SECONDARY GRADIENT DESCENT IN HIGHER CODIMENSION

Y. COOPER

Abstract. In this paper, we analyze discrete gradient descent and $\epsilon$-noisy gradient descent on a special but important class of functions. We find that when used to minimize a function $L : \mathbb{R}^n \to \mathbb{R}$ in this class we consider, discrete gradient descent can exhibit strikingly different behavior from continuous gradient descent. On long time scales, discrete gradient descent and continuous gradient descent tend toward different global minima of $L$. Discrete gradient descent preferentially finds global minima at which the graph of the function $L$ is shallowest, while gradient flow shows no such preference.

1. Introduction

In [C18], we analyzed the behavior of discrete gradient descent on functions of the form $L = |f|$ and the modified $\epsilon$-noisy gradient descent on functions of the form $L = f^2$. There, we found that on long time scales, discrete gradient descent exhibits strikingly different behavior from continuous gradient descent, also called gradient flow. When used to minimize the same function, discrete gradient descent will consistently find different minima from continuous gradient descent.

In this paper, we extend that work significantly by studying the behavior of discrete gradient descent on a much larger class of functions - functions that are of the form $L_1$ or $L_2$, as defined in Section 3. We analyze the behavior of discrete gradient descent on functions of the form $L_1$ and the modified $\epsilon$-noisy gradient descent on functions of the form $L_2$. For the functions studied in [C18], the locus $M$ of global minima of $L$ has codimension 1. For the functions studied in this paper, $M$ can have arbitrary codimension. This makes the analysis considerably more difficult, and as a result we make an additional assumption in this paper about the geometry of the submanifolds $M_i$ that was not necessary in the codimension 1 case.

In the general setting, we continue to see that discrete gradient descent displays strikingly different behavior from gradient flow. In the early stages the two processes behave similarly. Under continuous gradient descent, if the flow line reaches $M$, the process ends there. However, under discrete gradient descent on $L_1$ or $\epsilon$-noisy gradient descent on $L_2$, if the sequence reaches near $M$ and the algorithm continues running, a second phase of the process emerges. During this phase, the sequence proceeds along the submanifold $M$, effectively minimizing the function $K = tr(H(L_2))|_M$.

This has the effect that on long time scales, discrete gradient descent and continuous gradient descent tend toward different minima of $L$. Discrete gradient descent preferentially finds global minima at which the function $L$ is shallowest whereas gradient flow shows no such preference.

1.1. Acknowledgements. The author would like to thank several people who have been instrumental to this project. Avi Wigderson for posing a question about the distribution of linear neural

E-mail address: yaim@math.ias.edu.
Date: September 17, 2018.
nets found by gradient descent in the overparameterized setting that gave rise to this project, as well as warm encouragement and many enjoyable conversations. Cliff Taubes for several very helpful discussions. Nathaniel Bottman for many fruitful conversations and helping to code many simulations testing the results presented in this paper.

The author would also like to thank Dan Gulotta, Janos Kollar, Behnam Neyshabur, Matthias Schwarz, and Jacob Tsimmerman for many enjoyable conversations and valuable insights.

2. General facts

Since we are studying gradient descent in this work, we begin with an observation about gradients. In general, the computation of the gradient of a function is coordinate-dependent. However, if one makes an orthogonal change of coordinates, the gradient computed in new coordinates is correct. We state this fact precisely in the following lemma.

**Lemma 1.** If \( \phi : \mathbb{R}^n \to \mathbb{R}^n \) is an orthogonal change of coordinates from \((x_1, ..., x_n)\) to \((y_1, ..., y_n) = (\phi_1(x_1, ..., x_n), ..., \phi_n(x_1, ..., x_n))\), meaning that the Jacobian \( D\phi \) is an orthogonal matrix at every point, and \( f : \mathbb{R}^n \to \mathbb{R} \) is a smooth function, then taking the gradient with respect to the \( x \) coordinates is equivalent to taking the gradient with respect to the \( y \) coordinates. Concretely,

\[
\left( \frac{\partial}{\partial x_1} (f \circ \phi), ..., \frac{\partial}{\partial x_n} (f \circ \phi) \right) = D\phi^{-1} \left( \frac{\partial}{\partial y_1} f, ..., \frac{\partial}{\partial y_n} f \right).
\]

**Proof.** First, we compute the pull-back of \( \nabla_y f \) to the \( x \)-coordinates. This is simply

\[
D\phi^{-1} \nabla_y f,
\]

where

\[
D\phi = \begin{pmatrix}
\frac{\partial \phi_1}{\partial x_1} & \cdots & \frac{\partial \phi_n}{\partial x_1} \\
\vdots & \ddots & \vdots \\
\frac{\partial \phi_1}{\partial x_n} & \cdots & \frac{\partial \phi_n}{\partial x_n}
\end{pmatrix}
\]

is the Jacobian of the map.

