UCLouvain

École polytechnique de Louvain

Dimensionality Reduction for Global Optimization : Beyond Low Effective Dimension

Author: **Antonin OSWALD**
Supervisor: **Estelle MASSART**
Readers: **François GLINEUR, Timothé TAMINIAU**
Academic year 2024–2025
Master [120] in Mathematical Engineering

Acknowledgements

I would like to warmly thank Professor Massart for her support and her attentive supervision throughout my work. Her responsiveness at every stage, her expertise, and her encouragement have made a lasting difference. I am especially grateful for the clarity of her guidance and her unwavering dedication.

Many thanks as well to my readers, Professor Glineur and Mr. Taminiau, for taking the time to assess my work. I truly look forward to exchanging with them and hearing their insights.

I also wish to thank, my parents, grandparents, brother, sister, and friends for their constant and kind support throughout my studies. Their human perspective and wise advice have often helped me take a step back and move forward with confidence.

I am grateful to all my fellow students who shared this academic path with me. Their friendship, curiosity, and mutual support made these years not only intellectually enriching but also deeply enjoyable and memorable. I also thank them for the warm and convivial atmosphere they helped create throughout this journey.

Finally, a heartfelt thank you to my partner for her attentive presence and unwavering emotional support throughout this journey. Her ability to listen with care, her patience in every moment, and her constant encouragement have been an invaluable source of strength. Her love and kindness bring light even in the most demanding times, and I am deeply grateful for her enduring support and presence in my life.

Abstract

Many fields of engineering require the global minimization of objective functions defined over high-dimensional domains. To ensure tractability, it is crucial to develop optimization methods that exploit structural properties of these functions to accelerate computations. The baseline of this work concerns functions with *low effective dimension*, which depend only on a linear subspace of the full domain, referred to as the *effective subspace*. The objective of this thesis is to study generalizations of this concept.

The first part of this work focuses on functions with *approximate low effective dimension*, where deterministic variations remain small in the orthogonal complement of the effective subspace. Theoretical results are established to characterize this extension, and existing algorithms are adapted accordingly. Numerical experiments are performed and demonstrate the competitiveness of the resulting extension.

The second part addresses the problem of identifying the effective subspace of functions with *low effective dimension* in scenarios where only noisy evaluations of the objective function are available. A Riemannian optimization framework is adopted for this purpose. This approach improves estimation accuracy and is more robust to noise, although it incurs higher computational costs and reduced scalability with increasing ambient dimension.

Contents

Introduction

This master thesis considers the unconstrained global optimization problem

$$\min_{x \in \mathbb{R}^D} f(x), \tag{1}$$

of $f : \mathbb{R}^D \to \mathbb{R}$, with f continuously differentiable, possibly non-convex and deterministic. The domain dimension D could be very large. This situation is encountered for instance in (over-parametrized) neural networks [54, 7, 71, 42], where the number of (hyper-)parameters to be optimized is rather large, even for small networks. This situation is encountered in high-dimensional statistics, where the number of features in the data is typically large [29, 57]. In such contexts, estimating parameters through the minimization of a loss function, such as in maximum likelihood estimation, can be computationally demanding. High-dimensional Bayesian optimization is also relevant in areas such as control and policy search; see, for example, the introductions of [75, 27] for related applications. Learning theory problems also involve high-dimensional optimization, particularly in the recovery of sparse vectors in large-dimensional spaces [70]. Similarly, the inference of causal effects in the modeling of physical and biological systems often depends on a large number of parameters [50, 34], resulting in large-scale optimization problems linked to statistical estimation. Further applications arise in high-dimensional signal processing, driven by the increasing use of multi-sensor systems [45].

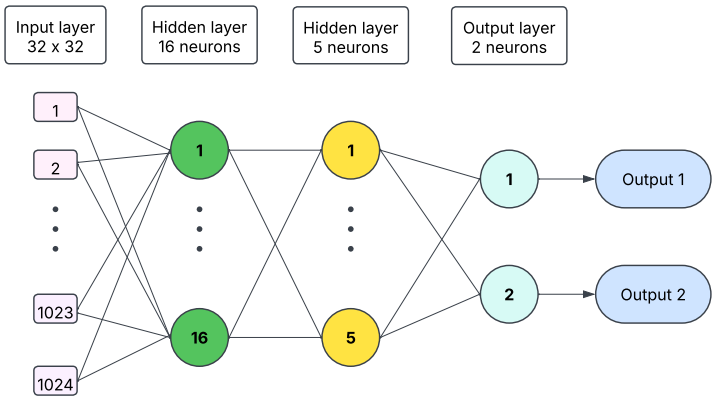


Figure 1: A simple fully connected neural network without bias terms. The number of parameters (weights) to tune is equal to $1024 \times 16 + 16 \times 5 + 5 \times 2 = 16474$.

High-dimensional problems are not only challenged by the exponential growth of the search space, but also by the increased computational cost of standard optimization tools, such as (finite-difference approximations of) gradients and Hessians. In other words, such problems face the *curse of dimensionality*. The growing dimensionality of modern problems in science and engineering creates a pressing need for scalable dimensionality reduction methods tailored to large-scale optimization.

A common strategy to mitigate the challenges of high dimensionality is to restrict the optimization process to a lower-dimensional subspace. Coordinate descent methods, a local method performing first-order updates along individual coordinates, significantly reduce the computational cost per iteration [2, 8]. Their effectiveness in large-scale settings has been established in various contexts, notably in [52]. These methods have been extended to their block-coordinate counterparts, enabling updates along subsets of variables, as developed in [43, 41]. Such approaches are widely applied in large-scale optimization tasks, particularly in machine learning and signal processing, see, e.g., [4, 73, 56]. Further generalizations include proximal and higher-order variants designed for nonlinear optimization, such as randomized subspace Newton methods and cubic regularization techniques [31, 33, 32]. When first-order informations are not available, one can rely on derivative-free optimization. Several local algorithms were developed using one-dimensional random subspaces, see [6, 53, 63]. A low-dimensional approach is proposed in [19], for the particular case of nonlinear least-squares problems.

While the previous methods focus on local optimization, this master thesis addresses global optimization, where high dimensionality further complicates the search due to the presence of numerous local minima. Stochastic approaches, such as simulated annealing and multistart methods, offer practical alternatives [28]. Simulated annealing mimics thermal annealing to probabilistically escape local minima, while multistart methods increase global coverage by running local searches from multiple initial points. For a brief history of global optimization, see [46, Section 1.3]. For a complete introduction, see [35].

Fortunately, in many practical applications, the objective function f of (1) exhibits an exploitable structure: it varies significantly only along a low-dimensional subspace $\mathcal{S} \subset \mathbb{R}^D$, with $\dim(\mathcal{S}) = d_e \ll D$. Such functions are commonly referred to in the literature as having *low effective dimension* or *effective subspace*, or as being *multi-ridge* or *low-rank*. Many methods aim to learn the subspace of variation of f for optimization purposes. Some of them build on active subspace theory, a framework that captures the subspace of variation of a function using a second-moment matrix of its gradient, see [22]. In [15], the authors leverage active subspace theory to develop A-ASM, an adaptive algorithm that estimates the effective subspace of variation of a function by iteratively solving low-dimensional optimization problems and sampling gradients. The method is supported by probabilistic convergence guarantees. Active subspaces are also used in [23], in order to extend genetic algorithms.

Other relevant frameworks include *basis adaptation* [66], which identifies subspaces associated with low-dimensional quantities of interest, and *global sensitivity analysis* [59], which assesses the influence of input parameters on the output of mathematical models. Other works rely on the estimation of *multi-ridge* functions through queries of point values, see [21, 26, 68]. In [21], the function f is such that $f(x) = g(a \cdot x)$, with $a \in \mathbb{R}^D$. With the assumptions that a is sparse and compressible, along with g exhibiting smoothness, a is estimated with polynomial complexity depending on the number of sampled points, providing an estimate of the direction of variation of f. The authors of [26] generalizes this to the setup $f(x) = g(Ax)$ with $A \in \mathbb{R}^{d_e \times D}$, still with smoothness and sparsity assumptions. The latter is dropped in [68].

Random embeddings, projections of high-dimensional points onto low-dimensional subspaces, have been applied to global optimization, particularly for functions with low effective dimensionality [17, 18], where the effective dimensionality is usually unknown. The authors of [18] assumes the objective varies only along a known low-dimensional subspace and leverages random matrix theory to derive sharp success probabilities. In contrast, the paper [17] introduces the more general X-REGO framework, which requires no prior knowledge of the effective dimension and applies to both constrained and unconstrained problems. Its convergence analysis is based on conic integral geometry and yields almost sure global convergence under mild conditions. Some works also cover the global optimization of functions with low effective dimension in the constrained case, see [16]. See also the recent work [55] for a broader perspective on the global optimization of functions with low effective dimension.

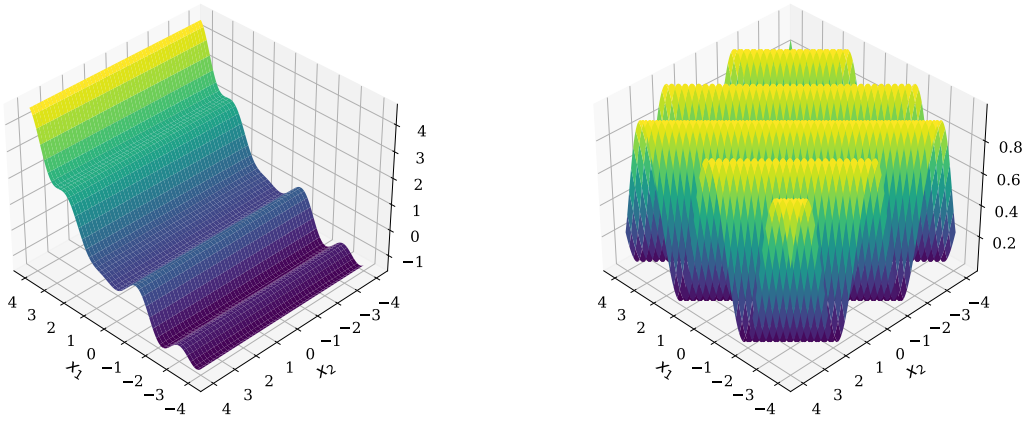
This thesis investigates a generalization of the low effective dimensionality assumption commonly employed in global optimization. The assumption is relaxed by considering functions that exhibit an approximate low effective dimension, in either a deterministic or stochastic sense. The first chapter focuses on stating and proving implications between assumptions used for characterizing the low effective dimensionality hypothesis. The second chapter focuses on deterministic perturbations, drawing on the theory of active subspaces and the generalization of theoretical results derived in the first part, to identify dominant directions. The third chapter addresses stochastic perturbations, using Riemannian optimization techniques to explore the underlying low-dimensional structure when facing noisy queries of the function as the only available information. This work stems from generalizing the work of [15], and will thus be based on some results presented in it.

All experiments in this thesis are conducted using Julia [9], a high-level programming language that combines execution speed with an expressive syntax. Julia's just-in-time (JIT) compilation and support for multiple dispatch make it well-suited for numerical optimization and the development of custom solvers. Its growing ecosystem and interoperability with Python, C, and Fortran further enhance its practicality for scientific computing.

Chapter 1

Existing Work

This chapter revisits foundational results concerning functions with *low effective dimension*, a structural property leveraged to reduce the complexity of high-dimensional optimization problems. A function is said to have *low effective dimension* if it exhibits significant variation only along a low-dimensional linear subspace of the input space. See Figure 1.1 for an illustration.



(a) (b)

Figure 1.1: Two functions with low effective dimension. (a) Levy function - Set $w_1 = 1 + \frac{\mathbf{x}_1 - 1}{4}$ and define $f_1 : \mathbb{R}^2 \to \mathbb{R} : (\mathbf{x_1}, \mathbf{x_2}) \to \sin^2(\pi w_1) + \left(\frac{\mathbf{x}_1 - 1}{4}\right)^2 \left[1 + \sin^2(2\pi w_1)\right]$. The effective subspace is spanned by $(1, 0)$ and the constant subspace is spanned by $(0, 1)$. (b) Define $f : \mathbb{R}^2 \to \mathbb{R} : (\mathbf{x}_1, \mathbf{x}_2) \to \sin^2(\mathbf{x}_1 - \mathbf{x}_2 - 0.5)$. The effective subspace is spanned by $(-1, 1)$ and the constant subspace is spanned by $(1, 1)$.

The discussion begins by precisely defining the low effective dimension property. Particular attention is given to the relationship between the function's gradient structure and the so-called *effective subspace*, along which the function varies. In practice, this subspace can be identified by analysing the spectral properties of a

matrix constructed from gradient information.

The purpose of this chapter is to make these assumptions explicit, clarify the links between different formulations of the low effective dimension property, and reformulate key results in a way that facilitates their extension to functions exhibiting *approximate* low effective dimension. The reformulated results serve as a foundation for the generalization pursued in the next chapter. We start with a definition of such functions, taken from [72, Section 3].

Definition 1.1 (Definition 1 in Wang et al, [72])**.** *A function $f : \mathbb{R}^D \to \mathbb{R}$ is said to have effective dimensionality d_e, with $d_e \leq D$, if*

- *there exists a linear subspace $\mathcal{S}$ of dimension d_e such that for all $x_\top \in \mathcal{S} \subset \mathbb{R}^D$ and $x_\perp \in \mathcal{S}^\perp \subset \mathbb{R}^D$, we have $f(x_\perp + x_\top) = f(x_\perp)$, where $\mathcal{S}^\perp$ denotes the orthogonal complement of $\mathcal{S}$;*
- *d_e is the smallest integer with this property.*

We call $\mathcal{S}$ the effective subspace of f and $\mathcal{S}^\perp$ the constant subspace of f.

In Figure 1.1, both functions have effective dimensionality $d_e = 1$. The effective direction does not necessarily coincide with a canonical axis. For functions $f : \mathbb{R}^D \to \mathbb{R}$ with *low effective dimension*, identifying the active subspace requires an analysis of the spectrum of the matrix

$$C = \int \nabla f(x) \nabla f(x)^T \rho(x) \mathrm{d}x = \mathbb{E}_{x\sim\rho}\left[\nabla f(x) \nabla f(x)^T\right], \tag{1.1}$$

where ∇f is the gradient of f and ρ is a probability density on $\mathbb{R}^D$. Note that $C \in \mathbb{R}^{D\times D}$. In practice, one uses Monte-Carlo sampling to approximate C,

$$\hat{C} = \frac{1}{M} \sum_{i=1}^{M} \nabla f(x^{(i)}) \nabla f(x^{(i)})^T. \tag{1.2}$$

When no first-order informations of f are available, one uses the following approximation

$$\nabla f(x) \approx g_h(x), \quad [g_h(x)]_i = \frac{f(x + he_i) - f(x)}{h} \quad \text{for } i = 1, ..., D, \tag{1.3}$$

where e_i is the i-th canonical vector of the D-dimensional Euclidean space.

In [15], it is shown that if f exhibits *low effective dimension*, the spectrum of the matrix C is related to the effective subspace of f. Specifically, under certain additional assumptions, the effective subspace $\mathcal{S}$ coincides with range (C). This chapter examines a set of assumptions that characterize functions with *low effective dimension*, with the aim of formally identifying the conditions under which this property holds.

1.1 Low effective dimension and active subspaces

This section presents results on functions with *low effective dimension* and active subspaces, which will be extended in Section 2.1. Although several of these results appear in [15], the purpose here is to clearly identify the elements that require generalization, as the relevant assumptions and their equivalence are not stated explicitly in that work. Particular emphasis is placed on the spectral properties of the matrix C.

While Definition 1.1 provides useful insight into the *low effective dimension* property, a more precise formulation is required to enable its practical exploitation in numerical algorithms, as well as its extension to functions with *approximate low effective dimension*. As a first formulation, consider Proposition 1.1, where the derivation, left to the reader in [15], is made explicit.

Proposition 1.1 (Reformulation of Assumption 2.3, [15])**.** *Assume that the function $f : \mathbb{R}^D \to \mathbb{R}$ has effective dimensionality d_e with effective subspace $\mathcal{S}$ and constant subspace $\mathcal{S}^\perp$ spanned by the orthonormal columns of the matrices $U \in \mathbb{R}^{D\times d_e}$ and $V \in \mathbb{R}^{D\times(D-d_e)}$, respectively. We write $x_\top = UU^T x$ and $x_\perp = VV^T x$, the unique Euclidean projections of any vector $x \in \mathbb{R}^D$ onto $\mathcal{S}$ and $\mathcal{S}^\perp$, respectively. One has for any $x \in \mathbb{R}^D$,*

$$f(x) = f(UU^T x). \tag{1.4}$$

Proof. Construct first $Q = \begin{pmatrix} U & V \end{pmatrix} \in \mathbb{R}^{D\times D}$. By construction, Q is an orthonormal matrix, as its columns are orthogonal with each other, are of unitary norm, and they span the whole $\mathbb{R}^D$. Hence $QQ^T = I_D$, with I_D the identity matrix of dimension $D \times D$. Note also that $QQ^T = UU^T + VV^T$. Then,

$$\begin{aligned} f(x) &= f(QQ^T x) = f(UU^T x + VV^T x) \\ &= f(x_\perp + x^\top) = f(x^\perp) = f(UU^T x). \end{aligned}$$

□

Proposition 1.1 characterizes the low effective dimension assumption in terms of the effective subspace spanned by the columns of U. For the purpose of generalization, it is helpful to introduce an assumption that explicitly reflects invariance along the subspace spanned by the columns of V. This perspective is motivated by [15, Lemma 2.7], which supports the following assumption for functions with *low effective dimension*.

Assumption 1.1 (No variation of the objective along a $(D-d_e)$-dimensional linear subspace)**.** *There exists $V \in \mathbb{R}^{D\times(D-d_e)}$ with orthonormal columns such that, for all $x \in \mathbb{R}^D$,*

$$\|V^T \nabla f(x)\| = 0.$$

This is equivalent to requiring that the projection of the gradient onto any basis vector of the constant subspace is zero. An alternative and intuitive formulation for functions that exhibit no variation in certain directions is to characterize this

invariance with respect to vectors lying in the subspace spanned by the columns of V.

Assumption 1.2 (Constant objective along a $(D-d_e)$-dimensional linear subspace)**.** *There exists $V \in \mathbb{R}^{D\times(D-d_e)}$ with orthonormal columns such that, for all $x \in \mathbb{R}^D$ and for all $u \in \mathbb{R}^{D-d_e}$,*

$$f(x) = f(x + Vu).$$

The vector u can be taken as $V^T y$, for $y \in \mathbb{R}^D$.

This implies that the function remains invariant under any linear combination of the vectors spanning the column space of V. The assumptions introduced above are related, as stated in the following proposition.

Proposition 1.2. *Let f be a continuously differentiable function with low effective dimension. Assumption 1.1 is equivalent to Assumption 1.2.*

Proof. **First implication.** Assumption 1.1 $\Rightarrow$ Assumption 1.2. Define

$$g(\tau) = f\left(x + \tau\left(x + Vu - x\right)\right) = f\left(x + \tau Vu\right).$$

By the fundamental theorem of calculus,

$$f(x + Vu) - f(x) = g(1) - g(0) = \int_0^1 g'(\tau)\mathrm{d}\tau.$$

One can compute

$$g'(\tau) = \frac{d}{d\tau} f(x + \tau Vu) = \nabla f(x + \tau Vu)^T (Vu).$$

Due to Assumption 1.1, it follows that

$$f(x + Vu) - f(x) = 0.$$

Second implication. Assumption 1.2 $\Rightarrow$ Assumption 1.1. Let $V \in \mathbb{R}^{D\times(D-d_e)}$ such that $V = \begin{pmatrix} v_1, & \ldots, & v_{D-d_e}\end{pmatrix}$, with V satisfying Assumption 1.2. Notice that

$$\left\|V^T \nabla f(x)\right\| = 0 \quad \Leftrightarrow \quad v_i^T \nabla f(x) = 0, \text{ for } i = 1, \ldots, D - d_e.$$

Consider an arbitrary $i \in \{1, \ldots, D - d_e\}$. Take $u = he_i$, with $h > 0$ and e_i the i-th canonical vector. By Assumption 1.2, and by the definition of the directional derivative

$$\mathrm{D}f(x)\left[v_i\right] = \lim_{h\to 0} \frac{f\left(x + hv_i\right) - f(x)}{h} = v_i^T \nabla f(x) = 0.$$

As it holds for an arbitrary $i \in \{1, \ldots, D-d_e\}$, we have shown the desired result. □

Consider now the *active subspaces* framework, as introduced in [22]. According to the following theorem, the effective subspace of a function with *low effective dimension* corresponds to the range of the matrix C.

Theorem 1.1 (Theorem 2.9 from [15]). *Let $f : \mathbb{R}^D \to \mathbb{R}$ be continuously differentiable, and let ρ have support over the whole $\mathbb{R}^D$. Let C be defined as in Equation (1.1). Then,* $\operatorname{rank}(C) = d_e$ *if and only if f has effective dimensionality d_e. In this case, the effective subspace of f is $\mathcal{S} = \operatorname{range}(C)$.*

Theorem 1.1 establishes an initial connection between the spectrum of C and the low effective dimension assumption. If $\operatorname{rank}(C) = d_e$, then C has exactly d_e positive eigenvalues. Each of these is associated with an eigenvector such that $\mathcal{S} = \operatorname{range}(C)$. Consequently, the remaining $D - d_e$ eigenvectors correspond to zero eigenvalues and span the constant subspace of f. The matrix U can therefore be formed by stacking the first d_e eigenvectors, and V by stacking the last $D - d_e$ eigenvectors.

Finally, it is useful to characterize the spectrum of C more precisely. In practice, C can be computed numerically, and its low-dimensional structure can be inferred through an eigenvalue decomposition. The following result from [22] formalizes this observation.

Lemma 1.1. *The matrix C as defined in Equation (1.1) is symmetric and positive semidefinite. Moreover, let (λ_i, v_i) be an eigenpair of C. The mean-squared directional derivative of f with respect to the eigenvector v_i is equal to the eigenvalue λ_i,*

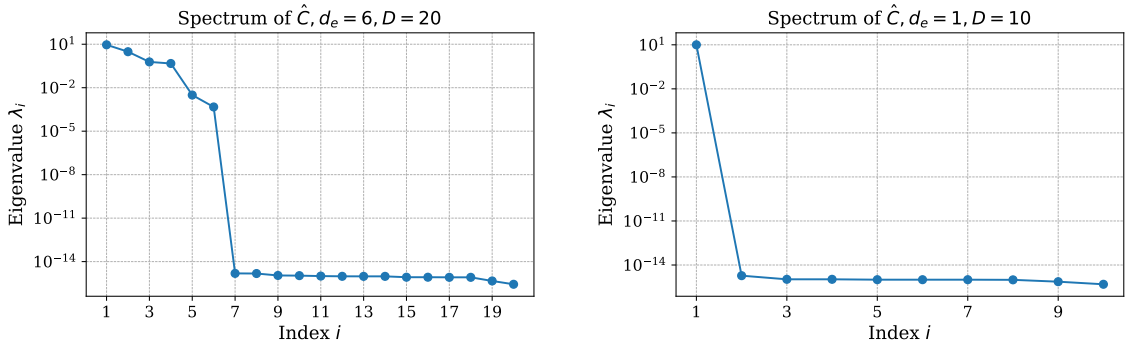
$$\lambda_i = v_i^T C v_i = \int \left(\nabla f(x)^T v_i\right)^2 \rho(x) \mathrm{d}x.$$

According to Lemma 1.1, if an eigenvalue of C is equal to zero, this indicates that the function exhibits no variation in the direction spanned by the corresponding eigenvector. Together, Lemma 1.1 and Assumption 1.1 motivate the following assumption.

Assumption 1.3 (Tail of the spectrum of C). *The eigenvalues of C are such that*

$$\lambda_i = 0, \quad i = d_e + 1, \ldots, D.$$

As an illustration of Assumption 1.3, see Figure 1.2.



(a) (b)

Figure 1.2: Analysis of the eigenvalues of $\hat{C}$ for functions with low effective dimension. (a) Hartman 6-dimensional function embedded in a 20-dimensional space. The effective dimension is 6. See Appendix A.2.7 for more details. The Monte-Carlo estimator $\hat{C}$ is constructed using $M = 100$ samples. (b) The function has effective dimension equal to one. $M = 10$. See Equation (A.1) for more details.

It turns out that Assumption 1.1 and Assumption 1.3 are equivalent. Before showing this equivalence, we state an intermediate result.

Proposition 1.3 (Theorem 1.2 of [60]). *Let C be defined as in Equation (1.1). Then,*

$$\min_{\substack{V \in \mathbb{R}^{D\times(D-d_e)} \\ V^T V = I_{D-d_e}}} \operatorname{trace}(V^T C V) = \sum_{i=d_e+1}^{D} \lambda_i$$

where λ_i, $i = d_e + 1, \ldots, D$ are the $D - d_e$ smallest eigenvalues of C, assuming $\lambda_1 \geq \cdots \geq \lambda_D$.

See Appendix C for a geometric illustration of this result.

