A mathematical analysis of machine learning algorithms.

Summer research supervised by Professor Greg Pavliotis

Pantelis Tassopoulos

Summer 2023

Contents

1 overview

1.1 objectives

- To examine the existing literature on recent developments in the context of theoretical machine learning that integrate tools from statistical physics and probability theory, i.e., the theory of interacting particle systems. | Engineering and Phys- ical Sciences Research Council (EPSRC) | Engineering and Phys- ical Sciences Research Council (EPSRC) item To analyse the approximation quality and trainability of neural networks using algorithms, such as Stochastic Gradient Descent (SGD), informed by such ideas on toy models and examples with real life examples such as the MNIST digit classification dataset.
- To perform numerical experiments by training neural networks under various circumstances, thereby graining practical insights.
- To try and extend results from the literature by attempting to provide theoretical guarantees for accuracy and robustness of machine learning algorithms other than SGD or new insights from numerical simulations.

1.2 Funding

I was awarded the Engineering and Physical Sciences Research Council (EPSRC) Vacation Bursary of č3024 to pursue this project.

1.3 outcomes

This Summer Project (UROP) gave me a better insight into cutting-edge research in theoretical machine learning and mathematical optimisation.

I reviewed the requisite background material in mathematics from reference material, including textbooks and relevant papers. For instance, I read up on topics in probability, namely, martingale inequalities (Doob's and Hoeffman's inequalities in the book of Bremaud entitled 'Probability Theory and Stochastic Processes' [1]) that Mei et al. in their 2018 paper entitled 'A mean-field view of the landscape of two-layer neural networks' used in proofs of convergence of the SGD dynamics to the evolution of a Partial Differential Equation (PDE) as the hidden layer had an ever-increasing number of nodes, which enabled to perform novel theoretical analyses and provide theoretical guarantees of convergence.

I did some additional reading to supplement my understanding of the 2019 papers by Spiliopoulos and Sirgiano entitled 'Mean Field Analysis of Neural Networks: A Law of Large Numbers' and its companion paper [12], [13] . I read part of the book entitled 'Markov Processes: Characterisation and Convergence' by Stewart N. Ethier Thomas G. Kurtz [5], specifically the chapter on weak convergence of probability measures with values on the Skorokhod space $\mathcal{D}_{E[0,\infty)}$, which was necessary for understating the author's arguments on propagation of chaos and analogous convergence arguments.

Another crucial component of the project was the emphasis on numerical experiments. They allowed me to demonstrate the validity of theoretical findings and strengthen the case for the arguments presented. Numerical simulations involved training neural networks using existing algorithms from the literature and using insights gained to develop new algorithms.

For instance, regarding the above papers by Spiliopoulos, to supplement my understanding and empirically demonstrate claims made in the above paper, I performed numerical simulations by training a family of single-layer neural networks that achieved single-digit classification on the MNIST data set (used for digit classification and is a well-known benchmark for testing models). Upon expanding their hidden layer and training them, I plotted histograms of the distribution parameters which, for sufficiently many hidden nodes, the distribution of node values seemed to stabilise around a fixed bimodal distribution, which is also what the authors reported (while they did not specify the exact nature of the neural network they trained).

The project's theme shifted from analysing plain SGD towards understanding the wildly non-convex landscape of the underlying objective/loss function one typically encounters in machine learning applications.

In this direction, I demonstrated, among other observations, which can be found on my GitHub page [14] using numerical simulations that Nesterov accelerated gradient descent escaped a 'bad minimum', where SGD got stuck in a loss function that was constructed in [8].

This motivated me to generalise further insights gained by examining the dynamics of SGD to momentum-based algorithms, including Nesterov's accelerated Gradient Descent.

At that time, I read the 2017 paper by Chaudhari et al. entitled 'Deep Relaxation: partial differential equations for optimising deep neural networks' [2]. They introduced various approaches centred around 'regularising' the loss function. I incorporated both momentum-based methods (including 'restarting' the momentum if the gradient in the change in position was in the direction of the gradient-maximal increase, as was introduced in the 2012 Candes et al. paper entitled 'Adaptive Restart for Accelerated Gradient Schemes' [10]) and regularising the potential (by leveraging the analytical properties of solutions to the Hamilton-Jacobi-Bellman equation) as suggested above to create an algorithm that attempted to escape bad minima.

I also revisited the 2023 paper by Andrew Stuart et al. entitled 'Gradient Flows for Sampling: Mean-Field Models, Gaussian, Approximations and Affine Invariance'[3] initially suggested

by my supervisor to produce another algorithm based on theoretical insights gained from the paper.

Furthermore, I read the paper coauthored by my supervisor entitled 'The sharp, the flat and the shallow: Can weakly interacting agents learn to escape bad minima?' [6] I was also led to study the analysis of multiscale algorithms in the literature, e.g. in the Weinan et al. (2005) paper [4] on the analysis of multiscale methods for SDEs. The authors devised an algorithm that escaped bad minima in a toy example they introduced in the paper.

As suggested by my supervisor, I implemented the above algorithms by performing descent on a loss that was a Muller-Brown potential (the canonical example of a potential surface in theoretical chemistry). My instance had a narrow global minimum; SGD would perform poorly and tend to converge to two local minima, of which there were two in a relatively confined domain. For each algorithm mentioned above, I performed random initialisations, ran the algorithms for a fixed number of steps and recorded the final 'losses'. One notable observation is that the implementation of the algorithm in my supervisors paper performed noticeably better than the rest, including plain SGD.

This research experience presented an excellent opportunity for me to go beyond the scope of material covered in class and explore developments in the literature in a structured and rigorous manner.

Note all the code referenced herein can be found on my personal GitHub page [14] .

2 Mean-Field Limits of Neural Networks

2.1 Background theory

The process of a neural network 'learning' from data requires solving a complex optimisation problem with millions of variables. This is done by stochastic gradient descent (SGD) algorithms. One can study the case of two-layer networks and derive a compact description of the SGD dynamics in terms of a limiting partial differential equation. This a major insight in [8], where they authors also suggest with their findings that SGD dynamics do not become more complex when the network size increases.

Now, more formally, one typically encounters, in the context of supervised learning the following:

- Observed data points $(x_i, y_i)_{i\in\mathbb{N}} \subseteq \mathbb{R}^d \times \mathbb{R}$, where they are modelled as being independent and indentically distributed (iid).
- The $x \in \mathbb{R}^d$ are called feature vectors and the $y \in \mathbb{R}$ the labels.
- The neural network essentially is a function that depends on some hidden parameters and the feature vector. In the case of a two-layer neural network, the dependence is modelled by:

$$\begin{aligned} \hat{y} : \mathbb{R}^d \times \mathbb{R}^{ND} &\to \mathbb{R} \\ (x;\theta) &\mapsto \tfrac{1}{N}\sum_{i=1}^{N} \sigma_*(x;\theta_i) \end{aligned} \tag{1}$$

  where N is the number of hidden units (neurons), $\sigma_* : \mathbb{R}^d \times \mathbb{R}^D \to \mathbb{R}$ an activation function and $\theta = (\theta)_{i\leqslant N}$, $\theta_i \in \mathbb{R}^D$ are parameters, often $\theta_i = (a_i, b_i, w_i)$ for real a_i, b_i, w_i and $\sigma_*(x;\theta_i) = a_i\sigma(\langle x, w_i\rangle + b_i)$ for some function $\sigma : \mathbb{R} \to \mathbb{R}$ (see figure 1).

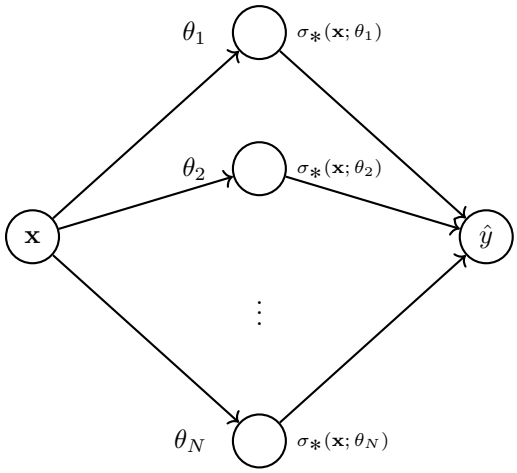


Figure 1: Illustration of a two layer neural network.

Naturally, one wants to chose parameters θ so as to minimise the risk function

$$R_N(x;\theta) = \mathbb{E}[\ell(y,\hat{y}(x;\theta))] \tag{2}$$

for a loss function $\ell : \mathbb{R}\times\mathbb{R}\to\mathbb{R}$, typically and in our case the square loss $\ell(y-\hat{y}) = (y-\hat{y})^2$. This is achieved in practice by stochastic gradient descent summarised below:

Stochastic Gradient Descent (SGD)

Initialise the parameters $(\theta_i)_{i\leqslant\mathbb{N}} \sim \rho_0$, that is according to some initial distribution ρ_0.
while loss is greater than tolerance **do**
  Generate iid sample $(x,y)\sim\mathbb{P}$
  for $1\leqslant i\leqslant N$ **do**
    $\theta_i \leftarrow \theta_i + 2s\cdot(y-\hat{y}(x;\theta))\cdot\nabla_{\theta_i}\sigma_*(x;\theta_i)$ ▷ square loss is used
    Update learning rate s
  end for
end while

Observe that we have the alternative characterisation of the loss

$$R_N(\theta) = R_\# + \frac{2}{N}\sum_{i=1}^{N} V(\theta_i) + \frac{1}{N^2}\sum_{i,j=1}^{N} U(\theta_i,\theta_j). \tag{3}$$

where $V(x;\theta) = -\mathbb{E}[y\cdot\sigma_(x:\theta)]$, $U(\theta_1,\theta_2) = \mathbb{E}[\sigma_(x;\theta_1)\cdot\sigma_*(x;\theta_2)]$ and $R_\# = \mathbb{E}[y^2]$ is the risk of the trivial predictor $\hat{y}=0$.

Notice that the collection of weights $\theta\in\mathbb{R}^{ND}$ induces a probability measure on $\mathbb{R}^{ND}$, namely its *empirical measure*:

$$\hat{\rho}^{(N)} = \frac{1}{N}\sum_{i=1}^{N}\delta_{\theta_i} \tag{4}$$

Consider the function on the space of probability measures on $\mathbb{R}^D$, $\mathcal{P}(\mathbb{R}^D)$:

$$R : \mathcal{P}(\mathbb{R}^D)\to\mathbb{R}$$
$$\rho \mapsto R_\# + 2\int V(\theta)\rho(\mathrm{d}\theta) + \int\int U(\theta_1,\theta_2)\rho(\mathrm{d}\theta_1)\rho(\mathrm{d}\theta_2)$$

Observe we can thus express $R_N(\theta) = R(\hat{\rho}^{(N)})$. Now, performing the SGD algorithm 2.1 for k steps say (with step size $s_k = \epsilon\cdot\xi(k\epsilon)$ for some $\epsilon>0$ and $\xi:\mathbb{R}_{\geqq 0}\to\mathbb{R}_{\geqq 0}$ sufficiently regular-see 2.1), we obtain the parameters $(\theta^k_{i\leqslant N})$ and their respective empirical measures $\hat{\rho}^{(N)}_k$. In [8], Theorem 2.1 here, it is shown that for all $t\geqq 0$, as $N\to\infty$ and $\epsilon\to 0$ in an appropriate way, the empirical

measures $\hat{\rho}^{(N)}_{t/\epsilon}$ converge in the weak sense to some probability measure ρ_t whose dynamics are governed by the following PDE, which is referred to as *distributional dynamics* (DD) in [8]

$$\begin{aligned} \partial_t \rho_t &= 2\xi(t)\, \nabla_\theta \cdot \Big(\rho_t \nabla_\theta \Psi(\theta; \rho_t) \Big), \\ \Psi(\theta; \rho) &\equiv V(\theta) + \textstyle\int U(\theta, \theta')\, \rho(\mathrm{d}\theta'). \end{aligned} \tag{5}$$

(Note that $\nabla_\theta \cdot \mathbf{v}(\theta)$ denotes the divergence of the vector field $\mathbf{v}(\theta)$). This should be interpreted as an evolution equation in $\mathcal{P}(\mathbb{R}^D)$.