Next, we use the chain rule

\[
\frac{\partial}{\partial x_i} (f \circ \phi) = \frac{\partial f}{\partial y_1} \frac{\partial y_1}{\partial x_i} + ... + \frac{\partial f}{\partial y_n} \frac{\partial y_n}{\partial x_i}
\]

to compute

\[
\nabla_x (f \circ \phi) = \begin{pmatrix}
\frac{\partial \phi_1}{\partial x_1} & \cdots & \frac{\partial \phi_n}{\partial x_1} \\
\vdots & \ddots & \vdots \\
\frac{\partial \phi_1}{\partial x_n} & \cdots & \frac{\partial \phi_n}{\partial x_n}
\end{pmatrix} \begin{pmatrix}
\frac{\partial f}{\partial y_1} \\
\vdots \\
\frac{\partial f}{\partial y_n}
\end{pmatrix} = D\phi^T \nabla_y f
\]

Because \( \phi \) is orthogonal, the Jacobian \( D\phi \) is an orthogonal matrix at every point. Hence \( D\phi^{-1} = D\phi^T \), and we conclude that the two vectors

\[
D\phi^{-1} \nabla_y f
\]

and

\[
D\phi^{T} \nabla_y f
\]
are equal.

We conclude that $\nabla_x (f \circ \phi)$ is equal to $D\phi^{-1} \nabla_y f$, as desired.

This lemma will be important to us, as it means that we can analyze the dynamics of gradient descent after a change of coordinates as long as the change of coordinates is orthogonal. We will find that for the problem of interest to us, there does exist an orthogonal change of coordinates in which the dynamics are much clearer.

3. Notation and assumptions

In this paper, we will study gradient descent as implemented discretely on a computer. We begin by defining discrete gradient descent with step size $\tau > 0$. The process is defined as follows. Suppose we wish to use gradient descent to minimize a function $L : \mathbb{R}^n \to \mathbb{R}$. Begin at an initial position $w_0 \in \mathbb{R}^n$ (often chosen randomly). Iteratively define a sequence $w_t$ by the rule

$$w_{t+1} = w_t - \tau \nabla L(w_t).$$

This is completely general, and can be applied to try to minimize any function $L : \mathbb{R}^n \to \mathbb{R}$. However, in this work we will be interested in functions of the following special form.

For $1 \leq i \leq d < n$, let $f_i : \mathbb{R}^n \to \mathbb{R}$ be a continuous function with 0 as a regular value. We study gradient descent on functions of either this form

$$L_1(w) = \sum_{i=1}^d |f_i(w)|$$

or this form

$$L_2(w) = \sum_{i=1}^d (f_i(w))^2.$$

When our discussion applies to both, we will call the function we are minimizing (often called a “loss function”) simply $L$.

By definition, $L$ is a nonnegative function, hence the smallest possible value for $L$ is 0. If 0 is achieved as a value of $L$, then the locus of global minima of $L$ is

$$M = L^{-1}(0).$$

Let us define

$$M_i = f_i^{-1}(0).$$

Note that

$$M = \bigcap_i M_i.$$

Generically, we expect that $M$ is $n - d$ dimensional. In this paper, we will make the following assumptions.

Assumption 1. We will assume that all of the hypersurfaces $M_i$ intersect pairwise perpendicularly, and that $M$ is indeed $n - d$ dimensional.
We conclude this section by fixing a coordinate system on a neighborhood $V$ of any $m \in M$. We choose to use tubular coordinates.

For any $p \in M$, there is a neighborhood $U \subset M$ of $p$ such that there is a diffeomorphism $\phi$ from $U \times [-\epsilon, \epsilon]^d$ to a tubular neighborhood of $U$. Let $(q_1, \ldots, q_{n-d})$ be an orthogonal coordinate system on $U$ (meaning the Jacobian of the change of coordinates is an orthogonal matrix) which we’ll denote by $q$. Each $M_i$ has a normal vector field. Let $n_i$ denote the unit normal vector in the direction $f_i$ is increasing. For $(s_1, \ldots, s_d) \in [-\epsilon, \epsilon]^d$, $\phi$ sends $(q, s)$ to $q + \sum_i s_i n_i$.

Assumption 2. We assume that such coordinates $(q, s)$ can be found, namely that at any $p \in M$, there exists an open neighborhood $U$ of $p$ on which an orthogonal coordinate system exists.

4. Preliminary calculations

Now we Taylor expand the function $f_i(q, s)$ in the $s$ coordinates.

$$f_i(q, s) = f_i(q, 0) + \sum_j \frac{\partial}{\partial s_j} f_i(q, 0) s_j + O(s^2)$$

Each $f_i$ vanishes on $M$, so

$$f_i(q, 0) = 0 \text{ for all } i.$$  

Furthermore, because we assumed that $M_i$ are perpendicular,

$$\frac{\partial f_i}{\partial s_j}(q, 0) = 0 \text{ for } j \neq i.$$  

Hence,

$$(4.1) \quad f_i(q, s) = \frac{\partial f_i}{\partial s_i}(q, 0) s_i + \ldots$$

We finish with a useful calculation of derivatives on the submanifold $M$. This calculation holds for any $s_i$ or $f_j$ (though the expression vanishes if $i \neq j$), so we suppress the indices.