Proposition 1.4. *Let f be a continuously differentiable function with low effective dimension. Assumption 1.1 is equivalent to Assumption 1.3.*

Proof. **First implication.** Assumption 1.1 $\Rightarrow$ Assumption 1.3. Note that

$$\lambda_i = 0, \quad i = d_e + 1, \ldots, D \quad \Leftrightarrow \quad \sum_{i=d_e+1}^{D} \lambda_i = 0,$$

since C is symmetric and positive semidefinite. Take $V \in \mathbb{R}^{D\times(D-d_e)}$ satisfying

Assumption 1.1. It follows that

$$\begin{aligned}
\operatorname{trace}\left(V^T C V\right) &= \operatorname{trace}\left(V^T \int \nabla f(x) \nabla f(x)^T \rho(x) \mathrm{d}x V\right) \\
&= \operatorname{trace}\left(\int V^T \nabla f(x) \nabla f(x)^T V \rho(x) \mathrm{d}x\right) \\
&= \operatorname{trace}\left(\int V^T \nabla f(x) \left(V^T \nabla f(x)\right)^T \rho(x) \mathrm{d}x\right) \\
&= \int \left(V^T \nabla f(x)\right)^T \left(V^T \nabla f(x)\right) \rho(x) \mathrm{d}x \\
&= \int \left\|V^T \nabla f(x)\right\|_2^2 \rho(x) \mathrm{d}x \\
&= 0.
\end{aligned}$$

Applying the reasoning above, as well as Proposition 1.3, one gets,

$$0 \leq \sum_{i=d_e+1}^{D} \lambda_i = \min_{\substack{Y \in \mathbb{R}^{D \times (D-d_e)} \\ Y^T Y = I_{D-d_e}}} \operatorname{tr}\left(Y^T C Y\right) \leq \operatorname{tr}\left(V^T C V\right) \leq 0,$$

which implies that

$$\sum_{i=d_e+1}^{D} \lambda_i = 0.$$

Second implication. Assumption 1.3 $\Rightarrow$ Assumption 1.1. Consider Lemma 1.1. The integrand is positive semidefinite, for each $i \in \{1, \ldots, D\}$. Then, if $\lambda_i = 0$ for some $i \in \{d_e + 1, \ldots, D\}$, one has $v_i^T \nabla f(x) = 0$, with (λ_i, v_i) an eigenpair of C. Then V in Assumption 1.1 can be constructed as the aggregation of the $D - d_e$ last eigenvectors.

□

Based on the above proposition, one can conclude that Assumptions 1.1 to 1.3 are equivalent.

Chapter 2

Deterministic perturbations

This chapter investigates functions that, although not strictly low-dimensional, vary predominantly along a low-dimensional subspace, while the orthogonal complement exhibits only limited variation, subject to a bounded deterministic perturbation. Such functions are said to have *approximately low effective dimension*. The analysis examines how this structural property can be exploited, using techniques such as active subspaces [22], to improve the scalability of global optimization algorithms. An illustration of this concept is provided in Figure 2.1. The approximate effective direction is spanned by $(-1, 1)$ and the approximate constant direction is spanned by $(1, 1)$. Each valley of the function, represented by coloured lines on the right plot, corresponds to a set of approximate minimizers. Thus, starting from an initial guess, optimizing only on the approximate effective direction, one can get an approximate global minimizer up to the magnitude of the perturbation, here $[-0.05, 0.05]$.

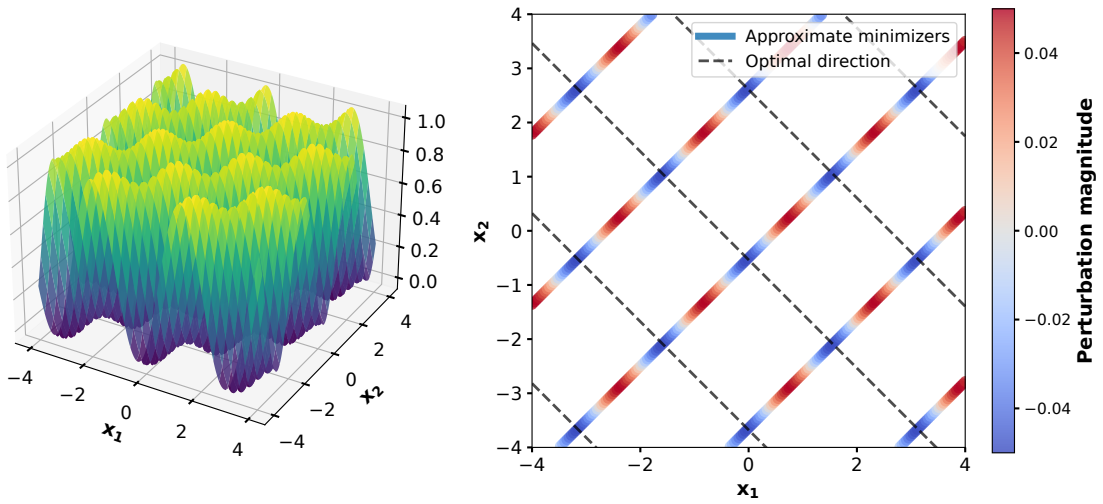


Figure 2.1: $f : \mathbb{R}^2 \to \mathbb{R} : x \to \sin^2(x_1 - x_2 - 0.5) + 0.05 \sin(x_1 + x_2 - 0.5)$.

The first section of this chapter focuses on formulating assumptions for the theoretical characterization of such functions. The second section describes the experimental setup, including the construction of the test set and the development of algorithms

that extend those proposed in [15]. The third section presents numerical results along with a discussion of the approach's limitations.

2.1 Generalization to approximate low effective dimension

This section aims to generalize the assumptions derived in Section 1.1. Under the *approximate low effective dimension* assumption, the equality $f(x_\perp + x^\top) = f(x_\perp)$ no longer holds. Consequently, deriving a definition analogous to Definition 1.1 is of limited use, as a computational criterion is required. It is therefore necessary to provide assumptions generalizing Assumptions 1.1 to 1.3. This motivates the work carried out in Section 1.1 to state explicit assumptions. Assumption 1.1 can be generalized as follows,

Assumption 2.1 (Bounded variations of the objective along a $(D-d_e)$-dimensional linear subspace)**.** *There exists $V \in \mathbb{R}^{D\times(D-d_e)}$ with orthonormal columns and $M_1 > 0$ (small) such that, for all $x \in \mathbb{R}^D$,*

$$\left\|V^T \nabla f(x)\right\| \leq M_1$$

As an illustration, consider the function

$$f : \mathbb{R}^2 \to \mathbb{R}, \quad x \mapsto \sin(x_1) + 10^{-5}\sin(x_2).$$

Clearly, the approximate constant subspace is spanned by $(0,1)$. Setting $V = (0,1)^T$,

$$\nabla f(x) = \begin{pmatrix} \cos(x_1) \\ 10^{-5}\cos(x_2) \end{pmatrix} \quad \Rightarrow \quad \left|V^T \nabla f(x)\right| \leq 10^{-5}.$$

The interpretation of Assumption 2.1 relies on the assumption that the projection of the gradient onto the approximate invariant subspace of f has small magnitude. As an attempt to generalize Assumption 1.2, a *Lipschitz-like* assumption can be stated.

Assumption 2.2 (Lipschitzness in the almost constant direction)**.** *There exists $V \in \mathbb{R}^{D\times(D-d_e)}$ with orthonormal columns and $M_2 > 0$ (small) such that for all $x \in \mathbb{R}^D$ and for all $u \in \mathbb{R}^{D-d_e}$,*

$$|f(x) - f(x+Vu)| \leq M_2 \|u\|.$$

Similarly to Section 1.1, one can show that Assumption 2.1 implies Assumption 2.2.

Proposition 2.1. *Let f be a continuously differentiable function satisfying Assumption 2.1. Then Assumption 2.2 is satisfied.*

Proof. By the use of the mean value theorem, one can find $\xi \in (0,1)$ such that

$$f(x+Vu) = f(x) + \nabla f\left((1-\xi)x + \xi(x+Vu)\right)^T (x + Vu - x).$$

Leading to

$$\begin{aligned} |f(x+Vu) - f(x)| &= \left|\nabla f(x+\xi Vu)^T (Vu)\right| \\ &= \left|\langle u, V^T \nabla f(x+\xi Vu)\rangle\right| \\ &\leq \|u\| \left\|V^T \nabla f(x+Vu)\right\| \\ &\leq M_1 \|u\| . \end{aligned}$$

□

It follows from the proof of Proposition 2.1 that $M_1 = M_2$.

Consider again the *active subspaces* framework. An extension of Assumption 1.3 can be formulated to account for small variations in the spectrum.

Assumption 2.3 (Bound on the tail of the spectrum of C)**.** *The eigenvalues $\lambda_1 \geq \cdots \geq \lambda_D \geq 0$ of the matrix C defined in Equation (1.1) satisfy*

$$\sum_{i=d_e+1}^{D} \lambda_i \leq M_3$$

We show that Assumption 2.1 implies Assumption 2.3.

Proposition 2.2. *Assumption 2.1 implies Assumption 2.3.*

Proof. Assume that V satisfies Assumption 2.1. Then,

$$\begin{aligned} \text{trace}\left(V^T C V\right) &= \text{trace}\left(V^T \int \nabla f(x) \nabla f(x)^T \rho(x) \mathrm{d}x V\right) \\ &= \text{trace}\left(\int V^T \nabla f(x) \nabla f(x)^T V \rho(x) \mathrm{d}x\right) \\ &= \text{trace}\left(\int V^T \nabla f(x) \left(V^T \nabla f(x)\right)^T \rho(x) \mathrm{d}x\right) \\ &= \int \left(V^T \nabla f(x)\right)^T \left(V^T \nabla f(x)\right) \rho(x) \mathrm{d}x \\ &= \int \left\|V^T \nabla f(x)\right\|_2^2 \rho(x) \mathrm{d}x \\ &\leq M_1^2. \end{aligned}$$

Putting together the above reasoning and Proposition 1.3,

$$\sum_{i=d_e+1}^{D} \lambda_i = \min_{\substack{Y \in \mathbb{R}^{D\times(D-d_e)} \\ Y^T Y = I_{D-d_e}}} \text{trace}\left(Y^T C Y\right) \leq \text{trace}\left(V^T C V\right) \leq M_1^2$$

□

For functions with *low effective dimension*, the matrix V is defined as the collection of the last $D-d_e$ eigenvectors of C, corresponding to the zero eigenvalues. In the case of functions with *approximate low effective dimension*, if V, constructed from the last $D-d_e$ eigenvectors, satisfies Assumption 2.1, then Assumption 2.3 also holds. This implies that the function exhibits only limited variation along the directions spanned by V. This insight motivates a numerical strategy for global optimization: the matrix C is computed as defined in Equation (1.1), and its spectrum is analyzed to identify the dominant eigenspace associated with directions of significant variation. Since the eigenvalues reflect the magnitude of directional derivatives along their corresponding eigenvectors, the spectral structure provides a direct basis for this optimization approach.

2.2 Experimental setup

This section describes the algorithms introduced in [15] generalized to functions with *approximate low effective dimension*. The construction of the testing set is presented first.

2.2.1 Testing set

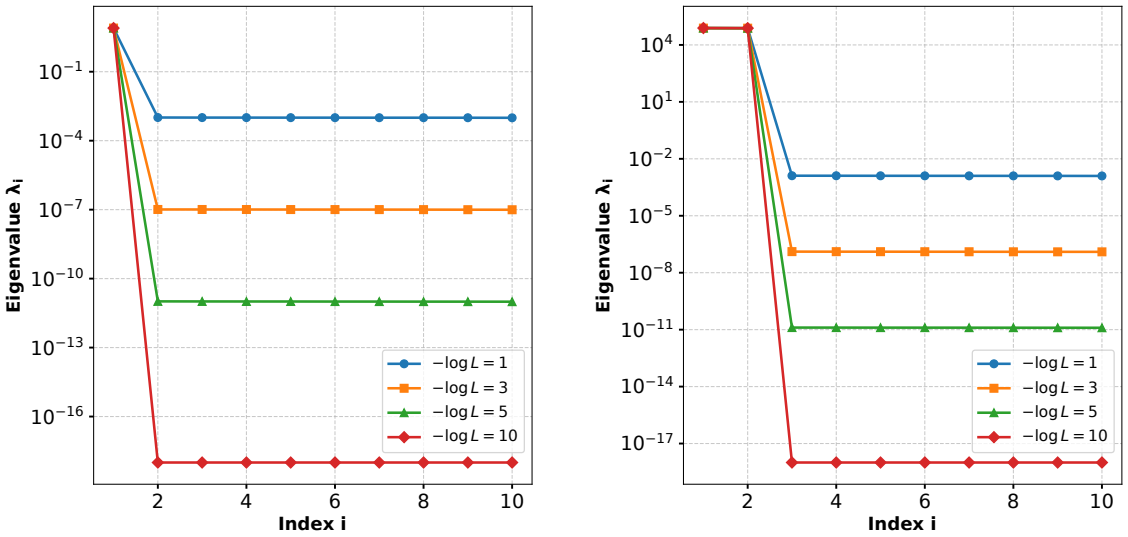
To construct a relevant set of functions for optimization, several benchmark functions from [64] are employed. This library includes many functions commonly used in global optimization as validation benchmarks; see, for example, [49, 40, 48], as well as [15]. These functions are specifically designed to be challenging to optimize, often featuring local minima and narrow valleys, making them suitable for assessing algorithmic performance. A complete list of the test functions used in this master thesis is provided in Appendix A.2.

To construct functions with approximate low effective dimension, the procedure begins with the approach described in [15, Appendix B], which is recalled in Appendix A.1.1. The "dummy directions" are modified to introduce low-magnitude variation, as detailed in Appendix A.1.2. Given a function $h : \mathbb{R}^{d_e} \to \mathbb{R}$, the resulting function is defined as

$$f : [-1,\, 1]^D \to \mathbb{R} : x \to h(\bar{x}) + \sum_{i=d_e+1}^{D} \frac{L}{D-d_e} \sin\left(\omega_i x_i + \varphi_i\right). \tag{2.1}$$

Here, $\bar{x}$ denotes the restriction of the D-dimensional vector x to its first d_e components. The constants $L > 0$, ω_i, and φ_i are appropriately chosen for each $i \in \{d_e + 1, \ldots, D\}$. A random orthogonal matrix Q is generated to rotate the basis, thereby avoiding alignment with the canonical axes. The resulting rotated function takes the form $f(Q^T x)$. By construction, the first d_e columns of Q span the *effective subspace*, while the remaining $D - d_e$ columns span the *invariant subspace*.

Typically, the parameter L is chosen to be small relative to the variation in the other directions, in order to ensure that the function exhibits an approximate low effective dimension. The function is rescaled so that its global minimum lies within the domain $[-1, 1]$. Figure 2.2 provides an illustration of how the spectrum evolves with respect to the magnitude of the perturbation. In both cases, it can be verified that the sum of the remaining $D - d_e$ eigenvalues is less than L^2 for each perturbation magnitude L, in accordance with the prediction of Assumption 2.3.



(a) (b)

Figure 2.2: Spectrum of two functions with approximate low effective dimension. (a) Adaptation of $f : \mathbb{R}^2 \to \mathbb{R} : x \to \sin^2(x_1 - x_2 - 0.5)$ to 10 dimensions with approximate low effective dimensions. (b) Styblinski-Tang function in 10 dimensions. See Appendix A for more details. $M = 2000$. The plot displays $\tilde{\lambda}_i = \lambda_i + 10^{-18}$, to avoid numerical issues when computing the logarithmic value. The eigenvalues associated with the approximate invariant directions of magnitude $-\log L = 10$ were less than 10^{-20}, which is what is expected from Assumption 2.3.

Finally, note that the optimal value of the new functions can be obtained by subtracting the magnitude of the perturbation L to the original optimal value.

2.2.2 Algorithms

Since the focus is on functions with *approximate* low effective dimension, the algorithms proposed in [15] must be refined and adapted to account for this relaxed structural assumption. For consistency, the original algorithms to be adapted are recalled in Appendix B. This section presents the necessary modifications for use with functions of approximate low effective dimension. The procedure for estimating the matrix C is outlined below.

Algorithm 1 Estimating C, taken from [22]

1: Generate M samples $\left\{x_S^j\right\}$ independently from the probability density function ρ. Set $h > 0$, a discretization parameter.
2: Compute $g_h\left(x_S^j\right)$ for $j = 1, \ldots, M$, as a finite-difference approximation of $\nabla f\left(x_S^j\right)$ for $j = 1, \ldots, M$.
3: Return the estimate $\hat{C}$

$$\hat{C} \coloneqq \frac{1}{M} \sum_{j=1}^{M} g_h\left(x_S^j\right) g_h\left(x_S^j\right)^T.$$

The first algorithm considered involves solving the optimization problem over the full-dimensional domain; see Algorithm 2. This algorithm serves as a baseline for comparison and will be referred to as **full domain** throughout the remainder of this document.

Algorithm 2 Full Domain

1: Use a global solver to solve

$$\min_{x \in \mathbb{R}^D} f(x)$$

Assume that the approximate effective dimensionality d_e is known a priori. Based on the discussion at the end of Section 2.1, a natural extension of Algorithm 11 is to consider the leading eigenspectrum of the matrix $\hat{C}$. Since only d_e dimensions are required, it is reasonable to take $M = d_e$. The corresponding algorithm, referred to as **approx-ASM**, is defined in Algorithm 3. As shown in [15], choosing $M = d_e$ allows for the recovery of the effective subspace with high probability. It is expected that in the context of *approximate* low effective dimension, this approach can still yield meaningful results.

Algorithm 3 approx-ASM (known d_e)

1: Initialize $p \in \mathbb{R}^D$. The approximate effective dimension d_e is known.
2: Apply Algorithm 1 with $M = d_e$ samples to compute $\hat{C}$.
3: Compute a basis A of the range of $\hat{C}$. The matrix A is such that $A \in \mathbb{R}^{D \times d_e}$.
4: Calculate y by solving approximately,

$$\min_{y \in \mathbb{R}^{d_e}} f\left(Ay + p\right).$$

5: Construct the solution estimate $x = Ay + p$.

Now consider the case where d_e is unknown. In this setting, a method is required to determine an appropriate cutoff in the spectrum of C, in order to select the

eigenvectors to be included in the matrix A. Since the variation of f along each direction is reflected by the corresponding eigenvalue, the selection strategy will primarily rely on the eigenvalues of $\hat{C}$. If the perturbation magnitude L is known, a natural approach is to apply the result of Proposition 2.2. As an example, consider a function constructed as in Equation (2.1),

$$f : [-1,\, 1]^D \to \mathbb{R} : h\left(x\left[1 : d_e\right]\right) + \sum_{i=d_e+1}^{D} \frac{L}{D-d_e} \sin(x_i).$$

The approximate effective subspace is spanned by

$$U = \begin{pmatrix} I_{d_e} \\ 0_{(D-d_e)\times d_e} \end{pmatrix}$$

Note that

$$\nabla f\left(x\right) = \begin{pmatrix} \frac{\partial}{\partial x_1} h\left(x\left[1 : d_e\right]\right) \\ \vdots \\ \frac{\partial}{\partial x_{d_e}} h\left(x\left[1 : d_e\right]\right) \\ L\cos\left(x_{d_e+1}\right)/(D-d_e) \\ \vdots \\ L\cos\left(x_D\right)/(D-d_e) \end{pmatrix}.$$

To find a $V \in \mathbb{R}^{D\times D-d_e}$ satisfying Assumption 2.1 such that $V^TV = I_{D-d_e}$, one can observe the gradient of f and set

$$V = \begin{pmatrix} 0_{d_e\times(D-d_e)} \\ I_{(D-d_e)\times(D-d_e)} \end{pmatrix}.$$

Observe that

$$\begin{aligned} \left\|V^T\nabla f\left(x\right)\right\| &= \sqrt{\left(L\cos\left(x_{d_e+1}\right)/\left(D-d_e\right)\right)^2 + \cdots + \left(L\cos\left(x_D\right)/\left(D-d_e\right)\right)^2} \\ &\leq \frac{L}{D-d_e} \sum_{i=d_e+1}^{D} \left|\cos\left(x_i\right)\right| \\ &\leq L. \end{aligned}$$

Moreover, this reasoning holds for any orthogonal basis of the approximate invariant subspace. Indeed, if $Q \in \mathbb{R}^{(D-d_e)\times(D-d_e)}$ and $Q^TQ = QQ^T = I_{D-d_e}$,

$$\left\|\left(VQ\right)^T\nabla f(x)\right\| = \left\|Q^TV^T\nabla f(x)\right\| \leq \left\|V^T\nabla f(x)\right\| \leq L.$$

This implies that the constant M_1 in Assumption 2.1 is known, and, by construction, the assumption holds up to an orthogonal rotation of the canonical basis. According to Proposition 2.2, this further implies that Assumption 2.3 is satisfied. As a result, the eigenvalues of C satisfy

$$\sum_{i=d_e+1}^{D} \lambda_i \leq L^2.$$

Note that if $L \in [0, 1]$, then $L^2 \leq L$. Assume the eigenvalues are ordered as $\lambda_1 \geq \cdots \geq \lambda_D \geq 0$. In this case, any eigenvalue greater than L cannot be associated with an eigenvector spanning the approximate invariant subspace. An illustration is provided in Figure 2.2, where for each value of L, the eigenvalues corresponding to the approximate invariant subspace remain below both L and L^2, up to a numerical tolerance introduced for readability. In this figure, for every considered value of L, exactly d_e eigenvalues exceed the threshold L (and L^2), while the remaining $D - d_e$ eigenvalues fall below it. Assuming $d_e \ll D$, it is reasonable to take $M = D/2$ so that $M \gg d_e$. The value L can then serve as a threshold for truncating the spectrum of $\hat{C}$. When $D = 10$, choosing $M = 10$ ensures coverage of the effective dimensionality d_e.

The procedure used to clip the spectrum and estimate the effective dimensionality d_e is detailed in Algorithm 4. To account for numerical instabilities, the eigenvalue threshold is set to $L + L/10$.

Algorithm 4 First Clipping Strategy (known L)

Input: The matrix $\hat{C}$, the perturbation magnitude L.
Set $\hat{d}_e = 0$.
Set $\hat{A} = [\ ]$.
for $i = 1, \ldots, M$ **do**
  Compute the eigenpair (λ_i, v_i).
  if $\lambda_i \geq L + L/10$ **then**
    $\hat{A} \leftarrow \left[\hat{A} \mid v_i\right]$
    $\hat{d}_e \leftarrow \hat{d}_e + 1$
  else
    break.
  end if
end for
Returns: $\hat{A}$, $\hat{d}_e$.

In Algorithm 4, it is not necessary to compute all eigenpairs of $\hat{C}$. A more efficient approach, particularly for large D, is to apply the Inverse Power method followed by deflation or any block generalization, as described in [58, Section 4.1]. This motivates the use of the algorithm presented in Algorithm 5, referred to as **clip-ASM**.

Algorithm 5 clip-ASM (known L)

1: Initialize $p \in \mathbb{R}^D$. The perturbation magnitude L is known.
2: Apply Algorithm 1 with $M = D/2$ samples (or $M = 10$) to compute $\hat{C}$.
3: Compute an approximation of the leading eigenspace of $\hat{C}$ using Algorithm 4. Call it $\hat{A}$, and let $\hat{d}_e$ be its dimension.
4: Calculate y by solving approximately,

$$\min_{y \in \mathbb{R}^{\hat{d}_e}} f\left(\hat{A}y + p\right)$$

5: Construct the solution estimate $x = \hat{A}y + p$.

If the perturbation magnitude is unknown, the spectrum of functions with approximate low effective dimension typically exhibits a noticeable drop at the transition between the dominant and invariant subspaces. This behavior, illustrated in Figure 2.2, can be exploited as a heuristic for identifying the approximate effective dimension. Based on this observation, a second clipping strategy is introduced in Algorithm 6, which identifies the effective subspace by locating a significant spectral gap. It is set to 100, as an heuristic choice.

Algorithm 6 Second Clipping Strategy (Unknown L)

Input: Sorted eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_M$ and corresponding eigenvectors $v_1, v_2, \ldots, v_M$.
Set $\hat{d}_e = 1$.
Set $\hat{A} = [\, v_1 \,]$.
for $i = 1, \ldots, M-1$ **do**
  $\lambda_{\text{current}} \leftarrow \lambda_i$ and $\lambda_{\text{next}} \leftarrow \lambda_{i+1}$
  if $\lambda_{\text{current}}/\lambda_{\text{next}} < 10^2$ **then**
    $\hat{A} \leftarrow \left[\hat{A} \mid v_{i+1}\right]$, and $\hat{d}_e \leftarrow \hat{d}_e + 1$
  else
    break.
  end if
end for
Returns: $\hat{A}$, $\hat{d}_e$.

2.2.3 Global Solver

In Algorithms 2, 3 and 5, a global solver is employed. Specifically, KNITRO [14] is used, a high-performance solver for large-scale nonlinear optimization that supports both convex and nonconvex problems. The termination criterion is set to its default value, and the multistart option is enabled with 200 starting points. All other parameters are left at their default settings. Unless stated otherwise, these configurations remain unchanged throughout the chapter.

2.3 Numerical results

This section is dedicated to the comparison of the four algorithms presented in Section 2.2.2. Their naming is recalled here:

- **full domain:** Solving the optimization problem on the full domain, as described in Algorithm 2.
- **approx-ASM:** The approximate effective dimensionality d_e is known, and Algorithm 3 is applied.
- **clip-ASM:** The approximate effective dimensionality d_e is unknown, and the magnitude of the approximate constant dimension L is known. It is Algorithm 5 with clipping procedure described in Algorithm 4.
- **clip-ASM (2):** The approximate effective dimensionality d_e and L are unknown. It is Algorithm 5 with clipping procedure described in Algorithm 6.

To address the challenge of benchmarking optimization algorithms across a large collection of test problems, this work adopts *performance profiles*, as introduced by Dolan and Moré [24]. As the test set size increases, presenting detailed results for each algorithm-function pair becomes impractical, often leading to overly lengthy or inconsistent presentations. Aggregated metrics such as averages can be misleading, particularly in the presence of outliers or failed runs, and typically require manual filtering. Performance profiles overcome these limitations by representing the cumulative distribution function of a chosen performance metric (such as runtime or number of function evaluations) across the entire test set. This provides a robust and compact summary that enables consistent and interpretable comparisons of algorithmic performance.