There is rich mathematical literature on the PDE 5 which was motivated by the study of interacting particle systems in mathematical physics (see the references in [8]). The authors in [8] use this to observe that 5 can be viewed as a gradient flow for the cost function $R(\rho)$ in the space $(\mathcal{P}(\mathbb{R}^D), W_2)$, of probability measures on $\mathbb{R}^D$ endowed with the Wasserstein metric.

Aside:
Regarding Wasserstein flows, I looked through the paper by Y. Chen , et al. [3] on Gradient Flows for Sampling and noted down some key insights from their paperIn brief, they study the problem of sampling a probability distribution with an unknown normalization constant, which is a fundamental problem in computational science and engineering. They recast it as an optimisation problem on the space of probability measures, using gradient flows.

- Given a gradient flow that one has constructed wrt a posterior distribution that one wants to sample from without having an explicit normalization, one can formulate a gradient flow and a system of particles with SDE of the MCKean Vlasov type with FK equation the gradient flow (i.e. the evolution equation of the density).
- By making the gradient flow invariant under affine reparameterizations (through preconditioning or by suitable choice of metric or energy functional on $\mathcal{P}(\mathbb{R}^d)$), one hope to improve performance of algorithms in the case of highly anisotripic posteriors, if there is an affine transformation that reduces the anisotropic nature of said posterior.

Recall that Wasserstein distance is defined as

$$W_2(\rho_1, \rho_2) = \Big(\inf_{\gamma \in \mathcal{C}(\rho_1, \rho_2)} \int \|\theta_1 - \theta_2\|_2^2 \gamma(\mathrm{d}\theta_1, \mathrm{d}\theta_2) \Big)^{1/2}. \tag{6}$$

In order to establish that these PDEs indeed describe the limit of the SGD dynamics, we make the following assumptions.

A1. $t \mapsto \xi(t)$ is bounded Lipschitz: $\|\xi\|_\infty, \|\xi\|_{\mathrm{Lip}} \leqslant K_1$, with $\int_0^\infty \xi(t)\, \mathrm{d}t = \infty$.

A2. The activation function $(\mathbf{x}, \theta) \mapsto \sigma_*(\mathbf{x}; \theta)$ is bounded, with sub-Gaussian gradient: $\|\sigma_*\|_\infty \leqslant K_2$, $\|\nabla_\theta \sigma_*(\mathbf{X}; \theta)\|_{\psi_2} \leqslant K_2$. Labels are bounded $|y_k| \leqslant K_2$.

A3. The gradients $\theta \mapsto \nabla V(\theta)$, $(\theta_1, \theta_2) \mapsto \nabla_{\theta_1} U(\theta_1, \theta_2)$ are bounded, Lipschitz continuous (namely $\|\nabla_\theta V(\theta)\|_2$, $\|\nabla_{\theta_1} U(\theta_1, \theta_2)\|_2 \leqslant K_3$, $\|\nabla_\theta V(\theta) - \nabla_\theta V(\theta')\|_2 \leqslant K_3 \|\theta - \theta'\|_2$, $\|\nabla_{\theta_1} U(\theta_1, \theta_2) - \nabla_{\theta_1} U(\theta_1', \theta_2')\|_2 \leqslant K_3 \|(\theta_1, \theta_2) - (\theta_1', \theta_2')\|_2$).

Theorem 2.1 (PM. Nguyen et al. (2018)). *Assume that conditions* A1, A2, A3 *hold. For* $\rho_0 \in \mathcal{P}(\mathbb{R}^D)$, *consider SGD with initialization* $(\mathbf{theta}_i^0)_{i\leqslant N} \sim_{iid} \rho_0$ *and step size* $s_k = \epsilon\xi(k\epsilon)$. *For* $t \geqslant 0$, *let* ρ_t *be the solution of PDE 5. Then, for any fixed* $t \geqslant 0$, $\hat{\rho}^{(N)}_{\lfloor t/\epsilon \rfloor} \Rightarrow \rho_t$ *almost surely along any sequence* $(N, \epsilon = \epsilon_N)$ *such that* $N \to \infty$, $\epsilon_N \to 0$, $N/\log(N/\epsilon_N) \to \infty$ *and* $\epsilon_N \log(N/\epsilon_N) \to 0$. *Further, there exists a constant* C *(depending uniquely on the parameters* K_i *of conditions* A1-A3*) such that, for any* $f : \mathbb{R}^D \times \mathbb{R} \to \mathbb{R}$, *with* $\|f\|_\infty, \|f\|_{Lip} \leqslant 1$, $\epsilon \leqslant 1$,

$$\begin{aligned} \sup_{k\in[0,T/\epsilon]\cap\mathbb{N}} \Big|\frac{1}{N}\sum_{i=1}^N f(\theta_i^k) - \int f(\theta)\rho_{k\epsilon}(\mathrm{d}\theta)\Big| &\leqslant Ce^{CT}\, Err_{N,D}(z)\,, \\ \sup_{k\in[0,T/\epsilon]\cap\mathbb{N}} \big|R_N(\theta^k) - R(\rho_{k\epsilon})\big| &\leqslant Ce^{CT}\, Err_{N,D}(z)\,, \end{aligned} \tag{7}$$

with probability $1 - e^{-z^2}$ *where* $Err_{N,D}(z)$ *is given by*

$$\sqrt{1/N \vee \epsilon} \cdot \left[\sqrt{D + \log N/\epsilon} + z\right] \tag{8}$$

Theorem 2.2 (Doob's martingale inequality). *Let* $(\mathcal{F}_t)_{t\geqslant 0}$ *be a filtration on the probability space* $(\Omega, \mathcal{F}, \mathbb{P})$ *and* $(M_t)_{t\geqslant 0}$ *be a continuous martingale adapted to the filtration* $(\mathcal{F}_t)_{t\geqslant 0}$. *Let* $p \geqslant 1$ *and* $T > 0$. *If* $\mathbb{E}[|M_T|^p] < \infty$ *and* $\lambda > 0$, *then*

$$\mathbb{P}\left(\sup_{t\in[0,T]} |M_t| \geqslant \lambda\right) \leqslant \frac{\mathbb{E}[|M_T|^p]}{\lambda^p} \tag{9}$$

Lemma 2.3 (Hoeffding's Lemma). *Let* $(M_n)_{n\in\mathbb{N}}$ *be a martingale adapted to the filtration* $(\mathcal{F}_n)_{n\in\mathbb{N}}$ *such that for some sequence* $c_1, c_2, \cdots$ *of real numbers*

$$\mathbb{P}(|M_n - M_{n-1}| \leqslant c_n) = 1 \quad \textit{for all } n \in N. \tag{10}$$

Then for all $x \geqslant 0$ *and all* $n \geqslant 1$,

$$\mathbb{P}(|M_n - M_{n-1}| \geqslant x) \leqslant 2\exp\left(-\frac{1}{2}x^2 \Big/ \sum_{i=1}^n c_i^2\right) \tag{11}$$

Proof. (Rough Sketch) The conditions A1 and A3 guarantee the existence and uniqueness of solutions to the PDE 5, interpreted in the weak sense. The discrete SGD dynamics $(\theta_i^k)_{i\leqslant N}$ approximate the continuous time dynamics. Then the proof becomes technical and the aim is to control error terms incurred when comparing the deviation of the discrete and continuous dynamics in probability.

Notice we can also re-express 7 in terms of the empirical measure to deduce for all function f with $\|f\|_{\mathrm{Lip}} \leqslant 1$, $\pi \in \mathcal{C}(\hat{\rho}_k^N, \rho_{k\epsilon})$ and $k \in [0, T/\epsilon]$

$$\begin{aligned} \Big|\int f(\theta)\hat{\rho}_k^N(\mathrm{d}\theta) - \int f(\theta)\rho_{k\epsilon}(\mathrm{d}\theta)\Big| &\leqslant \int |f(\theta) - f(\phi)|\pi(\mathrm{d}\theta, \mathrm{d}\phi) \\ &\leqslant \int \|\theta - \phi\|_2\, \pi(\mathrm{d}\theta, \mathrm{d}\phi) \leqslant W_2(\hat{\rho}_k^N, \rho_{k\epsilon}) \end{aligned} \tag{12}$$

using Cauchy-Schwarz and taking the infimum over such couplings. Hence we obtain the bound

$$\sup_{k\in[0,T/\epsilon]\cap\mathbb{N}} \Big|\int f(\theta)\hat{\rho}_k^N(\mathrm{d}\theta) - \int f(\theta)\rho_{k\epsilon}(\mathrm{d}\theta)\Big| \leqslant \sup_{k\in[0,T/\epsilon]\cap\mathbb{N}} W_2(\hat{\rho}_k^N, \rho_{k\epsilon}) \tag{13}$$

This estimate helps one get a sense of the terms that need to be controlled in the proof of the theorem.

Moreover, the sub-gaussianity and Lipscitz continuity feature prominently and the tools used to achieve bounds on the probabilities are mainly Doob's maximal inequality 2.2 and Hoefdding's lemma 2.3. □

The PDE formulation leads to several insights and simplifications. One can exploit symmetries in the data distribution $\mathbb{P}$ for instance. If $\mathbb{P}$ has rotational symmetry, then one can look for solutions to the PDE problem that share such rotational symmetry, thereby reducing the dimensionality of the problem which facilitates theoretical and numerical analysis. This is manifest in the case of two isotropic Gaussians considered later. Such symmetry cannot be achieved when considering the discrete dynamics since no set of poitns $\theta_1, \cdots, \theta_N \in \mathbb{R}^d$ is invariant under rotations (excluding trivial cases).

2.2 MNIST data classification

I studied the proofs for the propagation of chaos and the mean-field limit of the distribution of neural network weights in the 2019 paper of Spiliopoulos and Sirgiano entitled Mean Field Analysis of Neural Networks: A Law of Large Numbers and its companion paper [13], [12]. Note there are many similarities in the framerowk between these papers and the work in [8], though the gradient of the loss being Lipschitz is dropped. I did some background reading to supplement my understanding of the above papers of a book entitled Markov Processes: Characterization and Convergence by Stewart N. Ethier, Thomas G. Kurtz [5], specifically the chapter on weak convergence of probability measures with values in Skorokhod spaces defined below.

