THÈSE DE DOCTORAT DE

L'UNIVERSITÉ DE RENNES

ÉCOLE DOCTORALE Nº 601
Mathématiques, Télécommunications, Informatique, Signal, Systèmes, Électronique
Spécialité : *Informatique*

Par
Karim TIT

Reliability of Deep Learning with Rare Event Simulation : Theory and Practice.

Fiabilité du Deep Learning avec simulation d'événements rares : théorie et pratique.

Thèse présentée et soutenue à Inria, Rennes, le 22 avril 2024
Unité de recherche : IRISA, CNRS, UMR 6074

Remerciements/Acknowledgement

En premier lieu, je tiens à remercier mes directeurs de thèse Teddy Furon et Mathias Rousset ainsi que mon encadrant Louis-Marie Tranouez, pour la précieuse opportunité qu'ils m'ont offert en me confiant ce projet de recherche. Les nombreuses discussions communes qui ont jalonnées ces trois années de travail m'ont profondément et durablement marqué, et continueront de nourrir mes réflexions, sans doute pour le restant de ma carrière. Teddy, partager un bureau avec toi pendant si longtemps, a été une chance inouie que j'ai encore du mal à estimer, tant l'événement est rare et précieux. Sans la pédagogie remarquable dont tu as fait preuve, en particulier au début de ma thèse, il m'aurait été très difficile, voire impossible, de m'approrier ce projet de thèse très ambitieux et réellement interdisciplinaire. Mathias, ta patience, ta disponibilité et bien sûr tes connaissances abondantes en mathématiques ont jouées un rôle intégrale au bon déroulement de ce projet. Louis-Marie, ton encadrement au sein de la Ruche m'a permis de rapidement prendre en main de nombreuses applications de Réseaux de Neurones et de progresser en MLOps.

Ensuite, je souhaite exprimer ma profonde gratitude envers les membres éminents de mon jury de thèse, qui ont accepté d'honorer ce travail de leur temps et de leur expertise.

Au sein de Thales, je suis reconnaissant à mon manager Xavier, ainsi qu'à Catherine et Erwan qui ont tout fait pour que mon temps passé à la Ruche soit confortable et productif. Je suis aussi reconnaissant aux différents stagiaires et apprentis passé par l'équipe IA avec qui j'ai pu passé de très bons moments professionels : Nicolas, Agathe, Baptiste, Matthieu, Charles, Clément, Léo, ... et ce malgré le télé-travail auquel la Covid nous aura soumis.

Au sein de l'équipe LinkMedia, je voudrais d'abord remercier Aurélie Patier et Laurent Amsaleg pour votre acceuil et le suppport indispensable, tant sur le plan scientifique qu'administratif, que vous m'avez offert. Je tiens ensuite à remercier Thibault Maho et Benoît Bonnet sans qui les portes de la salle de Baby-Foot me seraient restées étrangères. Plus sérieusement, votre bienveillance à mon égard a été cruciale. Of course I want to thank the friends I made here: Hugo, Kassem, Morganne, Gautier, Pierre, Victor, ...

Finally, for their unwavering support and love, I would like to thank my family, my parents, my sister and my brothers, my close friends, and of course, my dear wife Angelina.

Table of Contents

Résumé substantiel en français

Fiabilité du Deep Learning avec simulation d'événements rares : théorie et pratique.

Avant-propos : La présente thèse explore des méthodologies avancées pour évaluer la fiabilité des prédictions de réseaux neuronaux profonds (DNNs), en se concentrant sur leur robustesse face à des perturbations aléatoires. Elle présente des contributions significatives dans l'application des algorithmes de simulation d'événements rares pour la robustesse des DNNs. Le résumé qui suit constitue un essai de reconstitution synthétique du contexte, des concepts et des contributions de cette thèse dans la langue de Voltaire.

Contexte Technologique et Motivation

L'évaluation de la fiabilité des systèmes d'ingénierie complexes est une tâche critique pour garantir que les innovations technologiques ne se traduisent pas par des risques accrus pour les individus et les sociétés. Dans une ère où la technologie de l'Intelligence Artificielle (IA) devient omniprésente dans divers secteurs et est prédite de perturber plusieurs domaines critiques pour la sécurité, la quantification rigoureuse de la fiabilité des systèmes pilotés par l'IA est d'une importance capitale. Puisque de nombreuses applications modernes de l'IA s'appuient principalement sur des modèles de Deep Learning, il est judicieux de prioriser l'évaluation de leur fiabilité.

Le Deep Learning englobe une variété de modèles d'apprentissage automatique conçus pour relever un large éventail de défis en grande dimension. Ces tâches incluent, mais ne se limitent pas à, la vision par ordinateur (CV), le traitement automatique du langage naturel (NLP) et la reconnaissance vocale. Ces différents modèles partagent généralement quatre caractéristiques clés :

1. L'utilisation d'architectures 'compositionnelles' profondes, où le modèle est le résultat de la composition fonctionnelle ou de l'empilement de couches neuronales.

2. L'utilisation de matériel massivement parallèle (par exemple, les GPU) pour accélérer l'inférence.
3. L'utilisation de la Différentiation Automatique (AD) pour calculer les gradients du modèle.
4. L'utilisation d'algorithmes de Descente de Gradient Stochastique (SGD) et de leurs variantes pour optimiser les paramètres du modèle.

Parmi les modèles de Deep Learning, les Réseaux Neuronaux Artificiels Profonds, également connus sous le nom de Deep Neural Networks (DNNs), occupent une position de premier plan. L'objectif principal de cette thèse est d'étudier la fiabilité de ces modèles en utilisant des techniques d'échantillonnage spécifiques appelées algorithmes de Simulation d'Événements Rares. En effet, bien que les DNNs représentent actuellement l'état de l'art dans des domaines tels que la vision par ordinateur, la reconnaissance vocale et le traitement automatique du langage naturel, leur application dans des systèmes critiques est encore limitée en raison d'un manque de garanties de robustesse face aux perturbations des données d'entrées (inputs). Cette préoccupation en matière de robustesse concerne tant le bruit conçu de manière adversariale que les corruptions aléatoires.

L'accent principal de cette thèse est mis sur l'évaluation de l'impact des perturbations *aléatoires*. Nous avons donc développé de nouvelles méthodes pour mesurer précisément la fiabilité des DNNs lorsqu'ils sont confrontés à différentes distributions de bruit, en tirant parti des caractéristiques communes des modèles de Deep Learning mentionnées ci-dessus.

Contexte Scientifique

Cette thèse offre une exploration approfondie de trois domaines scientifiques : l'Ingénierie de Fiabilité Statistique, la Simulation d'Événements Rares et la Robustesse des Réseaux de Neurones Profonds (DNNs). Chaque domaine joue un rôle important dans l'avancement de la compréhension des systèmes complexes, avec un focus particulier dans la dernière partie dans sur l'application aux DNNs. Nous proposons des introductions plutôt larges à ces différents sujets tout en soulignant leur rôle dans le développement de méthodes efficaces d'évaluation de la fiabilité pour les modèles d'Apprentissage Profond.

Les trois premiers chapitres de cette thèse sont consacrés aux concepts et méthodes fondamentaux dans les trois domaines scientifiques mentionnés ci-dessus. Nous commençons par donner de courts résumés ci-dessous, suivis par des présentations plus approfondies du contenu de chaque chapitre :

Chapitre 1 : - Ce chapitre offre une introduction au domaine de l'ingénierie de fiabilité statistique, discutant de ses principes fondamentaux et méthodes avancées (pour les modèles statiques), et préparant le terrain pour leur application dans divers domaines technologiques.

Chapitre 2 : - Le chapitre se penche sur les méthodologies et l'importance de la simulation d'événements rares. Il couvre diverses techniques et leurs applications plus larges, en mettant l'accent sur leur rôle dans la compréhension et l'atténuation des risques associés aux événements rares.

Chapitre 3 : - Se concentrant sur les DNNs, ce chapitre examine les défis de la robustesse adversarial, le développement de mécanismes défensifs et le concept de robustesse certifiée. Il fournit une exploration détaillée des approches formelles et statistiques pour assurer la fiabilité des DNN, en tirant parti des caractéristiques partagées de ces modèles.

Ingénierie de Fiabilité Statistique

L'exploration commence par une introduction approfondie à l'Ingénierie de Fiabilité Statistique (SRE), une discipline fondamentale pour l'évaluation et l'amélioration de la fiabilité des systèmes à travers divers champs d'ingénierie.

Principes fondamentaux : Nous introduisons d'abord les principes de base et les méthodologies de l'ingénierie de fiabilité, traditionnellement appliqués dans des domaines tels que l'ingénierie mécanique et structurelle. La discussion englobe l'évolution de ces principes et leur applicabilité large, préparant le terrain pour leur pertinence dans les contextes technologiques modernes, y compris les systèmes d'Apprentissage Profond. L'accent est mis sur les concepts statistiques fondamentaux qui sous-tendent les évaluations de fiabilité, cruciaux pour comprendre les comportements des systèmes sous incertitude et les modes de défaillance potentiels.

Estimateurs de fiabilité : Un examen détaillé des méthodes classiques d'estimation en ingénierie de fiabilité est ensuite présenté. Cela inclut un regard approfondi sur des méthodes comme la méthode de Fiabilité d'Ordre Premier (FORM) et la méthode de Fiabilité d'Ordre Second (SORM), qui offrent des approximations probabilistes simples pour l'évaluation des défaillances. De plus, une exploration de la technique de *Line Sampling* est incluse, fournissant un aperçu de son application potentielle dans l'analyse de fiabilité statistique complexe et de haute dimension.

Simulation d'Événements Rares

Après l'introduction à l'ingénierie de fiabilité, la thèse fait la transition vers le domaine de la Simulation d'Événements Rares, une technique clé dans l'étude des événements extrêmes.

Concepts et défis : Nous introduisons en premier lieu le concept de simulation d'événements rares et présentons les principaux défis techniques associés à leurs applications. La nature statistique précise des événements rares est discutée, soulignant les limitations et l'inefficacité des méthodes de simulation et d'analyse conventionnelles. Nous pouvons alors introduire des méthodes spécifiques de simulation d'événements rares comme des outils puissants pour modéliser et étudier les événements rares, en soulignant leur importance dans le cadre une évaluation des risques complète et rigoureuse.

Algorithmes de simulation : Une exploration de différentes méthodes de simulation d'événements rares est fournie, détaillant leurs avantages et applications dans différentes situations. La discussion comprend une analyse complète des techniques d'échantillonnage d'importance, qui impliquent de modifier les distributions de probabilité pour améliorer l'étude des événements rares. De plus, les méthodes de fractionnement d'importance sont examinées pour leur capacité à décomposer des événements rares complexes en sous-événements imbriqués, plus facile à appréhender en termes de techniques d'échantillonage.

Robustesse des Réseaux de Neurones Profonds

Le troisième domaine déplace spécifiquement l'accent sur le champ des DNNs, examinant leur robustesse dans des environnements adverses et stochastiques. Après un bref aperçu des concepts centraux et architectures d'Apprentissage Profond, nous nous penchons sur les problèmes de robustesse adversarial empirique et certifiée, mettant en lumière à la fois leurs similitudes et différences avec l'évaluation de la fiabilité statistique.

Robustesse adversarial : Nous présentons des aspects critiques de la robustesse adversarial dans les DNNs, en nous concentrant sur leur cohérence de performance face à des entrées délibérément manipulées. La nature et les méthodologies des attaques adverses, leur impact sur le comportement des DNN, et le développement de stratégies défensives sont explorés. Cette discussion est cruciale pour comprendre les avancements en cours dans les techniques adverses et l'évolution correspondante des stratégies de robustesse pour les réseaux de neurones profonds.

Robustesse certifiée : La robustesse certifiée dans les DNNs implique de valider et d'assurer leur performance sous conditions adverses. La section contraste les méthodes de vérification formelle, qui fournissent des garanties mathématiques du comportement du réseau, avec des approches de vérification statistique, offrant des évaluations probabilistes basées sur des données simulées. Cette comparaison met en lumière l'approche multifacette pour atteindre la robustesse dans les DNNs, combinant la rigueur théorique avec les contraintes d'applicabilité pratique.

Contributions

Cette thèse présente trois contributions de recherche significatives qui font avancer le domaine de l'évaluation de la fiabilité statistique pour les réseaux de neurones profonds. Ces contributions englobent divers aspects de la fiabilité des réseaux neuronaux et introduisent de nouvelles méthodologies pour aborder ces problèmes difficiles. Nous donnons plus bas des résumés des chapitres associés à chacune de ces contributions.

Chapitre 4 : Notre première contribution, publiée à NeurIPS 2021 [1], se concentre sur "l'Évaluation Statistique Efficace de la Robustesse des Réseaux Neuronaux à la Corruption". Cette recherche aborde le problème crucial de l'évaluation de la robustesse des réseaux neuronaux à la corruption, introduisant un cadre statistique efficace pour une évaluation fiable et évolutive de la performance sous une gamme de scénarios de corruption. Dans cet article, nous adoptons une approche de test d'hypothèse statistique plutôt qu'une approche d'estimation de probabilité et montrons qu'en utilisant une technique spécifique d'estimation d'événements rares séquentielle, il est possible d'obtenir des limites fiables sur le nombre d'itérations nécessaires pour un niveau de confiance donné.

Chapitre 5 : La deuxième contribution, intitulée "Estimation de la Robustesse Statistique des Réseaux Neuronaux Informée par Gradient", a été publiée à AISTATS 2023 [2]. Ce travail propose une approche novatrice exploitant l'information de gradient pour fournir des estimations plus précises et efficaces de la fiabilité des réseaux neuronaux, particulièrement dans le contexte de la robustesse statistique locale. Plus axé sur une estimation efficace en termes d'échantillons, cet article est un travail prometteur qui montre que l'information de gradient, facilement accessible via la rétropropagation, peut en effet accélérer considérablement la convergence. Cependant, cela implique l'utilisation de schémas de Monte Carlo séquentiels plus complexes . Un réglage fin est alors crucial et nous fournissons des détails d'implémentation précis pour les applications aux Réseaux Neuronaux Profonds.

Chapitre 6 : Enfin, notre troisième contribution est intitulée "Estimation Rapide de la Fiabilité des Réseaux Neuronaux avec Échantillonnage d'Importance Conduit par Attaque Adversarial" et est adaptée d'un article récemment accepté pour la conférence UAI 2024 [3]. Elle introduit un nouvel algorithme pour des évaluations de fiabilité efficaces et précises des classifieurs différentiables, combinant des idées de l'analyse de fiabilité et de la robustesse adversarial. Des algorithmes d'Attaque Adversarial sont utilisés pour améliorer l'efficacité de l'Échantillonnage d'Importance. Nous montrons que cette approche est à la fois plus rapide et plus facile à implémenter, dans de nombreux cas, par rapport aux méthodes de simulation d'événements rares présentées dans les premiers travaux, bien qu'elle repose fortement sur l'efficacité des attaques adversarials, qui peut être remise en cause en très grande dimension.

Ces contributions renforcent collectivement la compréhension et l'application pratique de l'ingénierie de fiabilité statistique pour les réseaux de neurones profonds, fournissant de nouvelles méthodologies, algorithmes et insights qui facilitent l'évaluation de la robustesse locale et de la fiabilité des réseaux neuronaux dans divers contextes stochastiques.

Remarque : Dans cette thèse, nous nous intéressons à la fiabilité d'un classifieur de Réseau Neuronal entraîné. C'est un sujet distinct, bien que lié, à celui de *l'utilisation* des Réseaux Neuronaux comme modèles substituts pour les systèmes physiques dans le contexte de la fiabilité pour divers domaines de l'ingénierie, par exemple, comme étudié dans le travail de Chojaczyk, Teixeira, Neves, *et al.* [4]. Ici, les Réseaux Neuronaux ne sont pas des modèles substituts, mais les objets principaux de notre étude.

Conclusion du résumé

En conclusion, cette thèse représente, nous espérons, une avancée notable dans l'évaluation de la fiabilité des Réseaux Neuronaux Profonds (DNNs), un domaine crucial dans l'ère actuelle de l'Intelligence Artificielle. Les trois contributions majeures présentées apportent des perspectives nouvelles et des méthodologies innovantes pour aborder les défis inhérents à la robustesse des DNNs face aux perturbations aléatoires.

L'importance de cette recherche réside dans sa capacité à combiner les principes de l'ingénierie de la fiabilité statistique et les avancées en simulation d'événements rares dans le domaine du Deep Learning, offrant ainsi des outils et des analyses plus précis pour évaluer la performance et la sûreté des systèmes d'IA. Ces méthodes étendent non seulement la compréhension théorique de la fiabilité des DNNs, mais ouvrent également la voie à des applications pratiques plus sûres et fiables dans des domaines critiques pour la sécurité et au-delà. La portée et la profondeur des recherches menées dans cette thèse soulignent en effet l'importance croissante de développer des modèles de Deep Learning robustes et fiables, un impératif dans un monde de plus en plus dirigé par l'IA. Nous espérons que les résultats obtenus ici puissent constituer une base solide pour les recherches futures et pour l'adoption responsable des technologies d'IA dans des contextes sensibles, garantissant ainsi une avancée technologique sûre et bénéfique pour la société.

List of Abbreviations

AI	Artificial Intelligence
GPU	Graphics Processing Unit
ANN	Artificial Neural Network
DNN	Deep Neural Network
CNN	Convolutional Neural Network
MLP	Multi-Layer Perceptron
CV	Computer Vision
NLP	Natural Language Processing
SRE	Statistical Reliability Engineering
MPFP	Most Probable Failure Point
RES	Rare Event Simulation
FORM	First-Order Reliability Method
SORM	Second-Order Reliability Method
LS	Line Sampling
(C)MC	(Crude) Monte Carlo
IS	Importance Sampling
AIS	Adaptive Importance Sampling
CE-AIS	Cross-entropy based Adaptive Importance Sampling
MLS	Multilevel Splitting
AMS	Adaptive Multilevel Splitting
cdf	cumulative distribution function
pdf	probability density function
w.r.t	with respect to
s.t.	such that
r.v.	random variable
i.i.d.	independent identically distributed

Notations

$\mathbb{N}$	set of natural integers
$\mathbb{R}$	set of real numbers
$\mathbf{x}, \mathbf{y}, \mathbf{z}$	vectors, in bold font and lowercase
$\langle \cdot, \cdot \rangle$	scalar product
$(\Omega, \mathcal{F}, \mathbb{P})$	overarching probability space
$\mathbb{E}[\cdot]$	expected value of r.v. in $(\Omega, \mathcal{F}, \mathbb{P})$
$\mathbb{V}[\cdot]$	variance of a r.v. in $(\Omega, \mathcal{F}, \mathbb{P})$
$\lVert \cdot \rVert_2$	euclidian norm
∇f	gradient function of f
$\mathcal{L}(Z)$	probability distribution of a r.v. Z
$Z \sim \nu$	Z is a r.v. with probability distribution ν
$\mathcal{X}$	state space, a.k.a. the X-space or physical space
$\mathcal{U}$	standard normal space, a.k.a. the U-space
$\mathbf{x}_0$	original or expected state of a system
$[k : n]$	subset of $\mathbb{N}$ defined as $\{i \in \mathbb{N},\ \text{s.t. } k \leq i \leq n\}$

Introduction

Context and Motivation

Assessing the reliability of complex engineering systems is a critical task to ensure that fast-paced technological innovations do not come at the expense of increased risks to individuals and societies. In an era where Artificial Intelligence (AI) technology is becoming ubiquitous across industries and is predicted to disrupt several safety-critical fields, the rigorous quantification of AI-driven systems reliability is of utmost importance. Since many modern AI applications rely predominantly on Deep Learning models, it is prudent to prioritize the assessment of their reliability.

Deep Learning encompasses a variety of machine learning models developed to address a wide range of challenges dealing with large, high-dimensional datasets. These tasks include, but are not limited to, Computer Vision (CV), Natural Language Processing (NLP), and Speech Processing. These various models typically share four key attributes:

1. The use of deep 'compositional' architectures, the result of the functional composition or stacking of neural layers.
2. The use of massively parallel hardware (e.g. GPUs) to accelerate inference.
3. The use of Automatic Differentiation (AD) to compute gradients of the model.
4. The use of Stochastic Gradient Descent (SGD) algorithms, and variations thereof, to optimize the model parameters.

Among Deep Learning models, Deep Artificial Neural Networks, also known as Deep Neural Networks (DNNs) [5], hold a prominent position. This thesis primary objective is to delve into the study of the reliability of these models using specific sampling techniques called Rare Event Simulation algorithms. Indeed, while DNNs currently represent the state-of-the-art in fields like computer vision, voice recognition, and natural language processing, their application in critical systems is still limited due to a perceived lack of robustness in the face of input perturbations. This robustness concern applies to both adversarially crafted perturbations [6] and random corruptions [7], [8].

The primary focus of this thesis is to assess the impact of *random* perturbations. We have thus developed new methods to precisely measure the reliability of DNNs when faced with different noise distributions, leveraging the aforementioned common characteristics of Deep Learning models.

Scientific Background

This thesis encompasses a comprehensive exploration of three pivotal scientific domains: Statistical Reliability Engineering, Rare Event Simulation, and the Robustness of Deep Neural Networks (DNNs). Each area plays an integral role in advancing the understanding of complex systems, with a particular focus on the latter part in the context of DNNs. We offer rather broad introductions to these different subjects while emphasizing their role in developing efficient reliability assessment methods for Deep Learning models.

Background Chapters Summaries

The three first chapters of this thesis are dedicated to the fundamental concepts and methods in the three scientific domains mentioned above. We first give short summaries below, followed by more in-depth presentations of each chapter's content:

Chapter 1: - This chapter offers an introduction to the field of statistical reliability engineering, discussing its foundational principles and advanced methods (for static models), and setting the stage for their application in diverse technological domains.

Chapter 2: - The chapter delves into the methodologies and significance of rare event simulation. It covers various techniques and their broader applications, emphasizing their role in understanding and mitigating risks associated with rare events.

Chapter 3: - Focusing on DNNs, this chapter examines the challenges of adversarial robustness, the development of defensive mechanisms, and the concept of certified robustness. It provides a detailed exploration of both formal and statistical approaches to ensuring DNN reliability, leveraging the shared characteristics of these models.

Statistical Reliability Engineering

The exploration begins with a thorough introduction to Statistical Reliability Engineering (SRE), a discipline fundamental to the assessment and enhancement of system reliability across various engineering fields.

Foundations of SRE. We first introduce the core principles and methodologies of reliability engineering, traditionally applied in fields such as mechanical and structural engineering. The discussion encompasses the evolution of these principles and their broad applicability, setting the stage for their relevance in modern technological contexts, including Deep Learning systems. The focus is on the foundational statistical concepts that underpin reliability assessments, crucial for understanding system behaviors under uncertainty and potential failure modes.

Reliability Estimators. A detailed examination of classical estimation methods in reliability engineering is presented. This includes an in-depth look at methods like the First Order Reliability Method (FORM) and the Second Order Reliability Method (SORM), which offer simple probabilistic approximations for failure assessment. Additionally, an exploration of the Line Sampling technique is included, providing insight into its potential application in complex, high-dimensional statistical reliability analysis.

Rare Event Simulation

Following the introduction to reliability engineering, the thesis transitions into the domain of Rare Event Simulation, a key technique in the study of extreme events.

Concepts and Challenges. We introduce the concept of rare event simulation and present the main technical challenges associated with their applications. The precise statistical nature of rare events is discussed, highlighting the limitations and inefficiency of conventional simulation and analysis methods. We can then introduce specific rare event simulation methods as powerful tools to model and study rare events, emphasizing their significance in comprehensive and rigorous risk assessment.

Simulation Algorithms. An exploration of different rare event simulation methods is provided, detailing their advantages and applications in different situations. The discussion includes a comprehensive analysis of importance sampling techniques, which involve modifying probability distributions to enhance the study of rare events. Additionally, importance splitting methods are examined for their ability to decompose complex rare events into simpler, more analyzable components.

Robustness of Deep Neural Networks

The third domain shifts the focus specifically to the field of DNNs, examining their robustness in adversarial and stochastic environments. After a brief overview of the central concepts and architectures of Deep Learning, we delve into the issues of empirical and certified adversarial robustness, highlighting both their similarity and differences with statistical reliability assessment.

Adversarial Robustness. We present critical aspects of adversarial robustness in DNNs, focusing on their performance consistency in the face of deliberately manipulated inputs. The nature and methodologies of adversarial attacks, their impact on DNN behavior, and the development of defensive strategies are explored. This discussion is crucial for understanding the ongoing advancements in adversarial techniques and the corresponding evolution of robustness strategies in DNNs.

Certified Robustness. Certified robustness in DNNs involves validating and ensuring their performance under adversarial conditions. The section contrasts formal verification methods, which provide mathematical guarantees of network behavior, with statistical verification approaches, offering probabilistic assessments based on simulated data. This comparison sheds light on the multifaceted approach to achieving robustness in DNNs, combining theoretical rigor with practical applicability.

Main Contributions

This thesis presents three significant research contributions that advance the field of statistical reliability assessment for deep neural networks. These contributions encompass various aspects of neural network reliability and introduce novel methodologies to address these challenging problems. We next give summaries of the chapters associated with each of these contributions.

Chapter 4: Our first contribution, published at NeurIPS 2021 [1], focuses on "Efficient Statistical Assessment of Neural Network Corruption Robustness." This research addresses the critical issue of assessing neural network robustness to corruption, introducing an efficient statistical framework for reliable and scalable performance evaluation under a range of corruption scenarios. In this paper, we take a statistical hypothesis-testing approach rather than a probability estimation approach and show that, while using a specific sequential rare event estimation technique, it is possible to obtain reliable bounds on the number of iterations needed for a given confidence level.