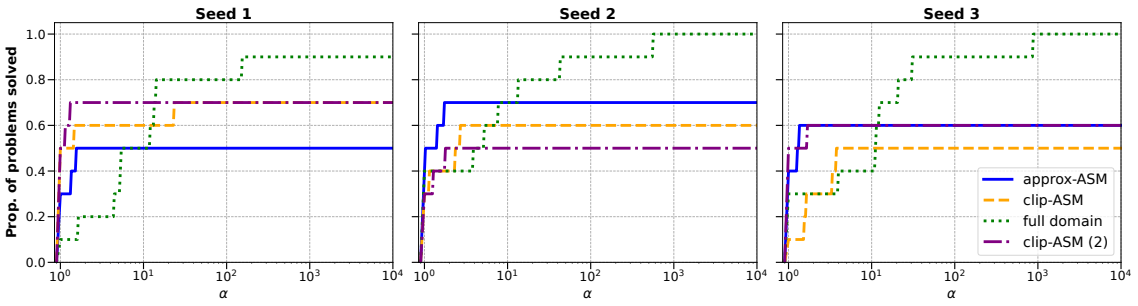
This thesis adopts the performance metric introduced in [15]. Let $\mathcal{F}$ denote the test set, as defined in Section 2.2.1 and Appendix A.2, and let $\mathcal{S}$ represent the set of algorithms described in Section 2.2.2 and summarized in the preceding bullet points. For each function $f \in \mathcal{F}$ and algorithm $s \in \mathcal{S}$, let $N_f(s)$ denote the number of function evaluations required by algorithm s to converge. The convergence threshold is set to $\epsilon = 10^{-3}$; that is, $N_f(s)$ is assigned the value ∞ if algorithm s does not converge to an ϵ-minimizer of f under the fixed solver settings detailed in Section 2.2.3. The performance probability is thus defined as

$$\pi_s(\alpha) = \frac{|\{f \in \mathcal{F} : N_f(s) \leq \alpha \min_{s \in \mathcal{S}} N_f(s)\}|}{|\mathcal{F}|}, \quad \alpha \geq 1,$$

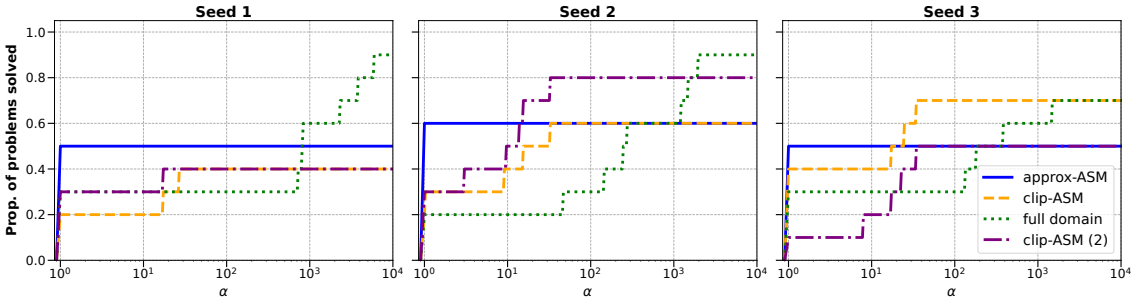
where the operator $|\cdot|$ outputs the cardinality of a given set.

Experiments are conducted at two levels of perturbation magnitude, $L \in \{10^{-4}, 10^{-6}\}$, and two ambient dimensions, $D \in \{10, 200\}$. Each configuration is evaluated under three random seeds, accounting for the randomness introduced both by KNITRO's initialization and by the sampling procedure used to construct the matrix $\hat{C}$. The convergence threshold is set to $\epsilon = 10^{-3}$, justified by the fact that the maximum

deviation between the optimal value obtained by optimizing over the approximate effective subspace and the true global minimum is bounded by the perturbation magnitude. Assessing convergence using a threshold smaller than the perturbation would be misleading, as the discrepancy would then depend solely on the initialization. We present in Figure 2.3 the results for $L = 10^{-4}$.



(a) $D = 10$



(b) $D = 200$

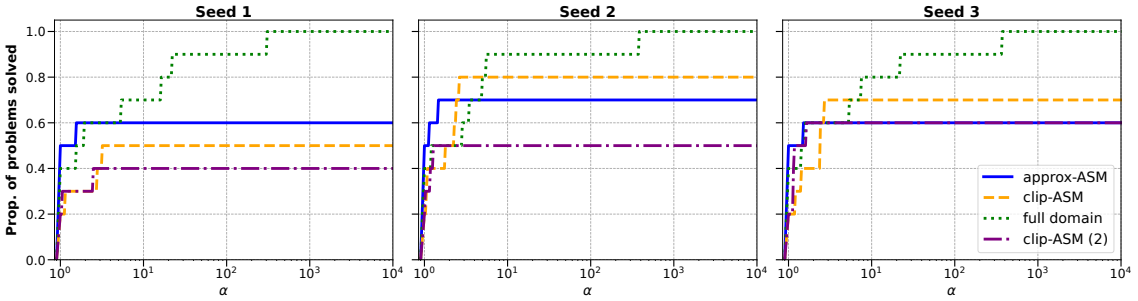
Figure 2.3: Performance profiles for $L = 10^{-4}$ with varying problem dimensions.

In the case $D = 10$, shown in Figure 2.3a, the three algorithms operating on a low-dimensional subspace exhibit comparable performance, with only minor variations across different seeds. The solver applied over the **full domain** generally outperforms the subspace-based methods within a factor of 10 of the optimal budget, and ultimately solves all problems within 10^4 times the best observed performance. In this low-dimensional setting, the benefits of dimensionality reduction are not clearly observable. Additionally, the low-dimensional algorithms solve approximately 60% of the problems in the test set, whereas the **full-domain** solver successfully addresses all problems under the specified KNITRO settings.

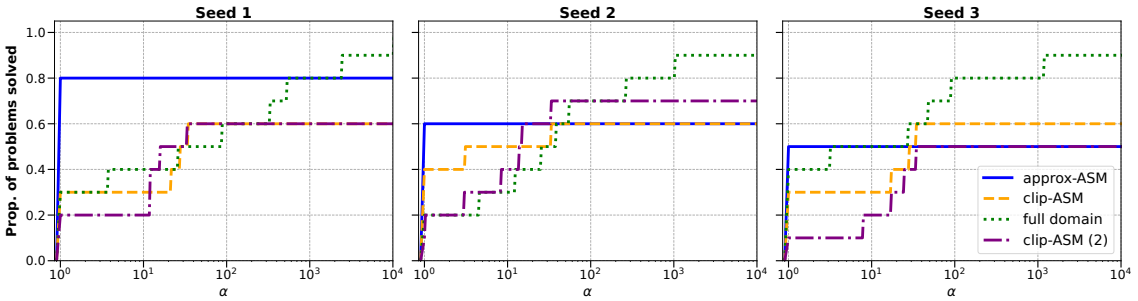
The case $D = 200$, presented in Figure 2.3b, shows that solving the problem over the full domain tends to offer greater robustness (i.e. more problems are solved) across the three seeds, though this comes at the expense of a higher computational budget. It takes a higher ratio α for **full domain** to achieve a good proportion of problem solved, around 10^3. For the second and third seeds, the methods **clip-ASM (2)** and **clip-ASM**, respectively, achieve a similar proportion of solved problems compared to the full-domain approach, but with improved efficiency. The method

approx-ASM also demonstrates a degree of robustness, consistently solving between 50% and 60% of the problems, with performance largely unaffected by the value of α. This is due to the fact that the problems solved by **approx-ASM** are either successfully optimized or not solved at all. It is also worth noting that, in this high-dimensional setting, the final proportion of problems solved by the **full-domain** method is lower than in the lower-dimensional case $D = 10$. The effects of dimensionality reduction are thus observable for $D = 200$.

The results for the case where $L = 10^{-6}$ is presented in Figure 2.4



(a) $D = 10$



(b) $D = 200$

Figure 2.4: Performance profiles for $L = 10^{-6}$ with varying problem dimensions.

For the case $D = 10$, shown in Figure 2.4a, the **full domain** approach solves a greater number of problems within a factor of 10 of the optimal performance. The method **approx-ASM** displays consistent behavior across the three random seeds. The strategy **clip-ASM** generally outperforms its variant **clip-ASM (2)** in terms of efficiency. Overall, the results and the corresponding analysis are consistent with those observed in Figure 2.3a.

In the case $D = 200$, shown in Figure 2.4b, the **full-domain** optimization is less efficient than **approx-ASM**, although it ultimately solves a slightly larger number of problems. The methods **clip-ASM** and **clip-ASM (2)** are competitive with **approx-ASM** in terms of the number of problems solved, but generally require more function evaluations. As observed previously, **approx-ASM** follows a distinct pattern: when it succeeds, it typically provides the best solution among all algo-

rithms; otherwise, the problem remains unsolved. Compared to its counterpart at $L = 10^{-4}$, **approx-ASM** demonstrates slightly improved efficiency in terms of the number of problems successfully solved.

When comparing the two perturbation magnitudes $L \in \{10^{-4}, 10^{-6}\}$ for $D = 10$, the performance of the clipping-based algorithms remains similar. This is due to the fact that, for small ambient dimensions, constructing $\hat{C}$ with $M = 10$ or $M = d_e$ using finite differences does not result in a significant difference in the number of function evaluations. A similar observation holds for the case $D = 200$, where the overall performance across both magnitudes remains comparable.

The three low-dimensional algorithms generally exhibit comparable performance, with variations as discussed above. However, the lack of prior knowledge of d_e results in increased computational cost, as additional queries are required to construct $\hat{C}$. In some cases, the clipping-based strategies outperform the **approx-ASM** approach in terms of the proportion of problems solved. Since these algorithms estimate the effective dimension d_e, it is natural to evaluate the quality of these estimations across the three random seeds, to investigate the robustness of the clipping approach, despite the greater number of queries required.

To this end, a reporting format similar to that of [15, Table 2, Section 5.2] is adopted. When all three seeds yield the correct estimate, a single number is reported. If the estimates differ across seeds, all three values are shown. The true dimension is indicated in bold. When none of the seeds correctly estimates d_e, the true value (in bold) is still reported alongside the incorrect estimates. The results for $L = 10^{-4}$ are presented in Table 2.1.

	Clip (1)		Clip (2)	
	$D = 10$	$D = 200$	$D = 10$	$D = 200$
Squared Sine	**1**	**1**	**1**	**1**
Styblinski-Tang 8D	**8**	**8**	**8**	**8**
Styblinski-Tang 2D	**2**	**2**	**2**	**2**
Beale	(3, **2**, 4)	(5,5,5) **2**	**2**	**2**
Hartman 3D	**3**	**3**	**3**	**3**
Hartman 6D	**6**	**6**	**6**	**6**
Trid	**5**	**5**	**5**	**5**
Goldstein-Price	(5,4,5) **2**	(55,5,42) **2**	(1, **2**, **2**)	**2**
Levy	**6**	**6**	**6**	**6**
Rosenbrock	**7**	**7**	**7**	**7**

Table 2.1: Estimation of d_e for the two clipping strategies. $L = 10^{-4}$.

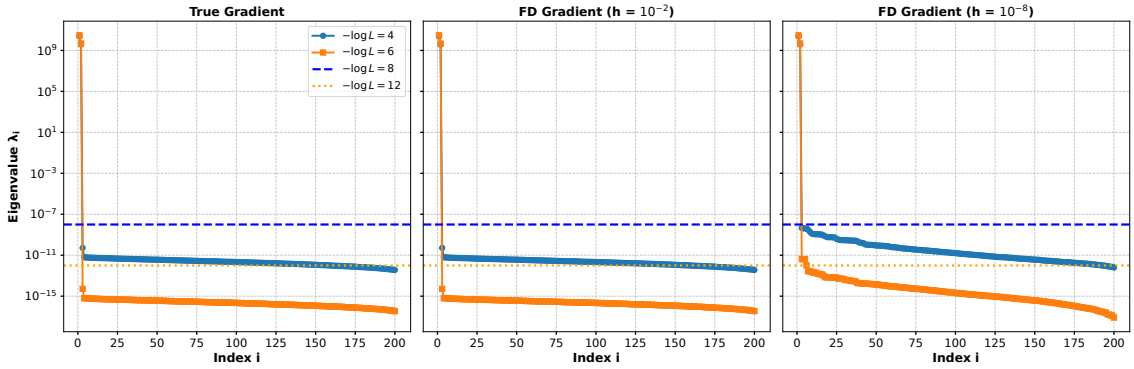
The results for $L = 10^{-6}$ are presented on Table 2.2.

	Clip (1)		Clip (2)	
	$D = 10$	$D = 200$	$D = 10$	$D = 200$
Squared Sine	**1**	**1**	**1**	**1**
Styblinski-Tang 8D	**8**	**8**	**8**	**8**
Styblinski-Tang 2D	**2**	**2**	**2**	**2**
Beale	(5,3,4) **2**	(35,28,12) **2**	**2**	**2**
Hartman 3D	**3**	**3**	**3**	**3**
Hartman 6D	**6**	**6**	**6**	**6**
Trid	**5**	**5**	**5**	**5**
Goldstein-Price	(7,8,7) **2**	(88,96,109) **2**	**2**	**2**
Levy	**6**	**6**	**6**	**6**
Rosenbrock	(**7**,8,**7**)	(8,8,8) **7**	**7**	**7**

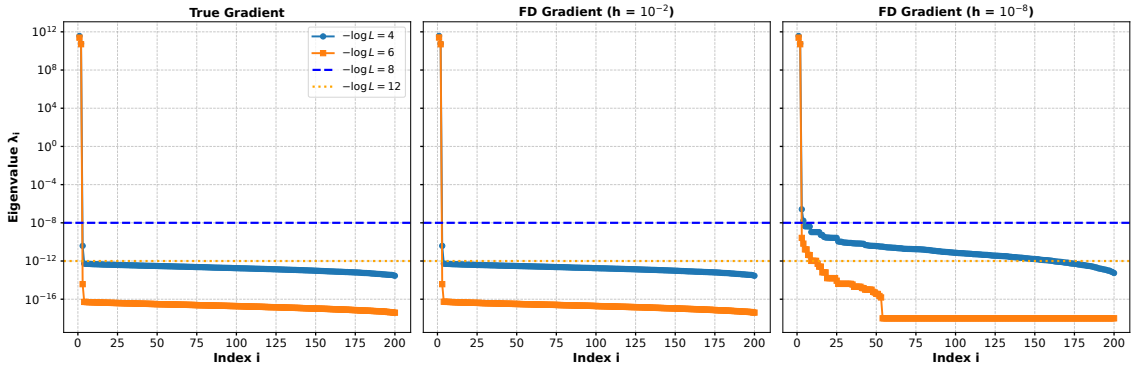
Table 2.2: Estimation of d_e for the two clipping strategies. $L = 10^{-6}$.

It is evident that the second strategy, **clip-ASM (2)**, yields more accurate estimates of d_e than its original counterpart. This is particularly noticeable for the Goldstein-Price and Beale functions, where the improvement holds consistently across both $D \in \{10, 200\}$ and $L \in \{10^{-4}, 10^{-6}\}$. These observations suggest that, while Assumption 2.1 remains valid, Assumption 2.3 may no longer be satisfied in these cases.

This overestimation error may arise from two factors. The first is related to the use of finite differences, rather than exact gradients, in constructing $\hat{C}$. If the step size h is too small, numerical errors can distort the spectrum of $\hat{C}$, as illustrated in Figure 2.5. Specifically, decreasing h can result in a spectrum that no longer satisfies Assumption 2.3, with some eigenvalues exceeding the theoretical threshold. This effect is particularly evident for the *Goldstein-Price* and *Beale* functions, which are most affected by this issue. Theoretically, the sum of the last $D - d_e$ eigenvalues should be bounded above by L^2, which is not observed for $h = 10^{-8}$ in these cases.



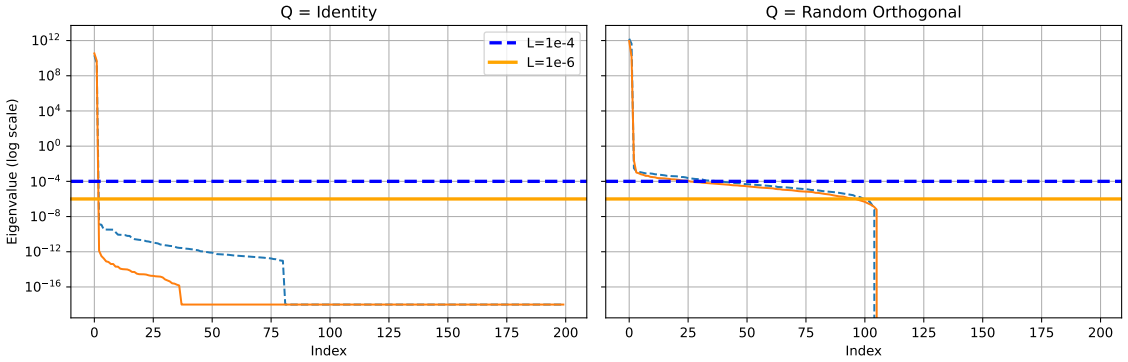
(a) Beale, appendix A.2.3. $d_e = 2$



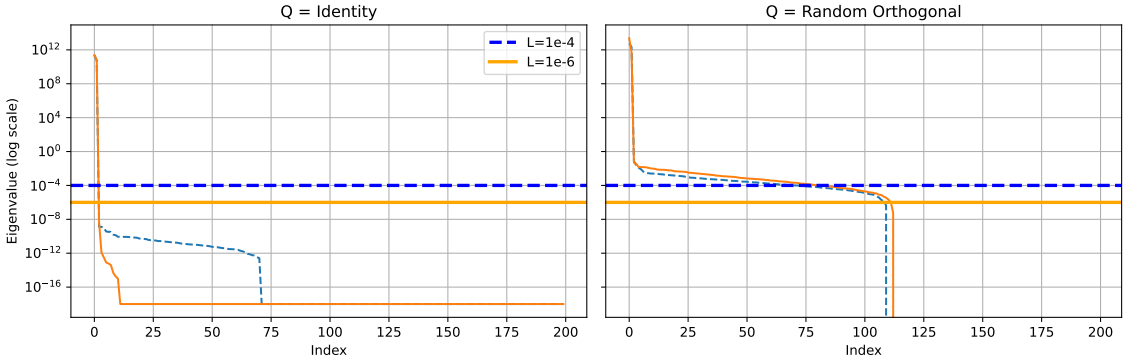
(b) Goldstein-Price, Appendix A.2.5. $d_e = 2$.

Figure 2.5: Influence of finite-difference approximation of the gradient in the spectrum of $\hat{C}$. The thresholds display L^2.

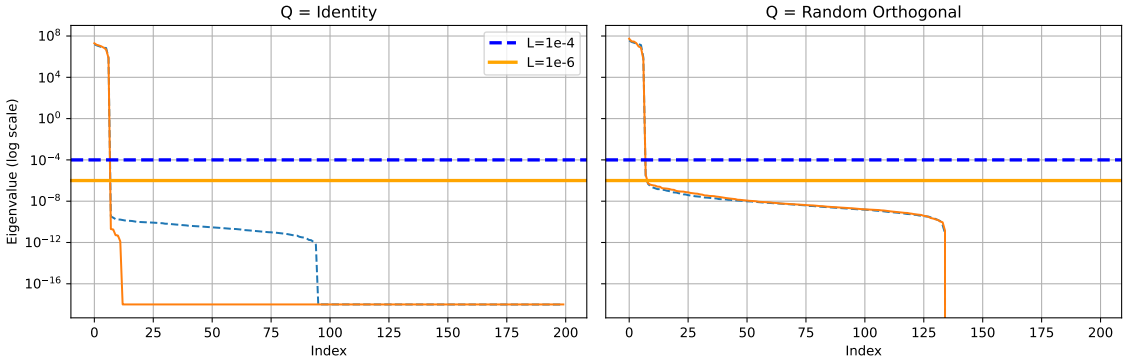
The second identified factor concerns the construction of the test set. Specifically, applying the rotation by evaluating $f(x) = h(Q^T x)$, where Q is a $D \times D$ orthogonal matrix, instead of directly implementing the rotated function, introduces numerical errors in the computation of the product $Q^T x$. These errors affect the reconstruction of the spectrum of $\hat{C}$. This phenomenon is illustrated in Figure 2.6. In the left-hand plots, where no rotation is applied (i.e., no matrix multiplication occurs within the test function definition), the spectrum is reconstructed accurately for all three displayed functions. In contrast, the right-hand plots show that the inclusion of the internal matrix multiplication introduces numerical errors that distort the spectrum, leading to an overestimation of d_e. This effect is particularly evident for the Beale and Goldstein-Price functions, while the Rosenbrock function remains largely unaffected. These observations are consistent with the results reported in Tables 2.1 and 2.2, where no overestimation is observed for the Rosenbrock function, but well for Beale and Goldstein-Price.



(a) Beale



(b) Goldstein-Price



(c) Rosenbrock

Figure 2.6: Influence of error propagation when computing the product $Q^T x$. The horizontal lines represent L, the perturbation magnitude.

A final idea was investigated to explain the overestimation of d_e by assuming that $\hat{C}$ is computed using the exact gradient, rather than a finite-difference approximation. Recall that

$$C = \mathbb{E}_{x\sim\rho}\left[\nabla f(x)\nabla f(x)^T\right],$$

and that $\hat{C}$, constructed from M samples, is an unbiased estimator of C. Consequently, its variance is expected to decrease as M increases, which should theoretically lead to improved accuracy in the detection of the active subspace. However, in practice, once M exceeds the minimum number of samples required to capture

the spectrum related to the active subspace, further increases in M do not yield significant improvements in the quality of the estimation of d_e.

Finally, the proportion of problems solved across the three seeds is reported for each algorithm and each dimension. Following the convention in [15], a value of 1 indicates that all three seeds successfully solved the problem up to the threshold $\epsilon = 10^{-3}$, meaning $|f_{\text{opt}} - f^*| \leq \epsilon$. A value of 0 indicates that none of the seeds achieved convergence. Intermediate values of the form $\alpha/3$ denote that the problem was solved by α out of the three seeds. The results for $L = 10^{-4}$ are shown in Table 2.3.

	Clip (1)		Clip (2)		approx-ASM		full domain	
	$D = 10$	$D = 200$	$D = 10$	$D = 200$	$D = 10$	$D = 200$	$D = 10$	$D = 200$
Squared Sine	1	1	1	1	1	1	1	0
Styblinski-Tang 2D	2/3	1	2/3	2/3	2/3	2/3	1	1
Beale	2/3	2/3	2/3	2/3	1/3	1/3	1	1
Hartman 3D	2/3	1/3	2/3	2/3	2/3	1	1	1
Hartman 6D	1/3	2/3	1/3	1/3	2/3	1/3	1	1
Trid	1/3	2/3	1/3	1/3	1	1/3	1	1
Goldstein-Price	2/3	1	1/3	0	0	1/3	1	1
Levy	1/3	1	2/3	2/3	2/3	0	2/3	1/3
Rosenbrock	0	0	1/3	1/3	0	1/3	1	1

Table 2.3: Global Results for $L = 10^{-4}$.

The results for $L = 10^{-6}$ are displayed on Table 2.4.

	Clip (1)		Clip (2)		approx-ASM		full domain	
	$D = 10$	$D = 200$	$D = 10$	$D = 200$	$D = 10$	$D = 200$	$D = 10$	$D = 200$
Squared Sine	1	1	1	1	1	1	1	1
Styblinski-Tang 2D	1	1/3	2/3	2/3	1/3	1	1	1
Beale	2/3	2/3	1/3	2/3	2/3	2/3	1	1
Hartman 3D	1/3	0	1/3	2/3	2/3	1	1	1
Hartman 6D	2/3	1/3	1/3	1/3	1/3	1/3	1	1
Trid	2/3	1/3	1/3	2/3	1	1/3	1	1
Goldstein-Price	1	1	1/3	1/3	1/3	1/3	1	1
Levy	1/3	2/3	2/3	0	2/3	0	1	1/3
Rosenbrock	0	1/3	0	1/3	1/3	1/3	1	1

Table 2.4: Global Results for $L = 10^{-6}$.

The results show that the performance of the **full domain** approach remains stable across both perturbation magnitudes, with no significant differences observed. In contrast, for any value of L, all other algorithms typically require more than one initialization (i.e., seed) to successfully solve a problem. This highlights the sensitivity of low-dimensional subspace optimization methods to the initial configuration of the sampled points, emphasizing the importance of initialization in such settings.

Chapter 3

Stochastic perturbations

This chapter investigates the problem of identifying effective subspaces of functions that satisfy a low effective dimensionality assumption but are accessible only through noisy evaluations. Let $f : \mathbb{R}^D \to \mathbb{R}$ be a continuously differentiable, possibly non-convex function with known effective dimensionality $d_e \ll D$. Formally, given a point $\bar{x} \in \mathbb{R}^D$ drawn at random, the only accessible quantity is a noisy observation

$$\bar{f}(\bar{x}) = f(\bar{x}) + \epsilon(\bar{x}), \tag{3.1}$$

where $\epsilon(\bar{x}) \sim \mathcal{N}(0, \sigma^2)$ denotes a realization of a Gaussian noise variable. The noise variance σ^2 is assumed to be known. As discussed in the introduction of Chapter 2, exploiting structural information about the low-dimensional nature of the function can significantly accelerate global optimization procedures, such as the one described in Equation (1). This direction is further explored in [15]. The objective of this chapter is to recover such effective subspaces in the presence of noisy function evaluations. As an illustration, see Figure 3.1. The approximate constant direction of the function is spanned by the vector $(1, 1)$.

