lies in $\text{range}(X^\top)$ whatever $\mathbf{r}_t$ is. Summing,

$$\mathbf{w}_t = \mathbf{w}_0 + X^\top \left(2\eta \sum_{k=0}^{t-1} \mathbf{r}_k \right) \in \mathbf{w}_0 + \text{range}(X^\top),$$

since $\text{range}(X^\top)$ is a subspace.

Let $P_\perp$ project onto $\text{null}(X)$. As $\text{null}(X) \perp \text{range}(X^\top)$ we have $P_\perp X^\top \mathbf{v} = \mathbf{0}$ for every $\mathbf{v}$, so applying $P_\perp$ to (11) gives $\mathbf{w}_{t+1}^\perp = \mathbf{w}_t^\perp$, and by induction $\mathbf{w}_t^\perp = \mathbf{w}_0^\perp$.

- (b) Let $X = U\Sigma V^\top$ be the full SVD, with $\sigma_1 \geq \dots \geq \sigma_n > 0$ and $\mathbf{v}_1, \dots, \mathbf{v}_d$ the columns of V .

Which $\mathbf{v}_i$ span $\text{null}(X)$? How many of the d coordinates $\mathbf{v}_i^\top \mathbf{w}_t$ can gradient descent change, and how many are fixed by $\mathbf{w}_0$?

Solution: The last $d - n$ columns $\mathbf{v}_{n+1}, \dots, \mathbf{v}_d$ span $\text{null}(X)$: they have zero singular values, so $X\mathbf{v}_i = \mathbf{0}$, and there are $d - n = \dim \text{null}(X)$ of them. The first n span $\text{range}(X^\top)$.

By part (a), $\mathbf{v}_i^\top \mathbf{w}_t = \mathbf{v}_i^\top \mathbf{w}_0$ for every $i > n$ and every t . So gradient descent sets only the n coordinates along $\mathbf{v}_1, \dots, \mathbf{v}_n$, and the other $d - n$ are fixed by the initialization.

- (c) Suppose gradient descent converges to some $\mathbf{w}_\infty$ with $X\mathbf{w}_\infty = \mathbf{y}$. (You may assume η is small enough for this; the learning-rate problem says how small.)

Show that $\mathbf{w}_\infty$ is the unique solution of

$$\min_{\mathbf{w}} \|\mathbf{w} - \mathbf{w}_0\|_2 \quad \text{subject to} \quad X\mathbf{w} = \mathbf{y}. \quad (12)$$

(Hint: for any other interpolating $\mathbf{w}$, split $\mathbf{w} - \mathbf{w}_0 = (\mathbf{w} - \mathbf{w}_\infty) + (\mathbf{w}_\infty - \mathbf{w}_0)$. Which subspace does each term lie in?)

Solution: Let $X\mathbf{w} = \mathbf{y}$ and write $\mathbf{w} - \mathbf{w}_0 = \mathbf{a} + \mathbf{b}$ with $\mathbf{a} = \mathbf{w} - \mathbf{w}_\infty$ and $\mathbf{b} = \mathbf{w}_\infty - \mathbf{w}_0$. Both $\mathbf{w}$ and $\mathbf{w}_\infty$ interpolate, so $X\mathbf{a} = \mathbf{0}$ and $\mathbf{a} \in \text{null}(X)$. By part (a) every iterate satisfies $\mathbf{w}_t - \mathbf{w}_0 \in \text{range}(X^\top)$, and a subspace of a finite-dimensional space is closed, so the limit does too:

$\mathbf{b} \in \text{range}(X^\top)$. The two are orthogonal, so

$$\|\mathbf{w} - \mathbf{w}_0\|_2^2 = \|\mathbf{a}\|_2^2 + \|\mathbf{b}\|_2^2 \geq \|\mathbf{w}_\infty - \mathbf{w}_0\|_2^2,$$

with equality only when $\mathbf{a} = \mathbf{0}$, i.e. $\mathbf{w} = \mathbf{w}_\infty$. So $\mathbf{w}_\infty$ is the unique minimizer.

