# SGD as a two step approximation to gradient flow

Ayoub El Hanchi

November 13, 2019

**Abstract**

We review the construction of stochastic gradient descent (SGD) as two step approximation to gradient flow. The first approximation comes from the Euler discretization of gradient flow which results in gradient descent. The second comes from the use of an estimator of the gradient instead of the full gradient in the update equation of gradient descent. This construction allows us to gain perspective on the workings of SGD. By making explicit the choices made in the discretization method and the estimator selection, we are able to point out how variants of SGD are able to improve its performance. A similar construction can be done for stochastic gradient Langevin dynamics from the continuous time Langevin dynamics [11]. This can also be extended to the accelerated version of both algorithms, namely Nesterov's accelerated gradient descent [7, 10], and underdamped Langevin dynamics [6].

## 1 Introduction

In many statistical and machine learning problems, we are interested in finding:

$$x^* \in \operatorname*{argmin}_{x \in \mathbb{R}^d} \left( f(x) = \sum_{i=1}^{N} f_i(x) \right) \tag{1}$$

where $f(x)$ is the negative log likelihood under the independence assumption in some statistical model, or a chosen empirical risk within the empirical risk minimization framework.

Many problems are in the regime where both $N$ and $d$ are large, so that computations that involve the use of the full dataset or the use of matrix computations are not available. The second limitation forces us to consider methods of order at most 1, while the first requires us to use only small subsets of the data at a time. The simplest algorithm that satisfies both requirements is SGD.

Here we construct SGD as a two step approximation to the gradient flow generated by $f$ on $\mathbb{R}^d$, the first of which comes from the discretization of gradient flow, and the second from the use of an estimator of the gradient instead of the full gradient. Our goal here is to make explicit the choices made in the construction of SGD, show how some extensions of SGD succeed in improving its convergence rate, and point out other directions in which it can be improved.

## 2 Assumptions and inequalities

We will assume that $f$ is twice differentiable, and that:

$$\forall x \in \mathbb{R}^d \quad \mu I \preceq \nabla^2 f(x) \preceq LI$$

The above assumption implies strong convexity which in turn implies the existence and uniqueness of $x^*$. It is also not hard to see using the mean value theorem that the above assumption, under the twice differentiability assumption, is equivalent to the following inequality:

$$\forall x, y \in \mathbb{R}^d \quad \mu \left\| y - x \right\|_2 \leq \left\| \nabla f(y) - \nabla f(x) \right\|_2 \leq L \left\| y - x \right\|_2 \tag{2}$$

and to the following inequalities taken together:

$$\forall x, y \in \mathbb{R}^d \quad f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{\mu}{2} \left\| y - x \right\|_2^2 \tag{3}$$

$$\forall x, y \in \mathbb{R}^d \quad f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + \frac{L}{2} \left\| y - x \right\|_2^2 \tag{4}$$

The upper and lower bounds above hold for all $x$ and $y$ in $\mathbb{R}^d$, and we can use this to obtain tighter bounds.

$$\begin{aligned} \forall x, y \in \mathbb{R}^d \quad f(y) \geq f(x) + \langle \frac{L}{L-\mu} \nabla f(x) - \frac{\mu}{L-\mu} \nabla f(y), y - x \rangle + \\ \frac{\mu L}{2(L-\mu)} \left\| y - x \right\|_2^2 + \frac{1}{2(L-\mu)} \left\| \nabla f(y) - \nabla f(x) \right\|_2^2 \end{aligned} \tag{5}$$

*Proof.* We have:

$$\begin{aligned} f(y) - f(x) &= [f(y) - f(z)] + [f(z) - f(x)] \\ &= -[f(z) - f(y)] + [f(z) - f(z)] \end{aligned}$$

using (4) on the first term and (3) on the second term, minimizing over $z$ and simplifying yields the result. □

and finally:

$$\forall x, y \in \mathbb{R}^d \quad \langle \nabla f(y) - \nabla f(x), y - x \rangle \geq \frac{\mu L}{L+\mu} \left\| y - x \right\|_2^2 + \frac{1}{L+\mu} \left\| \nabla f(y) - \nabla f(x) \right\|_2^2 \tag{6}$$

*Proof.* Use inequality (5) once with x and y as in the inequality, and a second time interchanging the roles of x and y. Adding the two inequalities, multiplying by $\frac{L-\mu}{L+\mu}$, and rearranging yields the result. □

## 3 Gradient Flow

Motivating the use of gradient flow as a way to reach the minimum of a differentiable strongly convex function can be done in many ways. We will content ourselves here with the fact that the negative gradient points in the direction of steepest descent, that is, for an infinitesimal displacement, the direction in which the value of the function decreases the most is that of the negative gradient. Intuitively then, if we follow the negative gradient direction at each point, we should arrive at the minimum of the function. Here we make this intuition precise.