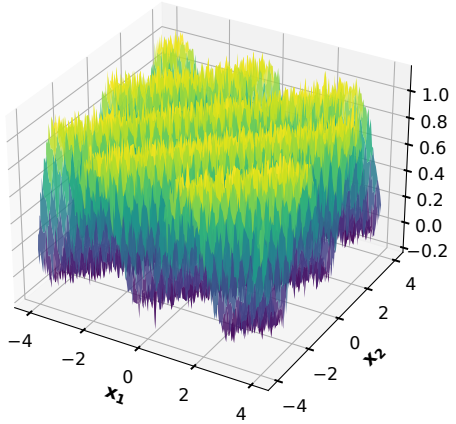


Figure 3.1: The represented function is $f : \mathbb{R}^2 \to \mathbb{R} : x \to \sin^2(x_1 - x_2 - 0.5) + \epsilon(x)$, with $\epsilon(x) \sim \mathcal{N}(0,\, \sigma^2 = 0.01)$.

Identifying this approximate effective subspace is therefore essential, as it yields an approximate minimizer up to the stochastic perturbation. In contrast to the approach developed in Chapter 2, the emphasis here lies in identifying the invariant and effective subspaces of the objective function, rather than integrating such structural insights directly into the global optimization procedure.

While the active subspace method offers a widely used approach for subspace identification, it exhibits notable limitations in noisy settings, as illustrated by examples in this chapter. To overcome these difficulties, an alternative framework based on Riemannian optimization is introduced. The optimization is performed over the Grassmann manifold, which provides a natural and elegant setting for representing subspaces of fixed dimension. Following a brief overview of manifolds and Riemannian optimization techniques, the problem is formally defined, and a theoretical analysis of the proposed formulation is conducted. Numerical experiments are then presented to evaluate the practical performance of the method, including comparisons with the active subspace approach. The respective advantages and limitations of each method are also discussed. The test functions used in this study are detailed in Appendix A.2.

3.1 Limitations of the active subspace approach

An initial approach leveraged the methodology presented in Section 1.1. Specifically, the leading eigenspace of the matrix $\hat{C}$ was estimated using a finite difference scheme, as defined in Equation (1.3). However, such schemes exhibit high sensitivity to noise. Assuming a Gaussian noise model $\epsilon(x) \sim \mathcal{N}(0, \sigma^2/2)$, the construction of $\hat{C}$ requires evaluating directional derivatives along the canonical basis vectors e_i, for $i = 1, \ldots, D$. For a finite step size $h > 0$, the i-th component of the estimator takes the form

$$[g_h(x)]_i = \frac{f(x + he_i) - f(x)}{h} + \frac{\epsilon'(x)}{h} \sim \mathcal{O}(h) + \frac{\epsilon'(x)}{h}, \tag{3.2}$$

where $\epsilon'(x) \sim \mathcal{N}(0, \sigma^2)$. As a result, the noise term scales as $\epsilon'(x)/h \sim \mathcal{N}(0, \sigma^2/h^2)$.

This scaling highlights the difficulty in selecting an appropriate step size h. A large value of h reduces the influence of noise but increases the discretization error. Conversely, a small h improves the discretization accuracy but amplifies the variance of the estimator, rendering the result nearly random. Recent studies [62, 61] examine the performance limitations of noisy finite-difference schemes in optimization settings. One of the key findings in these studies is that, given a known noise level in function evaluations, it is possible to select a finite-difference stepsize that minimizes the expected error between the true gradient and its approximation. Nevertheless, under relatively high noise variance, the underlying structure of the function may be preserve. However, the finite-difference method becomes ineffective for estimating the gradient reliably.

In Figure 3.2, the angle in radians is reported between the true invariant subspace and the eigenvector associated with the smallest eigenvalue of the matrix $\hat{C}$, which is constructed from noisy finite difference approximations. The purpose of this experiment is to illustrate the sensitivity of subspace recovery to noise in gradient approximations.

Two functions are considered for this analysis, both described in Appendix A.2. The first is a variant of the Squared Sine function, defined explicitly in Equation (A.1), and the second is a variant of the Levy function, described in Equation (A.5). Both functions are defined in ambient dimension $D = 2$, but admit a one-dimensional effective subspace, i.e., their effective dimensionality is $d_e = 1$. This implies that the invariant structure of interest is entirely captured by a single direction, spanned by $(1, 1)$. We use $M = 10$, according to the sample complexity results of [15] (M is of the same order of d_e).

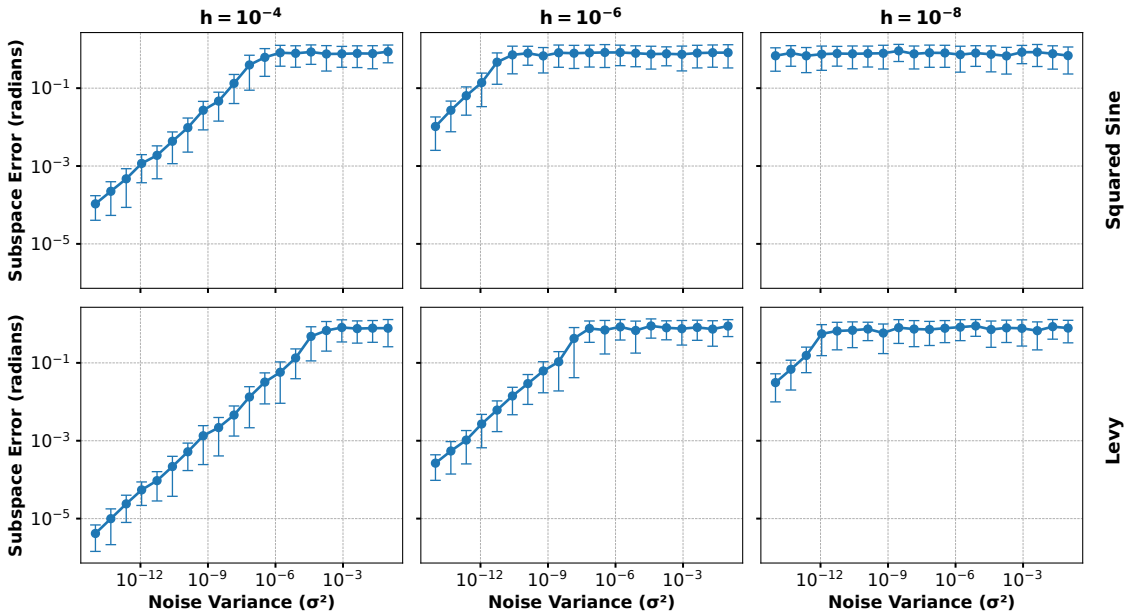


Figure 3.2: Illustration of the trade-off between the noise variance σ^2 and the discretization stepsize h in finite difference schemes. Each data point represents the average result over 50 independent trials to account for stochastic variability.

As expected, the experiment clearly highlights the instability caused by noise amplification for small discretization stepsizes h. In this regime, the finite-difference scheme becomes increasingly sensitive to noise, which severely hinders the recovery of the invariant subspace. This sensitivity is particularly evident even at relatively low noise levels. For instance, when $\sigma^2 = 10^{-6}$, the subspace estimation error (in radians) ranges from approximately 0.1 (Levy function with $h = 10^{-4}$) to around 1 for other combinations. This corresponds to an angular error between roughly 5° and 50°, illustrating the limitations of the approach in accurately identifying low-dimensional structure under stochastic perturbations.

To further illustrate the impact of noise on subspace recovery, consider 20 indepen-

dent runs of the active subspace method applied to the same two functions. The results are shown for three different levels of noise. Figure 3.3 displays the outcomes obtained using a step size of $h = 10^{-6}$, whereas Figure 3.4 corresponds to a step size of $h = 10^{-4}$. The results clearly demonstrate that when h is too small, any potential precision gain is overwhelmed by the noise, leading to significant variability and instability in the estimated invariant subspaces. A larger step size is therefore necessary to increase the likelihood of accurately recovering the true invariant subspace.

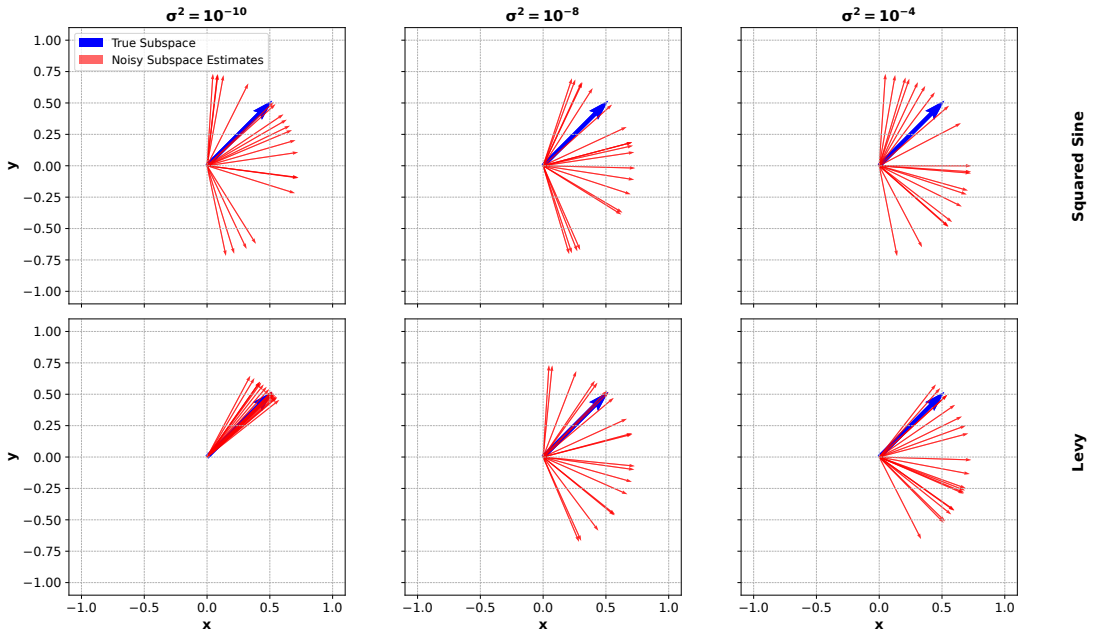


Figure 3.3: Representation of 20 trials of the Active subspace method, using $M = 10$ samples. The discretization step is set to $h = 10^{-6}$.

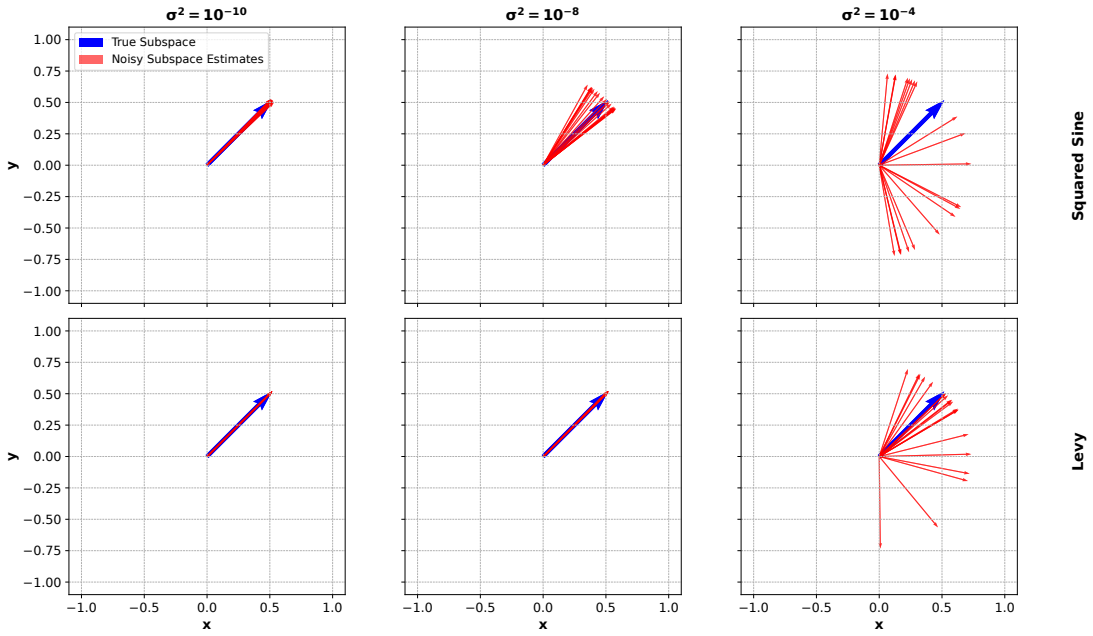


Figure 3.4: Representation of 20 trials of the Active subspace method, using $M = 10$ samples. The discretization step is set to $h = 10^{-4}$.

This section begins by addressing the trade-off associated with the choice of the finite difference stepsize h: selecting h too small amplifies noise and increases variance, whereas choosing h too large introduces a systematic bias. The focus here is on the latter scenario. As indicated in Equation (3.2), increasing h effectively reduces the variance of the estimator, but at the cost of introducing bias. In such cases, the finite difference scheme may converge to a biased direction or misidentify the dominant eigenspace of $\hat{C}$. In extreme cases, the method may even return the orthogonal complement of the true invariant subspace. This behaviour is illustrated in Figure 3.5.

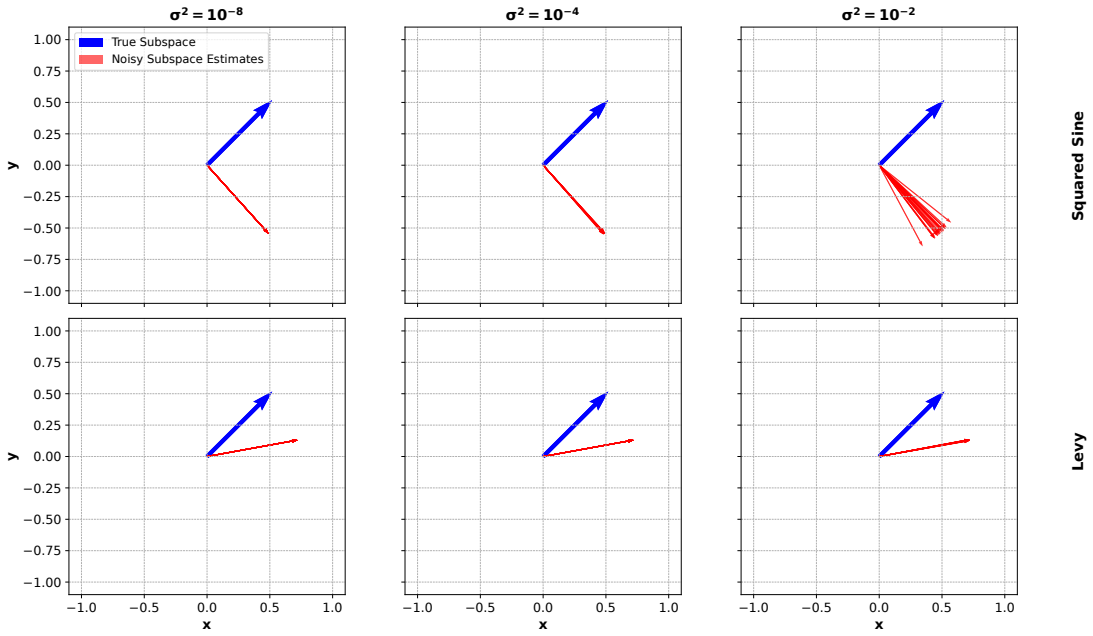


Figure 3.5: Representation of 20 trials of the Active subspace method, using $M = 10$ samples. The discretization step is set to $h = 0.5$.

To address these limitations, a new approach based on Riemannian optimization is proposed. The idea is to reformulate the spectral problem as a standard optimization problem, with the expectation that the resulting objective function may be more amenable to optimization in the presence of noise.

3.2 Matrix Manifolds and Orthogonality Constraints

This section provides a brief overview of theory of manifolds and their applications to orthogonality constraints. Consider the optimization problem

$$\min_{x \in \mathcal{M}} f(x), \tag{3.3}$$

where $\mathcal{M}$ denotes a Riemannian manifold. A detailed treatment of Riemannian manifolds falls outside the scope of this thesis, as it belongs to the domain of Riemannian geometry. There is the need to understand what is a manifold in order to tackle Equation (3.3) at best. A concise introduction can be found in [13].

3.2.1 Reminder on Manifolds

The discussion begins with the definition of a topological manifold, as presented in [44].

Definition 3.1 (Topological manifold)**.** *Let $\mathcal{M}$ be a topological space. We say that $\mathcal{M}$ is a topological manifold of dimension n or a topological n-manifold if it has the following properties:*

- *$\mathcal{M}$ is a Hausdorff space : for every pair of distinct points $p, q \in \mathcal{M}$, there are disjoint open subsets $U, V \subseteq \mathcal{M}$ such that $p \in U$ and $q \in V$.*
- *$\mathcal{M}$ is second-countable: there exists a countable basis for the topology of $\mathcal{M}$.*
- *$\mathcal{M}$ is locally Euclidean of dimension n: each point of $\mathcal{M}$ has a neighbourhood that is homeomorphic to an open subset of $\mathbb{R}^n$.*

The final point in Definition 3.1 motivates the introduction of the tangent space, a fundamental concept in differential geometry. The remainder of this section follows the work of Absil, Mahony, and Sepulchre [1].

Definition 3.2 (Tangent vector - Definition 3.5.1 of [1])**.** *Let $\mathfrak{J}_x(\mathcal{M})$ be set of smooth real-valued functions defined on a neighbourhood of x. A tangent vector ξ_x to a manifold $\mathcal{M}$ at a point x is a mapping from $\mathfrak{J}(\mathcal{M})$ to $\mathbb{R}$ such that there exists a curve γ on $\mathcal{M}$ with $\gamma(0) = x$, satisfying*

$$\xi_x f = \dot{\gamma}(0) f \coloneqq \left.\frac{\mathrm{d}\left(f(\gamma(t))\right)}{\mathrm{d}t}\right|_{t=0},$$

for all $f \in \mathfrak{J}_x(\mathcal{M})$. Such a curve γ is said to realize *the tangent vector ξ_x. The* tangent space *to $\mathcal{M}$ at x, denoted by $\mathcal{T}_x\mathcal{M}$, is the set of all tangent vectors to $\mathcal{M}$ at x. This set is a* vector space*.*

Definition 3.2 is fundamental, as the tangent space at each point $x \in \mathcal{M}$ provides a local first-order (linear) approximation of the manifold $\mathcal{M}$. This concept forms the basis for constructing optimization algorithms on smooth manifolds. A formal definition of smooth manifolds lies beyond the scope of this thesis; however, intuitively, it involves equipping a topological manifold with additional structure that enables the application of calculus-based tools. All manifolds considered in this work are assumed to be smooth.

For each vector in the tangent space, it is often necessary to define a notion of length, analogous to the Euclidean case. To this end, an inner product can be introduced on the tangent space at each point $x \in \mathcal{M}$,

$$\langle \cdot, \cdot \rangle_x : (\mathcal{T}_x\mathcal{M}, \mathcal{T}_x\mathcal{M}) \to \mathbb{R} : (\xi_x, \zeta_x) \to \langle \xi_x, \zeta_x \rangle. \tag{3.4}$$

This bilinear, symmetric, and positive-definite form induces a norm on the tangent space. If the operator defined by Equation (3.4) varies smoothly with x, it is referred to as a *Riemannian metric*. Equipping the tangent spaces of a smooth manifold with such a metric endows the manifold with additional geometric structure, resulting in what is known as a *Riemannian manifold*.

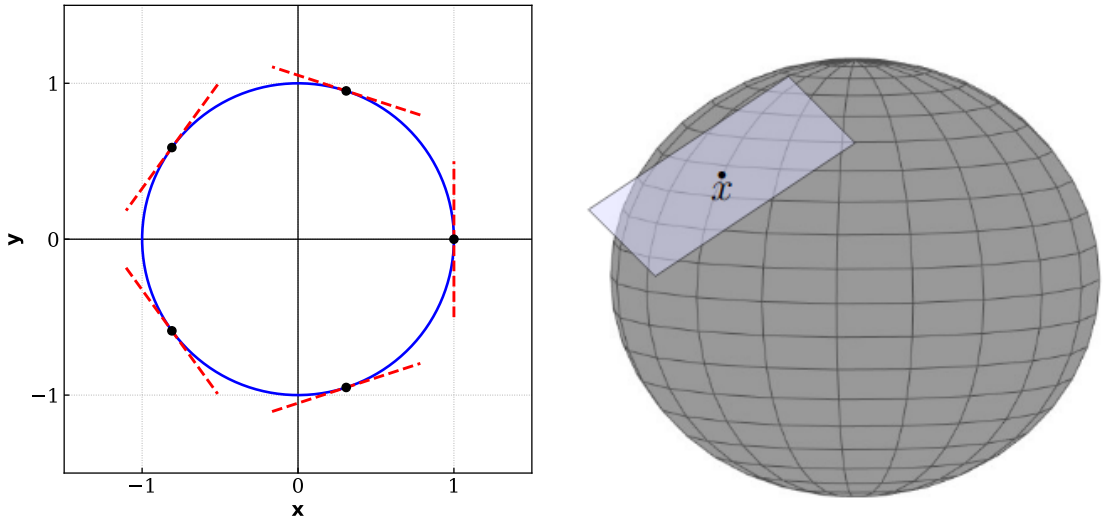
As a final intuition, a Riemannian manifold can be viewed as a topological space that admits curvature, while still locally resembling a Euclidean space. This local Euclidean structure allows for the generalization of many concepts and tools from classical differential calculus and linear algebra to the manifold setting.

3.2.2 Orthogonality Constraints

This subsection introduces several important manifolds that arise from orthogonality constraints. A comprehensive treatment can be found in [25]. The paper [25] is used as the main source for the subsection. A first example of a Riemannian manifold is the **n-dimensional unit sphere**, denoted by $\mathbb{S}^{n-1}$, and defined as

$$\mathbb{S}^{n-1} = \left\{ x \in \mathbb{R}^n \mid x^T x = 1 \right\}, \quad \langle \xi, \eta \rangle_x = \xi^T \eta.$$

Here, the inner product is independent of the base point x, and thus varies smoothly. This structure qualifies $\mathbb{S}^{n-1}$ as a Riemannian manifold. An illustration of $\mathbb{S}^1$ and $\mathbb{S}^2$ is provided in Figure 3.6.



(a) (b)

Figure 3.6: (a) $\mathbb{S}^1$, the unit circle embedded in $\mathbb{R}^2$, with one-dimensional tangent spaces illustrated at several points (red dashed lines). (b) $\mathbb{S}^2$, the unit sphere embedded in $\mathbb{R}^3$, with a two-dimensional tangent plane shown at a point x. In both cases, the dimension of the tangent space at each point is $n-1$, matching the dimension of the manifold.

Moreover, $\mathbb{S}^{n-1}$ can be classified as an *embedded submanifold*. While a rigorous definition lies beyond the scope of this thesis, an intuitive explanation is provided. Let $\mathcal{E}$ and $\mathcal{E}'$ be manifolds such that $\mathcal{E}' \subset \mathcal{E}$. If $\mathcal{E}'$ inherits the topology induced by $\mathcal{E}$, then $\mathcal{E}'$ is said to be an embedded submanifold of $\mathcal{E}$. In Figure 3.6, each sphere is an example of an embedded submanifold within the ambient Euclidean space. The next manifold generalizes these ideas to the case where $d_e > 1$.

The **Stiefel manifold** is defined as in [1, Section 3.3.2]:

$$\mathrm{St}(D, d_e) := \left\{ X \in \mathbb{R}^{D \times d_e} \,:\, X^T X = I_{d_e} \right\}, \tag{3.5}$$

where I_{d_e} denotes the $d_e \times d_e$ identity matrix. When $d_e = 1$, the Stiefel manifold reduces to the sphere $\mathbb{S}^{D-1}$. The Stiefel manifold forms an embedded submanifold

of $\mathbb{R}^{D\times d_e}$. Intuitively, a point $X \in \mathrm{St}(D, d_e)$ represents an orthonormal basis of a d_e-dimensional subspace of $\mathbb{R}^D$. Let

$$F : \mathrm{St}(D, d_e) \to \mathbb{R}, \quad Y \mapsto F(Y).$$

The gradient of a real-valued function on the Stiefel manifold is given by

$$\nabla F \coloneqq F_Y - Y F_Y^T Y, \tag{3.6}$$

where the matrix F_Y is the Euclidean gradient, defined component-wise as

$$(F_Y)_{ij} = \frac{\partial F}{\partial Y_{ij}}, \tag{3.7}$$

for $i \in \{1, \ldots, D\}$ and $j \in \{1, \ldots, d_e\}$. In certain settings, it is sufficient to optimize over the subspace spanned by the columns of a matrix $X \in \mathrm{St}(D, d_e)$. This observation motivates the introduction of the following manifold.

The **Grassmann manifold**, denoted by $\mathrm{Gr}(D, d_e)$, is defined as the set of all d_e-dimensional subspaces of $\mathbb{R}^D$. To relate an element of $\mathrm{St}(D, d_e)$ to a fiber in $\mathrm{Gr}(D, d_e)$, an *equivalence relation*, denoted by $\sim$, must be introduced. Specifically, for $X, Y \in \mathrm{St}(D, d_e)$,

$$X \sim Y \quad \Leftrightarrow \quad \mathrm{span}(X) = \mathrm{span}(Y) \quad \Leftrightarrow \quad X = YQ, \tag{3.8}$$

for some $Q \in \mathcal{O}(d_e)$, the group of $d_e \times d_e$ orthogonal matrices. Based on this equivalence, the following set can be defined:

$$[X] = \{Y \in \mathrm{St}(D, d_e) \,:\, X \sim Y\}. \tag{3.9}$$

This provides a way to define $\mathrm{Gr}(D, d_e)$ as

$$\mathrm{Gr}(D, d_e) \coloneqq \mathrm{St}(D, d_e)/\sim \coloneqq \{[X] \,:\, X \in \mathrm{St}(D, d_e)\}. \tag{3.10}$$

The Grassmann manifold thus has a *quotient geometry*, meaning that one can *quotient out* an equivalence relation from a set, making it a single point. It admits a canonical projection that maps an element of $\mathrm{St}(D, d_e)$ onto a representation of the equivalence class defining an element of $\mathrm{Gr}(D, d_e)$. Define the projection

$$\pi : \mathrm{St}(D, d_e) \to \mathrm{Gr}(D, d_e) : Y \mapsto YY^T, \tag{3.11}$$

as the mapping from the Stiefel manifold to a representative of the equivalence class defined by the Grassmann manifold. Specifically, for $Y \in \mathrm{St}(D, d_e)$ such that $\mathrm{span}(Y) = \mathcal{S}$, the matrix $P_{\mathcal{S}} = YY^T$ defines the orthogonal projection operator onto the subspace $\mathcal{S}$.