- (d) **Specialize part (c) to $\mathbf{w}_0 = \mathbf{0}$: write $\mathbf{w}_\infty$ in closed form and name the estimator from the earlier SVD problem that it equals. Then say what changes if $\mathbf{w}_0 \neq \mathbf{0}$. Is the limit still the minimum-norm solution?**

Solution: With $\mathbf{w}_0 = \mathbf{0}$, (12) reads $\min \|\mathbf{w}\|_2$ subject to $X\mathbf{w} = \mathbf{y}$, so

$$\mathbf{w}_\infty = X^\top (XX^\top)^{-1} \mathbf{y} = X^\dagger \mathbf{y},$$

the minimum-norm solution built from the pseudo-inverse in the earlier SVD problem. In singular-vector coordinates, writing $\mathbf{u}_i$ for the columns of U , this is $\mathbf{v}_i^\top \mathbf{w}_\infty = \mathbf{u}_i^\top \mathbf{y} / \sigma_i$ for $i \leq n$ and 0 for $i > n$.

For $\mathbf{w}_0 \neq \mathbf{0}$ this fails. Part (a) freezes $\mathbf{w}^\perp$ at $\mathbf{w}_0^\perp$, so the limit keeps whatever null-space component the initialization had, and by part (c) it is the interpolating solution closest to $\mathbf{w}_0$, not the one closest to the origin. The two agree exactly when $\mathbf{w}_0^\perp = \mathbf{0}$, i.e. when the initialization has no null-space component; starting at the origin is the simplest way to arrange that.

3. An Alternate MAP Interpretation of Ridge Regression

Ridge regression has two formulas that look nothing alike. One inverts a $d \times d$ matrix and the other inverts an $n \times n$ matrix, so when you have far more features than training points, only one of them is practical to compute. This problem shows they give the same estimator. The proof does not push matrices around. It reads ridge as a MAP estimate under a Gaussian prior, and the second formula falls out of the usual formula for conditioning one jointly Gaussian vector on another.

Consider the Ridge Regression estimator,

$$\underset{\mathbf{w}}{\operatorname{argmin}} \|X\mathbf{w} - \mathbf{y}\|_2^2 + \lambda \|\mathbf{w}\|^2$$

We know this is solved by

$$\hat{\mathbf{w}} = (X^T X + \lambda I)^{-1} X^T \mathbf{y} \quad (13)$$

An alternate form of the Ridge Regression solution (often called the Kernel Ridge form) is given by

$$\hat{\mathbf{w}} = X^T (X X^T + \lambda I)^{-1} \mathbf{y}. \quad (14)$$

We know that 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 $\mathbf{Y} = X\mathbf{W} + \sqrt{\lambda} \mathbf{N}$, and then we know that when we try to maximize the log likelihood we end up minimizing

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

The underlying probability space is that defined by the d iid standard normals that define the $\mathbf{W}$ and the n iid standard normals that give the n different N_i on the training points. Note that the X matrix, whose rows consist of the n different inputs for the n different training points, is not random.

Based on what we know about joint normality, it is clear that the random Gaussian vectors $\mathbf{W}$ and $\mathbf{Y}$ are jointly normal. **Use the following facts to show that the two forms of solution are identical.**

- (13) is the MAP estimate for $\mathbf{W}$ given an observation $\mathbf{Y} = \mathbf{y}$ (we showed this in HW0, and in discussion section)
- For jointly normal random variables, when you condition one set of variables on the values for the others, the resulting conditional distribution is still normal.
- A normal random variable has its density maximized at its mean.
- For jointly normal random vectors that are zero mean, the formula for conditional expectation is

$$E[\mathbf{W}|\mathbf{Y} = \mathbf{y}] = \Sigma_{WY} \Sigma_{YY}^{-1} \mathbf{y} \quad (15)$$

where the Σ_{YY} is the covariance $E[\mathbf{Y}\mathbf{Y}^T]$ of $\mathbf{Y}$ and $\Sigma_{WY} = E[\mathbf{W}\mathbf{Y}^T]$ is the appropriate cross-covariance of $\mathbf{W}$ and $\mathbf{Y}$.