$$\left. \frac{1}{2} \left( \frac{\partial}{\partial s} \left( \frac{\partial f^2}{\partial s} \right) \right) \right|_M = \left. \frac{1}{2} \left( \frac{\partial}{\partial s} \left( 2 f \frac{\partial f}{\partial s} \right) \right) \right|_M$$

$$= \left. \left( \frac{\partial f}{\partial s} \right)^2 \right|_M + f \left. \frac{\partial^2 f}{\partial s^2} \right|_M$$

The second term vanishes because $f$ is 0 on $M$. So we conclude that

$$(4.2) \quad \left. \frac{1}{2} \left( \frac{\partial}{\partial s} \left( \frac{\partial f^2}{\partial s} \right) \right) \right|_M = \left. \left( \frac{\partial f}{\partial s} \right)^2 \right|_M$$
5. L1 without noise

Using our Taylor expansion of $f_i$ in equation (4.1), we can approximate $L_1$ as

$$L_1(q, s) \simeq \sum_i \left| \frac{\partial}{\partial s_i} f_i(q, 0) \right| |s_i|.$$  

(5.1)

To analyze gradient descent on $L_1(q, s)$, we begin by writing expressions for the derivatives in each coordinate.

$$\frac{\partial}{\partial s_k} L_1(q, s) = \frac{\partial}{\partial s_k} \left( \sum_i \left| \frac{\partial f_i}{\partial s_i}(q, 0) \right| |s_i| \right).$$

We use the product rule to compute the derivative. The first factor is a function only of $q$ and not $s$, so the derivative vanishes. The second factor gives nonvanishing derivative only for $i = k$. Therefore

$$\frac{\partial}{\partial s_k} L_1(q, s) = \left| \frac{\partial f_k}{\partial s_k}(q, 0) \right| \text{sgn}(s_k).$$

(5.2)

Meanwhile, we simply record that

$$\frac{\partial}{\partial q_k} L_1(q, s) = \frac{\partial}{\partial q_k} \left( \sum_i \left| \frac{\partial f_i}{\partial s_i}(q, 0) \right| |s_i| \right).$$

(5.3)

Now we analyze the dynamics of gradient descent in the $s$ directions. Consider a single coordinate $s_k$. The gradient descent rule in this coordinate is

$$s_{k,t+1} = s_{k,t} - \tau \left| \frac{\partial f_k}{\partial s_k}(q, 0) \right| \text{sgn}(s_{k,t}).$$

So if at the $t^{th}$ step $(q_{1,t}, \ldots, q_{n-d,t}, s_{1,t}, \ldots, s_{d,t})$ we have

$$|s_{k,t}| > \tau \left| \frac{\partial f_k}{\partial s_k}(q, 0) \right|,$$

then $s_{k,t+1}$ moves closer to $M$, whether $s_{k,t}$ is positive or negative. However, if

$$|s_{k,t}| < \tau \left| \frac{\partial f_k}{\partial s_k}(q, 0) \right|,$$

and $s_{k,t}$ is positive, then $s_{k,t+1}$ becomes negative, i.e. the sequence crosses to the other side of $M_k$. Similarly, if $s_{k,t}$ is negative, then $s_{k,t+1}$ is positive. Furthermore, in this case, $|s_{k,t+1}|$ is also less than $\tau |\frac{\partial f_k}{\partial s_k}(q, 0)|$, which means that in the next step the sequence crosses $M_k$ again, and so on, indefinitely. This is the source of secondary gradient descent.

Now we wish to analyze the process to find where secondary gradient descent takes the sequence along $M$. To make the calculation cleanest, we analyze the steps in pairs. Assume that $|s_{k,t}| < \tau |\frac{\partial f_k}{\partial s_k}(q, 0)|$. Then
\[ s_{k,t+1} = s_{k,t} - \tau \left| \frac{\partial f_k(q,0)}{\partial s_k} \right| \text{sgn}(s_{k,t}) \]

and
\[
\begin{align*}
 s_{k,t+2} & = s_{k,t+1} - \tau \left| \frac{\partial f_k(q,0)}{\partial s_k} \right| \text{sgn}(s_{k,t+1}) \\
 & = \left( s_{k,t} - \tau \left| \frac{\partial f_k(q,0)}{\partial s_k} \right| \text{sgn}(s_{k,t}) \right) - \tau \left| \frac{\partial f_k(q,0)}{\partial s_k} \right| \text{sgn}(s_{k,t+1})
\end{align*}
\]

As just discussed,
\[ \text{sgn}(s_{k,t}) = -\text{sgn}(s_{k,t+1}) = \text{sgn}(s_{k,t+2}). \]

So
\[ (5.4) \quad s_{k,t+2} = s_{k,t}. \]

Furthermore,
\[ (5.5) \quad |s_{k,t}| + |s_{k,t+1}| = \tau \left| \frac{\partial f_k(q,0)}{\partial s_k} \right|. \]

To see this, write
\[ s_{k,t} = a \cdot \tau \left| \frac{\partial f_k(q,0)}{\partial s_k} \right| \]
for \(0 < a < 1\).