A point on the Grassmann manifold can thus be represented by any point on the Stiefel manifold spanning the same subspace. This point is described by a $D \times d_e$ matrix, and serves as the numerical representation of the associated subspace, i.e.,

the point on the Grassmann manifold. The Grassmann manifold is also an *embedded submanifold* of $\mathbb{R}^{D\times d_e}$. Let

$$F : \mathrm{Gr}(D, d_e) \to \mathbb{R} : Y \mapsto F(Y).$$

The gradient of a real-valued function on the Grassmann manifold is given by

$$\nabla F \coloneqq F_Y - YY^T F_Y, \tag{3.12}$$

where F_Y is the Euclidean gradient as defined in Equation (3.7).

3.3 Problem Formulation

The problem of identifying the effective subspace is formulated as a Riemannian optimization problem on the Grassmann manifold. The theoretical background supporting this section is presented in Section 3.2. The problem is formulated as

$$\min_{\substack{\mathcal{S}\subseteq\mathbb{R}^D \\ \dim(\mathcal{S})=d_e}} \sum_{i=1}^{M} \left[\bar{f}\left(x_{\mathcal{S}}^{(i)}\right) - \bar{f}\left(x^{(i)}\right)\right]^2, \tag{3.13}$$

where $\bar{f} : \mathbb{R}^D \to \mathbb{R}$ complies with the assumptions stated in the introduction of Chapter 3, defined by Equation (3.1), and $x_{\mathcal{S}}$ denotes the orthogonal projection of x on the subspace $\mathcal{S}$. This formulation stems from exploiting Proposition 1.1. Computationally, this will be implemented as

$$\min_{\substack{Y\in\mathbb{R}^{D\times d_e} \\ Y^TY=I_{d_e}}} \sum_{i=1}^{M} \left[\bar{f}\left(YY^Tx^{(i)}\right) - \bar{f}\left(x^{(i)}\right)\right]^2 \coloneqq F\left(Y\right), \tag{3.14}$$

where $\mathrm{span}(Y) = \mathcal{S}$. Note that $F(Y) = F(YQ)$ for any $Q \in \mathcal{O}(d_e)$, the group of $d_e \times d_e$ orthogonal matrices. This invariance implies that F depends only on the subspace spanned by the columns of Y. Consequently, problem (3.14) can be reformulated as an optimization problem on the Grassmann manifold $\mathrm{Gr}(D, d_e)$, which consists of all d_e-dimensional subspaces of $\mathbb{R}^D$. This reformulation aims to isolate the intrinsic solutions of the original problem. It is possible because the invariance property of the objective function allows us to quotient out the action of the orthogonal group. The formal definition of the objective function is given by

$$F : \mathrm{Gr}\left(D,\, d_e\right) \to \mathbb{R} : Y \to \sum_{i=1}^{M} \left[\bar{f}\left(YY^Tx^{(i)}\right) - \bar{f}\left(x^{(i)}\right)\right]^2, \tag{3.15}$$

with $\left\{x^{(i)}\right\}_{i=1}^{M}$ being a collection of points sampled at random.

The optimization problem is implemented using the `Manopt.jl` library [5], which provides a flexible framework for optimization on manifolds in the Julia programming language. The underlying manifold structures, including the Grassmann manifold, are represented and manipulated using the `Manifolds.jl` library [3], which

offers a comprehensive and extensible interface for differential geometry in Julia. This implementation builds upon the original `Manopt` toolbox developed for MATLAB in 2012 [11]. A Python implementation of the framework is also under active development; see [67] for details.

The rest of this section is dedicated to exhibit some properties related to the optimization problem described in Equation (3.13). The first results concerns the smoothness of the objective function. Next, the gradient of the objective function is presented. After that, a finite-difference scheme is discussed for optimization purposes. We also present the Gradient Descent with Armijo line-search, as well as Particle Swarm optimization. Finally, a small subsection discusses the error metrics on the Grassmann manifold. It is crucial if one wants to discuss the convergence of optimization algorithms.

3.3.1 Smoothness of the objective function

Consider the noiseless setting first, with the additional assumption that f is smooth. The goal is to analyse the smoothness of $F : \mathrm{Gr}(D, d_e) \to \mathbb{R}$. Looking at Equation (3.15), one can see that the objective function can be expressed with its domain being the Stiefel manifold, $\mathrm{St}(D, d_e)$. Define

$$\widetilde{F} : \mathrm{St}(D, d_e) \to \mathbb{R} : Y \to \sum_{i=1}^{N} \left[f\left(YY^T x^{(i)}\right) - f\left(x^{(i)}\right) \right]^2 . \tag{3.16}$$

Observe that

$$\widetilde{F} \coloneqq F \circ \pi : \mathrm{St}(D, d_e) \to \mathrm{Gr}(D, d_e) \to \mathbb{R} : Y \to P_{\mathcal{S}} \to \sum_{i=1}^{N} \left[f\left(P_{\mathcal{S}} x^{(i)}\right) - f\left(x^{(i)}\right) \right]^2 , \tag{3.17}$$

with the canonical projection defined by

$$\pi : \mathrm{St}\left(D, d_e\right) \to \mathrm{Gr}\left(D,\, d_e\right) : Y \to YY^{\top} \tag{3.18}$$

Several results are now gathered to support the analysis of the smoothness of the objective function defined in Equation (3.13). The following propositions are adapted from Chapter 3 of [1] and from [12].

Proposition 3.1 (Functions on quotient manifolds, Proposition 3.4.5 of [1])**.** *Let $\mathcal{M}/\sim$ be a quotient manifold and let f be a function on $\mathcal{M}/\sim$. Let $\pi = \mathcal{M} \to \mathcal{M}/\sim$ defined by $x \mapsto [x]$ be the canonical projection. Then $\tilde{f}$ is smooth if and only if $f \coloneqq \tilde{f} \circ \pi$ is a smooth function on $\mathcal{M}$.*

Proposition 3.2 (Embedded structure of Stiefel manifolds, Section 3.3.2 of [1])**.** *Let $St(D, d_e)$, $(d_e \leq D)$ be a Stiefel manifold. It holds that $St(D, d_e)$ is an embedded submanifold of $\mathbb{R}^{D \times d_e}$.*

Proposition 3.3 (Smoothness on embedded submanifolds, Proposition 3.31 of [12]). *Let $\mathcal{M}$ be an embedded submanifold of a manifold $\mathcal{E}$. Then $F : \mathcal{M} \to \mathbb{R}$ is smooth if there exists a neighbourhood $\mathcal{U}$ of $\mathcal{M}$ in $\mathcal{E}$ and a smooth map $\bar{F} : \mathcal{U} \to \mathbb{R}$ such that $F = \bar{F}|_{\mathcal{M}}$. Precisely, $\bar{F}(y) = F(y)$ for all $y \in \mathcal{M}$. The map $\bar{F}$ is called a smooth extension of F.*

Proposition 3.4 (Compactness of the Stiefel manifold, Example 8.28 of [12]). *$St(D, d_e)$ is closed in $\mathbb{R}^{D \times d_e}$ and bounded, and therefore compact.*

Proposition 3.5 (Smooth extension of the Stiefel manifold, Section 3.3 of [12]). *$\mathrm{St}(D, d_e)$ is closed in $\mathbb{R}^{D \times d_e}$. The map $F : \mathrm{St}(D, d_e) \to \mathbb{R}$ is smooth if and only if there exists a smooth $\bar{F} : \mathbb{R}^{D \times d_e} \to \mathbb{R}$ such that $\bar{F}|_{\mathrm{St}(D,d_e)} = F$.*

Proposition 3.6 (Linearity for smooth maps, Section 3.3 of [12]). *Let $G_1, G_2 : \mathcal{M} \to \mathcal{E}$ be smooth maps. For $a_1, a_2 \in \mathbb{R}$, $G(\cdot) \coloneqq a_1 G_1(\cdot) + a_2 G_2(\cdot)$ is smooth.*

Taken together, these results provide the necessary theoretical foundation for analysing the smoothness of the objective function $F : \mathrm{Gr}(D, d_e) \to \mathbb{R}$ via its lifted version $\widetilde{F} : \mathrm{St}(D, d_e) \to \mathbb{R}$. In particular, Proposition 3.1 allows one to study the smoothness of F on the quotient manifold $\mathrm{Gr}(D, d_e)$ by considering the smoothness of $\widetilde{F}$ on the Stiefel manifold. Propositions 3.2 to 3.6 then provide sufficient conditions to establish the smoothness of $\widetilde{F}$ as a restriction of a smooth function defined on the ambient Euclidean space $\mathbb{R}^{D \times d_e}$. The next result formalizes this reasoning by establishing that the smoothness of the original function f on $\mathbb{R}^D$ indeed implies the smoothness of the objective function F defined on the Grassmann manifold. The proof proceeds by explicitly constructing a smooth extension of the lifted objective and applying the aforementioned propositions.

Theorem 3.1 (Smoothness of the objective function). *If $f : \mathbb{R}^D \to \mathbb{R}$ is smooth, then the objective function defined by Equation (3.13) is smooth.*

Proof. Proposition 3.1 states that one can prove the smoothness of $\widetilde{F}$ to characterize the smoothness of F. Proposition 3.6 allows us to consider only one term of the sum in the cost function. Propositions 3.2, 3.3 and 3.5 allow us to consider an extension of $\widetilde{F}$

$$\bar{F} : \mathbb{R}^{D \times d_e} \to \mathbb{R} : Y \to \left[f\left(YY^T x\right) - f\left(x\right) \right]^2, \tag{3.19}$$

with $x \in \mathbb{R}^D$, where the superscript (i) is dropped, as one can consider only one term of the sum. It is easily verified that $\bar{F}|_{\mathrm{St}(D,d_e)}(y) = \widetilde{F}(y)$, for all $y \in \mathrm{St}(D, d_e)$, i.e. a restriction of $\bar{F}$ to matrices with mutually orthogonal columns gives $\widetilde{F}$.

Based on the above discussion, if one manages to show the smoothness of $\bar{F}$, then the smoothness of F is immediate. The euclidean case is expected to be easier to show, justifying the above reasoning. As the composition of smooth functions preserves

smoothness, the analysis is further restricted to the characterization of

$$f' : \mathbb{R}^{D\times d_e} \to \mathbb{R} : Y \to f(YY^T x). \tag{3.20}$$

It is easy to derive that for $x \in \mathbb{R}^D$ and $Y \in \mathbb{R}^{D\times d_e} := [v_1, ...v_p]$ and $v_j \in \mathbb{R}^D$ for $j \in \{1, .., p\}$,

$$YY^T x = \sum_{j=1}^{d_e} v_j \langle v_j, x\rangle,$$

with $\langle \cdot, \cdot \rangle$ the canonical euclidean product. Finally, define

$$g : \mathbb{R}^{D\times d_e} \to \mathbb{R}^D : Y \to \sum_{j=1}^{d_e} v_j \langle v_j, x\rangle. \tag{3.21}$$

with $x \in \mathbb{R}^D$ given, such that $f' \equiv f \circ g$. It is easily seen that g is a smooth function everywhere on $\mathbb{R}^{D\times d_e}$. Looking at each of its components, one notices that it is a quadratic form in terms of the entries of Y, and this is smooth. It follows that under the assumption of the smoothness of f, one has that F is also a smooth function from $\mathrm{Gr}(D, d_e)$ to $\mathbb{R}$. The Flowchart D.1 in Appendix represents this reasoning as well. □

It is important to note that, when considering a fixed realization of the noise, thereby removing all stochasticity from the objective function, the resulting modification corresponds to an affine shift. As such, the objective function remains smooth in this setting as well.

3.3.2 Gradient of the objective function

In this subsection, the gradient of the objective function is derived analytically. Recall that the Riemannian gradient of a real-valued function F defined on the Grassmann manifold $\mathrm{Gr}(D, d_e)$ is given by

$$\nabla F(Y) = F_Y - YY^T F_Y, \tag{3.22}$$

where $F_Y \in \mathbb{R}^{D\times d_e}$ denotes the Euclidean gradient of F, with components

$$(F_Y)_{ij} = \frac{\partial F}{\partial Y_{ij}}, \quad \text{for } i = 1, \ldots, D, \quad \text{and } j = 1, \ldots, d_e.$$

The Euclidean gradient of the objective function defined in Equation (3.15) is expressed as

$$F_Y = 2\sum_{i=1}^{N} \left(f(YY^T x^{(i)}) - f(x^{(i)})\right) \left[\nabla f(YY^T x^{(i)})^T x^{(i)\,T} Y + x^{(i)} \nabla f(YY^T x^{(i)}) Y\right]. \tag{3.23}$$

Given this expression for F_Y, the corresponding Riemannian gradient on $\mathrm{Gr}(D, d_e)$ follows directly from Equation (3.22). Additional details regarding the derivation are provided in Appendix D.

3.3.3 Finite-Difference scheme

In the current setup, one only have access to noisy queries of the function. The aim of this subsection is to design a finite-difference scheme on Riemannian manifolds. We start with the Euclidean case, in order to provide a generalization that is expected to work on the Riemannian case. Consider a continuously differentiable function $f : \mathbb{R}^D \to \mathbb{R}$, and a step size $h > 0$. Then

$$\nabla f(x) \approx \sum_{i=1}^{D} [g_h(x)]_i e_i, \quad [g_h(x)]_i = \frac{f(x + he_i) - f(x)}{h}, \tag{3.24}$$

with e_i denoting the i-th canonical basis of $\mathbb{R}^D$. Based on the above reasoning, one possible generalization would be to consider the canonical decomposition of the tangent space of the manifold $\mathcal{M}$ at x, that is $\mathcal{T}_x\mathcal{M}$.

Given a point $Y \in \mathrm{Gr}(D, d_e)$, the tangent space at Y consists of all $D \times d_e$ matrices Δ satisfying the condition:

$$Y^T \Delta = 0,$$

as detailed in Since this equation holds for any representative Y of the equivalence class $[Y]$, it suffices to check it for a single choice of Y. To see this, let $Z = YQ$ with $Q \in \mathcal{O}(d_e)$. Then $Z^T\Delta = Q^T Y^T \Delta = 0$. For more informations about tangent spaces for Grassmann manifolds, see [25, Equation 2.64 of Section 2.5]. Algorithm 7 represents this procedure.

Algorithm 7 Computation of the Tangent Space on $\mathrm{Gr}(p, n)$

1: **Input:** A matrix $Y \in \mathbb{R}^{D \times d_e}$ with orthonormal columns.
2: Compute the null space of Y^T to obtain an orthonormal basis $W = \{v_{d_e+1}, \ldots, v_D\}$ of the orthogonal complement of $\mathrm{col}(Y)$.
3: Initialize an empty set $\mathcal{B}$ to store basis matrices.
4: **for** each $v_i \in W$, with $d_e + 1 \leq i \leq D$ **do**
5: **for** $j = 1$ to d_e **do**
6: Define $\Delta_{ij} \in \mathbb{R}^{D \times d_e}$ as a matrix where all columns are zero except the j-th column, which is set to v_i.
7: Add Δ_{ij} to $\mathcal{B}$.
8: **end for**
9: **end for**
10: **Output:** The set $\mathcal{B}$ forming a basis for the tangent space. The tangent space is given by the span:

$$T_Y \mathrm{Gr}(D, d_e) = \mathrm{span}\{\Delta_{ij} \mid d_e + 1 \leq i \leq D, 1 \leq j \leq d_e\} = \mathrm{span}\{\mathcal{B}\}.$$

As a sanity check of Algorithm 7, consider the following propositions.

Proposition 3.7 (Dimension of the tangent space, Section 3.5.1 of [1])**.** *Let $\mathcal{M}$ be a manifold. The dimension of the vector space $\mathcal{T}_x\mathcal{M}$ is equal to the dimension of the manifold $\mathcal{M}$.*

Proposition 3.8 (Dimension of the Grassmann manifold, Section 2.5 of [25]). *The dimension of* $\mathrm{Gr}(D, d_e)$ *is* $d_e \times (D - d_e)$.

Clearly, the set $\mathcal{B}$ contains $d_e \times (D - d_e)$ linearly independent elements. Hence, the basis $\mathcal{B}$ spans a space of the same dimension as $\mathrm{Gr}(D, d_e)$, confirming the correctness of Algorithm 7.

Let $F : \mathrm{Gr}(D, d_e) \to \mathbb{R}$. Based on Algorithm 7, a natural extension of Equation (3.24) is to consider

$$\nabla F(Y) \approx \sum_{i,j} \frac{F(R_Y[h\Delta_{ij}]) - F(Y)}{h} \Delta_{ij}, \tag{3.25}$$

for $i = d_e + 1, \ldots, D$ and $j = 1, \ldots, d_e$. $R_Y(\cdot)$ is the Retraction operator on the Grassmann manifold. Intuitively, a Retraction is an operator that takes a point $X \in \mathcal{M}$ and a vector $v \in \mathcal{T}_X\mathcal{M}$, and move the point X along the manifold, in the direction v. A formal definition is provided.

Definition 3.3 (Definition 3.47 of [12]). *A retraction on a manifold* $\mathcal{M}$ *is a smooth map*

$$R : \mathcal{M} \times \mathcal{T}\mathcal{M} \to \mathcal{M} : (x, v) \mapsto R_x(v),$$

such that each curve $c(t) = R_x(tv)$ *satisfies* $c(0) = x$ *and* $c'(0) = v$.

This concept plays a central role in Riemannian optimization. A retraction operator enables movement along the manifold, allowing iterative updates that remain on the manifold and potentially converge to a solution. In this thesis, an intuitive explanation is sufficient; see Figure 3.7, downloaded from [47].

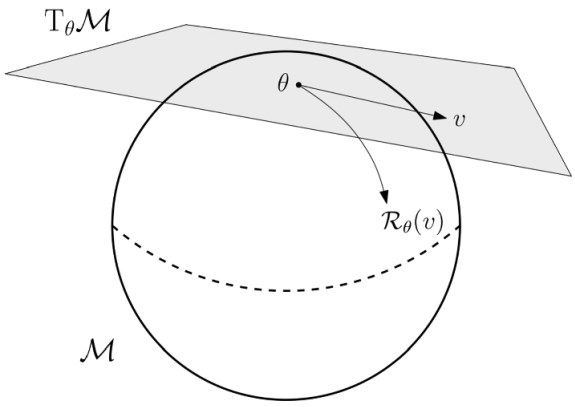


Figure 3.7: Illustration of a retraction operator on a manifold. The point θ lies on the manifold, and the vector v belongs to the tangent space at θ. The operator $\mathcal{R}_\theta(v)$ maps v back onto the manifold, with the effect of moving θ in the direction v. Figure courtesy of [47].

As illustrated, the retraction $\mathcal{R}_\theta(v)$ moves the point θ along the direction v, while ensuring that the resulting point remains on the manifold.

3.3.4 Gradient Descent with Armijo line search

Intuitively, a Riemannian gradient descent method operates similarly to its Euclidean counterpart, with the key distinction that iterates must be retracted back onto the manifold after each update. The algorithm is recalled below, following the formulation from [12], where the quantity ∇F denotes the Riemannian gradient.

Algorithm 8 Riemannian Gradient Descent, Algorithm 4.1 of [12]

1: **Input:** $x_0 \in \mathcal{M}$
2: **for** $k = 0, 1, 2, \ldots$ **do**
3: Pick a step-size $\alpha_k > 0$
4: $x_{k+1} = R_{x_k}(s_k)$, with step $s_k = -\alpha_k \nabla f(x_k)$
5: **end for**
6: **Output:** A sequence $x_0, x_1, x_2, \ldots$ generated by the Gradient Descent.

Note that the sc The choice of the step size α_k is typically handled using a backtracking line-search strategy. This approach is also recalled below, following the exposition in [12].

Algorithm 9 Armijo line-search, Riemannian optimization, Algorithm 4.2 of [12]

1: **Parameters:** $\tau, r \in (0, 1)$; for example $\tau = \frac{1}{2}$ and $r = 10^{-4}$
2: **Input:** $x \in \mathcal{M}$, $\bar{\alpha} > 0$
3: Set $\alpha \leftarrow \bar{\alpha}$
4: **while** $f(x) - f(R_x(-\alpha \nabla f(x))) < r\alpha \|\nabla f(x)\|^2$ **do**
5: Set $\alpha \leftarrow \tau\alpha$
6: **end while**
7: **Output:** α

For further details on convergence aspects in Riemannian optimization, the reader is referred to [12, Chapter 4].

3.3.5 Particle Swarm Optimization

Particle Swarm Optimization (PSO) was initially introduced as a metaheuristic algorithm simulating social behaviour [38]. As emphasized in the introduction of [37], it models the human capacity to process knowledge collectively within societies. From a high-level perspective, given a set of initial positions, referred to as the *swarm*, the algorithm iteratively updates each particle. Intuitively, particles explore the search space, updating their positions based on whether a new location leads to a lower objective function value. The process is summarized below, reproducing Equation 4 from [39]:

$$\begin{aligned} \text{NEW\_POSITION} = {} & \text{CURRENT\_POSITION} \\ & + \text{PERSISTENCE} \\ & + \text{SOCIAL\_INFLUENCE}. \end{aligned} \tag{3.26}$$

In this formulation, each particle moves from its current location, influenced by its momentum and the positions of other particles. Section 3 of [10] provides a comprehensive overview of PSO in the Euclidean setting, followed by an extension known as *Guaranteed Convergence PSO* (GCPSO). Since the Grassmann manifold is not a vector space, the algorithm must be adapted accordingly. The original PSO algorithm [38] does not ensure convergence to a stationary point. To address this limitation, the authors of [69] proposed modifications that improve convergence properties. A general PSO framework, adapted to the Grassmann manifold, is implemented in the `Manopt.jl` library and is summarized below.

Algorithm 10 Particle Swarm Optimization for Riemannian Optimization

Parameters:
$S = \{s_1, \ldots, s_n\}$; $s_k^{(i)}$ denotes the k-th particle at iteration i, for $k = 1, \ldots, n$.
$\mathcal{M}$, the current manifold
$\left\{X_k^{(i)}\right\}_{k=1}^{n}$: The velocities of k-th particle at iteration i
w, c, s : Inertia, cognitive weight, and social weight, respectively
r_1, r_2 : Newly generated random factor at each step.
$\mathcal{T}_{\{\cdot \leftarrow \cdot\}}$: A vector transport.
Retr^{-1} is an inverse retraction.
$\left\{p_k^{(i)}\right\}_{k=1}^{n}$: Best position of each particle at iteration i.
$g^{(i)}$: Best position among all particles at iteration i.
Initialization :
Draw randomly $\{s_k^{(0)}\}_{k=1}^{n}$ on $\mathcal{M}$.
Set $X_k^{(0)} = 0$, for $k = 1, \ldots, n$.
for i = 1,2, ... **do**
  for k = 1, ..., n **do**
    $X_k^{(i)} = \omega \mathcal{T}_{s_k^{(i)} \leftarrow s_k^{(i-1)}} X_k^{(i-1)} + cr_1 \operatorname{Retr}^{-1}_{s_k^{(i)}}(p_k^{(i)}) + sr_2 \operatorname{Retr}^{-1}_{s_k^{(i)}}(p_k^{(i)})$,
    $s_k^{(i+1)} = \operatorname{Retr}_{s_k^{(i)}}\left(X_k^{(i)}\right)$
    $p_k^{(i+1)} = \begin{cases} s_k^{(i+1)} & \text{if } F\left(s_k^{(i+1)}\right) < F\left(p_k^{(i)}\right) \\ p_k^{(i)} & \text{else.} \end{cases}$
  end for
  $g^{(i+1)} = \begin{cases} p_k^{(i+1)} & \text{if } F\left(p_k^{(i+1)}\right) < F\left(g^{(i)}\right) \text{ for some } k \in \{1, \ldots, n\}, \\ g^{(i)} & \text{else.} \end{cases}$
end for
OUTPUT : g contains the optimal point.

As described in [10, Section 3.3], in the context of the Grassmann manifold $\operatorname{Gr}(D, d_e)$, the **transport vector** is defined by

$$\mathcal{T}_{B \leftarrow A} : \mathbb{R}^{D \times d_e} \times \mathbb{R}^{D \times d_e} \to \mathbb{R}^{D \times d_e} : (A, B) \to \frac{\mathrm{d}}{\mathrm{dt}} \operatorname{Exp}_A\left(t\dot{B}\right)\Big|_{t=1}, \tag{3.27}$$

with

$$\operatorname{Exp}_A(tX) \coloneqq AV \cos(t\Theta) + U \sin(t\Theta), \tag{3.28}$$

where $X = U\Theta V^\top$ is a thin SVD. Note that Equation (3.28) characterizes the geodesic starting at a point $A \in \mathrm{Gr}(D, d_e)$ with velocity $X \in \mathcal{T}_A\mathrm{Gr}(D, d_e)$. This allows us to define the retraction on the Grassmann manifold

The **retraction** on the Grassmann manifold $\mathrm{Gr}(D, d_e)$ is defined by

$$\begin{aligned}\mathrm{Retr}_{s_k^{(i)}}\left(X_k^{(i)}\right) &: \mathrm{Gr}(D, d_e) \times \mathcal{T}\mathrm{Gr}(D, d_e) \to \mathrm{Gr}(D, d_e) \\ &: \left(s_k^{(i)}, X_k^{(i)}\right) \to \mathrm{Exp}_{s_k^{(i)}}\left(X_k^{(i)}\right).\end{aligned} \tag{3.29}$$

The **inverse retraction** on the Grassmann manifold, denoted by $\mathrm{Retr}_{\cdot}^{-1}(\cdot)$, provides the velocity of the geodesic linking two points on $\mathrm{Gr}(D, d_e)$. Assume $X, Y \in \mathrm{Gr}(D, d_e)$. It is defined by (see [1, Section 3.8] or [10, Equation 10]),

$$\mathrm{Log}_X(Y) \coloneqq U\mathrm{atan}(\Sigma)V^T, \tag{3.30}$$

where U, V and Σ are given by the thin SVD of $\Pi_X^\perp Y\left(X^TY\right)^{-1}$, where $\Pi_X^\perp \coloneqq I - XX^T$ is the orthogonal projection onto $\mathcal{T}_X\mathrm{Gr}(D, d_e)$. The thin SVD has the form

$$U\Sigma V^T = \Pi_X^\perp Y\left(X^TY\right)^{-1}.$$

3.3.6 Error Metric on the Grassmann manifold

A key element when evaluating a proposed solution is the error with respect to the true subspace. This section recalls the error metrics defined on the Grassmann manifold. Loosely speaking, the goal is to quantify the discrepancy between two subspaces of the same dimension. In this context, the error is measured via the principal angles between the two subspaces.