Solution:

We are given that (13) is $\text{MAP}(\mathbf{w} | \mathbf{Y} = \mathbf{y})$. Let's try to find $\text{MAP}(\mathbf{w} | \mathbf{Y} = \mathbf{y})$ in a different way.

We can condition the random variables $\mathbf{W}$ on the values for $\mathbf{Y} = \mathbf{y}$, and we know the resulting conditional distribution $\mathbf{W}|\mathbf{Y} = \mathbf{y}$ is still normal.

Since normal random variables has density maximized at the mean, the MAP estimate is equivalent to the conditional expectation $E[\mathbf{W}|\mathbf{Y} = \mathbf{y}]$. We are given the formula to calculate the conditional expectation.

Before we plug into the formula, let's calculate $\Sigma_{WY} = E[\mathbf{W}\mathbf{Y}^T]$. Using $\mathbf{Y} = X\mathbf{W} + \mathbf{N}$, we have:

$$\Sigma_{WY} = E[\mathbf{W}(X\mathbf{W} + \sqrt{\lambda}\mathbf{N})^T]$$

Take transposes and distribute, apply linearity of expectation

$$\Sigma_{WY} = E[\mathbf{W}\mathbf{W}^T X^T] + \sqrt{\lambda}E[\mathbf{W}\mathbf{N}^T]$$

Since $\mathbf{W}$ and $\mathbf{N}$ are uncorrelated, and X does not involve any randomness, we can write

$$\Sigma_{WY} = E[\mathbf{W}\mathbf{W}^T] X^T + \sqrt{\lambda}E[\mathbf{W}]E[\mathbf{N}^T]$$

Use the fact that $E[\mathbf{W}\mathbf{W}^T]$ is the covariance matrix of $\mathbf{W}$, which we are given is the identity. Also use the fact that $\mathbf{W}$ and $\mathbf{N}$ are zero-means.

$$\Sigma_{WY} = I X^T + 0 = X^T$$

Now let's find $\Sigma_{YY} = E[\mathbf{Y}\mathbf{Y}^T]$.

$$\Sigma_{YY} = E[(X\mathbf{W} + \mathbf{N})(X\mathbf{W} + \mathbf{N})^T]$$

Take transposes and distribute

$$\Sigma_{YY} = E[X\mathbf{W}\mathbf{W}^T X^T] + \sqrt{\lambda}^2 E[\mathbf{N}\mathbf{N}^T] + \sqrt{\lambda}E[X\mathbf{W}\mathbf{N}^T] + \sqrt{\lambda}E[\mathbf{N}\mathbf{W}^T X^T]$$

The cross-terms are 0 since the random vectors are zero-meaned. With some manipulation, we have

$$\Sigma_{YY} = XE[\mathbf{W}\mathbf{W}^T]X^T + \lambda I$$

$$\Sigma_{YY} = XX^T + \lambda I$$

Finally, plugging in Σ_{WY} and Σ_{YY} to the formula for $E[\mathbf{W}|\mathbf{Y} = \mathbf{y}]$, we get $MAP(\mathbf{w}|\mathbf{Y} = \mathbf{y})$ as

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

as desired.

4. Early Stopping is Ridge Regression in Disguise

In HW0 you saw several ways to think about ridge regression. Here is one more. Run ordinary gradient descent on the least-squares loss, with no penalty term, and stop before it has converged. We will show that after t steps the weights look like the ridge solution with λ roughly $\frac{1}{2\eta t}$, so stopping earlier acts like using a larger λ . We will also see when this stops being true.

Let $X \in \mathbb{R}^{n \times d}$ have full SVD $X = U\Sigma V^T$ with singular values $\sigma_1 \geq \dots \geq \sigma_r > 0$, where r is the rank of X . Gradient descent starts at $\mathbf{w}_0 = \mathbf{0}$ and takes steps

$$\mathbf{w}_{t+1} = \mathbf{w}_t + 2\eta X^T(\mathbf{y} - X\mathbf{w}_t) \quad (16)$$

on the loss $L(\mathbf{w}) = \|\mathbf{y} - X\mathbf{w}\|_2^2$. Write $\tilde{\mathbf{y}} = U^T\mathbf{y}$ and $\tilde{\mathbf{w}}_t = V^T\mathbf{w}_t$ for the coordinates of $\mathbf{y}$ and $\mathbf{w}_t$ in the singular vector bases.