- **Dynamics:**

$$x(0) = x_0$$
$$\frac{d}{dt}x(t) = -\nabla f(x(t))$$

- **Lyapunov function:**

$$T(x(t))) \coloneqq \|x(t) - x^*\|_2^2$$

- **Convergence rate:**

$$T(x(t)) \leq \exp(-\mu t) T(x_0)$$

*Proof.* We have:

$$\begin{aligned}
\frac{d}{dt}T(x(t)) &= -\langle \nabla f(x(t)), x(t) - x^* \rangle \\
&\overset{(6)}{\leq} -\frac{\mu L}{L+\mu} \|x(t) - x^*\|_2^2 - \frac{1}{L+\mu} \|\nabla f(x(t)) - \nabla f(x^*)\|_2^2 \\
&\overset{(2)}{\leq} -\mu \|x(t) - x^*\|_2^2 = -\mu T(x(t))
\end{aligned}$$

Applying Grönwall's inequality we get the result. □

## 4 Discretizing Gradient Flow

To obtain an algorithm from the the gradient flow of $f$, we have to discretize it. Ordinary differential equations (ODE) can in general be discretized in many different ways, depending on the goal of the discretization. One systematic way to obtain a discretization is by expanding the solution of the ODE at the desired point as a Taylor series around the current point. and truncating it at some chosen order to obtain an approximation of the desired point. When only the first order expansion is used, this is called Euler's method, and for the case of gradient flow, it yields gradient descent, which we now analyze.

### 4.1 Gradient Descent

- **Dynamics:**

$$x_0 = x_0$$
$$x_{k+1} - x_k = -\alpha \nabla f(x_k)$$
$$0 < \alpha \leq \frac{2}{L+\mu}$$

- **Lyapunov function:**

$$T(x_k) \coloneqq \|x_k - x^*\|_2^2$$

- **Convergence rate:**

$$T(x_k) \leq \left(1 - 2\alpha\frac{\mu L}{L+\mu}\right)^k T(x_0)$$

*Proof.*

$$\begin{aligned}
T(x_{k+1}) &= \|x_{k+1} - x^*\|_2^2 \\
&= \|x_k - \alpha\nabla f(x_k) - x^*\|_2^2 \\
&= \|x_k - x^*\|_2^2 - 2\alpha\langle \nabla f(x_k), x_k - x^*\rangle + \alpha^2 \|\nabla f(x_k)\|_2^2 \\
\overset{(6)}{\leq} &\left(1 - 2\alpha\frac{\mu L}{L+\mu}\right)\|x_k - x^*\|_2^2 + \alpha\left(\alpha - \frac{2}{L+\mu}\right)\|\nabla f(x_k)\|_2^2 \\
&\leq \left(1 - 2\alpha\frac{\mu L}{L+\mu}\right) T(x_k)
\end{aligned}$$

Where the last inequality holds since $\alpha \leq \frac{2}{L+\mu}$. Unrolling the recursion we obtain the result. □

## 4.2 Higher order discretizations

A natural question to ask is whether higher order discretizations can yield algorithms that have better computational complexity (measured in terms of number of gradient calls). For a given $\epsilon > 0$, it is not hard to see that gradient descent takes $O(\frac{L}{\mu}\log\frac{1}{\epsilon})$ iterations to reach a solution that is at most $\epsilon$ away from the optimum. When $\mu$ is small and $L$ is very large, this becomes expensive.

One problem with higher order discretization schemes based on Taylor expansions is that higher order derivatives become increasingly difficult to compute and to store. For moderately large $d$, even the Hessian cannot be stored. While that is true, one only needs to compute the application of these derivatives on a given set of vectors to obtain higher order discretizations, and this can be done efficiently using modern automatic differentiation software, at the cost of one additional gradient evaluation per higher order term. This allows us to make a direct comparison of the complexity of the higher order algorithms with gradient descent.

A second difficulty with using higher order discretizations schemes is that they become increasingly difficult to analyze, with more restrictive assumptions needed on $f$. Surprisingly, the second order case has a particularly simple analysis as we now show.