An error is represented by the distance between two points. Remember that a natural thing to do for $X, Y \in \mathrm{Gr}(D, d_e = 1)$, represented by a unitary vector on $\mathbb{S}^{D-1}$, is to compute the angle between X and Y. Thus

$$\mathrm{dist}_{\mathbb{S}^{D-1}}(X, Y) = \mathrm{acos}\left(X^\top Y\right). \tag{3.31}$$

One can generalize this result to $\mathrm{Gr}(D, d_e)$ for $d_e > 1$, as presented in [74, Introduction], using the *principal angles*. Let $X, Y \in \mathrm{Gr}(D, d_e)$. Those angles are computed from the *Singular Value Decomposition* (SVD) of $X^\top Y$:

$$X^\top Y = U\Sigma V^\top.$$

It turns out that $\Sigma \in \mathbb{R}^{d_e \times d_e}$ is such that

$$\Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_{d_e}) = \mathrm{diag}(\cos\theta_1, \ldots \cos\theta_{d_e}).$$

Each canonical angle is thus given by $\theta_i = \mathrm{acos}(\sigma_i)$ for $i = 1, \ldots, d_e$. Finally, the distance is given by

$$\mathrm{dist}_{\mathrm{Gr}(D,d_e)}(X, Y) = \left(\sum_{i=1}^{d_e}\theta_i^2\right)^{1/2}, \tag{3.32}$$

with θ_i the i-th principal angle, as discussed above. See that Equation (3.32) is consistent with Equation (3.31) with the restriction $d_e = 1$. For more informations about principal angles, see [30, Section 6.4.3].

3.4 Numerical experiments

The objective of this section is to present numerical results related to the material introduced in the second chapter. The discussion begins with illustrative examples in low-dimensional settings, namely in $\mathbb{R}^2$ and $\mathbb{R}^3$, to provide intuition about the objective function. The validity of the finite-difference approximation of the Riemannian gradient, as described in Equation (3.25), is then assessed. Finally, a comparison is made between the Riemannian optimization framework and the active subspace approach, as discussed in previous chapters and presented in [22].

3.4.1 Low-dimensional illustration

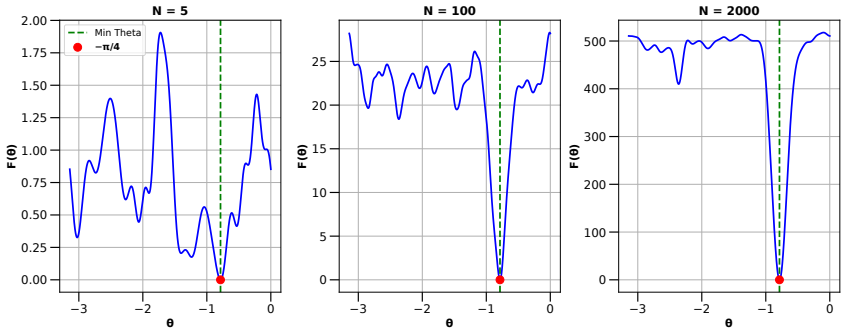
To better understand the behaviour of the objective function in the presence of noise, it is instructive to exploit the equivalence between the Grassmann manifold $\mathrm{Gr}(D, d_e = 1)$ and the unit sphere $\mathbb{S}^{D-1}$. In particular, for low-dimensional settings such as $D = 2$ and $D = 3$, the problem reduces to an optimization over the Euclidean parameter space, which allows for direct visualization of the objective function. In the case $D = 2$, the objective function can be expressed as a univariate function of an angle θ, yielding the formulation

$$F_\theta : [-2\pi,\, 0] \to \mathbb{R} : \theta \mapsto \sum_{i=1}^{N} \left[f\left(\begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix} \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix}^T x^{(i)} \right) - f\left(x^{(i)}\right) \right]^2, \tag{3.33}$$

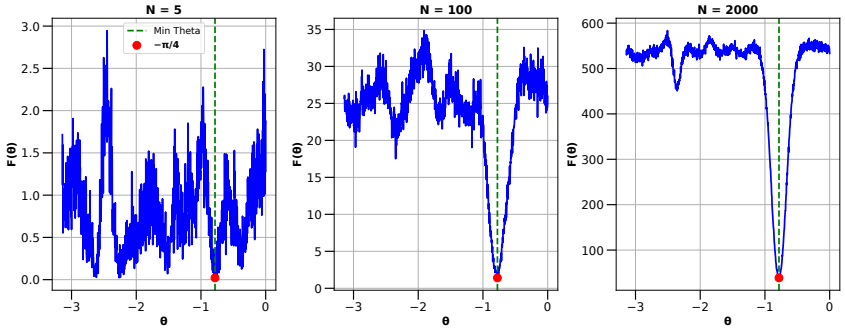
which represents a point on the circle. For the three-dimensional case ($D = 3$), the corresponding formulation is given by

$$\begin{aligned} F_{\varphi,\theta} &: [-2\pi,\, 0]^2 \to \mathbb{R} \\ &: (\varphi, \theta) \mapsto \sum_{i=1}^{N} \left[f\left(\begin{pmatrix} \sin\varphi\cos\theta \\ \sin\varphi\sin\theta \\ \cos\varphi \end{pmatrix} \begin{pmatrix} \sin\varphi\cos\theta \\ \sin\varphi\sin\theta \\ \cos\varphi \end{pmatrix}^T x^{(i)} \right) - f\left(x^{(i)}\right) \right]^2, \end{aligned} \tag{3.34}$$

which represents a point on the sphere. These parametrizations enable a detailed inspection of the landscape of the objective function as a function of the sample size N and the magnitude of the noise. The analysis focuses on the case $D = 2$, using the same test functions as described in Section 3.1, whose definitions are recalled in Appendix A.2. The results for the Squared Sine function are illustrated in Figure 3.8, and the corresponding plots for the Levy function are presented in Figure 3.9. Note that the points are sampled uniformly in $[-1,\, 1]^2$

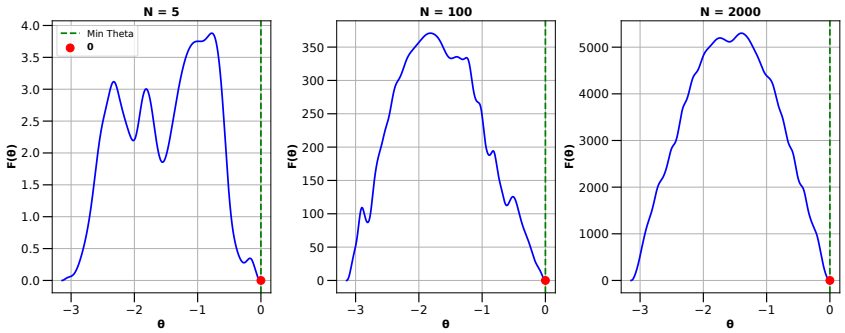


(a) Noiseless setting

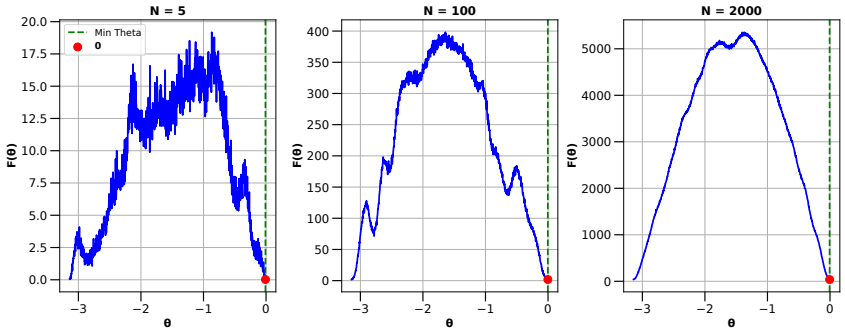


(b) $\sigma^2 = 0.01$

Figure 3.8: Squared Sine defined by Equation (A.1)



(a) Noiseless setting



(b) $\sigma^2 = 0.01$

Figure 3.9: Levy function, defined by Equation (A.5)

These plots illustrate both noiseless and noisy settings, with the addition of Gaussian noise of variance $\sigma^2 = 0.01$. While the global minimum appears clearly in both cases, the objective functions exhibit numerous local minima that can hinder optimization, excepted the setting of . This effect is particularly pronounced in the presence of noise, which reduces the smoothness of the surface and amplifies the number of misleading basins. Although this phenomenon seems to diminish as the number of evaluations N increases, we aim to deliberately restricts N to remain in a low-sample regime. This constraint reflects the goal of being competitive with active subspaces methods. Indeed, the more samples we have, the more queries of the function we have to do, in addition with the new projection $YY^T x^{(i)}$, that needs to be done at each iteration, for each sample, which depends on D.

These visual patterns help explain the limited effectiveness of gradient-based methods. The abundance of local minima and narrow valleys leads to high sensitivity to initialization, and noise further degrades the accuracy of finite-difference gradients. While Armijo line-search occasionally stabilizes convergence, consistent success remains elusive. Multistart strategies offer partial mitigation but are computationally costly and still prone to premature convergence in such rugged landscapes.

Given these difficulties, it is natural to turn to Particle Swarm Optimization (PSO). PSO is less sensitive to local irregularities and can explore the search space more globally. A gradient descent with Armijo line-search could still perform well for functions with acceptable profile, e.g. Figure 3.9a for $N = 5$, using a multistart. But those are not the usual profile we would get on random function, and this structure stems particularly from the regularity of the Levy function.

When increasing N on the Squared Sine function in both settings as in Figure 3.8, the obtained function can be related to the Easom function and its αEasom variant [36, 51, 15, 20]. These functions are notoriously difficult to optimize with local methods, because they exhibit flat landscape everywhere but around the global minimizer, being very sharp. But PSO is better equipped to handle such landscapes. Therefore, in scenarios where the objective function is highly multimodal and lacks smoothness, PSO provides a robust and justified alternative. The results presented in Appendix D.4, however, highlight that we may have scalability issues.

3.4.2 Finite-Differences scheme validation

By inspecting Figures 3.8 and 3.9, it appears that in the absence of noise and with a sufficiently large number of samples, convergence to a solution using first-order methods remains plausible. The objective of this section is therefore to numerically validate the scheme introduced in Section 3.3.3. This constitutes an essential preliminary step.

The gradient descent algorithm presented in Section 3.3.4 is employed, using $N =$

2000 samples and running for 2000 iterations. No convergence criterion is imposed on the gradient norm. A multistart strategy with five random initializations is used. The objective is to verify that the observed reconstruction error on the subspace exhibits the expected first-order convergence rate, i.e., $\mathcal{O}(h)$. The errors are displayed in radians.

This rate is evaluated across four test functions defined in Appendix A and listed in Table A.2: *Squared Sine*, *Beale*, *Goldstein-Price*, and *Levy*. The effective dimension is fixed at $d_e = 1$ and the ambient dimension is set to $D = 5$ for the function *Squared Sine*. The others are set to $(D, d_e) = (3,\ 2)$.

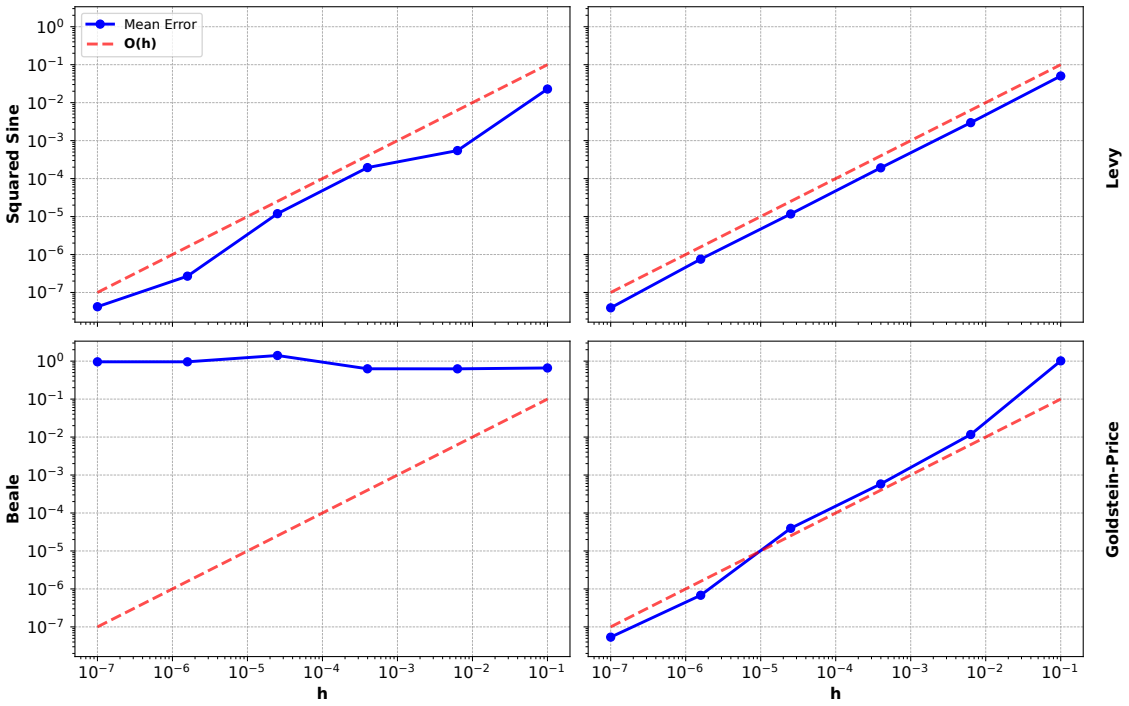


Figure 3.10: Error on the solution (mean over 20 runs) for four functions.

We see that besides *Beale function*, all the subspace are reconstructed according to a linear rate of convergence following $\mathcal{O}(h)$.

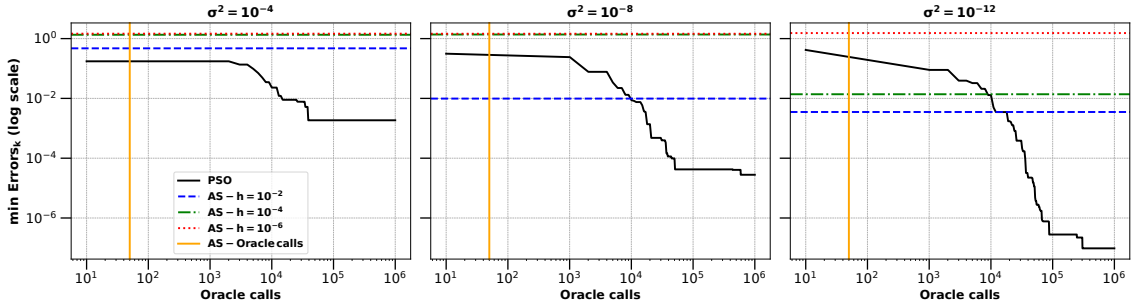
3.4.3 Numerical results

The goal of this section is to provide a rigorous numerical comparison between the PSO approach and the Active Subspace method. The analysis focuses on the four functions from the testing set detailed in Table A.2. These functions are selected because, within the broader test set described in Table A.1, they allow for clear control of the effective dimensionality, specifically for $d_e \in \{1,\ 2\}$.

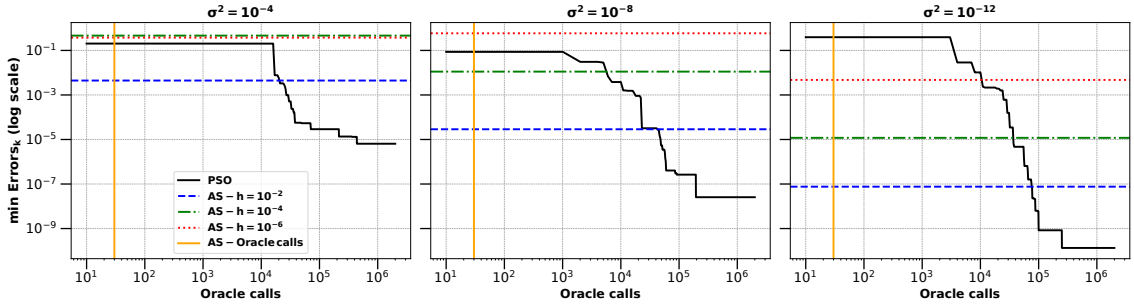
Results are presented for three different levels of noise variance in Figure 3.11. The number of samples is fixed at $N = 10$, aligning with the typical budget used in the Active Subspace method. The PSO algorithm is executed for 2000 iterations with a

swarm size of 100. A multistart strategy is used with five different seeds. The PSO performance curve displayed corresponds to the best result among the five runs.

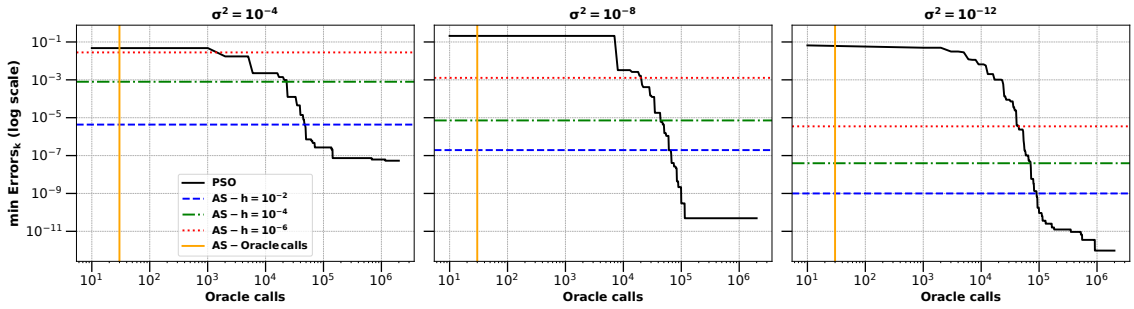
For comparative purposes, results obtained via the Active Subspace method, as discussed in the previous chapter, are also included. The gradient is approximated using finite-difference schemes with various discretization levels. The errors are displayed in radians.



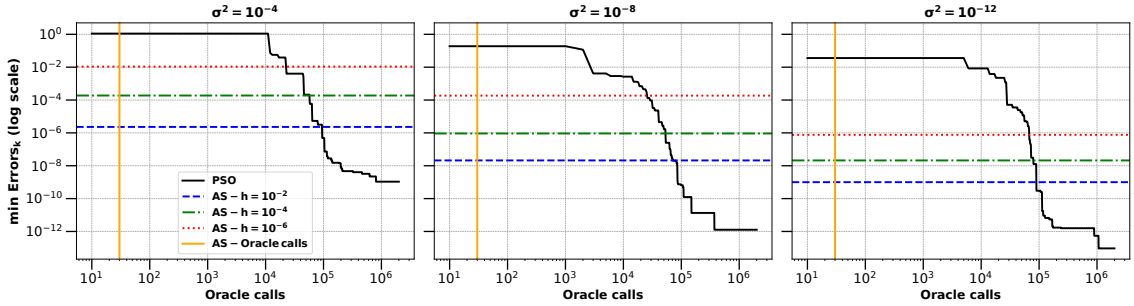
(a) Squared Sine. $d_e = 1$ and $D = 5$



(b) Levy. $d_e = 2$ and $D = 3$



(c) Beale. $d_e = 2$ and $D = 3$



(d) Goldstein-Price. $d_e = 2$ and $D = 3$

Figure 3.11: Comparison of the errors with respect to the number of oracle calls for PSO and Active subspaces. The PSO run represented (black) is the best curve obtained among the 5 multistarts.

The results presented in Figure 3.11 suggest that for low-dimensional problems, PSO outperforms Active Subspaces in terms of final accuracy. Note that the final accuracy also depends on the noise level, the lower the noise, the better the performance. However, achieving the best precision using Active Subspaces typically requires sig-

nificantly more oracle calls, approximately 1000 times more. Furthermore, in the case of Active Subspaces, employing a larger step size h improves the accuracy of the subspace reconstruction.

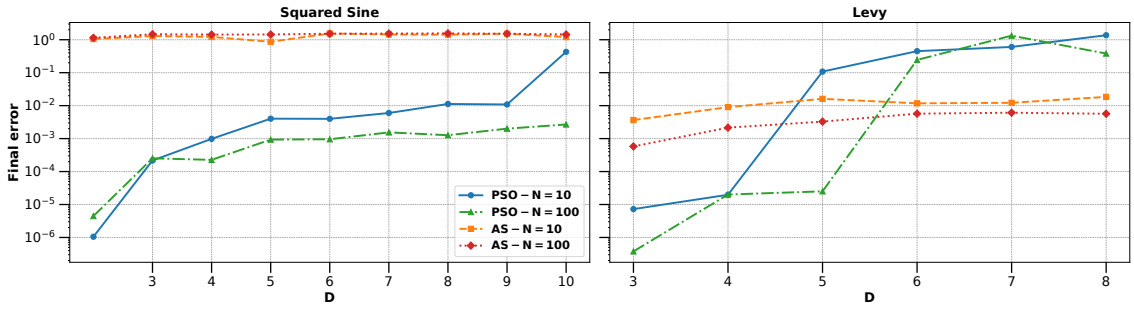
It is also worth noting that there is no built-in way to validate a solution obtained through the Active Subspaces method. In contrast, validation is feasible for the PSO algorithm, as one can analyze the objective function and its expected value at optimality. This aspect is discussed in Appendix D.3.

Numerical values corresponding to the best solutions obtained at the end of the optimization process are reported in Table 3.1. PSO consistently outperforms the Active Subspace technique, irrespective of the noise level. Additionally, higher noise levels result in decreased solution accuracy for both methods.

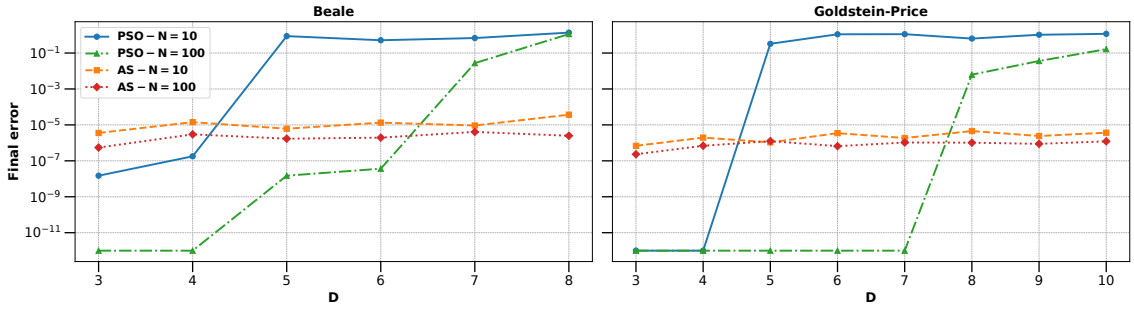
Description	σ^2	PSO	AS - $h = 10^{-2}$	AS - $h = 10^{-4}$	AS - $h = 10^{-6}$
Squared sine,	10^{-4}	0.0018	0.47	1.34	1.46
$D = 5$,	10^{-8}	2.75×10^{-5}	9.85×10^{-3}	1.37	1.45
$d_e = 1$.	10^{-12}	9.66×10^{-8}	3.51×10^{-3}	0.013	1.53
Beale,	10^{-4}	5.38×10^{-8}	4.33×10^{-6}	7.93×10^{-4}	2.81×10^{-3}
$D = 3$,	10^{-8}	4.91×10^{-11}	1.95×10^{-7}	7.18×10^{-6}	1.27×10^{-3}
$d_e = 2$.	10^{-12}	9.56×10^{-13}	1.28×10^{-9}	3.94×10^{-8}	3.5×10^{-6}
Levy,	10^{-4}	6.44×10^{-6}	4.44×10^{-3}	0.46	0.37
$D = 3$,	10^{-8}	2.56×10^{-8}	2.88×10^{-5}	1.12×10^{-2}	0.59
$d_e = 2$.	10^{-12}	1.34×10^{-10}	7.6×10^{-8}	1.19×10^{-5}	4.73×10^{-3}
Goldstein Price,	10^{-4}	1.07×10^{-9}	2.3×10^{-6}	1.88×10^{-4}	1.08×10^{-2}
$D = 3$,	10^{-8}	1.27×10^{-12}	2.11×10^{-8}	9.2×10^{-7}	1.85×10^{-4}
$d_e = 2$.	10^{-12}	9.33×10^{-14}	1.24×10^{-9}	2.11×10^{-8}	7.59×10^{-7}

Table 3.1: Final error for each algorithm (radians)

The previous analysis has focused on relatively small values of (D, d_e). It is of interest to investigate how the ambient dimension influences the convergence behavior of the algorithm. According to [15], the Active Subspace method achieves satisfactory subspace reconstruction with high probability when the number of samples N is of the order of d_e, with limited impact of the ambient dimension D. This raises the question of whether PSO exhibits a similar dependence on the ambient dimension in noisy settings. In particular, it is worth examining whether low-dimensional subspaces can be successfully recovered within higher-dimensional ambient spaces than those considered in Table 3.1 and Figure 3.11. Numerical results addressing this question are presented in Figure 3.12, with the noise variance set to $\sigma^2 = 10^{-4}$.



