UNIVERSITAT DE BARCELONA

Efficient and convergent natural gradient based optimization algorithms for machine learning

by

Borja Sánchez López,

supervised by

Dr. Jesús Cerquides Bueno

and tutored by

Dr. Maite López Sánchez

Submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in the

Departament de Matemàtiques i Informàtica

Universitat de Barcelona

2022

A mi madre, a mi hermana

Acknowledgments

Gracias Jesús por todas las oportunidades que me has brindado sin pestañear, incluyendo este proyecto en el que sinceramente me embarqué únicamente para poder seguir investigando contigo. De ti, he aprendido en todos los aspectos, tanto profesionales como personales.

Gracias Jordi y Andrés, por poder discutir ideas de la tesis con vosotros y nutrirme de vuestro conocimiento e intuición.

Gracias a mi madre Pilar, a Aldara, a Kate y a Jordi por vuestro apoyo y paciencia. He podido contar con vosotros en momentos duros, y habéis inclinado la balanza a nuestro favor. Más allá de ser indispensables para cualquier logro que cosechemos, me alegro de teneros conmigo. Mama y Aldara, os debo quien soy, miles de risas y la confianza de quien es querido. Kate, mis momentos son mejores contigo cerca, *Ya tebe kohayu*. Jordi, me gustaría seguir abordando proyectos demasiado grandes, que alimentan mi ilusión y me reconfortan por el tiempo bien invertido. Os quiero.

Abstract

Many times Machine Learning (ML) is casted as an optimization problem. This is the case when an objective function assesses the success of an agent in a certain task and hence, learning is accomplished by optimizing that function. Furthermore, gradient descent is an optimization algorithm that has proven to be a powerful tool, becoming the cornerstone to solving most ML challenges. Among its strengths, there are the low computational complexity and the convergence guarantee property to the optimum of the function, after certain regularities on the function. Nevertheless, large dimension scenarios show sudden drops in convergence rates which inhibit further improvements in an acceptable amount of time. For this reason, the field has contemplated the natural gradient to tackle this issue.

The natural gradient is defined on a Riemannian manifold $(\mathcal{M}, g)$. A Riemannian manifold is a manifold $\mathcal{M}$ equipped with a metric g. The natural gradient vector of a function f at a point p in $(\mathcal{M}, g)$ is a vector in the tangent space at p that points to the direction in which f locally increases its value faster taking into account the metric attached to the manifold. It turns out that the manifold of probability distributions of the same family, usually considered in ML, has a natural metric associated, namely the Fisher Information Metric (FIM). While natural gradient based algorithms show a better convergence speed in some limited examples, they often fail in providing good estimates or they even diverge. Moreover, they demand more calculations than the ones performed by gradient descent algorithms, increasing the computational complexity order.

This thesis explores the natural gradient descent algorithm for the function optimization task. Our research aims at designing a natural gradient based algorithm to solve a function optimization problem, whose computational complexity is comparable to those gradient based

and such that it benefits from higher rates of convergence compared to standard gradient based methods.

To reach our objectives, the hypothesis formulated in this thesis is that *the convergence property guarantee stabilizes natural gradient algorithms and it gives access to fast rates of convergence*. Furthermore, the natural gradient can be computed fast for particular manifolds named Dually Flat Manifold (DFM), and hence, fast natural gradient optimization methods become available.

The beginning of our research is mainly focused on the convergence property for natural gradient methods. We develop some strategies to define natural gradient methods whose convergence can be proven. The main assumptions require $(\mathcal{M}, g)$ to be a Riemannian manifold and f to be a differentiable function on $\mathcal{M}$. Moreover, it turns out that the Multinomial Logistic Regression (MLR) problem, a widely considered ML problem, can be adapted and solved by taking a DFM as the model. Hence, this problem is our most promising target in which the objective of the thesis can be completely accomplished.

Table of Contents

List of Figures

List of Algorithms

1 Introduction

In recent years our world is evolving at a dramatic and unstoppable speed in many aspects due to the impact that computers have brought on civilization. We work with the increasing presence of computers and automated processes, we have at our disposal a world of information on the internet and we even communicate differently, utilizing social networks. The number of tasks that computers can perform for us increases. And, of course, we also solve problems differently.

The branch of research known as Artificial Intelligence (AI) has emerged from this technological revolution. Surprisingly, AI is a science that lacks an accepted definition. As explained in (Stuart Russell, 1995), history has produced four main conceptions about what AI aims to understand and reproduce: systems thinking like a human, systems acting like a human, systems thinking rationally, and systems acting rationally. The reference states that a system or an agent is just something that perceives and acts. AI in this thesis is related to systems that act rationally, that is, "acting so as to achieve one's goals, given one's beliefs".

Learning is an attribute of our intelligence, and computers are capable of recreating it artificially. This thesis is centered on Machine Learning (ML) branch, "a subfield of AI concerned with programs that learn from experience" (Stuart Russell, 1995). In particular, we are interested in incremental learning, in which the agent updates its old knowledge acquired from experience as new information arrives. For example, if after some tries, we end up hitting the inner bull's eye constantly, we learned the task of throwing darts satisfactorily. The experience is processed and learned following the steps of an algorithm.

Algorithms can be implemented and automated by a computer. This means that many solutions are nowadays learned and found by systems. Different representations or models or the different suitable algorithms are some of the common topics treated in ML. Great results of such research allow surprising advances in practical problems such as in the image classification task in (Li et al., 2012), video recommendation from (Covington et al., 2016) or numerous examples in health and life sciences for analyzing nominal qualitative response variables appearing in (Daniels and Gatsonis, 1997; Bull et al., 2007; Biesheuvel et al., 2008; Leppink, 2020).

1.1 Function optimization

In a problem-solving situation, this thesis notates by $\mathcal{M}$ the set of all candidates, with an S the solution set which is the subset of $\mathcal{M}$ of candidates solving the problem, and by f a function going from $\mathcal{M}$ to $\mathbb{R}$, often called expected error function or loss function, that judges the performance of solution candidates towards solving the problem.

Moreover, some assumptions about such problem-solving elements are made. The first one is about set $\mathcal{M}$. This set is assumed to behave locally as an open set of $\mathbb{R}^k$ for some positive integer k. This means that moving through set $\mathcal{M}$ can be done as if it was done through $\mathbb{R}^k$. When this happens, it is said that $\mathcal{M}$ is a smooth manifold. This assumption can be imposed in a huge amount of cases, including most ML challenges.

The second assumption is imposed to function f. It is assumed to be a differentiable $\mathbb{R}$ valued function on the manifold $\mathcal{M}$. Furthermore, we assume that the lower the value of f at p, the better candidate p is. Therefore, learning how to properly execute a task translates into minimizing f. Hence set S in such a scenario is identified by lower values in f. This assumption is widely considered in ML. The faster the function f is minimized, the sooner S is reached, and hence the more successful the learning process is. From here, research in ML tries to figure out clever and clever ways to minimize such functions f. Just to clarify, this work poses the task of learning in ML as a function optimization problem.

1.2 Background

Learning in AI is in general accomplished by function optimization algorithms. Algorithms in this thesis are stochastic processes that determine, after previous candidates and possibly some random phenomena, which candidate should be considered next. An iteration is the process elapsed between two consecutive candidates, and it is crucial to establish what computations need to be completed in every iteration so the algorithm adequately optimizes a function f. If an agent is learning through an algorithm, then function f is minimized by the successive candidates or estimations of the solution given by the algorithm, until good enough estimates are found.

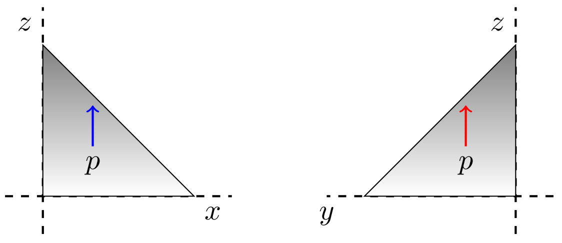
We assume that the function to minimize f is differentiable on $\mathcal{M}$. Therefore the gradient becomes available. The gradient is the cornerstone of nowadays optimization in ML. The reason is intuitive: the gradient points towards the steepest ascent. Therefore, the opposite of the gradient reveals the direction where the function locally decreases its value faster. This idea is first presented by Cauchy.

The most used and standard algorithm is the Gradient Descent (GD) first appearing in (Cauchy, 1847) (see Algorithm 1). It is simple and cheap in terms of computational complexity but it is often enough to achieve proper solutions in ML. Its strength consists of its low computational complexity, allowing many more iterations for the training process in comparison to other algorithms. However, harder problems are tackled every day, where effective learning is demanded. By effective we mean that the information acquired in every iteration from the candidate and its value in f needs to be applied to the learning process as much as possible. It is at this point that GD shows limitations. When the problem is more complex, GD often drops its convergence speed to the solution set demanding infeasible many iterations and an insane amount of time to effectively learn the task and provide a solution. Moreover, it highly depends on hyper-parameters which can be difficult to tune and it is vulnerable to the plateau phenomenon (Fukumizu and Amari, 2000; Dauphin et al., 2014) and ill-conditioning (Gill et al., 2019; Saarinen et al., 1993). The literature proposes many modifications and enhancements to GD seeking effectiveness of the training process. Chapter 2 explores some of these algorithms in more detail. Most modifications derive from GD but our interest lies in the natural gradient technique.

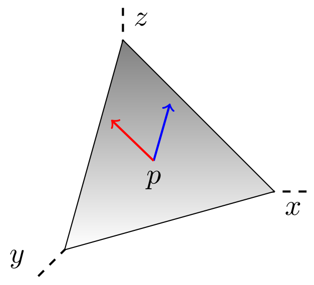
To understand the natural gradient it is necessary to meet Riemannian manifolds. To see the definitions related to this mathematical object see our revisiting in section 2.3 or see (do Carmo, 2013). Intuitively, a Riemannian manifold consists of equipping a manifold $\mathcal{M}$ with a metric tensor g. The Riemannian manifold $(\mathcal{M}, g)$ then allows to correctly capture the curvature of

$\mathcal{M}$. For example, a sphere locally resembles $\mathbb{R}^2$ however local metrics differ from those in $\mathbb{R}^2$. Just notice that angles of a triangle drawn on a sphere do not sum up $\frac{\pi}{2}$, for instance. This in particular exposes that the gradient of a function expressed in a given parametrization may be dependent on the parametrization itself in a Riemannian manifold. Indeed, to be convinced of this fact, consider the next simple example. The simplex S_2 is a 2-dimension manifold defined by points in $\mathbb{R}^3$ whose coordinates are positive and sum up to 1, or formally $S_2 = \{(x, y, z) \in \mathbb{R}^3 \mid x, y, z \geq 0, x+y+z = 1\}$. The simplex can be seen in its ambient space in figure 1.1b as a triangle in a 3-dimensional space. Since the simplex is a 2-dimension manifold (it is a surface), this means that it can be represented by a system of coordinates of dimension 2. For example, it is possible to project the simplex along the y axis to obtain a parametrization of S_2. Or a projection along the x axis would do it too. These 2 parametrizations appear in Figure 1.1a.

Define now the function $f_z(x, y, z) = z$ over the simplex. Maximizing this function is equivalent to climbing up the simplex to the peak, where the z coordinate is highest and the value of the function reaches its maximum value equal to 1. In Figure 1.1, points of the manifold are colored, identifying better candidates with darker points. Observe in the same figure the gradient of f_z at some point p in the two different parametrizations.



(a) Gradients of f_z at p in different parametrizations



(b) Gradients in S_2

Figure 1.1: Two parametrizations of S_2 and the gradients at p associated to each parametrization.

Both vectors are obtained by just writing the function with respect to the corresponding parametrization and then computing the partial derivatives to form the gradient. In both cases, this yields the vector $(0, 1)$ in the corresponding bases. In figure 1.1b the reader can see that both gradients are different. It is relevant to observe that none of these two gradients is pointing to the truly steepest ascent. Intuitively, to climb the simplex as fast as possible, the appropriate direction is the one pointing directly to the peak, since the z coordinate is increased faster. But each parametrization in figure 1.1a lacks information and the resulting gradient is a vertical vector. These misleading vectors that do not really point to the steepest ascent occur even with the flat manifold S_2. Now recall that most important optimization algorithms rely on the gradient. This means that most function optimization algorithms nowadays do not exploit the truly ascend direction and their performance strongly depends on the parametrization settled (see for example (Zaidi et al., 2014)).

The natural gradient at $p \in \mathcal{M}$ of a function f defined in a Riemannian manifold $(\mathcal{M}, g)$ is a vector that points towards the steepest ascent at p taking into account the local metric provided by g. This is proved in (Amari, 1998), and we also provide a different proof in Appendix A.1. Great theoretical advantages come along with the natural gradient. This object is uniquely and well defined, in the sense that it is not parametrization dependent. Moreover, it points to the truly most ascent direction according to the metric information of the Riemannian manifold.

In ML, often an appropriate metric to equip a manifold $\mathcal{M}$ with is the Fisher Information Metric (FIM) or Fisher-Rao metric first defined in (Rao, 1945) and worked in detail for example in (Amari, 2016) and (Nielsen, 2018). FIM is available for manifolds where every point in $\mathcal{M}$ can be related to a family of probability distributions. For instance, in the darts game, different ways of throwing darts define different probability distributions over the dartboard. Such probability distributions indicate regions where the dart is most likely to land. Learning the darts game consists of acquiring a correct intuition of which probability distribution is associated with every way of dart throwing. For such manifolds, where every element in $\mathcal{M}$ determines a probability distribution family, FIM is a standard metric obeying the most important metric intuitions of probability distributions as proven in (Čencov, 1982) such as being the only invariant metric under canonical sufficient statistics. This kind of Riemannian manifold, where points within represent probability distributions, is often called a statistical manifold (Amari et al., 1987; Murray and Rice, 1993; Nielsen, 2018).

From a more theoretical point of view, the advantage of applying the opposite of the natural gradient to minimize a function in a Riemannian manifold is the independence of the parametrization. At this point, we have provided no further reasons to use the natural gradient instead of the regular gradient. However, if the function to minimize is related to the metric of the manifold, then things change. It is proven in (Amari, 1998) that the Natural Gradient Descent (NGD) algorithm, provided it converges, is Fisher efficient when optimizing the conditional log-likelihood. This means that its convergence speed is as good as it can be. This result is promising to fix the low convergence speed of gradient variants mentioned before.

Also, both from a theoretical and a practical standpoint, natural gradient optimization methods carry a weakness, they demand a high order computational cost. Natural gradient based algorithms scale badly, often discarding them as good enough optimization methods to be applied in real (large-scale) problems.

Even if theory supports natural gradient variants, in practice it may show the worst results possible: low convergence speed can be sometimes observed, its computational greed rejects any option to obtain useful feedback from the algorithm or it even diverges and therefore never finds the solution, as it is shown in a simple example in (Thomas, 2014). The instability of such algorithms erases the good theoretical properties of convergence speed. Therefore, if explained shortly, in practice it turns out that natural gradient based algorithms need more computations but they often perform way worse than simpler methods. This is the starting point for this thesis.

1.3 Research questions

The general question we address is

Is it possible to take advantage of the local geometry of $\mathcal{M}$ to effectively guide optimization?

We hypothesize that once convergence is brought to natural gradient algorithms, then high efficiency is at hand. Moreover, we think that some statistical manifolds allow the definition of fast natural gradient algorithms. They are known as Dually Flat Manifold (DFM) in (Amari, 2016) and (Nielsen, 2018) and they are often encountered in ML, for example whenever $\mathcal{M}$ is an exponential family. Such manifolds are the perfect target to build convergent and efficient natural gradient based optimization algorithms.

We have found it necessary to break the main question into 3 concrete and approachable ones during our research. The first question appeared together with our hypothesis. If convergence is really needed for natural gradient based algorithms to achieve efficiency, it is desirable to prove the convergence of a natural gradient based algorithm to then assess its performance. This means we faced the following challenge:

$\boxed{\text{1. Can we generate convergent algorithms close to the natural gradient?}}$

We answered such a question in the positive. Nevertheless, we encountered the astonishing high computational complexity of natural gradient algorithms. This cost comes from a matrix inverse demanded in every iteration (or linear system solving). This fact makes such algorithms impractical for larger and more complex problems. Our main question can be answered only if the computational costs of a natural gradient based optimization method are drastically reduced, ideally to linear in the dimension of the manifold. At this point, we put our effort into finding a response to:

$\boxed{\text{2. Can we make them computationally efficient as well?}}$

Of course, the first question was not just left behind. Our work designing a natural gradient algorithm with reduced computational complexity aims to fulfill convergence property as well. We were able to define a fast natural gradient algorithm in a Dually Flat Manifold (DFM) that we named Dual Stochastic Natural Gradient Descent (DSNGD). However, the convergence property was not as easy to prove as in our previous work. Since a deeper study was needed to prove the convergence of our low computational complexity algorithm, we directly asked:

$\boxed{\text{3. What are the key factors determining the convergence of this type of algorithm?}}$

We made some research about the convergence of stochastic processes to tackle this question. Here, our strategy consists of making an abstraction of such convergence results and identifying the generalized properties that lead algorithms to theoretically satisfy convergence.

1.4 Contributions

This section relates the questions asked with our best answers and contributions. Before any question is faced, the first objective was to provide a stabilized natural gradient based optimization method and study its behavior. The algorithm is named Manifold Optimized Descent (MOD). It reduces the iterations where the metric is updated. This algorithm can be seen as a mixture of gradient descent variants and natural gradient ones. This study is made for my master's thesis in (Sánchez-López, 2018) and this work includes a summary and conclusions reached from that research. In the experiments run, we observed that MOD stabilizes and converges to the solution, unlike NGD which often diverges. Moreover, MOD surpasses GD providing better estimates of the optimum. However, the convergence property of MOD is not proven and its computational complexity is still high, even if matrix inverse computations are not performed at every iteration.

The next contribution can be found in (Sánchez-López and Cerquides, 2019). It answers research question 1. In there, optimization method Convergent Stochastic Natural Gradient Descent (CSNGD) is presented. CSNGD is defined to imitate Stochastic Natural Gradient Descent (SNGD) but ensuring the convergence. This algorithm has proven convergence and it is shown to support our hypothesis: it shows great convergence speed in every scenario where SNGD does not, or even diverges. Code of experiments can be found in (Sánchez-López and Cerquides, 2022a).

Research question 2 is addressed in article (Sánchez-López and Cerquides, 2020). We define a linear complexity order algorithm in a DFM often encountered in ML. The name of the algorithm is Dual Stochastic Natural Gradient Descent (DSNGD). We even manage to prove its convergence after our generalization of a result in (Sunehag et al., 2009). Experiments show great performance compared to standard SGD, where greater rates of convergence of DSNGD can be observed, supporting our hypothesis. The reader can check and reproduce the experiments in (Sánchez-López and Cerquides, 2022b).

Research question 3 finds an answer in (Sánchez-López and Cerquides, 2021). We created a new ground theory focused to prove the convergence of stochastic processes. This is useful to define a wider set of convergent algorithms, including natural gradient based.

As a summary, the contributions are listed below:

- MOD algorithm and a preliminary study on natural gradient based optimization methods in the master's thesis: (Sánchez-López, 2018).
- Towards answering research question 1, the natural gradient based optimization method CSNGD, with proof of convergence, shows great behavior and supports our hypothesis in the paper (Sánchez-López and Cerquides, 2019) and toy problem experiments (Sánchez-López and Cerquides, 2022a).
- DSNGD facing research question 2 in (Sánchez-López and Cerquides, 2020) with experiments in python in (Sánchez-López and Cerquides, 2022b). The article proves the convergence and the linear complexity of the algorithm in the discrete case.
- Our answer to research question 3 and the convergence of stochastic processes in (Sánchez-López and Cerquides, 2021).

1.5 Thesis structure

Apart from this chapter, this work dedicates one chapter to every contribution plus one additional chapter at the beginning containing the related work and another one at the end with conclusions and future work. Chapter 2 includes all necessary contents to understand the contributions. It starts with the definition of the stochastic process, the core mathematical object of study for the thesis. Most relevant gradient descent algorithms are described and commented on. A brief introduction to Riemannian manifolds is then given and DFM are presented. Then, some examples of natural gradient based algorithms are shown. The chapter finishes defining stochastic optimization and stating some useful convergence theorems of stochastic processes existing in the literature.

Chapter 3 contains preliminary research about natural gradient variants and their behavior. It summarizes the master's thesis in (Sánchez-López, 2018), and it studies the strengths of natural gradient based algorithms and their symptomatic performance issues. Algorithm MOD helps understand the weaknesses of natural gradient methods. MOD possesses a positive integer hyperparameter *eras* which determines the frequency of metric updates. The bigger the value of *eras* the closer to a gradient based algorithm becomes MOD. If *eras* has a low value then it imitates NGD closely, becoming exactly NGD when $eras = 1$. Hence MOD can be seen as an adjustable trade-off between GD variants and NGD. This allows us to judge our hypothesis and it supports it since MOD shows stability for big enough *eras*, yet having the convergence speed benefits of natural gradient based algorithms.

The reader can find our first natural gradient based algorithm with proven convergence in Chapter 4 named CSNGD. To prove it, some useful convergence theorems in the literature are used, which can be found at the end of Chapter 2. To reduce its high computational complexity, DFM are considered. In such spaces, matrix inverse computations are saved reducing the complexity order of the algorithm. However, the matrix-vector product still increases the complexity order up to quadratic. The chapter completely nourishes from (Sánchez-López and Cerquides, 2019).

DFM are worked in more detail in Chapter 5 to create a natural gradient based algorithm to predict a class variable $\mathcal{Y}$ given some feature variables $\mathcal{X}$ called DSNGD. The section proves that the complexity order of DSNGD is linear when $\mathcal{X}$ are discrete variables. As explained in the article (Sánchez-López and Cerquides, 2020), the convergence of the algorithm can not be deduced from convergence theorems found in the literature. Hence a generalization is stated and proven, which then ensures the convergence of DSNGD for the discrete case.

Chapter 6 is dedicated to the convergence of stochastic processes, based on (Sánchez-López and Cerquides, 2021). Research made towards proving DSNGD convergence allowed to generate a ground theory to prove the punctual convergence of a stochastic process, considering the *resemblance* of an algorithm to the half-space of a conservative vector field, in terms of its expected direction set. All terms are defined and illustrated in the chapter.

The thesis concludes with Chapter 7 that contains the conclusions extracted from the work done and some challenges left for future work.

2 Preliminary

This chapter summarises the main concepts needed to understand our contributions. Recall that we have posed the learning problem as an optimization problem with the objective function f. Because of the complexity of the function to optimize f, assume that finding the minimum must be accomplished by an iterative method. An iterative method or iterative algorithm is a stochastic process that provides a sequence $p_0, p_1, p_2, p_3...$ of refined candidates to optimize a given function f. If the algorithm is successful, then all candidates after a specific iteration belong to S. To define an algorithm it is enough to establish the maps deciding which candidate to consider next, given previous ones and their function values.

We start revisiting stochastic processes in Section 2.1. Afterward, we introduce some of the standard and most exploited gradient based algorithms nowadays in ML in Section 2.2. An overview of Riemannian manifolds is later given in Section 2.3, with a special interest in statistical manifolds carrying the Fisher Information Metric (FIM). The natural gradient, which is used heavily in Chapters 3,4 and 5, is presented in Section 2.3.2. Next, we give the definitions of a connection and a dually flat manifold which, together with a key property of the natural gradient in such spaces, are essential for building Chapter 5. Thereafter, we give the basic structure of natural gradient based algorithms with some relevant examples in Section 2.4. Then, Section 2.5 introduces stochastic optimization, pointing out that every previous method defined has its online version, illustrated with some examples. Section 2.6 briefly introduces the Multinomial Logistic Regression (MLR) problem, which is the problem faced in Chapter 5. Finally Section 2.7 states convergence theorems which are the basis of later Chapter 6.

2.1 A brief introduction to the stochastic process

The basic mathematical object of the thesis is the stochastic process. For us, an algorithm or optimization method is nothing else than a stochastic process, often subjected to random phenomena. For this reason, we provide some essential definitions and nomenclature related to stochastic processes and their convergence. We are not giving an extensive nor detailed introduction to the stochastic process. Instead, we briefly introduce the concepts that are relevant to the objectives of this thesis. For more information about the topic, the reader can visit (Ross, 1996; Bass, 2011; Billingsley, 1986).

Section 2.1.1 provides definitions regarding stochastic processes which are needed for the thesis. Right after the stochastic process is defined, we provide the concepts of almost sure convergence, filtration, and conditional expectation in Section 2.1.2, which are fundamental for studying the convergence of a stochastic process. Afterward, Section 2.1.3 introduces the director process and the learning rate, which together define the update direction of consecutive terms of a stochastic process.

2.1.1 Stochastic process definition

The goal of this section is to give the definition of the stochastic process, which is the main mathematical object of the thesis. However, we provide some basic concepts needed to understand the definition of a stochastic process.

Definition 2.1.1. *Let Ω be a nonempty space. A set $\mathcal{F}$ of subsets of Ω is a σ-algebra if following conditions hold:*

i) $\Omega \in \mathcal{F}$

ii) if $A \in \mathcal{F}$ then $A' = \Omega \setminus A \in \mathcal{F}$

iii) If $A_1, A_2, ...$ is a countable sequence where $A_i \in \mathcal{F}$ for all $i \in \mathbb{N}$, then $\cup_i A_i \in \mathcal{F}$

Definition 2.1.2. *A measurable space is a tuple $(\Omega, \mathcal{F})$ where Ω is a non empty space and $\mathcal{F}$ is a σ-algebra on Ω.*

Definition 2.1.3. *Let $(\Omega, \mathcal{F})$ and (S, Σ) be two measurable spaces. A $\mathcal{F}/\Sigma$-measurable function $f : \Omega \to S$ is a function such that*

$$(\forall A \in \Sigma) \quad f^{-1}(A) \in \mathcal{F}\ , \tag{2.1}$$

where $f^{-1}(A) = \{x \in \Omega \mid f(x) \in A\}$.

As abuse of notation, and whenever there is no confusion, an $\mathcal{F}/\Sigma$-measurable function is noted as just $\mathcal{F}$-measurable function. A measurable space $(\Omega, \mathcal{F})$ can be extended to a probability space $(\Omega, \mathcal{F}, P)$ where P is a probability measure on $\mathcal{F}$ (see for example (Billingsley, 1986)).

Definition 2.1.4. *Let $(\Omega, \mathcal{F}, P)$ be a probability space and (S, Σ) be a measurable space. A random variable X is a $\mathcal{F}/\Sigma$-measurable function where $X : \Omega \to S$.*

In this thesis, random variables are defined from a probability space to the measurable space (S, Σ) where $S = \mathbb{R}^k$ (or an open set of $\mathbb{R}^k$) and Σ is the corresponding Borel σ-algebra (Royden and Fitzpatrick, 1988). Now we are ready to meet the definition of a stochastic process.

Definition 2.1.5. *Let $(\Omega, \mathcal{F}, P)$ be a probability space and (S, Σ) be a measurable space. A discrete stochastic process is a sequence $Z = \{Z_t\}_{t \in \mathbb{N}}$ of random variables indexed by $\mathbb{N}$ such that $Z_t : \Omega \to S$.*

Stochastic processes usually describe random phenomena sequences. In ML, processes are used to optimize a function. They are intended to approach the minimum of some function f with estimation Z_t at time t as t moves forward. For the function optimization task, S corresponds to the parameter space and in the case of stochastic optimization, Ω is known as the sample space. Example 2.1.1 helps to identify these concepts in a toy example.

Example 2.1.1. Assume a coin $C = \{H, T\}$ with unknown probabilities is tossed providing observations $\omega = (\omega_0, \omega_1, \omega_2, ...)$ where $\omega_i \in C$. That is, $\omega \in \Omega$ is an infinite sequence of observations, each of them valued to either H or T. The coin is ruled by an unknown probability distribution $\overline{P}$. The objective is to learn this probability distribution given the observations. For this situation, the parameter space is $S = (0, 1) \subset \mathbb{R}$, and the sample space Ω is the set of all possible infinite sequences of observations. One can define the following stochastic process;

$$\begin{aligned} Z_0 &= \frac{1}{2} \in S \\ Z_{t+1}(\omega) &= \frac{(t+1)Z_t(\omega) + \mathbb{1}_{\{H\}}(\omega_t)}{t+2} , \end{aligned} \tag{2.2}$$

where

$$\mathbb{1}_A(x) = \begin{cases} 1 \text{ if } x \in A \\ 0 \text{ otherwise} , \end{cases} \tag{2.3}$$

is the indicator function on subset A. Hence $Z_t(\omega)$ reflects the proportion of H in the sequence ω up to time t. It is clear that as t goes to infinity, the stochastic process captures $\overline{P}(H)$ with higher precision.

2.1.2 Almost sure convergence, filtration and conditional expectation

We are interested in studying the almost sure convergence of Z to a point $\overline{\eta} \in \mathbb{R}^k$. We provide the formal definition in Definition 2.1.6.

Definition 2.1.6. *A stochastic process Z on the probability space $(\Omega, \mathcal{F}, P)$ almost surely (a.s) converges to a point $\overline{\eta} \in \mathbb{R}^k$ if*

$$P\left[\omega \in \Omega : \lim_{t\to\infty} Z_t(\omega) = \overline{\eta}\right] = 1 . \tag{2.4}$$

As an abuse of notation, throughout the thesis, the convergence of a stochastic process actually refers to the almost sure convergence of a stochastic process to a point, unless otherwise specified.

The convergence proofs of stochastic processes in this thesis are based mainly on the convergence theorem appearing in (Robbins and Siegmund, 1971). We recall this theorem in Section 2.7. To understand this result, concepts such as filtration and conditional expectation need to be introduced. The reader can check (Billingsley, 1986; Bass, 2011) to expand the concepts given.

Definition 2.1.7. *Let $(\Omega, \mathcal{F}, P)$ be a probability space. A filtration is a collection of σ-algebras $\{\mathcal{F}_t\}_{t\in\mathbb{N}}$ such that $\mathcal{F}_t \subset \mathcal{F}$ for each t and $\mathcal{F}_i \subset \mathcal{F}_j$ if $i \leq j$.*

Definition 2.1.8. *Let $(\Omega, \mathcal{F}, P)$ be a probability space. A stochastic process $Z = \{Z_t\}_{t\in\mathbb{N}}$ is adapted to a filtration $\{\mathcal{F}_t\}_{t\in\mathbb{N}}$ if Z_t is $\mathcal{F}_t$-measurable for all $t \in \mathbb{N}$.*

Definition 2.1.9. *Let A be a set of subsets of a nonempty space Ω. The σ-algebra generated by A denoted by $\sigma(A)$ is the set of subsets of Ω such that:*

i) $A \subset \sigma(A)$

ii) $\sigma(A)$ *is a* σ*-algebra*

iii) If $A \subset \mathcal{F}$ *where* $\mathcal{F}$ *is a* σ*-algebra, then* $\sigma(A) \subset \mathcal{F}$*.*

Definition 2.1.9 implies that $\sigma(A)$ is the smallest σ-algebra containing A and it can be obtained by the intersection of all σ-algebras containing A (see (Billingsley, 1986)). Moreover, the concept of σ-algebra generated by a class A allows generating a filtration given a stochastic process Z defined in a probability space $(\Omega, \mathcal{F}, P)$ to a measurable space (S, Σ). Define

$$\mathcal{F}_t = \sigma(Z_i^{-1}(A) \mid i \leq t, A \in \Sigma)\ , \tag{2.5}$$

where $Z_i^{-1}(A) = \{x \in \Omega \mid Z_i(x) \in A\}$. Then $\{\mathcal{F}_t\}_{t\in\mathbb{N}}$, or simply $\mathcal{F}_t$ as an abuse of notation, is the filtration generated by Z also known as the natural filtration associated to Z. Proving that $\{\mathcal{F}_t\}_{t\in\mathbb{N}}$ is actually a filtration is a straightforward task. By definition, it is also clear that Z is then adapted to $\mathcal{F}_t$. When there is no confusion, notation $\mathcal{F}_t$ in this thesis will refer to the filtration generated by the stochastic process Z.

Intuitively, every $\mathcal{F}_t$ of a filtration is a σ-algebra that classifies the elements of Ω. For example, if Ω is the set of colors, $\mathcal{F}_t$ can gather warm and cold colors into separate and complementary sets. The fact that a random variable Z_t is $\mathcal{F}_t$-measurable implies that Z_t sends all warm colors to the same value and all cold colors also to the same value. Z_t is then not providing any additional information about elements of Ω beyond the classification of $\mathcal{F}_t$. Recall that the sequence $\mathcal{F}_t$ of a filtration is increasing, in the sense that $\mathcal{F}_t \subset \mathcal{F}_{t+1}$ for all t. Therefore, a filtration characterizes space Ω with sequentially higher levels of information or classification. Saying that Z is adapted to $\mathcal{F}_t$ where $\mathcal{F}_t$ is the natural filtration of Z implies that the sequentially increasing information provided by the filtration is the incremental information discriminated from the sample space by the stochastic process.

Definition 2.1.10. *Let Z be an integrable random variable on a probability space $(\Omega, \mathcal{F}, P)$ and let $\mathcal{G}$ be a σ-algebra such that $\mathcal{G} \subset \mathcal{F}$. The conditional expectation of Z given $\mathcal{G}$ is a random variable noted as $\mathbb{E}[Z \mid \mathcal{G}]$ such that:*

i) $\mathbb{E}[Z \mid \mathcal{G}]$ *is $\mathcal{G}$-measurable and integrable*

ii) $\int_G \mathbb{E}[Z \mid \mathcal{G}]\, dP = \int_G Z dP \qquad G \in \mathcal{G}$

Denote $\mathbb{E}_t = \mathbb{E}[\cdot \mid \mathcal{F}_t]$ the conditional expectation given a σ-algebra $\mathcal{F}_t$ (Bass, 2011). Recall that if Z is a random variable in $(\Omega, \mathcal{F}, P)$ then $\mathbb{E}_t[Z]$ is in turn a $\mathcal{F}_t$-measurable random variable.