Without loss of generality, we can assume that \(s_{k,t}\) is positive. (If not, replace \(s_{k,t}\) with \(s_{k,t+1}\).)

Then
\[
\begin{align*}
 s_{k,t+1} & = s_{k,t} - \tau \left| \frac{\partial f_k(q,0)}{\partial s_k} \right| \text{sgn}(s_{k,t}) \\
 & = a \cdot \tau \left| \frac{\partial f_k(q,0)}{\partial s_k} \right| - \tau \left| \frac{\partial f_k(q,0)}{\partial s_k} \right| \\
 & = (a-1) \tau \left| \frac{\partial f_k(q,0)}{\partial s_k} \right|
\end{align*}
\]

We can make a more explicit derivation of (5.4) by noting that \(0 < a < 1\) implies \(-1 < a-1 < 0\). So
\[
\begin{align*}
 s_{k,t+2} & = s_{k,t+1} - \tau \left| \frac{\partial f_k(q,0)}{\partial s_k} \right| \text{sgn}(s_{k,t+1}) \\
 & = (a-1) \tau \left| \frac{\partial f_k(q,0)}{\partial s_k} \right| - \tau \left| \frac{\partial f_k(q,0)}{\partial s_k} \right|(-1) \\
 & = a\tau \left| \frac{\partial f_k(q,0)}{\partial s_k} \right| \\
 & = s_{k,t}
\end{align*}
\]
Importantly for us, $0 < a < 1$ implies

$$|a| + |a - 1| = a - (a - 1) = 1.$$ 

Hence

$$|s_{k,t}| + |s_{k,t+1}| = (|a| + |a - 1|)\tau|\frac{\partial f_k}{\partial s_k}(q, 0)|$$

$$= \tau|\frac{\partial f_k}{\partial s_k}(q, 0)|$$

showing (5.5).

Now, we use this analysis of the dynamics in $s$ to understand the behavior of gradient descent in the $q$ coordinates. We use equation (5.3) to analyze a pair of steps.

$$q_{k,t+1} = q_{k,t} - \tau \frac{\partial}{\partial q_k} \left( \sum_i \left| \frac{\partial f_i}{\partial s_i}(q, 0) \right| |s_{i,t}| \right)$$

$$q_{k,t+2} = q_{k,t+1} - \tau \frac{\partial}{\partial q_k} \left( \sum_i \left| \frac{\partial f_i}{\partial s_i}(q, 0) \right| |s_{i,t+1}| \right)$$

$$= q_{k,t} - \tau \frac{\partial}{\partial q_k} \left( \sum_i \left| \frac{\partial f_i}{\partial s_i}(q, 0) \right| |s_{i,t}| \right) - \tau \frac{\partial}{\partial q_k} \left( \sum_i \left| \frac{\partial f_i}{\partial s_i}(q, 0) \right| |s_{i,t+1}| \right)$$

$$= q_{k,t} - \tau \frac{\partial}{\partial q_k} \left( \sum_i \left| \frac{\partial f_i}{\partial s_i}(q, 0) \right| (|s_{i,t}| + |s_{i,t+1}|) \right)$$

$$= q_{k,t} - \tau \frac{\partial}{\partial q_k} \left( \sum_i \left( \frac{\partial f_i}{\partial s_i}(q, 0) \right)^2 \right)$$

Using the calculation (4.2),

$$q_{k,t+2} = q_{k,t} - \tau^2 \frac{\partial}{\partial q_k} \left( \frac{1}{2} \sum_i \frac{\partial f_i^2}{\partial s_i^2}(q, 0) \right)$$

But now we see that in the $q$ coordinates, that is, along $M$, secondary gradient descent behaves as discrete gradient descent within $M$ with step size $\frac{\tau}{2}$ minimizing the function

$$\text{tr}(\mathcal{H}(\sum_i f_i^2))|_M.$$ 

Thus we have proved the following.

**Theorem 1.** If discrete gradient descent is used to minimize a function $L_1$ of the form (5.2), then secondary gradient descent, if it emerges, can be modeled as discrete gradient descent on $K = \text{tr}(\mathcal{H}(L_2))|_M$ with step size $\frac{\tau}{2}$. 

7
6. L2 without noise

Now we consider gradient descent on the function $L_2$. Again we use the Taylor expansion (4.1) to approximate $L_2$. Now we have

\[ L_2(q, s) = \sum_i \left( \frac{\partial f_i}{\partial s_i}(q, 0)s_i \right)^2 + \ldots. \]  

We calculate the derivative in the $s$ then $q$ coordinates.

\[
\frac{\partial}{\partial s_k} L_2(q, s) = \frac{\partial}{\partial s_k} \left( \sum_i \left( \frac{\partial f_i}{\partial s_i}(q, 0)s_i \right)^2 \right)
\]
\[
= \sum_i \left( \frac{\partial}{\partial s_k} \left( \frac{\partial f_i}{\partial s_i}(q, 0) \right)^2 \right) s_i^2 + \left( \frac{\partial f_i}{\partial s_i}(q, 0) \right) \frac{\partial}{\partial s_k} s_i^2
\]

The first factor is a function only of $q$ and not $s$, so the derivative vanishes. The second factor gives nonvanishing derivative only for $i = k$. Therefore

\[
\frac{\partial}{\partial s_k} L_2(q, s) = 2 \left( \frac{\partial f_k}{\partial s_k}(q, 0) \right)^2 s_k
\]
\[
= \frac{\partial^2 f_k^2}{\partial s_k^2}(q, 0)s_k
\]

where the second equality uses (4.2).