(a)



(b)

Figure 3.12: Scaling of PSO with respect to the ambient dimension D. The test set is defined as Appendix A.2. The errors are in radians

The results indicate that PSO exhibits a strong dependence on both the number of samples N and the ambient dimension D. This observation aligns with the findings discussed in Section 3.1, where increasing the number of samples helps to better reveal the minimum of the objective function. However, as D increases, the search space becomes increasingly irregular and less suitable for global optimization, as illustrated in Appendix D.4. While increasing the sample size mitigates this issue to some extent, PSO ultimately provides only an approximate recovery of the effective subspace. That is, besides *Squared Sine*, no effective subspace can be reconstructed with PSO for $D = 10$.

In contrast, the Active Subspace method demonstrates consistent performance across varying values of D and N, provided that $N > d_e$. This consistency highlights the scalability advantage of the Active Subspace method, which is largely independent of the ambient dimension. The PSO-based approach, on the other hand, requires repeated evaluations of the objective function at each iteration and suffers from poor scalability with respect to D, rendering it less practical for high-dimensional problems.

Conclusion

Many fields of engineering require the global minimization of objective functions defined over large domains. To ensure tractability, it becomes essential to develop optimization methods that exploit the structure of these functions, thereby accelerating computations. In this context, the starting point of this master thesis was the *low effective dimension* framework, in which a function depends only on a linear subspace of the full domain.

This work focused on two extensions of this framework and evaluated their generalization potential relative to the basic case. The first extension concerns functions with *approximate low effective dimension*, where the function exhibits limited variation in the orthogonal complement of the effective subspace, up to deterministic perturbations. The second extension addresses functions with low effective dimension but subject to noisy function evaluations. Despite the additional computational costs introduced by these generalizations, the adapted methods demonstrate the ability to yield relevant and robust results.

The analyses carried out in this thesis established preliminary theoretical connections between the assumptions required for a function to possess a *low effective dimension*, as in [15], and those required to exhibit an *approximate low effective dimension*. This led to the design and adaptation of algorithms suited to this relaxed setting. The reformulation of certain propositions from [15] enabled an extension of the theoretical framework to encompass functions with approximate structure. It was also shown that the spectrum of the matrix C, as used in [22], remains a valid tool to identify the effective subspace, even under these generalized assumptions. In high-dimensional regimes (e.g., $D = 200$), the proposed algorithms demonstrated competitive performance compared to full-domain solvers.

Several open questions emerged from this analysis. Beyond identifying and optimizing over the *approximate effective subspace*, it would be valuable to study the trade-offs associated with increasing the number of selected directions. Allowing for more directions might improve accuracy or robustness, but at the cost of efficiency. Moreover, numerical instabilities introduced by rotating the canonical axes within the objective function were found to perturb the spectrum of C. It remains to be explored how a randomized testing set could be constructed to minimize such effects,

or how to design such a set with reduced computational burden.

In the presence of noisy function evaluations, the focus shifted to the accurate recovery of the effective subspace rather than solving the related optimization problem. The proposed approach employed Riemannian optimization techniques on the Grassmann manifold, using an objective function analogous to a sum-of-squares formulation. This strategy was motivated by the inadequacy of finite-difference approximations for constructing C in noisy settings. Particle Swarm Optimization (PSO) was adopted due to its suitability for the structure of the objective. This method exhibited higher accuracy in subspace reconstruction, albeit at significant computational cost. Experiments conducted for low-dimensional ambient spaces ($D \leq 5$) revealed promising results, but scalability remains a key challenge. Future investigations should focus on improving the tractability of this approach for higher dimensions, where dimensionality reduction becomes most beneficial.

In conclusion, the results from [15] can be effectively extended to the setting of *approximate low effective dimension*, even when the underlying function displays more complex behaviour. When returning to the original *low effective dimension* setting but allowing only noisy queries, the Riemannian framework provided improved performance in small dimensions. However, further work is needed to make such techniques computationally efficient and scalable for practical applications.

Bibliography

[1] P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, Princeton, NJ, 2008.

[2] Alfred Auslender. *Optimisation : méthodes numériques / par A. Auslender,...* Maîtrise de mathématiques et applications fondamentales. Masson, Paris, 1976.

[3] Seth D. Axen, Mateusz Baran, Ronny Bergmann, and Krzysztof Rzecki. Manifolds.jl: An extensible julia framework for data analysis on manifolds. *ACM Transactions on Mathematical Software*, 49(4), dec 2023.

[4] Francis Bach, Rodolph Jenatton, Julien Mairal, and Guillaume Obozinski. 2012.

[5] Ronny Bergmann. Manopt.jl: Optimization on manifolds in Julia. *Journal of Open Source Software*, 7(70):3866, 2022.

[6] El Houcine Bergou, Eduard Gorbunov, and Peter Richtárik. Stochastic three points method for unconstrained smooth minimization. *SIAM Journal on Optimization*, 30(4):2726–2749, 2020.

[7] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. 13(null):281–305, February 2012.

[8] Dimitri Bertsekas. *Nonlinear Programming*. 01 2003.

[9] Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B Shah. Julia: A fresh approach to numerical computing. *SIAM Review*, 59(1):65–98, 2017.

[10] Pierre B. Borckmans, Mariya Ishteva, and Pierre-Antoine Absil. A modified particle swarm optimization algorithm for the best low multilinear rank approximation of higher-order tensors. In Marco Dorigo, Mauro Birattari, Gianni A. Di Caro, René Doursat, Andries P. Engelbrecht, Dario Floreano, Luca Maria Gambardella, Roderich Groß, Erol Şahin, Hiroki Sayama, and Thomas Stützle, editors, *Swarm Intelligence*, pages 13–23, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[11] N. Boumal, B. Mishra, P.-A. Absil, and R. Sepulchre. Manopt, a Matlab toolbox for optimization on manifolds. *Journal of Machine Learning Research*, 15(42):1455–1459, 2014.

[12] Nicolas Boumal. *An introduction to optimization on smooth manifolds*. Cambridge University Press, 2023.

[13] F. Burstall. Basic riemannian geometry. 05 2012.

[14] Richard H. Byrd, Jorge Nocedal, and Richard A. Waltz. *Knitro: An Integrated Package for Nonlinear Optimization*, pages 35–59. Springer US, Boston, MA, 2006.

[15] Coralia Cartis, Xinzhu Liang, Estelle Massart, and Adilet Otemissov. Learning the subspace of variation for global optimization of functions with low effective dimension, 2024.

[16] Coralia Cartis, Estelle Massart, and Adilet Otemissov. Bound-constrained global optimization of functions with low effective dimensionality using multiple random embeddings. *Mathematical Programming*, 198(1):997–1058, 2023.

[17] Coralia Cartis, Estelle Massart, and Adilet Otemissov. Global optimization using random embeddings. *Mathematical Programming*, 200(2):781–829, 2023.

[18] Coralia Cartis and Adilet Otemissov. A dimensionality reduction technique for unconstrained global optimization of functions with low effective dimensionality. *Information and Inference: A Journal of the IMA*, 11(1):167–201, 05 2021.

[19] Coralia Cartis and Lindon Roberts. Scalable subspace methods for derivative-free nonlinear least-squares optimization. *Mathematical Programming*, 199(1):461–524, 2023.

[20] Chan-Jin Chung and Robert G. Reynolds. Caep: An evolution-based tool for real-valued function optimization using cultural algorithms. *International Journal on Artificial Intelligence Tools*, 07(03):239–291, 1998.

[21] Albert Cohen, Ingrid Daubechies, Ronald DeVore, Gerard Kerkyacharian, and Dominique Picard. Capturing ridge functions in high dimension from point queries. *Constructive Approximation*, 35, 05 2011.

[22] Paul G. Constantine. *Active Subspaces*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2015.

[23] Nicola Demo, Marco Tezzele, and Gianluigi Rozza. A supervised learning approach involving active subspaces for an efficient genetic algorithm in high-dimensional optimization problems. *SIAM Journal on Scientific Computing*, 43(3):B831–B853, 2021.

[24] Elizabeth D. Dolan and Jorge J. Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91(2):201–213, 2002.

[25] Alan Edelman, Tomás A. Arias, and Steven T. Smith. The geometry of algorithms with orthogonality constraints. *SIAM Journal on Matrix Analysis and Applications*, 20(2):303–353, 1998.

[26] Massimo Fornasier, Karin Schnass, and Jan Vybiral. Learning functions of few arbitrary linear parameters in high dimensions. *Foundations of Computational Mathematics*, 12(2):229–262, 2012.

[27] Lukas P. Fröhlich, Edgar D. Klenske, Christian G. Daniel, and Melanie N. Zeilinger. Bayesian optimization for policy search in high-dimensional systems via automatic domain selection, 2020.

[28] Michel Gendreau and Jean-Yves Potvin, editors. *Handbook of Metaheuristics*. International Series in Operations Research & Management Science. Springer, New York, NY, 2 edition, 2010.

[29] Christophe Giraud. *Introduction to High-Dimensional Statistics*. Chapman and Hall/CRC, 2 edition, 2021.

[30] G.H. Golub and C.F. Van Loan. *Matrix Computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, 1996.

[31] Robert M. Gower, Dmitry Kovalev, Felix Lieder, and Peter Richtárik. RSN: Randomized Subspace Newton. In *Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS)*, Vancouver, Canada, 2019. Available at https://telecom-paris.hal.science/hal-02365297v1/file/1905.10874.pdf.

[32] Dmitry Grishchenko, Franck Iutzeler, and Jérôme Malick. Proximal gradient methods with adaptive subspace sampling. *Mathematics of Operations Research*, 46(4):1303–1323, 2021.

[33] Filip Hanzely, Nikita Doikov, Peter Richtárik, and Yurii Nesterov. Stochastic subspace cubic newton method. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020.

[34] Miguel A. Hernán and James M. Robins. *Causal Inference: What If*. Chapman & Hall/CRC, Boca Raton, 2020.

[35] R. Horst, Panos M. Pardalos, and Nguyen Van Thoai. *Introduction to Global Optimization*, volume 48 of *Nonconvex Optimization and Its Applications*. Springer, New York, NY, 2 edition, 2000.

[36] M Jamil and X-S Yang. Literature review of benchmark functions for global optimization problems. *International Journal of Mathematical Modelling and Numerical Optimisation*, 4(2):150–194, 2013.

[37] J. Kennedy. The particle swarm: social adaptation of knowledge. In *Proceedings of 1997 IEEE International Conference on Evolutionary Computation (ICEC '97)*, pages 303–308, 1997.

[38] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Proceedings of ICNN'95 - International Conference on Neural Networks*, volume 4, pages 1942–1948 vol.4, 1995.

[39] James Kennedy. *Particle Swarm Optimization*, pages 760–766. Springer US, Boston, MA, 2010.

[40] Keshav Patel Keval, Vivek Shripad Borkar, and Ananya Singhal. Minkowski descent: An algorithm for stochastic global optimization. In *2024 60th Annual Allerton Conference on Communication, Control, and Computing*, pages 1–8, 2024.

[41] David Kozak, Stephen Becker, Alireza Doostan, and Luis Tenorio. Stochastic subspace descent, 2019.

[42] Ilja Kuzborskij and Csaba Szepesvári. Learning lipschitz functions by gd-trained shallow overparameterized relu neural networks, 2023.

[43] Jonathan Lacotte, Mert Pilanci, and Marco Pavone. *High-dimensional optimization in adaptive random subspaces*. Curran Associates Inc., Red Hook, NY, USA, 2019.

[44] John M. Lee. *Introduction to Smooth Manifolds*, volume 218 of *Graduate Texts in Mathematics*. Springer, New York, NY, 2 edition, 2012. eBook available via SpringerLink. Topics: Differential Geometry.

[45] Tianqi Li, Nan Chen, Zhike Peng, and Qingbo He. Super fourier analysis: A highly efficient framework for multivariate signal processing. *Signal Processing*, 231:109899, 2025.

[46] Leo Liberti, Politecnico Milano, P Zza, and L Vinci. Introduction to global optimization. 03 2006.

[47] Lizhen Lin, Bayan Saparbayeva, and Michael Zhang. Accelerated algorithms for convex and non-convex optimization on manifolds. *Machine Learning*, 114, 02 2025.

[48] Ying Ling, Yongquan Zhou, and Qifang Luo. Lévy flight trajectory-based whale optimization algorithm for global optimization. *IEEE Access*, 5:6168–6186, 2017.

[49] Luo Long, Coralia Cartis, and Paz Fink Shustin. Dimensionality reduction techniques for global bayesian optimisation, 2024.

[50] Marloes H. Maathuis, Markus Kalisch, and Peter Bühlmann. Estimating high-dimensional intervention effects from observational data. *The Annals of Statistics*, 37(6A):3133–3164, December 2009.

[51] Marek Molga and Czeslaw Smutnicki. Test functions for optimization needs. Technical report, AGH University of Science and Technology, 2005. Available online at `http://www.zsd.ict.pwr.wroc.pl/files/docs/functions.pdf`.

[52] Yurii Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. CORE Discussion Paper 2010/2, CORE, Université catholique de Louvain, 2010. Available at `http://hdl.handle.net/2078.1/29221`.

[53] Yurii Nesterov and Vladimir Spokoiny. Random gradient-free minimization of convex functions. *Foundations of Computational Mathematics*, 17(2):527–566, 2017.

[54] Albert Nigrin. *Neural Networks for Pattern Recognition*. The MIT Press, 09 1993.

[55] A Otemissov. *Dimensionality reduction techniques for global optimization*. PhD thesis, University of Oxford, 2020.

[56] Peter Richtárik and Martin Takáč. Parallel coordinate descent methods for big data optimization. *Mathematical Programming*, 156(1):433–484, 2016.

[57] Philippe Rigollet and Jan-Christian Hütter. High-dimensional statistics, 2023.

[58] Yousef Saad. *Numerical Methods for Large Eigenvalue Problems*. Society for Industrial and Applied Mathematics, 2011.

[59] Andrea Saltelli, Marco Ratto, Terry Andres, Francesca Campolongo, Jessica Cariboni, Debora Gatelli, Michaela Saisana, and Stefano Tarantola. *Global Sensitivity Analysis: The Primer*. John Wiley & Sons, 2008.

[60] Ahmed H. Sameh and John A. Wisniewski. A trace minimization algorithm for the generalized eigenvalue problem. *SIAM Journal on Numerical Analysis*, 19(6):1243–1259, 1982.

[61] Hao-Jun Michael Shi, Yuchen Xie, Melody Qiming Xuan, and Jorge Nocedal. Adaptive finite-difference interval estimation for noisy derivative-free optimization. *SIAM Journal on Scientific Computing*, 44(4):A2302–A2321, 2022.

[62] Hao-Jun Michael Shi, Melody Qiming Xuan, Figen Oztoprak, and Jorge Nocedal and. On the numerical performance of finite-difference-based methods for derivative-free optimization. *Optimization Methods and Software*, 38(2):289–311, 2023.

[63] S. U. Stich, C. L. Müller, and B. Gärtner. Optimization of convex functions with random pursuit. *SIAM Journal on Optimization*, 23(2):1284–1309, 2013.

[64] S. Surjanovic and D. Bingham. Virtual library of simulation experiments: Test functions and datasets. Retrieved May 14, 2025, from `http://www.sfu.ca/~ssurjano`.

[65] S. Surjanovic and D. Bingham. Virtual library of simulation experiments: Test functions and datasets, 2013. Accessed: 2025-04-05.

[66] Ramakrishna Tipireddy and Roger Ghanem. Basis adaptation in homogeneous chaos spaces. *Journal of Computational Physics*, 259:304–317, 2014.

[67] James Townsend, Niklas Koep, and Sebastian Weichwald. Pymanopt: A python toolbox for optimization on manifolds using automatic differentiation. *Journal of Machine Learning Research*, 17(137):1–5, 2016.

[68] Hemant Tyagi and Volkan Cevher. Learning non-parametric basis independent models from point queries via low-rank methods. *Applied and Computational Harmonic Analysis*, 37(3):389–412, 2014.

[69] F. van den Bergh and A.P. Engelbrecht. A new locally convergent particle swarm optimiser. In *IEEE International Conference on Systems, Man and Cybernetics*, volume 3, pages 6 pp. vol.3–, 2002.

[70] Martin J. Wainwright. Information-theoretic limits on sparsity recovery in the high-dimensional and noisy setting. *IEEE Transactions on Information Theory*, 55(12):5728–5741, 2009.

[71] Yifei Wang, Yixuan Hua, Emmanuel Candés, and Mert Pilanci. Overparameterized relu neural networks learn the simplest models: Neural isometry and exact recovery, 2023.

[72] Ziyu Wang, Frank Hutter, Masrour Zoghi, David Matheson, and Nando De Freitas. Bayesian optimization in a billion dimensions via random embeddings. *Journal of Artificial Intelligence Research*, 55(1):361–387, 2016.

[73] Stephen J. Wright. Coordinate descent algorithms. *Mathematical Programming*, 151(1):3–34, 2015.

[74] Ke Ye and Lek-Heng Lim. Schubert varieties and distances between subspaces of different dimensions. *SIAM Journal on Matrix Analysis and Applications*, 37(3):1176–1197, 2016.

[75] Miao Zhang, Huiqi Li, and Steven Su. High dimensional bayesian optimization via supervised dimension reduction, 2019.

Appendix A

Test set description

A.1 Transformation methods

A.1.1 Low effective dimension

This section is citation of [15], but is restated here for convenience and completeness. Each function h in Table A.1 is turned into a function f over $\mathbb{R}^D$, for D an integer chosen in a range spanned by [2, 300], with $d_e \leq D$. We proceeds in three steps. First, we apply a suitably chosen linear change of variable to each function of Table A.1, to turn it into a function $\bar{h}$ whose domain is $[-1, 1]^{d_e}$. Then, we add $D - d_e$ fake dimensions in the search space, with zero coefficient:

$$h(x) = \bar{h}(\bar{x}) + 0 \cdot x_{d_e+1} + \cdots + 0 \cdot x_D.$$

Finally, in order to obtain a non-trivial constant subspace, we rotate the function $h(x)$ by applying random orthogonal matrix Q to x. The D-dimensional function used in the tes set is then given by

$$f(x) = h\left(Q^T x\right).$$

Note that, by construction, the first d_e colmuns of Q form a basis of the effective subspace $\mathcal{S}$ of f, and the last $D - d_e$ columns of Q form a basis of the constant subspace $\mathcal{S}^\perp$ of f.

A.1.2 Approximate low effective dimension

We generalize the construction of functions with low effective dimension presented in Appendix B of [15], and reminded in Appendix A.1.1.

Assume that one wants to construct a function $f : \mathbb{R}^D \to \mathbb{R}$ from a function $g : \mathbb{R}^{d_e} \to \mathbb{R}$ taken in Appendix A.2. Three steps are considered.

First step. A suitable change of variable is applied on g to map its domain to the hypercube $[-1, 1]^{d_e}$. The resulting function is called $\bar{g}$. Consider only one variable x_i

such that $x_i \in [a_i, b_i]$. Consider $\xi_i \in [-1, 1]$. Then a change of variable $\zeta_i : \xi_i \to x_i$ is given by

$$\zeta_i : [-1, 1] \to [a_i, b_i] : \xi \to \frac{(b_i - a_i)}{2}\xi - \frac{(b_i + a_i)}{2},$$

and one can define ζ as

$$\zeta : [-1, 1]^{d_e} \to \mathrm{dom}(g) : (\xi_1, \ldots, \xi_{d_e}) \to (\zeta_1(\xi_1), \ldots, \zeta_D(\xi_D)).$$

The function $\bar{g}$ has thus the following definition

$$\bar{g} : [-1, 1]^{d_e} \to \mathbb{R} : x \to (g \circ \zeta)(x).$$

Second step. This step consists of adding $D - d_e$ low effective dimensions, that is, their variation is small in the considered dimension. The extension considered is

$$h(x) : [-1, 1]^D \to \mathbb{R} : \bar{g}(\bar{x}) + \sum_{i=d_e+1}^{D} A_i \sin(\omega_i x_i + \phi_i),$$

with $\bar{x}$ a restriction to the first d_e-th components of $x \in \mathbb{R}^D$. By default, one can set

$$\begin{aligned} A_i &= \frac{L}{D - d_e}, \\ \phi_i &\sim \text{ Context-dependant}, \\ \omega_i &= 1, \end{aligned}$$

for $i = d_e + 1, ..., D$. Those choices ensure that in the approximate low effective dimension, one knows the constant L in the characterization

$$|f(x) - f(x + Vu)| \leq L\|u\|,$$

with $V \in \mathbb{R}^{D \times (D - d_e)}$ and $u \in \mathbb{R}^{D - d_e}$. Any other choice of parameter works, and are relevant, as long as $L \ll 1$, for the approximate low effective dimension hypothesis to hold.

Third step. The final step is used to ensure that one does not always end up with the canonical axes as invariant subspace. Generate $Q \in \mathcal{O}(D)$, the set of orthogonal matrices, and use

$$f : [-1, 1]^D \to \mathbb{R} : x \to h(Q^T x).$$

Note that by definition, the d_e-th first columns of Q span the effective subspace, and the $(D - d_e)$-th last columns span the approximate low effective dimension subspace.

A.2 Test functions

All the functions described in this section are summarized in Table A.1. All of them were taken from [65].

Function	Domain	Global minimum
Squared sinus	$[-4,\ 4]$	$h(y^*) = 0$
Styblinski-Tang 2D	$[-5,\ 5]^2$	$h(y^*) = -78.332$
Styblinski-Tang 8D	$[-5,\ 5]^8$	$h(y^*) = -313.329$
Beale	$[-4.5,\ 4.5]^2$	$h(y^*) = 0$
Hartman 3D	$[0,\ 1]^3$	$h(y^*) = -3.86278$
Hartman 6D	$[0,\ 1]^6$	$h(y^*) = -3.32237$
Trid	$[-25,\ 25]^5$	$h(y^*) = -30$
Goldstein-Price	$[-2,\ 2]^2$	$h(y^*) = 3$
Levy	$[-10,\ 10]^6$	$h(y^*) = 0$
Rosenbrock	$[-2.048,\ 2.048]^7$	$h(y^*) = 0$

Table A.1

The following four functions are used as a test set in Chapter 3.

Function	Domain	Global minimum
Squared sinus	$[-4,\ 4]$	$h(y^*) = 0$
Beale	$[-4.5,\ 4.5]^2$	$h(y^*) = 0$
Goldstein-Price	$[-2,\ 2]^2$	$h(y^*) = 3$
Levy	$[-10,\ 10]^2$	$h(y^*) = 0$

Table A.2

The rest of this section is dedicated to presenting the analytical expression of the functions described in Table A.1.

A.2.1 Squared Sine

The function is defined by

$$f : [-4,\ 4] \to \mathbb{R} : (x,\ y) \to \sin\left(x - 0.5\right)^2,$$

with $d_e = 1$. Throughout this master thesis, the following variant is used as well,

$$f : \mathbb{R}^2 \to \mathbb{R} : (x,\ y) \to \sin\left(x - y - 0.5\right)^2, \tag{A.1}$$

with $d_e = 1$. This function is already rotated and is used for illustration purposes.