Recall from HW0 that the ridge solution $\mathbf{w}_{\text{ridge}}(\lambda) = (X^T X + \lambda I)^{-1} X^T \mathbf{y}$ has coordinates $\sigma_i \tilde{y}_i / (\sigma_i^2 + \lambda)$ in the V basis. We will write this as

$$[V^T \mathbf{w}_{\text{ridge}}(\lambda)]_i = s_i^{\text{ridge}}(\lambda) \cdot \frac{\tilde{y}_i}{\sigma_i} \quad \text{with} \quad s_i^{\text{ridge}}(\lambda) = \frac{\sigma_i^2}{\sigma_i^2 + \lambda} \quad \text{for } i \leq r, \quad \text{and } 0 \text{ for } i > r. \quad (17)$$

The number $\tilde{y}_i / \sigma_i$ is the least-squares answer along $\mathbf{v}_i$; it is the i -th coordinate of the minimum-norm solution from Problem 2. Ridge multiplies it by $s_i^{\text{ridge}}(\lambda)$, which lies between 0 and 1. We will call a multiplier like this a *shrinkage factor*. It is close to 1 when σ_i^2 is much larger than λ , so those directions are kept, and close to 0 when σ_i^2 is much smaller than λ , so those directions are mostly thrown away.

(a) **Show that in the V basis the update (16) becomes d separate scalar recurrences,**

$$[\tilde{\mathbf{w}}_{t+1}]_i = (1 - 2\eta\sigma_i^2)[\tilde{\mathbf{w}}_t]_i + 2\eta\sigma_i\tilde{y}_i \quad \text{for } i \leq r, \quad [\tilde{\mathbf{w}}_{t+1}]_i = [\tilde{\mathbf{w}}_t]_i \quad \text{for } i > r. \quad (18)$$

This is the same change of coordinates you used in Problem 1, applied to the whole matrix at once.

Solution: Substitute the SVD into (16) and multiply on the left by V^T . Using $V^T V = I$ and $U^T U = I$,

$$\begin{aligned} V^T \mathbf{w}_{t+1} &= V^T \mathbf{w}_t + 2\eta V^T V \Sigma^T U^T (\mathbf{y} - U \Sigma V^T \mathbf{w}_t) \\ &= \tilde{\mathbf{w}}_t + 2\eta \Sigma^T (\tilde{\mathbf{y}} - \Sigma \tilde{\mathbf{w}}_t). \end{aligned}$$

Σ is zero off the diagonal, so the i -th entry of $\Sigma^T (\tilde{\mathbf{y}} - \Sigma \tilde{\mathbf{w}}_t)$ is $\sigma_i(\tilde{y}_i - \sigma_i[\tilde{\mathbf{w}}_t]_i)$ for $i \leq r$ and 0 for $i > r$. Reading off coordinate i ,

$$[\tilde{\mathbf{w}}_{t+1}]_i = [\tilde{\mathbf{w}}_t]_i + 2\eta\sigma_i(\tilde{y}_i - \sigma_i[\tilde{\mathbf{w}}_t]_i) = (1 - 2\eta\sigma_i^2)[\tilde{\mathbf{w}}_t]_i + 2\eta\sigma_i\tilde{y}_i,$$

and for $i > r$ the increment is zero.

(b) Assume that

$$0 < 2\eta\sigma_i^2 < 1 \quad \text{for every } i \leq r. \quad (19)$$

This is the condition from Problem 1 under which each recurrence in (18) converges.