Meanwhile,

\[
\frac{\partial}{\partial q_k} L_2(q, s) = \frac{\partial}{\partial q_k} \left( \sum_i \left( \frac{\partial f_i}{\partial s_i}(q, 0)s_i \right)^2 \right)
\]

Now, we analyze the dynamics in the $s$ coordinates. Consider a single coordinate $s_k$. The gradient descent rule in this coordinate is

\[
s_{k,t+1} = s_{k,t} - \tau \frac{\partial^2 f_k^2}{\partial s_k^2}(q, 0)s_{k,t}
\]

Let $c = \tau \frac{\partial^2 f_k^2}{\partial s_k^2}(q, 0)$. Then we can write the gradient descent rule as

\[
s_{k,t+1} = (1 - c)s_{k,t}
\]

We can see now that the dynamics in the $s_k$ coordinate are simple. The sequence $s_{k,t}$ diverges for $c > 2$. For $0 < c < 2$, the sequence is a geometric sequence which converges to 0. We will assume that for all $k$,
\[ \tau \frac{\partial^2 f^2_k}{\partial s^2_k}(q, 0) < 1. \]

For \(0 < c < 1\), the sequence \(s_{k,t}\) stays on one side of \(M_k\) - that is, if \(s_{k,t}\) is positive, the future positions are all positive and if \(s_{k,t}\) is negative, the future positions are all negative. On the other hand, for \(1 < c < 2\), the sequence \(s_{k,t}\) changes sign at every step.

In both cases, \(s_{k,t}\) converges to 0, and secondary gradient descent does not emerge. This is because (6.2) is first order in \(s_k\) while (6.4) is second order in \(s_k\). The magnitude of the \(q\)-derivatives is much smaller than the magnitude of the \(s\)-derivatives, and so there is appreciable movement in the \(q\) directions, that is along \(M\), only if a very large total distance is traveled in the \(s\) directions. In the case that the \(L^2\) loss is minimized using discrete gradient descent, the process of gradient descent effectively terminates because the process converges, not because the process finds a stable point of the secondary gradient field.

7. \(L^2\) with noise

In the last section, we saw that when the function \(L^2\) is minimized by discrete gradient descent, secondary gradient descent does not arise. In this section we will see that if instead a noisy modification of discrete gradient descent is used, secondary gradient descent can arise. If and when it does depends on the details of the noise.

In [C18] two natural ways to add noise were considered. One way to add noise to the process is to add a small random vector to the gradient at each step. Another way is to add at each step a small random number to the label \(y_i\) of the data. Both methods were considered in [C18], where a heuristic argument was given for why secondary gradient descent should arise in the second case but not the first, and computer simulations of each were run confirming this.

In this section we will analyze the dynamics of gradient descent in the second setting, with noise added to the labels. Further motivation for studying this model is that it is a simplified model for stochastic gradient descent, which is of significant interest in the context of deep learning. We begin by giving a precise definition of the process and then mathematically analyze the dynamics of this process.

Given a noise vector

\[ \epsilon = (\epsilon_1, \ldots, \epsilon_d), \]

we define a perturbed loss function

\[ (7.1) \quad L_\epsilon(q, s) = \sum_{i=1}^{d} (f_i(q, s) + \epsilon_i)^2. \]

The noisy gradient descent we will analyze, which we will call \(\epsilon\)-noisy gradient descent, is defined by the rule

\[ (7.2) \quad w_{t+1} = w_t - \tau \nabla L_\epsilon(w_t) \]

where at each step a vector \(\epsilon_t\) is constructed by drawing each element \(\epsilon_{i,t}\) from a gaussian distribution \(G\) with mean zero and standard deviation \(\epsilon\).
Now define

\begin{equation}
R_{\epsilon}(q,s) = \sum_{i=1}^{d} 2\epsilon f_i(q,s) + \epsilon^2_i.
\end{equation}

We choose this because

\begin{equation}
L_{\epsilon}(q,s) = L(q,s) + R_{\epsilon}(q,s)
\end{equation}

Using the Taylor expansion of \(f_i\) \((4.1)\), we can approximate \(R_{\epsilon}\) as

\begin{equation}
R_{\epsilon}(q,s) = \sum_{i=1}^{d} 2\epsilon \frac{\partial f_i}{\partial s_i}(q,s) + \epsilon^2_i.
\end{equation}

Next, we calculate the derivative of \(L_{\epsilon}\) in the \(s\) and \(q\) coordinates. In the previous section we already computed the derivatives of \(L\), so now we compute the derivatives of \(R_{\epsilon}\).