Definition 2.4 (Skorokhod space)**.** *Let $E = (\mathcal{M}, d)$ be a metric space and $T > 0$. Then, we define the Skorokhod space*

$$\mathcal{D}([0,T]; E) := \{f : [0,T] \to E : f \text{ is cadlag}\}. \tag{14}$$

This mean field convergence of the empirical measures induced by the weights of neural networks was also performed in papers [12] and [13]. In a similar setup to [8], the SGD algorithm produces obtains empirical measures expended in a piewewise constant manner to $\mu_t^N = \hat{\rho}^N_{[Nt]}$ for $t \geqslant 0$, see figure 2.

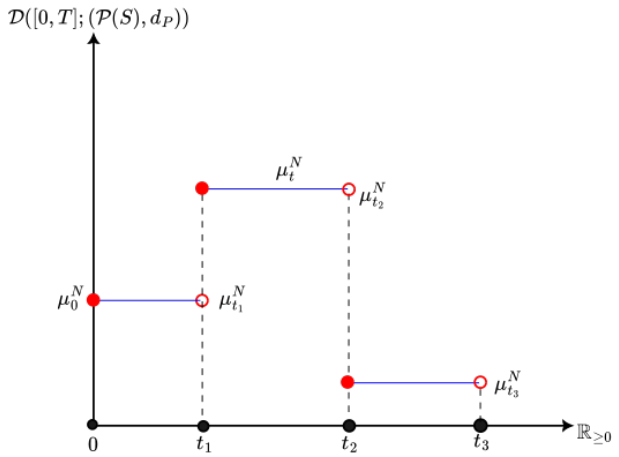


Figure 2: Piecewise constant extension of scaled empirical measures μ_t^N.

Now, by construction, we have that the empirical measure process $(\mu_t^N)_{t\in[0,T]}$ is an element of the space of locally finite Borel measures on $\mathbb{R}^d$, $\mathcal{M}(\mathbb{R})^d$. One can define the notion of *vague convergence* of a family $(\nu_n)_{n\in\mathbb{N}} \xrightarrow{v} \nu \in \mathcal{M}(\mathbb{R}^d)$ by

$$\int_{\mathbb{R}^d} f \, \mathrm{d}\nu_n \overset{n\to\infty}{\to} \int_{\mathbb{R}^d} f \, \mathrm{d}\nu_n \quad f \in \hat{\mathcal{C}}(\mathbb{R}^d). \tag{15}$$

where $\hat{\mathcal{C}}_(\mathbb{R}^d)$ denotes the space of all bounded continuous non-negative functions with bounded support. Note that the family of maps $\{\pi_f : f \in \hat{\mathcal{C}}_{\mathbb{R}^d}\}$ induces the vague topology $\mathcal{T}$:

Lemma 2.5 (Vague topology on $\mathcal{M}_{\mathcal{S}}$)**.** *Let $\mathcal{S}$ be a complete separable metric space, then there exists a topology $\mathcal{T}$ on $\mathcal{M}_{\mathcal{S}}$ such that*

- *$\mathcal{T}$ induces the convergence $\nu_n \xrightarrow{v} \nu$ in 15,*
- *$\mathcal{M}_{\mathcal{S}}$ is Polish under $\mathcal{T}$,*
- *$\mathcal{T}$ generates the Borel sigma algebra $\sigma(\{\pi_f : f \in \hat{\mathcal{C}}_{\mathbb{R}^d}\})$.*

Hence, we have that the scaled empirical measure $(\mu_t^N)_{t\in[0,T]})$ is a random element of $\mathcal{D}_{\mathcal{M}(\mathbb{R})^d} := \mathcal{D}([0,T];(\mathcal{M}(\mathbb{R}^d), d_{\mathcal{T}}))$, where $d_{\mathcal{T}}$ is the induced metric from 2.5. Note that $\mathcal{D}_{\mathcal{M}(\mathbb{R})^d}$ space is a Polish space in its own space endowed with the *Skorokhod topology* with well-understood criteria for compactness that feature prominently in the proof of the main theorem in [13], and [12].

The main result of the paper [13] concerns the convergence in distribution of μ_t^N in the aforementioned Skorokhod space under certain the 'reasonable' structural assumptions

S1. The activation function $\sigma \in C_b^2(\mathbb{R})$, i.e. σ is twice continuously differentiable and bounded.

S2. The sequence of data samples (x_k, y_k) is i.i.d. from a probability distributed $\pi(dx, dy)$ such that $\mathbb{E} \parallel x_k \parallel^4 + \mathbb{E}|y_k|^4$ is bounded.

S3. The randomly initialized parameters (c_0^i, w_0^i) are i.i.d. with a distribution $\bar{\mu}_0$ such that $\mathbb{E}[\exp\left(q|c_0^i|\right)] < C$ for some $0 < q < \infty$ and $\mathbb{E}[\parallel w_0^i \parallel^4] < C$.

Theorem 2.6 (Spiliopoulos LLN)**.** *For all $T > 0$, the scaled empirical measure μ_t^N on $[0,T]$ converges in distribution to a limit measure $\bar{\mu}_t$ with values in $\mathcal{D}_{\mathcal{M}_{\mathbb{R}^d}}$ as $N \to \infty$.*

Remark. μ_t has a characterisation as the unique deterministic weak solution to a PDE, interpreted in the weak sense. Also, since the limiting measure μ_t is deterministic for all $t \geqslant 0$, we have the stronger convergence in Probability, that is for all $\delta > 0$

$$\lim_{N\to\infty} \mathbb{P}(d_{\mathcal{D}_{\mathcal{M}_{\mathbb{R}^d}}}(\mu^N, \bar{\mu}) \geqslant \delta) = 0$$

Moreover, I read the companion paper of Spiliopoulos (2019) [12] where the authors proved a CLT for a one-layer neural network. To this end, the authors in [12] the fluctuation process

$$\eta_t^N = \sqrt{N}(\mu_t^N - \bar{\mu}_t) \tag{16}$$

The main result in [12] is that asymptotically, as $N \to \infty$, the fluctuations converge in distribution, in a way made precise below, to some measure-valued process $\bar{\eta}$, where satisfies a stochastic partial differential equation. This result achieves to give a characterisation of the fluctuations of the finite empirical measure μ^N around its mean-field limit $\bar{\mu}$ for large N . It is noted that the $\bar{\eta}$ has a Gaussian distribution.

Theorem 2.7 (Spiliopoulos CLT)**.** *Under the 'reasonable' assumptions 2.2, $J \geqslant 3\left\lceil \frac{d}{2} \right\rceil + 7$ and any $0 < T < \infty$. The sequence*

$$((\eta_t^N)_{t\in[0,T]})_{N\in\mathbb{N}} \xrightarrow{d} ((\bar{\eta}_t)_{t\in[0,T]})_{N\in\mathbb{N}} \tag{17}$$

in $\mathcal{D}([0,T];W^{-J,2})$, as $\mathbb{N} \to \infty$ where $W^{-J,2}$ is the space of all continuous linear functionals on the Sobolev space $W_0^{J,2}(\Theta)$, where $\Theta \subseteq \mathbb{R}^d$ is a bounded domain independent of N.

Remark. For a brief introduction into the Sobolev spaces mentioned above, refer to section 2 to in [12].

In [13], the authors prove that the neural network has the "propagation of chaos" property under suitable structural assumptions mentioned therein.

Theorem 2.8. *Consider $T < \infty$ and let $t \in (0, T]$. Define the probability measure $\rho_t^N \in \mathcal{M}(\mathbb{R}^{(1+d)N})$ where*

$$\rho_t^N(dx^1, \dots, dx^N) = \mathbb{P}\left[(c^1_{\lfloor Nt \rfloor}, w^1_{\lfloor Nt \rfloor}) \in dx^1, \dots, (c^N_{\lfloor Nt \rfloor}, w^N_{\lfloor Nt \rfloor}) \in dx^N\right].$$

Then, the sequence of probability measures $\rho_\cdot^N$ is $\bar{\mu}_\cdot$-chaotic. That is, for $k \in \mathbb{N}$

$$\lim_{N\to\infty} \langle f_1(x^1) \times \cdots \times f_k(x^k), \rho_\cdot^N(dx^1, \dots, dx^N)\rangle = \prod_{i=1}^{k} \langle f_i, \bar{\mu}_\cdot\rangle, \quad \forall f_1, \dots, f_k \in C_b^2(\mathbb{R}^{1+d}). \tag{18}$$

This means that as $N \to \infty$, the neural network converges (in probability) to a deterministic model. This is despite the fact that the neural network is randomly initialized and it is trained on a random sequence of data samples via stochastic gradient descent. The propagation of chaos result (18) indicates that, as $N \to \infty$, the dynamics of the weights (c_k^i, w_k^i) will become independent of the dynamics of the weights (c_k^j, w_k^j) for any $i \neq j$. Note that the dynamics (c_k^i, w_k^i) are still random due to the random initialization. However, the dynamics of the i-th set of weights will be uncorrelated with the dynamics of the j-th set of weights in the limit as $N \to \infty$.

Thus implemented a single-digit classifier neural network with a single hidden layer satisfying the assumptions 2.2, as discussed above using a sigmoid activation function, trained on the MNIST data set containing around $60,000$ images of hand-drawn digits. I tried to establish numerically whether one obtains convergence of the (asymptotically identical) distribution of any one of the parameters $(c_i)_{1\leqslant i\leqslant N}$, see figure 3. The result seems to match that presented in [13].

3 Non-convex landscape

3.1 Approaches

At this point in the project, the focus started to shift from the theoretical mean-field analysis of neural network algorithms towards studying possible approaches to alleviate the failure of STD to reach a global minimum by potentially getting stuck in very sharp, yet non-global minima when the potential is wildly non-convex.

In this direction, I read my supervisor's paper on shallow minima: The sharp, the flat and the shallow: Can weakly interacting agents learn to escape bad minima?, [6]. In the paper, the authors review several variants of SGD and illustrate that a system of interacting, rather than i.i.d., agents (essentially an interacting particle system) performing gradient descent can help to smooth out sharp minima and thus implicitly convexify the loss function.

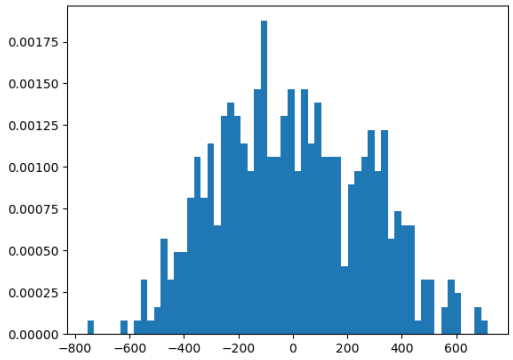
The setting is a modification of the Stochastic Gradient Langevin Dynamics (SGLD) framework:

$$\mathrm{d}X_t = -\nabla\Phi(X_t)\,\mathrm{d}t + \sqrt{2}\beta\,\mathrm{d}B_t, \quad X_0 \sim \eta_0 \tag{19}$$

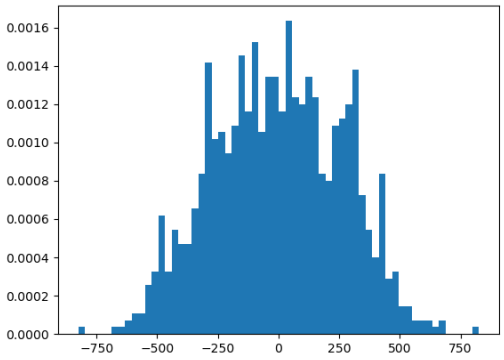
where Φ the loss function B_t is a standard Brownian motion and η_0 is the initial distribution. Three approaches are discussed before a synthesis of the last two yields their proposed algorithm, see 3.1.4.