2.1.3 The director process

The difference between two random variables of a stochastic process is a random variable known as increment. We say that random variable $I_t^s = Z_{t+s} - Z_t$ with $1 \geq s \in \mathbb{N}$ is an s-increment at time t. For example, the 1-increments of a stochastic process Z are

$$I_t^1 = Z_{t+1} - Z_t \; . \tag{2.6}$$

The 1-increments I_t^1 capture the difference random variable between two consecutive random variables of the process. This is useful since most algorithms are defined in terms of their 1-increments. That is, they define the updated variable Z_{t+1} departing from the actual random variable Z_t as a reference, by

$$Z_{t+1} = Z_t + I_t^1 \; . \tag{2.7}$$

It is common to factor the 1-increments I_t^1 into two elements as

$$I_t^1 = -\gamma_t \cdot X_t \; , \tag{2.8}$$

where γ_t is a positive number called the learning rate and X_t is a random variable mostly characterizing the 1-increments direction. The negative sign is just a convention. We provide a definition of this decomposition of the 1-increments for their relevance to this thesis.

Definition 2.1.11. *Let Z and X be stochastic processes and let $\gamma = \{\gamma_t\}_{t\in\mathbb{N}}$ be a sequence of positive numbers. Then (X, γ) is a decomposition of 1-increments of Z if*

$$Z_{t+1} = Z_t - \gamma_t X_t \tag{2.9}$$

Name X the director process of Z and γ the learning rate, and note it by $Z = (X, \gamma)$.

In this shape, where $Z = (X, \gamma)$, the stochastic process Z is known as a line search algorithm. As Definition 2.1.11 states, the sequence $X = \{X_t\}_{t\in\mathbb{N}}$ is a stochastic process over the same probability space of Z and it is named the director process of Z in this thesis. Different settings of the director process X and the learning rate γ_t provide different line search algorithms. Throughout the thesis, some of the most used and successful algorithms are shown by specifying its director process X. The learning rate γ_t has some generally accepted shapes. The most common ones are the constant learning rate $\gamma_t = c$ or choosing at every iteration a γ_t lowering the function the most possible according to the direction set by the director process when the function f is known (offline optimization). The most suitable learning rate for stochastic optimization (online optimization) is a decreasing sequence $\gamma_t = \frac{a}{1+b\cdot t}$, where the function f can only be approximated after a sequence of observations. Refer to Section 2.5 for this later case.

2.2 Optimization methods on Euclidean space

This section first states some simplifications done in the optimization problem. Such assumptions are usually assumed in ML when an optimization problem is solved by a gradient optimization method. Then Section 2.2.1 and Section 2.2.2 provide two main gradient descent algorithms exploited nowadays, known as GD and Adagrad respectively.

Throughout this thesis, we make the extremely simplifying assumption that f is a differentiable function defined in $\mathbb{R}^k$ with a unique global minimum $\overline{\eta} \in \mathbb{R}^k$. Clearly the function has the minimum value $f(\overline{\eta})$. The goal is to find a good enough point to optimize f.

This section makes the assumption that f is known. This is the case of offline optimization. In such a scenario, no sample is observed ($\Omega = \emptyset$) and we consider Z, and therefore the director process X as well, as just sequences of numbers in $\mathbb{R}^k$. Hence, once $Z_0 = \eta_0 \in \mathbb{R}^k$ is specified, the whole sequence Z_t is uniquely determined independently of any random phenomena. For the online or stochastic optimization, where Ω is not empty and the stochastic process Z depends on an outcome of Ω, refer to Section 2.5.

Since the function is differentiable, points whose gradient vanishes are candidates to minimize the function. However, assume the function is too complex to be optimized analytically. Instead, an iterative method is required. This section introduces standard yet most used optimization algorithms. The reader may want to check (Ruder, 2016) for more algorithms and extended information about them.

Without a global view of the function, it is often not easy to decide if a given estimation of the optimum is a decent solution. Hence a stopping condition must be set for an optimization process, that is, a criterion deciding when the algorithm must stop running iterations and yield its best estimate of the optimum. The stopping condition can be satisfied when some threshold on the value of the function is achieved (if for some reason the optimum value is known or deduced). Or also it can stop the learning process after a specified number of iterations defined beforehand. The most used stopping condition checks whether the activity of the algorithm has stopped from a practical point of view: it compares successive estimations (1-increment values), to compute the distance between updates. If the difference is not significant the stopping condition yields true and the algorithm stops searching for estimates since no significant improvement is being made between successive updates.

2.2.1 Gradient Descent

The assumption of f being differentiable provides a key tool for the optimization task. It is well known that the gradient of a function in $\mathbb{R}^k$ points towards the greatest local ascent, and that the opposite of the gradient points towards the greatest local descent. That is, after some estimation p of the solution, there are better estimations (lowering the value of f) in the opposite direction of the gradient. Therefore, the gradient becomes a useful tool for the task.

Definition 2.2.1. *Let f be a differentiable function. Gradient Descent (GD) is a stochastic* *process $Z = (X, \gamma)$ such that*

$$X_t = \nabla f(Z_t)\ . \tag{2.10}$$

See Definition 2.1.11 for the notation used in the definition of GD. As explained in Section 2.1.3, the learning rate is often fixed to a constant $\gamma_t = c$ for off-line optimization. Choosing a constant c for γ_t may be critical to the success of the task because too small learning rates may dramatically drop the convergence speed and too big learning rates can lead to divergence (Ruder, 2016).

Most used optimization methods nowadays are gradient based, all of them derived from the work of (Cauchy, 1847) known as the Gradient Descent (GD) algorithm. The reader can find it in this work as Algorithm 1.

Algorithm 1: Gradient descent

```
   Result: $Z_t$
1  $Z_0, t = 0$;
2  while stopping condition not satisfied do
3      $X_t = \nabla f(Z_t)$;
4      $Z_{t+1} = Z_t - \gamma_t X_t$;
5      $t = t + 1$;
6  end
```

2.2.2 Adagrad

The name of this algorithm stands for adaptive learning rate gradient descent, and it is first defined in (Duchi et al., 2011). It is gradient based, imitating the director process X of GD. Instead, learning rate γ_t is modified according to past iterations and it is not applying the same learning rate for each parameter. To clearly show this difference, the notation of the learning rate is changed to Γ_t, to point out that the learning rate is a diagonal matrix. Each element of the diagonal contains the learning rate for the associated parameter.

This method stores in the diagonal of Γ_t the inverse of the square root of past squared gradients sum. To define this algorithm, let us introduce some notation: Σ_t denotes the sum of squared gradients up to iteration t and $\nu(x) = \sqrt[-2]{x+\epsilon}$ where $\epsilon \approx 10^{-8}$ is a positive smoothing term that avoids division by zero.

Definition 2.2.2. *Let f be a differentiable function. Adagrad is a stochastic process $Z = (X, \gamma)$ such that*

$$\begin{aligned} X_t &= \nabla f(Z_t)\ , \\ \Gamma_t &= c \cdot \nu(\Sigma_t)\ , \qquad c \in \mathbb{R}^+ . \end{aligned} \tag{2.11}$$

The constant c is usually set to 0.01 (Ruder, 2016). Parameters that have been repeatedly updated have a lower learning rate. It is remarkable that Adagrad does not require tuning the learning parameter, since it automatically adapts its value according to previous iterations. Algorithm 2 shows Adagrad instructions, where $diag(u)$ of a vector u stands for the diagonal matrix with u in the diagonal.

Algorithm 2: Adagrad

```
  Result: $Z_t$
1 $Z_0, t = 0, \Sigma = 0$;
2 while stopping condition not satisfied do
3     $\Gamma_t = c \cdot \nu(\Sigma)$;
4     $X_t = \nabla f(Z_t)$;
5     $Z_{t+1} = Z_t - \Gamma_t \cdot X_t$;
6     $\Sigma = \Sigma + diag(X_t) \cdot diag(X_t)$;
7     $t = t + 1$;
8 end
```

2.3 A brief introduction to Riemannian Manifolds

This section introduces some basic concepts in Riemannian Manifolds needed for the contributions of the thesis. Since this section is not expected to be an introductory lesson to differential geometry, and instead, the purpose is to give an overview of the concepts, some rigorous content is skipped. Basically, nourishment comes from (do Carmo, 2013; Nielsen, 2018; Amari, 2016) and we recommended to check the references for a deeper understanding on the subject.

Let $(\mathcal{M}, \tau)$ be a second countable Hausdorff topological space. $\mathcal{M}$ is a manifold if for every $p \in \mathcal{M}$ it locally resembles to $\mathbb{R}^k$, for some positive integer. Therefore, a manifold is a topological space whose points can be referred to by using coordinate systems of $\mathbb{R}^k$. Such coordinate systems or parametrizations are homeomorphisms that relate the open sets from both $\mathbb{R}^k$ and τ. The formal definition is given as Definition 2.3.1.

Definition 2.3.1. *Let $(\mathcal{M}, \tau)$ be a second countable Hausdorff topological space. $\mathcal{M}$ is a manifold if for every $p \in \mathcal{M}$ there exists an open set $U \in \tau$ and a map $\phi : U \to V$ such that $p \in U$, V is an open set of $\mathbb{R}^k$ and ϕ is a homeomorphism. In such a case the dimension of $\mathcal{M}$ is k.*

2.3.1 Smooth Manifold

Differentiable or smooth manifolds are interesting objects since they can inherit properties and results known of $\mathbb{R}^k$, such as differentiable functions defined on $\mathcal{M}$, the tangent space at a point $p \in \mathcal{M}$, vectors, metrics in the tangent space, vector length and angle between vectors.

The pair (U, ϕ) of definition 2.3.1 is called a chart and the collection of all charts is a parametrization. This level of abstraction doesn't give any structure to $\mathcal{M}$, just a coordinate system in $\mathbb{R}^k$ to refer to points in $\mathcal{M}$. Such points in $\mathbb{R}^k$ are often called parameters and this text notes them with $\eta \in \mathbb{R}^k$. An atlas gives a structure to $\mathcal{M}$ to obtain a differentiable or smooth manifold.

Definition 2.3.2. *An atlas (resp. $\mathcal{C}^r$-atlas) is a set of charts $\{(U_i, \phi_i)\}_{i\in\mathcal{I}}$ such that $\cup_i U_i = \mathcal{M}$ and such that for any two pair (U_i, ϕ_i) and (U_j, ϕ_j) the function $\phi_2 \circ \phi_1^{-1}$ evaluated in the open set $\phi_1(U_1 \cap U_2)$ is a smooth function (resp. $\mathcal{C}^r$-function).*

An atlas is commonly referred to as a parametrization in ML. From here on, and whenever there is no confusion, notation is simplified for a parametrization $\{(U_i, \phi_i)\}_{i\in\mathcal{I}}$ and it is noted as (U, ϕ) or just ϕ.

Definition 2.3.3. *A smooth manifold is a manifold with an atlas assigned. A $\mathcal{C}^r$-differentiable manifold is a manifold with a $\mathcal{C}^r$-atlas assigned.*

The gradient

Observe now that a function f defined in a differentiable manifold $\mathcal{M}$ with the parametrization ϕ can be thought as a function from $\mathbb{R}^k$ to $\mathbb{R}$ as $f \circ \phi^{-1}$. If $f \circ \phi_i^{-1}$ is a differentiable map at $\eta = \phi_i(p) \in \mathbb{R}^k$ for all charts (U_i, ϕ_i) in the atlas where $p \in U_i$, then f is said to be differentiable at $p \in \mathcal{M}$. The function is differentiable in $\mathcal{M}$ if it is so at every point. Many examples and insights appear in the references that help to understand smooth manifolds better.

For a differentiable manifold $\mathcal{M}$ and a differentiable function f defined in $\mathcal{M}$, it is possible to compute the gradient of f with respect to a parametrization ϕ, by simply computing the partial derivatives vector of function $f \circ \phi^{-1}$ as a function from $\mathbb{R}^k$ to $\mathbb{R}$ as definition 2.3.4 reads.

Definition 2.3.4. *Let $\mathcal{M}$ be a differentiable manifold, ϕ a parametrization and f a differentiable function defined in $\mathcal{M}$. The gradient of f at $\eta = \phi(p)$ with respect to a parametrization ϕ denoted as $\nabla f(\eta)$ is*

$$\nabla f(\eta) = \nabla(f \circ \phi^{-1})(\eta), \;\; \eta \in \mathbb{R}^k \;. \tag{2.12}$$

In the trivial case where the manifold is simply $\mathbb{R}^k$ and the parametrization is the identity map, definition 2.3.4 of the gradient coincides with the well-known gradient of differentiable real-valued functions in $\mathbb{R}^k$. It is also the mathematical object employed to define the director process of all gradient based algorithms.

It is relevant to point out that this vector is not well defined since clearly depends on the parametrization of $\mathcal{M}$ taken. See Figure 1.1 for an illustrative example.

2.3.2 Riemannian Manifold

Since differentiable functions can be defined in a smooth manifold, it is possible to define directional derivatives at $p \in \mathcal{M}$, by means of curves passing through p and considering the differential at that point. The set of all directional derivatives at p is a vector space of dimension k, and it is known as the tangent space $\mathcal{T}_pM$ of $\mathcal{M}$ at p. At this point, one is ready to meet the definition of Riemannian Manifold.

Definition 2.3.5. *A Riemannian manifold is a pair $(\mathcal{M}, g_p)$ where $\mathcal{M}$ is a smooth manifold and g_p is a symmetric , bilinear, positive-definite metric tensor in $\mathcal{T}_p\mathcal{M}$ where the metric tensor depends on the point p, and the function $p \mapsto g_p$ is differentiable.*

The tensor of a Riemannian manifold induces an inner product in the tangent space as $< u, v >_{g_p} = g_p(u, v)$ for two vectors $u, v \in \mathcal{T}_p\mathcal{M}$. The inner product defines the length and the angle properties of vectors in a natural way. When there is no confusion, g_p is commonly just noted as g.

Assume a parametrization (U, ϕ) is selected. Then for a $p \in \mathcal{M}$ the tangent space at p is expressed in the basis $\{\partial\phi_i, 1 \leq i \leq n\}$ where $\partial\phi_i = \frac{\partial}{\partial\phi_i}_{|p}$. Then the matrix G_η with $\eta = \phi(p)$ such that $(G_\eta)_{ij} = g_p(\partial\phi_i, \partial\phi_j)$ is a positive-definite matrix that provides metric information at point p with respect to the parametrization ϕ. Note that this matrix depends on the parameterization (on the atlas) chosen. Again, when there is no confusion, G_η is commonly just noted as G. The inner product of two vectors $u, v \in \mathcal{T}_p\mathcal{M}$ is then expressed in matrix form as $< u, v >_{g_p} = u^\intercal \cdot G_\eta \cdot v$, and lengths and angles in $\mathcal{T}_p\mathcal{M}$ can be defined after it in the ordinary way.

The natural gradient

In a Riemannian manifold, it is possible not only to differentiate a function but also to measure vector lengths. So, given a differentiable function f on a Riemannian manifold $(\mathcal{M}, g)$, it is possible to consider all normalized vectors of the tangent space $\mathcal{T}_p\mathcal{M}$ at point p, and ask which normalized directional derivative (vector) applied to f is higher. This is accomplished by the natural gradient.

Natural gradient is written as $\widetilde{\nabla} f(\eta)$ in (Amari, 1998) and it is defined with respect to a parametrization ϕ.

Definition 2.3.6. *Let $(\mathcal{M}, g)$ be a Riemannian manifold, ϕ a parametrization and f a differentiable function defined in $\mathcal{M}$. The natural gradient of f at $\eta = \phi(p)$ with respect to a parametrization ϕ is*

$$\widetilde{\nabla} f(\eta) = G_\eta^{-1} \cdot \nabla f(\eta), \;\; \eta \in \mathbb{R}^k \; . \tag{2.13}$$

A result appearing in (Amari, 1998) proves that the natural gradient does not depend on the parametrization and that it points to the true ascent direction of a differentiable function f. We recall this result as Theorem 2.3.1. Check the reference for the proof, or visit our version of the proof using only basic concepts of Riemannian manifolds in appendix A.1.

Theorem 2.3.1. *The natural gradient is not parametrization dependent. Moreover, let $(\mathcal{M}, g)$ be a Riemannian Manifold, η be a parametrization and f be a differentiable function. Then the natural gradient of f at $\eta = \phi(p)$ points to the steepest ascent direction of f at p.*

Hence, the natural gradient is independent of the parametrization and the opposite direction of the natural gradient points to the truly local descent direction obeying the metrics. Suddenly natural gradient possesses great properties for the function optimization task.

The Fisher information metric

For this work, it is of particular relevance a certain kind of Riemannian manifolds. Specifically, the manifolds whose points represent probability distributions. They are known as statistical manifolds (Nielsen, 2018; Amari et al., 1987; Murray and Rice, 1993). This section defines the Fisher Information Metric (FIM), a positive-definite metric tensor first appearing in the work of (Rao, 1945).

Definition 2.3.7. *Let $\mathcal{M}$ be a statistical manifold and ϕ be a parametrization such that $\eta = \phi(p) \in \mathbb{R}^k$ is the parameter of a family of probability distributions P_η. The Fisher information metric (FIM) or Fisher-Rao metric at η is defined by $\mathbb{R}^{k\times k}$ matrix*

$$G_\eta = \mathbb{E}_{z\sim P_\eta}\left[\nabla_\eta \log P_\eta(z) \cdot \nabla_\eta \log P_\eta(z)^\intercal\right] , \tag{2.14}$$

where $\cdot$ stands for the matrix product.

It is a simple exercise (Lehmann and Casella, 2006; Sun and Nielsen, 2016) to prove that previous definition of FIM is equivalent to

$$G_\eta = -\mathbb{E}_{z\sim P_\eta}\left[\nabla^2_\eta \log P_\eta(z)\right] . \tag{2.15}$$

Riemannian manifold $(\mathcal{M}, g)$ where g is the FIM is in many senses the most convenient for statistical manifolds. For example, a manifold whose points are probability distributions is desirable to equip it with an invariant metric under sufficient statistics, up to re-scaling. It has been proved that FIM is the only metric capable of that, as shown in (Čencov, 1982; Amari and Nagaoka, 2000). It is recommended to visit the references for more information on the topic.

The FIM is the only metric used in this thesis, and g is used to refer to that metric and G for the metric matrix unless otherwise stated.

2.3.3 Conjugate connection Manifold

Two tangent spaces $T_p\mathcal{M}$ and $T_q\mathcal{M}$ for different points $p, q \in \mathcal{M}$ are completely different, and there is no further information about how they are related, even if they resemble more and more as points p and q get closer. In fact, a vector $v \in T_p\mathcal{M}$ does not belong to $T_q\mathcal{M}$. But if the projection of v to the space $T_q\mathcal{M}$ is given for infinitesimally close point q, then a complete recovering of $\mathcal{M}$ is possible. That is a connection of a Riemannian manifold. For the definition, let $X(\mathcal{M})$ denote the space of smooth vector fields.

Definition 2.3.8. *An affine connection ∇, or covariant derivative operator, or connection, defines the directional derivative $\nabla(X, Y) = \nabla_X Y$ of a vector field Y according to a vector field X;*

$$\begin{aligned} \nabla : X(\mathcal{M}) \times X(\mathcal{M}) &\to X(\mathcal{M}) \\ (X, Y) &\mapsto \nabla(X, Y) = \nabla_X Y \ , \end{aligned}$$

satisfying the following properties;

- $\nabla_{fX+gY} Z = f\nabla_X Z + g\nabla_Y Z$
- $\nabla_X (Y + Z) = \nabla_X Y + \nabla_X Z$
- $\nabla_X fY = f\nabla_X Y + X(f)Y$,

for all $X, Y, Z \in X(\mathcal{M})$ and smooth functions f, g.

In particular, given two vectors $u, v \in T_p\mathcal{M}$, a connection answers how vector v is projected in the tangent space situated infinitesimally close in the direction u. So, in fact, given the properties of Definition 2.3.8, it is only necessary to define how basis vectors of tangent space at p vary in the direction of the same basis vectors. This is accomplished by smooth functions $\Gamma^k_{i,j}(p)$ such that

$$\nabla_{\partial_i}\partial_j = \sum_k \Gamma^k_{i,j}\partial_k \ . \tag{2.16}$$

These functions $\Gamma^k_{i,j}(p)$ are called the Christoffel symbols. The reader can expand his knowledge about affine connections with (Kobayashi and Nomizu, 1963; Lee, 2018; Calin and Udrişte, 2014; do Carmo, 2013).

Definition 2.3.9. *Let $(\mathcal{M}, g)$ be a Riemannian Manifold. Two connections ∇ and ∇^* are conjugate connections with respect to the metric g if and only if for every $X, Y, Z \in X(\mathcal{M})$ smooth vector fields*

$$X < Y, Z >_g = < (\nabla_X Y), Z >_g + < Y, (\nabla^*_X Z) >_g \ . \tag{2.17}$$

In such case, the manifold $(\mathcal{M}, g, \nabla, \nabla^*)$ is said to be a Conjugate Connection Manifold (CCM)

Many interesting properties arise for CCM, for example with the parallel transport, but those are skipped in this text for the sake of brevity.

2.3.4 Dually flat Manifold

Definition 2.3.10. *Let $(\mathcal{M}, g, \nabla)$ be a Riemannian manifold with a connection defined on it. The Riemman-Christoffel curvature (RC) is defined as*

$$R(X,Y)Z = \nabla_X(\nabla_Y Z) - \nabla_Y(\nabla_X Z) - \nabla_{[X,Y]}Z, \qquad X, Y, Z \in X(\mathcal{M}) \ , \tag{2.18}$$

where $[X, Y] = XY - YX$ is the Lie bracket of vector fields

A manifold (M, g, ∇) is flat if RC vanishes. If moreover, the manifold is a CCM $(\mathcal{M}, g, \nabla, \nabla^*)$ then RC curvature also vanishes for the conjugate connection ∇^* (see theorem 6.5 in (Amari, 2016)). In such case, (M, g, ∇, ∇^*) is called a Dually Flat Manifold (DFM).

DFM construction from a convex function

There is a key result (theorem 6.7 in (Amari, 2016)) that we want to use. To understand it, this text explains the first two concepts needed which are manifolds derived from Bregman divergences and the Legendre-Fenchel transform. Both concepts briefly explained next are worked in more detail in (Amari, 2016; Nielsen, 2018).

Definition 2.3.11. *Let $F(\eta)$ be a convex smooth function defined in an open convex domain E. The Bregman divergence associated to F is*

$$B_F(\eta, \eta') := F(\eta) - F(\eta') - (\eta - \eta')^\intercal \nabla F(\eta') \ . \tag{2.19}$$

Definition 2.3.12. *Let $F(\eta)$ be a convex smooth function defined in an open convex domain E. The Legendre-Fenchel transform of F is*

$$F^*(\eta^*) := \sup_{\eta \in E} \eta^\intercal \eta^* - F(\eta) \ . \tag{2.20}$$

Every Bregman divergence induces a Riemannian Manifold (E, g) where

$$G_\eta = -\nabla^2_{\eta'} B_F(\eta, \eta') \mid_{\eta'=\eta} = \nabla^2 F(\eta) \ . \tag{2.21}$$

For a convex function $F(\eta)$ a dual parametrization η^* can be defined, by simply doing $\eta^* = \nabla F(\eta)$. Furthermore, it is possible to get back to η parametrization by doing $\eta = \nabla F^*(\eta^*)$ where F^* is the Legendre-Fenchel transform of F. As explained in the literature (Amari, 2016), F^* is also convex. That means F^* also induces a Riemannian Manifold (E^*, g_{η^*}) where

$$\begin{aligned} E^* &= \{\nabla F(\eta) \mid \eta \in E\} \ , \\ G_{\eta^*} &= \nabla^2 F^*(\eta^*) \ . \end{aligned} \tag{2.22}$$

Furthermore, two conjugate connections ∇, ∇^* can be built (section 6.2 in (Amari, 2016)). These connections are proven to be flat, so $(E, g_\eta, \nabla, \nabla^*)$ is a DFM.

The two parametrizations hold the Crouzeix identity;

$$\begin{aligned} Id &= \nabla^2 F(\eta) \nabla^2 F^*(\eta^*) \\ &= G_\eta G_{\eta^*} \ . \end{aligned} \tag{2.23}$$

Theorem 2.3.2 says that every DFM can always be constructed with a convex function and the Bregman divergence associated.

Theorem 2.3.2 (6.7 in (Amari, 2016))**.** *For a DFM, there exists a Legendre pair of convex functions $F(\eta), F^*(\eta^*)$ and a canonical divergence given by the Bregman divergence*

$$D[\eta, \eta'] = F(\eta) + F^*((\eta')^*) - \eta \cdot (\eta')^* \; . \tag{2.24}$$

Natural gradient in DFM

Observe that in a dually flat manifold, one can assure Theorem 2.3.3, which can be deduced after (Raskutti and Mukherjee, 2015).

Theorem 2.3.3. *Let $(\mathcal{M}, g, \nabla, \nabla^*)$ be a DFM. Then there exist η and η^* two dual parameterizations, and moreover*

$$\widetilde{\nabla} f(\eta) = \nabla f(\eta^*), \tag{2.25}$$

where $\eta = \eta(\eta^*)$.

Proof. Since $(\mathcal{M}, g, \nabla, \nabla^*)$ is a dually flat manifold, the previous theorem ensures the existence of dual parameterizations η and η^*, and also of a strictly convex function $F(\eta)$ defined for all $\eta \in E \subset \mathbb{R}^n$ such that $(\mathcal{M}, g) = (E, F)$ and a function $F^*(\eta^*)$ such that

$$\begin{aligned} \eta &= \nabla F^*(\eta^*) \\ G_{\eta^*} &= \nabla^2 F^*(\eta^*) \; . \end{aligned} \tag{2.26}$$

By the chain rule and Crouzeix's identity one writes;

$$\begin{aligned} \nabla f(\eta^*) &= \nabla^2 F^*(\eta^*) \nabla f(\eta) \\ &= G_{\eta^*} \nabla f(\eta) \\ &= (G_\eta)^{-1} \nabla f(\eta) \\ &= \widetilde{\nabla} f(\eta) \; . \end{aligned} \tag{2.27}$$

□

Therefore, above result states that in DFM the natural gradient in η equals the gradient in η^*.

2.3.5 Example: Exponential Family

The exponential family is a parametric family of probability distributions holding some desirable properties, the most important being the existence of sufficient statistics regarding the Pitman–Koopman–Darmois Theorem (Koopman, 1936). This Theorem basically says that only for the exponential family there exists a sufficient statistic whose number of scalar components does not increase with the sample size. This family includes a wide variety of famous sets of distributions, to list some: normal, exponential, gamma, categorical, Poisson, Dirichlet, beta, and more.

Exponential family (Wani, 1968) manifold is a well-known example of DFM. We summarize contents about the exponential family appearing in (Amari, 2016). Check the reference for complete development of the topic.

Let Ω be a set. A Linear Exponential Family (LEF) is a set of probability distributions $\{P_\eta \mid \eta \in \mathbb{R}^k\}$ defined such that;

$$P_\eta(x) = \frac{\exp^{T(x)^\intercal \eta}}{\lambda(\eta)}\ , \tag{2.28}$$

where $T : \Omega \to \mathbb{R}^k$ is a sufficient statistic and $\lambda(\eta) = \int_x \exp T(x)^\intercal \eta < \infty$. If T is minimal (that is, k is the least possible) Equation 2.28 makes LEF a manifold, since $\phi(\eta) = P_\eta$ makes a correspondence between $\mathbb{R}^k$ and LEF. In such a case, η is called the natural parametrization of that LEF. Observe that every point in manifold LEF is a probability distribution. It is typical to enrich such manifolds with the FIM to obtain a Riemannian manifold. This is the only metric considered from now on for LEF. As can be seen in (Amari, 2016), this Riemannian manifold is equivalently constructed from the convex function

$$F(\eta) = \log \lambda(\eta)\ . \tag{2.29}$$

The Bregman divergence associated after $\log \lambda(\eta)$ is known as Kullback-Leibler divergence (KL) and LEF is enriched with two flat conjugate connections, this means that any LEF is actually a DFM. The dual parametrization η^* is then;

$$\begin{aligned}\eta^* = \nabla F(\eta) &= \nabla \log \lambda(\eta) \\ &= \frac{\nabla \lambda(\eta)}{\lambda(\eta)} \\ &= \int_x T(x) \frac{\exp T(x)^\intercal \eta}{\lambda(\eta)} \\ &= \int_x T(x) P_\eta(x) \\ &= \mathbb{E}[T(x)]\ .\end{aligned} \tag{2.30}$$

Reasonably, η^* is called the expectation parametrization.

2.4 Optimization methods over a Riemannian manifold

The problem slightly changes in this section. The differentiable function f is defined over a k-Riemannian manifold $(\mathcal{M}, g)$ instead of being defined in flat $\mathbb{R}^k$. An atlas or parametrization ϕ is assumed on the manifold so f can be seen as a differentiable function that goes from $\mathbb{R}^k$ to $\mathbb{R}$ by means of that parametrization. Hence, gradient based algorithms of Section 2.2 can still be exploited ignoring completely the metric of the space. However, the regular gradient is not well defined in a Riemannian manifold, which means it varies depending on the parametrization (see Figure 1.1 or check (Zaidi et al., 2014)). Moreover, the steepest descent is not considered. The natural gradient is the appropriate theoretical object to use in a Riemannian manifold to optimize a function since it is well defined and it correctly points toward the steepest ascent. Nevertheless, the good theoretical properties of natural gradient do not stop parametrization-dependent gradient based algorithms to be the cornerstone of most ML training processes.

After definition 2.3.6, the computational complexity of the natural gradient may be higher than that of the gradient: the natural gradient demands a matrix inverse and a matrix-vector product per iteration. Having to find the inverse of a matrix at every iteration usually makes algorithms based on natural gradient impractical for large-dimension manifold scenarios. That is, the natural gradient scales badly as the dimension of the manifold grows. But this is not the only problem with this kind of algorithms. They often show bizarre behavior and tend to diverge, as it can be observed in (Thomas, 2014).

Since the goal of this thesis is to find natural gradient based algorithms whose previously mentioned weaknesses are overcame, this section shows some natural gradient based algorithms which have been widely considered in ML.

2.4.1 Natural gradient descent

Just as GD sets the director process to be the gradient of f, the Natural Gradient Descent (NGD) in (Amari, 1998) sets it to be the natural gradient of f. Recall the natural gradient is obtained by multiplying the inverse matrix of G (according to a parametrization) by the regular gradient, as Theorem 2.3.1 states.

Definition 2.4.1. *Let $(\mathcal{M}, g)$ be Riemannian manifold of dimension k and let f be a differentiable function defined in $(\mathcal{M}, g)$. Let ϕ be a parametrization of the manifold. Natural Gradient Descent (NGD) is a stochastic process $Z = (X, \gamma)$ such that*

$$X_t = G_{Z_t}^{-1} \cdot \nabla f(Z_t) \; , \tag{2.31}$$

where G_η stands for the metric matrix at $\eta \in \mathbb{R}^k$ in ϕ parametrization.

See Section 2.1.3 for the notation concerning the director process $Z = (X, \gamma)$. There are no further differences. So, assuming a Riemannian manifold $(\mathcal{M}, g)$, and that a parametrization ϕ has been fixed, the reader can find the instructions in Algorithm 3.

Algorithm 3: Natural gradient descent

```
  Result: $Z_t$
1 $Z_0, t = 0$;
2 while stopping condition not satisfied do
3     $X_t = G_{Z_t}^{-1} \cdot \nabla f(Z_t)$;
4     $Z_{t+1} = Z_t - \gamma_t X_t$;
5     $t = t + 1$;
6 end
```

2.4.2 Mirror descent

This optimization method of (Nemirovskiĭ and Yudin, 1983) uses two dual parametrizations. The basic idea behind it is that the authors observed that the gradient is a mathematical object belonging to the dual space. Hence, the update equation is mixing and adding elements whose ambient spaces are different. This may be difficult to justify theoretically. But if it is assumed an invertible map (mirror map) H that relates both dual spaces, then the following recipe can be derived.

1. Compute the gradient $\nabla f(Z_t)$.
2. Take the actual point Z_t to its dual space as $\Theta_t = H(Z_t)$.
3. Perform the gradient descent step in the dual $\Theta_{t+1} = \Theta_t - \gamma_t \nabla f(Z_t)$.
4. Map back the updated point to the ambient space as $Z_{t+1} = H^{-1}(\Theta_{t+1})$.

Mirror descent can be seen as a gradient based stochastic process actually run in the dual space, where the director process is in fact the gradient in the primal space.

Definition 2.4.2. *Let f be a differentiable function, and H a function relating the ambient and the dual space. Mirror descent is a stochastic process $\Theta = (X, \gamma)$ where*

$$X_t = \nabla f(H^{-1}(\Theta_t)) \ . \tag{2.32}$$

This different point of view but equivalently algorithm is more suitable for the notation given in this thesis. Algorithm 4 describes this version.

Algorithm 4: Mirror descent

```
Result: $\Theta_t$
1 $\Theta_0, t = 0$;
2 while stopping condition not satisfied do
3     $Z_t = H^{-1}(\Theta_t)$;
4     $X_t = \nabla f(Z_t)$;
5     $\Theta_{t+1} = \Theta_t - \gamma_t X_t$;
6     $t = t + 1$;
7 end
```

It is remarkable to see that in a DFM, mirror descent is exactly the natural gradient descent run in the dual space. Article (Raskutti and Mukherjee, 2015) is dedicated to prove this. The result is clear since, in a DFM, the mirror map is exactly the linear map defined by the inverse of the metric matrix. Hence, by Theorem 2.3.3 it is $\nabla f(Z_t) = \widetilde{\nabla} f(\Theta_t)$ yielding the same algorithm description as in Algorithm 3 in the dual parametrization.