\begin{equation}
\frac{\partial}{\partial s_k} R_{\epsilon}(q,s) = 2\epsilon_k \frac{\partial f_k}{\partial s_k}(q,0)
\end{equation}

Meanwhile,

\begin{equation}
\frac{\partial}{\partial q_k} R_{\epsilon}(q,s) = \frac{\partial}{\partial q_k} \left( \sum_{i} 2\epsilon_i \frac{\partial f_i}{\partial s_i}(q,0)s_i \right)
\end{equation}

Now we consider the derivative of \(L_{\epsilon}\) in the \(s_k\) coordinate,

\begin{align*}
\frac{\partial}{\partial s_k} L_{\epsilon}(q,s) &= \frac{\partial}{\partial s_k} R_{\epsilon}(q,s) + \frac{\partial}{\partial s_k} L(q,s) \\
&= 2\epsilon_k \frac{\partial f_k}{\partial s_k}(q,0) + 2 \left( \frac{\partial f_k}{\partial s_k}(q,0) \right)^2 s_k.
\end{align*}

So the gradient descent rule in \(s_k\) is

\begin{equation}
s_{k,t+1} = s_{k,t} - 2\tau \left( \epsilon_k \frac{\partial f_k}{\partial s_k}(q,0) + \left( \frac{\partial f_k}{\partial s_k}(q,0) \right)^2 s_k \right).
\end{equation}

This is an example of an \(AR(1)\) process \([H94]\). The \(AR(1)\) model is a discrete analog of the Ornstein-Uhlenbeck process. Both are well studied, and in particular, the stationary state in the limit \(t \to \infty\) limit is known, which will be helpful to us here.

An \(AR(1)\) process is one modeled by the following first order linear difference equation.

\[ x_{t+1} = x_t + \theta (\mu - x_t) + \epsilon_t, \]

where \(|\theta| < 1\), \(\mu\) is the model mean, and \(\epsilon_t\) is drawn from a gaussian distribution with mean 0 and standard deviation \(\sigma\).
The stationary solution is a normal distribution with mean \( \mu \) and variance
\[
Var = \frac{\sigma^2}{2\theta}.
\]
(7.7)

Now we apply this to our setting. We see that \((7.6)\) is an example of an \(AR(1)\) process, with
\[
\mu = 0,
\]
\[
\theta = 2\tau \left( \frac{\partial f_k}{\partial s_k}(q, 0) \right)^2,
\]
and
\[
\sigma = 2\tau \epsilon \frac{\partial f_k}{\partial s_k}(q, 0).
\]

We assume that
\[
2\tau \left( \frac{\partial f_k}{\partial s_k}(q, 0) \right)^2 < 1.
\]

Thus by \((7.7)\), the mean of \(s_{k,t}\) in the large \(t\) limit under noisy gradient descent is 0 and the variance is
\[
\mathbb{E}(s_{k,t}^2) = \frac{\left( 2\tau \epsilon \frac{\partial f_k}{\partial s_k}(q, 0) \right)^2}{4\tau \left( \frac{\partial f_k}{\partial s_k}(q, 0) \right)^2} = \tau \epsilon^2.
\]
(7.8) (7.9)

Now we use this to analyze the dynamics for gradient descent in the \(q\) variables.

\[
\frac{\partial}{\partial q_k} L_{\epsilon}(q, s) = \frac{\partial}{\partial q_k} L(q, s) + \frac{\partial}{\partial q_k} R_{\epsilon}(q, s)
\]
(7.10)
\[
= \frac{\partial}{\partial q_k} \left( \sum_i \left( \frac{\partial f_i}{\partial s_i}(q, 0)s_i \right)^2 \right) + \frac{\partial}{\partial q_k} \left( \sum_i 2\epsilon_i \frac{\partial f_i}{\partial s_i}(q, 0)s_{i,t} \right)
\]
(7.11)

So the gradient descent rule in \(q_k\) is
\[
q_{k,t+1} = q_{k,t} - \tau \left( \frac{\partial}{\partial q_k} \left( \sum_i \left( \frac{\partial f_i}{\partial s_i}(q, 0)s_i \right)^2 \right) + \frac{\partial}{\partial q_k} \left( \sum_i 2\epsilon_i \frac{\partial f_i}{\partial s_i}(q, 0)s_{i,t} \right) \right).
\]
(7.12)