3.1.1 Regularise potential directly by convolution

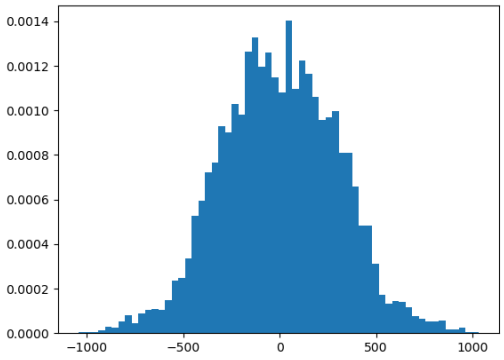
To eliminate sharp local minima one could replace the gradient term in the basic gradient descent algorithm with a smoother version. In order to eliminate these local minima one could



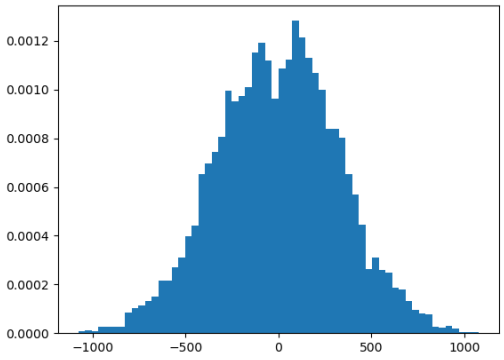
(a) $N = 10^2$, 10^2 epochs.



(b) $N = 10^3$, 10^3 epochs.



(c) $N = 5 \cdot 10^3$, $5 \cdot 10^3$ epochs.



(d) $N = 10^4$, 10^4 epochs.

Figure 3: Plots of histograms of parameters connecting hidden layer outputs to the final output for different values of N and epochs in line with Theorem 2.6.

simulate the gradient descent dynamics of a "smoothed" version of the cost function instead

$$dX_t = -\nabla \Phi^h(X_t)dt \tag{20}$$

where we denote

$$\Phi^h(y) = (G_h \star \Phi)(y) = \int G_h(y-x)\Phi(x)dx, \tag{21}$$

i.e. $\star$ denotes the convolution. A typical choice for the smoothing kernel G_h is the Gaussian kernel with variance h

$$G_h(z) = \frac{1}{(2\pi h)^{d/2}} \exp\left(-\frac{\|z\|^2}{2h}\right).$$

For technical conditions for the above modification of the gradient, see the references in [6]. Regardless of the choice of the smoothing kernel, Φ^h can be interpreted as an expectation

$$\Phi^h(x) = \int \Phi(x+y)\mu(dy),$$

for a suitably chosen probability measure μ. Furthermore, (under appropriate conditions)

$$\nabla \Phi^h(x) = \int \nabla \Phi(x+y)\mu(dy). \tag{22}$$

Loosely speaking the effect of μ here is to smooth Φ. It is natural to ask how one designs μ (or G_h) to get the desired effect of smoothing of Φ. There a multiple complications with such an approach, a pressing one being that computing the integral in 21 for the type of loss functions that appear in machine learning applications is intractable. To help mitigate these issues, the authors look at approaches where a smoothing measure μ does not act directly on Φ and is constructed from the stochastic process itself.

3.1.2 Regularise the potential implicitly weakly interacting agents

An alternative approach is to use interacting SGLD, as opposed to i.i.d. copies of the Langevin dynamics 19. In [6], a system of interacting SGLD of the form

$$dX_t^i = -\nabla \Phi(X_t^i)dt - \left(\nabla D \star \eta_t^N\right)(X_t^i)dt + \sqrt{2\beta^{-1}}dB_t^i, \tag{23}$$

where $i = 1, \dots, N$, $\eta_t^N = \frac{1}{N}\sum_{i=1}^N \delta_{X_t^i}$, $X_0^i \sim \eta_0(\cdot)$. Compared to the i.i.d. SGLD 19, the dynamics 23 uses $D(x,y)$ as an interaction potential, taken to be the so-called Curie-Weiss interaction

$$D(x,y) = \frac{\lambda}{2}\|x-y\|^2 \tag{24}$$

so that each particle experiences a linear attractive (mean reverting) force to the empirical mean of all particles

$$\nabla D \star \eta_t^N(X_t^i) = \lambda\left(X_t^i - \frac{1}{N}\sum_{j=1}^N X_t^j\right).$$

The framework developed here can be conceived of as an abstraction of popular machine learning algorithms, with ample references made in [6, p. 4].

Under appropriate assumptions on the loss function, and on the initial conditions the position of each agent converges, in the limit $N \to \infty$ to the solution of the McKean SDE

$$\begin{aligned} d\bar{X}_t &= -\nabla \Phi(\bar{X}_t)dt - \nabla D \star \eta_t(\bar{X}_t)dt + \sqrt{2\beta^{-1}}dB_t, \\ \eta_t &= \mathcal{L}aw\left(\bar{X}_t\right). \end{aligned}$$

The density of the law of the process $\bar{X}_t$ is given by the McKean-Vlasov equation:

$$\partial_t \eta = \nabla \cdot \left(\eta \nabla\left(\beta \log \eta + \Phi + D \star \eta\right)\right), \quad \eta(0,\cdot) = \eta_0(\cdot). \tag{25}$$

This approach uses $\Phi + D \star \tilde{\eta}$ instead of Φ and acts to regularise or smooth out the cost function. From an optimization point of view, substituting $-\nabla\Phi(x) - \nabla D \star \eta_t^N(x)$ and using a linear interaction for ∇D is equivalent to using an ℓ_2-penalty in the objective function for the constraint: $X_t^i = \frac{1}{N}\sum_{j=1}^N X_t^j$, for each agent i. Therefore, for an appropriate choice of the interaction strength λ, the objective function is approximately convex.

3.1.3 Homogenisation

In the previous section 3.1.2 the empirical measure η_t^N was used to smooth the potential based on empirical properties of interacting agents. In this section, the approach that was developed in [2] can be used to convert 19 into the following gradient descent algorithm:

$$d\tilde{X}_t = -\nabla\Phi^{\beta,\gamma}(\tilde{X}_t)dt, \quad \Phi^{\beta,\gamma}(x) = \int \Phi(x-y)\rho_\infty^x(dy), \tag{26}$$

is briefly discussed, where ρ_∞^x is the invariant measure of Y_t that appears in the limit when $\epsilon \to 0$ for the following fast/slow SDE system

$$dX_t = -\nabla\Phi(X_t - Y_t)dt \tag{27a}$$

$$dY_t = -\frac{1}{\epsilon}\left(\frac{1}{\gamma}Y_t - \nabla\Phi(X_t - Y_t)\right)dt + \sqrt{\frac{2\beta^{-1}}{\epsilon}}dB_t \tag{27b}$$

The parameter ϵ measures scale separation. The limit $\epsilon \to 0$ can be justified using multiscale analysis. Note that this is a gradient scheme for the modified loss function $\Phi(x - \frac{y}{\epsilon}) + \frac{1}{2\gamma}\left\|\frac{y}{\epsilon}\right\|^2$.

It is noted in [6] that γ acts as a regularization parameter, precisely like the inverse of the interaction strength λ in the previous section. We emphasize the similarities between 21 and 26. It is important to note that the smoothed loss function in 26 can also be calculated via convolution with a Gaussian kernel:

$$\Phi^{\beta,\gamma}(x) = \frac{1}{\beta}\log\Big(G_{\beta^{-1}\gamma} \star \exp(-\beta\Phi)\Big). \tag{28}$$

This is the Cole-Hopf formula for the solution of the viscous Hamilton-Jacobi equation with the loss function Φ as the initial condition, posed on the time interval $[0, \gamma]$. The larger γ is, the more regularized the effective potential (or relative entropy) $\Phi^{\beta,\gamma}(x)$ is.

Importantly for the authors in [6], in [2] there is an equivalent formulation to 27:

$$dX_t = -\frac{1}{\gamma}(X_t - Y_t)dt \tag{29a}$$

$$dY_t = -\frac{1}{\epsilon}\left(\nabla\Phi(Y_t) - \frac{1}{\gamma}(X_t - Y_t)\right)dt + \sqrt{\frac{2\beta^{-1}}{\epsilon}}dB_t. \tag{29b}$$

Here the regularized cost appears as $\Phi(\frac{y}{\epsilon}) + \frac{1}{2\gamma}\left\|x - \frac{y}{\epsilon}\right\|^2$. This form is more convenient for the numerical implementation and is the one that will be used in Algorithm 3.1.4.

3.1.4 Synthesis: combine both multi-scale analysis and weakly interacting gents for MF Hom SGLD

In brief this algorithm, 3.1.4 corresponds to a discretization of the dynamics of gradient descent against a potential with an ℓ_2 penalty and a regularized version of the original potential Φ, using the method introduced by Chaudhari et al. (2018). More precisely, combining 26 with 23 one obtains:

$$dX_t^i = -\frac{1}{\gamma}(X_t^i - Y_t^i)dt - \lambda\left(X_t^i - \frac{1}{N}\sum_{j=1}^N X_t^j\right)dt \tag{30}$$

$$dY_t^i = -\frac{1}{\epsilon}\left(\nabla\Phi(Y_t^i) - \frac{1}{\gamma}(X_t^i - Y_t^i)\right)dt + \sqrt{\frac{2\beta^{-1}}{\epsilon}}dW_t^i \tag{31}$$

This scheme was tested numerically in the context of learning for the single layer neural network (see Section 2.1) with a sufficiently small value of ϵ, to approximate better the homogenized limit, as per [6]. The theoretical justification of this algorithm requires the study of the joint limits $\epsilon \to 0$ and $N \to +\infty$ (see [6, p. 6] for details and references).

To discretise 30-31 effectively for small ϵ I followed [6] and used the heterogeneous multiscale method [4] in Algorithm 3.1.4:

MF Hom SGLD

Require: $\mathsf{X}_0^i \sim \eta_0, \lambda \sim 1\Delta > 0$ ▷ Δ is a step size

for $n \geqslant 1, i = 1, \dots, N$ **do**

Set $\mathsf{Y}_{n,0}^i = \mathsf{Y}_{n-1,m'+M-1}^i$;

for $m = 1, \dots, M$ **do**

$$\begin{aligned}\mathsf{Y}_{n,m}^i &= \mathsf{Y}_{n,m-1}^i - \tfrac{\delta}{\epsilon}\left(\nabla\Phi(\mathsf{Y}_{n,m-1}^i) - \tfrac{1}{\gamma}(\mathsf{X}_{n-1}^i - \mathsf{Y}_{n,m-1}^i)\right) \\ &\quad + \sqrt{\tfrac{2\beta^{-1}\delta}{\epsilon}} Z_{n,m}^i;\ Z_{n,m}^i \sim N(0, I).\end{aligned}$$

end for

Compute average $\mathcal{Y}_n^i = \frac{1}{(m'+M-1)} \sum_{m=m'}^{m'+M-1} \mathsf{Y}_{n,m-1}^i$

Update

$$\mathsf{X}_n^i = \mathsf{X}_{n-1}^i - \frac{1}{\gamma}(\mathsf{X}_{n-1}^i - \mathcal{Y}_n^i)\Delta - \lambda\left(\mathsf{X}_{n-1}^i - \frac{1}{N}\sum_{j=1}^{N}\mathsf{X}_{n-1}^i\right)\Delta$$

end for

3.1.5 Nesterov SGD

It is well-known that in the deterministic setting, the Nesterov gradient descent achieves acceleration over plain gradient descent. However, in the stochastic setting, this is not as clear [7]. I thus implemented a stochastic version of Nesterov's gradient descent algorithm below (where $\tilde{\nabla}$ denotes the stochastic gradient, that is the sample mean of a batch's worth of iid samples of data).