2.5 Stochastic optimization

This thesis is more interested in stochastic optimization. In this case, the function to optimize is unknown so the stochastic process is not determined by f, and it depends on an outcome $w \in \Omega \neq \emptyset$. Every method previously defined can be adapted to a such scenario and become a stochastic algorithm. But first, it is needed to clarify exactly what the new directions and conditions are.

Section 2.5.1 defines the optimization problem for the stochastic scenario. We reveal the probability space of stochastic process Z for the stochastic optimization problem in Section 2.5.2. Besides, we state two conditions that are usually assumed on the learning rate for stochastic optimization that are needed for convergence property purposes. Section 2.5.3 presents the maximum likelihood problem, which is a particular stochastic optimization problem largely faced in ML. Finally, Section 2.5.4 and Section 2.5.5 define two stochastic optimization methods named SGD and SNGD. These two algorithms are the stochastic version of Algorithms 1 and 3 respectively. In particular, SGD is the stochastic algorithm solving most stochastic optimization problems nowadays.

2.5.1 The stochastic optimization problem

As we mentioned at the beginning of this section, for stochastic optimization (Robbins and Monro, 1951) we assume f is not known. Nevertheless, assume it is possible to obtain an approximation of f after random phenomena.

Formally, let $(\overline{\Omega}, \overline{\mathcal{F}}, \overline{P})$ be a probability space where $\overline{P}$ is unknown and let $l(\eta, \overline{\omega})$ be a known function, usually called loss function, such that

$$f(\eta) = \mathbb{E}_{\overline{\omega} \sim \overline{P}} \left[l(\eta, \overline{\omega}) \right] . \tag{2.33}$$

The expectation term in Equation 2.33 is known as the expected risk or expected loss function (Vapnik, 1991) in the ML branch. We make another assumption about the loss function. Since this thesis studies gradient descent optimization methods, we need l to be differentiable with respect to η. Furthermore, we assume that the dominated convergence theorem (see for example (Dudley, 2002)) can be applied to ∇l and then

$$\mathbb{E}_{\overline{\omega} \sim \overline{P}} \left[\nabla l(\eta, \overline{\omega}) \right] = \nabla \mathbb{E}_{\overline{\omega} \sim \overline{P}} \left[l(\eta, \overline{\omega}) \right] = \nabla f(\eta). \tag{2.34}$$

It is theoretically possible to recover f after Equation 2.33, but in practice, it is not, due to the limitations faced when computing the expectation over the unknown probability $\overline{P}$. Nevertheless, a sample of observations can be drawn to approximate the function f. It is common to approximate it by means of the empirical risk or loss. Let $\omega = (\omega_0, \omega_1, \omega_2 ...)$ be observations where $\omega_i \sim \overline{P}$, the empirical risk or loss up to observation $m \in \mathbb{N}$ is

$$S_m(\eta) = \frac{1}{m} \sum_{i}^{m} l(\eta, \omega_i) \qquad m > 1 . \tag{2.35}$$

The stochastic optimization problem consists of optimizing the known function S_m as an alternative to optimizing the unknown function f.

2.5.2 Stochastic optimization methods

We need to state the probability space where the stochastic process Z is defined. Consider the product probability space

$$(\Omega = \prod_{t \in \mathbb{N}} \overline{\Omega}, \mathcal{F} = \prod_{t \in \mathbb{N}} \overline{\mathcal{F}}, P = \prod_{t \in \mathbb{N}} \overline{P}) , \tag{2.36}$$

guaranteed to exist according to the Kolmogorov extension theorem (see for example Theorem 2.4.4 and following examples in (Tao, 2011)) over infinite sequences. The stochastic process Z is defined in such product probability space of Equation 2.36. This means that after an infinite sequence of observations $\omega = (\omega_0, \omega_1, \omega_2, ...) \in \Omega$, the stochastic process Z defines the sequence $Z(\omega)$ of points in $\mathbb{R}^k$. In this scenario, the sample of observations $\omega = (\omega_0, \omega_1, ...)$ is drawn from P where every $\omega_i \in \overline{\Omega}$ is drawn independently from $\overline{P}$. By definition, ω_i are independent and identically distributed with distribution $\overline{P}$.

The stochastic process Z is usually written in terms of l because it is the only function available. The challenge of stochastic optimization is defining Z such that $Z_t(\omega)$ optimizes f as t increases, for every $\omega \sim P$. Numerous methods use function l with different approaches to update the stochastic process Z_t as more observations ω_i become available. For this reason, we assume that the random variables Z_t depend only on observations ω_i until time $i < t$, meaning that

$$\begin{aligned} &Z_0 \in \mathbb{R}^k \ , \\ &Z_t(\omega) = Z_t(\omega_0, ..., \omega_{t-1}) \quad t > 0 \ . \end{aligned} \tag{2.37}$$

The learning rate for stochastic optimization

Algorithms suffer another change in order to run it online, concerning the learning rate. The learning rate γ_t can not be constant in general, since the director process X in such algorithms does not tend to 0 as we approach the optimum of f because, commonly, X_t has no global information about function f. This means that X_t is not shortening the step size, no matter the estimation quality of Z_t. This task is commanded to the learning rate. Hence, the learning rate is often chosen freely obeying two conditions, which this work refers to it as the **learning rate constraint**

$$\begin{aligned} &\sum_t \gamma_t^2 < \infty \ , \\ &\sum_t \gamma_t = \infty. \end{aligned} \tag{2.38}$$

These conditions are imposed for convergence property reasons (Bottou, 2012; Sunehag et al., 2009). First summation $\sum_t \gamma_t^2 < \infty$ ensures the limit of γ_t to be 0 and therefore the method is encouraged to reduce more and more its update jumps as iterations go on. And after the second condition $\sum_t \gamma_t = \infty$, the algorithm is able to travel an infinite distance if the director process allows. This makes it possible to reach any point in the space, even if point Z_0 is far away from a solution point. The most used learning rate for stochastic optimization holding the **learning rate constraint** is $\gamma_t = \frac{a}{1+b \cdot t}$ for tuned hyper-parameters a and b.

2.5.3 Maximum likelihood problem

The maximum likelihood problem is an optimization problem consisting of the maximization of the likelihood function (or log-likelihood function). That is, let $(\overline{\Omega}, \overline{\mathcal{F}}, \overline{P})$ be a probability space and let $\omega = (\omega_0, \omega_1, ...) \sim P$ be a sample drawn from the product probability space. Also, let $\mathcal{M}$ be a statistical manifold with parametrization ϕ and parameter $\eta \in \mathbb{R}^k$, that is, $\mathcal{M}$ is a family of probability distributions $\{P_\eta : \eta \in \mathbb{R}^k\}$. Then the likelihood function up to time $m \in \mathbb{N}$ is defined as the probability of the observations assuming that they are generated by the probability distribution represented by the parameter η;

$$\mathcal{L}(\eta) = \prod_i^m P_\eta(\omega_i) \; . \tag{2.39}$$

Since random variables ω_i are independent and identically distributed, the likelihood function is a product of the marginal probabilities. To precisely shape the function as in Equation 2.35, apply the logarithm to obtain the log-likelihood function

$$L\mathcal{L}(\eta) = \log \mathcal{L}(\eta) = \sum_i^m \log P_\eta(\omega_i) \; . \tag{2.40}$$

This is not modifying the solution: the logarithm is a monotone increasing function, and optimizing the log-likelihood function is equivalent to optimizing the likelihood function. Hence, as a stochastic optimization problem, the maximum likelihood problem sets $S_m(\eta) = L\mathcal{L}(\eta)$ with $l(\eta, \overline{\omega}) = \log P_\eta(\overline{\omega})$ where $\overline{\omega} \in \overline{\Omega}$.

Example 2.5.1. Example 2.1.1 in page 13 describes a maximum likelihood problem situation; the manifold is the family of categorical distributions of dimension 1 and the function to optimize f can be seen as the KL divergence to the distribution $\overline{P}$:

$$f(\eta) = KL(\overline{P}, P_\eta) = \sum_{\overline{\omega} \in \{H,T\}} \overline{P}(\overline{\omega}) \log \frac{\overline{P}(\overline{\omega})}{P_\eta(\overline{\omega})} \; . \tag{2.41}$$

This is because maximizing the KL divergence means finding η such that P_η better approximates $\overline{P}$, which is the goal of example 2.1.1. Observe that maximizing this function is equivalent to minimize

$$\sum_{\overline{\omega} \in \{H,T\}} \overline{P}(\overline{\omega}) \log P_\eta(\overline{\omega}) = \mathbb{E}_{\overline{\omega} \sim \overline{P}} \log P_\eta(\overline{\omega}) \; . \tag{2.42}$$

This function closely relates to Equation 2.33, implying that it is an expected risk function with

$$l(\eta, \overline{\omega}) = \log P_\eta(\overline{\omega}) \ , \tag{2.43}$$

as in the maximum likelihood problem. Indeed, after $\omega = (\omega_0, \omega_1, ...) \sim P$ drawn from the product probability space, the empirical risk or loss up to time $m \in \mathbb{N}$ is the log-likelihood function:

$$S_m(\eta) = \sum_{i}^{m} \log P_\eta(\omega_i) = L\mathcal{L}(\eta) \ . \tag{2.44}$$

In Sections 2.2.1 and 2.4.1 we have introduced GD and NGD respectively. In Sections 2.5.4 and 2.5.5 we provide the stochastic version of those algorithms using the notation given in this section.

2.5.4 Stochastic gradient descent

The GD has an online version to optimize a function f which is not known, but approximated by a loss function l and a sequence of observations $\omega = (\omega_0, \omega_1, \omega_2, ...)$, with assumptions of Section 2.5. It requires the assumption that l is differentiable.

Definition 2.5.1. *Let l be a differentiable loss function and $\omega = (\omega_0, \omega_1, \omega_2, ...) \in \Omega$ be a sample drawn from the product probability space. Stocastic Gradient Descent (SGD) is a stochastic process $Z = (X, \gamma)$ such that*

$$X_t(\omega) = \nabla l(Z_t(\omega), \omega_t) \ . \tag{2.45}$$

X and Z are non-trivial (non-constant) stochastic processes. We mean that X_t and Z_t really depend on a sample ω, unlike the off-line optimization algorithms presented in Section 2.2. Observe that after Equation 2.34, the conditional expectation up to time t of X_t is

$$\mathbb{E}_t X_t = \nabla f(Z_t) \ . \tag{2.46}$$

Hence SGD is expected to approximate the gradient of f with X_t, as in GD. Asymptotic equivalence between GD and SGD is studied in (Murata, 1998). A complete description of the steps of SGD is detailed in Algorithm 5.

Algorithm 5: Stochastic gradient descent

```
  Result: $Z_t(\omega)$
1 $Z_0, t = 0, \omega = (\omega_0, \omega_1, ...)$;
2 while stopping condition not satisfied do
3     $X_t(\omega) = \nabla l(Z_t(\omega), \omega_t)$;
4     $Z_{t+1}(\omega) = Z_t(\omega) - \gamma_t X_t(\omega)$;
5     $t = t + 1$;
6 end
```

2.5.5 Stochastic natural gradient descent

Similarly as passing from GD to SGD, the optimization method NGD can derive to a natural gradient algorithm for online optimization.

Definition 2.5.2. *Let $(\mathcal{M}, g)$ be a Riemannian Manifold of dimension k, let l be a differentiable loss function defined in $(\mathcal{M}, g)$ and let ϕ be a parametrization of the manifold. Let $\omega = (\omega_0, \omega_1, \omega_2, ...) \in \Omega$ be a sample drawn from the product probability space. Stochastic Natural Gradient Descent (SNGD) is a stochastic process $Z = (X, \gamma)$ such that*

$$X_t(\omega) = G^{-1}_{Z_t(\omega)} \cdot \nabla l(Z_t(\omega), \omega_t)\ , \tag{2.47}$$

where G_η stands for the metric matrix at $\eta \in \mathbb{R}^k$ in ϕ parametrization.

In general, under same conditions needed for Equation 2.34, the expected value of X_t up to time t is then

$$\mathbb{E}_t X_t = \widetilde{\nabla} f(Z_t)\ . \tag{2.48}$$

The reader can find its instructions in Algorithm 6.

Algorithm 6: Stochastic natural gradient descent

```
Result: $Z_t(\omega)$
1 $Z_0, t = 0, \omega = (\omega_0, \omega_1, ...)$;
2 while stopping condition not satisfied do
3     $X_t(\omega) = G^{-1}_{Z_t(\omega)} \cdot \nabla l(Z_t(\omega), \omega_t)$;
4     $Z_{t+1}(\omega) = Z_t(\omega) - \gamma_t X_t(\omega)$;
5     $t = t + 1$;
6 end
```

2.6 Multinomial logistic regression

This thesis defines its main optimization method in Chapter 5. Such an algorithm is designed to solve a particular ML problem, which is the Multinomial Logistic Regression (MLR), briefly explained in this section.

Classification algorithms predict the value of a discrete variable (class) given some other variables (features). We use $\mathcal{Y}$ for the class variable and $\mathcal{X} \in \Omega$ for the features. We assume a finite set of classes $\mathcal{Y} \in \{1, ..., s\}$. We are interested in computing the unknown conditional probability distributions $P(\mathcal{Y} \mid \mathcal{X})$. This is accomplished by optimizing the expected risk function (Vapnik, 1991).

MLR is a widely used tool for classification. The core assumption (Banerjee, 2007) is that the log-odds ratio of the class posteriors $P(\mathcal{Y} \mid \mathcal{X})$ is an affine function of the features $\mathcal{X}$;

$$\log \frac{P(\mathcal{Y} = k \mid \mathcal{X} = x)}{P(\mathcal{Y} = h \mid \mathcal{X} = x)} = \eta^{\intercal} \cdot T(x) \ , \tag{2.49}$$

where $\eta \in \mathbb{R}^k$ and $T(x) \in \mathbb{R}^k$ is a statistic. Some relevant examples solving real-world tasks are (Li et al., 2012) for the image classification branch, (Covington et al., 2016) for video recommendation tasks, or numerous examples in health and life sciences for analyzing nominal qualitative response variables, to name some (Daniels and Gatsonis, 1997; Bull et al., 2007; Biesheuvel et al., 2008; Leppink, 2020).

The justification of MLR goes beyond practical. In statistical decision theory, it is well known that the choice probability can be derived assuming that (i) the random utilities are independent and identical distributed (i.i.d.) across alternatives and that (ii) their common distribution is a Gumbel function (Ben-Akiva et al., 1985). Recent results (Tadei et al., 2018) show that the Gumbel distribution for the choice variables is not necessary and that any distribution which is asymptotically exponential in its tail is sufficient to obtain the MLR model.

2.7 Convergence theorems

Since our work is lastly focused on the convergence of natural gradient based algorithms, this section recalls some convergence theorems of algorithms. The notation is translated from the original sources into the notation already introduced in the thesis.

Theorem 2.7.1 (Bottou's in (Bottou, 1998))**.** *Let $f : \mathbb{R}^k \to \mathbb{R}$ be a function with a unique minimum $\overline{\eta}$ and $Z_{t+1} = Z_t - \gamma_t X_t$ be a stochastic process. Then Z_t converges to $\overline{\eta}$ almost surely if the following conditions hold;*

Bottou resemblance $(\forall \delta > 0) \inf_{\|Z_t - \overline{\eta}\| > \delta} (Z_t - \overline{\eta})^\intercal \cdot \mathbb{E}_t [X_t] > 0 \quad a.s.$

Bottou algorithm bound $(\exists A, B)(\forall t)\ \mathbb{E}_t \|X_t\|^2 \leq A + B\|Z_t - \overline{\eta}\|^2 \quad a.s.$

Learning rate constraint

The expression $\mathbb{E}_t$ refers to the conditional expectation given the σ-algebra generated by $Z_0, ..., Z_t$ (see Section 2.1.2). Theorem 2.7.1 is a version discussed in (Bottou, 1998) as comments at the end of Section 4.5. We explain briefly the assumptions of Theorem 2.7.1. **Bottou resemblance** forces the dot product of the expectation of the director process X_t and the vector $Z_t - \overline{\eta}$ to be positive, which implies that the expectation of the update from Z_t to Z_{t+1} is made towards the solution. The **Bottou algorithm bound** ensures that the expected norm of X_t is bounded by a linear expression in terms of the distance from Z_t to the solution. As a consequence, the algorithm can not travel an arbitrarily big distance between Z_t and Z_{t+1}. The **learning rate constraint** is already discussed in Section 2.5.2. These conditions together guarantee the convergence of the stochastic process Z to the solution point $\overline{\eta}$.

Recall another convergence result that will be very useful throughout the thesis, appearing in (Sunehag et al., 2009).

Theorem 2.7.2 (Theorem 3.2 in (Sunehag et al., 2009))**.** *Let $f : \mathbb{R}^k \to \mathbb{R}$ be a twice differentiable cost function with a unique minimum $\overline{\eta}$ and let $Z_{t+1} = Z_t - \gamma_t B_t \cdot Y(Z_t)$ be a stochastic process where B_t is symmetric and only depends on information available at time t and Y be a continuous function from $\mathbb{R}^k$ to random vectors. Then Z converges to the $\overline{\eta}$ almost surely if the following conditions hold;*

C.1 $(\forall t)\ \mathbb{E}_t Y(Z_t) = \nabla f(Z_t)$

C.2 $(\exists K)(\forall \eta)\ \|\nabla^2 f(\eta)\| \leq 2K$

C.3 $(\forall \delta > 0)\ \inf\limits_{f(\eta)-f(\overline{\eta})>\delta} \|\nabla f(\eta)\| > 0$

C.4 $(\exists A, B)(\forall t)\ \mathbb{E}_t \|Y(Z_t)\|^2 \leq A + Bf(Z_t) \qquad a.s.$

C.5 $(\exists a, b : 0 < a < b < \infty)(\forall t)\ spec(B_t) \subset [a, b]$

C.6 Learning rate constraint

where $spec(B)$ are the eigenvalues of matrix B.

Theorem 2.7.2 proves convergence of a certain kind of gradient based algorithms. Intuitively, conditions **C.1** and **C.5** restrict the algorithm to be the well-known SGD modified by positive definite and symmetric matrices with strictly positive bounds of the eigenvalues. Moreover, condition **C.4** bounds the algorithm updates linearly by the function they try to optimize, similarly to **Bottou algorithm bound** of Theorem 2.7.1. However, to prove the convergence, the result requires two more conditions **C.2** and **C.3** on the function f. These conditions basically say that f has no saddle points and it has a bounded Hessian matrix.

Convergence Theorems 2.7.1 and 2.7.2 are basically proven after Robbins-Siegmund theorem appearing in (Robbins and Siegmund, 1971). The result is recalled for its relevance to this thesis.

Theorem 2.7.3 (Robbins-Siegmund)**.** *Let $(\Omega, \mathcal{F}, P)$ be a probability space and $\mathcal{F}_1 \subseteq \mathcal{F}_2 \subseteq \cdots$ a sequence of sub-σ-fields of $\mathcal{F}$. Let U_t, β_t, ϵ_t and ζ_t, $t = 1, 2, \ldots$ be non-negative $\mathcal{F}_t$-measurable random variables, such that*

$$\mathbb{E}(U_{t+1} \mid \mathcal{F}_t) \leq (1 + \beta_t)U_t + \epsilon_t - \zeta_t,\ \ t = 1, 2, \ldots \tag{2.50}$$

Then on the set $\{\sum_t \beta_t < \infty, \sum_t \epsilon_t < \infty\}$, U_t converges almost surely to a random variable, and $\sum_t \zeta_t < \infty$ almost surely.

Name the positive variations of a stochastic process Z, the positive difference of consecutive random variables of the process (that is, when $Z_t < Z_{t+1}$). Intuitively, the main idea behind Theorem 2.7.3 is that a positive stochastic process converges almost surely if the infinite sum of positive variations of the process is bounded almost surely (see also (Bottou, 1998)).

3 Manifold optimized descent

The natural gradient is a vector that is not parameter-dependent and it points to the steepest ascent of a function according to a metric established. Moreover, in (Amari, 1998) it is shown that SNGD is Fisher efficient when solving the maximum likelihood problem assuming that it converges. This means that using the natural gradient gives asymptotically the best possible result. Despite the good theoretical properties of natural gradient, the practical behavior, in general, leaves much to be desired. This chapter evaluates the performance of natural gradient based algorithms and exposes its weaknesses towards function optimization which are the high computational complexity and convergence issues.

Finally, the chapter introduces the algorithm MOD first presented as my master's thesis in (Sánchez-López, 2018). The description of MOD given in this thesis is a generalization to a wider set of optimization problems since the reference is restricted exclusively to multinomial logistic regression problems. Nevertheless, the idea is the same: Creating an adjustable trade-off algorithm between gradient and natural gradient based algorithms, thanks to a parameter *eras*.

3.1 Preliminary study on natural gradient descent

This section exposes two disadvantages of standard NGD, usually spread to other natural gradient based algorithms. These are the high computational complexity and divergence property. Such two issues are critical enough to discard this kind of algorithm as a solver for real problems. Recall that the thesis plans to define efficient and convergent natural gradient based algorithms. Hence, the section introduces the biggest challenges this thesis aims to surpass.

3.1.1 Computational complexity symptoms

The computational complexity of the natural gradient is high. According to definition 2.3.6, the natural gradient demands a matrix inverse computation (or alternatively to solve a linear system). That is if k is the dimension of the Riemannian manifold $(\mathcal{M}, g)$ it needs $O(k^3)$ operations with the Gaussian elimination method for example. In the same definition, a matrix-vector product can be observed. That makes $O(k^2)$ operations more. Both non-linear tasks make natural gradient descent scale awfully with the dimension of the manifold.

Precisely, if the dimension of the manifold is k, assume that A_k corresponds to the cost of computing the gradient of f at a given point and that B_k is the cost of inverting a matrix of dimension k. Then, the computational cost per iteration of NGD is

$$C(NGD) = O(k^2) + A_k + B_k \ . \tag{3.1}$$

3.1.2 Divergence symptoms in a toy example

An indicator of the divergence symptoms of natural gradient optimization methods is the identification of the deficiencies of such algorithms for solving real problems. Indeed, we decided to run a simple optimization problem to test SNGD. It leads to worse estimations of the optimum than SGD, even if it performs more operations per iteration. The problem is the die problem, which consists of finding the best categorical distribution fitting the probabilities of the faces of a die, by optimizing the KL divergence.

The die problem

A k dimensional die is a categorical probability distribution $\overline{P}$ over a discrete set $D = \{1, ..., k\}$. The die problem consists of recovering the probability distribution $\overline{P}$ after a sample of observations $\omega_i \sim \overline{P}$. Categorical distributions belong to the exponential family, and hence the natural parametrization is at hand. Check Example 2.3.5 to recall these concepts. Natural parametrization is the one considered from now on when solving the die problem, unless otherwise stated. For natural gradient based algorithms, the metric will be the FIM presented in Section 2.3.2.

Access the code of this reproducible experiment and all experiments of this section in (Sánchez-López and Cerquides, 2022a).

We created 3 groups of random dice, each group representing different scenarios: for the first group only high entropy dice are considered, the second group is filled with medium entropy dice and finally, the third set contains dice with a low entropy probability distribution. Entropy is defined to be

$$h(D) = -\sum_{i \in D} \overline{P}(i) \log \overline{P}(i) \; , \tag{3.2}$$

where the sum is over all faces of the die D and $\overline{P}(i)$ is the probability of i-th face. Entropy codifies the uncertainty of the die. The less uncertainty the lower entropy. Hence the first scenario refers to dice close to a fair die (since a fair die has the highest uncertainty and highest entropy). As entropy decreases, some faces gather more probability in comparison to others, so we can be more certain about the result of throwing the die.

Each scenario has 100 dice of the same kind, and every die has 200 faces (that is, $k = 199$). Every die is thrown 5000 times. We run classic algorithm SGD and standard natural gradient algorithm SNGD and plot the median for the 100 similar entropy dice of the KL divergence per iteration between the actual parameter and the i-th estimation. Following (Bottou, 2012), the learning rate γ_t is restricted to $\gamma_t = \frac{a}{b+t}$ where $a \in \{10^m | -2 \leq m \leq 7\}$ and $b \in \{10^m | -5 \leq m \leq 8\}$ were selected to minimize the error (measured as the KL divergence from the real parameter to the estimated parameter) in the last 10 observations. The results can be observed in Figure 3.1.

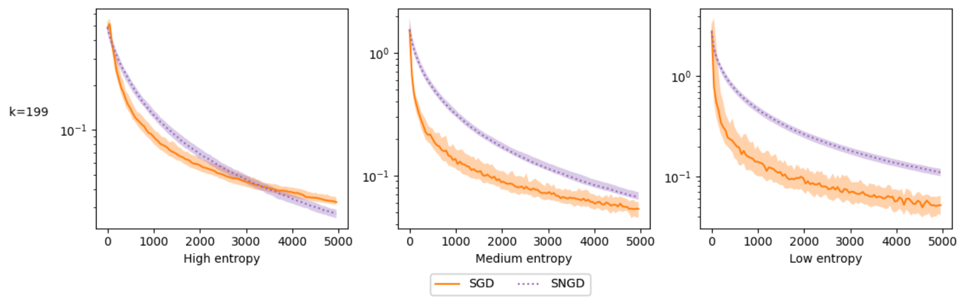


Figure 3.1: KL-Divergence of estimations per iteration of SGD and SNGD on the die problem.

Experiments show a small gain for SNGD with respect to SGD in the high entropy scenario, no gain in the medium entropy scenario, and a clear loss in the low entropy scenario. We argue that this is due to difficulties in convergence for SNGD which force the selection of learning rates performing extremely small steps which harden the effectiveness of SNGD. We add the interquartile range for every method with the shaded area. The interquartile range is thinner for SNGD, implying that the estimates have lower variance.

3.2 Manifold optimized descent

Manifold Optimized Descent (MOD) is a natural gradient based algorithm appearing in (Sánchez-López, 2018). It rises from the need of reducing the high computational complexity of NGD and control the divergence trends. The idea behind MOD is to keep using metric information, avoiding nevertheless its update at every iteration. The number of iterations where the metric is not updated is determined by a positive integer e (standing for *eras*). If the algorithm starts at some point $Z_0 = \eta_0$, the first iteration copies a NGD step, that is

$$X_0 = G_{Z_0}^{-1} \cdot \nabla f(Z_0) \,, \tag{3.3}$$

where X_t stands for the director process (see section 2.1.3). However, for the next iterations (as many as *eras*) the director process is defined as

$$X_t = G_{Z_0}^{-1} \cdot \nabla f(Z_t) \qquad 0 \leq t < e. \tag{3.4}$$

Observe that the same matrix is used while $t < e$. Then, at iteration $t = e$, again a NGD step is performed, followed by some iterations (as many as *eras*) where matrix $G^{-1}(Z_e)$ is exploited in the update. This is summarized by Definition 3.2.1.

Definition 3.2.1. *Let $(\mathcal{M}, g)$ be a Riemannian Manifold of dimension k and let f be a differentiable function defined in $(\mathcal{M}, g)$. Let ϕ be a parametrization of the manifold. Manifold Optimized Descent (MOD) is a stochastic process $Z = (X, \gamma)$ such that*

$$X_t = G_{Z_{\lambda e}}^{-1} \cdot \nabla f(Z_t) \qquad \lambda e \leq t < (\lambda + 1)e \;\; \lambda \in \mathbb{N} \,, \tag{3.5}$$

where G_η stands for the metric matrix at $\eta \in \mathbb{R}^k$ in ϕ parametrization.

In (Sánchez-López, 2018) we recommend starting with the highest entropy point of the statistical manifold as Z_0, where the metric information is as smooth as it can be.

If *eras* is large (e is large), MOD basically follows gradient steps modified by a constant matrix for many iterations. Hence, MOD closely relates to a gradient based algorithm. As positive integer e decreases to 1, MOD gets closer to NGD becomig exactly NGD when $e = 1$ as it can be seen in Algorithm 7.

Algorithm 7: Manifold optimized descent

```
  Result: $Z_t$
1 $Z_0, t = 0, eras$;
2 while stopping condition not satisfied do
3     $M = G^{-1}_{Z_t}$;
4     for $e = 0$ to eras do
5         $X_t = M \cdot \nabla f(Z_t)$ ;
6         $Z_{t+1} = Z_t - \gamma_t \cdot X_t$;
7         $t = t + 1$;
8     end
9 end
```

We defined the offline version of MOD for simplicity, however, the online version can be derived easily, by redefining the director process. See Section 2.5.2 for the notation employed in Definition 3.2.2. The reader can check the reference (Sánchez-López, 2018) for a particular version of SMOD for the MLR problem.

Definition 3.2.2. *Let $(\mathcal{M}, g)$ be a Riemannian Manifold of dimension k, let l be a differentiable loss function defined in $(\mathcal{M}, g)$ and let ϕ be a parametrization of the manifold. Let $\omega = (\omega_0, \omega_1, \omega_2, ...) \in \Omega$ be a sample drawn from the product probability space. Stochastic Manifold Optimized Descent (SMOD) is a stochastic process $Z = (X, \gamma)$ such that*

$$X_t(\omega) = G^{-1}_{Z_{\lambda e}(\omega)} \cdot \nabla l(Z_t(\omega), \omega_t) \qquad \lambda e \leq t < (\lambda + 1)e, \;\; \lambda \in \mathbb{N}, \tag{3.6}$$

where G_η stands for the metric matrix at $\eta \in \mathbb{R}^k$ in ϕ parametrization.

3.2.1 Computational complexity

Computational complexity per iteration of MOD is lower than that of NGD. The reason is that MOD is not computing the inverse of a matrix at every iteration, but once in *eras*. Moreover, it is necessary to compute the regular gradient and the matrix-vector product. If the dimension of the manifold is k, assume that A_k corresponds to the cost of computing the gradient of f at a given point and that B_k is the cost of inverting a matrix of dimension k. Hence, after e many iterations, the algorithm performed $eO(k^2) + eA_k + B_k$ operations, corresponding to e matrix-vector products, e many gradient computations and one inverse matrix respectively. That is, per iteration, the computational cost of MOD is

$$C(MOD) = O(k^2) + A_k + \frac{B_k}{e}. \tag{3.7}$$

If we assume that $B_k \in O(k^3)$, in particular, this means that the computational complexity of MOD can be reduced to quadratic when $e = k$ and A_k has quadratic complexity cost at most.

3.2.2 Maximum entropy gradient descent

A particular setting of MOD explained in (Sánchez-López, 2018) consist on using the matrix $M = G^{-1}_{\eta_m}$ for the director process, where η_m represents the maximum entropy probability distribution of $\mathcal{M}$.

Definition 3.2.3. *Let $(\mathcal{M}, g)$ be a Statistical Manifold of dimension k and let f be a differentiable function defined in $(\mathcal{M}, g)$. Let ϕ be a parametrization of the manifold. Let η_m refer to the maximum entropy probability distribution of $(\mathcal{M}, g)$. Maximum Entropy Gradient Descent (MEGD) is a stochastic process $Z = (X, \gamma)$ such that*

$$X_t = M \cdot \nabla f(Z_t)\ , \tag{3.8}$$

where $M = G^{-1}_{\eta_m}$ and G_η stands for the metric matrix at $\eta \in \mathbb{R}^k$ in ϕ parametrization.

In every step, the gradient at Z_t is computed and multiplied by the constant matrix M. That is why this algorithm can not be considered as a natural gradient descent variant, but instead, is a rescaled parametrization version of GD. This algorithm can be seen as a particular example of MOD where $e = \infty$, that is, the metric matrix is never updated. Hence, more specifically, previous algorithm MOD is in fact a trade-off between NGD and MEGD where variable *eras* allows the transition between them. MEGD is represented in Algorithm 8.

Algorithm 8: Maximum entropy gradient descent

```
    Result: $Z_t(\omega)$
1   $Z_0 = \eta_m, M = G^{-1}_{\eta_m}, t = 0$;
2   while stopping condition not satisfied do
3       $X_t = M \cdot \nabla f(Z_t)$ ;
4       $Z_{t+1} = Z_t - \gamma_t \cdot X_t$;
5       $t = t + 1$;
6   end
```

As expected, this algorithm is proven to converge using for example the convergence theorem appearing in (Sunehag et al., 2009) under certain regularities, just as GD does. However, the computational complexity could possibly be not as low as that of GD, since a matrix-vector product is demanded at every iteration that increases the complexity order to quadratic.

Also, the stochastic version of MEGD can be easily obtained, as given by Definition 3.2.4.

Definition 3.2.4. *Let $(\mathcal{M}, g)$ be a Statistical Manifold of dimension k, let l be a differentiable loss function defined in $(\mathcal{M}, g)$ and let ϕ be a parametrization of the manifold. Let η_m refer to the maximum entropy probability distribution of $(\mathcal{M}, g)$. Let $\omega = (\omega_0, \omega_1, \omega_2, ...) \in \Omega$ be a sample drawn from the product probability space. Stochastic Maximum Entropy Gradient Descent (SMEGD) is a stochastic process $Z = (X, \gamma)$ such that*

$$X_t(\omega) = M \cdot \nabla l(Z_t(\omega), \omega_t) \ , \tag{3.9}$$

where $M = G^{-1}_{\eta_m}$ and G_η stands for the metric matrix at $\eta \in \mathbb{R}^k$ in ϕ parametrization.

3.3 Experiments

This section reproduces the die problem described in Section 3.1.2. For the experiments, algorithms SGD, SNGD and on-line MOD (and MEGD) are considered. Two experiments are performed to test the behavior of MEGD and MOD respectively. In the first one, we experimentally show that using only the information provided by the maximum entropy point metric is enough to greatly surpass the solution quality of standard SGD, at least in such a toy problem. In the second one, as expected, the reader is able to see the transition of MOD from natural gradient based SNGD to gradient based MEGD.