We wish to compute the expected value
\[
\mathbb{E} \left( \tau \frac{\partial}{\partial q_k} \left( \sum_i \left( \frac{\partial f_i}{\partial s_i}(q, 0)s_{i,t} \right)^2 \right) + \tau \frac{\partial}{\partial q_k} \left( \sum_i 2\epsilon_{i,t} \frac{\partial f_i}{\partial s_i}(q, 0)s_{i,t} \right) \right)
\]
The contribution from the second term is 0, because $\epsilon_{i,t}$ is drawn from a normal distribution with mean 0. The first term is easy to compute now that we’ve computed the variance of $s_{i,t}$, namely $\mathbb{E}(s_{i,t}^2) = \tau \epsilon^2$. So

$$
\mathbb{E}
\left(
\sum_i \tau \frac{\partial}{\partial q_k} \left(\frac{\partial f_i}{\partial s_i}(q,0)s_{i,t}\right)^2
\right)
= \tau \left(\frac{\partial}{\partial q_k} \sum_i \left(\frac{\partial f_i}{\partial s_i}(q,0)\right)^2 \mathbb{E}(s_{i,t}^2)\right)
= \tau^2 \epsilon^2 \left(\frac{\partial}{\partial q_k} \sum_i \left(\frac{\partial f_i}{\partial s_i}(q,0)\right)^2\right).
$$

Rewriting this using (4.2), we find

$$
\mathbb{E}
\left(
\sum_i \tau \frac{\partial}{\partial q_k} \left(\frac{\partial f_i}{\partial s_i}(q,0)s_{i,t}\right)^2
\right)
= \frac{\tau^2 \epsilon^2}{2} \left(\frac{\partial}{\partial q_k} \sum_i \frac{\partial^2 f_i^2}{\partial s_i^2}(q,0)\right).
$$

By our choice of coordinates,

$$
\frac{\partial f_i}{\partial s_j} = 0 \text{ if } j \neq i.
$$

So

$$
\frac{\partial^2 f_i^2}{\partial s_i^2}(q,0) = \sum_j \frac{\partial^2 f_i^2}{\partial s_i^2}(q,0)
= \frac{\partial^2 L_2}{\partial s_i^2}(q,0)
$$

which is an eigenvalue of the Hessian of $L_2|_M$. We conclude that

$$
(7.13) \quad \mathbb{E}
\left(
\sum_i \tau \frac{\partial}{\partial q_k} \left(\frac{\partial f_i}{\partial s_i}(q,0)s_{i,t}\right)^2
\right)
= \frac{\tau^2 \epsilon^2}{2} \left(\frac{\partial}{\partial q_k} \sum_i \frac{\partial^2 L_2}{\partial s_i^2}(q,0)\right)
$$

$$
(7.14) \quad = \frac{\tau^2 \epsilon^2}{2} \left(\frac{\partial}{\partial q_k} \text{tr}(\mathcal{H}(L_2))(q,0)\right).
$$

So we can rewrite the approximation for gradient descent in the $q_k$ coordinate as

$$
(7.15) \quad q_{k,t+1} = q_{k,t} - \frac{\tau^2 \epsilon^2}{2} \left(\frac{\partial}{\partial q_k} \text{tr}(\mathcal{H}(L_2))(q,0)\right).
$$

We conclude that in implementing noisy gradient descent on $L_2$ if the sequence comes near $M$, a secondary phase arises that behaves as gradient descent within $M$ minimizing the function

$$
(7.16) \quad K = \text{tr}(\mathcal{H}(L_2))|_M.
$$

From our analysis, we can see that this is a second order effect which acts much more slowly than primary gradient descent. In the original gradient descent rule, the step size is $\tau$. However, the emergent step size for secondary gradient descent within $M$ is order $\tau^2$.

Thus we have proved the following.
Theorem 2. If $\epsilon$-noisy gradient descent is used to minimize a function $L_2$ of the form (3.3), then secondary gradient descent, if it emerges, can be modeled as discrete gradient descent on $K = \text{tr}(\mathcal{H}(L_2))|_M$ with step size $\frac{\tau^2\epsilon^2}{2}$.

8. Conclusion

In this paper we have analyzed the behavior of discrete gradient descent on functions of the form $L_1$ and the modified $\epsilon$-noisy gradient descent on functions of the form $L_2$. We have seen that discrete gradient descent displays strikingly different behavior from continuous gradient descent. Under continuous gradient descent, if the flow line reaches $M$, the process ends there. However, under discrete gradient descent on $L_1$ or $\epsilon$-noisy gradient descent on $L_2$, if the sequence reaches near $M$ and the algorithm continues running, a second phase of the process emerges. During this phase, the sequence proceeds along the submanifold $M$, effectively minimizing the function $K = \text{tr}(\mathcal{H}(L_2))|_M$.

The origin of this phase in the case of discrete gradient descent on $L_1$ is that the step size $\tau$ defining the process is constant, and at $(q, s)$ near $M$, the effective step size in the $s_k$ coordinate is approximately $\tau \left| \frac{\partial f_k}{\partial s_k}(q, 0) \right|$, which is also constant. So the process never reaches $(q, 0)$, but rather bounces around in a tube near $M$ of radius approximately $\tau \left| \frac{\partial f_k}{\partial s_k}(q, 0) \right|$ in the $s_k$ coordinate. On long time scales, this does not have any net effect in the $s$ directions, as in each coordinate $s_k$ the movement is as often negative and positive, and the process stays within a neighborhood of $M$.