Chapter 5: The second contribution, untitled "Gradient-Informed Neural Network Statistical Robustness Estimation," was published at AISTATS 2023 [2]. This work proposes a novel approach leveraging gradient information to provide more accurate and efficient estimations of neural network reliability, particularly in the context of local statistical robustness. More focused on sample efficient estimation, this paper is a promising work that shows that gradient information, readily accessible through back-propagation, can indeed substantially speed up convergence. However, it does come at the expense of more involved sequential Monte Carlo schemes [9]. Fine-tuning is then crucial and we provide precise implementation details for applications to Deep Neural Networks.

Chapter 6: Finally, our third contribution is untitled "Fast Reliability Estimation of Neural Networks with Adversarial Attack-Driven Importance Sampling", and is adapted from a work recently accepted at the 40th conference on Uncertainty in Artificial Intelligence (UAI 2024) [3]. It introduces a new algorithm for efficient and accurate reliability assessments of differentiable classifiers, combining ideas from reliability analysis and adversarial robustness. Adversarial attack algorithms are used to improve Importance Sampling efficiency. We show that this approach is both faster and easier to implement, in many cases, in comparison to rare event simulation methods presented in the first works, though it does heavily rely on the effectiveness of adversarial attacks.

These contributions collectively enhance the understanding and practical application of statistical reliability engineering for deep neural networks, providing new methodologies, algorithms, and insights that facilitate the assessment of neural network performance, local robustness, and reliability in various stochastic contexts.

Note: In this thesis we are interested in the reliability of a trained Neural Network classifier. This is a distinct, though related, subject to that of *using* Neural Networks as surrogate models for physical systems in the context of reliability for various fields of engineering, for example, as studied in the work of Chojaczyk, Teixeira, Neves, *et al.* [4]. Here, Neural Networks are not surrogate models, but the primary objects of our study.

Part I

Technical Background

CHAPTER 1

Introduction to Statistical Reliability Engineering

"The price of reliability is the pursuit of the utmost simplicity.
It is a price which the very rich find most hard to pay."
– C. A. R. Hoare, The Emperor's Old Clothes [10], 1981

1.1 Overview of Statistical Reliability Engineering

Statistical Reliability Engineering is a field dedicated to assessing and quantifying the reliability of complex engineering systems by leveraging probabilistic principles. In this section, we provide an overview of the motivations behind this discipline and present some illustrative examples, setting the stage for a deeper exploration of reliability analysis. **Note:** In this thesis we consider only *static* models, as defined in structural reliability [11].

1.1.1 Key Concepts

Statistical Reliability Engineering emerges from the necessity to handle the uncertainty inherent in the performance of complex systems. It relies on a foundation of probability theory to evaluate the likelihood of system failure. At its core, it involves modeling and analyzing the behavior of systems when faced with variations or perturbations in their inputs. The ultimate goal is to provide reliable and accurate estimates of system performance, helping engineers and designers make informed decisions.

This discipline employs a rigorous probabilistic framework, introducing concepts such as state spaces, random variables, and limit state functions to precisely define and hence quantify system reliability. A critical concept is the probability of failure, which is the probability of a system failing when subjected to perturbations of its inputs around a nominal or expected state $\mathbf{x}_0$, denoted by $P_{\mathrm{F}}(\mathbf{x}_0)$ or simply P_{F}. Additionally, a critical level of failure, P_C, represents the acceptable threshold for system performance, beyond which the system's reliability is considered inadequate. The goal is to ensure that $P_{\mathrm{F}} < P_C$.

1.1.2 Real-World Examples

Let's consider three examples from the real world that illustrate the significance of Statistical Reliability Engineering:

- **Aircraft Structural Design** [12]: In the design of aircraft structures, it is crucial to ensure the structural components can withstand various operational conditions, such as turbulence, extreme temperatures, and mechanical stresses. Statistical Reliability Engineering plays a vital role in assessing the reliability of these components by accounting for uncertainties in material properties and environmental factors. Engineers and designers aim to achieve a low enough P_{F}, ensuring that $P_{\mathrm{F}} \leq P_C$, where P_C represents an acceptable critical level of failure based on factors like the expected total number of flight hours and overall risk tolerance.
- **Medical Device Reliability** [13]: Medical devices, such as pacemakers or insulin pumps, must operate reliably to ensure patient safety. These devices may encounter variations in physiological conditions, usage patterns, or component performance. Statistical Reliability Engineering is employed to assess the reliability of these medical devices, considering uncertainties and variations in their operating environments. An acceptable P_C for these devices is determined based on the number of hours they are expected to operate and the acceptable level of risk in healthcare.
- **Self-Driving Car Reliability** [14]: In the realm of autonomous vehicles, such as self-driving cars, ensuring reliability is of utmost importance. These vehicles operate under various conditions and are exposed to uncertainties in traffic, weather, and sensor performance. Statistical Reliability Engineering can be employed to assess the likelihood of system failures during autonomous operations. Here, an acceptable P_C can be defined based on the number of hours of autonomous driving and the acceptable risk level, ensuring that the system operates safely.

These examples emphasize the real-world applications of Statistical Reliability Engineering in ensuring the safety and dependability of systems operating in uncertain conditions. The mathematical framework introduced in the subsequent section will provide a rigorous understanding of the principles and techniques employed in this discipline.

1.2 Mathematical Framework

Statistical Reliability Engineering relies on a probabilistic framework for assessing system reliability under uncertainty. This section introduces the mathematical objects of this discipline, emphasizing probabilistic transformations, random variables, and limit state functions. In this section, knowledge of fundamental concepts of probability theory (random variables, expected value, variance, etc.) is assumed, as presented for example in the monograph of Feller [15] or more recently in the monograph of J-F. Le Gall [16]. Thereafter, as is customary, we suppose the existence of a triplet $(\Omega, \mathcal{F}, \mathbb{P})$ where Ω is the sample space representing all possible outcomes, $\mathcal{F}$ is the σ-algebra representing all events on Ω and $\mathbb{P}$ is a probability measure on $(\Omega, \mathcal{F})$. Throughout this thesis, $\mathbb{E}[\cdot]$ and $\mathbb{V}[\cdot]$ denote respectively the expectation and the variance operators [1] on random variables and random vectors. Of particular interest in $\mathcal{F}$ is the event of a system failure, denoted $\mathcal{S}_{\mathrm{F}}$, as we seek to accurately estimate or bound its probability $P_{\mathrm{F}} = \mathbb{P}[\mathcal{S}_{\mathrm{F}}]$.

1.2.1 Random Variables and Limit State Functions

Key to statistical reliability engineering are the random variables modeling the input uncertainty:

- **Random Variable $\mathbf{X}$**: Represents the system's noisy inputs in the state space $\mathcal{X}$, also called the X-space. Formally, $\mathbf{X}$ is a deterministic mapping from the sample space Ω to the state space $\mathcal{X}$. Throughout this thesis, we denote its probability distribution by $\nu_{\mathbf{X}}$ and its cumulative distribution function (cdf) by F_X. Domain experts specify $\nu_{\mathbf{X}}$ either directly or through data analysis. It is generally centered around the state $\mathbf{x}_0$, i.e. $\mathbb{E}[\mathbf{X}] = \mathbf{x}_0$. In addition, it often has finite second moment $\mathbb{V}[\mathbf{X}] = \mathbf{\Sigma}_{\mathbf{x}}$. A common model for uncertainty is additive noise, in which $\mathbf{X} = \mathbf{x}_0 + \mathbf{Z}$, where $\mathbf{Z}$ is a centered random variable (i.e. $\mathbb{E}[\mathbf{Z}] = 0$) with finite variance.
- **Random Variable $\mathbf{U} = \mathcal{T}(\mathbf{X})$**: This random variable arises from mapping, via a bijective transform $\mathcal{T}$, the random variable $\mathbf{X}$ into the standard normal space, denoted $\mathcal{U}$ and also called the U-space. The U-space plays a crucial role in many statistical reliability methods introduced in this thesis.

1. As in [15], we use the same notation for the variance of real-valued random variables and the variance-covariance matrix of random vectors, defined as $\mathbb{V}[\mathbf{X}] \triangleq \mathbb{E}[(\mathbf{X} - \mathbb{E}[\mathbf{X}])(\mathbf{X} - E[\mathbf{X}])^T]$.

In this thesis: We assume that the state space $\mathcal{X}$ is a Borel subset of $\mathbb{R}^d$ of positive measure (e.g. $\mathcal{X} = [0,1]^d$) and that $\mathbf{X} = (X_1, \cdots, X_d)$ is a random vector with a joint probability density function (pdf) $p_{\mathbf{X}}$ w.r.t. Lebesgue measure so that $\nu_{\mathbf{X}}(d\mathbf{x}) = p_{\mathbf{X}}(\mathbf{x})d\mathbf{x}$. Accordingly, the U-space is simply $\mathbb{R}^d$, and the mapping $\mathcal{T}$ is designed so that $\mathbf{U}$ follows the standard normal distribution on $\mathbb{R}^d$. Therefore, working in the U-space is generally more convenient, as the uncertainty distribution $\nu_{\mathbf{X}}$ in the X-space can be quite intricate. The precise nature of the transform $\mathcal{T}$ is discussed further below (see section 1.2.2).

Limit State Functions. The system's safety performance is described at each point of the state space by a deterministic scalar function $g : \mathcal{X} \to \mathbb{R}$, known as the limit state function (LSF). It defines the boundary between safe region $\mathcal{X}_{\text{S}}$ and failure region $\mathcal{X}_{\text{F}}$ in the state space, and more generally gives a measure of how close or far the system is to this boundary. Formally, there exists a threshold value τ such that

$$\mathcal{X}_{\text{S}} := \{\mathbf{x} \in \mathcal{X}, g(\mathbf{x}) > \tau\} \tag{1.1}$$

$$\mathcal{X}_{\text{F}} := \{\mathbf{x} \in \mathcal{X}, g(\mathbf{x}) \leq \tau\}. \tag{1.2}$$

For simplicity and without loss of generality, in this thesis, we assume that $\underline{\tau = 0}$. In the space U-space, the LSF is represented by the function G, defined as,

$$\forall \mathbf{u} \in \mathbb{R}^d, G(\mathbf{u}) := g(\mathcal{T}^{-1}(\mathbf{u})). \tag{1.3}$$

Accordingly, we define the safe and failure regions in the U-space as

$$\mathcal{U}_{\text{S}} := \{\mathbf{u} \in \mathbb{R}^d, G(\mathbf{u}) > 0\} \tag{1.4}$$

$$\mathcal{U}_{\text{F}} := \{\mathbf{u} \in \mathbb{R}^d, G(\mathbf{u}) \leq 0\}. \tag{1.5}$$

Of particular interest are also the boundaries of these failure and safety regions, denoted by ∂g and ∂G in the X-space and the U-space respectively, and defined as

$$\partial g := \{\mathbf{x} \in \mathcal{X}, g(\mathbf{x}) = 0\} \tag{1.6}$$

$$\partial G := \{\mathbf{u} \in \mathbb{R}^d, G(\mathbf{u}) = 0\}. \tag{1.7}$$

The regularity of the LSF also plays an important role in reliability assessment. In essence, all reliability methods introduced in this chapter assume that G is *at least* differentiable.

As will be discussed in this thesis, the dimension of the problem and the shape of the LSF boundary are key factors of the 'hardness' of a statistical reliability estimation problem.

1.2.2 Isoprobabilistic Transformations

Isoprobabilistic transformations are vital for simplifying the reliability problem. In this section, after introducing the concept of copulas, we present two important families of isoprobabilistic transformations predominantly used in Statistical Reliability Engineering. We conclude with a brief discussion of the regularity of isoprobabilistic transforms.

Copulas and Sklar's Theorem

The notion of copulas, introduced by Sklar [17], is an important concept in statistical analysis, precious for modeling complex dependence structures in multivariate distributions. The main idea of copulas is to disentangle the multivariate dependence structure from the marginal law of each component in a random vector.

Definition of Copulas. From a probabilistic standpoint, a copula is the joint cdf of a random vector $\mathbf{Z} = (Z_1, \ldots, Z_d)$ taking its values in $[0,1]^d$ and such that the marginal law of each component Z_i is the uniform distribution on $[0,1]$. Analytically, a copula C is a function from the unit cube $[0,1]^d$ to the unit interval $[0,1]$ with the following properties:

- For any vector $(u_1, \ldots, u_d) \in [0,1]^d$, it holds that $C(u_1, \ldots, u_d) = 0$ if at least one of the u_i equals 0.
- For any vector $(u_1, \ldots, u_d) \in [0,1]^d$, it holds that $C(1, \ldots, 1, u_i, 1, \ldots, 1) = u_i$.
- C is an increasing and *positive* function, meaning here that the volume of any hyperrectangle $[a_1, b_1] \times \ldots \times [a_d, b_d]$ under C is non-negative, for any $0 \leq a_i \leq b_i \leq 1$.

We next present Sklar's Theorem, the fundamental result of the theory of copulas.

Theorem 1 (Sklar's Theorem [17]). *Given any multivariate cumulative distribution function F of a random vector $(X_1, \ldots, X_d)$ with marginal cumulative distribution functions $F_1, \ldots, F_d$, there exists a copula C such that for all $x_1, \ldots, x_d$ in the extended real line,*

$$F(x_1, \ldots, x_d) = C(F_1(x_1), \ldots, F_d(x_d)). \tag{1.8}$$

Furthermore, if the marginal distributions are continuous, then the copula C is unique.

This result essentially means that a given distribution ν on $\mathbb{R}^d$ can be characterized by its marginals and a copula function. Moreover, this characterization is *unique* as long as its marginals are continuous, i.e. for all $x \in \mathbb{R}$ and $i \in [1:d]$, we have $\mathbb{P}_{\mathbf{Z}\sim\nu}[Z_i = x] = 0$.

Gaussian Copulas. Of particular interest are Gaussian copulas, derived from the multivariate normal distribution, defined by their correlation matrix $\mathbf{\Sigma}_C \in [-1,1]^d$, which captures the linear dependencies between variables. Mathematically, if Φ is the cdf of univariate normal distribution and $\Phi_{\mathbf{\Sigma}_C}$ is the cdf of a multivariate normal distribution with correlation matrix $\mathbf{\Sigma}_C$ and zero mean, the associated Gaussian copula $C_{\mathbf{\Sigma}_C}$ is given by:

$$\forall (u_1, \cdots, u_d) \in [0,1]^d, C^{\text{Gauss}}_{\mathbf{\Sigma}_C}(u_1, \ldots, u_d) = \Phi_{\mathbf{\Sigma}_C}(\Phi^{-1}(u_1), \ldots, \Phi^{-1}(u_d)). \tag{1.9}$$

Notably Gaussian copulas model linear dependence between variables in a transformed normal space, despite the individual marginals not necessarily being normal. Therefore, a distribution on $\mathbb{R}^d$ with Gaussian copula $C^{\text{Gauss}}_{\mathbf{\Sigma}}$ has independent components iff $\mathbf{\Sigma} = I_d$.

Elliptic Copulas. Another important family of copulas is that of the elliptic copulas, a substantial generalization of Gaussian copulas to elliptic distributions [18]. A centered $\mathbb{R}^d$-valued r.v. $\mathbf{Z}$ is said to be elliptic if, its characteristic function $\varphi_{\mathbf{Z}}$, defined as

$$\forall \mathbf{t} \in \mathbb{R}^d, \varphi_{\mathbf{Z}}(\mathbf{t}) \triangleq \mathbb{E}[e^{\mathrm{i}\langle \mathbf{t}, \mathbf{Z}\rangle}] \tag{1.10}$$

can be expressed using a positive-definite matrix $\mathbf{\Sigma}$ and a scalar function $\psi : \mathbb{R} \mapsto \mathbb{R}$, s.t.

$$\forall \mathbf{t} \in \mathbb{R}^d, \varphi_{\mathbf{Z}}(\mathbf{t}) = \psi(\langle \mathbf{t}, \mathbf{\Sigma} \cdot \mathbf{t}\rangle). \tag{1.11}$$

The associated copula function is then noted $C^{\psi}_{\mathbf{\Sigma}}$. Important families of elliptic distributions include the already covered Gaussian distributions $\left(\text{taking } \psi(x) = e^{-\frac{x}{2}}\right)$, the symmetric multivariate Laplace distributions $\left(\text{taking } \psi(x) = \frac{1}{1+x}\right)$, as well as the multivariate Logistic distributions [19] and the multivariate Student t-distributions [20]. In contrast to Gaussian copulas, elliptic copulas can model complex non-linear dependency between variables, e.g. where $\mathbf{\Sigma} = I_d$ and yet the components are not independent.
The theory of copulas thus provides a robust framework for analyzing and modeling dependencies in multivariate distributions, which is of paramount importance in fields

such as reliability engineering [21] and quantitative finance [22], where understanding interdependencies can significantly impact risk assessment and decision-making processes. We next show how knowledge of the copula of the r.v. $\mathbf{X}$ and its marginals can be used to construct isoprobabilistic transforms.

Nataf Transforms

We now suppose the random vector $\mathbf{X}$ is known (or assumed) to have a Gaussian copula with correlation matrix $\mathbf{\Sigma}_C$, and its marginal cdfs, denoted $F_1, \ldots, F_d$, are known. One can then use the method introduced by Nataf [23] to build an isoprobabilistic transform $\mathcal{T}^{\text{Nataf}}$. This transformation can be written as $\mathcal{T}^{\text{Nataf}} \equiv \mathcal{T}_2^{\text{Nataf}} \circ \mathcal{T}_1^{\text{Nataf}}$, where $\mathcal{T}_1^{\text{Nataf}}, \mathcal{T}_2^{\text{Nataf}}$ are given by,

$$\begin{aligned}
\mathcal{T}_1^{\text{Nataf}} : \mathbb{R}^d &\mapsto \mathbb{R}^d \\
\mathbf{x} &\to \mathcal{T}_1^{\text{Nataf}}(\mathbf{x}) \triangleq \begin{pmatrix} \Phi^{-1}(F_1(x_1)) \\ \vdots \\ \Phi^{-1}(F_d(x_d)) \end{pmatrix} \qquad (1.12) \\
\mathcal{T}_2^{\text{Nataf}} : \mathbb{R}^d &\mapsto \mathbb{R}^d \\
\mathbf{v} &\to \mathcal{T}_2^{\text{Nataf}}(\mathbf{v}) := \mathbf{\Gamma}\mathbf{v} \qquad (1.13)
\end{aligned}$$

where Φ is the cdf of the standard normal law and $\mathbf{\Gamma}$ is the inverse of the lower triangular factor in the Cholesky decomposition of $\mathbf{\Sigma}_C$. Thus, $\mathbf{X}$ is first mapped to the centered Gaussian r.v. $\mathbf{Z} = \mathcal{T}_1^{\text{Nataf}}(\mathbf{X})$, with correlation $\mathbf{\Sigma}_C$, and then mapped to the $\mathbf{U}$ with a simple linear operation. In the case where the marginal laws of $\mathbf{X}$ are already Gaussian, with $X_i \sim \mathcal{N}(\mu_i, \sigma_i)$ for all $i \in [1:d]$, the transform $\mathcal{T}_1^{\text{Nataf}}$ can be replaced by the linear normalization transform $\mathcal{T}_1^{\text{Gauss}}$, with $\mathcal{T}_1^{\text{Gauss}}(\mathbf{x}) \triangleq \left(\frac{x_1-\mu_1}{\sigma_1}, \ldots, \frac{x_d-\mu_d}{\sigma_d}\right)$, for all $\mathbf{x}$ in $\mathbb{R}^d$. Lebrun and Dutfoy [24] highlighted that the assumption of a Gaussian copula is quite restrictive and thus proposed a generalization of this method to elliptic copulas.

Rosenblatt Transforms

A different type of isoprobabilistic transform was introduced by Rosenblatt [25]. In contrast with Nataf transforms it does not require any assumption on the copula of $\mathbf{X}$. However, it can be more computationally involved as it requires computing cdf of the conditional probability law $\mathcal{L}(X_j | X_1, \ldots, X_{j-1})$, denoted $F_{j|1:j-1}$, for all j in $[2:d]$.

Formally $F_{j|1:j-1}$ can be defined, when it exists, as the following limit,

$$\forall \mathbf{x} = (x_1, \dots, x_d) \in \mathcal{X}, \forall j \in [2:d],$$
$$F_{j|1:j-1}(x_j|x_1, \dots, x_j) = \lim_{\boldsymbol{\delta} \to \mathbf{0}} \frac{\mathbb{P}[X_1 \in [x_1, x_1+\delta_1], \dots, X_{j-1} \in [x_{j-1}, x_{j-1}+\delta_{j-1}], X_j \leq x_j]}{\mathbb{P}[x_1 \leq X_1 \leq x_1+\delta_1, \dots, x_{j-1} \leq X_{j-1} \leq x_{j-1}+\delta_{j-1}]}$$

Noting $F_{1:j}$ the joint cdf of the subvector $(X_1, \dots, X_j)$ for j in $[2:d]$ we can write $F_{j|1:j-1}$ as,

$$F_{j|1:j-1}(x_j|x_1, \dots, x_{j-1}) = \frac{\frac{\partial^{j-1} F_{1:j}(x_1, \dots, x_j)}{\partial x_1 \cdots \partial x_{j-1}}}{\frac{\partial^{j-1} F_{1:j-1}(x_1, \dots, x_{j-1})}{\partial x_1 \cdots \partial x_{j-1}}} \tag{1.14}$$

Denoting the joint pdf $(X_1, \dots, X_j)$ by $p_{X_{1:j}}$ for $j \in [1:d]$, this can be written again as,

$$F_{j|1:j-1}(x_j|x_{1:j-1}) = \frac{1}{p_{X_{1:j-1}}(x_{1:j-1})} \int_{-\infty}^{x_j} p_{X_{1:j-1}}(x_1, \dots, x_{j-1}, x) dx \tag{1.15}$$

where $x_{1:j-1}$ is an abbreviation for $x_1, \dots, x_{j-1}$.
Once the conditional cdfs are determined the Rosenblatt Transform $\mathcal{T}^{\text{Ros}}$ can be decomposed as $\mathcal{T}^{\text{Ros}} = \mathcal{T}_1^{\text{Ros}} \circ \mathcal{T}_2^{\text{Ros}}$ where $\mathcal{T}_1^{\text{Ros}}, \mathcal{T}_2^{\text{Ros}}$ are invertible functions given by,

$$\mathcal{T}_1^{\text{Ros}} : \mathbb{R}^d \mapsto \mathbb{R}^d$$
$$\mathbf{x} \to \mathcal{T}_1^{\text{Ros}}(\mathbf{x}) \triangleq \begin{pmatrix} F_1(x_1) \\ F_{2|1}(x_2|x_1) \\ \vdots \\ F_{d|1:d-1}(x_d|x_{1:d-1}) \end{pmatrix} \tag{1.16}$$

$$\mathcal{T}_2^{\text{Ros}} : \mathbb{R}^d \mapsto \mathbb{R}^d$$
$$\mathbf{v} \to \mathcal{T}_2^{\text{Ros}}(\mathbf{v}) := \begin{pmatrix} \Phi^{-1}(v_1) \\ \Phi^{-1}(v_2) \\ \vdots \\ \Phi^{-1}(v_d) \end{pmatrix} \tag{1.17}$$

The idea is to first map $\mathbf{X}$ to the r.v. $\mathbf{V} = \mathcal{T}_1^{\text{Ros}}(\mathbf{X})$ following the uniform distribution on $[0,1]^d$, denoted $\mathcal{U}([0,1]^d)$. Then to map $\mathbf{V}$ to the r.v. $\mathbf{U} = \mathcal{T}_2^{\text{Ros}}(\mathbf{V})$, via a component-wise application of the inverse of the cdf of the standard normal distribution. In some cases, the Nataf and Rosenblatt transform coincide, for example, Lebrun and Dutfoy [26] showed that when $\mathbf{X}$ has a Gaussian copula, then the transforms $\mathcal{T}^{\text{Nataf}}$ and $\mathcal{T}^{\text{Ros}}$ are identical.

Regularity of the Mapping to the Standard Space

As mentioned above an isoprobabilistic transformation, denoted as $\mathcal{T} : \mathcal{X} \to \mathbb{R}^d$, is an invertible mapping that changes the original input space to the standard normal space, such that each component of $\mathbf{U} = \mathcal{T}(\mathbf{X})$ follows a standard normal distribution. Typically $\mathcal{T}$ is not only bijective but also a diffeomorphism: a differentiable bijective mapping whose reciprocal function $\mathcal{T}^{-1}$ is also differentiable. Indeed, this is necessary in most cases, as the limit state function $G = g \circ \mathcal{T}^{-1}$ must be differentiable for many reliability estimation methods to be applied. In fact, from equation (1.12) and (1.13) it can be seen that $\mathcal{T}^{\text{Nataf}}$ will have the same regularity as the marginals cdfs $F_1, \dots, F_d$. Therefore if these functions are of class $\mathcal{C}^k$ (k continuously differentiable) then the Nataf Transform is a $\mathcal{C}^k$-diffeomorphism (both $\mathcal{T}$ and $\mathcal{T}^{-1}$ are of class $\mathcal{C}^k$). In contrast, we can see from equation (1.15) that the regularity of Rosenblatt Transform $\mathcal{T}^{\text{Ros}}$ can be limited by the regularity of the pdf of $\mathbf{X}$. However for a pdf of class $\mathcal{C}^\infty$, for example when $\mathbf{X}$ is Gaussian, this distinction is inconsequential. Since many continuous distributions have differentiable pdfs (exceptions include the Laplace distribution, whose pdf is not differentiable at zero), in the rest of the thesis, we assume $\mathcal{T}$ is at least a C^1-diffeomorphism.