- **Dynamics:**

$$\begin{aligned}
x_0 &= x_0 \\
x_{k+1} - x_k &= -\alpha\nabla f(x_k) + \frac{\alpha^2}{2}\nabla^2 f(x_k)\nabla f(x_k)
\end{aligned}$$

- **Additional assumptions:**

$$\mu_2 I \preceq \nabla^2 f(x) - \frac{\alpha}{2}\nabla^2 f(x) - \frac{\alpha}{2}\nabla^3 f(x)\nabla f(x) \preceq L_2 I$$

$$\mu_2 > 0$$

- **Lyapunov function:**

$$T(x_k) \coloneqq \|x_k - x^*\|_2^2$$

- **Convergence rate:**

$$T(x_k) \leq \left(1 - 2\alpha \frac{\mu_2 L_2}{L_2 + \mu_2}\right)^k T(x_0)$$

*Proof.*

Consider the function:

$$g(x) = f(x) - \frac{\alpha}{4} \|\nabla f(x)\|_2^2$$

the additional assumption ensures that g is $\mu_2$-strongly convex and $L_2$-smooth, and the update equation corresponds to gradient descent with the function $g$. Applying the analysis of gradient descent on $g$ with the new strong convexity and smoothness constants yields the result. $\square$

The above additional assumption is unnatural and should instead be derived from assumptions on $f$ itself, although it is not clear to me what the assumption should be (Hessian Lipschitzness ?). Furthermore, $\mu_2$ and $L_2$ depend on $\alpha$, so in practice one should optimize over the choice of $\alpha$'s, under the appropriate constraints. Nevertheless, one can imagine scenarios where the ratio $\frac{\mu_2}{L_2}$ is much smaller than $\frac{\mu}{L}$ (the easiest example would be when $f$ is quadratic), and the cost of computing the additional term is more than made up for by the decrease in the number of iterations needed. Aside from quadratic functions, this could perhaps be extended to the case when $f$ is the negative log likelihood of generalized linear model. Many variance reduction algorithms can also be extended in a straightforward way to accommodate the use of higher order terms. We leave the full development of these ideas for future work.

## 5 Stochastically estimating the gradient

When the number of data points is large, computing the full gradient becomes computationally infeasible, and we replace the gradient by stochastic estimates of it in the gradient descent update equation. The choice of the 'optimal' estimator is still a subject of ongoing research, for some definition of optimal that takes into account the computational cost of the estimator, its memory requirements, and its associated rate of convergence.

### 5.1 Stochastic Gradient Descent

Here we perform the classical analysis of stochastic gradient descent with constant step size. We abstract away the specific choice of estimator and assume unbiasedness and uniform boundedness of the variance. Since we are using estimators of the gradient at each iteration, our iterates are now random variables. We denote them by $X_k$ and their realizations by $x_k$.

- **Dynamics:**

$$\begin{aligned}
X_0 &= x_0 \\
X_{k+1} - X_k &= -\alpha \widehat{\nabla f(x_k)} \\
0 < \alpha &\leq \frac{2}{L+\mu} \\
\mathbb{E}\left[\widehat{\nabla f(x_k)}\right] &= \nabla f(x_k) \\
\mathbb{E}\left[\left\|\widehat{\nabla f(x_k)} - \nabla f(x_k)\right\|_2^2\right] &\leq \sigma^2 d \quad \forall k \in \mathbb{N}
\end{aligned}$$

- **Lyapunov function:**

$$T(X_k) \coloneqq \|X_k - x^*\|_2^2$$

- **Convergence rate:**

$$\mathbb{E}\left[T(X_k)\right] \leq \left(1 - 2\alpha\frac{\mu L}{L+\mu}\right)^k T(x_0) + \alpha\frac{L+\mu}{2\mu L}\sigma^2 d$$

*Proof.* Conditionned on $\{X_k = x_k, ..., X_0 = x_0\}$, we have:

$$\begin{aligned}
\mathbb{E}\left[T(X_{k+1})\right] &= \mathbb{E}[\|X_{k+1} - x^*\|_2^2] \\
&= \mathbb{E}\left[\left\|x_k - \alpha\widehat{\nabla f(x_k)} - x^*\right\|_2^2\right] \\
&= \|x_k - x^*\|_2^2 - 2\alpha\langle \mathbb{E}\left[\widehat{\nabla f(x_k)}\right], x_k - x^*\rangle + \alpha^2\,\mathbb{E}\left[\left\|\widehat{\nabla f(x_k)}\right\|_2^2\right] \\
&= \|x_k - x^*\|_2^2 - 2\alpha\langle \nabla f(x_k), x_k - x^*\rangle + \alpha^2\,\mathbb{E}\left[\left\|\widehat{\nabla f(x_k)}\right\|_2^2\right] \\
{}_{(6)} &\leq \left(1 - 2\alpha\frac{\mu L}{L+\mu}\right)\|x_k - x^*\|_2^2 + \alpha\left(\alpha - \frac{2}{L+\mu}\right)\|\nabla f(x_k)\|_2^2 \\
&+ \alpha^2\,\mathbb{E}\left[\left\|\widehat{\nabla f(x_k)} - \nabla f(x_k)\right\|_2^2\right] \\
&\leq \left(1 - 2\alpha\frac{\mu L}{L+\mu}\right)T(x_k) + \alpha^2\sigma^2 d
\end{aligned}$$