However, it is important that this effect causes discrete gradient descent to continue iterating indefinitely, in particular long after the sequence has gotten near $M$. At a finite distance from $M$ the gradient field of $L_1$ is not perfectly perpendicular to $M$. So at each step there is a small tangential component to the process, and these do not cancel each other. In fact, we prove that each of these small tangential vectors is in the direction minimizing $K = \text{tr}(\mathcal{H}(L_2))|_M$.

The cumulative effect of many steps taken in the same direction is that secondary gradient descent emerges, which effectively behaves as discrete gradient descent along $M$ with step size $\frac{\tau^2\epsilon^2}{2}$, minimizing the function $K$. Geometrically, at a global minimum $m \in M$, $K(m)$ is a measure of the steepness of the graph of $L_2$ near $m$. So secondary gradient descent gives discrete gradient descent an implicit bias for shallow minima of $L_1$ over steeper ones.

The origin of secondary gradient descent in the case of $\epsilon$-noisy gradient descent on $L_2$ is related, but more subtle. The norm of the gradient field of $L_2$ does go to zero as one approaches $M$, and discrete gradient descent without any noise added behaves similarly to gradient flow. However, with the addition of noise, the process can be effectively prevented from getting too close to $M$. This allows the possibility that secondary gradient descent can arise. Whether secondary gradient descent does or not depends on the details of how noise is added.

When $\epsilon$-noisy gradient descent is implemented to minimize a function of the form $L_2$, the small tangential components of the gradient at each step are on average in the same direction, again minimizing $K = \text{tr}(\mathcal{H}(L_2))|_M$, and secondary gradient descent does emerge. In this case, secondary gradient descent effectively behaves as discrete gradient descent along $M$ with step size $\frac{\tau^2\epsilon^2}{2}$, minimizing the function $K$.

The existence of secondary gradient descent has the effect that on long time scales, discrete gradient descent and continuous gradient descent tend toward different minima of $L$. Discrete gradient
descent preferentially finds global minima at which the function $L$ is shallowest whereas gradient flow shows no such preference.

Secondary gradient descent really is a second order effect - in both cases we've analyzed here, the emergent step size for secondary gradient descent is order $\tau^2$, compared to the original step size of $\tau$ in the primary phase. So we expect that secondary gradient descent proceeds very slowly compared to the first phase of gradient descent when the sequence goes from its initial point to a region near the critical manifold $M$.

Note furthermore the difference in the effective step size for secondary gradient descent as it arises during discrete gradient descent on $L_1$ versus as it arises during $\epsilon$-noisy gradient descent on $L_2$. In the first case, it is $\frac{\tau^2}{2}$, while in the second, it is $\frac{\tau \epsilon^2}{2}$. As $\epsilon$ likely needs to be smaller than 1 for the process to be stable (ie. not diverge), this means that secondary gradient descent will usually proceed faster under discrete gradient descent on $L_1$ than under $\epsilon$-noisy gradient descent on $L_2$.

It matches intuition that secondary gradient descent arising during discrete gradient descent on $L_1$ would work more effectively because at every step, the function $K = \text{tr}(H(L_2))$ is being minimized. In contrast, when secondary gradient descent arises during $\epsilon$-noisy gradient descent on $L_2$, $K$ on average minimized but there is randomness in the direction traveled at each individual step, which slows the progress along $M$.

In fact, for a completely general function $L : \mathbb{R}^n \rightarrow \mathbb{R}$, secondary gradient descent should emerge under some implementations of discrete gradient descent. The main ingredients necessary for the emergence of secondary gradient descent is that the process continue indefinitely - e.g. because the effective step size is bounded from below, or due to noise - and that the small tangential component at each step isn’t masked by noise. In this case, secondary gradient descent should behave as though one were implementing discrete gradient descent on $M$ minimizing some function $\tilde{J}$ which is a function of the original function $L$.

Currently we don’t have a formula for $\tilde{J}$ in the most general setting, which is why we restricted our attention in this work to functions of the form $L_1$ and $L_2$ which satisfy Assumptions 1 and 2. These assumptions were used in the proofs of the main theorems in this paper to simplify the calculations. If they are not satisfied, we still expect secondary gradient descent to arise when implementing discrete gradient descent on functions of the form $L_1$, or $\epsilon$-noisy gradient descent on functions of the form $L_2$. However, in that case, the second phase will approximate discrete gradient descent within $M$ minimizing a function $J$ that we are currently unable to compute in full generality. The function $K$ we derived in this paper will be an approximation of $J$, and the closer the assumptions are to being satisfied the better an approximation $K$ will be for $J$. In future work, we wish to analyze the behavior of secondary gradient descent under weaker versions of Assumptions 1 and 2.

In this work, we have given a qualitative description of the origins of secondary gradient descent, as well as found a mathematical description of the effects of secondary gradient descent in two specific settings. Given that gradient descent is often implemented discretely, secondary gradient descent likely arises in many applications, and may have significant effects in practical applications.

References

[C18] Y. Cooper. *Discrete gradient descent differs qualitatively from gradient flow*. [arXiv:1808.04839](http://arxiv.org/abs/1808.04839)

[H94] J.D. Hamilton. *Time Series Analysis*. Princeton University Press, 1994.