Solve (18) starting from $\mathbf{w}_0 = \mathbf{0}$ and show that

$$[\tilde{\mathbf{w}}_t]_i = s_i^{\text{GD}}(t) \cdot \frac{\tilde{y}_i}{\sigma_i} \quad \text{with} \quad s_i^{\text{GD}}(t) = 1 - (1 - 2\eta\sigma_i^2)^t \quad \text{for } i \leq r, \quad \text{and } 0 \text{ for } i > r. \quad (20)$$

Then show that $s_i^{\text{GD}}(t)$ lies between 0 and 1, increases with t , and tends to 1 as $t \rightarrow \infty$. So gradient descent also multiplies the least-squares answer along $\mathbf{v}_i$ by a shrinkage factor, and this factor grows toward 1 as training continues.

Solution: Coordinate i follows $a_{t+1} = c a_t + b$ with $c = 1 - 2\eta\sigma_i^2$ and $b = 2\eta\sigma_i\tilde{y}_i$. Unrolling from $a_0 = 0$,

$$[\tilde{\mathbf{w}}_t]_i = b \sum_{k=0}^{t-1} c^k = 2\eta\sigma_i\tilde{y}_i \cdot \frac{1 - (1 - 2\eta\sigma_i^2)^t}{2\eta\sigma_i^2} = (1 - (1 - 2\eta\sigma_i^2)^t) \frac{\tilde{y}_i}{\sigma_i}.$$

For $i > r$ the coordinate never leaves 0.

By (19), $0 < 1 - 2\eta\sigma_i^2 < 1$, so $(1 - 2\eta\sigma_i^2)^t$ is positive, decreases with t , and tends to 0. Hence $s_i^{\text{GD}}(t)$ lies in $[0, 1)$, increases with t , and tends to 1. In the limit every coordinate with $i \leq r$ reaches $\tilde{y}_i/\sigma_i$ and the rest stay at 0, which is the minimum-norm solution $X^\dagger \mathbf{y}$ from Problem 2.

(c) Now compare the two shrinkage factors, $s_i^{\text{ridge}}(\lambda)$ from (17) and $s_i^{\text{GD}}(t)$ from (20).

Show that $s_i^{\text{GD}}(t) \geq 1 - e^{-2\eta t \sigma_i^2}$. Then set $\lambda = \frac{1}{2\eta t}$ and compute both factors for a direction with $\sigma_i^2 = \lambda$. (Hint: $1 - x \leq e^{-x}$, and $(1 - 1/t)^t \rightarrow e^{-1}$.) Running gradient descent for t steps acts like ridge regression with which value of λ ? What happens to this value as t grows?

Solution: With $x = 2\eta\sigma_i^2 \in (0, 1)$ we have $0 < 1 - x \leq e^{-x}$, and raising both sides to the t -th power keeps the inequality, so

$$(1 - 2\eta\sigma_i^2)^t \leq e^{-2\eta t \sigma_i^2} \implies s_i^{\text{GD}}(t) = 1 - (1 - 2\eta\sigma_i^2)^t \geq 1 - e^{-2\eta t \sigma_i^2}.$$

Now take $\lambda = \frac{1}{2\eta t}$ and a direction with $\sigma_i^2 = \lambda$. Ridge gives

$$s_i^{\text{ridge}}(\lambda) = \frac{\lambda}{\lambda + \lambda} = \frac{1}{2},$$

and for gradient descent $2\eta\sigma_i^2 = 1/t$, so

$$s_i^{\text{GD}}(t) = 1 - \left(1 - \frac{1}{t}\right)^t \xrightarrow{t \rightarrow \infty} 1 - e^{-1} \approx 0.63.$$

So at $\sigma_i^2 = \frac{1}{2\eta t}$ both factors are of order one. For $\sigma_i^2 \gg \frac{1}{2\eta t}$ both are close to 1: ridge directly, and gradient descent by the bound just proved. For $\sigma_i^2 \ll \frac{1}{2\eta t}$ both are close to 0: ridge directly, and gradient descent because Bernoulli's inequality $(1 - x)^t \geq 1 - tx$ gives $s_i^{\text{GD}}(t) \leq 2\eta t \sigma_i^2$. Running

gradient descent for t steps therefore acts like ridge regression with

$$\lambda \approx \frac{1}{2\eta t}.$$

This value shrinks like $1/t$. Every additional step weakens the regularization. Stopping early corresponds to a large λ ; training to convergence corresponds to $\lambda \rightarrow 0$, which matches part (b), where the limit was the unregularized min-norm solution.