3.3.1 MEGD solving the die problem

We first repeat the die problem experiment of Section 3.1.2 including now the algorithm MEGD, which uses a matrix but never updates it. This can be seen in Figure 3.2.

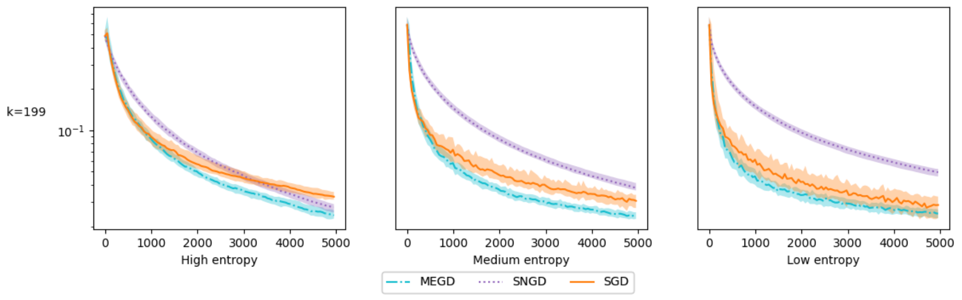


Figure 3.2: KL-Divergence of estimations per iteration of SGD, SNGD and MEGD on the die problem.

Figure 3.2 shows that convergence issue of SNGD is not repeated for MEGD. That may be caused by the fact that MEGD is not a natural gradient based algorithm, and because it is a proven convergence optimization method. Moreover, it shows better estimations with less variance compared to SGD, as the interquartile ranges are thinner. Our guess is that even if we are using only the maximum entropy metric for every point in the space, such a matrix gathers information of the die (corresponding to most uncertainty), in contrast with SGD which provides no information at all to the updates. However, no remarkable enhancement is appreciated in the estimations, nor a better rate of convergence speed is observed.

3.3.2 MOD solving the die problem

This section empirically assesses the optimization performance of MOD. We are interested in watching the transition from gradient based algorithms to natural gradient algorithms, in terms of convergence and solution quality. Method MOD is perfect for the task since variable *eras* controls the trade-off between such two groups of algorithms. Hence for this experiment, we repeat the die problem running several instances of MOD algorithms with different values for *eras*. As a reminder, MOD algorithm with *eras* equal to 1 is exactly the natural gradient based SNGD, while MOD instance with *eras* equal or higher than 5000 (which is the length of the sample) is the gradient based algorithm MEGD.

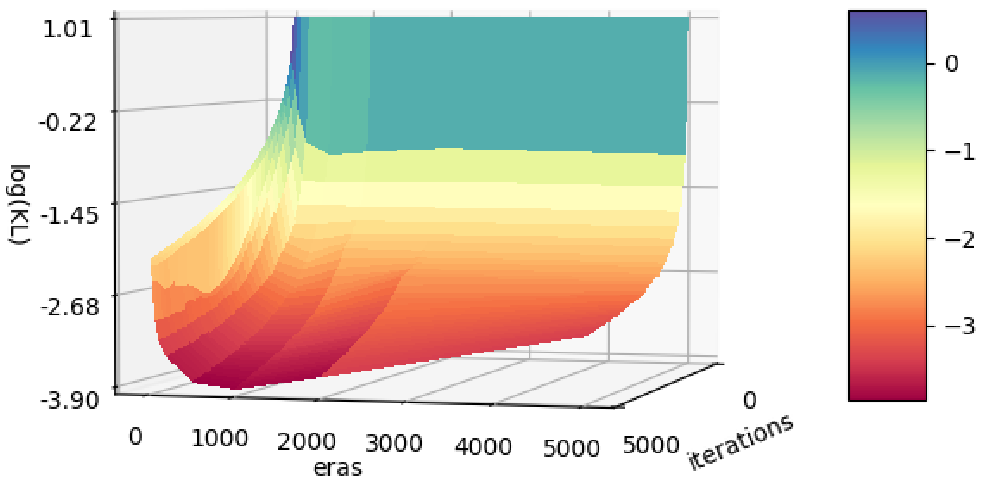


Figure 3.3: Logarithm of the KL-Divergence of estimations per iteration of MOD with different *eras* on the die problem..

The results of the experiment are given in Figure 3.3 with only the low entropy case included. Only MOD algorithm with several values of *eras* is considered. 'Iterations' axis and vertical axis together are used to show the error made by estimations along a learning process, just as Figure 3.2. The axis 'eras' determines the value of *eras* employed for MOD. Hence a cross-section fixing a value for axis 'eras' presents the learning process of MOD with a such value of *eras*. Just to clarify, cross-section corresponding to the plane with 'eras' fixed to 1 (leftest) is exactly SNGD while cross-section determined by 'eras' fixed to 5000 (rightest) is MEGD learning process. The transition of MOD can be appreciated. As *eras* increases, MOD improves its estimations, thus lowering the KL divergence. This is the case until *eras* reaches the values near 1000. For upper values of *eras*, the algorithm performs worse, approaching to MEGD estimates as *eras* approaches 5000. Figure 3.3 reveals that for *eras* value fixed around 1000, MOD potentially defines its fastest convergence speed version. To assess this, we add MOD algorithm with $eras = 1000$ to Figure 3.2 to compare empirically the rate of convergence in Figure 3.4.

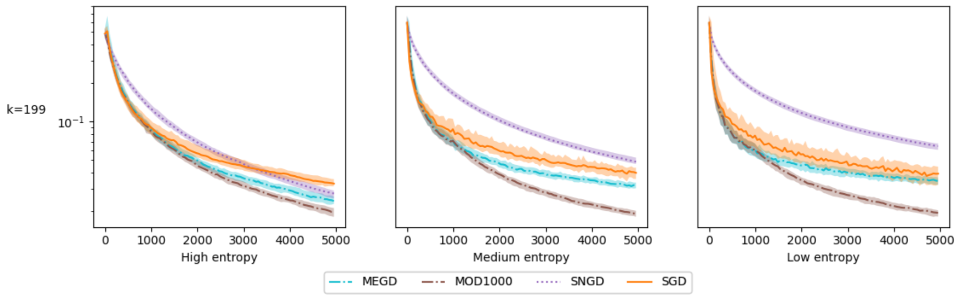


Figure 3.4: KL-Divergence of estimations per iteration of MOD with $eras = 1000$, SGD, SNGD and MEGD on the die problem.

Observe Figure 3.4. With a first glance, the reader may notice that two groups of curves can be spotted according to their general shape. SGD and MEGD curves are closer to horizontal lines while MOD and SNGD curves maintain their slope. Realize that this fact classifies gradient and natural gradient based optimization methods. It is of particular interest pointing out that natural gradient based algorithm MOD, with a range of values for *eras* around 1000, reaches better estimates of the solution than gradient based algorithms SGD and MEGD. But our main observation relates to the seemingly higher orders of convergence speed exhibited by MOD: while gradient based algorithms seem to radically reduce their convergence speed by flattening their curve, MOD algorithm keeps a steep curve downside improving over other optimization methods and promising to enlarge this advantage along more training. Hence, natural gradient algorithms have the potential to widely overcome gradient based algorithms. Nevertheless, the behavior of MOD is not always stable. Only for some values of *eras*, the algorithm seems to both converge, and also maintain a great convergence speed.

3.4 Comments

This chapter explored some issues that usually discard natural gradient based algorithms as optimization problem solvers. Those are computational complexity and convergence.

MOD optimization method is not reducing the computational complexity to that of SGD neither his convergence is proven theoretically. It only makes it to show the potential of natural gradient based algorithms in our toy experiments.

We wonder whether natural gradient based algorithms can be brought to solve real practical problems. That is natural gradient based optimization methods whose behavior is stable and whose computational complexity is reduced significantly. The next chapters of the thesis pursue the challenging task of creating convergence-proven, computationally efficient algorithms.

4 Convergent stochastic natural gradient descent

As explained in Chapter 3, SNGD convergence problems can harm its performance. That chapter proposed a natural gradient based algorithm called MOD to assess and partially overcome this issue. MOD seems to converge fast for certain values of *eras* when the metric matrix is forced to stabilize for several iterations. However, the theoretical convergence of MOD remains unknown.

The objective of this chapter is to propose a SNGD-like algorithm for optimization on a Riemannian manifold $(\mathcal{M}, g)$ with proven convergence.

Recall that, assuming a parametrization ϕ is fixed, SNGD update equation is defined by its director process (see Section 2.5.5):

$$X_t(\omega) = G^{-1}_{Z_t(\omega)} \cdot \nabla l(Z_t(\omega), \omega_t) \ . \tag{4.1}$$

where G_η is the metric matrix at $\eta \in \mathbb{R}^k$ in ϕ parametrization.

Current results about convergence with variable metrics (Sunehag et al., 2009) require the metric matrix $M_t = G^{-1}_{Z_t}$ to have bounded eigenvalues. This can be shown by proving that M_t is a convergent sequence of matrices that converges to a positive-definite matrix by continuity arguments (as we prove in Section 4.1.1). This is easy to prove if we assume the sequence Z_t is convergent, but that is exactly what we aim at proving. The fundamental idea is to eliminate this direct dependence of M_t and Z_t to achieve convergence. To effectively decouple the two sequences, we redefine M_t.

Section 4.1 explores this idea and develops the theoretical part to then create a new optimization method CSNGD. Furthermore, the proof of convergence of CSNGD is derived from the work of (Sunehag et al., 2009). In Section 4.2, CSNGD solves the die problem of Section 3.1.2 in different scenarios of entropy and manifold dimension. The algorithm shows a stable behavior in every experiment, competing with the maximum a posteriori estimates, which are known to be the best for such toy problems. The chapter concludes with some comments in Section 4.3 summarizing the achievements made so far and evaluating the challenges yet to be faced.

4.1 Convergent stochastic natural gradient descent

The main intuition is as follows; to effectively decouple the sequences M_t and Z_t, build M_t from a different sequence $\overline{Z}_t$ whose convergence can be proven. This new stochastic process $\overline{Z}_t$ is defined in the same probability space as Z_t, and similarly, as with Z_t in Equation 2.37, it must only depend on observations up to time $i < t$;

$$\begin{aligned} &\overline{Z}_0 \in \mathbb{R}^k \\ &\overline{Z}_t(\omega) = \overline{Z}_t(\omega_0, ..., \omega_{t-1}) \quad t > 0 \ . \end{aligned} \tag{4.2}$$

Thus, the main idea behind Convergent Stochastic Natural Gradient Descent (CSNGD) is to keep two independent sequences: $\overline{Z}_t$ for which we use an already known convergent estimation method such as SGD and Z_t which is the estimator of the optimum defined by the director process. See Section 2.5.2 for the notation used in Definition 4.1.1

Definition 4.1.1. *Let $(\mathcal{M}, g)$ be a Riemannian Manifold of dimension k, let l be a differentiable function defined in $(\mathcal{M}, g)$ and let ϕ be a parametrization of the manifold. Let $\omega = (\omega_0, \omega_1, \omega_2, ...) \in \Omega$ be a sample drawn from the product probability space. Let $\overline{Z}$ be a convergent stochastic process. Convergent Stochastic Natural Gradient Descent (CSNGD) is a stochastic process $Z = (X, \gamma)$ such that*

$$X_t(\omega) = G^{-1}_{\overline{Z}_t(\omega)} \cdot \nabla l(Z_t(\omega), \omega_t) \ , \tag{4.3}$$

where G_η stands for the metric matrix at $\eta \in \mathbb{R}^k$ in ϕ parametrization.

The reader can find the instructions of CSNGD in Algorithm 9.

Algorithm 9: Convergent stochastic natural gradient descent

```
Result: $Z_t(\omega)$
1 $Z_0, \omega = (\omega_0, \omega_1, ...), \{\overline{Z}_0(\omega), \overline{Z}_1(\omega), ...\}, t = 0$;
2 while stopping condition not satisfied do
3     $X_t(\omega) = G^{-1}_{\overline{Z}_t(\omega)} \cdot \nabla l(Z_t(\omega), \omega_t)$;
4     $Z_{t+1}(\omega) = Z_t(\omega) - \gamma_t X_t(\omega)$;
5     $t = t + 1$;
6 end
```

The main difference between SNGD and CSNGD is that instead of using the metric at Z_t, CSNGD uses the metric at $\overline{Z}_t$, which is known to be a convergent sequence. It is desirable to choose for $\overline{Z}_t$ a sequence that converges to the solution. This is because Z_t converges to the solution (see Theorem 4.1.1), and then the natural gradient approximation given by equation 4.3 is more accurate as iterations go on and sequences Z_t and $\overline{Z}_t$ get closer. Hence, if $\overline{Z}_t$ converges to the solution, the more iterations, the closer CSNGD is to SNGD. However, CSNGD has a proven convergence shown in Section 4.1.1.

CSNGD optimization method is originally stated for online optimization, but this contribution can be trivially exploited for offline optimization as well by just setting

$$X_t = G^{-1}_{\overline{Z}_t} \cdot \nabla f(Z_t) \,. \tag{4.4}$$

4.1.1 CSNGD convergence proof

The main result of this section is the following Theorem.

Theorem 4.1.1. *Let $f : \mathbb{R}^k \to \mathbb{R}$ be a twice differentiable function with a unique minimum $\overline{\eta} \in \mathbb{R}^k$ and let $(\mathcal{M}, G)$ be a Riemannian Manifold of dimension k. Then, CSNGD converges to* *the minimum $\overline{\eta}$ almost surely if the following conditions hold*

$$\begin{aligned}
&\textbf{\textit{C.2}}\ (\exists K)(\forall \eta)\ \|\nabla^2 f(\eta)\| \leq 2K \\
&\textbf{\textit{C.3}}\ (\forall \delta > 0) \inf_{f(\eta)-f(\eta^*)>\delta} \|\nabla f(\eta)\| > 0 \\
&\textbf{\textit{C.4'}}\ (\exists A, B)(\forall t)\ \mathbb{E}_t \|\nabla l(Z_t, \cdot)\|^2 \leq A + Bf(Z_t) \\
&\textbf{\textit{C.6 Learning rate constraint}}
\end{aligned} \tag{4.5}$$

Before proceeding with the proof, let us analyze the conditions of Theorem 4.1.1. Conditions **C.2,C.3** are exactly the same as in Theorem 2.7.2 already commented in Section 2.7. Condition **C.4'** is a particular case of condition **C.4** with $Y(Z_t) = \nabla l(Z_t, \cdot)$.

Proof. The proof relies on applying the work of (Sunehag et al., 2009). The result needed is recalled in this thesis in the preliminaries Chapter 2 as Theorem 2.7.2.

To prove Theorem 4.1.1, test that conditions **C.1**-**C.6** in Theorem 2.7.2 are satisfied. As a matter of notation, realize that

$$\begin{aligned}
Y_t &= \nabla l(Z_t, \cdot) \\
B_t &= G^{-1}_{\overline{Z}_t} \quad .
\end{aligned} \tag{4.6}$$

Conditions **C.2**, **C.3** and **C.6** are already imposed by Theorem 4.1.1. **C.4** is also imposed since corresponds with **C.4'**. Condition **C.1** is clear by the assumptions on the loss function l (see Equation 2.34). Thus, we only need to prove condition **C.5** to complete the proof. Also realize that the assumption stated in Theorem 2.7.2 saying that "B_t is symmetric and only depends on information available at time t" is accomplished, since $\overline{Z}_t$ is known at time t, and therefore $B_t = G^{-1}_{\overline{Z}_t}$ is a real valued, fixed matrix at time t.

C.5 is checked in two steps. First, prove that B_t converges to a positive definite matrix. Second, prove that condition **C.5** is fulfilled.

Since $\overline{Z}_t$ is convergent, say convergent to $\overline{Z}_\infty$, and G^{-1} is a continuous function, then B_t converges to $B = G^{-1}_{\overline{Z}_\infty}$, which is a symmetric positive definite matrix.

Continue with the second step. Define continuous functions $\lambda_{\min}(M) = \min spec(M)$ and $\lambda_{\max}(M) = \max spec(M)$ where $spec(M)$ are the eigenvalues of matrix M. Since B is a positive definite matrix then $L_{\min} = \lambda_{\min}(B) > 0$ and $L_{\max} = \lambda_{\max}(B) > 0$. Select

$\epsilon \in \mathbb{R}$ such that $0 < \epsilon < L_{\min}$. Because B_t converges to B and $\lambda_{\min}$ and $\lambda_{\max}$ are continuous, there exists s such that for every $t > s$ all eigenvalues of matrix B_t belong to interval $(L_{\min} - \epsilon, L_{\max} + \epsilon)$.

Hence,

$$\begin{aligned} a &= \min(L_{\min} - \epsilon, \min_{0 \le t \le s} \lambda_{\min}(M_t)) > 0 \\ b &= \max(L_{\max} + \epsilon, \max_{0 \le t \le s} \lambda_{\max}(M_t)) < \infty \,, \end{aligned} \tag{4.7}$$

fulfill condition **C.5**, completing the proof of Theorem 4.1.1. □

4.2 Experiments

We wonder about the solution quality of CSNGD in the die problem. As shown in Section 3.1.2, employing the natural gradient to optimize a function may lead to divergence symptoms. However, CSNGD is convergent by virtue of Theorem 4.1.1. So this point of the thesis is an inflection point because we are now able to judge the hypothesis of the thesis. All experiments of the section can be found in (Sánchez-López and Cerquides, 2022a).

If our hypothesis is correct, convergent natural gradient based algorithms, as CSNGD, should show high efficiency and convergence speed. Therefore, the first experiment consists of adding CSNGD to the die problem. CSNGD is a good candidate to test whether the convergence property provides stability to natural gradient based algorithms or not. Sequence $\overline{Z}_t$ is chosen to be the one generated by SGD as it is known to converge.

CSNGD is solving the die problem in Figure 4.1. Our convergent natural gradient algorithm shows great efficiency. It improves over the other algorithms in every scenario. It is remarkable that natural gradient based algorithms keep showing a specific shape of the curve (steeper) different from the flatter curve drawn by gradient based algorithms, as already observed in Figure 3.4. In fact, MOD can already reach the solution quality of CSNGD. However, a relevant difference with CSNGD is that CSNGD does not require to tune any parameter, while MOD requires to locate the appropriate values for *eras* in order to obtain good results.

We decide to run a wider variety of experiments to test CSNGD with dice of dimensions $49, 199$ and 499 and high, medium, and low entropy. Moreover, we added the Maximum Like-

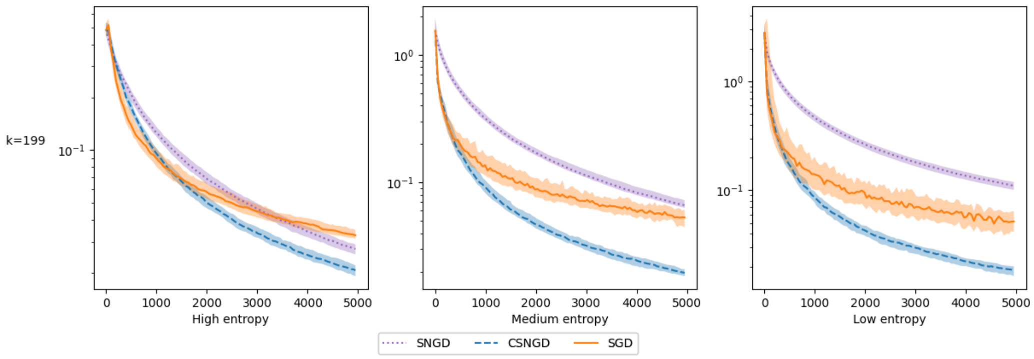


Figure 4.1: KL-Divergence of estimations per iteration of CSNGD, SGD and SNGD on the die problem.

lihood Estimator (MLE) which provides the best estimations as more observations are processed. It maximizes the likelihood up to time t. To compute the MLE, we use the expectation parametrization η^* of the exponential family, recall Section 2.3.5. If a sample of observations $\omega = (\omega_0, \omega_1, ...) \in \Omega$ is drawn, the MLE algorithm for the die problem is defined by

$$X_t(\omega) = \Theta_t(\omega) - T(\omega_t) \ , \tag{4.8}$$

where $\gamma_t = \frac{1}{1+t}$. The algorithm is described in more detail in Algorithm 10.

Algorithm 10: Maximum likelihood estimator (die problem)

```
  Result: $\Theta_t(\omega)$
1 $\Theta_0, t = 0, \omega = (\omega_0, \omega_1, ...)$;
2 while stopping condition not satisfied do
3     $X_t(\omega) = \Theta_t(\omega) - T(\omega_t)$;
4     $\Theta_t(\omega) = \Theta_t(\omega) - \frac{1}{t+1} X_t(\omega)$;
5     $t = t + 1$;
6 end
```

It turns out that this algorithm for the die problem is equivalent to SNGD run in the dual space (see proof in (Sánchez-López, 2018)). This also implies that MLE reproduces exactly the same updates as mirror descent. Figure 4.2 is the result of the experiment. Stability and

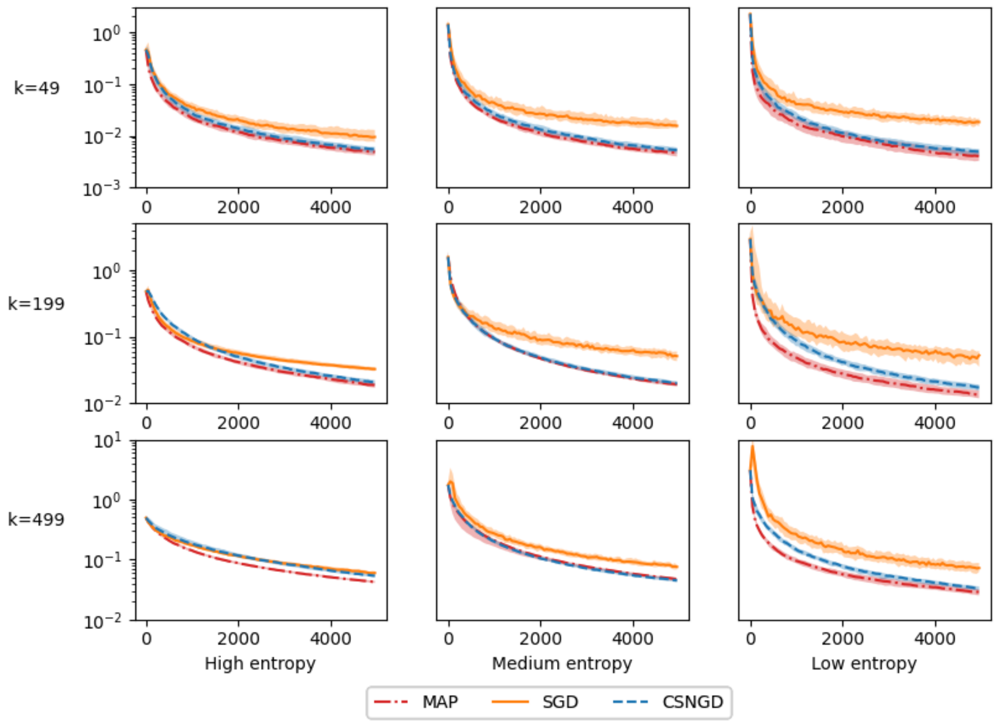


Figure 4.2: KL-Divergence of estimations per iteration of ML, SGD and CSNGD on the die problem.

efficiency shown by CSNGD is great, imitating MLE closely. Therefore CSNGD is an optimization method that approximates the natural gradient and avoids the divergence issues of SNGD. Furthermore, the estimations of the solution found by CSNGD in many of the scenarios tested are close to that of MLE, meaning that the convergence speed is experimentally optimal. Only on the scenario where dimension is $k = 499$ and entropy is high, CSNGD behaves closely as SGD. But it may be caused by the fact that only in this scenario SGD is not flattening its learning process curve producing great estimations of the solution as well.

4.3 Comments

This chapter has introduced a natural gradient based optimization method called CSNGD. We have managed to prove its convergence. This is possible thanks to a convergent auxiliary stochastic process $\overline{Z}_t$ which is plugged in the metric matrix, instead of the main sequence Z_t, when computing the natural gradient. This yields a stable sequence of metric matrices which generates a convergent optimization method.

Experiments reflect great stability for CSNGD with a high convergence speed in every scenario exposed, where different dimensions and entropy levels are considered. This supports our hypothesis about the convergence and speed convergence of convergent natural gradient based optimization methods.

However, CSNGD is not efficient, in the sense of computational complexity. The algorithm scales really badly in the dimension k of the manifold since a matrix inverse (or linear system solving) is demanded in every iteration. In addition, the algorithm has to perform matrix-vector products. This fact discards CSNGD to solve high dimensional problems.

The research takes us to the next challenge: defining a convergent natural gradient based algorithm whose computational complexity order is comparable to that of SGD. Chapter 5 gathers all the knowledge and contributions made so far to face the objective. To that end, the reader will see that the efficient and convergent natural gradient based algorithm DSNGD is defined at the expense of some restrictions on the problem to solve.

5 Dual stochastic natural gradient descent

In Chapter 4 we define a natural gradient based algorithm called CSNGD and afterward, we prove its convergence. However, the high computational complexity of the natural gradient is not addressed. This chapter aims to define a natural gradient based optimization method whose convergence is proved and whose computational complexity is reduced to that of SGD.

To that end, the chapter has to restrict to a particular learning problem in ML: the Multinomial Logistic Regression (MLR) problem (see Section 2.6). The main assumption of MLR (Banerjee, 2007) is that the log-odds ratio of the class posteriors $P(\mathcal{Y} \mid \mathcal{X})$ is an affine function of the features $\mathcal{X}$.

Banerjee in (Banerjee, 2007) proved (Theorem 2) that a class of distributions fulfills the core MLR assumption if and only if for each value of $\mathcal{Y}$, the class of conditional distributions $P(\mathcal{X} \mid \mathcal{Y})$ belongs to the same Linear Exponential Family (LEF) (for the definition of LEF see Section 2.3.5 or visit (Wani, 1968)). Such result is used in Section 5.1 to prove that the class of joint distributions $P(\mathcal{X}, \mathcal{Y})$ is also a LEF. It is well known that a LEF is a DFM (manifold introduced in Section 2.3.4, or see (Amari, 2016)). Usually, finding the minimum expected risk MLR parameters is formulated as an optimization problem in $\mathbb{R}^k$ which is solved by means of SGD. Instead, this thesis proposes to formulate the problem as a manifold optimization problem (Hu et al., 2020), over the manifold $(\mathcal{M}, g)$ of probability distributions $P(\mathcal{X}, \mathcal{Y})$ fulfilling the main assumption of MLR, where the metric equipped is the FIM defined in Section 2.3.2. The joint distribution is also considered in (Lin et al., 2019) for instance, where dual parameterizations allow to run a fast SNGD.

The chapter starts by establishing in Section 5.1 that the family of joint distributions $P(\mathcal{X}, \mathcal{Y})$ satisfying the main MLR assumption is a LEF and hence a DFM. Then, it is possible to rely on duality to provide efficient computation of the natural gradient of the conditional log-loss function in Section 5.2. Dual Stochastic Natural Gradient Descent (DSNGD) is defined in Section 5.3. The chapter finishes by providing the convergence and linear computational complexity results for the case where $\mathcal{X}$ is a set of discrete variables.

5.1 MLR generative model. The joint distribution

The next result proves that the the family of joint distributions $P(\mathcal{X}, \mathcal{Y})$ satisfying the core MLR assumption is a LEF.

Proposition 5.1.1. *The log-odds ratio of the class posteriors $P(\mathcal{Y} \mid \mathcal{X})$ is an affine function of the features $\mathcal{X}$ if and only if the joint distribution $P(\mathcal{X}, \mathcal{Y})$ belongs to LEF.*

Furthermore, there exists the LEF natural parametrization of the joint distribution

$$\begin{aligned} P_\eta(x, y) &= \frac{\exp\left(S(y)^\intercal \alpha + T(x)^\intercal \beta_y\right)}{\lambda(\eta)} \\ \lambda(\eta) &= \int_x \sum_y \exp\left(S(y)^\intercal \alpha + T(x)^\intercal \beta_y\right) , \end{aligned} \tag{5.1}$$

where $\eta = (\alpha, \beta)$, $\alpha \in \mathbb{R}^{s-1}$, $\beta \in \mathbb{R}^{s \times t}$, β_y is the y-th row of β and

$$\begin{aligned} T &: \Omega \to \mathbb{R}^t \\ S &: [1, ...s] \to \mathbb{R}^{s-1} , \end{aligned} \tag{5.2}$$

are sufficient and minimal statistics of $\mathcal{X}$ and $\mathcal{Y}$ respectively.

The proof of this proposition relies strongly on theorem 2 in (Banerjee, 2007) and can be found in appendix $B.1$.

The logistic regression core assumption then translates into assuming $P(\mathcal{X}, \mathcal{Y})$ belongs to the exponential family, and such space is a well known DFM (Amari, 2016; Nielsen, 2018). This is convenient for the purpose of this chapter, because, after Section 2.3.4, in a DFM the costs of natural gradient computations can be highly reduced, based on the property shown by Equation 2.27. Next, we provide the dually flat parametrization of $P(\mathcal{X}, \mathcal{Y})$.

5.1.1 Dually flat parametrization of the joint distribution

We have seen that $P(\mathcal{X}, \mathcal{Y})$ is a DFM and that we can choose the natural parametrization of Equation 5.1. The conditional probability distributions with η parametrization are

$$P_\eta(y \mid x) = \frac{\exp\left(S(y)^\intercal \alpha + T(x)^\intercal \beta_y\right)}{\sum_y \exp\left(S(y)^\intercal \alpha + T(x)^\intercal \beta_y\right)} \,. \tag{5.3}$$

The parameterization of Equation 5.3 can be simplified, by considering Definition 5.1.1.

Definition 5.1.1. *Let S be a statistic of a discrete random variable $\mathcal{Y} = \{1, ..., s\}$. S is a canonical statistic if*

$$\begin{aligned} S(i) =& e_{s-1}(i) \qquad \textit{for } 1 \leq i < s \,, \\ S(s) =& 0 \in \mathbb{R}^{s-1} \,, \end{aligned} \tag{5.4}$$

where $e_j(i)$ is the i-th canonical vector of dimension j.

With a linear transformation, we can assume that S is a canonical statistic in Equations 5.1 and 5.3, as Proposition 5.1.2 states. Its proof can be found in Appendix B.2.

Proposition 5.1.2. *The minimal and sufficient statistic S of $\mathcal{Y}$ in the joint distribution can be transformed into a canonical statistic with a linear transformation.*

This implies that we can assume, for the sake of simplicity, that S is a canonical statistic from now on unless otherwise stated.

As (Amari, 2016) proves, the exponential family manifold is built after the convex function $F(\eta) = \log \lambda(\eta)$ (see the DFM construction from a convex function in Section 2.3.4). In (Amari, 2016), the author also proves that this Riemannian manifold derived from $F(\eta)$ has the FIM metric, as is usually considered for statistical manifolds. Recall that the FIM metric is defined as

$$G_\eta = -\mathbb{E}_{x,y \sim P_\eta}\left[\nabla^2 \log P_\eta(x, y)\right] \,. \tag{5.5}$$

The dual parametrization $\eta^* = \nabla F(\eta)$ can also be considered (see Section 2.3.5). For LEF, it is called the expectation parametrization and it is shown in this section as Equation 5.6.

For more properties of the dual parametrization see (Amari, 2016). To simplify the notation, if $x = \begin{pmatrix} x_1 & \cdots & x_n \end{pmatrix}$, we note $\nabla_x = \begin{pmatrix} \frac{\partial}{\partial x_1} & \cdots & \frac{\partial}{\partial x_n} \end{pmatrix}^\intercal$. So for every $i \in \{1, ..., s\}$ write

$$\begin{aligned}
\alpha^* &= \nabla_\alpha F(\eta) = \sum_y S(y) P_\eta(y) = \mathbb{E}_\mathcal{Y}[S(y)] = (P_\eta(\mathcal{Y} = 1), ..., (P_\eta(\mathcal{Y} = s-1))^\intercal \,, \\
\beta_i^* &= \nabla_{\beta_i} F(\eta) = P_\eta(\mathcal{Y} = i) \int_\mathcal{X} T(x) P_\eta(x \mid \mathcal{Y} = i) = P_\eta(\mathcal{Y} = i) \mathbb{E}_{\mathcal{X}|\mathcal{Y}=i}[T(x)] \,.
\end{aligned} \tag{5.6}$$

Define $\eta^* = (\alpha^*, \beta^*)$ with $\beta^* = (\beta_1^*, ..., \beta_s^*)$ the dual parameterization, or equivalently, the expectation parameters.

Observe that $P(\mathcal{Y})$ is the categorical distribution (since $\mathcal{Y}$ is discrete and finite) and therefore it is a LEF, where α^* contains actually the expectation parameters. Moreover

$$\theta_i := \theta_i(\alpha^*, \beta_i^*) = \frac{\beta_i^*}{P_{\eta^*}(\mathcal{Y} = i)} = \mathbb{E}_{\mathcal{X}|\mathcal{Y}=i}[T(x)] \,, \tag{5.7}$$

are the expectation parameters of the conditional distribution $P(\mathcal{X} \mid \mathcal{Y} = i)$.

5.2 Fast natural gradient of the log-loss

This section allows us to compute the natural gradient of the log-loss function without having to use the metric matrix directly but using both dual parametrizations instead.

Given $(x, y) \in \mathcal{X} \times \mathcal{Y}$ and $\eta \in \mathbb{R}^k$, the log-loss function is defined as

$$l(\eta, x, y) = -\log P_\eta(y \mid x) \tag{5.8}$$

Proposition 5.2.1 reveals $\widetilde{\nabla} l(\eta, x, y)$ using both dual parametrizations η and η^*.