1.2.3 Probability of Failure

Given this probabilistic framework, the probability of failure can be rewritten using the fact that $\mathcal{S}_\text{F} = [\mathbf{X} \in \mathcal{X}_\text{F}]$,

$$P_\text{F} = \mathbb{P}[\mathbf{X} \in \mathcal{X}_\text{F}] = \int_{\mathcal{X}} \mathbb{1}_{\mathcal{X}_\text{F}}(\mathbf{x}) p_X(\mathbf{x}) d\mathbf{x} = \int_{\mathcal{X}} \mathbb{1}_{g(\mathbf{x}) \leq 0} p_X(\mathbf{x}) d\mathbf{x}. \tag{1.18}$$

In the standard normal space, we obtain, using $\mathcal{S}_\text{F} = [\mathbf{U} \in \mathcal{U}_\text{F}]$,

$$P_\text{F} = \int_{\mathbb{R}^d} \mathbb{1}_{\mathcal{U}_\text{F}}(\mathbf{u}) \phi_d(\mathbf{u}) d\mathbf{u} = \int_{\mathbb{R}^d} \mathbb{1}_{G(\mathbf{u}) \leq 0} \phi_d(\mathbf{u}) d\mathbf{u} \tag{1.19}$$

where ϕ_d is the joint probability density function (pdf) of the standard normal distribution on $\mathbb{R}^d$, i.e.

$$\forall \mathbf{u} \in \mathbb{R}^d, \phi_d(\mathbf{u}) = \frac{1}{(2\pi)^{d/2}} e^{-\frac{\|\mathbf{u}\|_2^2}{2}}. \tag{1.20}$$

The mathematical framework introduced here is essential for applying various statistical assessment methods, leveraging probabilistic concepts to quantify the reliability of complex systems under uncertainty. We next present several important reliability estimators.

1.3 Statistical Reliability Estimators

In this section, we introduce four key statistical reliability engineering estimators: the First-Order Reliability Method (FORM), the Second-Order Reliability Method (SORM), the First-Order Second Moment Method (FOSM), and the Line Sampling (LS) method. These estimators originate from structural reliability [11], where they are used to assess the reliability of mechanical structures. Important additional classes of estimators are covered next in chapter 2, which focuses on rare event simulation methods.

1.3.1 First-Order Reliability Method

The First-Order Reliability Method (FORM) is a cornerstone of Statistical Reliability Engineering. It approximates the probability of failure by linearizing the limit state function at the Most Probable Failure Point (MPFP), also known as the Design Point.

Definition and Importance of the MPFP

The MPFP, or design point, is defined, when it exists, as the point in the U-space with the highest probability density on the subspace of $\mathbb{R}^d$ where $G(\mathbf{u}) \leq 0$. Therefore, formally, the MPFP, noted $\mathbf{u}^*$, can be defined, when it exists, by

$$\mathbf{u}^* = \underset{\mathbf{u}\in\mathbb{R}^d}{\operatorname{argmin}} \|\mathbf{u}\|_2^2 \;\; s.t.\; G(\mathbf{u}) \leq 0. \tag{1.21}$$

This point is crucial as it represents the most likely failure point in the transformed standard normal space. Its existence and uniqueness are essential assumptions of the FORM and SORM methods, presented below. It is possible however to adapt these methods in the case where several MPFPs exist, see the end of section 1.3.2 for more details. Another crucial assumption is that G is at least differentiable around the MPFP.

Finding the MPFP. The difficulty of this step globally depends on the shape of the failure region boundary ∂G and the dimension of the problem d. In the case of Neural Networks, where the limit state function is non-linear and the input dimension can be in the order of hundreds of thousands, this task can be challenging. Even in low-dimensional spaces, precisely locating it can be challenging if the boundary ∂G has a complex geometry and if the computational budget is limited. After presenting the ideal FORM estimator, we discuss examples of MPFP search algorithms below (see section 1.3.1).

FORM Procedure and Probability Integration

The main idea of FORM estimation is to approximate the LSF by linearization at the MPFP using a first-order Taylor expansion, i.e.

$$G(\mathbf{u}) \approx L^*(\mathbf{u}) = \underbrace{G(\mathbf{u}^*)}_{=0} + \langle \nabla G(\mathbf{u}^*), (\mathbf{u} - \mathbf{u}^*) \rangle \tag{1.22}$$

where $\langle \cdot, \cdot \rangle$ denotes the scalar product on $\mathbb{R}^d$. The ideal FORM estimator for the probability of failure can then be formulated as:

$$P_{\mathrm{F}}^{FORM} = \mathbb{P}[L^*(\mathbf{U}) \leq 0]. \tag{1.23}$$

As $\mathbf{U}$ is a Gaussian r.v. and L^* is linear, $L^*(\mathbf{U})$ is also a Gaussian random variable. We now *assume* that $\nabla G(\mathbf{u}^*)$ is not the null vector. Otherwise $P_{\mathrm{F}}^{\mathrm{FORM}} = 1$, which is uninformative. Under this assumption, we obtain,

$$P_{\mathrm{F}}^{FORM} = \Phi\left(-\frac{\mu_{L^*}}{\sigma_{L^*}}\right), \tag{1.24}$$

where $\mu_{L^*} = \mathbb{E}[L^*(\mathbf{U})]$ and $\sigma_{L^*} = \sqrt{\mathbb{V}[L^*(\mathbf{U})]}$. Using equation (1.22) we obtain,

$$\mu_{L^*} = \langle \nabla G(\mathbf{u}^*), \underbrace{\mathbb{E}[\mathbf{U}]}_{=0} \rangle - \langle \nabla G(\mathbf{u}^*), \mathbf{u}^* \rangle = -\langle \nabla G(\mathbf{u}^*), \mathbf{u}^* \rangle \tag{1.25}$$

$$\sigma_{L^*} = \sqrt{\mathbb{V}[\langle \nabla G(\mathbf{u}^*), \mathbf{U}]\rangle} = \sqrt{\langle \nabla G(\mathbf{u}^*), \nabla G(\mathbf{u}^*) \rangle} = \|\nabla G(\mathbf{u}^*)\|_2. \tag{1.26}$$

Thus,

$$P_{\mathrm{F}}^{FORM} = \Phi(-\langle \boldsymbol{\alpha}_*, \mathbf{u}^* \rangle)$$

where $\boldsymbol{\alpha}_* = \frac{-\nabla G(\mathbf{u}^*)}{\|\nabla G(\mathbf{u}^*)\|}$. An important point here is that, since $\mathbf{u}^*$ is a solution to the constrained optimization problem (1.21), we know by the Lagrange Multiplier Theorem that $\nabla G(\mathbf{u}^*)$ and $\mathbf{u}^* = \frac{1}{2}\nabla_{\mathbf{u}}\|\mathbf{u}\|^2|_{\mathbf{u}=\mathbf{u}^*}$ are colinear. Since it's also clear that they must be of opposite directions, we obtain $\boldsymbol{\alpha}_* = \frac{\mathbf{u}^*}{\|\mathbf{u}^*\|}$ and thus finally, the ideal FORM estimator can be rewritten as,

$$P_{\mathrm{F}}^{FORM} = \Phi(-\beta_{HL}) \tag{1.27}$$

where $\beta_{HL} = \langle \boldsymbol{\alpha}_*, \mathbf{u}^* \rangle = \|\mathbf{u}^*\|_2$, is known as the Hasofer-Lind reliability index [27].

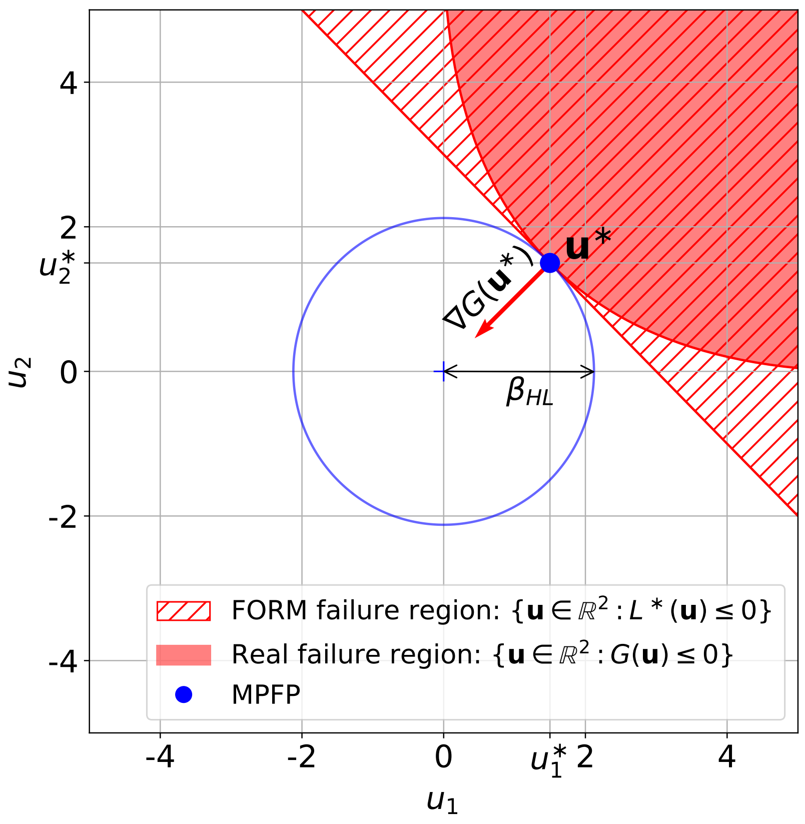


Figure 1.1 – Illustration of FORM, applied in $\mathbb{R}^2$ to the LSF given by $\forall \mathbf{u} = (u_1, u_2) \in \mathbb{R}^2, G(\mathbf{u}) = (u_1 - 3)^2 + (u_2 - 3)^2 - \frac{1}{2}(u_1 + u_2)^2$.

In practice, the FORM procedure can be broken down into 3 steps:

1. Design an Isoprobabilistic Transformation $\mathcal{T}$ that maps random variables from the X-space to standard Gaussian vectors in the U-space.
2. Use an optimization algorithm to find an approximate MPFP $\mathbf{u}^{\text{opt}} \approx \mathbf{u}^*$ in the U-space and compute the approximate Hasofer-Lind reliability index $\hat{\beta}_{HL} = \|\mathbf{u}^{\text{opt}}\|_2$.
3. Compute the failure probability estimation as $\hat{P}_{\text{F}}^{FORM} = \Phi(-\hat{\beta}_{HL})$.

Note that when $\mathbf{u}^{\text{opt}}$ is not exactly an MPFP, it may not be colinear to its associated normal vector $\boldsymbol{\alpha}_{\text{opt}} = \frac{-\nabla G(\mathbf{u}^{\text{opt}})}{\|\nabla G(\mathbf{u}^{\text{opt}})\|}$, and thus the computations above fall short, and we might want to compute instead an alternate reliability index given by

$$\beta_{\text{opt}} = \frac{G(\mathbf{u}^{\text{opt}})}{\|\nabla G(\mathbf{u}^{\text{opt}})\|_2} + \langle \boldsymbol{\alpha}_{\text{opt}}, \mathbf{u}^{\text{opt}} \rangle. \tag{1.28}$$

We illustrate the FORM procedure in Figure 1.1, with a quadratic LSF on $\mathbb{R}^2$.

We next discuss methods commonly used to solve problem (1.21) in reliability analysis.

HL-RF Algorithms and Other MPFP Search Methods

The Hasofer and Lind [27], Rackwitz and Flessler [28] (HL-RF) algorithm, was designed in the 70s as an efficient MPFP search method. The basic idea of the HL-RF algorithm is to iteratively locate the MPFP in the standard normal space using a linear approximation of the limit state function at each iteration to update the estimate of the MPFP. This leads to a simple yet efficient Newton-type optimization algorithm outlined in the pseudo-code (1) below, where the usual choice of the step size is $\eta = 1$. Given a threshold value ϵ, the ConvergeCriterion procedure in (1) checks that one of the following conditions is met:

- The distance between the current MPFP estimate and the last estimate is small enough, formally: $\|u_{k+1} - u_k\|_2 \leq \epsilon$.
- The previous and the current estimate of the LSF gradient at the MPFP are close enough: $\|\nabla G(u_{k+1}) - \nabla G(u_k)\|_2 \leq \epsilon$.
- The difference between the current and the previous reliability index estimates is small enough, that is: $|\beta_k - \beta_{k+1}| \leq \epsilon$.

Algorithm 1 HL-RF Algorithm

```
Require: Limit state function $G$, Limit state gradient function $\nabla G$, Initial estimate $\mathbf{u}_0$,
  Convergence criterion ConvergeCriterion, Convergence threshold $\epsilon$, Maximum number of
  iterations $K$, Descent step size $\eta \in (0, 1]$
Ensure: Approximate MPFP $\mathbf{u}^{\text{HL-RF}}$
  Initialize iteration counter $k \leftarrow 0$
  Initialize MPFP estimate $\mathbf{u}_k \leftarrow \mathbf{u}_0$
  while $k < K$ do
      Compute gradient $\nabla G(\mathbf{u}_k)$
      Compute normal vector $\boldsymbol{\alpha}_k \leftarrow \frac{-\nabla G(\mathbf{u}_k)}{\|\nabla G(\mathbf{u}_k)\|_2}$
      Compute reliability estimate $\beta_k \leftarrow \frac{G(\mathbf{u}_k)}{\|\nabla G(\mathbf{u}_k)\|_2} + \langle \boldsymbol{\alpha}_k, \mathbf{u}_k \rangle$
      Compute descent direction $\mathbf{d}_k \leftarrow \beta_k \boldsymbol{\alpha}_k - \mathbf{u}_k$
      Update $\mathbf{u}_{k+1} \leftarrow \mathbf{u}_k + \eta \mathbf{d}_k$          ▷ When $\eta = 1$, $u_{k+1} = \beta_k \boldsymbol{\alpha}_k$
      if ConvergeCriterion($\mathbf{u}_k, \mathbf{u}_{k+1}, \epsilon$) is True then
          break
      end if
      $k \leftarrow k + 1$
  end while
  return $\mathbf{u}^{\text{HL-RF}} = \mathbf{u}_k$
```

However, this method does not come with convergence guarantees and fails to converge in some cases, as noted by Liu and Der Kiureghian [29]. Indeed, the step size η in (1) might be too large and overshoot. Therefore a first improved version, called the MHL-RF (M for modified), was proposed by Liu and Der Kiureghian [29], in which the step size η is adjusted adaptively at each iteration via a line search with merit function m_1 given, by,

$$\mathsf{m}_1(\mathbf{u}; c) = \frac{1}{2}\left\|\mathbf{u} - \frac{\langle \nabla G(\mathbf{u}), \mathbf{u}\rangle}{\|\nabla G(\mathbf{u})\|^2}\nabla G(\mathbf{u})\right\|^2 + \frac{1}{2}c \cdot G(\mathbf{u})^2 \tag{1.29}$$

where c is a fine-tuning parameter. This improves robustness but still has issues, as the descent direction $\mathbf{d}_k$ described in (1) is not necessarily a descent direction for m_1 and m_1 can attain local minima at points which are not optimal for (1.21). A second improved HL-RF algorithm, abbreviated iHL-RF, was introduced in the 90s by Zhang and Der Kiureghian [30] to enhance convergence efficiency through an adaptive step size adjustment using the rule proposed in the 60s by Armijo [31] and the merit function m_2 given by,

$$\mathsf{m}_2(\mathbf{u}; c) = \frac{1}{2}\|\mathbf{u}\|^2 + c|G(\mathbf{u})|. \tag{1.30}$$

We give more details and numerical experiments on the implementations of HL-RF algorithms for Neural Networks in chapter 6, where these algorithms are compared with techniques from the field of Neural Networks Adversarial Robustness (see section 3.2). In addition to the HL-RF algorithms, other methods can be used for MPFP identification in reliability analysis. We give below brief presentations of three alternatives:

- **Gradient Projection Methods (GP):** The Gradient Projection method, as presented by Rosen [32], consists of projecting the gradient of the objective function at each step so that the new estimate should lie in the feasible space (in our case ∂G). In (1.21), the objective function is $f : \mathbf{u} \to \frac{1}{2}\|\mathbf{u}\|^2$ and its gradient is the identity function. The descent direction of the GP method is given by:

  $$\mathbf{d}_k^{\mathrm{GP}} = -(I - \boldsymbol{\alpha}_k\boldsymbol{\alpha}_k^T) \cdot \mathbf{u}_k. \tag{1.31}$$

  Since G is non-linear, the next estimate of the MPFP can lie outside of the failure boundary ∂G. Therefore it is pushed towards it by a second descent with direction $\frac{-G(\mathbf{u})}{\|\nabla G(\mathbf{u})\|_2}\boldsymbol{\alpha}_k$, until the next point satisfies $G(\mathbf{u}) = 0$ within desired accuracy. As above, the descent step size must be adjusted, for example, using Armijo's rule [31].

- **Augmented Lagrangian (AL):** In the context of MPFP search (1.21), the Lagrangian function is given by

$$J(\mathbf{u}, \lambda) = \frac{\|u\|^2}{2} + \lambda G(\mathbf{u}). \tag{1.32}$$

  The augmented Lagrangian is then obtained by adding a term,

$$J_c^{\#}(\mathbf{u}, \lambda) = \frac{\|u\|^2}{2} + \lambda G(\mathbf{u}) + \frac{c}{2} G(\mathbf{u})^2. \tag{1.33}$$

  It can be shown [33] that for c large enough, $\mathbf{u}^*$ is a local minima of $J_c^{\#}$. The Augmented Lagrangian methods then consist of sequentially applying the duality method for the Lagrangian J_c for increasing values of c. While this algorithm theoretically converges, in practice the initialization of c and λ is crucial [30].

- **Sequential Quadratic Programming (SQP):** Sequential Quadratic Programming is an effective method for nonlinear optimization problems like the MPFP search [34]. In the context of (1.21), SQP iteratively solves a quadratic programming subproblem, approximating the objective function and constraints of the original problem. The descent direction $\mathbf{d}_k^{\text{SQP}}$ at iteration k can be obtained as a solution of:

$$\min_{\Delta\mathbf{u} \in \mathbb{R}^d} \frac{1}{2} \Delta\mathbf{u}^T B_k \Delta\mathbf{u} + \nabla\mathbf{u}_k \Delta\mathbf{u} \quad \text{s.t.} \quad G(\mathbf{u}_k) + \nabla G(\mathbf{u}_k)^T \Delta\mathbf{u} = 0. \tag{1.34}$$

In general, the selection of an appropriate search method depends on the limit state function's complexity, the problem's dimensionality, and the computational budget.

Limitations of FORM

From the above analysis, we obtain the following error decomposition:

$$\hat{P}_{\text{F}}^{FORM} = P_{\text{F}} + \underbrace{(P_{\text{F}}^{FORM} - P_{\text{F}})}_{\text{Linear Approximation Error}} + \underbrace{(\hat{P}_{\text{F}}^{FORM} - P_{\text{F}}^{FORM})}_{\text{Optimization-Related Error}}. \tag{1.35}$$

While FORM is computationally efficient and widely used, it has limitations:

- Assumes linearity in the U-Space, which does not hold in many complex systems.
- Relies on the assumption that finding the MPFP is relatively easy, which in practice can be difficult, especially for high dimensional input spaces.

1.3.2 Second-Order Reliability Method

The Second-Order Reliability Method (SORM) is an advanced reliability assessment technique that employs a second-order Taylor expansion to approximate the performance function at the MPFP. Therefore this method accounts for the curvature of the limit state function, offering a more accurate estimation of failure probability, particularly when the limit state function has strong curvature at the failure region boundary ∂G.

Quadratic Approximation

In SORM, the approximation of the performance function at the MPFP is mathematically represented by a second-order Taylor expansion. A crucial assumption for the rest of this section is that G is *twice* continuously differentiable. Specifically, the performance function G is approximated by considering its Hessian matrix $\mathbf{H}_G(\mathbf{u}^*)$ at the MPFP:

$$G(\mathbf{u}) \approx \mathsf{Q}^*(\mathbf{u}) := \underbrace{G(\mathbf{u}^*)}_{=0} + \langle \nabla G(\mathbf{u}^*), (\mathbf{u}-\mathbf{u}^*)\rangle + \frac{1}{2}\langle (\mathbf{u}-\mathbf{u}^*), \mathbf{H}_G(\mathbf{u}^*)(\mathbf{u}-\mathbf{u}^*)\rangle. \quad (1.36)$$

This approximation integrates the second-order effects by incorporating the curvature of the limit state function through the Hessian matrix.

SORM Reliability Estimator

The ideal SORM estimator for the probability of failure, P_{F}, is derived using the above approximation and is expressed using the quadratic function defined in Eq. (1.36) as:

$$P_{\mathrm{F}}^{SORM} = \mathbb{P}[\mathsf{Q}^*(\mathbf{U}) \leq 0] = \int_{\mathbb{R}^d} \mathbb{1}_{\mathsf{Q}^*(\mathbf{u})\leq 0}\phi_d(\mathbf{u})d\mathbf{u}.$$

This integral is intractable in general and it must be approximated. A commonly used approximation introduced by Breitung [35], valid when β_{HL} is large enough, is given by:

$$P_{\mathrm{F}}^{SORM} \approx \Phi(-\beta_{HL}) \prod_{i=1}^{d-1} (1+\beta\kappa_i)^{-\frac{1}{2}}\,, \quad (1.37)$$

where β_{HL} is the Hasofer-Lind reliability index, and $(\kappa_i)_{i\in[1:d-1]}$ represent the eigenvalues of $\mathbf{H}_G(\mathbf{u}^*)$ restricted to the subspace orthogonal to $\boldsymbol{\alpha}_*$, denoted $\mathcal{H} = span(\boldsymbol{\alpha}_*)^\perp$. Of course, this formula is only applicable if: $\min_{i\in[1:d-1]} \kappa_i > -1/\beta$. The product term accounts for the effect of curvatures, thereby refining the probability of failure estimate compared to FORM. A more refined approximation, proposed by Hohenbichler and Rackwitz [36], is known to be more accurate for smaller values β_{HL} and is given by

$$P_{\mathrm{F}}^{SORM} \approx \widetilde{P}_{\mathrm{F}}^{SORM} = \Phi(-\beta_{HL}) \prod_{i=1}^{d-1} \left(1 + \frac{\phi(\beta_{HL})}{\Phi(-\beta_{HL})}\kappa_i\right)^{-\frac{1}{2}}, \tag{1.38}$$

where, $\phi(\cdot)$ is the standard normal distribution's pdf. Both these approximations are obtained by first simplifying the quadratic polynomial Q^* in a carefully chosen orthogonal basis $\mathcal{O} = (e'_1, \cdots, e'_d)$ such that $e'_d = \boldsymbol{\alpha}_*$, and so that

$$\mathsf{Q}^*(\mathbf{u}) \approx \tilde{\mathsf{Q}}(\mathbf{u}') = \beta_{HL} - u'_d + \frac{1}{2}\sum_{i=1}^{d-1} \kappa_i {u'_i}^2, \tag{1.39}$$

where $\mathbf{u}' = (u'_1, \cdots, u'_d)$ is the expression of $\mathbf{u}$ in this orthogonal basis.
As in the FORM procedure, in practice, the MPFP is approximately located via an optimization algorithm. We denote $\mathbf{u}^{\mathsf{opt}}$ the approximate MPFP and direction $\boldsymbol{\alpha}_{\mathsf{opt}}$, and $\hat{\beta}_{HL} = \|\mathbf{u}^{\mathsf{opt}}\|_2$ and $(\hat{\kappa}_i)_{i\in[1:d-1]}$ the eigenvalues of $\mathbf{H}_G$ restricted to $\mathsf{span}({\boldsymbol{\alpha}_{\mathsf{opt}}}^{\perp})$. The SORM is then given by,

$$\hat{P}_{\mathrm{F}}^{SORM} = \Phi(-\hat{\beta}_{HL}) \prod_{i=1}^{d-1} \left(1 + \frac{\phi(\hat{\beta}_{HL})}{\Phi(-\hat{\beta}_{HL})}\hat{\kappa}_i\right)^{-\frac{1}{2}}. \tag{1.40}$$

Efficiency and Accuracy Comparison with FORM

From the analysis above, we obtain the following error decomposition for the SORM method,

$$\hat{P}_{\mathrm{F}}^{SORM} = P_{\mathrm{F}} + \underbrace{(P_{\mathrm{F}}^{SORM} - P_{\mathrm{F}})}_{\text{Quadratic Approx. Error}} + \underbrace{(\widetilde{P}_{\mathrm{F}}^{SORM} - P_{\mathrm{F}}^{SORM})}_{\text{Integral Approx. Error}} + \underbrace{(\hat{P}_{\mathrm{F}}^{SORM} - \widetilde{P}_{\mathrm{F}}^{SORM})}_{\text{Optimization-Related Error}}.$$

While SORM generally provides a more accurate estimate than FORM, especially for non-linear limit state functions, it requires the computation of the second-order derivative, which can be computationally intensive. The efficiency of SORM is lower compared to FORM, particularly when the derivatives are evaluated numerically (via finite-difference or auto-differentiation) as opposed to analytically. The number of performance function evaluations needed for SORM is higher than that required for FORM, making it less efficient in terms of computational resources. In high dimension in particular, the derivation of the Hessian matrix $\mathbf{H}_G$ has a prohibitively high computational cost, as it scales in $O(d^2)$ whereas Gradient derivation has complexity $O(d)$. Moreover, the eigenvalues of $\mathbf{H}_G(\mathbf{u}^*)$ in the subspace $\mathbf{u}^*$ must be computed to obtain the estimate $\hat{P}_{\mathrm{F}}^{SORM}$. This means an additional cost of $O(d^3)$ complexity. In the case of Deep Neural Networks, the Hessian can in principle be obtained rather efficiently via auto-differentiation. However, in use cases where the input dimension is in the order 10^4 or more, SORM becomes too costly. As an important sidenote, here, We presented only the *curvature-fitting* variant of SORM, though there exists a second variant, called the *point-fiiting* SORM, in which the LSF surface ∂G is approximated by a paraboloid function around the MPFP. We refer to the HDR of J-M. Bourinet [37] for more details on this method.