- (d) Now suppose the condition (19) fails for some direction i , that is, $2\eta\sigma_i^2 > 1$.

Show that $s_i^{\text{GD}}(t)$ is then larger than 1 for every odd t . A shrinkage factor larger than 1 means the estimate along $\mathbf{v}_i$ is larger in magnitude than the least-squares answer $\tilde{y}_i/\sigma_i$. **Can this happen with ridge regression for some choice of $\lambda > 0$? Show why or why not.**

Solution: Write $c_i = 1 - 2\eta\sigma_i^2$, so $s_i^{\text{GD}}(t) = 1 - c_i^t$. If $2\eta\sigma_i^2 > 1$ then $c_i < 0$, so $c_i^t < 0$ for every odd t , and then $s_i^{\text{GD}}(t) = 1 - c_i^t > 1$. With $c_i = -\frac{1}{2}$, for instance, the factor takes the values 1.5, 0.75, 1.125, $\dots$, jumping back and forth around 1 instead of rising toward it. If moreover $2\eta\sigma_i^2 > 2$ then $|c_i| > 1$ and the iterates diverge, which is the failure from Problem 1.

This cannot happen with ridge. For any $\lambda > 0$,

$$1 - s_i^{\text{ridge}}(\lambda) = \frac{\lambda}{\sigma_i^2 + \lambda} > 0,$$

so $s_i^{\text{ridge}}(\lambda) < 1$ for every λ and every direction. Ridge always returns a shrunk copy of the least-squares answer. Gradient descent does too, but only while (19) holds; with too large a learning rate it can return an estimate larger than the least-squares one.

5. Coding: ReLU Networks and Their Elbows

A ReLU hidden layer is a pile of elbows. Each unit adds one kink to the function, and where that kink sits is a parameter like any other. When you train every layer, the elbows slide toward the places where the target function bends. When you train only the output layer, they stay wherever the initialization dropped them. Watch for that difference in this notebook, because it is what feature learning looks like in this setting.

Work through `q_relu_vis.ipynb`. It fits a one-hidden-layer ReLU network to a piecewise-linear target at several hidden widths, under three regimes: SGD on the output layer only, SGD on all layers, and a closed-form ridge regression on the hidden-layer features. Training takes a few minutes, so start it running and read ahead while you wait. Submit a `.pdf` export of the completed notebook along with your write-up, and answer the questions below.

- (a) After enough training, the test error sits *below* the train error, the opposite of what we usually expect. **What causes this?**

Solution: The training data carry additive noise, $y = f(x) + \epsilon$, while the test data are generated noise-free from f . The train error therefore includes an irreducible noise floor that the test error does not, so a well-fit model shows lower test error.

- (b) **How does the hidden layer width change the learned function and the test error? If you could place the elbows by hand, where would you put them?**

Solution: Wider hidden layers give lower test error and a visibly better fit, with no overfitting at the widths in the notebook. Since each ReLU unit contributes one kink, the useful place for the elbows is at the kinks of the target function. One elbow per kink suffices, and elbows elsewhere contribute little.

- (c) **Compare the three regimes on final test error and on how long each takes. In which of them do the elbows move, and why does that matter?**

Solution: Ridge on the hidden features is the fastest, being a closed-form solve rather than an iterative one. Output-layer-only SGD is next. All-layer SGD is slowest, since it optimizes far more parameters with noisier gradients, which is why the notebook gives it a smaller learning rate. The elbows move only when all layers are trained; output-layer-only training leaves them wherever initialization put them. That matters because elbows sitting at the target's kinks make the remaining linear problem easy, which is why all-layer training can beat a fixed random featurization at the same width. The exact test-error ordering depends on width, so report the numbers you actually measured.

Contributors:

- Anant Sahai.
- Sheng Shen.
- Gireeja Ranade.
- Matteo Guarrera.
- Mert Cemri.
- Sizhe Chen.
- Saagar Sanghavi.
- Kumar Krishna Agrawal.
- Kevin Li.