Proposition 5.2.1. *Let l be the log-loss function. Then, if $P(\mathcal{X}, \mathcal{Y})$ is a DFM, it is*

$$\widetilde{\nabla} l(\eta, x, y) = \nabla h(\eta^*, x) \cdot (q_\mathcal{Y}(x, P_\eta) - e_s(y)) \tag{5.9}$$

where

$$q_\mathcal{Y}(x, P) = \begin{pmatrix} P(\mathcal{Y} = 1|x) \\ \vdots \\ P(\mathcal{Y} = s|x) \end{pmatrix}, \tag{5.10}$$

$h(\eta^*,x) = (\log P_{\eta^*}(\mathcal{Y} = 1,x), \ldots \log P_{\eta^*}(\mathcal{Y} = s,x))$ and $e_s(k)$ is the k-th canonical s-dimensional vector.

The proof of Proposition 5.2.1 is presented in Appendix B.3. Proposition 5.2.1 expresses the natural gradient of the log-loss function without the need of inverting a matrix, although it makes use of both dual parametrizations η and η^*. It is an opportunity to define fast natural gradient based algorithms that avoid inverse metric matrices. An important remark is that the natural gradient is written as the product of two terms where one of them is clearly bounded (the term $(q_{\mathcal{Y}}(x, P_\eta) - e_s(y))$). Hence, divergence symptoms of the natural gradient are possibly created by the possibly unbounded term $\nabla h(\eta^*,x)$. So it is at our disposal a strategy to create fast natural gradient based algorithms that are convergent as well: if we control the term $\nabla h(\eta^*,x)$, then the algorithm may stabilize and converge. In Section 5.3 we put into practice these ideas to define a natural gradient based algorithm named Dual Stochastic Natural Gradient Descent (DSNGD).

5.3 Dual stochastic natural gradient descent

Dual Stochastic Natural Gradient Descent (DSNGD) aims to solve the MLR optimization problem using the natural parametrization η of the LEF on $\mathcal{X} \times \mathcal{Y}$: If $(\overline{\Omega}, \overline{\mathcal{F}}, \overline{P})$ is a probability space where $\overline{\Omega} = \mathcal{X} \times \mathcal{Y}$ and $\overline{P}$ is an unknown probability distribution, optimize $f(\eta) = \mathbb{E}_{x,y\sim\overline{P}}[l(\eta,x,y)]$ for $\eta \in \mathbb{R}^k$ where $l(\eta,x,y)$ is the conditional log-loss function. The solution $\overline{\eta} \in \mathbb{R}^k$ to this problem refers to the conditional distributions $P_{\overline{\eta}}(\mathcal{Y} \mid \mathcal{X})$ that better fits the unknown conditional distributions $\overline{P}(\mathcal{Y}|\mathcal{X})$. To that end, we define a stochastic natural gradient based algorithm. By looking to the notation introduced in Section 2.5, in this scenario an observation drawn on the product probability space is of the form $\omega = (\omega_1, \omega_2, \ldots)$ such that $\omega_t = (x_t, y_t) \in \overline{\Omega}$ for all $t \in \mathbb{N}$.

Using Proposition 5.2.1 and assuming that a sample of observations $\omega = (\omega_0, \omega_1, \ldots)$ is drawn from the product probability space, DSNGD is defined in Definition 5.3.1.

Definition 5.3.1. *Let $(\mathcal{M}, g)$ be the Riemannian Manifold of dimension k where $\mathcal{M}$ is the probability space of joint distributions $P(\mathcal{X},\mathcal{Y})$ belonging to LEF and g is the FIM. Let l be* *the conditional log loss function defined in $(\mathcal{M}, g)$ and let ϕ be the natural parametrization of the manifold. Let $\omega = (\omega_0, \omega_1, \omega_2, ...) \in \Omega$ be a sample drawn from the product probability space. The Dual Stochastic Natural Gradient Descent (DSNGD) is a stochastic process $Z = (X, \gamma)$ such that*

$$X_t(\omega) = \nabla h(\zeta_t^*(\omega), x_t) \cdot (q_{\mathcal{Y}}(x_t, P_{Z_t(\omega)}) - e_s(y_t)) \ , \tag{5.11}$$

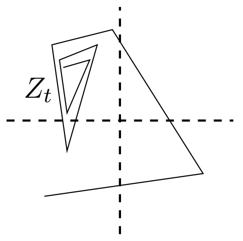
where $\{\zeta_t\}_{t\in\mathbb{N}}$ is a convergent sequence in the natural parametrization and $\{\zeta_t^*\}_{t\in\mathbb{N}}$ is the same sequence expressed in the dual parametrization.

Note that $q_{\mathcal{Y}}(x_t, P_{Z_t})$ is a stable term (it only takes values between 0 and 1). Moreover, DSNGD forces the stability of the $\nabla h(\zeta_t^*(\omega), x_t)$ term, since ζ_t is a convergent sequence. This is the same strategy of CSNGD, and similarly, it is going to ensure the convergence of the algorithm in Theorem 5.6.2. Observe that Equation 5.11 is also well defined when the parametrization is not minimal, therefore DSNGD can be run in such a general case, where S and T are not minimal. Steps taken by DSNGD are specified in Algorithm 11.

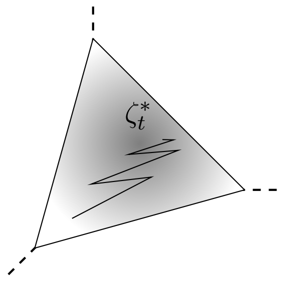
Algorithm 11: Dual stochastic natural gradient descent

```
   Result: $Z_t(\omega)$
1  $Z_0, \omega = ((x_0, y_0), (x_1, y_1), ...), \{\zeta_0^*(\omega), \zeta_1^*(\omega), ...\}, t = 0$;
2  while stopping condition not satisfied do
3      $X_t(\omega) = \nabla h(\zeta_t^*(\omega), x_t) \cdot (q_{\mathcal{Y}}(x_t, P_{Z_t(\omega)}) - e_s(y_t))$;
4      $Z_{t+1}(\omega) = Z_t(\omega) - \gamma_t X_t(\omega)$;
5      $t = t + 1$;
6  end
```

The sequence $\{\zeta_t^*\}_{t\in\mathbb{N}}$, or simply ζ_t^* as an abuse of notation, can be any sequence in the dual space whose expression in the natural parametrization $\{\zeta_t\}_{t\in\mathbb{N}}$ is convergent. For example, it can be constant. Or also, it can be the sequence of estimations provided by SGD, if a such sequence converges. The resulting algorithm keeps track of two independent sequences; the main sequence Z_t which estimates the solution $\overline{\eta}$ to the problem, and the sequence ζ_t^* selected with the convergence constraint and whose space is the dual. For example, assume the trivial case where $\mathcal{X} = \{0\}$ and $\mathcal{Y} = \{0, 1, 2\}$. The only conditional probability distribution of the problem is the Categorical distribution $P(\mathcal{Y} \mid \mathcal{X} = 0)$. This space is represented by $\mathbb{R}^2$ and its dual space is represented by the simplex S^2. Then, the main sequence Z_t moves in $\mathbb{R}^2$ while the independent sequence ζ_t^* traces its path in S^2. Figure 5.1 illustrates iterations followed by Z_t (instruction line 4 of the algorithm) and ζ_t^* when running DSNGD for this simple example.



(a) Z_t sequence in $\mathbb{R}^2$



(b) ζ_t^* sequence in the dual space S^2

Figure 5.1: Z_t and ζ_t^* sequences obtained in DSNGD where $\mathcal{X} = \{0\}$ and $\mathcal{Y} = \{0, 1, 2\}$

Recall that the sequence ζ_t^* can be chosen freely as long as its dual is convergent. However, DSNGD is designed a natural gradient based algorithm. The algorithm effectively takes a natural gradient step only when Z_t and ζ_t^* refer to the same probability distribution point, according to equation 5.11 and Proposition 5.2.1. In Section 5.6 there is our proof of DSNGD convergence to the solution $\overline{\eta}$, and if ζ_t^* is selected such that it also converges to the solution, then both sequences get closer along the optimization process, turning DSNGD steps into more accurate approximations of natural gradient steps. Therefore, in order to benefit from natural gradient speed up properties, it is recommended that sequence ζ_t^* converges to the solution $\overline{\eta}^* = \nabla F(\overline{\eta})$. For example, this can be accomplished by determining ζ_t^* using a maximum a posteriori estimator of the parameters of $P(\mathcal{X}, \mathcal{Y})$ obtained from data up to t.

5.4 Computational complexity of Natural Gradient

To evaluate the computational complexity of using equation 5.9 we determine an expression of $\nabla h(\eta^*, x)$ with respect to the expectation parameters θ_y of $\mathcal{X}$ given $\mathcal{Y}$ already mentioned in

Equation 5.7. The following notation is used

$$(5.12)\qquad K_i = \left(\begin{array}{c|c} & -1 \\ Id^{i-1} & \vdots \\ & -1 \end{array}\right), \qquad d(x,y,\eta^*) = \frac{1-\theta_y^\intercal \nabla_{\theta_y} \log P_{\theta_y}(x \mid y)}{P_{\eta^*}(y)},$$

and the proof is shown in Appendix B.4.

Proposition 5.4.1.

$$(5.13)\qquad \nabla h(\eta^*,x) = \begin{pmatrix} \nabla_{\alpha^*} h(\eta^*,x) \\ \nabla_{\beta_1^*} h(\eta^*,x) \\ \vdots \\ \nabla_{\beta_s^*} h(\eta^*,x) \end{pmatrix},$$

where

$$(5.14)\qquad \begin{aligned} \nabla_{\alpha^*} h(\eta^*,x) =& K_s \cdot diag(d(x,1,\eta^*),...,d(x,s,\eta^*)) \\ \nabla_{\beta_k^*} h(\eta^*,x) =& \frac{\nabla_{\theta_k} \log P_{\theta_k}(x \mid \mathcal{Y}=k) \cdot e_s(k)^\intercal}{P_{\eta^*}(\mathcal{Y}=k)}. \end{aligned}$$

The complexity analysis of the natural gradient is presented now, and the reader can find the proof in appendix B.5. The result uses the expectation parameters θ_y of the conditional distribution specified in Equation 5.7.

Proposition 5.4.2. *The computational complexity of the natural gradient $\widetilde{\nabla} l(\eta,x,y)$ using Proposition 5.2.1 is $O(s(A+t))$ where A is the cost of computing $\nabla_{\theta_y} \log P_{\theta_y}(x \mid y)$, s is the number of classes and t is the dimension of statistic T.*

Observe that the manifold dimension is $k = s-1+st$ and therefore, a computation is linear on the number of the variables of the model if its complexity order is $O(k) = O(s(1+t)) = O(st)$. Therefore, the costs of computing the natural gradient can be reduced to linear if the cost A is low enough, precisely, if sA is at most linear ($O(sA) \leq O(k)$). This is the case when $\mathcal{X}$ is discrete and finite, as presented in Section 5.5.

5.5 Computational complexity of discrete DSNGD

This section assumes that space $\mathcal{X}$ is discrete, that is $\mathcal{X} = \{1,...,m\}$ for some $m \in \mathbb{N}$. For simplicity, we assume T to be also a canonical statistic (see Definition 5.1.1). Theorem 5.5.1

deduces and proves that the complexity order for discrete DSNGD of one iteration is linear on the dimension of the parameter η. Let us show a simple example of discrete DSNGD to begin with.

5.5.1 Example

Let $\mathcal{Y} = \{1, 2\}$ and $\mathcal{X} = \{1, 2\}$ and minimal and canonical statistics S and T. Let $\eta = (\alpha, \beta)$ be the natural parameter and $\zeta^* = (\alpha^*, \beta^*)$ be the independent dual parameter. Observe that in this case, α and α^* are 1-element vectors and β and β^* are 2×1 matrices. In this example, we complete an iteration of the discrete DSNGD algorithm, following the instructions listed in algorithm 11.

Let $(x, y) = (1, 2)$ be an observation. Statistics T and S are assumed to be canonical. To complete instruction line 3 start computing $q_{\mathcal{Y}}(x, P_Z)$ and $\nabla h(\zeta^*, x = 1)$. Using Equation 5.3, it is

$$q_{\mathcal{Y}}(x, P_Z) = \begin{pmatrix} P(\mathcal{Y} = 1 \mid x) \\ P(\mathcal{Y} = 2 \mid x) \end{pmatrix} = R \cdot \begin{pmatrix} \exp(\alpha_1 + \beta_1) \\ \exp \beta_2 \end{pmatrix} , \tag{5.15}$$

where $R = \frac{1}{\exp(\alpha_1+\beta_1)+\exp\beta_2}$. For the term $\nabla h(\zeta^*, x = 1)$, use Equation 5.6, then apply the gradient.

$$h(\zeta^*, x = 1) = (\log \beta_1^*, \log \beta_2^*) \to \nabla h(x = 1, \zeta^*) = \begin{pmatrix} 0 & 0 \\ \frac{1}{\beta_1^*} & 0 \\ 0 & \frac{1}{\beta_2^*} \end{pmatrix} . \tag{5.16}$$

Finish instruction line 3 by computing the approximation of the natural gradient and the director process that DSNGD uses for the Z_t update.

$$\begin{aligned} X_t =& \nabla h(\zeta^*, x = 1)(q_{\mathcal{Y}}(x, P_\eta) - e_2(y = 2)) \\ =& \begin{pmatrix} 0 & 0 \\ \frac{1}{\beta_1^*} & 0 \\ 0 & \frac{1}{\beta_2^*} \end{pmatrix} \cdot \begin{pmatrix} R \cdot \exp(\alpha_1 + \beta_1) \\ (R \cdot \exp \beta_2) - 1 \end{pmatrix} \\ =& \begin{pmatrix} 0 \\ \frac{R\cdot\exp(\alpha_1+\beta_1)}{\beta_1^*} \\ \frac{(R\cdot\exp\beta_2)-1}{\beta_2^*} \end{pmatrix} . \end{aligned} \tag{5.17}$$

The next instruction lines of the algorithm are standard to update the parameter vector $(\alpha, \beta_1, \beta_2)$ using the director process X_t, so there is no need to go further. If for instance, the observation obtained is $(x, y) = (2, 2)$, then the approximation of the natural gradient is

$$X_t = \begin{pmatrix} \frac{R \exp \alpha_1}{\alpha_1^* - \beta_1^*} - \frac{R-1}{1-\alpha_1^* - \beta_2^*} \\ \frac{-R \cdot \exp \alpha_1}{\alpha_1^* - \beta_1^*} \\ \frac{R-1}{1-\alpha_1^*-\beta_2^*} \end{pmatrix}, \tag{5.18}$$

where $R = \frac{1}{1+\exp \alpha_1}$.

5.5.2 Discrete DSNGD linear computational complexity proof

Before analyzing the computational complexity of DSNGD, it is necessary to determine the generator of ζ_t^* sequence. Sequence ζ_t^* belongs to the dual space of the LEF distributions on $\mathcal{X} \times \mathcal{Y}$, and if S and T are canonical statistics then it implies that ζ_t^* are directly the probabilities $P(x, y)$ after Equation 5.6. It is possible to select the well-known maximum a posteriori estimator with parameter $a \in \mathbb{R}$. This estimator is a simple counting of observations over the discrete space $\mathcal{X} \times \mathcal{Y}$ with a starting assumption of incidence of a for every event x, y. This estimator is linear and it clearly converges (to the solution).

First, a result similar to Proposition 5.4.1 is stated, taking into account the discreteness assumption on $\mathcal{X}$. The proof of Proposition 5.5.1 is found in appendix B.6.

Proposition 5.5.1. *Let $\mathcal{X} = \{1, ..., m\}$ and let T be a minimal and canonical statistic. Then*

$$\begin{aligned} \nabla_{\alpha^*} h(\eta^*, x) &= \begin{cases} 0 & x \neq m \\ K_s \cdot diag(\frac{1}{P_{\eta^*}(x, \mathcal{Y}=1)}, ..., \frac{1}{P_{\eta^*}(x, \mathcal{Y}=s)}) & x = m \end{cases} \\ \nabla_{\beta_y^*} h(\eta^*, x) &= \frac{1}{P_{\eta^*}(x, y)} \cdot \begin{cases} e_{m-1}(x) \cdot e_s(y)^\intercal & x \neq m \\ -\boldsymbol{1}_{m-1} \cdot e_s(y)^\intercal & x = m \end{cases} \end{aligned} \tag{5.19}$$

where $\boldsymbol{1}_n \in \mathbb{R}^n$ is a vector filled with ones at every coordinate.

Now it is possible to analyze the computational complexity of discrete DSNGD. Theorem 5.5.1 proves that DSNGD, just as SGD, is a linear algorithm.

Theorem 5.5.1. *Let $\mathcal{X} = \{1, ..., m\}$ and let T be a minimal and canonical statistic. Assume estimator ζ^* of DSNGD is linear. Then discrete DSNGD iterations have linear complexity order on the manifold dimension.*

Proof. Let $k = (s-1) + s \cdot t$ be the dimension of the manifold and hence, the dimension of parameter η. Then $O(k) = O(st)$. Analyze the computational complexity of discrete DSNGD. That is, analyze the computational cost of instruction line 3 and 4 shown in Algorithm 11.

Complexity of instruction line 3 is given by Proposition 5.4.2, which is $O(sA + st))$ where sA is the cost of computing $\nabla_{\theta_y} \log P_{\theta_y}(x \mid y)$ for all $y \in \mathcal{Y}$. Observe Equations 5.6 and 5.7 assuming T canonical and write

$$\begin{aligned} \alpha^* &= (P_{\zeta^*}(\mathcal{Y} = 1), ..., P_{\zeta^*}(\mathcal{Y} = s-1))^\intercal \\ \theta_y &= (P_{\zeta^*}(\mathcal{X} = 1 \mid y), ..., P_{\zeta^*}(\mathcal{X} = m-1 \mid y))^\intercal \,. \end{aligned} \tag{5.20}$$

Deduce then that after Proposition 5.5.1 it is $O(sA) = O(k)$.

Instruction line 4 adds k operations coming from the learning rate and X_t product, and k operations more of vector subtraction. That is a total of $2k$ operations.

Finally, recall that a linear complexity order estimator is chosen for ζ_t^* sequence, implying we should consider $O(k)$ operations more.

In conclusion, the computational complexity order of DSNGD is

$$O(sA + st) + 2k + O(k) = O(k) \,, \tag{5.21}$$

and therefore linear. □

5.6 Discrete DSNGD and convergence

This section proves the convergence of the discrete DSNGD. Discrete DSNGD refers to the case where $\mathcal{X} = \{1, ..., m\}$ for some $m \in \mathbb{R}$. Recall Theorem 3.2 in (Sunehag et al., 2009) introduced in Section 2.7 as Theorem 2.7.2). The section starts by generalizing this result in Section 5.6.1. This generalization provides enough flexibility so as to be used later to prove the convergence of DSNGD in Section 5.6.2.

5.6.1 Generalizing Sunehag et. al. variable metric stochastic approximation theory

Theorem 2.7.2 is used to prove CSNGD convergence in this thesis, however it can not be used to prove DSNGD convergence. First, because it requires the vector it follows to be factored as the product of a symmetric and positive definite matrix B_t and a vector Y_t that approximates the gradient (condition **C.1**). But DSNGD is defined to directly approximate the natural gradient, without the gradient as reference. And second, even if DSNGD is written as the product of a matrix and a vector, matrix $\nabla h(\zeta_t^*, x_t)$ is not squared. So we need a more general convergence theorem, a result that directly contemplates the director process X assuming no further factorization of such term.

The main modification with respect to Theorem 2.7.2 is the unification of conditions **C.1** and **C.3**

$$\begin{aligned} &\textbf{C.1} \quad (\forall t) \quad \mathbb{E}_t Y_t = \nabla l(\eta_t) \\ &\textbf{C.3} \quad (\forall \delta > 0) \quad \inf_{l(\eta)-l(\overline{\eta})>\delta} \|\nabla l(\eta)\| > 0\,, \end{aligned} \tag{5.22}$$

to instead require

$$\textbf{C.3'} \quad (\forall \delta > 0) \quad \inf_{l(Z_t)-l(\overline{\eta})>\delta} \nabla l(Z_t)^T \mathbb{E}_t\left[X_t\right] > 0\,. \tag{5.23}$$

Theorem 2.7.2 imposes that the expectation of the step taken must be the gradient and that the norm of the gradient must not approach zero outside any environment of the minimum. Instead, we impose that the expectation of the step taken must not approach the border of the half-space which has the gradient as its normal vector, unless we are approaching the minimum simultaneously. This is a more general condition. Furthermore, condition **C.5** on the maximum and minimum eigenvalues of the matrix B_t can also be removed. In fact, our result can be used to prove the convergence of algorithms with scaling matrices B_t whose spectrum is not bounded from below by a strictly positive number, as long as the new version of condition **C.3'** holds.

The result is formally stated in Theorem 5.6.1. Proof can be found in appendix B.7.

Theorem 5.6.1. *Let $f : \mathbb{R}^k \to \mathbb{R}$ be a twice differentiable function with a unique minimum $\overline{\eta}$ and $Z_{t+1} = Z_t - \gamma_t X_t$. Then Z_t converges to $\overline{\eta}$ almost surely if the following conditions hold*

C.2 $(\exists K)(\forall \eta)\ \|\nabla^2 f(\eta)\| \leq 2K$

C.3' $(\forall \delta > 0)\ \inf_{l(Z_t)-l(\overline{\eta})>\delta} \nabla f(Z_t)^T \mathbb{E}_t\left[X_t\right] > 0$

C.4 $(\exists A, B)(\forall t)\ \mathbb{E}_t\|X_t\|^2 \leq A + Bl(Z_t)$

C.6 Learning rate constraint

5.6.2 Discrete DSNGD convergence proof

Next, Theorem 5.6.1 is used to to prove DSNGD convergence in the discrete case. That is, we use it to prove Theorem 5.6.2.

Theorem 5.6.2. *Discrete DSNGD with canonical statistics S and T converges almost surely to the optimum.*

The proof consists of showing that conditions **C.2, C.3', C.4** and **C.6** of Theorem 5.6.1 hold. Condition **C.6** is assumed to hold, by just selecting an appropriate sequence of learning rates γ_t. Conditions **C.2** and **C.4** are proved in Appendices B.8 and B.9 respectively. Proof of condition **C.3'** is shown below.

Proof. Compute the gradient of $f(\eta)$ (see Equation B.16) and use Proposition 5.2.1 to obtain $\mathbb{E}_t\left[X_t\right]$ involved in condition **C.3'**.

$$\begin{aligned}\nabla f(\eta) =& \mathbb{E}_t\left[\nabla l(\eta, x, y)\right]\\ =& \sum_x \nabla h(\eta, x) \sum_y (q_{\mathcal{Y}}(x, P_\eta) - e_s(y))\overline{P}(x, y)\\ =& \sum_x \nabla h(\eta, x) R_{\mathcal{Y}}(x, \eta)\\ \mathbb{E}_t\left[X_t\right] =& \sum_x \nabla h(\zeta^*, x) R_{\mathcal{Y}}(x, \eta)\ ,\end{aligned} \tag{5.24}$$

where

$$R_{\mathcal{Y}}(x, \eta) = (q_{\mathcal{Y}}(x, P_\eta) - q_{\mathcal{Y}}(x, \overline{P}))\overline{P}(x)\ . \tag{5.25}$$

Further evolve equation 5.24 to finally multiply $\nabla f(\eta)^\intercal \mathbb{E}_t\left[X_t\right]$ and check condition **C.3'**. Continue by developing $\nabla f(\eta)$ first, precisely compute $\nabla h(\eta, x)$. To simplify the notation, decompose $\nabla = (\nabla_\alpha, \nabla_{\beta_1}, ..., \nabla_{\beta_s})$

$$\begin{aligned} \nabla_\alpha h(\eta,x) &= S + u(P_\eta)\cdot(1,...,1) & u(P) &= -\sum_y S(y)P(y)\ , \\ \nabla_{\beta_y} h(\eta,x) &= T(x)e_s(y)^\intercal + v(y,P_\eta)\cdot(1,...,1) & v(y,P) &= -\sum_x T(x)P(x,y)\ , \end{aligned} \tag{5.26}$$

where S is the $s-1\times s$ matrix having $S(i)$ as i-th column. Since $(1,...,1)\cdot R_{\mathcal{Y}}(x,\eta)=0$ then

$$\begin{aligned} \nabla_\alpha f(\eta) =& \sum_x \nabla_\alpha h(\eta,x) R_{\mathcal{Y}}(x,\eta) \\ =& \sum_x S\cdot R_{\mathcal{Y}}(x,\eta) \\ =& S\cdot R_{\mathcal{Y}}(\eta)\ , \\ \nabla_{\beta_y} f(\eta) =& \sum_x \nabla_{\beta_y} h(\eta,x) R_{\mathcal{Y}}(x,\eta) \\ =& \sum_x T(x)e_s(y)^\intercal R_{\mathcal{Y}}(x,\eta) \\ =& T\cdot R_{\mathcal{X}}(y,\eta)\ \ , \end{aligned} \tag{5.27}$$

where T is the $m-1\times m$ matrix having $T(i)$ as i-th column.

$$\begin{aligned} R_{\mathcal{Y}}(\eta) &= \begin{pmatrix} P_\eta(\mathcal{Y}=1) - \overline{P}(\mathcal{Y}=1) \\ \vdots \\ P_\eta(\mathcal{Y}=s) - \overline{P}(\mathcal{Y}=s) \end{pmatrix} \\ R_{\mathcal{X}}(y,\eta) &= \begin{pmatrix} (P_\eta(\mathcal{Y}=y\mid\mathcal{X}=1) - \overline{P}(\mathcal{Y}=y\mid\mathcal{X}=1))\overline{P}(\mathcal{X}=1) \\ \vdots \\ (P_\eta(\mathcal{Y}=y,\mathcal{X}=m) - \overline{P}(\mathcal{Y}=y\mid\mathcal{X}=m))\overline{P}(\mathcal{X}=m) \end{pmatrix}\ . \end{aligned} \tag{5.28}$$

Now develop $\mathbb{E}_t X_t$ further. Recall that S and T are canonical statistics so plug in Proposition 5.5.1 into Equation 5.24. Decompose $\mathbb{E}_t = (\mathbb{E}_{t,\alpha^*}, \mathbb{E}_{t,\beta_1^*}, ..., \mathbb{E}_{t,\beta_s^*})$

$$\begin{aligned}\mathbb{E}_{t,\alpha^*}\left[X_t\right] =& \sum_x \nabla_{\alpha^*} h(\zeta^*, x) R_{\mathcal{Y}}(x,\eta) \\ &= K_s \cdot diag(d(m,1,\zeta^*), ..., d(m,s,\zeta^*)) \cdot R_{\mathcal{Y}}(m,\eta) \ , \\ \mathbb{E}_{t,\beta_y^*}\left[X_t\right] =& \sum_x \nabla_{\beta_y^*} h(\zeta^*, x) R_{\mathcal{Y}}(x,\eta) \\ &= K_m \cdot diag(d(1,y,\zeta^*), ..., d(m,y,\zeta^*)) \cdot R_{\mathcal{X}}(y,\eta) \ .\end{aligned} \tag{5.29}$$

Proceed now to check the condition. Develop the products until obtaining

$$\begin{aligned}\nabla_\alpha f(\eta)^\intercal \mathbb{E}_{t,\alpha^*}\left[X_t\right] =& \sum_y c(y) \ , \\ \nabla_{\beta_y} f(\eta)^\intercal \mathbb{E}_{t,\beta_y^*}\left[X_t\right] =& - c(y) + \sum_x d(x,y,\zeta^*)(P_\eta(y|x) - \overline{P}(y|x))^2 \overline{P}(x)^2 \ ,\end{aligned} \tag{5.30}$$

where $c(y) = d(m,y,\zeta^*)(P_\eta(y) - \overline{P}(y))(P_\eta(y|x=m) - \overline{P}(y|x=m))\overline{P}(x=m)$.

Finally,

$$\begin{aligned}\nabla f(\eta)^\intercal \mathbb{E}_t\left[X_t\right] =& \nabla_\alpha f(\eta)^\intercal \mathbb{E}_{t,\alpha^*}\left[X_t\right] + \sum_y \nabla_{\beta_y} f(\eta)^\intercal \mathbb{E}_{t,\beta_y^*}\left[X_t\right] \\ =& \sum_y c(y) + \sum_y -c(y) + \sum_x d(x,y,\zeta^*)(P_\eta(y|x) - \overline{P}(y|x))^2 \overline{P}(x)^2 \\ =& \sum_{y,x} d(x,y,\zeta^*)(P_\eta(y|x) - \overline{P}(y|x))^2 \overline{P}(x)^2 \ .\end{aligned} \tag{5.31}$$

Notice in equation 5.31 that $\nabla f(\eta)^\intercal \mathbb{E}_t\left[X_t\right]$ is a sum of positive numbers, and it vanishes only if $\eta = \overline{\eta}$. Also, since $d(x,y,\zeta^*) > 1$, observe that

$$\begin{aligned}\nabla f(\eta)^\intercal \mathbb{E}_t\left[X_t\right] >& \sum_{y,x} (P_\eta(y|x) - \overline{P}(y|x))^2 \overline{P}(x)^2 \ , \\ =& \sum_y \|R_{\mathcal{X}}(y,\eta)\|^2 \ .\end{aligned} \tag{5.32}$$

To finish proving the result, let $\{\eta_i\}_{i\in\mathbb{N}}$ be a sequence such that

$$\sum_y \|R_{\mathcal{X}}(y,\eta_i)\|^2 \xrightarrow[i\to\infty]{} 0 \ , \tag{5.33}$$

since every term is positive, then for every $y \in \mathcal{Y}$

$$\|R_{\mathcal{X}}(y, \eta_i)\|^2 \xrightarrow[i\to\infty]{} 0 \ , \tag{5.34}$$

implying that $P_{\eta_i}(y|x) - \overline{P}(y|x) \xrightarrow[i\to\infty]{} 0$ for all x, y and that

$$l(\eta_i) - l(\overline{\eta}) \xrightarrow[i\to\infty]{} 0 \ . \tag{5.35}$$

Hence it's proven

$$(\forall \delta > 0) \qquad \inf_{f(\eta)-f(\overline{\eta})>\delta} \sum_{y} \|R_{\mathcal{X}}(y, \eta_i)\|^2 > 0 \ , \tag{5.36}$$

and therefore, after Equation 5.32, condition **C.3'** holds. □

5.7 Experiments

This section runs some experiments testing the behavior of our natural gradient based algorithm DSNGD. The purpose is to figure out whether the convergence property and the low computational complexity of the algorithm are reflected positively in practical problems. As a comparative reference, we add the fast SGD algorithm. Check the reference (Sánchez-López and Cerquides, 2022b) to reproduce or inspect the code of the experiments.

The experiment settings are similar to that of Section 4.2: we consider 3 different manifolds $\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3$ and 3 different scenarios concerning the entropy (high, medium, and low) of the hidden probability distribution that generates the data.

We define the manifolds used in the experiments. Variables $\mathcal{Y}$ and $\mathcal{X}$ are discrete. Furthermore, we assume variable $\mathcal{X}$ splits into $\mathcal{X} = (\mathcal{X}_1, \mathcal{X}_2, ..., \mathcal{X}_q)$ where $\mathcal{X}_i$ is discrete, holding the Naive Bayes independence assumption;

$$P(\mathcal{X}_i, \mathcal{X}_j \mid \mathcal{Y}) = P(\mathcal{X}_i \mid \mathcal{Y})P(\mathcal{X}_j \mid \mathcal{Y}) \qquad \forall i \neq j \tag{5.37}$$

If Y is a discrete variable having values in $\{1, ..., s\}$ we just write $Y = s$. In the case of $\mathcal{X} = (\mathcal{X}_1, \mathcal{X}_2, ..., \mathcal{X}_q)$ where $\mathcal{X}_i = \{1, ..., m_i\}$ we just write $\mathcal{X} = (m_1, m_2, ..., m_q)$. Recall that

k stands for the dimension of the manifold. With this notation in mind, the manifolds considered are

$$\mathcal{M}_1 \begin{cases} Y & = 10 \\ X & = (10,5) \\ k & = 139 \end{cases} , \qquad \mathcal{M}_2 \begin{cases} Y & = 20 \\ X & = (10,5,10,5) \\ k & = 539 \end{cases} ,$$

(5.38)

$$\mathcal{M}_3 = \begin{cases} Y & = 30 \\ X & = (10,5,10,5,10,5) \\ k & = 1199 \end{cases} .$$

The 3 different scenarios are identified with parameters $\overline{\eta} \in \mathbb{R}^k$ generated randomly from the normal distribution $N(0,\sigma^2)^k$ where σ equals to $0.1, 0.5$ and 1 for the high, medium and low entropy scenarios respectively.

For every manifold and every entropy scenario, we run 100 instances. For every instance $\overline{\eta}$ we draw a 10^7 length sample from $P_{\overline{\eta}}$. The objective is to optimize the expectation of the conditional log-loss, which recall it is

(5.39)
$$f(\eta) = \mathbb{E}_{(x,y)\sim P_{\overline{\eta}}} l(\eta, x, y) ,$$
$$l(\eta, x, y) = -\log P_\eta(y \mid x) .$$

We consider the median and interquartile ranges of the 100 instances for every case.

For the learning rate selection, we run the algorithms with several learning rate candidates in 500 iterations over a newly generated sample. The candidate with the best estimations in the last 10 iterations is then chosen to solve the problem in the 10^7 length sample. Just as in Section 3.1.2, the learning rate γ_t is restricted to $\gamma_t = \frac{a}{b+t}$ where $a \in \{10^m | -2 \leq m \leq 7\}$ and $b \in \{10^m | -5 \leq m \leq 8\}$.