FORM/SORM Estimation with Multiple Design Points

The FORM and SORM estimation procedures presented up to this point heavily rely on both the existence and the *unicity* of the MPFP. When the latter assumption is not met, it is still possible to adapt these estimators, following the approach introduced by Der Kiureghian and Dakessian [38].

1.3.3 First-Order Second-Moment Method

The First-Order Second-Moment (FOSM) method, also known as the Mean-Values First-Order Second-Moment (MVFOSM) method in reliability engineering, provides an efficient approach to reliability analysis. It utilizes a first-order Taylor expansion of the performance function around the mean values of the input variables. We present here an application of this method in the U-space, where the estimator has a very simple form.

Methodological Overview of FOSM

The probability of failure in FOSM is estimated using the standard normal cumulative distribution function (cdf), applied to the normalized mean of the performance function.

FOSM approximates the random performance score $G(\mathbf{U})$ using the following First-Order Taylor expansion at the mean value of the random input, which in the U-space is simply $\mathbf{0}$, i.e.,

$$G(\mathbf{u}) \approx L(\mathbf{u}) = G(\mathbf{0}) + \langle \nabla G(\mathbf{0}), \mathbf{u} \rangle. \tag{1.41}$$

The FOSM estimator is thus given by,

$$P_{\mathrm{F}}^{FOSM} = \Phi\left(-\frac{\mu_L}{\sigma_L}\right)$$

where $\mu_L = \mathbb{E}[L(\mathbf{U})]$ and $\sigma_L = \sqrt{\mathbb{V}[L(\mathbf{U})}$ are the mean and standard deviation of $L(\mathbf{U})$, respectively. Now, μ_L and σ_L are easily derived from the first-order Taylor expansion of G at the mean value $\mathbf{U}$. Indeed, taking equation (1.41) with $\mathbf{u} = \mathbf{U}$, we obtain

$$\mu_L = \mathbb{E}[L(\mathbf{U})] = G(\mathbf{0}) + \langle \nabla G(\mathbf{0}), \underbrace{\mathbb{E}[\mathbf{U}]}_{=0} \rangle = G(\mathbf{0}) \tag{1.42}$$

$$\sigma_L = \sqrt{\mathbb{V}[L(\mathbf{U})]} = \|\nabla G(\mathbf{0})\|_2. \tag{1.43}$$

Therefore, the FOSM estimator in the U-space is simply given by,

$$P_{\mathrm{F}}^{FOSM} = \Phi\left(-\frac{G(\mathbf{0})}{\|\nabla G(\mathbf{0})\|_2}\right). \tag{1.44}$$

Comparison with FORM and Limitations

Although, both FORM and FOSM methods rely on first-order Taylor expansions, the functional assumption underlying FOSM is much stricter. Indeed, while FORM estimation assumes that the failure region boundary ∂G be linear, the FOSM estimation works under the assumption that G itself is a globally linear function. FOSM is known for its computational efficiency, as it does not require the identification of the Most Probable Point (MPFP) like FORM and SORM. In addition, if the limit state function is linear, then it becomes exact. It is particularly suitable for *preliminary* assessments and systems where the performance function's behavior is only moderately non-linear. While FOSM offers simplicity and efficiency, its accuracy will often be less than FORM or SORM's in scenarios involving more complex limit state functions. In particular, this method's simplistic linear approximation at the origin may not accurately capture the actual failure region in complex systems, including for Deep Neural Networks applications.

1.3.4 Line Sampling

Line Sampling, as introduced by Koutsourelakis, Pradlwarter, and Schuëller [39] is an essential reliability engineering estimator used to assess complex systems, typically in the

Standard Normal Space. It follows a stochastic procedure as opposed to other methods introduced in this section. Line Sampling is particularly advantageous for complex and high-dimensional systems, offering a ubiquitous and robust estimation method.

Line Sampling Procedure

The first step of the Line Sampling Procedure is to find an importance direction $\boldsymbol{\alpha}$ which should point to the region of dominant contribution to the probability of failure. Given this importance direction, the Line Sampling procedure involves generating samples in the orthogonal space $\mathcal{H} = span(\boldsymbol{\alpha})^{\perp}$. It then estimates the probability of failure for each sample along a line parallel to this direction. The final estimator $\hat{P}_{\mathrm{F}}^{\mathrm{LS}}$ is calculated as the average of individual estimators $\hat{P}_{\mathrm{F}}^{i}$, making it a valuable tool for assessing system reliability in the presence of random perturbations. The LS algorithm is outlined in the pseudo-code (2) below and illustrated in Figure 1.2.

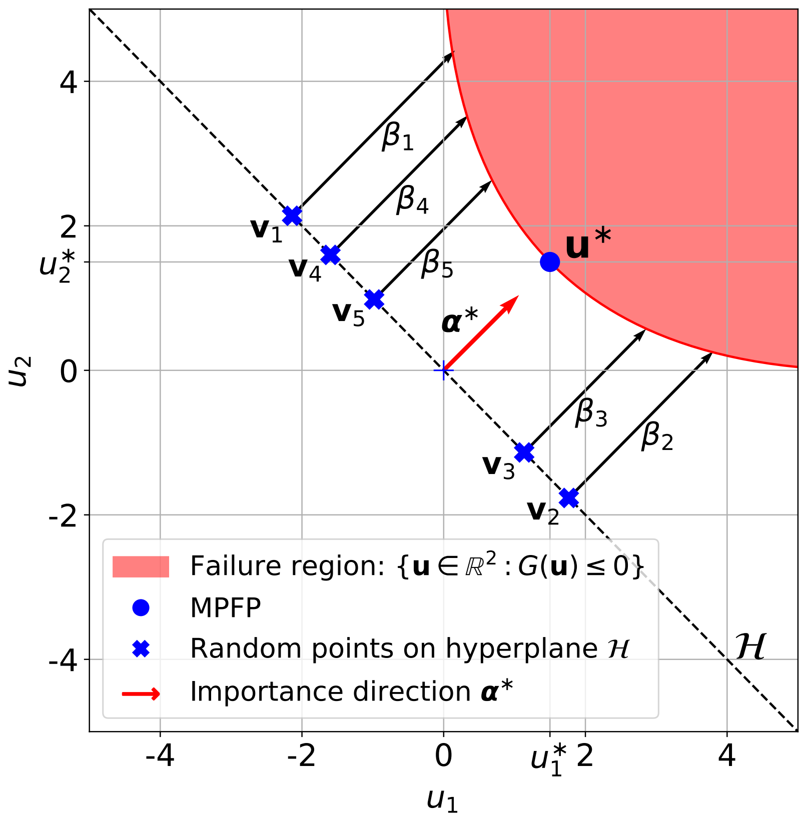


Figure 1.2 – Illustration of Line Sampling in $\mathbb{R}^2$.

Algorithm 2 Line Sampling Procedure

```
Require: Importance direction $\boldsymbol{\alpha}$, Number of samples $N$, Line search algorithm LineSearch
Ensure: Estimated probability of failure $\hat{P}_{\mathrm{F}}^{\mathrm{LS}}$
1: Initialize estimate $\hat{P}_{\mathrm{F}}^{\mathrm{LS}} \leftarrow 0$
2: for $i$ from 1 to $N$ do
3:     Generate a normal sample in $\mathbb{R}^d$: $\mathbf{u}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$
4:     Project sample $\mathbf{u}_i$ on $\mathcal{H}$: $\mathbf{v}_i \leftarrow (\mathbf{I}_d - \boldsymbol{\alpha}\boldsymbol{\alpha}^T) \cdot \mathbf{u}_i$
5:     Use LineSearch to find $\beta_i$ minimum such that $G(\mathbf{v}_i + \beta_i \boldsymbol{\alpha}) = 0$
6:     Compute line estimate $\hat{P}_{\mathrm{F}}^{i} = \Phi(-\beta_i)$
7:     Update estimate $\hat{P}_{\mathrm{F}}^{\mathrm{LS}} \leftarrow \hat{P}_{\mathrm{F}}^{\mathrm{LS}} + \frac{1}{N} \times \hat{P}_{\mathrm{F}}^{i}$
8: end for
```

Advantages and Limitations

While FORM and SORM provide valuable approximations of system reliability, they work best when the LSF is approximately linear or quadratic. In scenarios where the system's behavior deviates significantly from these assumptions, the accuracy of the estimates may be compromised. It's not the case with LS which can adapt to complex failure region boundaries. Another key advantage of LS is the ability to increase the estimation precision by increasing the number of samples, which follows from the law of large numbers. Besides, it may not require knowledge of the LSF gradients if an importance direction can be found without it. With that in mind, LS also has key limitations:

- Line Sampling relies on carefully determining the importance direction $\boldsymbol{\alpha}$, and the estimator's accuracy depends on this selection.
- The convergence of LS to the true failure probability P_{F} is not guaranteed, particularly when several important failure regions exist, in which case it will underestimate it.
- Similarly, if there are several roots of the LSF in one direction, or if the failure region is bidirectional from the origin, then the standard LS technique presented here will fail to converge to the true failure probability.

We note the last point can be addressed by replacing the line search step at line 5 in the pseudo-code above with a more sophisticated root search method. As with FORM and SORM, when several important failure regions exist one can extend the algorithm by performing sequential optimization techniques where each newly discovered failure region is used to modify the LSF so that other failure regions can be found more easily [38].

1.4 Chapter Conclusion

In this chapter, we introduced the foundational principles of Statistical Reliability Engineering, with a focus on classical reliability estimators, including the First Order Reliability Method (FORM), the Second Order Reliability Method (SORM) and Line Sampling (LS). While these methods have their merits and have been applied successfully in various engineering fields, their limitations become apparent when dealing with complex systems, particularly deep learning models, including neural network classifiers.

Statistical Reliability Engineering is the science of assessing the reliability of complex systems in the presence of uncertainties and variations. It leverages probability theory within a measure-theoretic probabilistic framework. The key concepts include the probability of failure (P_F), representing the likelihood of system failure, and the critical level of failure (P_C), which defines an acceptable threshold for system performance. Real-world examples demonstrate the application of this discipline to ensure the reliability of aircraft structural components, medical devices, and emerging technologies like self-driving cars.

One of the primary limitations of classical estimators is their reliance on linear assumptions and the assumption of data distributions approximating the Standard Normal Space (SNS). While it's true that many distributions can be mapped to the Gaussian distribution through transformations, the central issue lies elsewhere. Deep learning systems, including neural network classifiers, inherently exhibit characteristics that challenge the applicability of classical reliability estimators. These characteristics encompass non-linearity, high dimensionality, and complex data distributions.

Indeed, the non-linearity of deep learning models introduces complexities that can render linear approximations insufficient. High dimensionality, a hallmark of deep learning systems, leads to input spaces that classical methods find challenging to navigate efficiently. These limitations serve as a strong motivation to explore advanced techniques for reliability estimation, particularly for rare events in challenging scenarios. In the next chapter, "Introduction to Rare Event Simulation", we delve into these advanced methods, addressing the intricacies of estimating rare event probabilities and their application in the context of deep learning systems.

Rare event simulation offers a promising solution to the reliability concerns posed by neural network classifiers and deep learning systems. By addressing the issues of non-linearity, high dimensionality, and complex data distributions, we can enhance the assessment of reliability and improve the safety and robustness of deep learning applications.

To go further

This chapter is a primer on Statistical Reliability Engineering, covering only the necessary conceptual and mathematical backgrounds to apply static reliability analysis in various fields, including Deep Neural Networks applications, presented in chapter 3.
A complete introduction to statistical reliability is given in the recent monograph of Der Kiureghian [40]. We also omitted important topics in reliability analysis such as kriging and surrogate models, for which we refer the reader to the HDR of J-M. Bourinet [37].

CHAPTER 2

Introduction to Rare Event Simulation

"It is impossible that the improbable will never happen."
– Emil J. Gumbel, Statistics of Extremes [41], 1958

In the context of rare event simulation, various techniques and methods are used to efficiently estimate the probability of events that occur with (very) low probability. This chapter introduces several important strategies and methods for rare event simulation.

2.1 Motivation

2.1.1 Rare Events: Probability Estimation and Simulation

General Motivation. Rare event simulation (RES) is important in various scientific fields, including nuclear physics, systems biology, as well as mechanical and telecommunication engineering [42]. Rare event simulation in physics and engineering addresses phenomena that, despite their low probability (typically $< 10^{-3}$), are pivotal for understanding fundamental processes or have serious safety implications. This motivates the development of robust simulation methods capable of accurately representing these infrequent but critical events. Since each simulation of the underlying physical model often has a high computational cost, these methods should also be as efficient as possible.

Use in Reliability Engineering. In reliability engineering, complex systems are designed so that the probability of a failure P_F is low enough. Thus, in many cases the event of a failure $\mathcal{S}_\mathrm{F}$ is by design a rare event. The reliability methods presented in chapter 1 can, under the appropriate assumptions, estimate rare event probability P_F. However, they do not generate rare events per design. More precisely, they do not output samples from the rare event conditional distribution $\nu_{\mathcal{X}_\mathrm{F}} = \mathcal{L}(\mathbf{X}|\mathbf{X} \in \mathcal{X}_\mathrm{F})$, given by

$$\nu_{\mathcal{X}_\mathrm{F}}(d\mathbf{x}) := \frac{\mathbb{1}_{\mathcal{X}_\mathrm{F}}(\mathbf{x})\nu_\mathbf{X}(d\mathbf{x})}{\nu_\mathbf{X}(\mathcal{X}_\mathrm{F})} = \frac{\mathbb{1}_{\mathcal{X}_\mathrm{F}}(\mathbf{x})}{P_\mathrm{F}} p_X(\mathbf{x})d\mathbf{x}. \tag{2.1}$$

In contrast, the methods introduced in the present chapter output in one go both an estimation $\hat{P}_\mathrm{F}$ of the failure probability and a (potentially random) number N_F of *weighted* rare event samples $\mathbf{X}_1^{(\mathrm{F})}, \dots, \mathbf{X}_{N_F}^{(\mathrm{F})}$ which, at least asymptotically, allow to compute expectations under the exact conditional law $\nu_{\mathcal{X}_\mathrm{F}}$. This can be a useful plus, e.g., in the case of Neural Networks, rare events can be used to augment the training dataset and make the model more robust (see section 3.2.2 in the next chapter on model hardening). The most decisive advantage of rare event simulation, however, (shared only with Line Sampling among methods in chapter 1), is the ability to increase the estimation precision at will by increasing the computational cost in return. The efficiency of these methods is therefore quantified as the gain in precision obtained per unit of computational cost. The most direct rare event sampling technique is the crude Monte Carlo simulation. Unfortunately, as is shown below, this method is fundamentally inefficient for rare events. This inadequacy justifies the development of more sophisticated Monte Carlo sampling techniques, presented in the following sections throughout this chapter.

Conditional Distribution in the U-space. Instead of simulating rare events directly in the original space, it is sometimes possible to work in a transformed standard normal space (see section 1.2.2). As with methods presented in chapter 1, working in the U-space is often simpler. In that case, the conditional distribution of interest is π_F, given by

$$\pi_\mathrm{F}(d\mathbf{u}) := \frac{\mathbb{1}_{\mathcal{U}_\mathrm{F}}(\mathbf{u})\pi_0(d\mathbf{u})}{\pi_0(\mathcal{U}_\mathrm{F})} = \frac{\mathbb{1}_{\mathcal{U}_\mathrm{F}}(\mathbf{u})}{P_\mathrm{F}} \phi_d(\mathbf{u})d\mathbf{u}. \tag{2.2}$$

where ϕ_d is the pdf of the standard normal distribution on $\mathbb{R}^d$.

2.1.2 Inefficiency of Crude Monte Carlo Simulation

Monte Carlo methods are widely used for estimating probabilities and expectations by simulating random variables [43]. While these methods are powerful due to their simplicity and generality, they can exhibit significant limitations when applied to the simulation of rare events, which are critical in the fields of reliability engineering and risk assessment.

Monte Carlo Integration. For the rest of this chapter, we consider a given absolutely continuous random vector $\mathbf{Z}$ taking values in a Borel set $\mathcal{Z} \subset \mathbb{R}^d$ and with pdf $p_{\mathbf{Z}}$. Given a bounded and non-negative measurable function $\varphi : \mathcal{Z} \mapsto \mathbb{R}_+$, the objective of Monte Carlo integration is to estimate the integral $I(\varphi)$, defined as

$$I(\varphi) := \int_{\mathcal{Z}} \varphi(\mathbf{z}) p_{\mathbf{Z}}(\mathbf{z}) d\mathbf{z} = \mathbb{E}[\varphi(\mathbf{Z})], \tag{2.3}$$

as precisely as possible, by simulating random variables. The Monte Carlo method was initially formalized in the 1940s by Stanislas Ulam and John von Neumann [44] and has since then become a crucial tool in practically all fields of science and engineering.

In its simplest form, which we call Crude Monte Carlo (CMC), it consists of simulating independent samples drawn from the same distribution as $\mathbf{Z}$ and then estimating $I(\varphi)$ by the empirical mean of φ over these samples. Mathematically, this can be expressed as

$$\hat{I}_N^{\text{CMC}}(\varphi) = \frac{1}{N} \sum_{i=1}^{N} \varphi(\mathbf{Z}_i), \tag{2.4}$$

where N is the number of samples and $\mathbf{Z}_1, \cdots, \mathbf{Z}_N \overset{\text{i.i.d.}}{\sim} p_{\mathbf{Z}}$ represent the random samples. A natural assumption for the convergence of this estimator is that $I(\varphi)$ is finite. Since φ is bounded, this is the case, and so the strong law of large numbers ensures that

$$\mathbb{P}[\lim_{N \to \infty} \hat{I}_N^{\text{CMC}}(\varphi) = I(\varphi)] = 1. \tag{2.5}$$

However, a pivotal assumption here is the ability to generate independent samples with pdf $p_{\mathbf{Z}}$. More fundamentally, given that simulations are typically computer-based, this process hinges on the ability to simulate a sequence of independent random numbers using computers, which is a research area in itself [45]. The three main solutions are pseudorandom number generators (PRNGs) which mimic randomness with deterministic iterative algorithms [46], "true" random number generators (TRNGs) which gather entropy from hardware or sensors [47], and quasi-Monte Carlo methods as an alternative to uniform random sampling [48]. In this thesis, we rely on the availability of an efficient PRNG for generating samples from the uniform distribution over $[0, 1]^d$. Thus, the challenge is to simulate non-uniform random variables using these uniform samples. When the components of $\mathbf{Z}$ are independent, and their cdfs are known, a straightforward component-wise transformation of uniform samples in $[0, 1]^d$ is applicable. However, in more complex scenarios, such as when $p_{\mathbf{Z}}$ is only known up to a normalization constant, other methods like Rejection Sampling and Monte Carlo by Markov Chains (MCMC) [49] can be necessary. We refer the reader to appendix A for more details on these sampling techniques and next focus on the application of Crude Monte Carlo in the context of rare event analysis.

CMC for Rare Event Simulation. The inefficiency of Crude Monte Carlo simulation in the context of rare events can be attributed to its reliance on the Law of Large Numbers. We seek to estimate the probability $\mathbb{P}(F)$ of a rare event $F \in \mathcal{F}$ defined as $F = [\mathbf{Z} \in A]$ where A is a measurable subset of $\mathcal{Z}$. This corresponds to estimating the integral $I(\mathbb{1}_A)$ where $\mathbb{1}_A$ is the indicator function of the set A, that is $\mathbb{1}_A(x)$ is 1 if x is in A and 0 otherwise. In this context, Crude Monte Carlo estimation is obtained by simulating a large number of independent samples and then calculating the proportion of samples that fall into the event A. Mathematically, this can be expressed as:

$$\hat{P}_{\mathrm{F}}^{\mathrm{CMC}} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}_A(\mathbf{Z}_i) = \frac{1}{N}\sum_{i=1}^{N} B_i, \tag{2.6}$$

where $B_1 = \mathbb{1}_A(\mathbf{Z}_1), \ldots, B_N = \mathbb{1}_A(\mathbf{Z}_N)$. It is clear that $B_1, \ldots, B_N$ are independent Bernoulli random variables with parameter $p = \mathbb{P}(F)$. Consequently, the variance of $\hat{P}_{\mathrm{F}}^{\mathrm{CMC}}$ is given by:

$$\mathbb{V}[\hat{P}_{\mathrm{F}}^{\mathrm{CMC}}] = \frac{1}{N^2}\sum_{i=1}^{N} \mathbb{V}[B_i] = \frac{\mathbb{P}(F)(1-\mathbb{P}(F))}{N}. \tag{2.7}$$

In practice, this variance is estimated by its empirical counterpart $\hat{\mathbb{V}}[\hat{P}_{\mathrm{F}}^{\mathrm{CMC}}]$ given by

$$\hat{\mathbb{V}}[\hat{P}_{\mathrm{F}}^{\mathrm{CMC}}] = \frac{\hat{P}_{\mathrm{F}}^{\mathrm{CMC}}(1-\hat{P}_{\mathrm{F}}^{\mathrm{CMC}})}{N}. \tag{2.8}$$

The fundamental problem with CMC in rare event simulation arises from the very low probability of the event F. As a consequence, the vast majority of the N samples do not contribute to the sum in equation (2.6), leading to an estimate $\hat{P}_{\mathrm{F}}^{\mathrm{CMC}}$ with a high relative variance. This inefficiency can be quantified via the relative standard error, also called the relative error or the coefficient of variation, denoted by $\Delta_N[\hat{P}_{\mathrm{F}}^{\mathrm{CMC}}]$ and defined as:

$$\Delta_N[\hat{P}_{\mathrm{F}}^{\mathrm{CMC}}] := \frac{\sqrt{\mathbb{V}(\hat{P}_{\mathrm{F}}^{\mathrm{CMC}})}}{\mathbb{E}[\hat{P}_{\mathrm{F}}^{\mathrm{CMC}}]} = \sqrt{\frac{1-\mathbb{P}(F)}{N\cdot\mathbb{P}(F)}} \approx \sqrt{\frac{1}{N\cdot\mathbb{P}(F)}}. \tag{2.9}$$

Therefore the coefficient of variation of CMC scales $\mathbb{P}(F)^{-1/2}$, which limits its application to rare event probability estimation. Following the central limit theorem (CLT), which holds here since $\mathbb{E}[\mathbb{1}_A(\mathbf{Z})^2] = \mathbb{P}(F) < \infty$, an asymptotic confidence interval (CI) for $\hat{P}_{\mathrm{F}}^{\mathrm{CMC}}$ at level $(1-\alpha)$ can be computed using the following formula:

$$\mathrm{CI}^{\mathrm{CMC}} = \left[\hat{P}_{\mathrm{F}}^{\mathrm{CMC}} - z_{1-\alpha/2}\sqrt{\frac{\mathbb{P}(F)(1-\mathbb{P}(F))}{N}}, \hat{P}_{\mathrm{F}}^{\mathrm{CMC}} + z_{1-\alpha/2}\sqrt{\frac{\mathbb{P}(F)(1-\mathbb{P}(F))}{N}}\right]$$

where $z_{1-\alpha/2}$ is the critical value from the standard normal distribution corresponding to the tail probability $\alpha/2$, i.e. $z_{1-\alpha/2} = \Phi^{-1}(1-\alpha/2)$. This CI is unusable in practice as it involves $\mathbb{P}(F)$, however using equation (2.8) one obtains the following empirical CI:

$$\hat{\mathrm{CI}}^{\mathrm{CMC}} = \left[\hat{P}_{\mathrm{F}}^{\mathrm{CMC}} - z_{1-\alpha/2}\sqrt{\frac{\hat{P}_{\mathrm{F}}^{\mathrm{CMC}}(1-\hat{P}_{\mathrm{F}}^{\mathrm{CMC}})}{N}}, \hat{P}_{\mathrm{F}}^{\mathrm{CMC}} + z_{1-\alpha/2}\sqrt{\frac{\hat{P}_{\mathrm{F}}^{\mathrm{CMC}}(1-\hat{P}_{\mathrm{F}}^{\mathrm{CMC}})}{N}}\right].$$

Importantly, these CIs only make sense when $N \gg \frac{1}{\mathbb{P}(F)}$. In practice, this means that an impractically large number of samples is needed to estimate or obtain reliable bounds on low probabilities, making CMC an intrinsically inefficient method for rare event simulation.

We next illustrate this issue in the context of reliability analysis. See also Figure 2.1.

Example: Estimating the Probability of a System Failure

We now consider the problem of assessing the reliability of a complex engineering system where failure occurs only under rare conditions. Let's say the probability of failure is $P_{\mathrm{F}} = 10^{-7}$. Using Crude Monte Carlo, we would simulate the system's operation numerous times and count the number of failures to estimate this probability, in each simulation. Using equation (2.9) the minimum number of simulations N_{min} achieve a relative standard error less than 5%, can be calculated as:

$$N_{\mathrm{min}} = \frac{1-P_{\mathrm{F}}}{P_{\mathrm{F}} \times 0.05^2} \simeq 4 \times 10^9. \tag{2.10}$$

Thus, about 4 billion simulations are needed for a relatively satisfying accuracy. Such a large number of simulations can be computationally prohibitive, especially when each call to the LSF is expensive, demonstrating the impracticality of Crude Monte Carlo for rare event estimation. This inefficiency motivates the need for advanced variance reduction techniques such as Importance Sampling, which we explore in the next two sections.