Nesterov SGD

Require: $m \in \mathbb{N}$, $\gamma \in (0, 1)$ and $\eta_1 > 0$

Initialise the parameters $(\theta_i)_{i \leqslant \mathbb{N}} \sim \rho_0$, that is according to some initial distribution ρ_0.

while loss is greater than tolerance **do**

Compute stochastic gradient $\tilde{\nabla}$ using a mini-batch of some predetermined size m.

$$\begin{aligned}\mathbf{w}_{t+1} &\leftarrow \mathbf{u}_t - \eta_1 \tilde{\nabla} f(\mathbf{u}_t), \\ \mathbf{u}_{t+1} &\leftarrow (1+\gamma)\mathbf{w}_{t+1} - \gamma \mathbf{w}_t.\end{aligned} \tag{32}$$

end while

I also came across the paper [10], which provided a scheme for the dynamic selection of the parameter γ in 3.1.5, called gradient restarting. They prove in the case of convex potentials, acceleration over plain Nesterov is achieved, and I implemented the following algorithm to test its performance in the non-convex case.

Gradient restarted Nesterov SGD

Require: $m \in \mathbb{N}$, $\eta_1 > 0$ and $\alpha = 1$

$\gamma = \frac{\alpha - 1}{\alpha + 2} < 1$

Initialise the parameters $(\theta_i)_{i \leqslant \mathbb{N}} \sim \rho_0$, that is according to some initial distribution ρ_0.

while loss is greater than tolerance **do**

Compute stochastic gradient $\tilde{\nabla} f(u_t)$ using a mini-batch of some predetermined size m.

$$\begin{aligned}\mathbf{w}_{t+1} &\leftarrow \mathbf{u}_t - \eta_1 \tilde{\nabla} f(\mathbf{u}_t), \\ \mathbf{u}_{t+1} &\leftarrow (1+\gamma)\mathbf{w}_{t+1} - \gamma \mathbf{w}_t.\end{aligned} \tag{33}$$

if $\tilde{\nabla} f(\mathbf{u}_t)^T \cdot (\mathbf{w}_{t+1} - \mathbf{w}_t) > 0$ **then** ▷ Adaptive restart

$\alpha \leftarrow 1$

end if

$\alpha \leftarrow \alpha + 1$
end while

3.1.6 MaSS algorithm

I also came across a paper entitled Accelerating SGD with momentum for over-parameterised learning by Liu and Belkin, where the authors claim that Nesterov SGD with any parameter selection does not in general provide acceleration over ordinary SGD (as opposed to the acceleration provided by the deterministic Nesterov GD over plain GD). There the authors come up with a modified algorithm which they call Momentum-added stochastic solver (MaSS) and prove that for a strongly convex loss, in a certain batch regime (batch size $< m_1^*$), i.e. the 'linear' regime MaSS out-performs both Nesterov and plain SGD. They observe that the larger the batch size, the closer the MaSS algorithm becomes to deterministic Nesterov gradient descent They also demonstrate numerically that MaSS also outperforms Nesterov and plain SGD on deep neural networks, which are non-convex. When testing examples, we use relatively small batch sizes to observe this acceleration.

$$\begin{aligned} \mathbf{w}_{t+1} &\leftarrow \mathbf{u}_t - \eta_1 \tilde{\nabla} f(\mathbf{u}_t), \\ \mathbf{u}_{t+1} &\leftarrow (1+\gamma)\mathbf{w}_{t+1} - \gamma \mathbf{w}_t + \underline{\eta_2 \tilde{\nabla} f(\mathbf{u}_t)}. \end{aligned} \tag{34}$$

Here, $\tilde{\nabla}$ represents the stochastic gradient. The step size η_1, the momentum parameter $\gamma \in (0,1)$ and the compensation parameter η_2 are independent of t. Following the authors in [7], I implemented the algorithm using an equivalent form for the update rule 34(introducing an additional variable $\mathbf{v}$):

MaSS accelerated SGD
Require: $m \in \mathbb{N}$, $\eta_1, \eta_2 > 0$ and $\alpha = 1$
$\gamma = \frac{\alpha - 1}{\alpha + 2} < 1$
Initialise the parameters $(\theta_i)_{i \leqslant \mathbb{N}} \sim \rho_0$, that is according to some initial distribution ρ_0.
while loss is greater than tolerance **do**
Compute stochastic gradient $\tilde{\nabla} f(u_t)$ using a mini-batch of some predetermined size m.

$$\begin{aligned} \mathbf{w}_{t+1} &\leftarrow \mathbf{u}_t - \eta_1 \tilde{\nabla} f(\mathbf{u}_t), \\ \mathbf{u}_{t+1} &\leftarrow (1+\gamma)\mathbf{w}_{t+1} - \gamma \mathbf{w}_t + \underline{\eta_2 \tilde{\nabla} f(\mathbf{u}_t)}. \end{aligned} \tag{35}$$

if $\tilde{\nabla} f(\mathbf{u}_t)^T \cdot (\mathbf{w}_{t+1} - \mathbf{w}_t) > 0$ **then** ▷ Adaptive restart
$\alpha \leftarrow 1$
end if
$\alpha \leftarrow \alpha + 1$
end while

I used a variety of batch sizes, from $m = 1$ to the classification problems in [8] and for the MNIST single-digit classification neural networks from [13], to $m = 10^2$ for the 3−sphere spin function and for the Muller Brown potential, classical variants of the algorithms are used, that is replacing the stochastic with the full gradient, corresponding to $m \to \infty$. We also use the gradient-adaptive restarting to reset γ roughly when the change in weights points in the direction of the gradient, whose benefits were expanded upon in [10].

3.2 Applications

3.2.1 Centred isotropic Gaussians

The authors in [8] were interested in numerically testing their PDE framework on the classification problem of Gaussians with the same mean. That is, assume the joint law $\mathbb{P}$ of $(\mathbf{x}, y)$ to

be:

$$\begin{aligned} &\text{with probability } 1/2 : y = +1, \mathbf{x} \sim N(0, (1+\Delta)^2 \cdot \mathrm{Id}_d) \\ &\text{with probability } 1/2 : y = -1, \mathbf{x} \sim N(0, (1-\Delta)^2 \cdot \mathrm{Id}_d) \end{aligned} \tag{36}$$

For the activation function set $\sigma(\mathbf{x}; \theta) = \sigma(\langle w, \mathbf{x} \rangle)$ where σ is a simple piecewise linear activation function.

To try and reproduce the findings in [8],I implemented the SGD and the asymptotic PDE for the isotropic Gaussian case (SGD for isotropic gaussians with 10^7 iterations) with $(w_i^0)_{i \leqslant \mathbb{N}} \sim_{iid} \rho_0$, where ρ_0 is spherically symmetric. More specifically, I ran a monte carlo simulation of the discrete SGD dynamics and recorded the distance of the particles (hidden parameters θ) after a set amount of iterations and aggregated them, producing figure 4. It compares nicely with the corresponding simulation in the paper by Nguyen et al. [8].

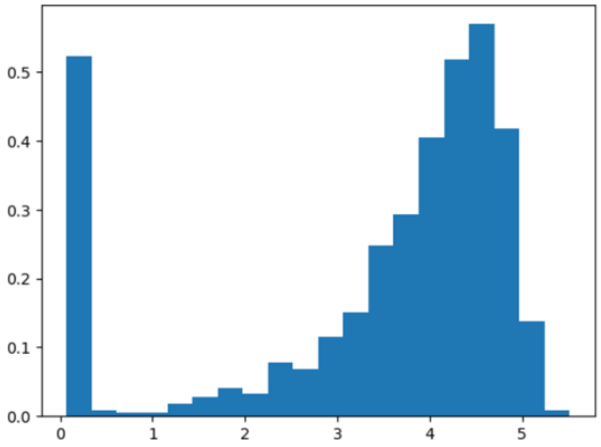


Figure 4: fig: SGD histogram for isotropic Gaussians with 10^7 iterations.

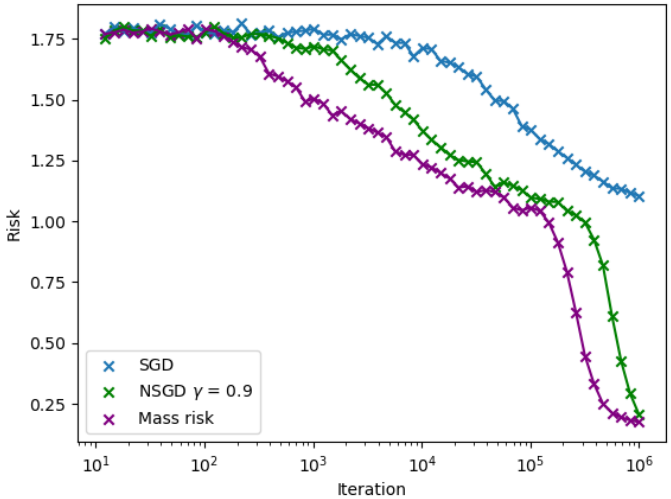


Figure 5: Plot of losses by various variants of SGD for the centred isotropic Gaussian.

I made some observations regarding the loss profiles in figure 5. Plain SGD perhaps coming as no surprise, performs the worst. Nesterov achieves acceleration over plain SGD, though not exponentially, and the decline in loss is rather slow. The MaSS algorithm slightly outperforms the Nesterov accelerated SGD, in line with theoretical predictions made in [7], where the stochastic gradient seems to not be saturated.

In the meantime, upon suggestion of my supervisor, I studied the possible non-uniqueness of stationary states for the mean field PDE that is derived in the paper by Mei et al. and its

connection to the fact that SGD does not always converge to a near global optimum. There they introduce a non-monotone activation function

$$\sigma_*(\mathbf{x};\theta) = \sigma(\langle w, x\rangle), \tag{37}$$

where $\sigma(t) = -2.5$ for $t \leqslant 0, \sigma(t) = 7.5$ for $t \geqslant 1.5$, and $\sigma(t)$ linearly interpolates from $(0, -2.5)$ to $(0.5, -4)$, and from $(0.5, -4)$ to $(1.5, 7.5)$, see figure 6.

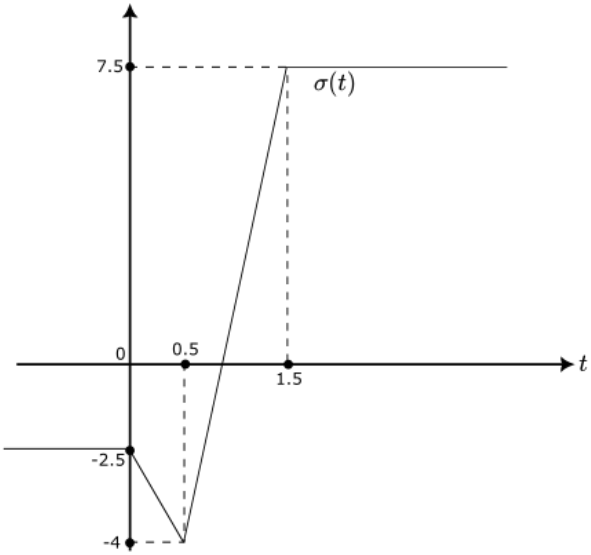


Figure 6: Non-monotone activation function σ.