Unrolling the recursion by taking successive expectations, we obtain:

$$\begin{aligned}
\mathbb{E}\left[T(X_k)\right] &\leq \left(1 - 2\alpha\frac{\mu L}{L+\mu}\right)^k T(x_0) + \alpha^2\sigma^2 d\sum_{i=1}^{k}\left(1 - 2\alpha\frac{\mu L}{L+\mu}\right)^{k-i} \\
&= \left(1 - 2\alpha\frac{\mu L}{L+\mu}\right)^k T(x_0) + \alpha^2\sigma^2 d\sum_{i=0}^{k-1}\left(1 - 2\alpha\frac{\mu L}{L+\mu}\right)^{i} \\
&\leq \left(1 - 2\alpha\frac{\mu L}{L+\mu}\right)^k T(x_0) + \frac{\alpha^2\sigma^2 d}{2\alpha\frac{\mu L}{L+\mu}} \\
&= \left(1 - 2\alpha\frac{\mu L}{L+\mu}\right)^k T(x_0) + \alpha\frac{L+\mu}{2\mu L}\sigma^2 d
\end{aligned}$$

□

The bounded variance assumption is unnatural, and should be proved instead from more basic assumptions. In [1], SGD is studied under a weaker assumption on the variance of the estimator, and in [8] this condition is proved to hold in the strongly convex setting. One can also remark that SGD does not converge to the optimum for any fixed step size. Decreasing the step size gradually achieves convergence, but then the convergence rate goes from linear to $O(\frac{1}{k})$.

## 5.2 Abstract Stochastic Gradient Descent

A core concept in statistical estimation is that of the bias-variance trade-off. The mean squared error is usually used as a way to measure the quality of an estimator, and in general, unbiased estimators are unlikely to have the lowest mean squared error. Here we consider a general estimator $\widehat{\nabla f(x_k)}$ of the gradient and analyze the additional terms that appear due to the introduction of bias, and argue that the SGD estimator might be inefficient due to its high variance and low (zero) bias.

- **Dynamics:**

$$X_0 = x_0$$
$$X_{k+1} - X_k = -\alpha\widehat{\nabla f(X_k)}$$

- **Lyapunov function:**

$$T(X_k) := \|X_k - x^*\|_2^2$$

- **Analysis:**

  Conditionned on $\{X_k = x_k, ..., X_0 = x_0\}$, define:

$$\delta_k := \mathbb{E}\left[\widehat{\nabla f(x_k)}\right] - \nabla f(x_k)$$
$$\sigma_k^2 := \mathbb{E}\left[\left\|\widehat{\nabla f(x_k)} - \mathbb{E}\left[\widehat{\nabla f(x_k)}\right]\right\|_2^2\right]$$

  Then we have:

$$\begin{aligned}
\mathbb{E}\left[T(X_{k+1})\right] &= \mathbb{E}\left[\|X_{k+1} - x^*\|_2^2\right] \\
&= \mathbb{E}\left[\left\|x_k - \alpha\widehat{\nabla f(x_k)} - x^*\right\|_2^2\right] \\
&= \|x_k - x^*\|_2^2 - 2\alpha\langle\mathbb{E}\left[\widehat{\nabla f(x_k)}\right], x_k - x^*\rangle + \alpha^2\,\mathbb{E}\left[\left\|\widehat{\nabla f(x_k)}\right\|_2^2\right] \\
&= \|x_k - x^*\|_2^2 - 2\alpha\langle\nabla f(x_k), x_k - x^*\rangle - 2\alpha\langle\delta_k, x_k - x^*\rangle \\
&+ \alpha^2(\sigma_k^2 + \|\delta_k\|_2^2 + 2\langle\delta_k, \nabla f(x_k)\rangle + \|\nabla f(x_k)\|_2^2) \\
&= \|x_k - x^*\|_2^2 - 2\alpha\langle\nabla f(x_k), x_k - x^*\rangle + \alpha^2\|\nabla f(x_k)\|_2^2 \\
&- 2\alpha\langle\delta_k, x_k - \alpha\nabla f(x_k) - x^*\rangle + \alpha^2(\sigma_k^2 + \|\delta_k\|_2^2)
\end{aligned}$$