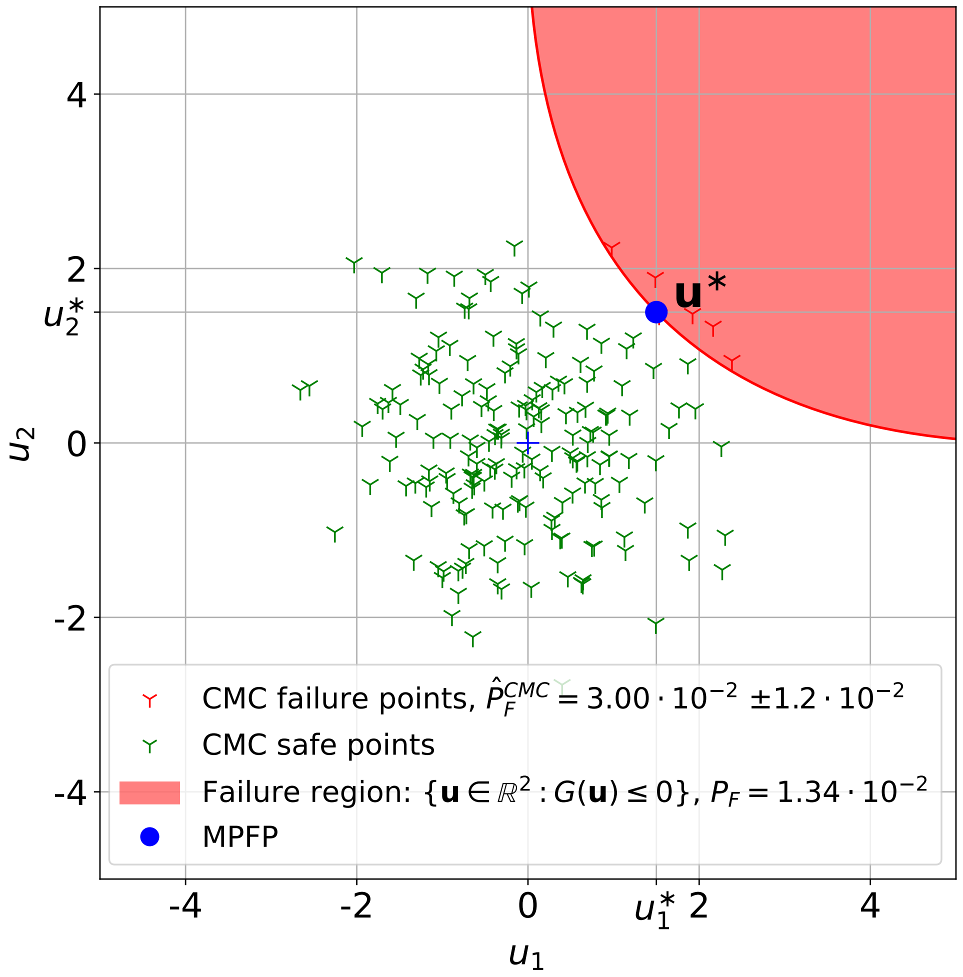


Figure 2.1 – Illustration of CMC, with 200 samples, in $\mathbb{R}^2$.

2.2 Basic Importance Sampling

2.2.1 General Principles

Importance Sampling (IS) is a variance reduction technique for Monte Carlo simulation, which is particularly common in rare event simulation. We first present IS in the general setting introduced in section 2.1.2, where the objective is to approximate the integral $I(\varphi)$, defined in equation (2.3), as efficiently as possible. In IS, an alternative pdf, denoted q, is introduced to improve computational efficiency [50]. The associated distribution is called the *importance distribution*. From this point forward, by a slight abuse of notation and for simplicity, we sometimes identify a probability distribution by its pdf. Throughout the next two sections, we also consider an alternative random vector $\widetilde{\mathbf{Z}}$ with distribution q. To properly define the IS estimator, q must satisfy the following condition:

$$\forall \mathbf{z} \in \mathcal{Z}, q(\mathbf{z}) = 0 \implies \varphi(\mathbf{z}) p_{\mathbf{Z}}(\mathbf{z}) = 0. \tag{C1}$$

We denote by $\mathcal{Q}$ the set of eligibles pdfs on $\mathcal{Z}$ satisfying (C1) and we suppose that $q \in \mathcal{Q}$. The fundamental idea of IS is then to rewrite the integral $I(\varphi)$ as an expectation under the importance distribution. Indeed,

$$\mathbb{E}[\varphi(\mathbf{Z})] = \int_{\mathcal{Z}\setminus\partial[\varphi p_{\mathbf{Z}}]} \varphi(\mathbf{z}) p_{\mathbf{Z}}(\mathbf{z}) d\mathbf{z} = \int_{\mathcal{Z}\setminus\partial[q]} \varphi(\mathbf{z}) \frac{p_{\mathbf{Z}}(\mathbf{z})}{q(\mathbf{z})} q(\mathbf{z}) d\mathbf{z} = \mathbb{E}[\varphi(\widetilde{\mathbf{Z}}) w_q(\widetilde{\mathbf{Z}})], \tag{2.11}$$

where $\partial[f] := \{\mathbf{z} \in \mathcal{Z} : f(\mathbf{z}) = 0\}$ for all $f : \mathcal{Z} \mapsto \mathbb{R}$ and w_q is a weight function given by,

$$\forall \mathbf{z} \in \mathcal{Z} \setminus \partial[q], w_q(\mathbf{z}) := \frac{p_{\mathbf{Z}}(\mathbf{z})}{q(\mathbf{z})}. \tag{2.12}$$

This weight function is sometimes called the likelihood ratio, as it involves a ratio of densities [43]. A practical assumption for IS is the ability to efficiently draw independent samples with distribution q. Additionally, we assume, for now, that the weight function w can be efficiently evaluated (which is not true, e.g., if the normalization constant of p_Z is unknown). The basic importance sampling estimator of $I(\varphi)$ can then be formulated as,

$$\hat{I}_N^{\text{IS}}(\varphi; q) = \frac{1}{N} \sum_{i=1}^{N} w_q(\widetilde{\mathbf{Z}}_i) \varphi(\widetilde{\mathbf{Z}}_i), \tag{2.13}$$

where $\widetilde{\mathbf{Z}}_1, \ldots, \widetilde{\mathbf{Z}}_N$ are i.i.d. samples from distribution q. As a direct result from equation (2.11), we note that it is an unbiased estimator of integral $I(\varphi)$, i.e. $\mathbb{E}[\hat{I}_N^{\text{IS}}(\varphi; q)] = I(\varphi)$. Thus, by the strong law of large numbers, we have,

$$\mathbb{P}[\lim_{N\to\infty} \hat{I}_N^{\text{IS}}(\varphi; q) = I(\varphi)] = 1. \tag{2.14}$$

Importantly, at this point, there is no guarantee that IS can be more efficient than CMC. The variance $\sigma_q^2(\varphi) = \mathbb{V}[w_q(\widetilde{\mathbf{Z}})\varphi(\widetilde{\mathbf{Z}})]$ plays a crucial role in understanding the computational efficiency of Importance Sampling. Indeed, by the independence of $\widetilde{\mathbf{Z}}_1, \ldots, \widetilde{\mathbf{Z}}_N$ the variance of the IS estimator is readily computed as:

$$\mathbb{V}[\hat{I}_N^{\text{IS}}(\varphi; q)] = \frac{\sigma_q^2(\varphi)}{N}. \tag{2.15}$$

Therefore, it is natural to seek an importance distribution q that minimizes this variance. When such a minimizer exists, we call it the optimal density and denote it by q^*. In our case, since φ is a non-negative and bounded function, this optimal distribution exists as long as $I(\varphi)$ is non-zero, it then attains a zero variance and is given by,

$$q^*(\mathbf{z}) = \frac{\varphi(\mathbf{z})p_{\mathbf{Z}}(\mathbf{z})}{I(\varphi)}. \tag{2.16}$$

Indeed, on the one hand, it is clear q^* is a pdf that satisfies condition (C1), and on the other one can readily check $\sigma_q^2(\varphi) = 0$ iff $q = q^*$, noting that

$$\begin{aligned}
\sigma_q^2(\varphi) &= \int_{\mathcal{Z}\setminus\partial[q]} \left(\frac{\varphi(\mathbf{z})p_{\mathbf{Z}}(\mathbf{z})}{q(\mathbf{z})} - I(\varphi)\right)^2 q(\mathbf{z})d\mathbf{z} \\
&= \int_{\mathcal{Z}\setminus\partial[q]} \left(\frac{q^*(\mathbf{z})I(\varphi) - q(\mathbf{z})I(\varphi)}{q(\mathbf{z})}\right)^2 q(\mathbf{z})d\mathbf{z} \\
&= I(\varphi)^2 \int_{\mathcal{Z}\setminus\partial[q]} \left(\frac{q^*(\mathbf{z})}{q(\mathbf{z})} - 1\right)^2 q(\mathbf{z})d\mathbf{z}.
\end{aligned} \tag{2.17}$$

Crucially, the normalization constant of q^* is *unknown*, as it is the integral want to estimate. As mentioned above, sampling is then difficult and requires methods like MCMC (see appendix A). Moreover, we cannot evaluate the weight function which involves $I(\varphi)$. Thus a common approach is to use a pdf q in $\mathcal{Q}$, with which both sampling and weight evaluation are simple and that is close enough to the optimal density q^* in a certain sense.

Self-normalized IS. Until now we supposed that one could readily evaluate the weight function w_q. As mentioned above, however, this is not the case when either the importance density q or the nominal density $p_{\mathbf{Z}}$ itself are only known up to a normalization constant. Let us assume now that $q(\cdot) = \frac{q_u(\cdot)}{C_q}$ and $p_Z(\cdot) = \frac{p_u(\cdot)}{C_z}$, where q_u and p_u are known non-negative functions on $\mathcal{Z}$ but C_q and C_z are unknown normalization constants. Then it is still possible to apply a modified version of IS called self-normalized IS, in which both the integral $I(\varphi)$ and the ratio of normalization constants $\frac{C_q}{C_z}$ are estimated simultaneously. However, in that case, it is required that q satisfies the following condition:

$$\forall \mathbf{z} \in \mathcal{Z}, q(\mathbf{z}) = 0 \implies p_{\mathbf{Z}}(\mathbf{z}) = 0, \tag{C2}$$

which is stronger than condition C1. The self-normalized IS estimator is then given by,

$$\hat{I}_N^{\text{SNIS}}(\varphi; q) = \frac{\frac{1}{N}\sum_{i=1}^N \widetilde{w}_q(\widetilde{\mathbf{Z}}_i)\varphi(\widetilde{\mathbf{Z}}_i)}{\frac{1}{N}\sum_{i=1}^N \widetilde{w}_q(\widetilde{\mathbf{Z}}_i)} \tag{2.18}$$

where $\widetilde{\mathbf{Z}}_1, \ldots, \widetilde{\mathbf{Z}}_N$ are i.i.d. samples with density q and $\widetilde{w}_q$ is the unnormalized weight function given by,

$$\forall \mathbf{z} \in \mathcal{Z} \setminus \partial[q], \widetilde{w}_q(\mathbf{z}) := \frac{p_u(\mathbf{z})}{q_u(\mathbf{z})} \left(= \frac{C_q}{C_z} w_q(\mathbf{z})\right). \tag{2.19}$$

The next result assures us that the self-normalized IS estimator is consistent.

Theorem 2. *Let $p_Z(\cdot) = \frac{p_u(\cdot)}{C_z}$ be a pdf on $\mathcal{Z}$ and $q(\cdot) = \frac{q_u(\cdot)}{C_q}$ be a density satisfying (C2). Then the self-normalized IS estimator proposed in (2.18) converges almost surely to the integral $I(\varphi)$, i.e.*

$$\mathbb{P}[\lim_{N\to\infty} \hat{I}_N^{SNIS}(\varphi; q) = I(\varphi)] = 1. \tag{2.20}$$

Moreover, the empirical mean of the unnormalized weights converges a.s. to $\frac{C_q}{C_z}$, i.e.

$$\mathbb{P}\left[\lim_{N\to\infty} \frac{1}{N}\sum_{i=1}^N \widetilde{w}_q(\widetilde{\mathbf{Z}}_i) = \frac{C_q}{C_z}\right] = 1. \tag{2.21}$$

Proof. The strong LLN assures the numerator in the equation (2.18) converges a.s. to its expected value $\mathbb{E}[\widetilde{w}_q(\widetilde{\mathbf{Z}})\varphi(\widetilde{\mathbf{Z}})] = \frac{C_q}{C_z}\mathbb{E}[w_q(\widetilde{\mathbf{Z}})\varphi(\widetilde{\mathbf{Z}})] = \frac{C_q}{C_z}I(\varphi)$. Similarly, the denominator converges a.s. to its expectation $\mathbb{E}[\widetilde{w}_q(\widetilde{\mathbf{Z}})] = \frac{C_q}{C_z} \neq 0$. Thus, by the continuous mapping theorem, as the ratio of these random sequences, $\hat{I}_N^{\text{SNIS}}$ converges a.s. to $I(\varphi)$. $\square$

Mixture IS and Defensive IS. A commonly used strategy, called Mixture IS, is to use a mixture of M densities for the importance density: $q = \sum_{m=1}^{M} \alpha_m q_m$ where $q_1, \ldots, q_m$ are densities on $\mathcal{Z}$ and $\alpha_1, \ldots, \alpha_M$ are positive reals in $(0,1)$ such that $\sum_{m=1}^{M} \alpha_m = 1$ [43]. A known failure of IS happens when the rate of decay of the importance density q does not match with that of $q^{*2} \propto (\varphi(\mathbf{z}) p_{\mathbf{Z}}(\mathbf{z}))^2$ outside of the important region (see example 1 in [51]). This can be understood by rewriting $\sigma_q^2(\varphi)$ as,

$$\sigma_q^2(\varphi) = \int_{\mathcal{Z}} \frac{(\varphi(\mathbf{z}) p_{\mathbf{Z}}(\mathbf{z}))^2}{q(\mathbf{z})} dz - I(\varphi)^2. \tag{2.22}$$

A special case of Mixture IS, called Defensive IS, was introduced by Hesterberg [52] to mitigate this phenomenon. In Defensive IS, the proposed importance density q_{DIS} is a mixture between the nominal density $p_{\mathbf{Z}}$ and an importance density q close enough to the optimal density q^*, formally,

$$\forall \mathbf{z} \in \mathcal{Z}, q_{\text{DIS}}(\mathbf{z}) = \lambda p_{\mathbf{Z}}(\mathbf{z}) + (1-\lambda) q(\mathbf{z}) \tag{2.23}$$

where $\lambda \in (0,1)$ can be interpreted as the rate of "defensiveness". The important point here is that

$$\sigma_{q_{\text{DIS}}}^2(\varphi) \leq \frac{1}{\lambda} \int_{\mathcal{Z}} \varphi(\mathbf{z})^2 p_{\mathbf{Z}}(\mathbf{z}) dz - I(\varphi)^2 \leq \frac{1}{\lambda}(\sigma_{p_{\mathbf{Z}}}^2(\varphi) + (1-\lambda) I(\varphi)^2). \tag{2.24}$$

One can easily see that $\sigma_{p_{\mathbf{Z}}}^2(\varphi)$ is the variance of the CMC estimator. Therefore, unfortunately, this is not a good bound when CMC is very inefficient, as in rare event simulation. Still, this illustrates how mixtures can be used to increase estimation robustness in IS.

Challenges in IS. Multimodal distributions, i.e. distributions with more than one mode, present a unique challenge in Importance Sampling. While it is not the focus of this thesis, it is an important issue to have in mind. We refer to the work of Oh and Berger [53], where multimodal distributions are tackled using a mixture of importance densities. In more recent work of Qiu and Wang [54], propose a more data-driven strategy: they use instead specific DNN-based isoprobabilistic transforms called Normalizing Flows (see section A.4). Another important challenge faced in IS is that of high-dimensional input spaces. Being especially relevant in the context of the reliability estimation of DNNs, this issue deserves a thorough and separate discussion and is the topic of section 2.2.3.

We next focus on the application of IS in the context of rare event simulation.

2.2.2 Importance Sampling for Rare Event Simulation

As mentioned above, in rare event simulation, φ is the indicator function $\mathbb{1}_A$ and we seek to estimate the probability $P_{\mathrm{F}} = \mathbb{P}[\mathbf{Z} \in A]$. Henceforth, we suppose, for simplicity, that it is possible to work in the U-space, for example, using an isoprobabilistic transform. Thus, we identify the random vector $\mathbf{Z}$ with a canonical Gaussian vector $\mathbf{U}$ in $\mathbb{R}^d$, so that $\mathcal{Z} = \mathbb{R}^d$ and the nominal density is ϕ_d, i.e. the pdf of the standard Gaussian law on $\mathbb{R}^d$. This assumption is quite restrictive, as most methods introduced in this chapter can, in principle, be adapted for any absolutely continuous distribution on $\mathbb{R}^d$. However, we note this is a working assumption for most of the methods presented in the previous chapter. Additionally, in the same vein, we now suppose that A is the zero-superlevel set of a given continuous score function $S : \mathbb{R}^d \mapsto \mathbb{R}$, that is:

$$A = \{\mathbf{u} \in \mathbb{R}^d : S(\mathbf{u}) \geq 0\}. \tag{2.25}$$

Thus, the probability of the rare event P_{F} can be written as,

$$P_{\mathrm{F}} = \mathbb{P}[S(\mathbf{U}) \geq 0] = \int_{\mathbb{R}^d} \mathbb{1}_{\mathbb{R}_+}(S(\mathbf{u}))\phi_d(\mathbf{u})d\mathbf{u}. \tag{2.26}$$

In this context, the set $\mathcal{Q}$ of eligible pdfs consists of densities q satisfiying the condition:

$$\forall \mathbf{u} \in \mathbb{R}^d, S(\mathbf{u}) \geq 0 \implies q(\mathbf{u}) > 0. \tag{C3}$$

Given an importance density q in $\mathcal{Q}$, the IS estimator of P_{F} is given by,

$$\hat{P}_{\mathrm{F}}^{\mathrm{IS}} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}_A(\widetilde{\mathbf{U}}_i) w_q(\widetilde{\mathbf{U}}_i) = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}_{S(\widetilde{\mathbf{U}}_i)\geq 0} w_q(\widetilde{\mathbf{U}}_i), \tag{2.27}$$

where $\widetilde{\mathbf{U}}_1, \dots, \widetilde{\mathbf{U}}_N$ are i.i.d. samples from q, and w_q is the function defined in eq. (2.12). As in the general case, if q is in $\mathcal{Q}$, the IS estimator is unbiased, i.e. $\mathbb{E}[\hat{P}_{\mathrm{F}}^{\mathrm{IS}}] = P_{\mathrm{F}}$, and thus by the strong LLN, its estimator is consistent: $\mathbb{P}[\lim_{N\to\infty} \hat{P}_{\mathrm{F}}^{\mathrm{IS}} = P_{\mathrm{F}}] = 1$. The variance of this estimator is $\mathbb{V}[\hat{P}_{\mathrm{F}}^{\mathrm{IS}}] = \frac{\sigma_q^2}{N}$ where $\sigma_q^2 := \mathbb{V}[\mathbb{1}_{S(\widetilde{\mathbf{U}})\geq 0} w_q(\widetilde{\mathbf{U}})] = \mathbb{E}[\mathbb{1}_{S(\mathbf{U})\geq 0} w_q(\mathbf{U})] - P_{\mathrm{F}}^2$. Consequently, assuming that $\mathbb{E}[w_q(\widetilde{\mathbf{U}})^2] = \mathbb{E}[w_q(\mathbf{U})]$ is finite, the CLT ensures that $\sqrt{N}(\hat{P}_{\mathrm{F}}^{\mathrm{IS}} - P_{\mathrm{F}})$ converges in law to the centered normal distribution with variance σ_q^2, i.e.

$$\forall x \in \mathbb{R}, \lim_{N\to\infty} \mathbb{P}\left[\frac{\sqrt{N}(\hat{P}_{\mathrm{F}}^{\mathrm{IS}} - P_{\mathrm{F}})}{\sigma_q} \leq x\right] = \Phi\left(x\right). \tag{2.28}$$

Therefore, an asymptotic CI at level $(1-\alpha)$ for IS given by:

$$\mathrm{CI}^{\mathrm{IS}} = \left[\hat{P}_{\mathrm{F}}^{\mathrm{IS}} - z_{1-\alpha/2}\sqrt{\frac{\sigma_q^2}{N}}, \hat{P}_{\mathrm{F}}^{\mathrm{IS}} + z_{1-\alpha/2}\sqrt{\frac{\sigma_q^2}{N}}\right]$$

where $z_{1-\alpha/2} = \Phi^{-1}(1-\alpha/2)$. This CI is unusable in practice as it involves σ_q^2, however, one can use the following empirical CI:

$$\hat{\mathrm{CI}}^{\mathrm{IS}} = \left[\hat{P}_{\mathrm{F}}^{\mathrm{IS}} - z_{1-\alpha/2}\sqrt{\frac{\hat{\sigma}_q^2}{N}}, \hat{P}_{\mathrm{F}}^{\mathrm{IS}} + z_{1-\alpha/2}\sqrt{\frac{\hat{\sigma}_q^2}{N}}\right]$$

where $\hat{\sigma}_q^2 = \frac{1}{N-1}\sum_{i=1}^N \left(\mathbb{1}_A(\widetilde{\mathbf{U}}_i)w_q(\widetilde{\mathbf{U}}_i) - \hat{P}_{\mathrm{F}}^{\mathrm{IS}}\right)^2$ is the unbiased estimator of the variance σ_q^2.

This variance is zero, and therefore minimal, at the optimal density q^*, given above in equation (2.16) for a generic non-negative integrand φ. In rare event simulation, it coincides with the pdf of the conditional distribution $\pi_F = \mathcal{L}(\mathbf{U}|S(\mathbf{U}) \geq 0)$, that is:

$$q^*(\mathbf{u}) = \frac{\mathbb{1}_{S(\mathbf{u})\geq 0}\phi_d(\mathbf{u})}{P_{\mathrm{F}}} \tag{2.29}$$

However, as we saw before, this density is impractical, as sampling from an unnormalized density is difficult and the associated weight function w_q cannot be evaluated without knowledge of the probability P_{F}, which we seek to estimate. Moreover, even if we could generate i.i.d. samples $\mathbf{U}_1^*, \ldots, \mathbf{U}_N^*$, from q^* it is not possible to use self-normalized IS either, as this density does not satisfy the condition (C2). An alternative is thus to find an importance density fairly close to the optimum, in a sense we formalize below. We first write the relative error, a.k.a the coefficient of variation, of the IS estimator as

$$\Delta_N[\hat{P}_{\mathrm{F}}^{\mathrm{IS}}] := \frac{\sqrt{\mathbb{V}[\hat{P}_{\mathrm{F}}^{\mathrm{IS}}]}}{\mathbb{E}[\hat{P}_{\mathrm{F}}^{\mathrm{IS}}]} = \sqrt{\frac{\sigma_q^2}{NP_{\mathrm{F}}^2}}. \tag{2.30}$$

Using equation (2.17), we obtain:

$$\Delta_N^2[\hat{P}_{\mathrm{F}}^{\mathrm{IS}}] = \frac{1}{N}\int_{\mathbb{R}^d}\left(\frac{q^*(\mathbf{u})}{q(\mathbf{u})} - 1\right)^2 q(\mathbf{u})d\mathbf{u} = \frac{D_{\chi^2}(q^*||q)}{N}, \tag{2.31}$$

where $D_{\chi^2}(\cdot||\cdot)$ represents the chi-squared divergence [55]. Thus, IS is more sample efficient than CMC as long as $D_{\chi^2}(q^*||q) < P_{\mathrm{F}}^{-1}$, though other quantities are relevant. In particular, the Kullback-Leibler divergence can be used in practice for optimization, see section 2.3.2. We end this section by presenting a heuristic from the field of reliability engineering.

MPFP-Based Importance Sampling

In the context of reliability analysis, Melchers [56] proposed a heuristic based on the MPFP, denoted as $\mathbf{u}^*$, introduced in section 1.3.1 for the FORM/SORM estimators. The importance distribution is choosen as $\mathcal{N}(\mathbf{u}^*, \mathbf{I}_d)$, thus the importance density is $q_{\text{MPP}}(\mathbf{u}) := \frac{1}{(2\pi)^{d/2}} \exp\left(-\frac{\|\mathbf{u}-\mathbf{u}^*\|^2}{2}\right)$. The rationale here is that failure inputs are sampled more frequently near the MPFP (vs. near the origin, see Figure 2.2). Moreover, those samples would exhibit a relatively low likelihood ratio variance, as the MPFP is the failure point *closest* to the origin (see the example just below). It is also advantageous computationally, as it trades off optimization in the space of pdfs with optimization in $\mathbb{R}^d$. Unfortunately, as discussed before, this optimization task is difficult in high-dimension. Yet, we show in chapter 6 that it is feasible for DNNs and leads to efficient estimation.