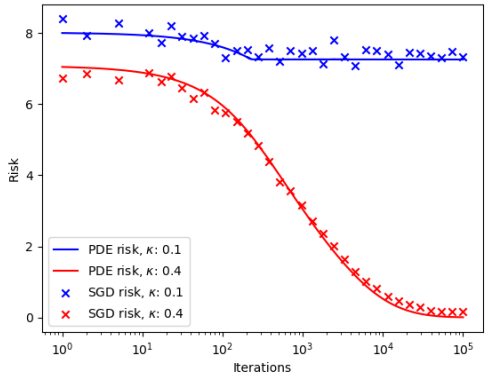
Depending on the initialization,

$$(w_i^0)_{i\leqslant N} \sim_{iid} N(0, \kappa^2/d\,\mathrm{Id}_d) \tag{38}$$

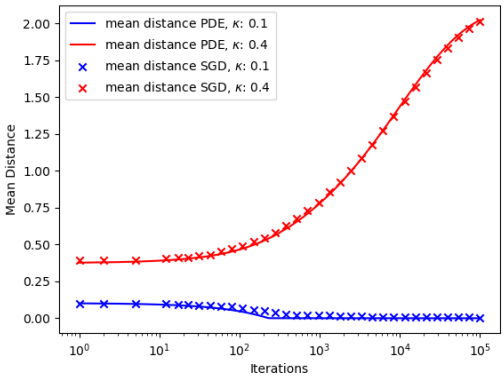
with $\kappa = 0.4, 0.1$, the SGD converges to two different limits, one with a small risk, and the second with high risk (respectively). I reproduced this phenomenon in with a close match to the data presented in the [8], see figure 7.

I implemented a single layer neural network with a non-monotone activcation function (see figure 6) and trained in on jointly Bernoulli and isotropic Gaussian data constructed in the 2018 paper by Mei et al. entitled 'A mean field view of the landscape of two-layer neural networks' using stochastic gradient descent. I studied limiting properties of the evolution of the network parameters, treated as interacting particles and made comparisons in monitoring losses incurred by the discretised SGD algorithm and the distributional dynamics introduced in the paper. I also studied the loss profiles incurred by running various algorithms from plain SGD, to second order algorithms like stochastic MaSS with gradient restart and an interacting particle system-based approach using the MF Hom SGLD algorithm.

Again, similarly to the isotropic Gaussian case with a monotone activation function, the PDE dynamisc were simulated as in the paper by Nguyen et al., see [8, p. 99] and the SGD seemed to match the distributional dynamics at least qualitatively. It is noteworthy to mention that in plot 7b there is almost perfect agreement, beautifully showcasing the theory and validating the empirical results in [8].



(a) Plot comparing PDE vs SGD risk for the non-monotone activation σ.



(b) Plot comparing the average distance of the weights $\|w\|_2$ in the PDE vs SGD simulations for the non-monotone activation.

Figure 7: Separating two isotropic Gaussians, with a non-monotone activation function σ. Here $N = 80, d = 32, \Delta = 0.5$. Continuous lines are prediction obtained with the Distributional Dynamics simplified to reflect the spherical symmetry of the problem.

The non-monotone activation function in the neural network introduced some non-global minima the specific initialisation was around a non-global minimum, that is taking $\kappa = 0.4$ in equation 38. This was observed in the loss profiles in figure 8, where the plain SGD and MF Homm SGLD-driven trajectories seemed to get stuck, whereas Nesterov SGD and MaSS seem to avoid such this bad minima. Note that the MaSS algorithm does not achieve acceleration over stochastic Nesterov. This could be because of the high dimensionality of the problem and the non-convexity of the landscape, in addition to the batch size being around 10^2, maybe being close to the saturation regime (see [7] for details), though this does not appear to be the case in the other classification loss profiles (see 9b and 5). The MF-HomSGLD algorithm seems to take longer to converge, maybe the hyper-parameters of the algorithm are not optimally tuned, though it has inherently a higher computational complexity. This instance is the only in which it performs poorly, and may just be an artefact of poor parameter selection.

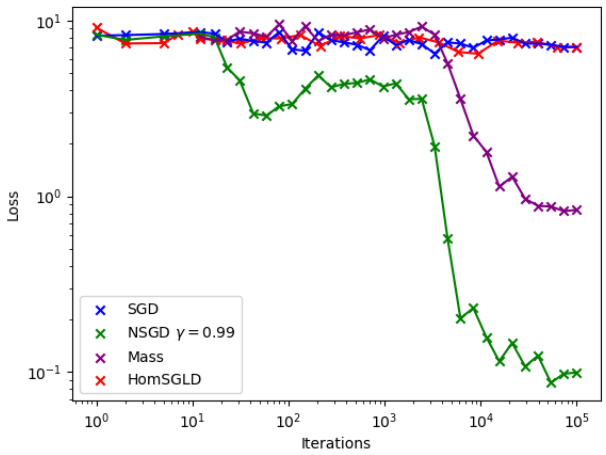


Figure 8: Risk for variants of SGD for Isotropic Gaussian with non-monotone activation.

3.2.2 Anisotropic Gaussians

Having extensively studied the isotropic Gaussian cse, I also decided to implement code for SGD and PDE simulations of risk for the non-isotropic Gaussian case of failure of SGD given two

initializations motivated by the theory developed in [8], following mostly the setup in [8, p. 98–99]. More precisely, the data is now

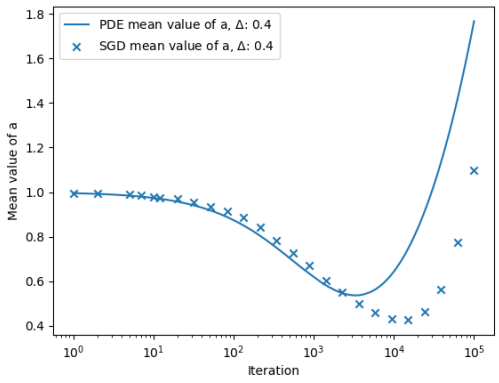
$$\begin{aligned} &\text{with probability } 1/2 : y = +1, \mathbf{x} \sim N(0, \Sigma_+) \\ &\text{with probability } 1/2 : y = -1, \mathbf{x} \sim N(0, \Sigma_-) \end{aligned} \tag{39}$$

where

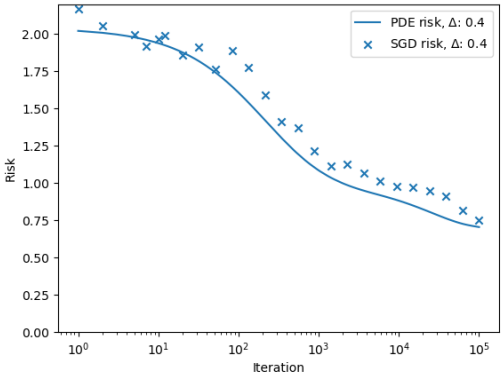
$$\begin{aligned} \Sigma_+ &= \text{Diag}(\underbrace{(1+\Delta)^2, \cdots, (1+\Delta)^2}_{s_0}, \underbrace{1, \cdots, 1}_{d-s_0}) \\ \Sigma_- &= \text{Diag}(\underbrace{(1-\Delta)^2, \cdots, (1-\Delta)^2}_{s_0}, \underbrace{1, \cdots, 1}_{d-s_0}) \end{aligned} \tag{40}$$

and as in the previous case, we choose $\sigma_*(\mathbf{x}; \theta_i) = a_i \sigma_{\text{ReLU}}(\langle \mathbf{x}, w_i \rangle + b_i)$ where $\sigma_{\text{ReLU}}(x) = \max\{x, 0\}$.

The SGD runs fail to match the PDE profiles exactly, though qualitatively, they are similar (figure 9). As for the loss profiles in figure 10, the second order methods MaSS and gradient restarted Nesterov achieve acceleration over plain SGD and Nesterov with $\gamma = 0.9$. The latter two have very similar behaviour. The above suggests Nesterov with optimal γ could beat plain SGD.



(a) Mean value of a for PDE and SGD in the anisotropic Gaussian case.



(b) PDE vs SGD risk for Anisotropic Gaussian

Figure 9: PDE versus SGD risk for Anisotropic Gaussian

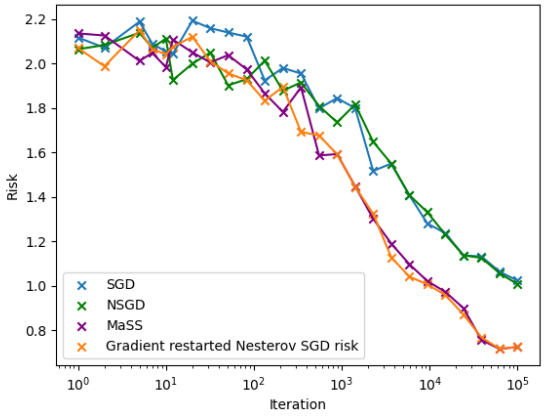


Figure 10: Risk for variants of SGD for Anisotropic Gaussian.

3.2.3 MNIST data single digit classification

I implemented a single-digit classification algorithm on the MNIST dataset and implemented various algorithms. For more details regarding the implementation, see the end of section 2.2. Nesterov acceleration beat plain SGD, but was beaten by MF-HomSGLD and MaSS, the former plateaued the earliest achieving the steepest descent however, achieving a substantially smaller loss in the same amount of time. Again, we are in the low saturation regimes in this simulation too, which is consistent with the theoretical predictions in [7] regarding the acceleration of MaSS over plain SGD and Nesterov. Note finally that MH Homm SGLD is computationally more complex than the rest, but ca be terminated sooned due to the quick plateau, thereby saving computational time.