These large magnitudes of dimension and sample length can be contemplated in our experiments only because DSNGD is also a linear learning algorithm. Running higher-order methods such as SNGD would be unfeasible in terms of computational resources.

Figure 5.2 contains the learning process and interquartile range of DSNGD and SGD in the experiments. Columns represent different entropy scenarios while rows correspond to the manifolds $\mathcal{M}_1, \mathcal{M}_2$ and $\mathcal{M}_3$ by their dimensions. Results are promising for our natural gradient

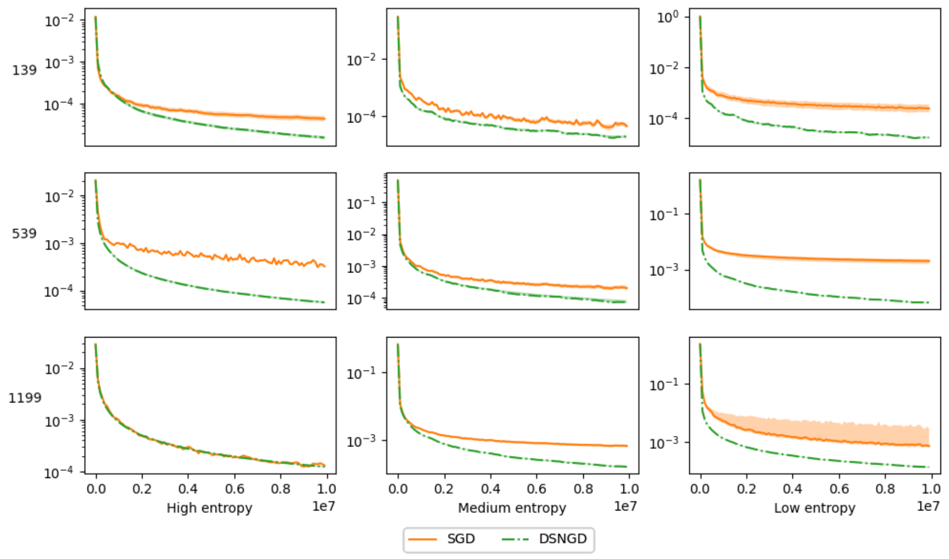


Figure 5.2: Median of KL-Divergences of estimations per iteration of SGD and DSNGD on the MLR problem.

algorithm since DSNGD surpasses SGD in almost every scenario. Only for $\mathcal{M}_3$ in the high entropy scenario standard SGD is able to compete with DSNGD. This may be caused by the fact that this experiment is the one with more uncertainty. In this scenario, staying in the highest entropy neighborhood (equiprobability) determined by $0 \in \mathbb{R}^k$ already hits a high-quality estimator. In this case, there is not much more room left for learning. For the rest of the cases, we can observe in general that algorithm DSNGD is not only finding better estimators of the solution by reducing the KL divergence but also the curve is presenting a steeper slope towards minimizing the error if more data was added to the learning process. In contrast, SGD usually shows a flat horizontal line meaning that further significant improvements in the quality of the estimations are impossible to reach in practice. If we take a look at the vertical axis, which is log scaled, the estimations of DSNGD are about one order of magnitude better than that of SGD. Furthermore, the interquartile range of DSNGD is thinner, which implies that the algorithm provides estimations with lower variance.

5.8 Comments

Natural gradient based algorithms behave erratically when tested in practical problems. However, as CSNGD shows, this kind of algorithm may stabilize once convergence is guaranteed. With this in mind, we defined DSNGD, which approximates the natural gradient at each step and whose convergence in the discrete case can be proved. To that end, we stated and proved a general result showing the convergence of interior half-space gradient approximations. Furthermore, we point out that this convergence result may prove the convergence of more general algorithms since it doesn't require the expectation of the update's direction to factor as a symmetric positive-definite matrix and the gradient.

The convergence proof of DSNGD was not possible to obtain from the stochastic convergence results in the literature. Instead, our generalization of such convergence theorems is convenient for proving DSNGD convergence. In order to realize the conditions that lead natural gradient based algorithms to converge, we decided to continue our research on stochastic process convergence.

6 Convergence of stochastic processes that *resemble* to conservative vector fields

The hypothesis of this thesis states that a natural gradient based algorithm is stabilized when a such method has proven convergence. Chapters 3,4 and 5 seem to support this hypothesis. Natural gradient based algorithms can be understood as stochastic processes. Hence, convergence theorems of stochastic processes are a key tool for defining stable natural gradient based optimization methods.

This chapter starts generating a ground theory oriented to stochastic process convergence proofs. Our point of view allows a geometric perception of the conditions that result in convergence property. The stochastic process considered in this chapter holds some bound constraints on its decomposition of 1-increments, described in Section 6.1. The main contribution of the chapter is a generalization of convergence theorems and it is stated in Section 6.2. To completely understand our theorem, Section 6.3 and Section 6.4 define two key concepts, which are the *Expected direction set* and *resemblance*. Afterward, the proof of our theorem is shown in Section 6.5. Finally, convergence theorems appearing in (Bottou, 1998) and (Sunehag et al., 2009), recalled in Section 2.7, are deduced as corollaries from our main convergence theorem, proving this claim in the end of the chapter in Section 6.6. Our main result also proves convergence of the discrete DSNGD of Chapter 5.

6.1 Director process and learning rate bound constraints

This section starts by recalling some basic concepts of stochastic processes and it provides some properties and definitions that are helpful to study the almost sure convergence to a point of a stochastic process. Since this section exploits concepts already given in this thesis, we strongly recommend checking first Section 2.1. Moreover, the reader can expand their knowledge about the topic by visiting (Ross, 1996; Bass, 2011; Billingsley, 1986). Section 6.1.1 defines conditions on the decomposition elements of $Z = (X, \gamma)$ that we will assume during the whole chapter.

As a reminder, let $(\Omega, \mathcal{F}, P)$ be a probability space and (S, Σ) be a measurable space. A discrete stochastic process on $(\Omega, \mathcal{F}, P)$ indexed by $\mathbb{N}$ is a sequence of random variables $Z = \{Z_t\}_{t\in\mathbb{N}}$ such that $Z_t : \Omega \to S$. In this chapter, $S = \mathbb{R}^k$, and Σ is the corresponding Borel σ-algebra. As random variables are used to describe general random phenomena, stochastic processes indexed by $\mathbb{N}$ are usually used to model random sequences.

Recall Definition 2.1.11 about the decomposition of a stochastic process Z into $Z = (X, \gamma)$. Remember that $X = \{X_t\}_{t\in\mathbb{N}}$ is a stochastic process on $(\Omega, \mathcal{F}, P)$ called the director process of Z and $\gamma = \{\gamma_t\}t \in \mathbb{N}$ is a sequence of positive numbers named the learning rate.

This way of expressing a stochastic process allows us to define Z_{t+1} with respect to Z_t, which gives us control of the difference between both values by means of $\gamma_t X(t)$, as Figure 6.1 shows. This is really useful if we plan to analyze the convergence of a stochastic process. The naming of γ as learning rate is commonly used in the ML research branch (Bottou, 1998; Duchi et al., 2011; Zeiler, 2012; Kingma and Ba, 2015). The director process X determines the direction X_t at time t of the update Equation 2.1.11 with Z_t as reference point, while γ_t specifies a certain distance to travel along that direction X_t.

As represented in Figure 6.1, we can think of Z_t as the value of the process at time t, while $-\gamma_t X_t$ is the vector going from Z_t to Z_{t+1}. It is important to remember this since we are constantly referring to Z_t as points in $\mathbb{R}^k$ while X_t are managed as direction vectors in $\mathbb{R}^k$. This distinction is only practical for our purposes.

Stochastic processes in ML, such as SGD are usually expressed by means of their decomposition of 1-increments, as can be seen in Chapter 2. Another example is the one treated in (Sunehag et al., 2009) described next.

Example 6.1.1. In (Sunehag et al., 2009), the estimation update of the minimum $\overline{\eta}$ is defined

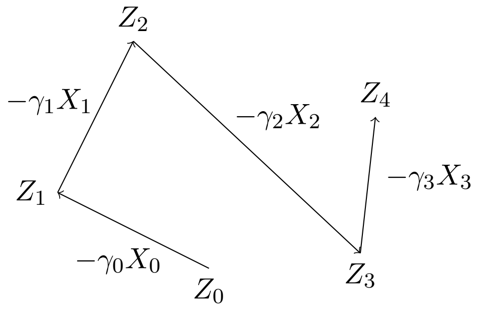


Figure 6.1: Path of stochastic process $Z = (X, \gamma)$

as

(6.1)
$$\begin{aligned} Z_{t+1} =& Z_t - \gamma_t B_t \cdot Y_t \\ \gamma_t >& 0 \quad t \in \mathbb{N} \end{aligned}$$

where B_t is a matrix in $\mathbb{R}^{k\times k}$ known after information $Z_0, ..., Z_t$ available at time t and $Y_t = Y(Z_t)$ where Y is a function mapping each $\eta \in \mathbb{R}^k$ to a random variable on the same probability space $(\Omega^*, \mathcal{F}^*, P^*)$.

Y can be thought as a random variable in the product probability space (Equation 2.36) that depends on previous Z_t, such that for every $\omega \in \Omega$ it is $Y_t(\omega) = Y(Z_t; \omega) = Y(Z_t; \omega_t)$. So if $X_t = B_t \cdot Y(Z_t)$, then $Z = (X, \gamma)$ is a decomposition of 1-increments of Z with $X = \{X_t\}_{t\in\mathbb{N}}$.

6.1.1 Locally bounded stochastic process

We demand some constraints to both factors of $Z = (X, \gamma)$.

- Condition imposed to γ is the **learning rate constraint** of Equation 2.38 usually found in the literature (Bottou, 1998; Sunehag et al., 2009; Sánchez-López and Cerquides, 2019).

- X is **locally and linearly bounded** by $\phi : \mathbb{R}^k \to \mathbb{R}$ if

  (6.2)
  $$(\exists A, B)(\forall t)\ \mathbb{E}_t \|X_t\|^2 \leq A + B \cdot \phi(Z_t)$$

These two constraints are finally combined to present the kind of stochastic processes we are interested in.

Definition 6.1.1. *Let Z be a stochastic process and $\phi : \mathbb{R}^k \to \mathbb{R}$ be a function. We say that Z is* ***locally bounded by*** *ϕ if there is a decomposition of 1-increments (X, γ) with γ holding the* ***learning rate constraint*** *and X* ***locally and linearly bounded*** *by ϕ.*

Furthermore, if $Z_0 = \eta_0$ a.s. we say η_0 is the initial point of Z.

Further on, Z is assumed to be locally bounded by ϕ where (X, γ) is its corresponding decomposition of 1-increments, unless otherwise indicated. Examples 6.1.2 and 6.1.3 show that we can understand the results in (Bottou, 1998; Sunehag et al., 2009) as the almost sure convergence of some locally bounded stochastic processes. In this thesis, we are interested in proving the almost sure convergence of a wider set of locally bounded stochastic processes.

Example 6.1.2. In reference (Bottou, 1998), the optimization algorithm is asked to hold additional conditions in order to prove its convergence. The reader can recall the convergence theorem by going to Section 2.7. Some of the conditions are

$$\begin{gathered}\sum_t \gamma_t^2 < \infty,\ \sum_t \gamma_t = \infty \\ Z_0 = \eta_0 \in \mathbb{R}^k \\ (\exists A, B)(\forall t)\ \mathbb{E}_t\|X_t\|^2 \leq A + B\|Z_t - \overline{\eta}\|^2\ , \end{gathered} \tag{6.3}$$

where $\overline{\eta} \in \mathbb{R}^k$ is the optimal point of L. **Learning rate constraint** is clearly asked. Moreover, η_0 is a starting point. It remains to see if X is **locally and linearly bounded** by some function $\phi : \mathbb{R}^k \to \mathbb{R}$. Indeed, define $\phi(\eta) = \|\eta - \overline{\eta}\|^2$, then the property is easily checked. Hence Z is **locally bounded** by ϕ with initial point η_0

Example 6.1.3. Recall Example 6.1.1. Convergence theorem in (Sunehag et al., 2009), which is added in Section 2.7, demands following conditions;

$$\begin{gathered}\sum_t \gamma_t^2 < \infty,\ \sum_t \gamma_t = \infty \\ Z_0 = \eta_0 \in \mathbb{R}^k \\ (\exists A, B)(\forall t)\ \mathbb{E}_t\|X_t\|^2 \leq A + BL(Z_t)\ , \end{gathered} \tag{6.4}$$

where L is a function to optimize. For this example, Z is **locally bounded** by ϕ with $\phi = L$.

6.2 Main result

The objective of this chapter is to prove Theorem 6.2.1, which will be proven in Section 6.5.1.

Theorem 6.2.1. *Let Z be a stochastic process on probability space $(\Omega, \mathcal{F}, P)$ to $\mathbb{R}^k$. Then Z almost surely converges to a point $\overline{\eta} \in \mathbb{R}^k$ if there is a twice differentiable convex function ϕ defined in $\mathbb{R}^k$ with unique minimum $\overline{\eta}$ and bounded Hessian norm, such that*

- *Z is locally bounded by ϕ*
- *Z resembles $\nabla\phi$.*

There is a concept of the theorem that needs a definition. That is when a stochastic process *resembles* a vector field. Sections 6.3 and 6.4.2 fill this gap.

6.3 Expected direction set

We now define one key mathematical object of the chapter named the expected direction set. It focuses on gathering all directions that the update may take at time t conditioned to $\mathcal{F}_t$. Before the definition, we provide some concepts and notation. Recall that the notation related to conditional expectation $\mathbb{E}_t$ is given in Section 2.1.2.

Random variable $\mathbb{E}_t[X_t]$ determines all expected directions of $Z = (X, \gamma)$ at time t that the stochastic process may follow assuming $\mathcal{F}_t$. For example, if $\omega \in \Omega$ is an observation, then $\mathbb{E}_t[X_t](\omega) \in \mathbb{R}^k$ is a vector pointing to the expected update direction departing from point $Z_t(\omega)$ given $\mathcal{F}_t$. Denote the expected direction of Z at $\omega \in \Omega$ and time t as

$$D_Z(\omega, t) = \mathbb{E}_t[X_t](\omega)\,. \tag{6.5}$$

The expected direction from point $\eta = Z_t(\omega)$ of Equation 6.5 depends on ω. That is, the path followed until reaching $\eta = Z_t(\omega) \in \mathbb{R}^k$ matters. For instance, if $\omega_1, \omega_2 \in \Omega$ are different observations such that $\eta = Z_t(\omega_1) = Z_t(\omega_2)$, then possibly $D_Z(\omega_1, t) \neq D_Z(\omega_2, t)$. We collect all expected directions at $\eta = Z_t(\omega)$ and time t in the vector set

$$S_Z(\eta, t) = \{D_Z(\omega, t) \mid \omega \in \Omega, Z_t(\omega) = \eta\}\,. \tag{6.6}$$

The tools to define the expected direction set at $\eta \in \mathbb{R}^k$ after time $T \in \mathbb{N}$ are given, so we proceed to its formal definition.

Definition 6.3.1. *Let $Z = (X, \gamma)$. Define the expected directions set of Z at $\eta \in \mathbb{R}^k$ after time $T \in \mathbb{N}$ as*

$$EDS_Z(\eta, T) := \bigcup_{t \geq T} S_Z(\eta, t) \ . \tag{6.7}$$

With a few words, $EDS_Z(\eta, T)$ is a vector set containing all expected directions (provided by the director process X) conditioned to $\mathcal{F}_t$ for every outcome ω such that $Z_t(\omega) = \eta$ where $t \geq T$. In definition 6.3.1, EDS depends on T. That is because to assess the convergence of an algorithm it is not important to consider all expected directions throughout the process. For example, if an algorithm converges we can modify randomly all directions of the director process for just a particular time $T \in \mathbb{N}$, and the resulting algorithm still converges. Roughly speaking, only the *tail* of a process matters to determine the convergence property. This concept is better addressed with definition 6.3.2 in the next section.

Example 6.3.1. Assume that Z is SGD. Then, $EDS_Z(\eta, T)$ is a singleton. Indeed, $D_Z(\omega, t) = \nabla f(\eta)$ is the same vector for all $t \in \mathbb{N}$ and all ω with $Z_t(\omega) = \eta$ and hence $S_Z(\eta, t) = \{D_Z(\omega, t)\}$ with any $\omega \in \Omega$ with $Z_t(\omega) = \eta$. Finally

$$EDS_Z(\eta, T) = \{\nabla f(\eta)\} \ . \tag{6.8}$$

6.3.1 Essential expected direction set

The convergence property of an algorithm relates closely to directions followed after time $T \in \mathbb{N}$ as T tends to infinity. Equivalently, directions appearing repeatedly through the whole optimization process matter, while directions only contemplated for a finite amount of iterations change nothing, in terms of convergence guarantee. The direction set containing only directions appearing repeatedly through the whole optimization process is named the essential expected directions set in this chapter.

To define properly the essential expected directions set, we will use the convex vector subspace of a given vector set. Given a vector set U in $\mathbb{R}^k$, let $C(U)$ be the smallest convex vector subspace containing U. See Figure 6.2 as an illustrative example.

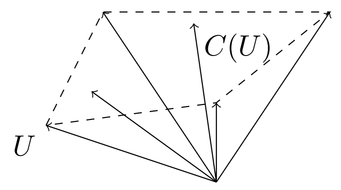


Figure 6.2: Set of vectors U and its convex vector subspace $C(U)$ in $\mathbb{R}^2$

Observe that $C(U)$ is always closed, but it is clearly unbounded when U contains vectors arbitrarily large. Next, we define the essential expected direction set, which may help to identify divergence symptoms of a stochastic process.

Definition 6.3.2. *Let $Z = (X, \gamma)$. Define the essential expected directions set of Z at η as*

$$EEDS_Z(\eta) := \cap_T C(EDS_Z(\eta, T)) \; . \tag{6.9}$$

Definition of $EEDS_Z(\eta)$ delimits the smallest subspace where all directions at η tend to. Clearly, $EEDS_Z(\eta)$ is also convex and closed (possibly empty). Deeper properties of this set lead to identifying divergence symptoms. For example, if it is empty or unbounded at η, we face instability of the process at η. To see this, observe the next result. The proof can be found in Appendix C.1.

Corollary 6.3.1. *Let $\eta \in \mathbb{R}^k$. Then $EEDS_Z(\eta)$ is a non empty bounded set if and only if there exists $T \in \mathbb{N}$ such that $C(EDS_Z(\eta, T))$ is bounded.*

This result is a corollary because derives from Proposition C.1.1 proven in the Appendix C.1. Corollary 6.3.1 relates $EEDS_Z(\eta)$ with instability properties of Z. If $EEDS_Z(\eta)$ is empty or unbounded, then $C(EDS_Z(\eta, T))$ is unbounded and the algorithm is unstable at η, since expected directions with arbitrarily large norms exist after enough iterations. Clearly, if this situation is found for all points near the optimum, the algorithm can not converge to the solution. It is desirable instead that $C(EDS_Z(\eta, T))$ is compact (bounded) for some T for every $\eta \in \mathbb{R}^k$, or equivalently, that $EEDS_Z(\eta)$ is compact (bounded) and not empty.

In fact, since we are interested in the case where Z is locally bounded by ϕ (recall definition 6.1.1), we can assume that $EEDS_Z(\eta)$ is a nonempty compact set, by virtue of Proposition 6.3.1.

Proposition 6.3.1. *Let stochastic process Z be locally bounded by ϕ. Then $C(EDS_Z(\eta, 0))$ is a non empty compact set.*

Proof. We know that X is locally and linearly bounded. Hence, applying Jensen's inequality

$$\|\mathbb{E}_t[X_t]\|^2 \leq \mathbb{E}_t\|X_t\|^2 \leq A + B \cdot \phi(Z_t) \tag{6.10}$$

Let $\eta \in \mathbb{R}^k$ and $\omega \in \Omega$ such that $Z_t(\omega) = \eta$ for some $t \geq 0$. Therefore, every $v = \mathbb{E}_t[X_t](w) \in EDS_Z(\eta, 0)$ has bounded norm by $A + B \cdot \phi(\eta)$, implying that $C(EDS_Z(\eta, 0))$ is a non empty compact set. □

Corollary 6.3.2 is a consequence of Proposition 6.3.1 and Corollary 6.3.1.

Corollary 6.3.2. *Let stochastic process Z be locally bounded by ϕ. Then $EEDS_Z(\eta)$ is a non empty compact set for all $\eta \in \mathbb{R}^k$.*

Example 6.3.2. Assume that for every η and T, the expected direction set of Z contains only one vector, such as SGD. Then $EEDS_Z(\eta) = EDS_Z(\eta, T)$ for any T. Indeed, in the case of SGD, we have seen in Example 6.3.1 that $EDS_Z(\eta, T) = \{\nabla f(\eta)\}$. Hence

$$EEDS_Z(\eta) = \cap_T C(EDS_Z(\eta, T)) = C(\{\nabla f(\eta)\}) = \{\nabla f(\eta)\} = EDS_Z(\eta, T) \tag{6.11}$$

6.4 Vector field half-spaces and stochastic processes. Resemblance.

This section defines the main concept of this chapter; the property of *resemblance* between a stochastic process and a vector field. The definition highlight some commonalities between theorems 2.7.1 and 2.7.2. Both of them prove the convergence of stochastic processes that *resemble* to particular vector fields. A geometric interpretation and explanation of convergence theorems conditions is later established in Section 6.5.

Some previous definitions are needed and stated before introducing the main concepts of the chapter, such as ϵ-acute vector pair sets and the half-space of a vector field. The section starts with some basic concepts about vectors.

Definition 6.4.1. *Let $u, v \in \mathbb{R}^k$ be two vectors. The pair (u, v) is acute if u and v form an acute angle, that is, if $u^\intercal \cdot v > 0$. Furthermore, if $u^\intercal \cdot v \geq \epsilon > 0$ then (u, v) is ϵ-acute.*

Proposition 6.4.1. *Let $u, v \in \mathbb{R}^k$ be two vectors. Then, the pair (u, v) is ϵ-acute if and only if there exists a symmetric positive-definite matrix B such that $B \cdot u = v$ and $u^\intercal \cdot B \cdot u \geq \epsilon$.*

A vector pair set V is a set of vector pairs $V = \{(u_i, v_i) \in \left(\mathbb{R}^k\right)^2 \mid i \in I\}$ where I is an index set.

Definition 6.4.2. *Let V be a vector pair set. V is ϵ-acute if every vector pair $(u, v) \in V$ is ϵ-acute.*

Vector pair sets holding Definition 6.4.2 can be understood as sets whose vector pairs form an angle of at most $\frac{\pi}{2}$. The larger ϵ is in the definition, the bigger the inner product is of all vector pairs. The next result is a direct consequence of the definition of the ϵ-acute vector pair set.

Proposition 6.4.2. *Let V be a vector pair set, indexed by I. Then, V is ϵ-acute for some $\epsilon > 0$ if and only if;*

$$\inf_{\substack{i \in I \\ (u_i, v_i) \in V}} \quad u_i^\intercal v_i > 0 \tag{6.12}$$

Proposition 6.4.3. *Let V be a vector pair set, indexed by I. Then, V is ϵ-acute for some $\epsilon > 0$ if and only if there exist a set of symmetric positive-definite matrices $B = \{B_i \mid i \in I\}$ such that*

$$\begin{gathered} \inf_{\substack{i \in I \\ (u_i, v_i) \in V}} \quad u_i^\intercal B_i u_i > 0 \\ B_i u_i = v_i \end{gathered} \tag{6.13}$$

Proof. Prove first that if there exist a set of matrices $B = \{B_i \mid i \in I\}$ holding equation 6.13 then V is ϵ-acute for some $\epsilon > 0$. Observe that after equation 6.13;

$$\inf_{i \in I} u_i^\intercal v_i = \inf_{i \in I} u_i^\intercal B_i u_i > 0 \tag{6.14}$$

Then, proposition 6.4.2 implies that V is ϵ-acute and finishes this part of the proof.

Now assume that V is ϵ-acute, prove then that there exist a set of matrices $B = \{B_i \mid i \in I\}$ holding equation 6.13. Since V is ϵ-acute, in particular, the pair $(u_i, v_i) \in V$ is ϵ-acute for every $i \in I$. Apply proposition 6.4.1: for every $i \in I$ there exists a symmetric positive-definite matrix B_i such that $B_i u_i = v_i$ and $u_i^\intercal \cdot B_i \cdot u_i \geq \epsilon$. This finishes the proof. □

Proposition 6.4.2 and Proposition 6.4.3 provide different properties that equivalently identify ϵ-acute vector pair sets.

6.4.1 The half-space of a vector field

The half-space determined by a vector u is the set of vectors that conform to an acute angle with u. This region clearly occupies half of the total space. Also, the ϵ-half-space of u with $\epsilon > 0$ is the set of vectors v such that the vector pair (u, v) is ϵ-acute. This object is needed for afterward defining the half-space of a vector field. We first define these concepts and then illustrate the ϵ-half-space of a vector u in Figure 6.3.

Definition 6.4.3. *Let u be a vector of $\mathbb{R}^k$. The half-space of u is the set*

$$H(u) = \{v \in \mathbb{R}^k \mid u^\mathsf{T} \cdot v > 0\} \tag{6.15}$$

Similarly, the ϵ-half-space of u with $\epsilon > 0$ is the set

$$H_\epsilon(u) = \{v \in \mathbb{R}^k \mid u^\mathsf{T} \cdot v \geq \epsilon\} \tag{6.16}$$

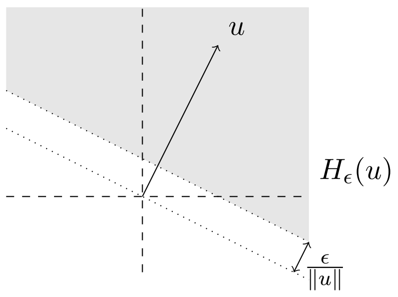


Figure 6.3: Shaded area representing $H_\epsilon(u)$

A vector field $\mathbb{X}$ over $\mathbb{R}^k$ is a function assigning to every $\eta \in \mathbb{R}^k$ a vector of $\mathbb{R}^k$, that is $\mathbb{X} : \mathbb{R}^k \to \mathbb{R}^k$. For example, if $f : \mathbb{R}^k \to \mathbb{R}$ is a differentiable function, we can consider the vector field consisting of the gradient vectors at each point η. Precisely, denote the gradient vector field (GVF) as $\mathbb{X}_{\nabla f}$, where $\mathbb{X}_{\nabla f}(\eta) = \nabla f(\eta)$.

We are ready to define the half-space of a vector field.

Definition 6.4.4. *Let $\mathbb{X}$ be a vector field over $\mathbb{R}^k$. The half-space of $\mathbb{X}$ is a function $H(\mathbb{X})$ mapping every η to $H(\mathbb{X})(\eta) = H(\mathbb{X}(\eta))$. Similarly, the ϵ-half-space of $\mathbb{X}$ with $\epsilon > 0$ is a function $H_\epsilon(\mathbb{X})$ mapping every η to $H_\epsilon(\mathbb{X})(\eta) = H_\epsilon(\mathbb{X}(\eta))$.*

6.4.2 Resemblance between a stochastic process and a vector field

The convergence of any locally bounded process can be proved by comparing the expected directions set of the algorithm with some vector fields. When the expected directions *resemble* to the vector field we compare to, then we can ensure the almost sure convergence to a point of the stochastic process, after some reasonable conditions. By *resemblance*, we mean that the expected directions set after some time T is a subset of the ϵ-half-space of $\mathbb{X}$, among other things explained later. Therefore, *resemblance* asks for every $\eta \in \mathbb{R}^k$ that every vector $D_Z(\omega, t)$ with $t \geq T$ and every $\omega \in \Omega$ with $\eta = Z_t(\omega)$ form an acute angle with the vector field at $\mathbb{X}(\eta)$.

However, if the vector field sends a specific point η to $0 \in \mathbb{R}^k$, then no direction can be set by the $D_Z(\omega, t)$ to form an acute vector pair. Therefore *resemblance* property is evaluated outside the neighborhood of these vanished points. That's why we must consider now the set of vanished points of a vector field and the neighborhoods around the points of this set.

Formally, let $\mathbb{X}$ be a vector field defined in $\mathbb{R}^k$. The set $K_{\mathbb{X}}$ is the set of points of $\mathbb{R}^k$ that vanish by $\mathbb{X}$, that is, $K_{\mathbb{X}} := \{\eta \in \mathbb{R}^k \mid \mathbb{X}(\eta) = 0\}$. Moreover, consider the closed ball centered on $K_{\mathbb{X}}$ of radius δ as $B_\delta(K_{\mathbb{X}}) := \cup_{\eta \in K_{\mathbb{X}}} B_\delta(\eta)$ where $B_\delta(\eta)$ is the closed ball of radius δ centered on η.

We also use the notation $A' = \mathbb{R}^k \setminus A$ for the complement set of subset $A \subset \mathbb{R}^k$. We say that Z *ϵ-resembles* to $\mathbb{X}$ at η from T on if $EDS_Z(\eta, T) \subset H_\epsilon(\mathbb{X})(\eta)$. Observe an illustrative example in Figure 6.4.

This intuition is naturally extended to *ϵ-resemblance* at sets when the property is satisfied for every η in the set. With this in mind, we can define the key concept of this chapter.

Definition 6.4.5. *Let $Z = (X, \gamma)$ be a stochastic process and $\mathbb{X}$ be a vector field over $\mathbb{R}^k$. We say that Z resembles to $\mathbb{X}$ from $T \in \mathbb{N}$ on, if;*

$$(\forall \delta > 0)(\exists \epsilon > 0)\ Z\ \epsilon\text{-resembles to } \mathbb{X} \text{ at } B_\delta(K_{\mathbb{X}})' \text{ from } T \text{ on}\ . \tag{6.17}$$

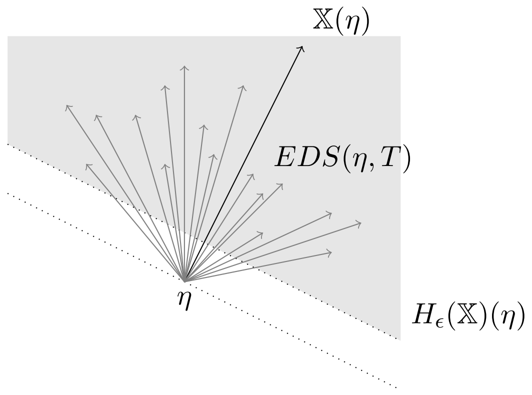


Figure 6.4: A stochastic process Z that ϵ-resembles to $\mathbb{X}$ at η from T on, since vector set $EDS_Z(\eta, T)$ of all expected directions of Z at η after time T belongs to $H_\epsilon(\mathbb{X})(\eta)$

We say that Z resembles to $\mathbb{X}$ if there is $T \in \mathbb{N}$ such that it resembles to $\mathbb{X}$ from T on.

Everything is set up to accomplish the goal of this chapter, which is proving the main theorem.

6.5 Proof of main result

The objective of the chapter is within reach now. That is, proving the main Theorem 6.2.1.

6.5.1 Resemblance to conservative vector fields and convergence

Recall main Theorem 6.2.1 and observe that it asks the stochastic process Z to be locally bounded by some function ϕ and Z to resemble $\nabla\phi$. Therefore, $\nabla\phi$ is a particular type of vector field called a conservative vector field. That is a vector field that appears from the derivation of a function. That is why we understand our main theorem as a convergence result of locally bounded processes of resemblance to the conservative vector field.

In the theorem statement, it says that ϕ has a bounded Hessian norm. Similarly to Theorem 2.7.2, it means that ;

$$(\exists K)(\forall \eta)\ \|\nabla^2_\eta \phi(\eta)\| \leq K' \ .$$

We are ready to prove the main result of the chapter.

Proof of main Theorem 6.2.1. Observe that ϕ is bounded from below. Indeed, $\overline{\eta}$ is a minimum and ϕ is convex with $\mathbb{X}(\overline{\eta}) = 0$ where $\mathbb{X} = \nabla\phi$. Therefore there exist a constant $m \geq 0$ such that $\phi(\eta) + m \geq 0$ for all η. Define $\psi(\eta) = \phi(\eta) + m$. Clearly, $\nabla\psi = \nabla\phi = \mathbb{X}$, and therefore Z *resembles* to $\nabla\psi$. Moreover, Z is locally bounded by ψ and ψ clearly satisfies the **Hessian norm bound**.