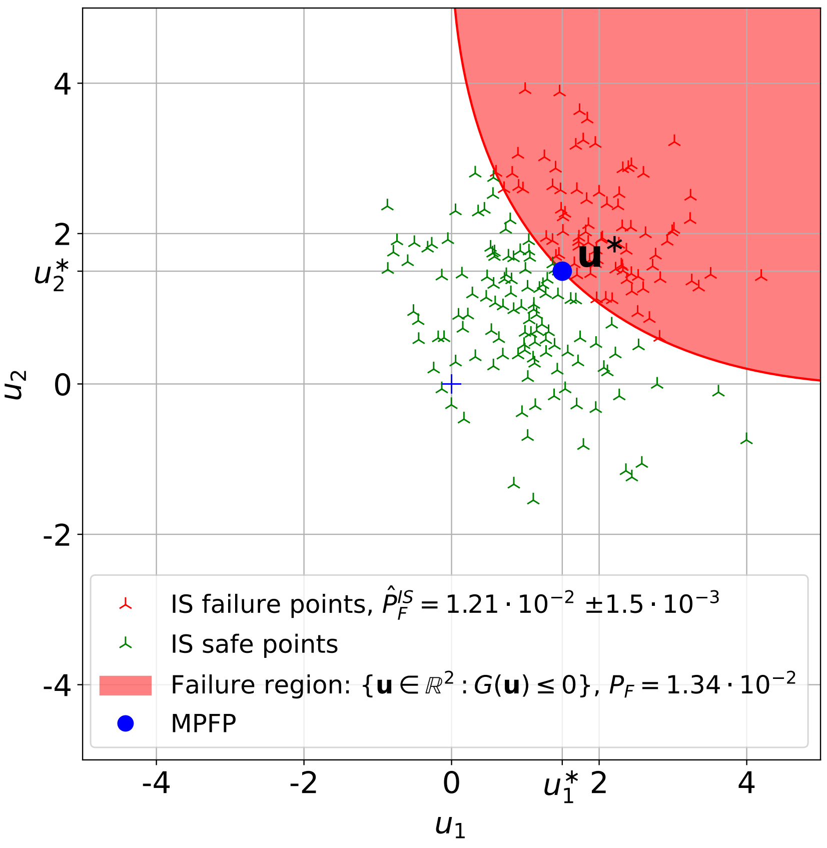


Figure 2.2 – Illustration of MPFP-based Importance Sampling, with 200 samples, in $\mathbb{R}^2$.

2.2.3 Importance Sampling in High-Dimensional Spaces

In high-dimensional settings, the application of Importance Sampling methods faces significant challenges. The primary concern is the phenomenon of *weight degeneracy*, which can lead to inefficient estimations even when using close-to-optimal importance densities.

The Curse of Dimensionality and Weight Degeneracy

In high-dimensional spaces, the curse of dimensionality (COD) amplifies the difficulty of efficiently in regions that contribute to the rare event A and adversely affects the variance of the likelihood ratio. COD was studied and observed in the context of importance sampling by Li, Bengtsson, and Bickel [57]. In particular, in a quite general IS setting, they show that the maximum of the normalized weights converges to 1 as both dimension d and the number of samples N go to infinity, while $\frac{\log(N)}{d^{1/3}} \overset{(N,d)\to\infty}{\to} 0$. Consequently, even adaptive Importance Sampling methods, presented in the next section, may fail to effectively use a large number of samples in these critical regions. Put differently, the weight degeneracy phenomenon arises when the quasi-totality of the samples drawn from the importance density q receives extremely small or even zero-importance normalized weights. In rare event simulation, this occurs as the rare event becomes more difficult to locate, making it increasingly unlikely for samples to both fall into the rare event region A *and* have reasonably high likelihood under the nominal density ϕ_d, resulting in negligible estimation weights $w_{q,A}(\mathbf{u}) = \mathbb{1}_A(\mathbf{u})w_q(\mathbf{u})$. We next study the effect of the dimension on w_q alone.

Examples: Gaussian Mean Shift and Variance Change

We now illustrate the weight degeneracy phenomenon with two simple examples of Importance Sampling in the U-space. However, we only look at the effect of dimension on the weight function w_q, which limits the scope of these examples.

Gaussian Mean Shift. The first example is the Gaussian mean shift, where we sample $\widetilde{\mathbf{U}}_i$ independently from the distribution $\mathcal{N}(\lambda\boldsymbol{\alpha}, I_d)$ where $\boldsymbol{\alpha}$ is a unit vector in $\mathbb{R}^d$ and λ a positive real. The importance weight is easily computed as:

$$\forall \mathbf{u} \in \mathbb{R}^d, w_q(\mathbf{u}) = \exp(-\lambda\langle \mathbf{u}, \boldsymbol{\alpha}\rangle + \frac{\lambda^2}{2}). \tag{2.32}$$

The second moment of the likelihood ratio is therefore given by

$$\mathbb{E}[w_q(\widetilde{\mathbf{U}})^2] = \frac{e^{2\lambda^2}}{(2\pi)^{d/2}} \int_{\mathbb{R}^d} \exp(-2\lambda\langle \mathbf{u}, \boldsymbol{\alpha}\rangle) e^{-\|\mathbf{u}\|_2^2 + 2\lambda\langle \mathbf{u}, \boldsymbol{\alpha}\rangle - \lambda^2 \|\boldsymbol{\alpha}\|_2^2} d\mathbf{u} \tag{2.33}$$

$$= e^{\lambda^2} \underbrace{\int_{\mathbb{R}^d} \frac{e^{-\|\mathbf{u}\|_2^2}}{(2\pi)^{d/2}} d\mathbf{u}}_{=1} = e^{\lambda^2}. \tag{2.34}$$

An important remark here is that the square expected value of the likelihood ratio does not depend directly on the vector space dimension d, but rather on the shift vector norm λ. Thus introducing a small shift in a large number of dimensions, e.g. taking $\boldsymbol{\alpha} = \left(\frac{1}{\sqrt{d}}, \dots, \frac{1}{\sqrt{d}}\right)$ and $\lambda = 1$, will have the same effect on weight degeneracy as introducing a larger shift on a single dimension, e.g. taking $\boldsymbol{\alpha} = e_1 = (1, 0, \dots, 0)$ and $\lambda = 1$.

Isotropic Variance Change. In this second example, the variance is changed such that $\mathbf{U} \overset{q}{\sim} \mathcal{N}(\mathbf{0}, \sigma^2 I_d)$. In this case, the likelihood ratio is given by,

$$\forall \mathbf{u} \in \mathbb{R}^d, w_q(\mathbf{u}) = \sigma^d \exp\left(-\frac{1}{2}\|\mathbf{u}\|_2^2 \left(1 - \frac{1}{\sigma^2}\right)\right). \tag{2.35}$$

Therefore the second moment of the importance weight is computed as,

$$\mathbb{E}[w_q(\widetilde{\mathbf{U}})^2] = \mathbb{E}[w_q(\mathbf{U})] = \frac{\sigma^d}{(2\pi)^{d/2}} \int_{\mathbb{R}^d} \exp\left(-\frac{\|\mathbf{u}\|_2^2}{2}\left(1 - \frac{1}{\sigma^2}\right)\right) e^{-\frac{\|\mathbf{u}\|_2^2}{2}} d\mathbf{u}$$

$$= \frac{\sigma^d}{(2\pi)^{d/2}} \int_{\mathbb{R}^d} \exp\left(-\frac{\|\mathbf{u}\|_2^2}{2}\left(2 - \frac{1}{\sigma^2}\right)\right) d\mathbf{u}. \tag{2.36}$$

If $\sigma > \frac{1}{\sqrt{2}}$, we obtain,

$$\mathbb{E}[w_q(\widetilde{\mathbf{U}})^2] = \left(\frac{\sigma}{\sqrt{2 - \frac{1}{\sigma^2}}}\right)^d = \left(\frac{\sigma^2}{\sqrt{2\sigma^2 - 1}}\right)^d. \tag{2.37}$$

Thus we see that even a small isotropic change in the variance can lead to acute weight degeneracy in high-dimensional spaces.

Anisotropic Variance Change. We now consider anisotropic variance, where the intensity of the variance change varies along directions. We restrict the change to diagonal covariance matrices so that $\mathbf{U} \overset{q}{\sim} \mathcal{N}(\mathbf{0}, \mathsf{diag}(\sigma_1, \dots, \sigma_d))$ where $\sigma_1, \dots, \sigma_d$ are positive reals and $\mathsf{diag} : \mathbb{R}^d \mapsto \mathbb{R}^{d\times d}$ is the mapping defined by

$$\mathsf{diag} : \mathbb{R}^d \ni (a_1, \dots, a_d) \to \begin{bmatrix} a_1 & & \\ & \ddots & \\ & & a_d \end{bmatrix}.$$

When $\min_{i\in[1:d]} \sigma_i > \frac{1}{\sqrt{2}}$, we obtain by the same method as above :

$$\mathbb{E}[w_q(\widetilde{\mathbf{U}})^2] = \prod_{i\in[1:d]} \frac{\sigma_i^2}{\sqrt{2\sigma_i^2 - 1}}. \tag{2.38}$$

Thus, while even small isotropic variance changes in high-dimension quickly lead to weight degeneracy, a sparse, targeted, variance change does not lead to weight degeneracy.

Impact on Estimation Accuracy and Diagnostics

The weight degeneracy phenomenon significantly affects the accuracy of Importance Sampling. To understand this, it is natural to consider the variance of IS. As discussed in section 2.2.2, the variance can be expressed as $\mathbb{V}[\hat{P}_{\mathrm{F}}^{\mathrm{IS}}] = \frac{\sigma_q^2}{N} = \frac{\mathbb{E}[\mathbb{1}_A(\widetilde{\mathbf{U}})w_q(\widetilde{\mathbf{U}})^2] - P_{\mathrm{F}}^2}{N}$. Now, roughly speaking, the more the repartition of the weights is unbalanced, the higher the expectation $\mathbb{E}[\mathbb{1}_A(\widetilde{\mathbf{U}})w_q(\widetilde{\mathbf{U}})^2]$, and in turn σ_q^2, will be. Note that, in practice, this variance is estimated as

$$\hat{\mathbb{V}}[\hat{P}_{\mathrm{F}}^{\mathrm{IS}}] = \frac{\hat{\sigma}_q^2}{N} = \frac{1}{N(N-1)} \sum_{i=1} (\mathbb{1}_A(\widetilde{\mathbf{U}}_i) w_q(\widetilde{\mathbf{U}}_i)^2 - \hat{P}_{\mathrm{F}}^{\mathrm{IS}})^2. \tag{2.39}$$

Unfortunately, this estimate depends on the weights and thus is generally not a good diagnostic of (non)convergence [58]. The weight degeneracy phenomenon is often diagnosed using the effective sample size (ESS), which quantifies the number of effectively contributing samples. The original idea behind the ESS, in the context of self-normalized IS, can be found in the technical report of Kong [59]. The ideal ESS, which cannot be computed in practice, is defined as $N_{\text{eff}} := N \frac{\mathbb{V}[\hat{P}_{\mathrm{F}}^{\mathrm{CMC}}]}{\mathbb{V}[\hat{P}_{\mathrm{F}}^{\mathrm{IS}}]}$. Thus, a small ESS indicates a relative inefficiency of IS compared to CMC. Using the delta method [60], Kong [59] shows that this quantity can be approximated by $\widetilde{N}_{\text{eff}} = \frac{N}{1+\mathbb{V}[w_q(\widetilde{\mathbf{U}})]}$, which in turn can be estimated by its empirical counterpart $\hat{N}_{\text{eff}}$, given by

$$\hat{N}_{\text{eff}} = \frac{1}{\sum_{i\in[1:N]} \bar{w}_q^2(\widetilde{\mathbf{U}}_i)^2}. \tag{2.40}$$

where $\bar{w}_q(\widetilde{\mathbf{U}}_i) = \frac{w_q(\widetilde{\mathbf{U}}_i)}{\sum_{i=1}^N w_q(\widetilde{\mathbf{U}}_i)}$ is the normalized weight of i-th the sample.

While $\hat{N}_{\text{eff}}$ is independent of A, it is possible to replace w_q by $w_{q,A}(\cdot) = \mathbb{1}_A(\cdot) w_q(\cdot)$. Elvira, Martino, and Robert [61] show that, despite its popularity, the ESS comes with many drawbacks. To cite only one, $\hat{N}_{\text{eff}}$ is lower bounded by 1, whereas (by its original definition) it should clearly be (close to) 0 in cases where IS has (quasi-)infinite variance. Chatterjee and Diaconis [58] propose an alternative diagnostic quantity D_n, defined as,

$$D_N := \max_{i\in[1:N]} \bar{w}_q(\widetilde{\mathbf{U}}_i) = \frac{\max_{i\in[1:N]} w_q(\widetilde{\mathbf{U}}_i)}{\sum_{i=1}^N w_q(\widetilde{\mathbf{U}}_i)} \in [0,1], \tag{2.41}$$

in which case IS is deemed to have converged if D_n is less than a given value, say 0.05. The use of this metric is motivated by a detailed analysis of its expectation $d_N = \mathbb{E}[D_N]$, in particular when performing IS with Gibbs measures[1], see [58, Theorem 3.5].

Alternatives. Many strategies have been proposed to improve basic IS, including:

- **Adaptive Importance Sampling [62]:** Presented next in section 2.3, Adaptive Importance Sampling (AIS) is an iterative procedure in which the importance density is *progressively* drawn near the target density q^* at each step, using samples from the importance density of the previous step. In addition, intermediary target densities are used at each non-terminal step to reduce the risk of weight degeneracy.
- **Multilevel Splitting Methods [63]:** The main idea of importance splitting is to divide the estimation problem into more manageable subproblems, and then combine the results. This is done by managing a set of samples called particles, which are killed, resampled, and mutated, in a similar fashion to genetic algorithms. This reduces the severity of weight degeneracy at each iteration. This approach, also known as Subset Simulation in reliability analysis, is presented in section 2.4.
- **Dimension Reduction Methods [64]:** Using projections to carefully choosen lower-dimensional spaces one can reduce the effective dimensionality of the problem, making the sampling process more manageable and alleviating weight degeneracy. El Masri, Morio, and Simatos [65], in particular, propose to use a one-dimension projection within an AIS procedure and determined *without* gradient computation.

Nonetheless, efficiently handling weight degeneracy in high-dimensional spaces is an active area of research and a key issue in applying reliability analysis for Deep Learning.

1. Families of distributions $(\nu_\beta)_{\beta\geq 0}$ with pdfs $p_\beta(\mathbf{u}) \propto \exp(-\beta V(\mathbf{u}))\phi_d(\mathbf{u})$, see sections 5.2.2 and A.3.4.

2.3 Adaptive Importance Sampling

Adaptive Importance Sampling (AIS) refines the Importance Sampling approach by iteratively updating the importance density q. We refer the reader to the survey of Bugallo, Elvira, Martino, *et al.* [66] for a comprehensive overview of AIS. In AIS, the importance density begins as an initial guess q_0 and is adaptively modified through subsequent iterations. This iterative adjustment, which approximates the target density q^*, is crucial in cases where the probability landscape is complex and a fixed importance function is insufficient. Since approximating q^* from the start is difficult, these methods also use a multilevel approach, where an intermediary target density q_k^{target} is used at each step k. In that case, the algorithm typically stops after some k^* iterations, s.t. $q_{k^*}^{\text{target}} \approx q^*$. In the context of rare event simulation, we first define, for all $\gamma \in \mathbb{R}$, subset $A(\gamma)$ as the γ-superlevel set of the score function S, that is

$$A(\gamma) := \{\mathbf{u} \in \mathbb{R}^d : S(\mathbf{u}) \geq \gamma\}. \tag{2.42}$$

Accordingly, we note $P_\gamma = \mathbb{P}[\mathbf{U} \in A(\gamma)]$ and $q^{(\gamma)}$ the pdf of the conditional distribution $\pi_\gamma := \mathcal{L}(\mathbf{U}|S(\mathbf{U}) > \gamma)$, that is,

$$q^{(\gamma)}(\mathbf{u}) := \frac{\mathbb{1}_{A(\gamma)}(\mathbf{u})\phi_d(\mathbf{u})}{P_\gamma} = \frac{\mathbb{1}_{S(\mathbf{u})\geq\gamma}(\mathbf{u})\phi_d(\mathbf{u})}{P_\gamma}. \tag{2.43}$$

Iterative Refinement. Iterative refinement in AIS involves updating the bias distribution based on feedback from the current sample set. This process can be represented at a high level of abstraction as:

$$q_{k+1} = \mathsf{UpdateFunction}(q_k, \{\widetilde{\mathbf{U}}_i^{(k)}\}_{i\in[1:N]}, \gamma_{k+1}), \tag{2.44}$$

where q_{k+1} is the updated distribution for the next iteration, $\{\widetilde{\mathbf{U}}_i^{(k)}\}_{i\in[1:N]}$ are the samples of the k-th iteration and γ_{k+1} is the ρ-quantile of the score over these samples, see line 7 below. Various update strategies are discussed throughout this section. The common attribute of these strategies, however, is that q_{k+1} is chosen to be as close as possible, in a sense to be defined, to the intermediary target $q^{(\gamma_{k+1})}$. This approach is called Multilevel AIS, *distinct* from Multilevel Monte Carlo, which consists of combining estimates from simulations of varying fidelities for better efficiency [67]. Given a generic update function $\mathsf{UpdateFunction}$ the AIS procedure is outlined in the pseudo-code (3).

Algorithm 3 Generic Multilevel AIS Procedure For Rare Event Simulation

```
Require: Initial biasing distribution pdf $q_0$, Number of samples $N$, Updating rule for the importance density UpdateFunction, Maximum number of iterations $K$, Quantile parameter $\rho$
Ensure: Rare Event Probability estimate $\hat{P}_{\mathrm{F}}^{AIS}$
 1: Stop ← False
 2: $k \leftarrow 0$
 3: while Stop is False and $k < K$ do
 4:     Generate samples $\{\widetilde{\mathbf{U}}_i^{(k)}\}_{i\in[1:N]} \sim q_k^{\otimes N}$
 5:     Compute scores $(S_1, \dots, S_N) = (S(\widetilde{\mathbf{U}}_1^{(k)}), \dots, S(\widetilde{\mathbf{U}}_N^{(k)}))$
 6:     Compute ordered scores $S_{(1)} \leq S_{(2)} \leq \dots \leq S_{(N)}$
 7:     Compute threshold level $\gamma_{k+1} = \min(S_{(\lfloor(1-\rho)N\rfloor)}, 0)$
 8:     Update $q_{k+1} = \mathsf{UpdateFunction}(q_k, \{\widetilde{\mathbf{U}}_i^{(k)}\}, \gamma_{k+1})$
 9:     $k \leftarrow k + 1$
10:     if $\gamma_k = 0$ then
11:         Stop ← True
12:     end if
13: end while
14: Estimate $\hat{P}_{\mathrm{F}}^{\mathrm{AIS}} = \frac{1}{N}\sum_{i=1}^N \mathbb{1}_A(\widetilde{\mathbf{U}}_i^{(k)}) w_{q_k}(\widetilde{\mathbf{U}}_i^{(k)})$
15: return $\hat{P}_{\mathrm{F}}^{\mathrm{AIS}}$
```

Adaptive Importance Sampling methods are generally divided into parametric and nonparametric methods, which are presented in subsections 2.3.1 and 2.3.3 respectively.

2.3.1 Parametric Adaptive Importance Sampling

A common strategy in AIS is to restrict the biasing distribution to a parametric family of distributions $\mathcal{Q}_\Theta = \{Q_{\boldsymbol{\theta}}(d\mathbf{u}) = q_\theta(\mathbf{u})d\mathbf{u}, \boldsymbol{\theta} \in \Theta\}$, where Θ is generally a vector space. It is convenient for $\mathcal{Q}_\Theta$ to contain the nominal probability distribution, i.e. $\nu_{\mathbf{X}}$ in the X-space or $\pi = \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$ in the U-space. Now, at each iteration k of the algorithm, the pdf is fully characterized by the current parameter $\boldsymbol{\theta}_k$, as $q_k = q_{\boldsymbol{\theta}_k}$. Informally the pdf update procedure $\mathsf{UpdateFunction}$ in line 8 is replaced by a parameter update process $\mathsf{UpdateParam}$. More precisely, at each step k, the ideal next parameter $\boldsymbol{\theta}^*_{k+1}$ is given by,

$$\boldsymbol{\theta}^*_{k+1} = \operatorname*{argmin}_{\boldsymbol{\theta}\in\Theta} D(q^{(\gamma_{k+1})} || q_{\boldsymbol{\theta}}), \tag{2.45}$$

where D is a divergence (not necessarily a distance) on the space of eligible pdfs $\mathcal{Q}$, which, for the sake of this thesis, is a functional on $\mathcal{Q}^2$ satisfying two following conditions:

$$\text{Non-negativity}: \quad \forall p, q \in \mathcal{Q}, D(p||q) \geq 0 \tag{2.46}$$

$$\text{Positivity}: \quad \forall p, q \in \mathcal{Q}, D(p||q) = 0 \Leftrightarrow p = q \text{ a.e.} \tag{2.47}$$

In many cases $D(p||q)$ can be expressed as an expectation w.r.t. the importance density q. This is the case, in particular, for f-divergences [68], defined by

$$D_f(p||q) := \mathbb{E}_q \left[f\left(\frac{p(\widetilde{\mathbf{U}})}{q(\widetilde{\mathbf{U}})} \right) \right], \tag{2.48}$$

for all convex function $f : \mathbb{R}_+ \to \mathbb{R}_+$ such that $f(x) = 0$ iff $x = 1$. In this situation, while it is not possible to solve the optimization program (2.45) directly, one can either use a stochastic approximation (SA) method [69], or define $\boldsymbol{\theta}_{k+1}$ as,

$$\boldsymbol{\theta}_{k+1} = \underset{\boldsymbol{\theta}\in\Theta}{\operatorname{argmin}} \sum_{i=1}^{N} f\left(\frac{q^{(\gamma_{k+1})}(\widetilde{\mathbf{U}}_i^{(k)})}{q_{\boldsymbol{\theta}}(\widetilde{\mathbf{U}}_i^{(k)})} \right). \tag{2.49}$$

Thus, this reduces to locally optimizing an empirical performance metric w.r.t. to $\boldsymbol{\theta}$. Common choices include the empirical variance, taking $f(x) = (x-1)^2$ (corresponding to the χ^2-divergence, see eq. (2.31)), and the empirical cross-entropy, with $f(x) = x \log x$, leading respectively to the variance minimization (VM) and cross-entropy (CE) variants of parametric AIS. Chan, Glynn, and Kroese [70] compare these methods and show that, under reasonable assumptions, they converge to the same parameter $\boldsymbol{\theta}^*$, as $P_{\mathrm{F}} \to 0$.

Gaussian Families. A natural family when working in the U-space, is the Gaussian family $\mathcal{Q}^{\text{Gauss}} := \{\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) | \boldsymbol{\theta} = (\boldsymbol{\mu}, \boldsymbol{\Sigma}) \in \mathbb{R}^d \times S^d_{++}\}$ where S^d_{++} denotes the set of positive-definite symmetrical matrices of size $d \times d$. In high-dimension, estimating a matrix of size $d \times d$ can be too costly. Thus, we are inclined to work instead with the smaller family of isotropic Gaussian distributions $\mathcal{Q}^{\text{I-Gauss}} := \{\mathcal{N}(\boldsymbol{\mu}, \sigma^2 \mathbf{I}_d) | \boldsymbol{\theta} = (\boldsymbol{\mu}, \sigma) \in \mathbb{R}^d \times \mathbb{R}_{++}\}$ or with the less restrictive class of Gaussian distributions with uncorrelated components $\mathcal{Q}^{\text{U-Gauss}} := \{\mathcal{N}(\boldsymbol{\mu}, \mathsf{diag}(\sigma_1, \ldots, \sigma_d)) | \boldsymbol{\theta} = (\boldsymbol{\mu}, \sigma_1, \sigma_2, \ldots, \sigma_d) \in \mathbb{R}^d \times (\frac{1}{\sqrt{2}}, \infty)^d\}$. However, as we saw in the second example in section 2.2.3, isotropic variance changes in high dimension readily lead to weight degeneracy, so we favor the use of the latter family of distribution.

We next focus on the cross-entropy approach, which is predominant in parametric AIS.

2.3.2 Cross-Entropy Based Importance Sampling

The Cross-Entropy (CE) method, as presented, e.g., in the textbook of Rubinstein and Kroese [62, Chapter 8], is a powerful tool also used for solving both complex optimization and sampling problems. It has significant application in the domain of Importance Sampling, in refining the biasing distribution for efficient rare event simulation. The CE method is rooted in information theory, where it was formulated to minimize the Kullback-Leibler divergence, a measure of the difference between two probability distributions. In the context of Adaptive Importance Sampling, the CE-AIS method iteratively updates the importance density to minimize its cross-entropy w.r.t. to a target density.

Mathematical Formulation

The mathematical core of the CE method in this context ultimately involves minimizing the Kullback-Leibler (KL) divergence between the biasing distribution and the conditional rare event distribution, i.e. $\nu_{\mathcal{X}_\mathrm{F}}$ in the X-space or π_F in the U-space. Now, for two continuous distributions P and Q on $\mathbb{R}^d$, such that P is absolutely continuous w.r.t. Q [2], the KL divergence is given by:

$$D_{KL}(P||Q) := \int_{\mathbb{R}^d} P(d\mathbf{x}) \log\left(\frac{dP}{dQ}(\mathbf{x})\right) \tag{2.50}$$

where $\frac{dP}{dQ}$ is the Radon-Nikodym derivative of P w.r.t. Q. When both these distributions are admit densities w.r.t. to the Lebesgue measure, denoted respectively p and q, equation (2.51) can be rewritten as,

$$D_{KL}(p||q) := \int_{\mathbb{R}^d} \log\left(\frac{p(\mathbf{x})}{q(\mathbf{x})}\right) p(\mathbf{x})d\mathbf{x}. \tag{2.51}$$

In CE-AIS, the iterative process adjusts q to minimize $D_{KL}(q^*||q) = D_{KL}(\pi_\mathrm{F}||Q)$, leading to a distribution that is more focused on the rare event. To understand why it may be a good idea to minimize this divergence we first note that:

$$\frac{\mathbb{E}[\mathbb{1}_{\mathcal{U}_\mathrm{F}}(\widetilde{\mathbf{U}})^2 w_q(\widetilde{\mathbf{U}})^2]}{P_\mathrm{F}^2} = \int_{\mathbb{R}^d} \frac{\phi_d(\mathbf{u})}{q(\mathbf{u})} \frac{\mathbb{1}_{\mathcal{U}_\mathrm{F}}(\mathbf{u})^2}{P_\mathrm{F}^2} \phi_d(\mathbf{u}) d\mathbf{u} = \mathbb{E}_{\mathbf{U}\sim\pi_\mathrm{F}}\left[\frac{\mathbb{1}_{\mathcal{U}_\mathrm{F}}(\mathbf{U})}{P_\mathrm{F}} w_q(\mathbf{U})\right]. \tag{2.52}$$

2. One also says that Q dominates P and writes '$P \ll Q$', meaning here that for all Borel set $A \in \mathcal{B}(\mathbb{R}^d)$, $Q(A) = 0$ implies that $P(A) = 0$.