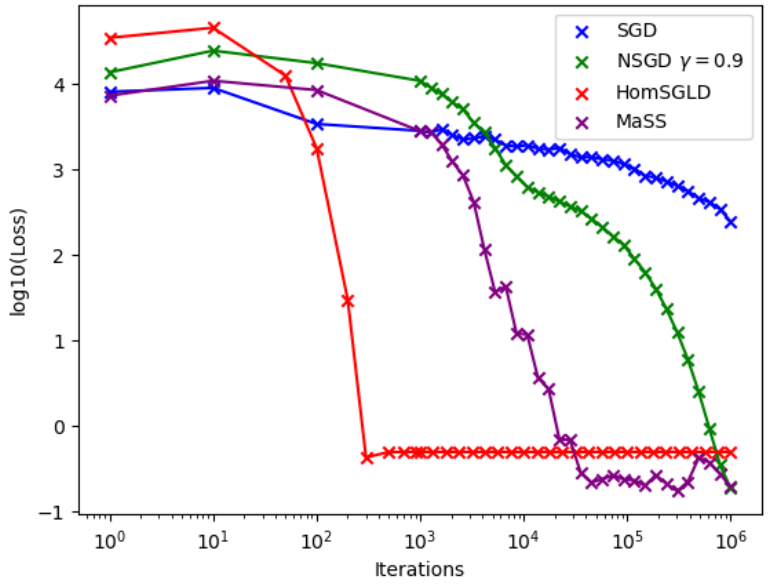


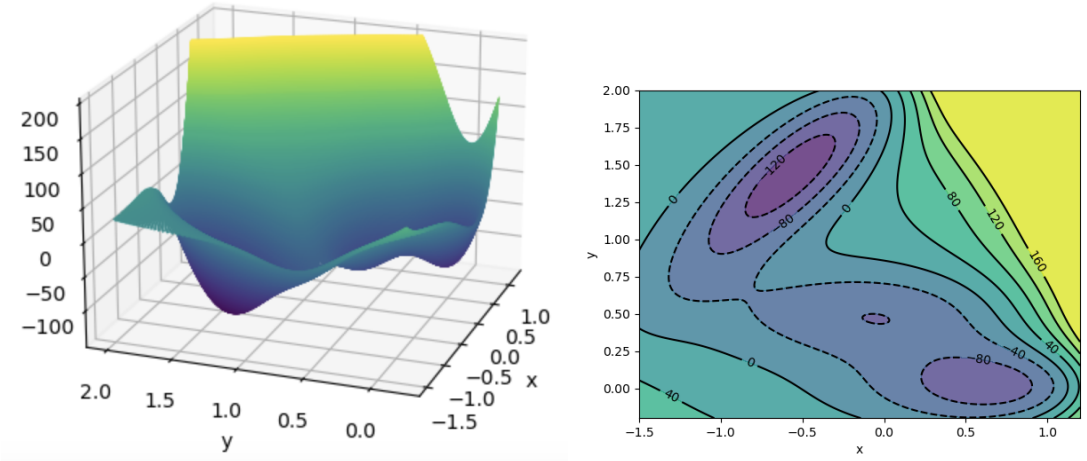
Figure 11: MNIST single-digit classification losses.

3.2.4 Muller Brown Potential

Having developed some theory regarding modifications to the vanilla SGD for non-convex landscapes, we applied it to the canonical example, at least from chemistry [9] of the Muller-Brown potential

$$V(x,y) = \sum_{i=i}^{4} A_i \cdot \exp[a_i \cdot (x-x_0)^2 + b_i \cdot (x-x_0) + c_i \cdot (y-y_0) + d_i \cdot (y-y_0)^2] \tag{41}$$

where $A_i, a_i, b_i, c_i, d_i, i \leqslant 4$ are as in the paper [9]. This potential has multiple local minima in close proximity, making it difficult for SGD to converge to the global minimum, see 12a.



(a) 3d plot of the Muller-Brown potential

(b) Contour plot of Muller-Brown potential.

Figure 12: Contour plot of the Muller-Brown potential

I have implemented deterministic second-order methods, namely, Nesterov and MaSS accelerated gradient descent as well as sstochastic algorithms such as plain SGD and MF Homm SGLD. I have made some observations below.

Nesterov seems to perform slightly better than the MaSS algorithm, with similar characteristics to the SGD-driven trajectories and convergence to local minima appears to be faster than the rest of the algorithms. The drawback is that it still gets stuck in non-global minima. Full Gradient MaSS seems to behave as expected, that is behaves like the full-gradient Nesterov algorithm. It is not really clear whether any acceleration with respect to plain SGD is observed, which might be due to the highly non-convex nature of the potential.SGD Seems to perform slightly better than the MaSS algorithm, with similar characteristics to the Nesterov-driven trajectories. Since it essentially performs descent in the heat-regularised Muller-Brown potential by solving numerically the SDE 19. A drawback is that it still gets stuck in non-global minima. MF-HomSGLD seems to avoid the 'bad' minima in the Muller-Brown potential, as evidenced by the plots in figure 13 with great efficacy and substantially outperforms all other methods. However it suffers from higher algorithmic complexity, dealing with multi-scale SDE system simulations, essentially performing a gradient flow of a regularised potential, where regularisation is done at the level of the Gibbs measure.

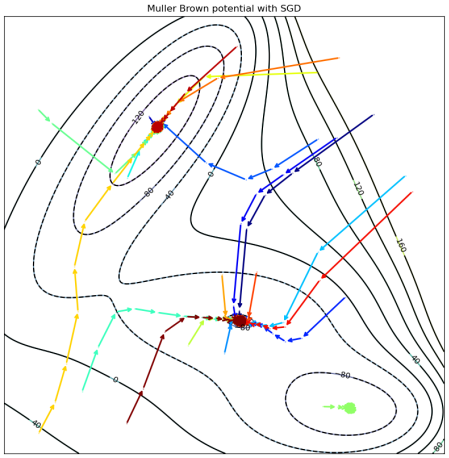
3.2.5 3−d spin model analysis

At the same time that I was investigating the non-uniqueness of stationary states in the distributional dynamics in the [8] paper, I began to read the paper by Van Eijnden et al. [11] and studied the proofs of asymptotic convergence to a gradient flow in the mean field limit and how this is preferable due to the convexification of the loss (as a functional of measures), similar to that observed in [8]. The framework the authors develop is similar to the aforementioned paper and I focused more on applying the algorithms thus explored to the task of accurately representing high dimensional functions, such as the energy function of the continuous 3-spin model on the sphere.

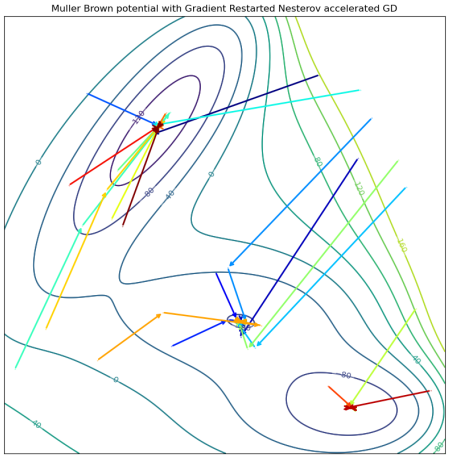
More precisely, by the spherical 3-spin model, we mean the function $f : S^{d-1}(\sqrt{d}) \to \mathbb{R}$, given by

$$f(\mathbf{x}) = \frac{1}{d} \sum_{p,q,r=1}^{d} a_{p,q,r} x_p x_q x_r, \qquad \mathbf{x} \in S^{d-1}(\sqrt{d}) \subset \mathbb{R}^d \tag{42}$$

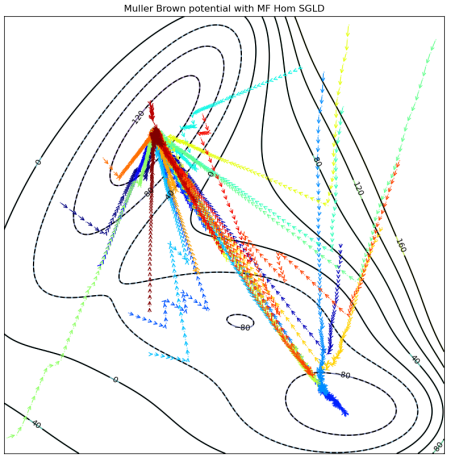
where the coefficients $\{a_{p,q,r}\}_{p,q,r=1}^{d}$ are independent Gaussian random variables with mean zero and variance one. The function (42) is known to have a number of critical points that grows exponentially with the dimensionality d [11]. We train networks to learn f with a particular (random) realization of $a_{p,q,r}$ and study the accuracy of that representation.



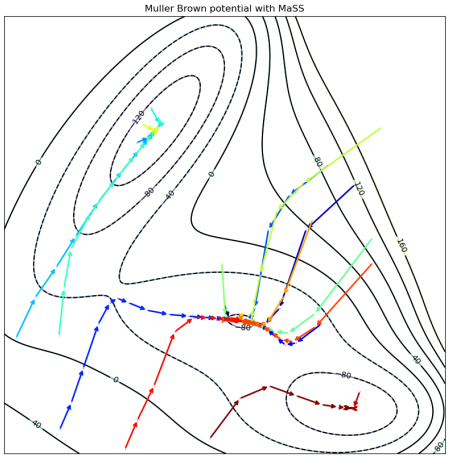
(a) Plain SGD



(b) Nesterov accelerated GD

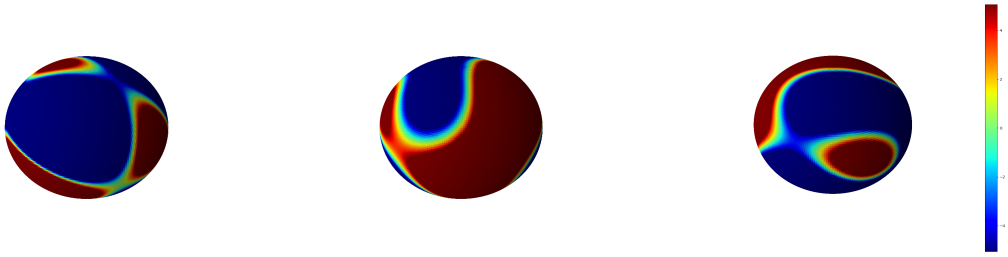


(c) MF Hom SGLD



(d) Full-gradient MaSS

Figure 13: Muller Brown potential trajectories.



(a) d=3 (b) d=4 (c) d=5

Figure 14: Spherical 3− spin function plots.

We first consider the case when $D = S^{d-1}(\sqrt{d})$ and following [11], I used the Gaussian kernel

$$\varphi(\mathbf{x}, \mathbf{z}) = e^{-\frac{1}{2}\alpha|\mathbf{x}-\mathbf{z}|^2} \tag{43}$$

for some fixed $\alpha > 0$. In this case, the parameters are elements of the domain of the function (here the d-dimensional sphere). Note that, since $|\mathbf{x}| = |\mathbf{z}| = \sqrt{d}$, up to an irrelevant constant that can be absorbed in the weights c, we can also write (43) as

$$\varphi(\mathbf{x}, \mathbf{z}) = e^{-\alpha \mathbf{x}\cdot\mathbf{z}} \tag{44}$$

This setting allow us to simplify the problem. Using

$$f^{(n)}(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} c_i \varphi(\mathbf{x}, \mathbf{z}_i) = \frac{1}{n}\sum_{i=1}^{n} c_i e^{-\alpha \mathbf{x}\cdot\mathbf{z}_i}, \tag{45}$$

To test the accuracy of the representation, we used the following Monte Carlo estimate of the loss function

$$\mathcal{L}_P[f_t^{(n)}] = \frac{1}{2P}\sum_{p=1}^{P} \left| f(\mathbf{x}_p) - f_t^{(n)}(\mathbf{x}_p) \right|^2 . \tag{46}$$

which is in close analogy to the risk 2 This empirical loss function was computed with a batch of 10^6 points $\mathbf{x}_p$ uniformly distributed on the sphere.