We therefore obtain two additional terms compared to the analysis of gradient descent, both of which depend only on the bias and the variance of the gradient estimator. We can recognize the second argument of the inner product in the first term as $\tilde{x}_{k+1} - x^*$ where $\tilde{x}_{k+1}$ is the iterate obtained from $x_k$ if we were to run a gradient descent step. In particular, if we take the constant estimator satisfying $\delta_k = \nabla f(\tilde{x}_{k+1})$, then the additional terms simplify to:

$$\begin{aligned} & -2\alpha\langle \nabla f(\tilde{x}_{k+1}), \tilde{x}_{k+1} - x^* \rangle + \alpha^2 \left\| \nabla f(\tilde{x}_{k+1}) \right\|_2^2 \\ \text{(6)} \quad & \leq -2\alpha \frac{\mu L}{L+\mu} \left\| \tilde{x}_{k+1} - x^* \right\|_2^2 + \alpha \left( \alpha - \frac{2}{L+\mu} \right) \left\| \nabla f(\tilde{x}_{k+1}) \right\|_2^2 \end{aligned}$$

so that under the same step size constraint as in gradient descent, we obtain an additional negative term, and faster convergence (is there a connection with Nesterov acceleration ?).

If we ignore the first term for now, then we are left with the mean squared error term, which suggests that the estimator that would result in the fastest convergence is the one that minimizes this error. In particular, unbiased estimators are unlikely to be optimal, despite their widespread use. They are, on the other hand, the easiest to analyze.

Variance reduction algorithm such as SAGA [2], SAG [9], and SVRG [5] work by decreasing the above additional terms fast enough so that they recover the linear convergence rate, with worse constants, but better overall complexity. SAGA and SVRG both use unbiased estimates of the gradient and reduce the variance, whereas SAG uses a biased estimate that reduces the total mean square error. This is one of the reasons why the analysis of SAG is more complicated than that of the other unbiased algorithms. Experimentally, SAG is found to perform as well as or better than the other algorithms [4], which can be explained by the fact that it reduces the mean squared error more efficiently.

## 6 Discussion

We constructed SGD from gradient flow in two steps, the first being the discretization of gradient flow and the second being the choice of the estimator. Using this construction, one can build many variants of SGD by picking a discretization scheme and a different estimator. We argued that the Euler discretization scheme used might not be optimal in cases where we know a lot about the function we would like to minimize, for example, when minimizing the negative log likelihood of a generalized linear models. We also argued that the choice of an unbiased estimator made in SGD might not be optimal since the rate of convergence of the algorithm depends on the mean squared error. We ended our discussion by explaining how variance reduction algorithms are able to recover the linear convergence rate of gradient descent.

## References

[1] Léon Bottou, Frank E. Curtis, and Jorge Nocedal. Optimization Methods for Large-Scale Machine Learning. jun 2016.

[2] Aaron Defazio Ambiata, Francis Bach, and Simon Lacoste-Julien. SAGA: A Fast Incremental Gradient Method With Support for Non-Strongly Convex Composite Objectives. Technical report.

[3] Nidham Gazagnadou, Robert M. Gower, and Joseph Salmon. Optimal mini-batch and step sizes for SAGA. jan 2019.

[4] Robert M. Gower, Peter Richtárik, and Francis Bach. Stochastic Quasi-Gradient Methods: Variance Reduction via Jacobian Sketching. may 2018.

[5] Rie Johnson and Tong Zhang. Accelerating Stochastic Gradient Descent using Predictive Variance Reduction. Technical report.

[6] Yi-An Ma, Niladri Chatterji, Xiang Cheng, Nicolas Flammarion, Peter Bartlett, and Michael I. Jordan. Is There an Analog of Nesterov Acceleration for MCMC? feb 2019.

[7] Yu Nesterov. Introductory Lectures on Convex Programming Volume I: Basic course. Technical report, 1998.

[8] Lam M. Nguyen, Phuong Ha Nguyen, Marten van Dijk, Peter Richtárik, Katya Scheinberg, and Martin Takáč. SGD and Hogwild! Convergence Without the Bounded Gradients Assumption. feb 2018.

[9] Mark Schmidt, Nicolas Le Roux, and Francis Bach. Minimizing Finite Sums with the Stochastic Average Gradient. sep 2013.

[10] Weijie Su, Stephen Boyd, and Emmanuel J Candès. A Differential Equation for Modeling Nesterov's Accelerated Gradient Method: Theory and Insights. Technical report, 2015.

[11] Max Welling, D Bren, and Yee Whye Teh. Bayesian Learning via Stochastic Gradient Langevin Dynamics. Technical report, 2010.