Now by Jensen's Inequality, we have,

$$\log\left(\mathbb{E}_{\pi_{\mathrm{F}}}\left[\frac{\mathbb{1}_{\mathcal{U}_{\mathrm{F}}}(\mathbf{U})}{P_{\mathrm{F}}}w_q(\mathbf{U})\right]\right) \geq \mathbb{E}_{\pi_{\mathrm{F}}}\left[\log\left(\frac{\mathbb{1}_{\mathcal{U}_{\mathrm{F}}}(\mathbf{U})}{P_{\mathrm{F}}}w_q(\mathbf{U})\right)\right] = \int_{\mathcal{U}_{\mathrm{F}}}\log\left(\frac{\phi_{\mathrm{F}}(\mathbf{u})}{q(\mathbf{u})}\right)\phi_{\mathrm{F}}(\mathbf{u})d\mathbf{u} \tag{2.53}$$

$$= D_{KL}(\pi_{\mathrm{F}}||Q). \tag{2.54}$$

Thus, we obtain the following lower bound:

$$\frac{\mathbb{E}_{\mathbf{U}\sim Q}[\mathbb{1}_{\mathcal{U}_{\mathrm{F}}}(\mathbf{U})w_q(\mathbf{U})^2]}{P_{\mathrm{F}}^2} \geq e^{D_{KL}(\pi_{\mathrm{F}}||Q)}. \tag{2.55}$$

In terms of relative error of the IS estimator $\Delta_N[\hat{P}_{\mathrm{F}}^{\mathrm{IS}}]$, this rewrites as,

$$\Delta_N[\hat{P}_{\mathrm{F}}^{\mathrm{IS}}] \geq e^{D_{KL}(\pi_{\mathrm{F}}||Q)} - 1 = e^{D_{KL}(q^*||q)} - 1. \tag{2.56}$$

This inequality shows that the relative error grows at least exponentially as a function of the Kullback-Leilber divergence between the optimal density and the importance density. Moreover, Chatterjee and Diaconis [58, Theorem 1.3] provide both an upper and a lower bound, both of order $e^{D_{KL}(\pi_{\mathrm{F}}||Q)}$, on the number of samples needed in IS to approach P_{F} within a given level of accuracy. On another note, as a result of equation (2.13), the inequality (2.56) above can be interpreted in terms of divergence on probability distribution, as it translates to:

$$\forall q \in \mathcal{Q}, D_{\chi^2}(q^*||q) \geq e^{D_{KL}(q^*||q)} - 1, \tag{2.57}$$

which is a classical result in both information theory [71] and distribution testing [55].

Optimization in Practice. An important point for optimization, is that $D_{KL}(q^*||q_{\boldsymbol{\theta}})$ can be rewritten as

$$\begin{aligned} D_{KL}(q^*||q_{\boldsymbol{\theta}}) &= -\log(P_{\mathrm{F}}) + \int_{\mathbf{U}\sim\mathcal{U}_{\mathrm{F}}}\log\left(\frac{\phi_d(\mathbf{u})}{q_{\boldsymbol{\theta}}(\mathbf{u})}\right)q^*(\mathbf{u})d\mathbf{u} \\ &= -\log(P_{\mathrm{F}}) + \mathbb{E}_{\pi_{\mathrm{F}}}\left[\log(w_{q_{\boldsymbol{\theta}}}(\mathbf{U}))\right] \\ &= -\frac{1}{P_{\mathrm{F}}}\mathbb{E}_{q_{\boldsymbol{\theta}}}[\mathbb{1}_A(\widetilde{\mathbf{U}})w_{q_{\boldsymbol{\theta}}}(\widetilde{\mathbf{U}})\log(q_{\boldsymbol{\theta}}(\widetilde{\mathbf{U}}))] + C \propto -\mathbb{E}_{q_{\boldsymbol{\theta}}}[\mathbb{1}_A(\widetilde{\mathbf{U}})w_{q_{\boldsymbol{\theta}}}(\widetilde{\mathbf{U}})\log(q_{\boldsymbol{\theta}}(\widetilde{\mathbf{U}}))] \end{aligned}$$

where C is independent of the value of $\boldsymbol{\theta}$. Thus, one can readily optimize an empirical counterpart of $D_{KL}(q^*||q_{\boldsymbol{\theta}})$ as proposed above (see line 9 in the pseudocode below).

Algorithmic Implementation

The implementation of the (multilevel) Cross-Entropy method in Adaptive Importance Sampling (CE-AIS) is outlined in pseudocode (4) below.

Algorithm 4 Multilevel CE-AIS Procedure For Rare Event Simulation

```
Require: Parametric family of densities $\{q_{\boldsymbol{\theta}} | \boldsymbol{\theta} \in \Theta\}$, Initial parameter $\boldsymbol{\theta}_0 \in \Theta$, Number of samples $N$, Updating rule for the importance density UpdateFunction, Maximum number of iterations $K$, Quantile parameter $\rho$
Ensure: Rare Event Probability estimate $\hat{P}_{\mathrm{F}}^{AIS}$
 1: Stop $\leftarrow$ False
 2: $q_0 \leftarrow q_{\boldsymbol{\theta}_0}$
 3: $k \leftarrow 0$
 4: while Stop is False and $k < K$ do
 5:     Generate samples $\{\widetilde{\mathbf{U}}_i^{(k)}\}_{i \in [1:N]} \sim q_k^{\otimes N}$
 6:     Compute scores $(S_1, \ldots, S_N) = (S(\widetilde{\mathbf{U}}_1^{(k)}), \ldots, S(\widetilde{\mathbf{U}}_N^{(k)}))$
 7:     Compute ordered scores $S_{(1)} \leq S_{(2)} \leq \ldots \leq S_{(N)}$
 8:     Compute threshold level $\gamma_{k+1} = \min(S_{(\lfloor (1-\rho)N \rfloor)}, 0)$
 9:     Update $\boldsymbol{\theta}_{k+1} = \mathrm{argmax}_{\boldsymbol{\theta} \in \Theta} \sum_{i=1}^N \mathbb{1}_{S(\widetilde{\mathbf{U}}_i^{(k)}) \geq \gamma_{k+1}} w_{q_{\boldsymbol{\theta}}}(\widetilde{\mathbf{U}}_i^{(k)}) \log(q_{\boldsymbol{\theta}}(\widetilde{\mathbf{U}}_i^{(k)}))$
10:     $q_{k+1} = q_{\boldsymbol{\theta}_{k+1}}$
11:     $k \leftarrow k + 1$
12:     if $\gamma_k = 0$ then
13:         Stop $\leftarrow$ True
14:     end if
15: end while
16: Estimate $\hat{P}_{\mathrm{F}}^{\mathrm{AIS}} = \frac{1}{N} \sum_{i=1}^N \mathbb{1}_A(\widetilde{\mathbf{U}}_i^{(k)}) w_{q_k}(\widetilde{\mathbf{U}}_i^{(k)})$
17: return $\hat{P}_{\mathrm{F}}^{\mathrm{AIS}}$
```

CE-AIS with Gaussian Families. As mentioned above, in the U-space, it is natural to use a Gaussian family in parametric AIS. In CE-AIS this approach yields significant simplifications. Indeed, considering for example the class $\mathcal{Q}^{\mathrm{Gauss}}$ defined above parametrized by $\boldsymbol{\theta} = (\boldsymbol{\mu}, \boldsymbol{\Sigma})$, Rubinstein and Kroese [62] show that the optimal parameter solution of

$$\boldsymbol{\theta}_{k+1}^* = \underset{\boldsymbol{\theta} \in \Theta}{\mathrm{argmax}} = \mathbb{E}_{\widetilde{\mathbf{U}} \sim q_{\boldsymbol{\theta}}}[\mathbb{1}_{S(\widetilde{\mathbf{U}}) \geq \gamma_{k+1}} w_{q_{\boldsymbol{\theta}}}(\widetilde{\mathbf{U}}) \log(q_{\boldsymbol{\theta}}(\widetilde{\mathbf{U}}))] \tag{2.58}$$

is given by $\boldsymbol{\theta}_{k+1}^* = (\boldsymbol{\mu}_{k+1}^*, \boldsymbol{\Sigma}_{k+1}^*)$ where,

$$\boldsymbol{\mu}_{k+1}^* = \mathbb{E}_{\pi_{\gamma_{k+1}}}[\mathbf{U}] = \mathbb{E}[\mathbf{U} | S(\mathbf{U}) \geq \gamma_{k+1}], \tag{2.59}$$

$$\boldsymbol{\Sigma}_{k+1}^* = \mathbb{V}_{\pi_{\gamma_{k+1}}}[\mathbf{U}] = \mathbb{V}[\mathbf{U} | S(\mathbf{U}) \geq \gamma_{k+1}]. \tag{2.60}$$

Thus, in this case, the optimization program at line 9 in the pseudocode above, can be replaced by a simple update, obtained with self-normalized IS, $\theta_{k+1} = (\hat{\boldsymbol{\mu}}_{k+1}, \hat{\boldsymbol{\Sigma}}_{k+1})$ where,

$$\hat{\boldsymbol{\mu}}_{k+1} = \sum_{i=1}^{N} \bar{w}_{q_{\boldsymbol{\theta}_k}, A(\gamma_{k+1})}(\widetilde{\mathbf{U}}_i^{(k)}) \cdot \widetilde{\mathbf{U}}_i^{(k)}, \tag{2.61}$$

$$\hat{\boldsymbol{\Sigma}}_{k+1} = \sum_{i=1}^{N} \bar{w}_{q_{\boldsymbol{\theta}_k}, A(\gamma_{k+1})}(\widetilde{\mathbf{U}}_i^{(k)}) \cdot (\widetilde{\mathbf{U}}_i^{(k)} - \hat{\boldsymbol{\mu}}_{k+1}) \times (\widetilde{\mathbf{U}}_i^{(k)} - \hat{\boldsymbol{\mu}}_{k+1})^T \tag{2.62}$$

where $\bar{w}_{q_{\boldsymbol{\theta}_k}, A(\gamma_{k+1})}(\widetilde{\mathbf{U}}_i^{(k)})$ is the normalized weight of the i-th sample w.r.t. to level γ_{k+1}, given by,

$$\bar{w}_{q_{\boldsymbol{\theta}_k}, A(\gamma_{k+1})}(\widetilde{\mathbf{U}}_i^{(k)}) = \frac{\mathbb{1}_{A(\gamma_{k+1})}(\widetilde{\mathbf{U}}_i^{(k)}) w_{q_{\boldsymbol{\theta}_k}}(\widetilde{\mathbf{U}}_i^{(k)})}{\sum_{j=1}^{N} \mathbb{1}_{A(\gamma_{k+1})}(\widetilde{\mathbf{U}}_j^{(k)}) w_{q_{\boldsymbol{\theta}_k}}(\widetilde{\mathbf{U}}_j^{(k)})} = \frac{\mathbb{1}_{S(\widetilde{\mathbf{U}}_i^{(k)}) \geq \gamma_{k+1}} w_{q_{\boldsymbol{\theta}_k}}(\widetilde{\mathbf{U}}_i^{(k)})}{\sum_{j=1}^{N} \mathbb{1}_{S(\widetilde{\mathbf{U}}_j^{(k)}) \geq \gamma_{k+1}} w_{q_{\boldsymbol{\theta}_k}}(\widetilde{\mathbf{U}}_j^{(k)})}.$$

Improved CE. Papaioannou, Geyer, and Straub [72] propose an improvement of the Multilevel CE method we have presented, by approaching the optimal density q^* via a sequence of unnormalized densities $\tilde{q}(\cdot;\beta_1), \dots, \tilde{q}(\cdot;\beta_K)$, of the form $\tilde{q}(\cdot;\beta) \propto \psi(\cdot;\beta)\phi_d(\cdot)$, where the function ψ is such that: $\forall \mathbf{u} \in \mathbb{R}^d, \psi(\mathbf{u};\beta) \overset{\beta\to\infty}{\to} \mathbb{1}_A(\mathbf{u})$. In particular, they propose to use $\psi(\mathbf{u};\beta) = \Phi(\beta \cdot S(\mathbf{u}))$. The scheduling, i.e. the choice of the values $\beta_1, \dots, \beta_K$, and the stop criterion are based on the empirical counterpart of the coefficient of variation $\Delta_N[\cdot]$, see equation (2.31). We refer to their work and the Ph.D. thesis of El Masri [73] for more details on this method and its variants. We note, however, that this method is somewhat related to the approach we propose in our second work, presented in chapter 5, in the context of Multilevel Splitting methods, that we introduce next in section 2.4.

Advantages and Limitations. The CE method can be advantageous in settings where basic Importance Sampling methods struggle, with some caveats. In general, Multivel AIS's iterative nature allows for a progressive exploration and exploitation of the sample space, alleviating the weight degeneracy phenomenon. In parametric AIS, the initial optimization problem in the space of pdfs is replaced by an optimization problem in $\mathbb{R}^p$ where p is the number of scalar parameters associated with the chosen parametric family. However, the choice of this latter family is crucial. We saw that in the U-space, choosing Gaussian families simplifies the optimization problem. A limitation of this approach lies in the limited expressive power of these families which, for example, cannot accurately model multimodal distributions. Estimating a matrix also of $O(d(d+1)/2)$ parameters is also costly, for DNN applications one has to either use the family $Q^{\text{U-Gauss}}$ defined above or resort to a dimension reduction technique. These include the failure-informed projection method, called 'ICEred', proposed by Uribe, Papaioannou, Marzouk, *et al.* [64] and the gradient-free projection method, 'CE-m^*', proposed by El Masri, Morio, and Simatos [65].

2.3.3 Nonparametric Adaptive Importance Sampling

Nonparametric Adaptive Importance Sampling (NAIS), as presented in the works of Kim, Roh, and Lee [74] and of Morio [75], is an alternative to the parametric methods presented above, which can be particularly powerful due to its more flexible data-driven approach. However, as discussed below, this method does suffer from the curse of dimensionality and thus is not a good candidate for DNN applications.

Concept of NAIS. NAIS differs from parametric Importance Sampling by employing nonparametric methods to adaptively update the biasing distribution. This approach allows for a more flexible adjustment of the sampling distribution in response to the characteristics of the target distribution observed from empirical data, particularly in cases where parametric models are inadequate or impractical.

Kernel Density Estimation. A key element of NAIS is the use of Kernel Density Estimation (KDE), specifically the Parzen [76]-Rosenblatt [77] [3] method. KDE is used to estimate the probability density function of a dataset by averaging over a sample of kernel functions, typically Gaussian. A significant challenge in KDE is the selection of an appropriate bandwidth (window size), as a poor choice can lead to overfitting (too narrow bandwidth) or underfitting (too wide bandwidth).

NAIS in High-Dimension. While NAIS is powerful, its efficiency diminishes in high-dimensional spaces, which can be related to the inefficiency of KDE in high dimensions. In high-dimensional spaces, it becomes difficult to accurately estimate densities without a prohibitively large number of samples [78]. This problem directly impacts the performance of NAIS as soon as the problem dimension d is higher than 10. For this reason, we believe NAIS, in this form, is not adapted to applications with DNNs. That being said, a novel approach to NAIS recently proposed by Demange-Chryst, Bachoc, Morio, *et al.* [79], using deep neural networks via variational autoencoder architecture, shows promising results.

We next present the Importance Splitting method, as an alternative to IS techniques.

3. Murray Rosenblatt, the same statistician whose eponymous transform we presented in chapter 1.

2.4 Importance Splitting

This method, introduced as *Subset Simulation* (SubSim) by Au and Beck [63] in the field of reliability analysis, is a variance reduction technique that can be particularly effective for estimating probabilities of rare events of complex, high-dimensional systems.

2.4.1 Fixed-Levels Importance Splitting

Multilevel[4] Splitting (MLS) a.k.a. Importance Splitting, involves sequentially estimating the probability $\mathbb{P}[F]$ of a rare event $F \in \mathcal{F}$. The main idea, which can be traced back to Kahn and Harris [80], is to define a sequence of K nested events $\Omega = F_0 \supset F_1 \supset \ldots \supset F_K = F$, and express $\mathbb{P}(F)$ as the product:

$$\mathbb{P}(F) = \prod_{i=1}^{K} \mathbb{P}(F_i \,|\, F_{i-1}), \tag{2.63}$$

such that the subproblems of estimating the conditional event probabilities $\mathbb{P}(F_{i+1} \,|\, F_i)$ are easy enough to be solved efficiently. Applying this methodology to the random variable $\mathbf{U}$ in the U-space, the intermediary events will be defined as $F_i = [\mathbf{U} \in A_i]$. More precisely, since in our case $\mathbb{P}[F] = \mathbb{P}[\mathbf{U} \in A]$ with $A = \{\mathbf{u} \in \mathbb{R}^d : S(\mathbf{u}) = 0\}$, for a given increasing sequence of levels $L_0, L_1, \ldots, L_K$ (with $L_0 = -\infty$ and $L_K = 0$), we define the intermediary subsets $A_1, \ldots, A_K$ as

$$\forall i \in [0:K], A_i := \{\mathbf{u} \in \mathbb{R}^d \,|\, S(\mathbf{u}) \geq L_i\}. \tag{2.64}$$

Accordingly, we denote by $\pi_{L_i} = \mathcal{L}(\mathbf{U}|S(\mathbf{U}) > L_i)$ the associated distributions on $\mathbb{R}^d$.

Algorithmic Description. MLS consists of managing a set of N particles, initialized as samples from the nominal distribution π. Then, at each step k, like in a genetic algorithm, all the particles whose score is below a certain threshold are 'killed', i.e. deleted from the pool of particles. To keep an identical number of particles, each of these killed particles is replaced by a mutated copy of a particle selected at random and with equal probability amongst the surviving particles. However, unlike in a genetic algorithm, the precise statistical nature of the mutations is crucial and we discuss it in the next paragraph. Notwithstanding this process is performed until reaching the last level $L_K = 0$, which

4. Multilevel is here used with the same meaning as in 'Multilevel AIS', unrelated to Multilevel Monte Carlo, see section 2.3 for more details.

corresponds to the rare event. This algorithm is outlined in the pseudo-code 5 below. Note that, since the levels are predefined, extinctions can occur (see line 4 below), in which case one has to rerun the simulation from the start.

Algorithm 5 Subset Simulation with Fixed Levels in the U-space

```
Require: Sequence of levels $L_1, L_2, \dots, L_K$, Number of kernel iteration $T$, Proposal
    strength parameter $v$, Family of kernel $(\mathsf{Ker}_L)_{L \leq 0}$ leaving invariant the distributions
    $(\pi_L)_{L \leq 0}$
Ensure: Probability estimate of the rare event
 1: Generate initial samples $\mathbf{U}_1^{(0)}, \dots, \mathbf{U}_N^{(0)} \overset{\text{i.i.d.}}{\sim} \pi_0 = \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$
 2: for $k = 1$ to $K$ do
 3:     Determine set of surviving particles $\mathcal{I}_{\text{surv}} = \{i \in [1:N] : S(\mathbf{U}_i^{(k-1)}) \geq L_k\}$
 4:     if $\mathcal{I}_{\text{surv}}$ is $\emptyset$ then
 5:         Restart simulation.
 6:     end if
 7:     Estimate the probability $\mathbb{P}[F_k|F_{k-1}]$ as $\hat{P}_k = \frac{\text{Card}(\mathcal{I}_{\text{surv}})}{N} = \frac{N_k}{N}$
 8:     for each index $i$ in $[1:N] \setminus \mathcal{I}_{\text{surv}}$ do
 9:         Select randomly the index a of surviving particle: $J \sim \mathcal{U}(\mathcal{I}_{\text{surv}})$
10:         Sample mutated particle: $\hat{\mathbf{U}} \sim \mathsf{Ker}_{L_k}(\mathbf{U}_J^{(k-1)}, d\mathbf{u}; T, v)$
11:         Define next particle of index $i$ as the mutated particle: $\mathbf{U}_i^{(k)} = \hat{\mathbf{U}}$
12:     end for
13:     for each index $i$ in $\mathcal{I}_{\text{surv}}$ do
14:         Define next particle of index $i$ as the old particle $\mathbf{U}_i^{(k)} = \mathbf{U}_i^{(k-1)}$
15:     end for
16: end for
17: Compute the probability estimate $\hat{P}_{\text{F}}^{\text{MLS}} = \prod_{k=1}^{K} \hat{P}_k = \prod_{k=1}^{K} \frac{N_k}{N}$
```

Mutation via MCMC. Mutations are applied to each cloned particle to generate a new set of particles from the set of surviving particles only while reducing correlation among them. In the U-space, these mutation proposals can be obtained via a simple MCMC method, which is essentially the Metropolis-Hastings algorithm (see appendix A), outlined in algorithm 6. Notice that this algorithm does require a kernel leaving the nominal law $\pi = \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$ invariant. Conveniently, in the U-space, it can be realized by a simple Gaussian transition kernel $\mathsf{Ker}_\pi(\cdot, \cdot; v)$, defined by,

$$\forall \mathbf{u} \in \mathbb{R}^d, \mathsf{Ker}_\pi(\mathbf{u}, d\mathbf{u}; v) := \mathcal{N}((1-v)\mathbf{u}, \frac{v^2}{1+v^2}\mathbf{I}_d) \tag{2.65}$$

where $v \in \mathbb{R}_{++}$ is called the strength parameter.

Indeed, one readily checks that $\mathsf{Ker}_\pi(\cdot, \cdot; v)$ it satisfies the detailed balance condition for π.

Algorithm 6 $\mathsf{Ker}_L(\mathbf{u}, d\mathbf{u}; T, v)$: MCMC kernel with invariant measure π_L

```
Require: Initial values $\mathbf{u} \in \mathbb{R}^d$, Level $L$, Number of iterations $T$, Kernel leaving $\pi$ invariant $\mathsf{Ker}_\pi$, Score function $S$, Proposal strength parameter $v$.
Ensure: r.v. $\hat{\mathbf{U}}_T$ such that $\mathcal{L}(\hat{\mathbf{U}}_T) \approx \pi_L$, for $T$ large enough.
 1: Initialize $\hat{\mathbf{U}}_0 = \mathbf{u}$
 2: for $t = 1$ to $T$ do
 3:     Generate proposal $\hat{\mathbf{U}} \sim \mathsf{Ker}_\pi(\hat{\mathbf{U}}_{t-1}, d\mathbf{u}; v)$
 4:     if $S(\hat{\mathbf{U}}) > L$ then
 5:         Accept the mutation: $\hat{\mathbf{U}}_t = \hat{\mathbf{U}}$
 6:     else
 7:         Reject the mutation: $\hat{\mathbf{U}}_t = \hat{\mathbf{U}}_{t-1}$
 8:     end if
 9: end for
10: return $\hat{\mathbf{U}}_T$
```

Advantages and Limitations

This algorithm has the advantage of being an intuitive and easily implemented method. Moreover, unlike the adaptive IS presented above, it does not require specifying a parametric family of distribution or choosing a kernel function for density estimation. It does however require access to an MCMC transition kernel, and the consistency of the estimator relies on both the law of large numbers, as the number of particles N goes to infinity, and on the ergodicity of the Markov Chains, as $T \to \infty$. An important result is that the MLS estimator is unbiased, we refer the reader to the work of Amrein and Künsch [81] for a proof. For a variance analysis of this estimator, we refer to the HDR of Bourinet [37].

In high-dimension, the MCMC kernel proposed above might encounter difficulties as noted by Papaioannou, Betz, Zwirglmaier, *et al.* [82]. In some cases, this can be alleviated by changing the MCMC kernel, e.g. as proposed by Katafygiotis and Zuev [83].

2.4.2 Adaptive Multilevel Splitting

Adaptive Multilevel Splitting (AMS), is an important adaptive variant of the algorithm presented in the previous section. This algorithm was first introduced as Subset Simulation by Au and Beck [63], in the context of reliability analysis, and rediscovered, in a more general rare event simulation context, the work of Cérou and Guyader [84]. For a comprehensive overview and a historical perspective on this method, we refer the reader to the work of Cérou, Guyader, and Rousset [85]. Unlike in the fixed-level approach, AMS adaptively determines the sequence of levels during the simulation using empirical data.

Methodology

AMS employs a dynamic approach to identify intermediate levels or thresholds. In each iteration, these levels are adjusted based on empirical quantiles from the score function, see line 8. This adaptive process helps in efficiently homing in on the rare event. We outline the AMS algorithm in the pseudocode 7.