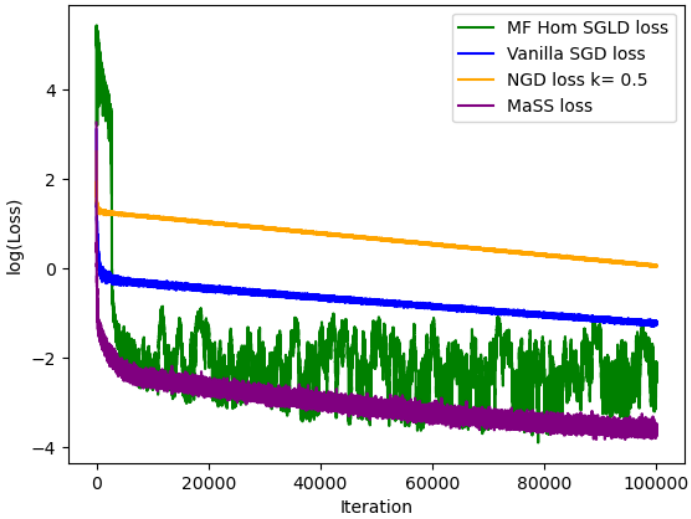
I tested the representation (45) in $d = 3, 4, 5$ using $n = 256$, and setting $\alpha = 5/d = 1$. The training was done by running performing stochastic gradient descent on the loss of the form 2, which was numerically implemented by running a Monte Carlo simulation over 10^2 points chosen uniformly on the sphere $S^{d-1}(\sqrt{d})$, that is performing gradient descent at each step on a loss of the form 46 with time step $\Delta t = 10^{-6}$ for 10^5 steps. The plots showing a contour plot of the original function f as well as any representation $f^{(n)}$ are done so through a slice of the sphere defined as

$$\mathbf{x}(\theta) = \begin{cases} \sqrt{d}\,(\sin(\theta)\cos(\phi), \sin(\theta)\sin(\phi), \cos(\theta))\,, & d = 3 \\ \sqrt{d}\,(\sin(\theta)\cos(\phi), \sin(\theta)\sin(\phi), \cos(\theta), 0)\,, & d = 4 \\ \sqrt{d}\,(\sin(\theta)\cos(\phi), \sin(\theta)\sin(\phi), \cos(\theta), 0, 0)\,, & d = 5. \end{cases} \tag{47}$$

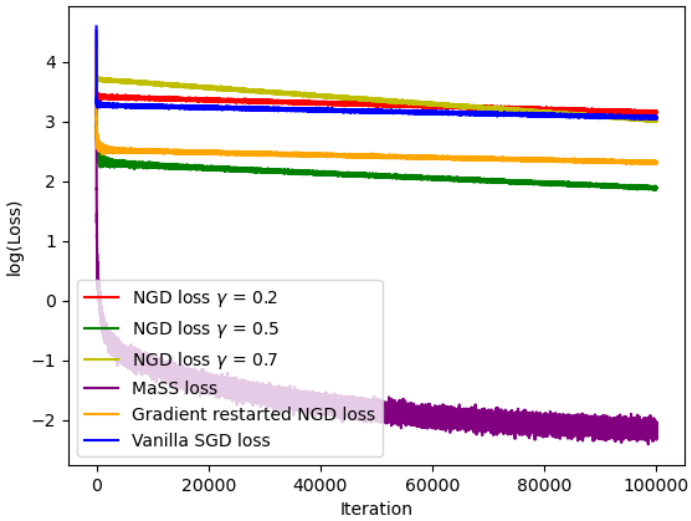
with $\theta \in [0, \pi]$ and $\phi \in [0, 2\pi)$. The level sets of both functions are generally in good agreement, see the figure 16. Also shown on this figure is the projection on the slice of the position of the particles on the sphere. In this result, the parameters c_i take values that are initially uniformly distributed by about $-40d^2 = -10^3$ and $40d^2 = 10^3$. Observe however, that increasing the dimensionality of the problem worsens the approximations. Below are some observations I made.

Mass outperforms all the other algorithms in smaller dimensions $d = 3, 4$. This is in line with the fact that the stochastic gradient is not yet fully saturated, which places the algorithm in the regime where MaSS achieves acceleration over plain SGD and stochastic Nesterov. However, as the

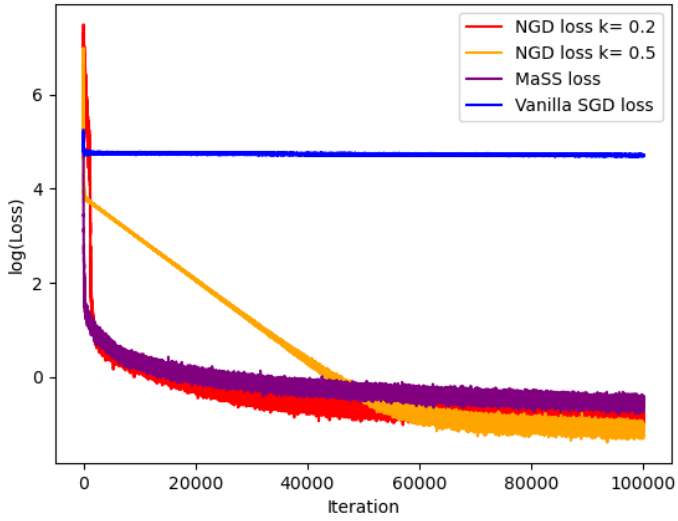
dimensionality of the problem increases, the algorithms seem to converge in terms of performance, except plain SGD, with substantially worse performance. Nesterov has mixed behaviour in these simulations. The plots in figure 15 suggest that optimal selection for γ leads to better performance than plain SGD, but in dimensions $d = 3, 4$ the MaSS algorithm has better performance except for the dimension $d = 5$, where second order methods seems to converges relatively early and outperform plain SGD. MF-Hom SGLD matches the performance of plain MaSS, achieving the second best loss profile for $d = 3$. However it suffers from higher algorithmic complexity, and seems to be volatile, with fluctuations of size two or so orders of magnitudes, though it displays a downward trend. Plain SGD consistently performs the worst as expected, still achieving exponential convergence though like the rest of the algorithms.



(a) $d = 3$

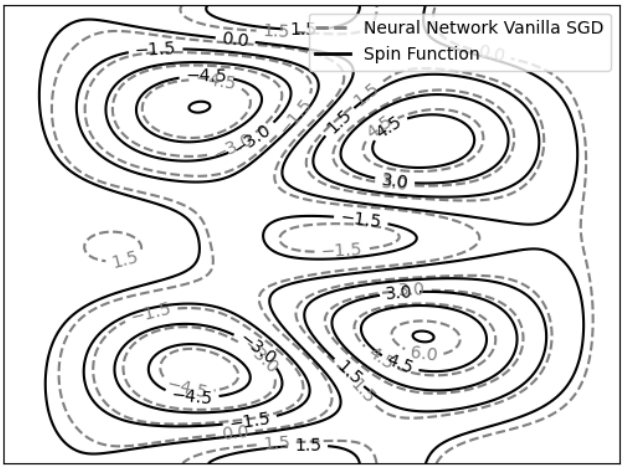


(b) $d = 4$

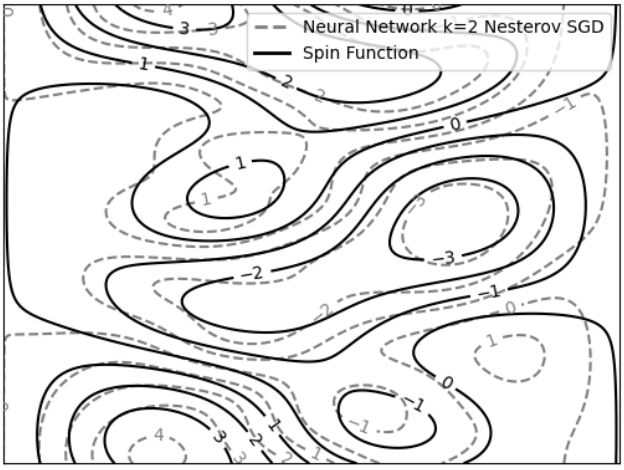


(c) $d = 5$

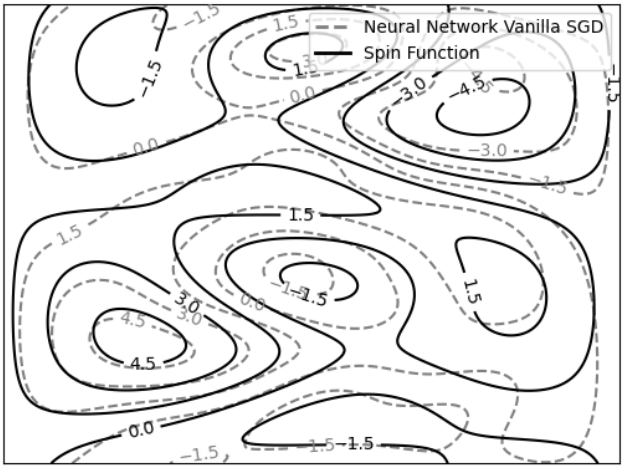
Figure 15: Plots of the logarithm of the loses of variants of SGD for the 3−sphere spin function.



(a) $d = 3$(Hom MF SGLD)



(b) $d = 4$(MaSS)



(c) $d = 5$(Mass)

Figure 16: Muller Brown potential comparison to neural network (using algorithm incurring the smallest final) output contour plots.

References

[1] P. Brémaud. *Probability Theory and Stochastic Processes*. Universitext. Springer International Publishing, 2020.

[2] Pratik Chaudhari, Adam M. Oberman, S. Osher, Stefano Soatto, and Guillaume Carlier. Deep relaxation: partial differential equations for optimizing deep neural networks. *Research in the Mathematical Sciences*, 5, 2017.

[3] Yifan Chen, Daniel Zhengyu Huang, Jiaoyang Huang, Sebastian Reich, and Andrew M. Stuart. Gradient flows for sampling: Mean-field models, gaussian approximations and affine invariance, 2023.

[4] Weinan E, Di Liu, and Eric Vanden-Eijnden. Analysis of multiscale methods for stochastic differential equations. *Communications on Pure and Applied Mathematics*, 58(11):1544–1585, November 2005.

[5] S.N. Ethier and T.G. Kurtz. *Markov Processes: Characterization and Convergence*. Wiley Series in Probability and Statistics. Wiley, 2009.

[6] Nikolas Kantas, Panos Parpas, and Grigorios A. Pavliotis. The sharp, the flat and the shallow: Can weakly interacting agents learn to escape bad minima?, 2019.

[7] Chaoyue Liu and Mikhail Belkin. Accelerating sgd with momentum for over-parameterized learning, 2019.

[8] Song Mei, Andrea Montanari, and Phan-Minh Nguyen. A mean field view of the landscape of two-layer neural networks. *Proceedings of the National Academy of Sciences*, 115(33), July 2018.

[9] Klaus Müller and Leo D Brown. Location of saddle points and minimum energy paths by a constrained simplex optimization procedure. *Theoretica chimica acta*, 53:75–93, 1979.

[10] Brendan O'Donoghue and Emmanuel Candes. Adaptive restart for accelerated gradient schemes, 2012.

[11] Grant Rotskoff and Eric VandenEijnden. Trainability and accuracy of artificial neural networks: An interacting particle system approach. *Communications on Pure and Applied Mathematics*, 75(9):18891935, July 2022.

[12] Justin Sirignano and Konstantinos Spiliopoulos. Mean field analysis of neural networks: A central limit theorem, 2019.

[13] Justin Sirignano and Konstantinos Spiliopoulos. Mean field analysis of neural networks: A law of large numbers, 2019.

[14] Pantelis Tassopoulos. Imperial Summer Research 2023 Code Repository, August 2023.