A.2.2 Styblinski-Tang

The function is defined by

$$f : [-5,\ 5]^{d_e} \to \mathbb{R} : x \to \frac{1}{2}\sum_{i=1}^{d_e}\left(x_i^4 - 16x_i^2 + 5x_i\right) \tag{A.2}$$

A.2.3 Beale

The function is defined by

$$\begin{aligned} f : [-1,\, 1]^2 \to \mathbb{R} : x \to & \left(1.5 - x_1 + x_1 x_2\right)^2 \\ & + \left(2.25 - x_1 + x_1 x_2^2\right)^2 \\ & + \left(2.625 - x_1 + x_1 x_2^3\right)^2 . \end{aligned} \tag{A.3}$$

A.2.4 Levy

The function is defined by

$$\begin{aligned} & f : [-10,\, 10]^{d_e} \to \mathbb{R} \\ & : x \to \sin^2(\pi w_1) + \sum_{i=1}^{d_e - 1} (w_i - 1)^2 \left[1 + 10 \sin^2(\pi w_i + 1)\right] \\ & \quad + (w_{d_e} - 1)^2 \left[1 + \sin^2(2\pi w_{d_e})\right], \end{aligned} \tag{A.4}$$

with

$$w_i = 1 + \frac{x_i - 1}{4}, \quad \text{for } i = 1, \ldots, d_e.$$

A variant is also used in this master thesis, that is

$$\begin{aligned} & f : [-10, 10]^2 \to \mathbb{R} : x \to \sin^2(\pi w_1) + (w_1 - 1)^2 \left[1 + \sin^2(2\pi w_1)\right], \\ & w_1 = 1 + \frac{x_1 - x_2 - 1}{4}. \end{aligned} \tag{A.5}$$

A.2.5 Goldstein-Price

The function is defined by

$$\begin{aligned} f : [-1,\, 1]^2 \to \mathbb{R} : x \to & \left[1 + (x_1 + x_2 + 1)^2 \left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1 x_2 + 3x_2^2\right)\right] \\ & \times \left[30 + (2x_1 - 3x_2)^2 \left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1 x_2 + 27x_2^2\right)\right] \end{aligned} \tag{A.6}$$

A.2.6 Rosenbrock

The function is defined by

$$f : \mathbb{R}^d \to \mathbb{R} : x \mapsto \sum_{i=1}^{d-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right]. \tag{A.7}$$

A.2.7 Hartman 6D

The function is defined by

$$f(x) = -\sum_{i=1}^{4} \alpha_i \exp\left(-\sum_{j=1}^{6} A_{ij}(x_j - P_{ij})^2\right), \tag{A.8}$$

where

$$\boldsymbol{\alpha} = \begin{pmatrix} 1.0 & 1.2 & 3.0 & 3.2 \end{pmatrix}^T, \quad A = \begin{pmatrix} 10 & 3 & 17 & 3.5 & 1.7 & 8 \\ 0.05 & 10 & 17 & 0.1 & 8 & 14 \\ 3 & 3.5 & 1.7 & 10 & 17 & 8 \\ 17 & 8 & 0.05 & 10 & 0.1 & 14 \end{pmatrix},$$

$$P = 10^{-4} \times \begin{pmatrix} 1312 & 1696 & 5569 & 124 & 8283 & 5886 \\ 2329 & 4135 & 8307 & 3736 & 1004 & 9991 \\ 2348 & 1451 & 3522 & 2883 & 3047 & 6650 \\ 4047 & 8828 & 8732 & 5743 & 1091 & 381 \end{pmatrix}.$$

A.2.8 Hartman 3D

The function is defined by

$$f(x) = -\sum_{i=1}^{4} \alpha_i \exp\left(-\sum_{j=1}^{3} A_{ij}(x_j - P_{ij})^2\right), \tag{A.9}$$

where

$$\boldsymbol{\alpha} = \begin{pmatrix} 1.0 & 1.2 & 3.0 & 3.2 \end{pmatrix}^T, \quad A = \begin{pmatrix} 3.0 & 10 & 30 \\ 0.1 & 10 & 35 \\ 3.0 & 10 & 30 \\ 0.1 & 10 & 35 \end{pmatrix}, \quad P = 10^{-4} \times \begin{pmatrix} 3689 & 1170 & 2673 \\ 4699 & 4387 & 7470 \\ 1091 & 8732 & 5547 \\ 381 & 5743 & 8828 \end{pmatrix}.$$

A.2.9 Trid

The function is defined by

$$f : \mathbb{R}^d \to \mathbb{R} : x \mapsto \sum_{i=1}^{d} (x_i - 1)^2 - \sum_{i=1}^{d} x_i x_{i-1}. \tag{A.10}$$

Appendix B

Algorithm for functions with low effective dimension

We recall here for consistency the algorithm presented in [15] for functions with low effective dimension. Assume that we aim to solve

$$f^* = \min_{x \in \mathbb{R}^D} f(x),$$

where $f : \mathbb{R}^D \to \mathbb{R}$ is a real-valued continuously differentiable, possibly non-convex, deterministic function. Assume further that f has low effective dimensionality. One can use the following algorithm to efficiently solve this optimization problem.

Algorithm 11 ASM-1

1: Initialize $p \in \mathbb{R}^D$ and M, a number of sample gradients
2: Apply Algorithm 1 with M samples to compute $\hat{C}$.
3: Compute a basis A of the range of $\hat{C}$, and let d be its dimension.
4: Calculate y by solving approximately, and possibly, with a given probability,

$$\min_{y \in \mathbb{R}^D} f(Ay + p)$$

5: Construct the solution estimate, $x = Ay + p$.

Appendix C

Additional theoretical results for deterministic perturbations

Proposition C.1. *Let C be defined as in Equation (1.1). Then,*

$$\min_{\substack{V\in\mathbb{R}^{D\times(D-d_e)}\\ V^TV=I_{D-d_e}}} \operatorname{trace}(V^TCV) = \sum_{i=d_e+1}^{D} \lambda_i$$

where λ_i, $i = d_e+1,\dots,D$ are the $D-d_e$ smallest eigenvalues of C.

Proof. The framework adopted here is the Riemannian optimization on the Stiefel manifold. We aim to solve the following optimization problem

$$\min_{\substack{V\in\mathbb{R}^{D\times(D-d_e)}\\ V^TV=I_{D-d_e}}} \operatorname{trace}(V^TCV),$$

which is defined as the minimization of the following objective function

$$F : \mathrm{St}(D, D-d_e) \to \mathbb{R} : V \to \operatorname{trace}\left(V^TCV\right),$$

with $\mathrm{St}(n,p)$ defined by

$$\mathrm{St}(n,p) = \left\{X \in \mathbb{R}^{n\times p} \;\middle|\; X^TX = I_p,\ p \le n\right\}.$$

The gradient of a real-valued function defined on $\mathrm{St}(n,p)$ is given by

$$\nabla F\left(V\right) = F_V - VF_V^TV,$$

with F_V the Euclidean gradient of F, such that

$$\left(F_V\right)_{ij} = \frac{\partial F}{\partial V_{ij}}.$$

Let $Y \in \mathbb{R}^{D\times(D-d_e)}$ arbitrary. It follows that

$$\begin{aligned}
\mathrm{D}F\left(V\right)\left[Y\right] &= \left\langle F_V,\, Y\right\rangle \\
&= \lim_{h\to 0} \frac{\operatorname{trace}\left[\left(V+hY\right)^T C\left(V+hY\right)\right] - \operatorname{trace}\left[V^T C V\right]}{h} \\
&= 2\operatorname{trace}\left(Y^T C V\right) \\
&= \left\langle 2CV,\, Y\right\rangle .
\end{aligned} \tag{C.1}$$

We infer that $F_V = 2CV$, and

$$\nabla F\left(V\right) = 2CV - 2VV^T CV.$$

Any stationary points must satisfy

$$CV = VV^T CV.$$

We recognize an eigendecomposition structure. If we choose V such that it contains orthonormal eigenvectors of C, then the equality is satisfied.

$$CV = VV^T CV \;\Leftrightarrow\; CV = V\Lambda,$$

with Λ a diagonal matrix containing the eigenvalues of C associated with the chosen eigenvectors in V. Turns out that with this choice of V

$$\operatorname{trace}(V^T CV) = \sum_i \lambda_i,$$

for the chosen $i \in \{1,\ldots,D\}$. The minimum is thus obtained by taking as columns of V the $D-d_e$ last eigenvectors of V (associated to the $D-d_e$ lowest eigenvalues). □

Example. As an illustration of Proposition C.1, assume that

$$C = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix}.$$

Then,

$$F : \mathrm{St}(2,1) \to \mathbb{R} : \begin{pmatrix} \sin\theta \\ \cos\theta \end{pmatrix} \to \begin{pmatrix} \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix} \begin{pmatrix} \sin\theta \\ \cos\theta \end{pmatrix} = 1 + \cos^2\theta. \tag{C.2}$$

By abuse of notation, we consider $F(\theta)$ directly. The solution is given by $F(\theta^*) = 1$, with $\theta^* = \pi/2$. Looking at the corresponding vector, $v^* = \begin{pmatrix} 1 & 0 \end{pmatrix}$, which is exactly the eigenvector associated to the smallest eigenvalue of C. As an illustration of the objective function, see Figure C.1.

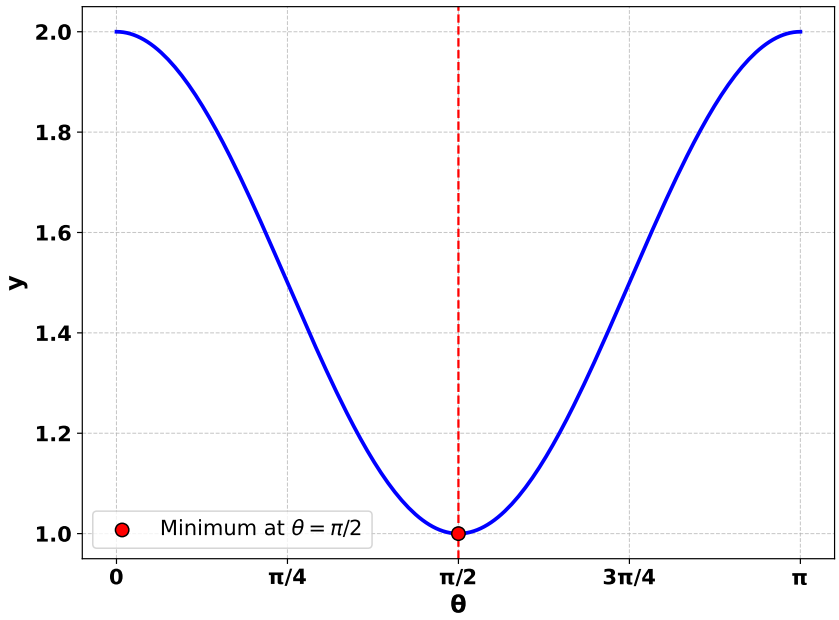


Figure C.1: See that the function C.2 admits two extrema. One is the minimum at $\theta = \pi/2$. It spans $v = \begin{pmatrix} 1 & 0 \end{pmatrix}$. The other is the maximum, spanning $v = \begin{pmatrix} 0 & 1 \end{pmatrix}$.

Appendix D

Additional material about Riemannian optimization

D.1 Smoothness of the objective function

Figure D.1 illustrates the process required to prove the smoothness of an objective function on Grassmann manifolds.

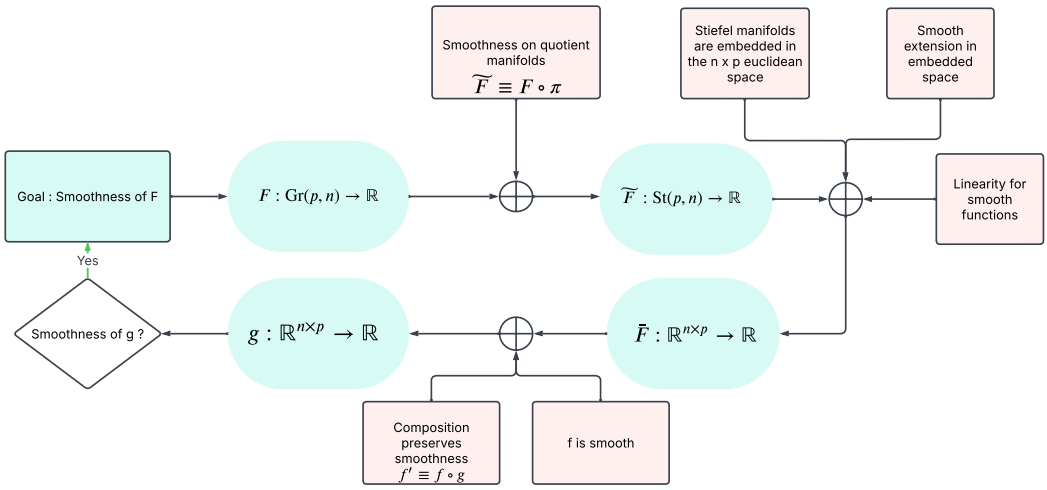


Figure D.1: The convention is (p, n) is a tuple associated with $n \times p$ orthonormal frames. For consistency, note that $(D, d_e) \equiv (p, n)$ with $D = n$ and $p = d_e$.

D.2 Details about the derivation of the gradient

The objective of this note is to compute the riemannian gradient of the objective function on the Grassmann manifold. Define F_Y such that

$$(F_Y)_{ij} = \frac{\partial F}{\partial Y_{ij}}, \tag{D.1}$$

that is, an euclidean differentiation in the euclidean space $\mathbb{R}^{D\times d_e}$. Define

$$g_i : \mathbb{R}^{D\times d_e} \to \mathbb{R}^D : Y \to YY^T x^{(i)}, \tag{D.2}$$

and

$$F_i : \mathrm{Gr}(D, d_e) \to \mathbb{R} : Y \to \left[f\left(YY^T x^{(i)}\right) - f\left(x^{(i)}\right)\right]^2 \tag{D.3}$$

This redefinition is justified by the linearity of the gradient. One can then only consider one term of the sum defining the objective function (3.13).

We are now interested in finding the gradient of F as defined in (D.1). Denote by $\dot{Y}$ an arbitrary direction in $\mathbb{R}^{D\times d_e}$. The directional derivative of F_i at the point Y in the direction $\dot{Y}$ is such that

$$\mathrm{D}F_i\left(Y\right)\left[\dot{Y}\right] = \langle F_Y^i, \dot{Y}\rangle. \tag{D.4}$$

It follows that

$$\mathrm{D}F_i\left(Y\right)\left[\dot{Y}\right] = 2\left(f(YY^T x^{(i)}) - f(x^{(i)})\right)\mathrm{D}f(YY^T x^{(i)})[\dot{Y}]. \tag{D.5}$$

Using the parametrization defined in (D.2) as well as the chain rule (see Appendix A of [1]), note that

$$\mathrm{D}f(YY^T x^{(i)})[\dot{Y}] = \mathrm{D}f(g(Y))\left[Dg(Y)[\dot{Y}]\right] = \langle \nabla f(g(Y)),\, \mathrm{D}g(Y)[\dot{Y}]\rangle. \tag{D.6}$$

Using the definition of g as in (D.2), one can retrieve the expression of the directional derivative by direct computation,

$$\begin{aligned}
\mathrm{D}\left(g(Y)\right)\left[\dot{Y}\right] &= \lim_{h\to 0} \frac{g(Y+h\dot{Y}) - g(Y)}{h} \\
&= \lim_{h\to 0} \frac{\left(Y+h\dot{Y}\right)\left(Y+h\dot{Y}\right)^T - YY^T}{h} x \\
&= \lim_{h\to 0} \frac{hY\dot{Y}^T + h\dot{Y}Y^T + h^2\dot{Y}\dot{Y}^T}{h} x \\
&= Y\dot{Y}^T x + \dot{Y}Y^T x.
\end{aligned} \tag{D.7}$$

With this computation, one can compute the directional derivative of $f(YY^T x^{(i)})$ in the direction $\dot{Y}$. Then, one can express the directional derivative of F_i in the direction $\dot{Y}$. Let $k = 2\left(f(YY^T x^{(i)}) - f(x^{(i)})\right)$. Then,

$$\begin{aligned}
\mathrm{D}F_i\left(Y\right)\left[\dot{Y}\right] &= k\mathrm{D}f(YY^T x^{(i)})[\dot{Y}] \\
&= k\mathrm{D}f(g_i(Y))[\mathrm{D}g(Y)[\dot{Y}]] \\
&= k\langle \nabla f(g_i(Y)),\, \mathrm{D}g_i(Y)[\dot{Y}]\rangle \\
&= k\nabla f(YY^T x^{(i)})^T\left(\dot{Y}Y^T x^{(i)} + Y\dot{Y}^T x^{(i)}\right) \\
&= k\,\mathrm{tr}\left(\nabla f(YY^T x^{(i)})^T\left(\dot{Y}Y^T x^{(i)} + Y\dot{Y}^T x^{(i)}\right)\right) \\
&= k\left\{\mathrm{tr}(Y^T x^{(i)}\nabla f(YY^T x^{(i)})\dot{Y}) + \mathrm{tr}((x^{(i)}\nabla f(YY^T x^{(i)})Y)^T\dot{Y}\right) \\
&= k\,\mathrm{tr}\left\{\left[\nabla f(YY^T x^{(i)})^T x^{(i)\,T} Y + x^{(i)}\nabla f(YY^T x^{(i)})Y\right]^T\dot{Y}\right\}
\end{aligned} \tag{D.8}$$

By identification according to (D.4), and by additivity property, one has that

$$F_Y = 2\sum_{i=1}^{N} \left(f(YY^T x^{(i)}) - f(x^{(i)})\right) \left[\nabla f(YY^T x^{(i)})^T x^{(i)\,T} Y + x^{(i)} \nabla f(YY^T x^{(i)}) Y\right] \tag{D.9}$$

Knowing the euclidean gradient, the riemannian gradient is immediate, that is

$$\nabla F(Y) = F_Y - YY^T F_Y \tag{D.10}$$

D.3 Validation of a candidate subspace through the objective function

Consider the objective function

$$F(Y) = \sum_{i=1}^{N} \left[f\left(YY^T x^{(i)}\right) - f\left(x^{(i)}\right)\right]^2.$$

We still assume to have only access to f through noisy queries

$$\tilde{f}(x) = f(x) + \varepsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2).$$

The objective function becomes

$$\tilde{F}\left(Y\right) = \sum_{i=1}^{N} \left[\tilde{f}\left(YY^T x^{(i)}\right) - \tilde{f}\left(x^{(i)}\right)\right]^2.$$

Assume that Y is the correct $D \times d_e$ orthonormal frames spanning the invariant subspace of F, that is, the optimal solution of

$$\min_{\substack{Y \in \mathbb{R}^{D \times d_e} \\ Y^T Y = I_{d_e}}} \sum_{i=1}^{N} \left[f\left(YY^T x^{(i)}\right) - f\left(x^{(i)}\right)\right]^2.$$

At optimality, we can thus define the following random variable

$$X = \sum_{i=1}^{N} \left(\varepsilon_i' + \varepsilon_i''\right)^2, \quad \text{and} \quad \varepsilon_i', \varepsilon_i'' \sim \mathcal{N}(0, \sigma^2), \quad \text{i.i.d..}$$

We are thus interested in finding the expectation and the variance of the above quantity. If the value is drawn from X with high probability, then we can consider that we converged toward the optimal solution. Remember that

$$\begin{gathered} Y \sim \mathcal{N}(\mu_Y, \sigma_Y^2), \\ Z \sim \mathcal{N}(\mu_Z, \sigma_Z^2), \\ Y + Z \sim \mathcal{N}(\mu_Y + \mu_Z, \sigma_Y^2 + \sigma_Z^2). \end{gathered}$$

Then

$$X = \sum_{i=1}^{N} X_i^2, \quad X_i \sim \mathcal{N}(0,\, 2\sigma^2).$$

A squared normally distributed random variable has a Chi-Squared distribution, that is, $X_i \sim 2\sigma^2\chi_1^2$. By remembering the expectation and the variance of a chi-squared distribution, along with the sum, one gets that

$$\mathbb{E}[X] = 2N\sigma^2, \quad \mathbb{V}[X] = 8N^2\sigma^4.$$

As an example, see for $N \in \{10,\, 100\}$ and $\sigma^2 = 10^{-4}$ the probability density function of the associated X on Figure D.2.

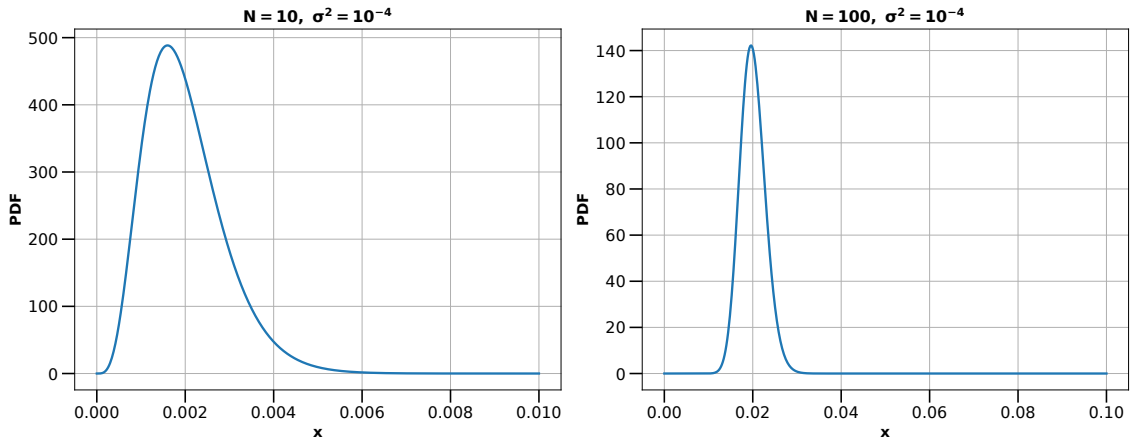


Figure D.2: Probability Density Function (pdf) of the random variable X at optimality

On Table D.1 are displayed the expectations and variances of X for different number of samples and variances of noise.

	$\sigma^2 = 10^{-4}$		$\sigma^2 = 10^{-8}$		$\sigma^2 = 10^{-12}$	
	$\mathbb{E}[X]$	$\mathbb{V}[X]$	$\mathbb{E}[X]$	$\mathbb{V}[X]$	$\mathbb{E}[X]$	$\mathbb{V}[X]$
$N = 10$	0.002	8×10^{-6}	2×10^{-7}	8×10^{-14}	2×10^{-11}	8×10^{-22}
$N = 100$	0.02	8×10^{-4}	2×10^{-6}	8×10^{-12}	2×10^{-10}	8×10^{-20}

Table D.1: Expectation and Variance of the objective function at optimality

The quantities presented on Table D.1 provide a way to verify the optimality of the solution found using the riemannian approach. On each scenario, based on the expectation and the variance, one is able to derive confidence interval for a proposed solution, as a correct solution would be centered around its expectation with high probability.

D.4 Low-dimensional Illustration in 3D

This section aims to present the evolution of the objective function for *Squared Sine*, for $d_e = 1$ and $D = 3$.

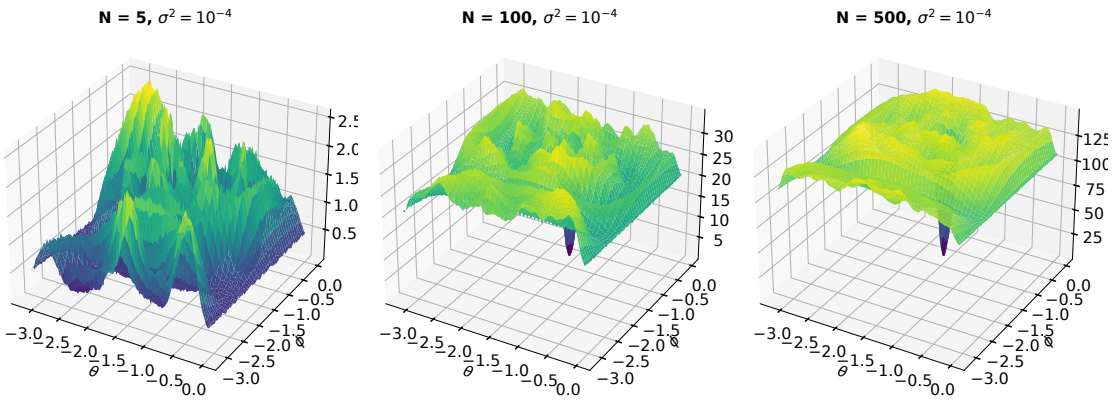


Figure D.3: $\sigma^2 = 10^{-4}$

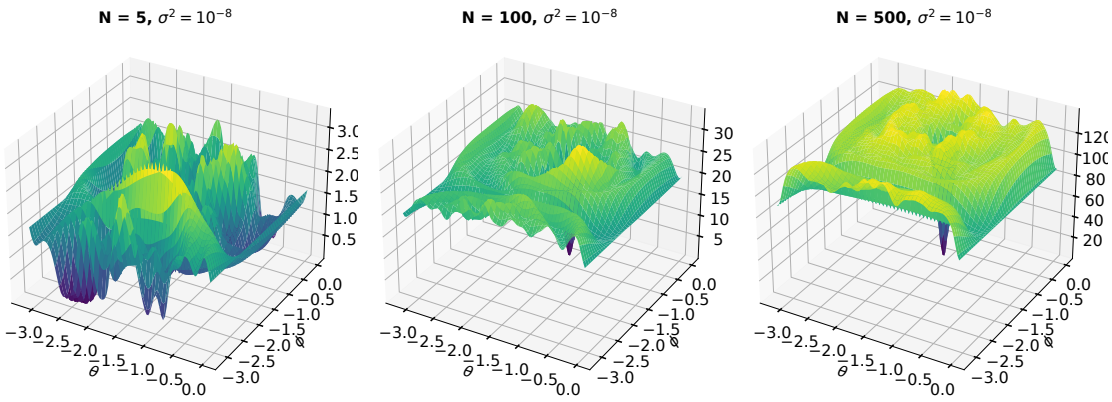


Figure D.4: $\sigma^2 = 10^{-8}$

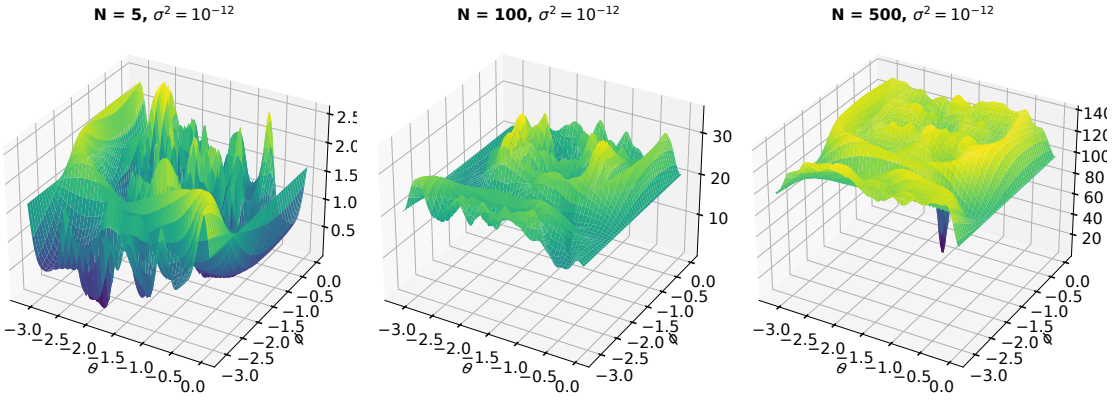


Figure D.5: $\sigma^2 = 10^{-12}$

Appendix E

Use of AI

ChatGPT and DeepL were used to assist with grammar and language refinement. Example of ChatGPT prompt used in this master thesis : "*Correct the grammar and the scientifical soundness of this paragraph. Do not alter the content in any way, focus on the rephrasing and structure only.*"

During the course of this project, GitHub Copilot was used as a coding assistant to streamline repetitive implementation tasks. Its contribution remained strictly limited to low-level programming support and did not influence any theoretical developments or methodological decisions.

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
École polytechnique de Louvain
Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve, Belgique | **www.uclouvain.be/epl**