From here, the proof follows the steps of Theorem 2.7.2's proof. Taylor inequality and **Hessian norm bound**;

$$\begin{aligned} \psi(Z_{t+1}) &= \psi(Z_t - \gamma_t X_t) \\ &\leq \psi(Z_t) - \gamma_t \mathbb{X}(Z_t)^\intercal X_t + \gamma_t^2 K \|X_t\|^2 \,, \end{aligned} \tag{6.18}$$

where $K = \frac{K'}{2}$. Apply expectation conditioned to information until time t and then use that Z is locally bounded by ψ;

$$\begin{aligned} \mathbb{E}_t\psi(Z_{t+1}) &\leq \psi(Z_t) - \gamma_t \mathbb{X}(Z_t)^\intercal \mathbb{E}_t\left[X_t\right] + \gamma_t^2 K E_t\left[\|X_t\|^2\right] \\ &\leq \psi(Z_t) - \gamma_t \mathbb{X}(Z_t)^\intercal \mathbb{E}_t\left[X_t\right] + \gamma_t^2 K(A + B\psi(Z_t)) \\ &\leq (1 + \gamma_t^2 KB)\psi(Z_t) - \gamma_t \mathbb{X}(Z_t)^\intercal \mathbb{E}_t\left[X_t\right] + \gamma_t^2 KA \,. \end{aligned} \tag{6.19}$$

Use now that Z *resembles* to $\mathbb{X}$. Then there exists T such that for every $t \geq T$, the term $-\gamma_t \mathbb{X}(Z_t)^\intercal \mathbb{E}_t\left[X_t\right]$ is negative. All other conditions of Sigmund-Robbins theorem (in (Robbins and Siegmund, 1971), added in Section 2.7) also hold for the algorithm after time T, thanks to **learning rate constraints**. Apply it and deduce that random variables $\psi(Z_t)$ converge almost surely to a random variable (and so does $\phi(Z_t)$) and that;

$$\sum_t \gamma_t \mathbb{X}(Z_t)^\intercal \mathbb{E}_t\left[X_t\right] < \infty \qquad \text{a.s}\,. \tag{6.20}$$

Prove now that stochastic process $\phi(Z_t)$ converges almost surely to value $\phi(\overline{\eta})$. Proceed by contradiction. Assume that for $\delta_1 > 0$

$$P\left[\omega \in \Omega \mid \lim_t \phi(Z_t(w)) \in B_{\delta_1}(\phi(\overline{\eta}))'\right] > 0 \,, \tag{6.21}$$

this implies, by continuity and convexity of function ϕ, that there exists δ

$$P\left[A = \{\omega \in \Omega \mid \lim_t Z_t(w) \in B_\delta(\overline{\eta})'\}\right] > 0 \,. \tag{6.22}$$

By *resemblance* and definition of the limit, there exists T and ϵ such that $EDS_Z(\eta, T) \subset H_\epsilon(\mathbb{X})(\eta)$ for every $\eta \in B_\delta(\overline{\eta})'$. This leads to a contradiction, since using learning rate standard constraint we have

$$\sum_{t \geq T} \gamma_t \mathbb{X}(Z_t(\omega))^\intercal \mathbb{E}_t \left[X_t\right](\omega) > \sum_{t \geq T} \gamma_t \cdot \epsilon = \infty \,, \tag{6.23}$$

for every $\omega \in A$, which has measure different to 0 by Equation 6.22. This clearly contradicts Equation 6.20.

Hence, $\phi(Z_t)$ converges almost surely to $\phi(\overline{\eta})$ and Z_t converges almost surely to $\overline{\eta}$ as wanted. □

6.6 Reinterpretation of convergence theorems

Moreover, this section addresses afterward the task of proving that theorems 2.7.1 and 2.7.2 are particular examples of our main Theorem 6.2.1.

6.6.1 Reinterpretation of Bottou's convergence theorem

The goal now is to deduce Theorem 2.7.1 as a direct consequence of the main Theorem 6.2.1. Consider a particular case of main Theorem 6.2.1 where $\phi(\eta) = \|\eta - \overline{\eta}\|^2$, that reads as follows.

Corollary 6.6.1. *Let $\phi(\eta) = \|\eta - \overline{\eta}\|^2$ and Z be a stochastic process on probability space $(\Omega, \mathcal{F}, P)$. Then Z almost surely converges to $\overline{\eta}$ if*

- *Z is locally bounded by ϕ*
- *Z resembles $\nabla\phi$.*

Additional conditions to ϕ, such as Hessian bound or twice differentiability, are not specified in the corollary since with the particular definition of ϕ all those conditions are already satisfied.

To see that Corollary 6.6.1 proves Theorem 2.7.1 statement, we need to prove that Theorem 2.7.1 is assuming that Z is locally bounded by ϕ and that Z resembles $\nabla\phi$. Example 6.1.2 already proves that Bottou is assuming that Z is locally bounded by ϕ. Therefore it remains to check that Z resembles $\nabla\phi$. To that end, see Proposition 6.6.1 proved in Appendix C.2.

Proposition 6.6.1. *Let $Z = (X, \gamma)$ be a stochastic process and $\mathbb{X}$ be a vector field over $\mathbb{R}^k$. Then Z resembles to $\mathbb{X}$ if and only if*

$$(\exists T \in \mathbb{N})(\forall \delta > 0) \inf_{\substack{\eta \in \mathbb{R}^k \setminus B_\delta(K_{\mathbb{X}}) \\ v \in EDS_Z(\eta, T)}} \mathbb{X}(\eta)^\intercal \cdot v > 0 \,. \tag{6.24}$$

By taking a look at condition **Bottou resemblance** of Theorem 2.7.1 and Proposition 6.6.1, we can deduce from it, that the algorithm Z of the theorem *resembles* to vector field $\nabla\phi$.

Corollary 6.6.2. *Let $Z = (X, \gamma)$ be a stochastic process and $\overline{\eta} \in \mathbb{R}^k$. Then Z resembles to $\nabla\phi$ with $\phi(\eta) = \|\eta - \overline{\eta}\|^2$ if and only if* ***Bottou resemblance*** *holds.*

6.6.2 Reinterpretation of Sunehag's convergence theorem

Theorem 2.7.2 is a bit too restrictive, so our main theorem can not directly imply such a result. Hence, this section starts with a generalization of Theorem 2.7.2.

Theorem 6.6.1 (Generalization of Theorem 2.7.2)**.** *Let $f : \mathbb{R}^k \to \mathbb{R}$ be a twice differentiable cost function with a unique minimum $\overline{\eta}$ and let $Z_{t+1} = Z_t - \gamma_t B_t Y_t$ a stochastic process where B_t is defined after information available at time t. Then Z converges to the $\overline{\eta}$ almost surely if the following conditions hold;*

$$\textbf{\textit{C.1}}\ (\forall t)\ \mathbb{E}_t Y_t = \nabla f(Z_t) \qquad \eta_t \neq \overline{\eta}$$

$$\textbf{\textit{Hessian bound}}\ (\exists K)(\forall \eta)\ \|\nabla^2_\eta f(\eta)\| \leq 2K$$

$$\textbf{\textit{Sunehag resembance}}\ (\forall \delta > 0) \inf_{f(Z_t) - f(\overline{\eta}) > \delta} \nabla f(Z_t)^\intercal B_t \nabla f(Z_t) > 0$$

$$\textbf{\textit{Sunehag algorithm bound}}\ (\exists A, B)(\forall t)\ \mathbb{E}\|B_t Y_t\|^2 \leq A + Bl(Z_t)$$

Learning rate constraint

Theorem 6.6.1 is deduced from main Theorem 6.2.1. Similarly to the previous section, we provide a version of our main theorem for the case where $\phi = f$ is a function that we aim to minimize.

Corollary 6.6.3. *Let $f : \mathbb{R}^k \to \mathbb{R}$ be a twice differentiable cost function with a unique minimum $\overline{\eta}$ and bounded Hessian norm, and let Z be a stochastic process on probability space $(\Omega, \mathcal{F}, P)$. Then Z converges to the minimum $\overline{\eta}$ of f almost surely if*

- *Z is locally bounded by f*

- *Z resembles ∇f.*

The stochastic process described in Theorem 6.6.1 has some more properties, such as $X_t = B_t \cdot Y_t$. But if we prove that Z of that theorem is locally bounded by f and that Z *resembles* ∇f, then it is clear that Corollary 6.6.3 implies Theorem 6.6.1. Recall Example 6.1.3, where we already proved that Z is locally bounded by f. The remaining property is acquired after Proposition 6.6.2 that we prove in Appendix C.3.

Proposition 6.6.2. *Let $Z = (X, \gamma)$ be a stochastic process and $\mathbb{X}$ be a vector field over $\mathbb{R}^k$. Then Z resembles to $\mathbb{X}$ if and only if there exists T such that for every $t \geq T$ there are random vectors Y_t to $\mathbb{R}^k$ and symmetric and positive-definite random matrices B_t defined after information available at time t such that*

$$B_t \cdot Y_t = X_t \;, \tag{6.25}$$

$$\mathbb{E}_t[Y_t] = \mathbb{X}(Z_t) \qquad Z_t(\omega) \notin K_{\mathbb{X}} \;, \tag{6.26}$$

$$(\forall \delta > 0) \quad \inf_{\substack{\eta \in \mathbb{R}^k \setminus B_\delta(K_{\mathbb{X}}) \\ t \geq T \\ \omega \in \Omega, Z_t(\omega) = \eta}} \mathbb{X}(\eta)^\intercal \cdot B_t(\omega) \cdot \mathbb{X}(\eta) > 0 \;. \tag{6.27}$$

It is only necessary to put together Proposition 6.6.2 and condition **C.1** and **Sunehag resemblance** to finish our objective with Corollary 6.6.4.

Corollary 6.6.4. *Let f be a differentiable function and $Z = (X, \gamma)$ be a stochastic process. Then Z resembles to ∇f if and only if there exist T such that for every $t \geq T$ there are random vectors Y_t to $\mathbb{R}^k$ and symmetric and positive-definite random matrices B_t defined after information available at time t such that $B_t \cdot Y_t = X_t$ and conditions **C.1** and **Sunehag resemblance** hold.*

Corollaries 6.6.2 and 6.6.4 nicely show the value of Theorem 6.2.1 for proving convergence. To reinforce this, we notice that the convergence of algorithm DSNGD in Chapter 5 is easily proved by means of Corollary 6.6.3, by combining both Theorem 6.6.1 and Corollary 6.6.4. This shows that Theorem 6.2.1 allows proving convergence of a wider set of stochastic processes and function optimization methods.

6.7 Comments

We have presented a result that allows us to prove the convergence of some stochastic processes. We have proven that two useful convergence results in the literature are a consequence of our theorem. This is made after introducing a new theory that compares the expected directions of the algorithm to conservative vector fields. If the expected directions at a point η *resemble* enough to vector $\mathbb{X}(\eta)$ with $\nabla\phi = \mathbb{X}$ a conservative vector field, then the process is stable at that point. If this happens for every $\eta \in \mathbb{R}^k$, and in addition, the process is locally bounded by ϕ, then the process is globally stable and converges.

Some inspiring paths remain unexplored after this chapter. For example, finding the ϕ function is the key to proving convergence, and it is asked to be a convex twice differentiable function. It is interesting to study how ϕ can be obtained, for instance as a sum of other convex twice differentiable functions ϕ_i.

Another promising research line is a deeper analysis of EDS and $EEDS$ objects, which may guarantee the existence of a function ϕ without the need of finding it. If sufficient conditions are established for a stochastic process to ensure *resemblance* to some unknown conservative vector field, then ϕ searching can be dodged. Even proving the non-existence of such function after a wider study of EDS and $EEDS$ is useful, forbidding the use of our theorem.

It is also interesting to study the reverse implication. Specifically, investigating the conditions that lead to divergent instances based on the theory explained in this chapter. In this sense, Lyapunov characterization of convergent processes becomes a helpful and key theory, since great similarities arise between these two techniques.

Furthermore, on many occasions, the function ϕ to optimize can be established beforehand (convex and twice differentiable). Therefore the opposite process can be considered, that is, generating a set of stochastic processes that *resemble* to $\nabla\phi$, assuring in consequence the convergence of such candidates.

In (Robbins and Monro, 1951), one finds another relevant convergence result. It assures the convergence in probability of a stochastic process, instead of the almost sure convergence used in this chapter. We wonder about the existing commonalities with our theorem, and the possibility to relax the conditions our theorem imposes while ensuring convergence in probability of a process.

We are currently working on two weaker *resemblance* properties, that we name *weak* and *essential resemblance*. The intention is to deduce almost sure convergence of a process by only studying its essential expected direction set (EEDS).

7 Conclusion

This thesis tackles the problem of bringing the natural gradient to the function optimization task efficiently. To that end, we faced the convergence issue and the high computational complexity usually linked to natural gradient based algorithms.

We started proposing the MOD algorithm, which allowed us to support our hypothesis, namely: *if SNGD is stabilized then high convergence speed is at hand*. However, we could not find a proof of convergence for MOD. Nonetheless, we got encouraged to design CSNGD, a natural gradient algorithm close to SNGD whose convergence is proven. Automatically, experiments in a toy problem, that we refer to as the die problem in the thesis, reflected really promising results. CSNGD even competes with MLE, which is arguably the optimum estimator for this toy example. This contribution also supports our hypothesis: CSNGD has a convergence proof which stabilizes in theory its learning process, and we collect fast convergence speed in turn. Nevertheless, the computational complexity of CSNGD is higher than that of standard SGD. Optimization problems in ML nowadays have usually a high number of variables, and the samples drawn for the learning process are often huge. Hence, fast algorithms are the only option to face many actual problems, and slower algorithms that scale badly with manifold dimension, such as CSNGD, are directly discarded as suitable optimization methods.

In order to design fast and convergent natural gradient based algorithms, we found it necessary to delimit the problem we target. We had to restrict to the MLR problem in the joint distribution manifold, where we proved that it is possible to compute the natural gradient of the empirical loss function really fast. Applying the knowledge acquired when we designed CSNGD, we defined a natural gradient based method named DSNGD which is also convergent, although its convergence proof demanded a generalization of existing convergence results in the literature. Furthermore, our new algorithm is of linear order in the discrete case (when the feature variables are discrete). This is why we could run high-dimensional experiments with large samples. We observed how DSNGD outperforms standard SGD in almost every scenario, except for the scenario where SGD was already obtaining really low orders of error in its estimations. Hence, DSNGD seems to avoid the low convergence ratio in those scenarios in which SGD does not. Again, our hypothesis is supported, by relating our convergent natural gradient based algorithms with learning processes with high convergence speed. In addition, DSNGD answers positively our research main question in this thesis: the natural gradient can be effectively exploited for the function optimization task in ML, although our algorithm can be only employed for the MLR problem.

Finally, the proof of convergence of DSNGD leads us to unify some convergence theorems in the literature, by providing a general convergence result. Our convergence result is used to prove the convergence of DSNGD and it also presents apparently distant convergence theorems in (Bottou, 1998) and (Sunehag et al., 2009) as particular cases of our main theorem.

This thesis leaves unexplored many research paths. The most tempting ones are probably concerning algorithm DSNGD. In this thesis, we focus on the theoretical aspects of DSNGD. We are currently working on a flexible implementation of the algorithm that can be easily set up for different LEF linked to the conditional distributions $P(\mathcal{X} \mid \mathcal{Y})$, including several commonly used continuous distributions such as the normal, Poisson and exponential, and also discrete and mixed distributions attained to naive Bayes conditional independence assumption. For the continuous case, the constraint issue must be addressed accordingly. Moreover, DSNGD can potentially be used in high dimensional scenarios due to its low computational complexity. The benefits of approximating the natural gradient are especially promising in this case since the parameter space is potentially twisted and using metric information can be crucial for an algorithm's good performance. In preliminary empirical studies, we are observing how it increasingly outperforms SGD as the manifold dimension grows larger. We plan to compare DSNGD against the most effective algorithms nowadays, in order to expose its weaknesses and reveal its strengths.

Research to expand DSNGD to nonlinear exponential families remains open. The algorithm strongly relies on two dual parametrizations, which may complicate the task. Moreover, the approximation of the natural gradient performed by DSNGD is accurate only when both sequences Z_t and ζ_t^* refer to the same point. It remains open whether this requirement can be appropriately fulfilled in other problems.

For the more theoretical part, in the future we plan to study the convergence of continuous and mixed DSNGD, that is when $\mathcal{X}$ is continuous, and in cases where $\mathcal{X} = (\mathcal{X}_d, \mathcal{X}_c)$ is divided into a discrete and a continuous part.

More questions appear at the end of our research. Our main convergence theorem can lead to defining new natural gradient based algorithms for different ML optimization problems out of MLR. This means that maybe fast convergence speed can be obtained by natural gradient based algorithms for more optimization problems. However, this thesis provides no clue whether natural gradient can be computed fast in other manifolds or problems outside of the MLR problem.

The theory developed in this thesis related to our main convergence theorem shows some similarities with the theory of Lyapunov functions and convergence (as in (Bottou, 1998)). In future work, we also want to figure out exactly what are the differences and similarities between these two theories. Besides, Lyapunov's theory may answer what conditions of a stochastic process characterize the almost sure convergence of the process.

A Appendix: Natural gradient

A.1 Proof of theorem 2.3.1

Proof. Find the steepest vector $\widetilde{v}$ of T_pM, that maximizes;

$$\nabla f(p)^T \cdot \frac{v}{\|v\|_{G_p}}, \qquad v \in T_pM \tag{A.1}$$

where $\| \cdot \|_{G_p}$ is the norm function at T_pM according to the metric G_p. which can be rewritten as;

$$\frac{\nabla f(p)^T \cdot v}{(< v, v >_{G_p})^{\frac{1}{2}}} \qquad v \in T_pM \tag{A.2}$$

where $<, >_{G_p}$ is the scalar product of vectors at T_pM considering the metric. Recall that G and G^{-1} are in particular symmetric invertible matrices. Furthermore, G^{-1} can be seen as an automorphism in the vector space T_pM. So equivalently, find the steepest vector $\widetilde{u}$ of T_pM, that maximizes;

$$\frac{\nabla f(p)^T G^{-1} u}{(< G^{-1}u, G^{-1}u >_{G_p})^{\frac{1}{2}}} \tag{A.3}$$

and recover the solution to the original problem by doing $\widetilde{v} = G^{-1}\widetilde{u}$. The previous equation equals to

$$\frac{\nabla f(p)^T G^{-1} u}{((G^{-1}u)^T G (G^{-1}u))^{\frac{1}{2}}} = \frac{\nabla f(p)^T G^{-1} u}{(u^T (G^{-1})^T u)^{\frac{1}{2}}} \tag{A.4}$$

Recall that G^{-1} is symmetric and $(G^{-1})^T = G^{-1}$. So $\widetilde{u}$ maximizes

$$\frac{\nabla f(p)^T G^{-1} u}{(u^T G^{-1} u)^{\frac{1}{2}}} =< \nabla f(p), \frac{u}{\|u\|_{G_p^{-1}}} >_{G_p^{-1}} \tag{A.5}$$

By definition of inner product, the solution is $\widetilde{u} = \lambda \nabla f(p)$ with $\lambda \in \mathbb{R}$, which finally implies that $\widetilde{v} = G^{-1}\widetilde{u} = \lambda G^{-1} \nabla f(p)$ as wanted. $\square$

B Appendix: Dual stochastic natural gradient descent

B.1 Proof of Proposition 5.1.1

Proof. Prove first that if the logg-odds ratio of $P(\mathcal{Y} \mid \mathcal{X})$ is an affine function of $\mathcal{X}$ then the joint distribution $P(\mathcal{Y}, \mathcal{X})$ belongs to LEF.

According to theorem 2 in Banerjee (2007), assume that $P(\mathcal{X} \mid \mathcal{Y} = i)$ belongs to the same LEF for all $i \in \mathcal{Y}$. Also, since $\mathcal{Y}$ is discrete and finite, $P(\mathcal{Y})$ is a categorical distribution and hence, it belongs to LEF. This means that there exist parameters $\overline{\alpha} \in \mathbb{R}^{s-1}$ and $\overline{\theta_i} \in \mathbb{R}^t$ for all $i \in \mathcal{Y}$ such that

$$
\begin{aligned}
P_{\overline{\alpha}}(\mathcal{Y} = i) =& \frac{\exp S(i)^\intercal \overline{\alpha}}{\sum_y \exp S(y)^\intercal \overline{\alpha}} \\
P_{\overline{\theta_i}}(x \mid \mathcal{Y} = i) =& \frac{\exp T(x)^\intercal \overline{\theta_i}}{\int_x \exp T(x)^\intercal \overline{\theta_i}}
\end{aligned}
\tag{B.1}
$$

where S and T are sufficient statistics of $\mathcal{Y}$ and $\mathcal{X}$ respectively. If $\overline{\theta}$ is the matrix having $\overline{\theta_i}$ as i-th row, name $\overline{\eta} = (\overline{\alpha}, \overline{\theta})$ and write

$$
\begin{aligned}
P_{\overline{\eta}}(\mathcal{Y} = i, x) =& P_{\overline{\alpha}}(\mathcal{Y} = i) P_{\overline{\theta_i}}(x \mid \mathcal{Y} = i) \\
=& \frac{\exp S(i)^\intercal \overline{\alpha}}{\sum_y \exp S(y)^\intercal \overline{\alpha}} \frac{\exp T(x)^\intercal \overline{\theta_i}}{\int_x \exp T(x)^\intercal \overline{\theta_i}} \\
=& \frac{\exp S(i)^\intercal \overline{\alpha} + T(x)^\intercal \overline{\theta_i}}{\sum_y \exp S(y)^\intercal \overline{\alpha} \int_x \exp T(x)^\intercal \overline{\theta_i}}
\end{aligned}
\tag{B.2}
$$

To prove the result, it is enough to find a change of variables from $\overline{\eta} = (\overline{\alpha}, \overline{\theta})$ to $\eta = (\alpha, \beta)$ satisfying $P_{\overline{\eta}}(x, y) = P_\eta(x, y)$ where

$$
P_\eta(\mathcal{Y} = i, x) = \frac{\exp S(i)^\intercal \alpha + T(x)^\intercal \beta_i}{\int_x \sum_y \exp S(y)^\intercal \alpha + T(x)^\intercal \beta_y}
\tag{B.3}
$$

since η is the natural parametrization of a LEF.

In particular, the change of variables has to satisfy that $P_{\overline{\eta}}(x \mid \mathcal{Y} = i) = P_\eta(x \mid \mathcal{Y} = i)$ and $P_{\overline{\eta}}(y) = P_\eta(y)$. Start with the conditional probability and observe that

$$\text{(B.4)} \qquad \begin{aligned} P_\eta(x \mid \mathcal{Y} = i) = \frac{P_\eta(\mathcal{Y} = i, x)}{\int_x P_\eta(\mathcal{Y} = i, x)} &= \frac{\exp S(i)^\intercal \alpha + T(x)^\intercal \beta_i}{\int_x \exp S(i)^\intercal \alpha + T(x)^\intercal \beta_i} \\ &= \frac{\exp T(x)^\intercal \beta_i}{\int_x \exp T(x)^\intercal \beta_i} \end{aligned}$$

Last equation matches exactly with Equation $B.1$ by just setting $\beta = \overline{\theta}$. To complete the change of variables continue by matching $P_{\overline{\eta}}(y) = P_\eta(y)$.

$$\text{(B.5)} \qquad \begin{aligned} P_\eta(\mathcal{Y} = i) &= \frac{\int_x \exp S(i)^\intercal \alpha + T(x)^\intercal \beta_i}{\sum_j \int_x \exp S(j)^\intercal \alpha + T(x)^\intercal \beta_j} \\ &= \frac{\exp\left(S(i)^\intercal \alpha\right) \int_x \exp T(x)^\intercal \beta_i}{\sum_j \exp\left(S(j)^\intercal \alpha\right) \int_x \exp T(x)^\intercal \beta_j} \\ &= \frac{\exp S(i)^\intercal \alpha + \log A_i}{\sum_j \exp S(j)^\intercal \alpha + \log A_j} \end{aligned}$$

where $A_i = \int_x \exp T(x)^\intercal \beta_i$. Last equation must coincide with Equation B.1. That is

$$\text{(B.6)} \qquad P_\eta(\mathcal{Y} = i) = P_{\overline{\eta}}(\mathcal{Y} = i) \iff \frac{\exp S(i)^\intercal \alpha + \log A_i}{\sum_j \exp S(j)^\intercal \alpha + \log A_j} = \frac{\exp S(i)^\intercal \overline{\alpha}}{\sum_y \exp S(y)^\intercal \overline{\alpha}}$$

To simplify, assume S is canonical. That is $S(i) = e_i$ is the i-th canonical vector for all $i \neq s$ and $S(s) = 0 \in \mathbb{R}^{s-1}$. Note that it is enough to prove that there exists a $\mu \in \mathbb{R}$ such that

$$\text{(B.7)} \qquad S(i)^\intercal \alpha + \log A_i - \mu = S(i)^\intercal \overline{\alpha}, \qquad \forall i \in \mathcal{Y}$$

because as a consequence, Equation $B.6$ clearly holds. In our case, it is $S(i)^\intercal \alpha = \alpha_i$ when $i \neq s$ and $S(i)^\intercal \alpha = 0$, and therefore the solution is

$$\text{(B.8)} \qquad \begin{aligned} \alpha + \begin{pmatrix} \log A_1 \\ \vdots \\ \log A_{s-1} \end{pmatrix} - \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \cdot \mu &= \overline{\alpha} \\ \mu &= \log A_s \end{aligned}$$

and the proof is completed when S is canonical.

Prove now the result for a general sufficient statistic S. Equation B.7 describes the below linear equations system

$$\begin{aligned}
&\mathbf{S}\alpha + \begin{pmatrix} \log A_1 \\ \vdots \\ \log A_{s-1} \end{pmatrix} - \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \cdot \mu = \mathbf{S}\overline{\alpha} \\
&S(s)^\intercal \alpha + \log A_s - \mu = S(s)^\intercal \overline{\alpha}
\end{aligned} \tag{B.9}$$

where $\mathbf{S}$ is the matrix having $S(1), ..., S(s-1)$ as rows. Since S is a sufficient statistic, assume without loss of generality that $S(1), ..., S(s-1)$ are linearly independent vectors, and then $\mathbf{S}$ is invertible. Finally, it is easy to check that the change of variables is

$$\begin{aligned}
&\alpha + \mathbf{S}^{-1} \left(\begin{pmatrix} \log A_1 \\ \vdots \\ \log A_{s-1} \end{pmatrix} - \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \cdot \mu \right) = \overline{\alpha} \\
&\mu = \frac{S(s)^\intercal \mathbf{S}^{-1} \begin{pmatrix} \log A_1 \\ \vdots \\ \log A_{s-1} \end{pmatrix} - \log A_s}{S(s)^\intercal \mathbf{S}^{-1} \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} - 1}
\end{aligned} \tag{B.10}$$

The converse implication is straightforward. Assuming that $P(\mathcal{X}, \mathcal{Y})$ belongs to LEF, and therefore assuming Equation B.3, start by expressing the conditional probability distributions of $\mathcal{Y}$ given $\mathcal{X}$ in η.

$$\begin{aligned}
P_\eta(y \mid x) =& \frac{P_\eta(x, y)}{\sum_y P_\eta(x, y)} \\
=& \frac{\exp S(y)^\intercal \alpha + T(x)^\intercal \beta_y}{\sum_y \exp S(y)^\intercal \alpha + T(x)^\intercal \beta_y}
\end{aligned} \tag{B.11}$$

and compute the log-odds ratio

$$\begin{aligned}
\log \frac{P_\eta(x \mid \mathcal{Y} = k)}{P_\eta(x \mid \mathcal{Y} = h)} =& S(k)^\intercal \alpha + T(x)^\intercal \beta_k - (S(h)^\intercal \alpha + T(x)^\intercal \beta_h) \\
=& (S(k) - S(h))^\intercal \alpha + T(x)^\intercal (\beta_k - \beta_h)
\end{aligned} \tag{B.12}$$

which is clearly an affine function of features $\mathcal{X}$. □

B.2 Proof of Proposition 5.1.2

Proof. First prove that we can assume that $S(s) = 0$: observe that statistics $S_2 = S - S(s)$ and S are equivalent in Equation 5.3 (it also holds in Equation 5.1);

$$\begin{aligned} P_\eta(y \mid x) =& \frac{\exp S(s)^\intercal \cdot \alpha}{\exp S(s)^\intercal \cdot \alpha} P_\eta(y \mid x) \\ =& \frac{\exp\left((S(y) - S(s))^\intercal \alpha + T(x)^\intercal \beta_y\right)}{\sum_y \exp\left((S(y) - S(s))^\intercal \alpha + T(x)^\intercal \beta_y\right)} \\ =& \frac{\exp\left(S_2(y)^\intercal \alpha + T(x)^\intercal \beta_y\right)}{\sum_y \exp\left(S_2(y)^\intercal \alpha + T(x)^\intercal \beta_y\right)} . \end{aligned} \tag{B.13}$$

In fact, all statistics are equivalent under translation. Observe that for this new statistic, it is $S_2(s) = 0 \in \mathbb{R}^{s-1}$. So we can always directly assume that $S(s) = 0$.

Prove now that if S is a statistic such that $S(s) = 0$, then S can be converted into a canonical statistic with a linear transformation. Let **S** be the matrix of dimension $s - 1$ having $S(i)$ as i-th row for $1 \leq i < s$. Assuming that S is minimal and sufficient where $S(s) = 0$ implies that **S** is not singular. Hence we can apply the linear transformations $S_2 = \mathbf{S}^{-1} \cdot S$ and $\alpha_2 = \mathbf{S}^\intercal \cdot \alpha$ whereby construction, S_2 is a canonical statistic because:

$$\begin{aligned} S_2(i) =& \mathbf{S}^{-1} \cdot S = \delta_{i=j} \qquad \text{for } 1 \leq i < s \\ S_2(s) =& \mathbf{S}^{-1} \cdot 0 = 0 \, . \end{aligned} \tag{B.14}$$

Observe now that

$$\begin{aligned} S(y)^\intercal \cdot \alpha =& S(y)^\intercal \cdot \mathbf{S}^{-\intercal} \cdot \mathbf{S}^\intercal \cdot \alpha \\ =& S_2(y)^\intercal \cdot \alpha_2, \end{aligned} \tag{B.15}$$

Therefore, Equations 5.1 and 5.3 yield the same using S_2 and α_2 instead. □

B.3 Proof of Proposition 5.2.1

Proof. First, claim that

$$\nabla l(\eta, x, y) = \nabla h(\eta, x) \cdot (q_{\mathcal{Y}}(x, P_\eta) - e_s(y)) \tag{B.16}$$

Indeed,

$$
\begin{aligned}
\nabla \log P_\eta(y \mid x) =& \nabla \log P_\eta(x, y) - \nabla \log \sum_y P_\eta(x, y) \\
=& \nabla \log P_\eta(x, y) - \frac{\sum_y \nabla P_\eta(x, y)}{\sum_y P_\eta(x, y)} \\
=& \nabla \log P_\eta(x, y) - \frac{\sum_y P_\eta(x, y) \nabla \log P_\eta(x, y)}{\sum_y P_\eta(x, y)} \\
=& \nabla \log P_\eta(x, y) - \sum_y P_\eta(y \mid x) \nabla \log P_\eta(x, y) \\
=& \nabla h(\eta, x, y) - \mathbb{E}_{\mathcal{Y}|x}[\nabla h(\eta, x, y)]
\end{aligned} \tag{B.17}
$$

where $h(\eta, x, y) = \log P_\eta(x, y)$. Observe we can rewrite Equation B.17 as;

$$\nabla \log P_\eta(y \mid x) = -\nabla h(\eta, x) \cdot (q(x, \eta) - e_s(i)) \tag{B.18}$$

where $h(\eta, x) = (h(1, x, \eta), ..., h(s, x, \eta))$ implying the claim. From Equation B.16 observe that

$$\widetilde{\nabla} l(\eta, x, y) = \widetilde{\nabla} h(\eta, x) \cdot (q_{\mathcal{Y}}(x, P_\eta) - e_s(y)) \tag{B.19}$$

Finally, since the conditional log-loss is defined in a DFM, then use the previous equation and theorem 2.3.3 to finish the proof. □

B.4 Proof of Proposition 5.4.1

Proof. To simplify, break $\nabla_{\eta^*} = (\nabla_{\alpha^*}, \nabla_{\beta_1^*}, ..., \nabla_{\beta_s^*})$ and then it's clear that

$$\nabla h(\eta, x^*) = \begin{pmatrix} \nabla_{\alpha^*} h(\eta, x^*) \\ \nabla_{\beta_1^*} h(\eta, x^*) \\ \vdots \\ \nabla_{\beta_s^*} h(\eta, x^*) \end{pmatrix} \tag{B.20}$$

Start with $\nabla_{\alpha^*} h(\eta, x^*)$ expression. Observe that i-th column of $\nabla_{\alpha^*} h(\eta, x^*)$ is

$$
\begin{aligned}
\nabla_{\alpha^*} \log P_{\eta^*}(\mathcal{Y} = i, x) =& \nabla_{\alpha^*} \log P_{\alpha^*}(\mathcal{Y} = i) + \nabla_{\alpha^*} \log P_{\alpha^*}(x \mid \mathcal{Y} = i) \\
=& \nabla_{\alpha^*} \log P_{\alpha^*}(\mathcal{Y} = i) + d_{\alpha^*} \theta_i \nabla_{\theta_i} \log P_{\theta_i}(x \mid \mathcal{Y} = i)
\end{aligned} \tag{B.21}
$$

where in the last step the chain rule is applied and $d_{\alpha^*}\theta_i$ stands for the Jacobian of θ_i with respect to α^*.

Assume the canonical parametrization is used, then according to Equation 5.6 write

$$\alpha^* = \begin{pmatrix} P_{\eta^*}(\mathcal{Y}=1) \\ \vdots \\ P_{\eta^*}(\mathcal{Y}=s-1) \end{pmatrix} \tag{B.22}$$

From equations $B.22$ and 5.7 obtain

$$\begin{aligned} \nabla_{\alpha^*} \log P_{\alpha^*}(\mathcal{Y}=i) =& \frac{1}{P_{\alpha^*}(\mathcal{Y}=i)} \begin{cases} e_{s-1}(i) & i \neq s \\ (-\mathbf{1}) & i = s \end{cases} \\ d_{\alpha^*}\theta_i =& \frac{-1}{P_{\alpha^*}(\mathcal{Y}=i)} \begin{cases} e_{s-1}(i) \cdot \theta_i^\intercal & i \neq s \\ (-\mathbf{1}) \cdot \theta_i^\intercal & i = s \end{cases} \end{aligned} \tag{B.23}$$

where $e_{s-1}(i)$ is the i-th canonical $s-1$ dimensional vector and $\mathbf{1} = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$. From here deduce,

$$\begin{aligned} \nabla_{\alpha^*} h(\eta, x^*) =& K_{(s-1)\times s} \cdot diag(d(x,1,\zeta^*), ..., d(x,s,\zeta^*)) \\ d(x,y,\zeta^*) =& \frac{1 - \theta_y^\intercal \nabla_{\theta_y} \log P(x \mid y)}{P_{\zeta^*}(y)} \end{aligned} \tag{B.24}$$

The part $\nabla_{\beta_k^*} h(\eta, x^*)$ follows the same steps. Observe that i-th column of $\nabla_{\beta_y^*} h(\eta, x^*)$ is

$$\begin{aligned} \nabla_{\beta_y^*} \log P_{\eta^*}(\mathcal{Y}=i, x) =& \nabla_{\beta_y^*} \log P_{\beta_y^*}(x \mid \mathcal{Y}) \\ =& d_{\beta_y^*}\theta_i \nabla_{\theta_i} \log P_{\theta_i}(x \mid \mathcal{Y}=i) \\ =& \begin{cases} 0 & y \neq i \\ \frac{\nabla_{\theta_i} \log P_{\theta_i}(x \mid \mathcal{Y}=i)}{P_{\zeta^*}(y)} & y = i \end{cases} \end{aligned} \tag{B.25}$$

and therefore the claim is proved. □

B.5 Proof of Proposition 5.4.2

Proof. Let A be the cost of computing $\nabla_{\theta_k} \log P_{\theta_k}(x \mid y)$. Prove first the next claim: the number of operations required to compute $\nabla_{\zeta^*} h(x, \zeta^*)$ is

$$s \cdot (A + 3t + 2) - 1 \tag{B.26}$$

and hence $O(s \cdot (A + t))$.

Indeed, use Proposition 5.4.1 to prove the claim. Start with $d(x, 1, \zeta^*)$ computation cost. Terms $\nabla_{\theta_k} \log P_{\theta_k}(x \mid y)$ for every $y \in \mathcal{Y}$ need $s \cdot A$ operations. The cost of computing $P_{\zeta^*}(y)$ for every $y \in \mathcal{Y}$ is $s - 1$ according to Equation 5.6 (only the term $P_{\zeta^*}(s)$ requires operations). Obtain term $d(x, y, \zeta^*)$ after $2t + 1$ operations ($2t - 1$ for the scalar product of vectors, 1 for the subtraction in the numerator and 1 last operation for the division). Since this needs to be done for every $y \in \mathcal{Y}$ then $d(x, 1, \zeta^*), ..., d(x, s, \zeta^*)$ is known with $s \cdot (2t + 1)$ operations. Continue now with the costs of $\nabla_{\alpha^*} h(x, \zeta^*)$. The term $\nabla_{\alpha^*} h(x, \zeta^*)$ is obtained with the product of matrices K_s (which is almost the identity matrix) and a diagonal matrix, which does not require any operation (it is just a transformation). Finally $\nabla_{\beta_k^*} h(x, \zeta^*)$ demands for t divisions for every $y \in \mathcal{Y}$ and therefore for $s \cdot t$ operations. The total amount of operations needed to compute $\nabla_{\zeta^*} h(x, \zeta^*)$ is then

$$\begin{aligned} C(\nabla_{\zeta^*} h(x, \zeta^*)) &= sA + (s - 1) + s(2t + 1) + st \\ &= s(A + 3t + 2) - 1 \end{aligned}, \tag{B.27}$$

and hence, the claim is proved.

To the previous analysis, add the costs represented by Equation 5.9. That is, analyze the costs of computing $q_{\mathcal{Y}}(x, P_\eta)$ and then the products shown in that equation.

The vector $q_{\mathcal{Y}}(x, P_\eta)$ consist on computing $P_\eta(y \mid x)$ for every $y \in \mathcal{Y}$. Using Equation 5.3, $q_{\mathcal{Y}}(x, P_\eta)$ needs $2t + 1$ operations for the scalar products $T(x)^\intercal \beta_y$, 1 subtraction in $S(y)^\intercal \alpha - T(x)^\intercal \beta_y$ (recall that S statistic is canonical), then 1 exponentiation and finally 1 division. This is done for every $y \in \mathcal{Y}$. The denominator is the same for every y so it can be computed just once with $s - 1$ sums. The total is

$$2ts + 5s - 1 \tag{B.28}$$

operations.

Finally, the operations described in Equation 5.9 are 1 for subtraction $(q_{\mathcal{Y}}(x, P_\eta) - e_s(y))$, $s-1$ for the product $\nabla_{\alpha^*} h(x,\zeta^*) \cdot \nabla(q_{\mathcal{Y}}(x,P_\eta) - e_s(y))$ and t operations for $\nabla_{\beta_y^*} h(x,\zeta^*) \cdot \nabla(q_{\mathcal{Y}}(x,P_\eta) - e_s(y))$ product, where this last product needs to be done for every $y \in \mathcal{Y}$. The total operations for this block it is then

$$s + s \cdot t \tag{B.29}$$

To conclude the proof, the total operations needed is

$$s \cdot (A + 6t + 8) - 2 \tag{B.30}$$

and the complexity order is $O(s \cdot (A + t))$. □

B.6 Proof of Proposition 5.5.1

Proof. Compute $\nabla_{\theta_y} \log P_{\theta_y}(x|y)$ and proposition 5.4.1 finishes the proof. Parameters $\theta_y = (\theta_{y,1}, ..., \theta_{y,m-1})$ are the expectation parameters of the probability distribution $P_{\theta_y}(x|y)$, which belong to LEF.

Recall that the canonical statistics S and T are taken and by Equation 5.7 deduce

$$P_{\theta_y}(x|y) = \begin{cases} \theta_{x,y} & x \neq m \\ 1 - \sum_j \theta_{y,j} & x = m \end{cases} \tag{B.31}$$

which clearly implies

$$\nabla_{\theta_y} \log P_{\theta_y}(x|y) = \frac{1}{P_{\theta_y}(x|y)} \begin{cases} e_{m-1}(x) & x \neq m \\ \textbf{-1}_{m-1} & x = m \end{cases} \tag{B.32}$$

Finally, observe

$$d(x, y, \eta^*) = \frac{1 - \theta_y^\intercal \nabla_{\theta_y} \log P_{\theta_y}(x \mid y)}{P_{\eta^*}(y)} = \begin{cases} 0 & x \neq m \\ \frac{1}{P_{\theta_y}(x|y) P_{\eta^*}(y)} & x = m \end{cases} \tag{B.33}$$

Substitute the computations in proposition 5.4.1 to finish the proof. □

B.7 Proof of Theorem 5.6.1

Proof. The proof uses the Robbins-Siegmund theorem as a key tool. The steps taken are closely inspired by those taken in the proof of Theorem 3.2 in Sunehag et al. (2009).

Compute Taylor' second order approximation of $f(Z_{t+1})$, and after condition **C.2** apply Taylor's inequality

$$f(Z_{t+1}) \leq f(Z_t) - \gamma_t \nabla f(Z_t)^\intercal \cdot Y_t + \gamma_t^2 K \|X_t\|^2 \tag{B.34}$$

Therefore, applying the expectation conditioned to information at time t obtain

$$\mathbb{E}_t[f(Z_{t+1})] \leq f(Z_t) - \gamma_t \nabla f(Z_t)^\intercal \mathbb{E}_t[X_t] + \gamma_t^2 K \mathbb{E}_t \|X_t\|^2 \tag{B.35}$$

Use bound of **C.4** to the third term of the right-hand side

$$\mathbb{E}_t[f(Z_{t+1})] \leq f(Z_t) - \gamma_t \nabla f(Z_t)^\intercal \mathbb{E}_t[X_t] + \gamma_t^2 K (A + B f(Z_t)) \tag{B.36}$$

Finally, substitute $U_t = f(Z_t)$ and arrange terms to match with Equation 2.50

$$\mathbb{E}_t[U_{t+1}] \leq (1 + B\gamma_t^2 K) U_t - \gamma_t \nabla f(Z_t)^\intercal \mathbb{E}_t[X_t] + \gamma_t^2 K A \tag{B.37}$$

Note that theorem 2.7.3 conditions are satisfied, since condition **C.6** implies $\sum_t \beta_t = \sum_t BK\gamma^2 = BK \sum_t \gamma^2 < \infty$ and $\sum_t \epsilon_t = \sum_t KA\gamma^2 < \infty$. Hence, the Robbins-Siegmund theorem ensures that $U_t = f(Z_t)$ converges almost surely to a random variable and

$$\sum_t \zeta_t = \sum_t \gamma_t \nabla f(Z_t)^T \mathbb{E}_t[Y_t] < \infty \tag{B.38}$$

Now prove that $\lim_t f(Z_t) = f(\overline{\eta})$. If $f(Z_t)$ converges to some different random variable, condition **C.3**, second condition of **C.6** and Equation B.38 lead to a contradiction. Indeed, if $\lim_t f(Z_t) = v \neq f(\overline{\eta})$, use condition **C.3** and deduce that for a fixed $0 < \delta < v - f(\overline{\eta})$ there exists an N large enough and $\epsilon > 0$ such that

$$\nabla f(Z_t)^\intercal \mathbb{E}_t [X_t] \geq \epsilon \tag{B.39}$$

for all $t > N$. Therefore, Equation B.38 becomes

$$\begin{aligned}\sum_t \gamma_t \nabla f(Z_t)^T \mathbb{E}_t[X_t] &= \sum_t^N \gamma_t \nabla f(Z_t)^\intercal \mathbb{E}_t[X_t] + \sum_{t>N} \gamma_t \nabla f(Z_t)^\intercal \mathbb{E}_t[X_t] \\ &\geq \sum_t^N \gamma_t \nabla f(Z_t)^\intercal \mathbb{E}_t[X_t] + \sum_{t>N} \epsilon\gamma_t \\ &\geq \epsilon \sum_{t>N} \gamma_t\end{aligned} \tag{B.40}$$

Second condition in **C.6** applied to the right-hand side of the above equation assures that

$$\sum_t \gamma_t \nabla f(Z_t)^\intercal \mathbb{E}_t[X_t] = \infty \qquad a.s. \tag{B.41}$$

which contradicts Equation B.38.

Finally, it is only possible that $\lim_t f(Z_t) = f(\overline{\eta})$ almost surely as we wanted to prove. $\square$

B.8 Proof of condition C.2 in Theorem 5.6.2

Proof. Compute the hessian of

$$l(\eta) = \sum_{x,y} l(\eta, x, y)\overline{P}(x, y) \ , \tag{B.42}$$

where $\overline{P}$ is the unknown probability distribution in the probability space $(\overline{\Omega} = \mathcal{Y} \times \mathcal{X}, \overline{\mathcal{F}}, \overline{P})$. The gradient of $l(\eta, x, y)$ is

$$\begin{aligned}\nabla_\alpha l(\eta, x, y) &= S \cdot (q_\mathcal{Y}(x) - e_s(y)) \\ \nabla_{\beta_{y'}} l(\eta, x, y) &= (q_\mathcal{Y}(x)_{y'} - \delta_{y=y'}) \cdot T(x) \ ,\end{aligned} \tag{B.43}$$

where S is the matrix having $S(i)$ as i-th column for $i \in \mathcal{Y}$. Therefore, the hessian is

$$\begin{aligned}\nabla^2_\alpha l(\eta, x, y) &= S \cdot (diag(q_\mathcal{Y}(x)) - q_\mathcal{Y}(x) \cdot q_\mathcal{Y}(x)^\intercal) \cdot S^\intercal \\ \nabla_{\beta_{y_2}} \nabla_{\beta_{y_1}} l(\eta, x, y) &= \nabla_{\beta_{y_2}} q_\mathcal{Y}(x)_{y_1} \cdot T(x) \\ &= -T(x) \cdot T(x)^\intercal q_\mathcal{Y}(x)_{y_1} (q_\mathcal{Y}(x)_{y_2} - \delta_{y_1=y_2}) \\ \nabla_\alpha \nabla_{\beta_{y'}} l(\eta, x, y) &= \nabla_\alpha q_\mathcal{Y}(x)_{y'} \cdot T(x) \\ &= T(x) \cdot (q_\mathcal{Y}(x) - e_s(y')^\intercal \cdot S^\intercal \ .\end{aligned} \tag{B.44}$$

Observe how all matrices in Equation B.44 have their elements bounded once S and T statistics are fixed, since $\|q_{\mathcal{Y}}(x)\| \leq 1$. Therefore

$$\|\nabla^2 l(\eta, x, y)\| \leq 2K_{x,y} \tag{B.45}$$

for some positive numbers $K_{x,y}$. Define $K = \max_{x,y} K_{x,y}$. then finally

$$\begin{aligned}
\|\nabla^2_\eta l(\eta)\| =& \|\nabla^2 \sum_{x,y} l(\eta, x, y) \cdot \overline{P}(x, y)\| \\
=& \| \sum_{x,y} \nabla^2 l(\eta, x, y) \cdot \overline{P}(x, y)\| \\
\leq& \sum_{x,y} \|\nabla^2 l(\eta, x, y)\| \cdot \overline{P}(x, y) \\
\leq& \sum_{x,y} 2 \cdot K_{x,y} \cdot \overline{P}(x, y) \\
\leq& 2 \cdot K \sum_{x,y} \overline{P}(x, y) \\
=& 2 \cdot K
\end{aligned} \tag{B.46}$$

□ □

B.9 Proof of condition C.4 in Theorem 5.6.2

Proof. Observe that for any ϵ and t large enough there exists A_{x_t} such that

(B.47)
$$\begin{aligned}
\|X_t(\omega)\|^2 &= (q_{\mathcal{Y}}(x_t, P_{Z_t(\omega)}) - e_s(y_t))^\intercal \cdot h(x_t, \zeta_t^*)^\intercal h(x_t, \zeta_t^*) \cdot (q_{\mathcal{Y}}(x_t, P_{Z_t(\omega)}) - e_s(y_t)) \\
&\leq A_{x_t} \|q_{\mathcal{Y}}(x_t, P_{Z_t(\omega)}) - e_s(y_t)\|^2
\end{aligned}$$

where

$$A_{x_t} \geq \|h(x_t, \zeta_t^*)^\intercal h(x_t, \zeta_t^*)\| + \epsilon \tag{B.48}$$

This is because ζ_t converges and because of theorem 5.5.1. Now

$$\begin{aligned}
\|q_{\mathcal{Y}}(x_t, P_{Z_t(\omega)}) - e_s(y_t)\|^2 &= 1 - 2P_{\eta_t}(y_t \mid x_t) + \sum_y P_{\eta_t}(y \mid x_t)^2 \\
&\leq s + 1
\end{aligned} \tag{B.49}$$

therefore $\|X_t\|^2 \leq A_{x_t}(s+1)$ and

$$\begin{aligned} \mathbb{E}_t\|X_t\|^2 &\leq \mathbb{E}_t[A_{x_t}(s+1)] \\ &\leq A'(s+1) = A \end{aligned} \tag{B.50}$$

where $A' = \max_x A_x$ and then condition **C.4** holds. $\square$

C Appendix: Convergence of Stochastic process

C.1 Proof of Corollary 6.3.1

To prove the corollary, it is enough to prove the generic Proposition C.1.1.

Proposition C.1.1. *Let $U_t \subset \mathbb{R}^k$ be non empty, closed and connected sets where $U_{t+1} \subset U_t$ for $t \in \mathbb{N}$ and let $V = \cap_t U_t$. Then V is a nonempty bounded set if, and only if, U_T is bounded for some $T \in \mathbb{N}$.*

Proof. Prove first that if U_T is bounded for some $T \in \mathbb{N}$, then $V = \cap_t U_t$ is a non-empty bounded set. Clearly, $V \subset U_T$ and, therefore, V is bounded, possibly empty. Observe that U_t for all $t \geq T$ is compact and closed. Then V is not empty, by the Cantor's intersection theorem.

Conversely, prove now that if V is a nonempty bounded set, then there exists T such that U_T is bounded. Assume V is non-empty bounded set, then there exists $r > 0$, such that $V \subset B_r(0)$ where $B_r(0)$ is the ball centered at 0 with radius r. Define

$$\begin{aligned} R &= \overline{B_{2r}(0)} \cap B_r(0)' \\ U_t^* &= U_t \cap R\,, \end{aligned} \tag{C.1}$$

where $\overline{B_{2r}(0)}$ is the closed ball of radius $2r$ and center 0 and $A' = \mathbb{R}^k \setminus A$. The sequence U_t^* is of compact and closed subsets, where $U_{t+1}^* \subset U_t^*$ and $\cap_t U_t^*$ is empty. Therefore, by Cantor's intersection theorem, there exists T such that U_T^* is empty. Then $U_T \subset R' = \overline{B_{2r}(0)}' \cup B_r(0))$. U_t is connected and R' it is not, which implies that either $U_T \subset \overline{B_{2r}(0)}'$ or $U_T \subset B_r(0))$. Since $V \subset U_T$ and $V \subset B_r(0))$, then it must be $V \subset U_T \subset B_r(0)$ and hence it is bounded as wanted to prove. $\square$

C.2 Bottou's Resemblance

Proposition 6.6.1 is a direct consequence of Proposition C.2.1, that we state and prove below, and Proposition 6.4.2.

Proposition C.2.1. *Let $Z = (X, \gamma)$ be a stochastic process and $\mathbb{X}$ be a vector field over $\mathbb{R}^k$. For $\delta > 0$ and $T \in \mathbb{N}$, define the vector pair set*

$$V_{\delta,T}(\mathbb{X}, Z) = \{(\mathbb{X}(\eta), v) \mid \eta \in \mathbb{R}^k \setminus B_\delta(K_{\mathbb{X}}), v \in EDS_Z(\eta, T)\} . \tag{C.2}$$

Then Z resembles to $\mathbb{X}$ if, and only if,

$$(\exists T \in \mathbb{N})(\forall \delta > 0)(\exists \epsilon > 0) \quad V_{\delta,T}(\mathbb{X}, Z) \text{ is } \epsilon\text{-acute} . \tag{C.3}$$

Proof. By definition, $V_{\delta,T}(\mathbb{X}, X)$ is ϵ-acute if, and only if, every vector pair (u, v) in $V_{\delta,T}(\mathbb{X}, X)$ is ϵ-acute. By definition, such vector pairs $(\mathbb{X}(\eta), v)$ with $v \in EDS_X(\eta, T)$ are ϵ-acute if, and only if,

$$(\forall \eta \in \mathbb{R}^k \setminus B_\delta(K_{\mathbb{X}})) \quad \mathbb{X}(\eta)^\intercal \cdot v \geq \epsilon > 0, \quad v \in EDS_X(\eta, T) . \tag{C.4}$$

Previous equation holds if, and only if, $EDS_X(\eta, T) \subset H_\epsilon(\mathbb{X})(\eta)$, $\eta \in \mathbb{R}^k \setminus B_\delta(K_{\mathbb{X}})$ as wanted to prove. □

C.3 Sunehag's Resemblance

The result that translates Theorem 6.6.1 with *resemblance* concepts is Proposition 6.6.2, that we prove below.

Proof. After Proposition C.2.1 and 6.4.3 deduce that Z belongs to the half-space of $\mathbb{X}$ if, and only if, there exists $T \in \mathbb{N}$ such that for every $\delta > 0$ and every $t \geq T$ there exist symmetric positive-definite $\mathcal{F}_t$-measurable random matrices B_t, such that

$$\begin{aligned} &\inf_{\substack{\eta \in \mathbb{R}^k \setminus B_\delta(K_{\mathbb{X}}) \\ t \geq T \\ \omega \in \Omega, Z_t(\omega) = \eta}} \mathbb{X}(\eta)^\intercal \cdot B_t(\omega) \cdot \mathbb{X}(\eta) > 0 , \\ &B_t \cdot \mathbb{X}(Z_t) = \mathbb{E}_t[X_t] \qquad Z_t(\omega) \notin K_{\mathbb{X}} . \end{aligned} \tag{C.5}$$

This matches with Equation (6.27). Matrix B_t is correctly and uniquely defined for all $t \geq T$ and all $\omega \in \Omega$, such that $Z_t(\omega) \notin K_{\mathbb{X}}$. Define $B_t = Id$ the identity matrix if $Z(\omega) \in K_{\mathbb{X}}$ and also define

$$Y_t := B_t^{-1} \cdot X_t \ . \tag{C.6}$$

Observe that $B_t \cdot Y_t = X_t$ and that Equation (6.26) is then met too finishing the proof. $\square$

Bibliography

Shun-ichi Amari. Natural Gradient Works Efficiently in Learning. *Neural Computation*, 276: 251–276, 1998.

Shun-ichi Amari. *Information geometry and its applications*, volume 5416. Springer, 2016. ISBN 978-4-431-55977-1.

Shun-ichi Amari and Hiroshi Nagaoka. *Methods of information geometry*, volume 191. American Mathematical Soc., 2000.

Shun-ichi Amari, Ole E Barndorff-Nielsen, Robert E Kass, Steffen L Lauritzen, and CR Rao. Differential geometry in statistical inference. *Lecture Notes-Monograph Series*, 10:i–240, 1987. ISSN 07492170. URL http://www.jstor.org/stable/4355557.

Arindam Banerjee. An Analysis of Logistic Models: Exponential Family Connections and Online Performance. In *Proceedings of the 2007 SIAM International Conference on Data Mining*, Proceedings, pages 204–215. Society for Industrial and Applied Mathematics, April 2007. ISBN 978-0-89871-630-6. doi: 10.1137/1.9781611972771.19. URL https://epubs.siam.org/doi/abs/10.1137/1.9781611972771.19.

Richard F Bass. *Stochastic processes*, volume 33. Cambridge University Press, 2011.

Moshe E. Ben-Akiva, Steven R. Lerman, and Steven R. Lerman. *Discrete Choice Analysis: Theory and Application to Travel Demand*. MIT Press, 1985. ISBN 978-0-262-02217-0. Google-Books-ID: oLC6ZYPs9UoC.

CJ Biesheuvel, Yvonne Vergouwe, EW Steyerberg, DE Grobbee, and KGM Moons. Polytomous logistic regression analysis could be applied more often in diagnostic research. *Journal of clinical epidemiology*, 61(2):125–134, 2008.

Patrick Billingsley. *Probability and measure*. (Wiley series in probability and mathematical statistics). Wiley, New York, 2nd edition, 1986. ISBN 0471804789.

Léon Bottou. Online algorithms and stochastic approximations. In David Saad, editor, *Online Learning and Neural Networks*. Cambridge University Press, Cambridge, UK, 1998. URL `http://leon.bottou.org/papers/bottou-98x`. revised, oct 2012.

Leon Bottou. *Stochastic Gradient Descent Tricks*, volume 7700 of *Lecture Notes in Computer Science (LNCS)*, pages 430–445. Springer, neural networks, tricks of the trade, reloaded edition, January 2012.

Shelley B Bull, Juan Pablo Lewinger, and Sophia SF Lee. Confidence intervals for multinomial logistic regression in sparse data. *Statistics in Medicine*, 26(4):903–918, 2007.

Ovidiu Calin and Constantin Udrişte. *Geometric modeling in probability and statistics*, volume 121. Springer, 2014.

Augustin Cauchy. Méthode générale pour la résolution des systemes d'équations simultanées. *Comp. Rend. Sci. Paris*, 25(1847):536–538, 1847.

Nikolai Nikolaevich Čencov. *Statistical decision rules and optimal inference*, volume 53 of *Translations of Mathematical Monographs*. American Mathematical Society, Providence, R.I., 1982. ISBN 0-8218-4502-0. Translation from the Russian edited by Lev J. Leifman.

Paul Covington, Jay Adams, and Emre Sargin. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys '16, page 191–198, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450340359. doi: 10.1145/2959100.2959190. URL `https://doi.org/10.1145/2959100.2959190`.

Michael J Daniels and Constantine Gatsonis. Hierarchical polytomous regression models with applications to health services research. *Statistics in Medicine*, 16(20):2311–2325, 1997.

Yann Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization, 2014.

Manfredo Perdigão do Carmo. *Riemannian geometry*. Mathematics: theory & applications. Birkhäuser, Boston Basel Berlin, corrected at 14th printing$h2013 edition, 2013. ISBN 978-0-8176-3490-2 978-1-4757-2201-7.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, 12(null):2121–2159, 07 2011. ISSN 1532-4435.

R. M. Dudley. *Real Analysis and Probability*. Cambridge Studies in Advanced Mathematics. Cambridge University Press, 2 edition, 2002. doi: 10.1017/CBO9780511755347.

Kenji Fukumizu and Shun-ichi Amari. Local minima and plateaus in hierarchical structures of multilayer perceptrons. *Neural Networks*, 13(3):317–327, 2000. ISSN 0893-6080. doi: https://doi.org/10.1016/S0893-6080(00)00009-5. URL https://www.sciencedirect.com/science/article/pii/S0893608000000095.

Philip E Gill, Walter Murray, and Margaret H Wright. *Practical optimization*. SIAM, 2019.

Jiang Hu, Xin Liu, Zai-Wen Wen, and Ya-Xiang Yuan. A Brief Introduction to Manifold Optimization. *Journal of the Operations Research Society of China*, 8(2):199–248, June 2020. ISSN 2194-6698. doi: 10.1007/s40305-020-00295-9. URL https://doi.org/10.1007/s40305-020-00295-9.

D. P. Kingma and L. J. Ba. Adam: A Method for Stochastic Optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, 2015. URL https://dare.uva.nl/search?identifier=a20791d3-1aff-464a-8544-268383c33a75.

Shōshichi Kobayashi and Katsumi Nomizu. *Foundations of Differential Geometry*, volume 1. Interscience Publishers, 1963. ISBN 9780470496473.

Bernard Osgood Koopman. On distributions admitting a sufficient statistic. *Transactions of the American Mathematical Society*, 39(3):399–409, 1936. ISSN 00029947. URL http://www.jstor.org/stable/1989758.

John M Lee. *Introduction to Riemannian manifolds*. Springer, 2018.

Erich L Lehmann and George Casella. *Theory of point estimation*. Springer Science & Business Media, 2006.

Jimmie Leppink. Multicategory nominal choices. In *The Art of Modelling the Learning Process*, pages 103–110. Springer, 2020.

Jun Li, José M Bioucas-Dias, and Antonio Plaza. Spectral–spatial hyperspectral image segmentation using subspace multinomial logistic regression and markov random fields. *IEEE Transactions on Geoscience and Remote Sensing*, 50(3):809–823, 2012. doi: 10.1109/TGRS.2011.2162649.

Wu Lin, M. E. Khan, and Mark W. Schmidt. Fast and simple natural-gradient variational inference with mixture of exponential-family approximations. In *ICML*, 2019.

Noboru Murata. A statistical study of on-line learning. *Online Learning and Neural Networks. Cambridge University Press, Cambridge, UK*, pages 63–92, 1998.

Michael K Murray and John W Rice. *Differential geometry and statistics*, volume 48. CRC Press, 1993.

A Nemirovskiĭ and D. Yudin. *Problem Complexity and Method Efficiency in Optimization*. A Wiley-Interscience publication. Wiley, 1983. ISBN 9780471103455.

Frank Nielsen. An elementary introduction to information geometry. *arXiv:1808.08271 [cs, math, stat]*, August 2018. URL http://arxiv.org/abs/1808.08271. arXiv: 1808.08271.

C. R. Rao. Information and accuracy attainable in the estimation of statistical parameters. *Bull Calcutta. Math. Soc.*, 37:81–91, 1945.

G. Raskutti and S. Mukherjee. The Information Geometry of Mirror Descent. *IEEE Transactions on Information Theory*, 61:1451–1457, March 2015. ISSN 0018-9448. doi: 10.1109/TIT.2015.2388583.

H. Robbins and D. Siegmund. A convergence theorem for non negative almost supermartingales and some applications. In Jagdish S. Rustagi, editor, *Optimizing Methods in Statistics*, pages 233 – 257. Academic Press, 1971. ISBN 978-0-12-604550-5.

Herbert Robbins and Sutton Monro. A Stochastic Approximation Method. *The Annals of Mathematical Statistics*, 22(3):400 – 407, 1951. doi: 10.1214/aoms/1177729586. URL https://doi.org/10.1214/aoms/1177729586.

Sheldon M Ross. *Stochastic Processes*, volume 2 of *Wiley series in probability and mathematical statistics*. Wiley, 1996. ISBN 9780471120629.

Halsey Lawrence Royden and Patrick Fitzpatrick. *Real analysis*, volume 32. Macmillan New York, 1988.

Sebastian Ruder. An overview of gradient descent optimization algorithms. *CoRR*, abs/1609.04747, 2016. URL http://arxiv.org/abs/1609.04747.

Sirpa Saarinen, Randall Bramley, and George Cybenko. Ill-conditioning in neural network training problems. *SIAM Journal on Scientific Computing*, 14(3):693–714, 1993.

Borja Sánchez-López and Jesus Cerquides. Convergent stochastic almost natural gradient descent. *Artificial Intelligence Research and Development- Proceedings of the 22nd International Conference of the Catalan Association for Artificial Intelligence*, 319:54–63, 2019.

Peter Norvig Stuart Russell. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 1st edition, 1995. ISBN 0131038052, 9780131038059.

Ke Sun and Frank Nielsen. Relative natural gradient for learning large complex models. *arXiv preprint arXiv:1606.06069*, 2016.

Peter Sunehag, Jochen Trumpf, S. V. N. Vishwanathan, and Nicol Schraudolph. Variable Metric Stochastic Approximation Theory. In *Artificial Intelligence and Statistics*, pages 560–566, 04 2009. URL http://proceedings.mlr.press/v5/sunehag09a.html.

Borja Sánchez-López. Multinomial logistic regression and stochastic natural gradient descent. Master's thesis, Universitat de Barcelona, 2018. URL http://hdl.handle.net/2445/129823.

Borja Sánchez-López and Jesus Cerquides. Dual stochastic natural gradient descent and convergence of interior half-space gradient approximations, 2020. URL https://arxiv.org/abs/2001.06744.

Borja Sánchez-López and Jesus Cerquides. On the convergence of stochastic process convergence proofs. *Mathematics*, 9(13), 2021. ISSN 2227-7390. doi: 10.3390/math9131470. URL https://www.mdpi.com/2227-7390/9/13/1470.

Borja Sánchez-López and Jesus Cerquides. die-problem, 9 2022a. URL https://github.com/Kissyfur/die-problem.

Borja Sánchez-López and Jesus Cerquides. dsngd, 9 2022b. URL https://github.com/Kissyfur/dsngd.

Roberto Tadei, Guido Perboli, and Daniele Manerba. A Recent Approach to Derive the Multinomial Logit Model for Choice Probability. In Patrizia Daniele and Laura Scrimali, editors, *New Trends in Emerging Complex Real Life Problems: ODS, Taormina, Italy, September 10–13, 2018*, AIRO Springer Series, pages 473–481. Springer International Publishing, Cham, 2018. ISBN 978-3-030-00473-6.

Terence Tao. *An Introduction to Measure Theory*. Graduate Studies in Mathematics. American Mathematical Society, 2011.

Philip S. Thomas. Genga: A generalization of natural gradient ascent with positive and negative convergence results. *31st International Conference on Machine Learning, ICML 2014*, 5: 3533–3541, 01 2014.

Vladimir N Vapnik. Principles of risk minimization for learning theory. In *Proceedings of the 4th International Conference on Neural Information Processing Systems*, NIPS'91, pages 831–838, San Francisco, CA, USA, December 1991. Morgan Kaufmann Publishers Inc. ISBN 978-1-55860-222-9.

J. K. Wani. On the linear exponential family. *Mathematical Proceedings of the Cambridge Philosophical Society*, 64(2):481–483, 1968. doi: 10.1017/S0305004100043097.

Nayyar A Zaidi, Mark J Carman, Jesús Cerquides, and Geoffrey I Webb. Naive-bayes inspired effective pre-conditioner for speeding-up logistic regression. In *2014 IEEE International Conference on Data Mining*, pages 1097–1102. IEEE, 2014.

Matthew D. Zeiler. ADADELTA: An Adaptive Learning Rate Method. *arXiv:1212.5701 [cs]*, 12 2012. URL http://arxiv.org/abs/1212.5701. arXiv: 1212.5701.

Acronyms

AI Artificial Intelligence. 1, 2

CCM Conjugate Connection Manifold. 26

CSNGD Convergent Stochastic Natural Gradient Descent. xv, 8–10, 55–57, 59–63, 68, 74, 81, 101

DFM Dually Flat Manifold. viii, 6–10, 26–30, 33, 63–66, 111

DSNGD Dual Stochastic Natural Gradient Descent. xv, 7–10, 62, 64, 67–69, 71–75, 78–81, 83, 99, 101–103

FIM Fisher Information Metric. vii, 5, 11, 24, 25, 29, 44, 63, 65, 67

GD Gradient Descent. 3, 8, 10, 17–19, 37, 38, 48, 49

KL Kullback-Leibler divergence. xv, 29, 36, 44, 45, 50–52, 60, 61, 80

LEF Linear Exponential Family. 29, 63–65, 67, 72, 102, 107–109, 114

MEGD Maximum Entropy Gradient Descent. xv, 48–52

ML Machine Learning. vii, viii, xv, 1–3, 5, 6, 8, 11, 13, 17, 21, 30, 31, 33, 34, 39, 61, 63, 84, 101–103

MLE Maximum Likelihood Estimator. 59–61, 101

MLR Multinomial Logistic Regression. viii, 11, 39, 40, 47, 63, 64, 67, 101–103

MOD Manifold Optimized Descent. xv, 8–10, 43, 46–53, 55, 59, 101

NGD Natural Gradient Descent. 6, 8, 10, 31, 37, 38, 43, 44, 46–49

RC Riemman-Christoffel curvature. 26

SGD Stocastic Gradient Descent. xv, 8, 33, 37, 38, 41, 44–46, 49, 50, 52, 53, 56, 59–63, 68, 72, 78–80, 84, 88, 90, 101, 102

SMEGD Stochastic Maximum Entropy Gradient Descent. 49

SMOD Stochastic Manifold Optimized Descent. 47

SNGD Stochastic Natural Gradient Descent. xv, 8, 33, 38, 43–46, 49–52, 55, 57, 60, 61, 63, 79, 101

Index