Algorithm 7 AMS in the U-space

```
Require: Quantile parameter $\rho$, Number of kernel iteration $T$, Proposal strength parameter $v$, Family of kernel $(\mathsf{Ker}_L)_{L\leq 0}$ leaving invariant the distributions $(\pi_L)_{L\leq 0}$,
Ensure: Probability estimate of the rare event $\hat{P}_{\mathrm{F}}^{\mathrm{AMS}}$
 1: Generate initial samples $\mathbf{U}_1^{(0)},\dots,\mathbf{U}_N^{(0)} \overset{\text{i.i.d.}}{\sim} \pi_0 = \mathcal{N}(\mathbf{0},\mathbf{I}_d)$
 2: $m \leftarrow \lfloor (1-\rho)N \rfloor$
 3: $k \leftarrow 0$
 4: $L_k \leftarrow -\infty$
 5: while $L_k < 0$ and $k < K$ do
 6:     Compute scores $(S_1,\dots,S_N) = (S(\hat{\mathbf{U}}_1^{(k)}),\dots,S(\hat{\mathbf{U}}_N^{(k)}))$
 7:     Compute ordered scores $S_{(1)} \leq S_{(2)} \leq \dots \leq S_{(N)}$
 8:     Compute threshold level $L_{k+1} = \min(S_{(m)},0)$
 9:     Let $\mathcal{I}_{\mathrm{surv}}$ be the indices of surviving particles
10:     for each index $i$ in $[1:N]\setminus\mathcal{I}_{\mathrm{surv}}$ do
11:         Select uniformly the index of a surviving particle: $J \sim \mathcal{U}(\mathcal{I}_{\mathrm{surv}})$
12:         Sample a mutated particle: $\hat{\mathbf{U}} \sim \mathsf{Ker}_{L_{k+1}}(\mathbf{U}_J^{(k)}, d\mathbf{u}; T, v)$    ▷ See alg.6.
13:         Define next particle of index $i$ as the mutated particle: $\mathbf{U}_i^{(k+1)} = \hat{\mathbf{U}}$
14:     end for
15:     for each index $i$ in $\mathcal{I}_{\mathrm{surv}}$ do
16:         Define next particle of index $i$ as the old particle: $\mathbf{U}_i^{(k+1)} = \mathbf{U}_i^{(k)}$
17:     end for
18:     $k \leftarrow k+1$
19: end while
20: Compute the probability estimate $\hat{P}_{\mathrm{F}}^{\mathrm{AMS}} = (1-\frac{m}{N})^{k-1} \times \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}_A(\mathbf{U}_i^{(k)})$
```

Advantages Over Fixed Levels Approach

AMS's adaptability in setting levels based on real-time simulation data makes it particularly effective in static cases where predefined thresholds might not adequately capture the dynamics leading to rare events. This method can more accurately and efficiently home in on critical regions of the state space, enhancing the precision of rare event probability estimation. One disadvantage over the fixed-levels approach, however, is that, unlike $\hat{P}_{\mathrm{F}}^{\mathrm{MLS}}$, the AMS estimator $\hat{P}_{\mathrm{F}}^{\mathrm{AMS}}$ is *not* a priori an unbiased estimator P_{F}.

2.4.3 Last Particle Algorithm

The Last Particle Algorithm is a particular case of the Adaptive Multilevel Splitting (AMS) method introduced and analyzed by Guyader, Hengartner, and Matzner-Løber [86], corresponding to an iterative elimination of the particle with the lowest score. This approach is particularly convenient for performing non-asymptotic analysis (w.r.t. the number of particles N) of convergence.

Algorithmic Principle

This algorithm distinguishes itself by concentrating on the particle with the minimum score at each iteration. This targeted approach enables efficient exploration of the state space near the critical threshold of system failure.

Pseudocode for Estimating Probability of Failure

The Last Particle Algorithm operates under the premise of the previously defined failure event $F = \mathcal{S}_{\mathrm{F}}$. We outline its procedure in the U-space in 0, though it can also be applied verbatim in the X-space.

Algorithm 8 Last Particle Algorithm for Probability of Failure Estimation

```
Require: Number of particles $N$, score function $S$, Maximum number of iteration $K$,
  Number of mutations per iteration $T$, Family of kernel $(\mathsf{Ker}_\gamma)_{\gamma \leq 0}$ leaving the invariant
  the distribution $(\pi_\gamma)_{\gamma \leq 0}$
Ensure: Probability estimate of the failure event $\mathcal{S}_{\mathrm{F}}$
  Initialize particles $(\mathbf{U}_i)_{i \in [1:N]} \overset{\text{i.i.d.}}{\sim} \pi = \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$
  $k \leftarrow 0$
  $L_k \leftarrow -\infty$
  while $L_k < 0$ and $k < K$ do
      Identify the particle $\mathbf{U}_{\min}$ with the lowest score $S(\mathbf{U}_{\min})$
      Define the next level as $L_{k+1} = S(\mathbf{U}_{\min})$
      Eliminate $\mathbf{U}_{\min}$ and replace at random it by a surviving particle $\mathbf{U}_{\text{cloned}}$
      Mutate $T$-times the clone $\mathbf{U}_{\text{clone}}$ with a kernel $\mathsf{Ker}_{L_{k+1}}$ leaving $\pi_{L_{k+1}}$ invariant
      $k \leftarrow k + 1$
  end while
  Compute Estimate $\hat{P}_{\mathrm{F}}^{\mathsf{LP}} = (1 - \frac{1}{N})^{k-1} \times \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}_A(\mathbf{U}_i^{(k)})$
```

2.5 Chapter Conclusion

This chapter presented a comprehensive study on the methodologies of Rare Event Simulation, with a particular emphasis on Importance Splitting and Importance Sampling techniques. While both methods often achieve the objective of being more sample-efficient than Crude Monte Carlo simulations, they have unique advantages and limitations in the context of simulating rare events, which can lead to choosing either one of them depending on the situation. We quickly review these characteristics in the following paragraphs.

Importance Sampling. Importance Sampling, on the other hand, excels in its versatility and applicability to a wide range of problems. The method's strength lies in its ability to reduce variance by using a change of measure, making it particularly useful in rare event simulation. Nevertheless, the technique requires careful design of the importance distribution, and improper choice can lead to inefficiency, particularly in high-dimensional spaces.

Importance Splitting. Importance Splitting is advantageous in scenarios where sequential decomposition of the rare event is feasible. Its ability to iteratively focus computational efforts on increasingly critical regions allows for efficient use of resources. However, the method's effectiveness is heavily reliant on the choice of splitting levels (scheduling), the associated score function, and the transition kernel used to perform mutations, which can be challenging to select optimally.

Comparative Analysis. Comparing both methods, Importance Splitting is often more intuitive and easier to implement, especially for dynamic problems with a clear sequential structure, which is not the topic of this thesis as we focus on static models. In contrast, Importance Sampling is generally more robust as it only relies on the law of large numbers. That being said, in complex and high-dimensional scenarios, choosing a good importance density is often difficult while adaptive methods can converge towards the optimal density, this can come at a high computational cost that may hinder efficiency in practice. In conclusion, while both Importance Splitting and Importance Sampling provide valuable tools for Rare Event Simulation, their optimal application depends on the specific characteristics and requirements of the problem at hand. While our first and second works, presented in chapters 4 and 5 respectively, focus on the importance splitting approach, our last work, presented in chapter 6 compares the efficiency of different IS and Multilevel Splitting algorithms in reliability estimations with Deep Neural Networks.

CHAPTER 3

Introduction to Deep Learning Robustness

"The best material model of a cat is another, or preferably the same cat."
– Norbert Wiener, The Role of Models in Science [87], 1945

Deep learning models have achieved remarkable success in various applications, but their robustness remains a significant concern. In this chapter, after a brief overview of deep learning essential concepts and architectures, we delve into critical aspects of deep learning robustness, exploring both adversarial and certified robustness methods.

3.1 Deep Learning in a Nutshell

Deep Learning, a subset of machine learning, leverages Deep Neural Networks (DNNs) to learn representations from complex and high-dimensional data. DNNs are characterized by their depth, which refers to the number of layers through which data is transformed. As mentioned in the introduction of the thesis, important shared characteristics of these models include:

1. **Compositional Architectures [88]:** A DNN is composed of multiple layers, where each layer can be viewed as a mathematical function. The output of one layer becomes the input of the next, forming a chain of transformations. Mathematically, a DNN with L layers can be represented as:

$$f(x) = f_L(f_{L-1}(\dots f_2(f_1(x))\dots)) = f_L \circ f_{L-1} \circ \dots \circ f_1(x)$$

   where f_i is the transformation function of the i-th layer, and x is the input data.

2. **Parallel Computing:** Modern DNNs are trained on parallel computing architectures, such as GPUs, to handle massive computations efficiently.

3. **Auto-differentiation [89]:** This process computes the gradients of the model's output w.r.t. its parameters, essential for learning. This is often achieved via backpropagation [89], which is a method of applying the chain rule of calculus to compute gradients efficiently.
4. **Stochastic Gradient Descent (SGD) [90]:** SGD and its variants are optimization algorithms used to adjust the model's parameters to minimize a loss function. The optimization is done iteratively by updating the model parameters in the direction opposite to the gradient of the loss function.

The first three characteristics of these models are of particular importance in this thesis as efficient reliability assessment methods for DNNs as they pertain to inference, whereas the last one pertains to training.

A Brief History of Artificial Neural Networks

The journey of Artificial Neural Networks (ANNs) began with the perceptron model, introduced by Mcculloch and Pitts [91] in their seminal 1943 paper. This model was an early attempt to mimic the processing of a biological neuron, as illustrated in Figure 3.1. Building upon this, Rosenblatt [92] [1] developed the Mark I Perceptron in 1957 (Figure 3.2),

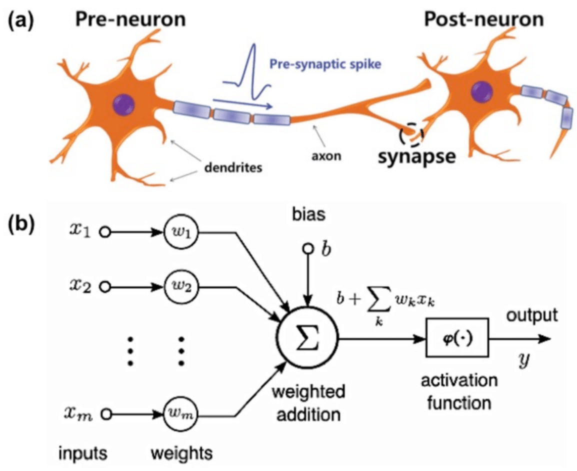


Figure 3.1 – (a) A biological neuron; (b) a Perceptron model with m inputs

a significant step towards linking computational models to biological neural processes.

1. Frank Rosenblatt, unrelated with Murray Rosenblatt, of the Rosenblatt Transform presented in chapter 1 (see section 1.2.2), though they were both born in the 1920s and PhD graduates from Cornell.

However, the growth of neural networks encountered a major roadblock with the 1969 publication of a book by Minsky and Papert [93]. Their critique revealed intrinsic limitations of perceptrons, notably their incapacity to solve non-linearly separable problems like the Exclusive OR (XOR) classification. This led to the first AI winter, a period (roughly from the 1970s to the late 1980s) characterized by reduced interest and funding in neural network research.

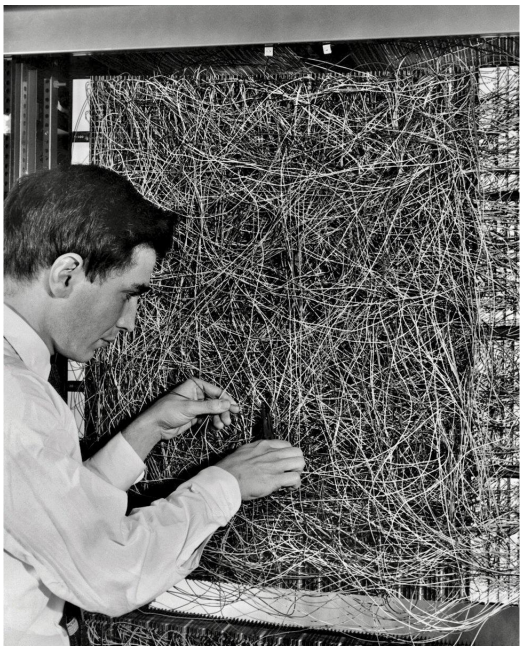


Figure 3.2 – F. Rosenblatt with the Mark I Perceptron in 1957 (source: Getty Images).

The resurgence in neural network research began in the late 1980s and continued into the 1990s, thanks to both new theoretical insights [94], [95] and computational advancements [96], [97]. The field, however, really came back to the forefront of AI research in 2012 with the success of AlexNet [98] in the CV competition 'ImageNet' [99] which marked a significant turning point. AlexNet, a convolutional neural network (see section 3.1.3), demonstrated the power of deep learning models in handling complex tasks like image recognition with unprecedented accuracy. This success spurred a renewed and intense interest in neural networks, leading to rapid advancements and the development of more sophisticated models that continue to push the boundaries of what is possible in machine learning. In 2015 another important milestone was reached as the model ResNet50, an instance of the Residual Network architecture introduced by He, Zhang, Ren, *et al.* [100], obtained it a sub-human error rate on unseen test data, as illustrated in Figure 3.3.

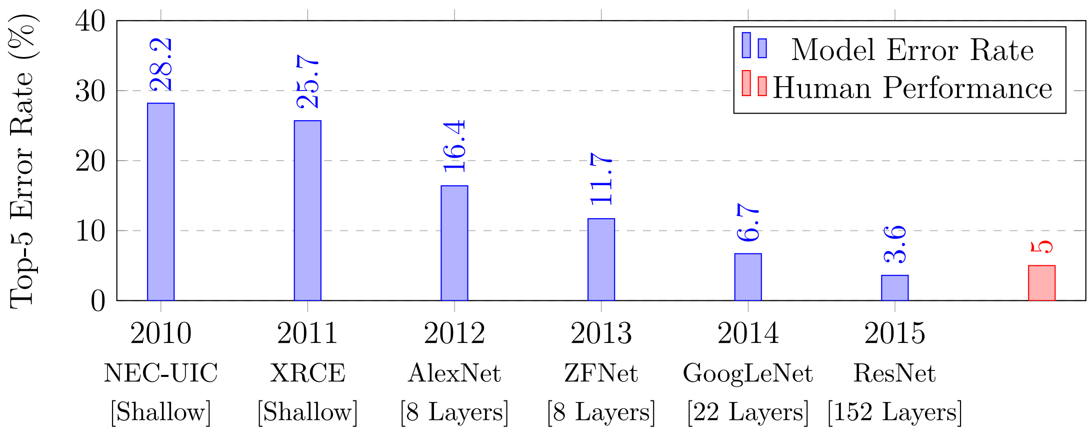


Figure 3.3 – ImageNet Competition Winners (2010-2015) and their Top-5 Error Rates.

The revival was not just limited to academic interest; it marked the beginning of a new era in machine learning, with ANNs becoming key models in various applications, from speech processing [101] to autonomous driving [102] and protein structure prediction [103].

3.1.1 The Classification Problem in Machine Learning

The classification problem is a fundamental task in machine learning where the goal is to assign labels or categories to data points based on their features. It is a case of supervised learning, where a model is trained on a dataset containing inputs and their corresponding labels. The model learns to map inputs to the correct labels from a set of training samples, hoping it will able to accurately classify unseen, but similar, data points.

Binary and Multi-Class Classification

Classification tasks can be broadly divided into two categories:

- **Binary Classification:** This involves categorizing data points into two distinct classes. The outcome is dichotomous, meaning there are only two possible classes. Examples include email spam detection (spam or not spam) and medical diagnoses (disease present or not).
- **Multi-Class Classification:** In multi-class classification, there are C classes with $C > 2$. Each data point is categorized into one of these classes. An example is image recognition, where images are classified into various categories like cars, planes, etc...

Mathematical Formulation the Problem

To perform training and inference each data point (image, email, audio recording) is encoded into a vector of features $\mathbf{x}$ in $\mathbb{R}^d$ where d is the number of features. Similarly, in binary classification, labels are mapped to a discrete set, e.g., for spam detection, one might encode the class "spam" as 1, and the class "not a spam" as 0. In the case of multi-class classification, it can be useful to use a "one-hot" encoding, whereby each class label is mapped to a vertex of the simplex $\mathcal{S}^{C-1}$ defined by

$$\mathcal{S}^{C-1} = \{\mathbf{y} = (y_1, \dots, y_C) \in [0,1]^C, \sum_{i=1}^{C} y_i = 1\}. \tag{3.1}$$

For example, in the Iris classification task [104] one might map the classes "Iris seposa", "Iris virginica" and "Iris versicolor" to the vectors $(1,0,0)$, $(0,1,0)$ and $(0,0,1)$ respectively.

Roughly speaking, the classification problem can then be formulated as follows: given a training dataset of n data points $\mathcal{D}_{\text{train}} = \{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \dots, (\mathbf{x}_n, \mathbf{y}_n)\}$ where $\mathbf{x}_i \in \mathbb{R}^d$ represents the features of the i-th data point and $\mathbf{y}_i \in \mathcal{S}^{C-1}$ the encoding of its true label, the goal is to learn a function f such that $f(\mathbf{x}) \approx \mathbf{y}$, in a certain sense to be defined.

More formally, given a loss function $\mathcal{L} : \mathbb{R}^C \times \mathcal{S}^{C-1} \to \mathbb{R}_+$ and a joint distribution $\mathcal{D}$ on $\mathbb{R}^d \times \mathcal{S}^{C-1}$ one first defines the risk $\mathcal{R}$ of a measurable function $f : \mathbb{R}^d \mapsto \mathbb{R}^C$ as:

$$\mathcal{R}(f) := \mathbb{E}_{(\mathbf{X},\mathbf{Y})\sim\mathcal{D}}[\mathcal{L}(f(\mathbf{X}), \mathbf{Y})]. \tag{3.2}$$

Given a class of hypothesis functions $\mathcal{H}$, the fundamental objective in statistical learning, as presented in the textbook of Vapnik [105], is to find a classifier f^* in $\mathcal{H}$ that minimizes the risk $\mathcal{R}$ function on $\mathcal{H}$, that is,

$$f^* \in \operatorname*{argmin}_{f\in\mathcal{H}} \mathcal{R}(f). \tag{3.3}$$

Since in practice one only has access to training data $\mathcal{D}_{\text{train}} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$, supposed to be independent realizations of the distribution $\mathcal{D}$, the common strategy is to minimize instead the empirical risk $\mathcal{R}_{\text{emp}}$, defined as,

$$\forall f \in \mathcal{H}, \mathcal{R}_{\text{emp}}(f) := \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(f(\mathbf{x}_i), \mathbf{y}_i). \tag{3.4}$$

A learning algorithm consists of an optimization method producing an (approximate) minimizer of the empirical risk in $\mathcal{H}$ denoted $\hat{f}$. The generalization gap $\Delta\mathcal{R}$ is the difference between the real risk and the empirical risk, i.e. $\Delta\mathcal{R}(f) := \mathcal{R}(f) - \mathcal{R}_{\text{emp}}(f)$. The choice of the hypothesis class is crucial to obtaining good performance. If this class is not rich enough, then it leads to under-fitting: even the best classifier in $\mathcal{H}$ obtains low performance in terms of empirical risk (see Fig. 3.4). In contrast, if $\mathcal{R}$ is too large, it leads to over-fitting, where the empirical risk $\mathcal{R}_{\text{emp}}(\hat{f})$ is near zero, but $\hat{f}$ may exhibit a high generalization gap. The usual countermeasure is to define a regularization function $M : \mathcal{H} \to \mathbb{R}_+$ that penalizes complexity and define $\hat{f}$ as a minimizer of the regularized empirical risk given by,

$$\mathcal{R}_{\text{emp}}(f; M) := \mathcal{R}_{\text{emp}}(f) + M(f).$$

The complexity of $\mathcal{H}$ is often measured by its VC (when $C = 2$) or Natarajan dimension (when $C > 2$) [106], as they relate to upper bounds on the generalization gap $\Delta\mathcal{R}(\hat{f})$ [105]. In the context of this thesis, $\mathcal{H}$ is mostly a parametric class $\mathcal{H}_{\boldsymbol{\theta}} = \{f_{\boldsymbol{\theta}} | \boldsymbol{\theta} \in \Theta\}$ of neural networks sharing the same architecture. Other common techniques used to build classifiers include Logistic Regression (LR) [107], Support Vector Machines (SVMs) [108], Decision Trees (DTs) [109], and Random Forests (RFs) [110]. For neural networks, the parameter space Θ is a subset of $\mathbb{R}^p$, where p is the total number of scalar parameters of the architecture, consisting of all the weights and biases at each layer of the neural network, see section 3.1.2 below. In deep learning, it is not unusual to have $p \gg n$, i.e. the number of parameters largely exceeds the number of training points, in which case the architecture is said to be over-parametrized. A peculiar phenomenon in deep learning is that over-parametrization, contrary to intuition drawn from statistical learning theory, can lead to better generalization [111]. Moreover, this phenomenon occurs *even* if the learned model overfits, i.e. $\mathcal{R}_{\text{emp}}(\hat{f}) \approx 0$, which is called 'benign over-fitting' [112].

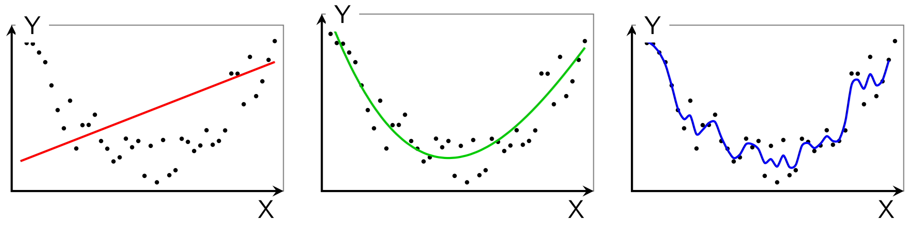


Figure 3.4 – Schematic illustration of under-fitting (on the left), of a 'good'/robust fit (in the center), and over-fitting (on the right), represented for regression for simplicity.

Loss Function for ANNs. When training ANNs for classification, the cross-entropy (CE) loss (a.k.a. the logistic loss [113]), is commonly employed [114] and given by:

$$\mathcal{L}_{\text{CE}}(f_{\boldsymbol{\theta}}(\mathbf{x}), \mathbf{y}) = -\sum_{c=1}^{C} y_c \log(f_{\boldsymbol{\theta}}(\mathbf{x})_c), \tag{3.5}$$

where $\mathbf{y}$ is the one-hot encoded true label and $\log(f_{\boldsymbol{\theta}}(\mathbf{x})_c)$ is logit for input $\mathbf{x}$ and class c.

SGD and Backpropagation. Optimization of network parameters $\boldsymbol{\theta}$ is typically performed using Stochastic Gradient Descent (SGD) [90] and its variants like RMSProp [115], and Adam [116]. SGD optimizes the loss function iteratively using a subset of the data at each step, known as a batch. This batch-based approach efficiently handles large datasets. Backpropagation is an efficient algorithm for computing the gradients of the loss function w.r.t. each parameter in the network [89]. It consists of a forward pass, where the loss is computed, followed by a backward pass, where gradients are calculated through the layers using the chain rule. These gradients are then used to update the network parameters. The update rule in SGD, considering a batch of data $\mathcal{B}_t \subset \mathcal{D}_{\text{train}}$, is given by:

$$\boldsymbol{\theta}_{t+1} \leftarrow \boldsymbol{\theta}_t - \frac{\eta}{Card(\mathcal{B}_t)} \sum_{(\mathbf{x},\mathbf{y})\in\mathcal{B}_t} \nabla_{\boldsymbol{\theta}} \mathcal{L}(f_{\boldsymbol{\theta}_t}(\mathbf{x}), \mathbf{y}) = \boldsymbol{\theta}_t - \eta \mathbf{d}_t \tag{3.6}$$

where η is the learning rate, which plays a crucial role in convergence, and $\nabla_{\boldsymbol{\theta}}\mathcal{L}$ represents the gradient of the loss w.r.t. $\boldsymbol{\theta}$. Variants of SGD often consist of adding momentum in the descent direction (i.e. taking $\widetilde{\mathbf{d}}_t = (1-\beta)\widetilde{\mathbf{d}}_{t-1} + \beta\mathbf{d}_t$) [117] and setting η adaptively.

Evaluation Metrics. The performance of a classification model is assessed using metrics such as accuracy (rate of correct prediction), precision (rate of relevant examples amongst correct predictions), and recall (rate of relevant/positive examples correctly classified). These metrics, if measured on test data $\mathcal{D}_{\text{test}} = \{(\mathbf{x}'_i, \mathbf{y}'_i)\}_{i=1}^{n}$ (independent realizations of the distribution $\mathcal{D}$), provide insights into different aspects of generalization performance.

Challenges. Despite advancements in machine learning, classification tasks present challenges like the handling of ambiguous or noisy data. Ensuring the model's ability to generalize well to unseen data is a crucial aspect of building robust classification systems. Beyond generalization, Sanyal, Dokania, Kanade, *et al.* [118] note that the 'benign-overfitting' phenomenon mentioned above, comes at the price of low robustness to input perturbations, especially adversarial perturbations, which is the topic of section 3.2.

3.1.2 Multi-Layer Perceptrons

The Multi-Layer Perceptron (MLP), a.k.a. the dense Neural Network, is a foundational architecture in deep learning. It consists of an input layer, several hidden layers, and an output layer. Each layer comprises multiple neurons, and each neuron in one layer connects to every neuron in the next layer. The mathematical representation of a neuron's activation output is given by:

$$a = \sigma\left(\sum_{i=1}^{n} w_i x_i + b\right)$$

where x_i are the neuron's inputs, w_i are the weights, b is the bias, and σ is a non-linear activation function such as the ReLU or the $\mathsf{logistic}$ function, given by

$$\mathsf{ReLU} : x \rightarrow \max(0, x) \tag{3.7}$$

$$\mathsf{logistic} : x \rightarrow (1 + e^{-x})^{-1} \tag{3.8}$$

See also Figure 3.6 for a matrix expression at the layer level. Figure 3.5 illustrates the basic structure of an MLP. As data passes through each layer, it undergoes a series of linear and non-linear transformations that enable the network to learn complex patterns from data. The penultimate layer's activations correspond to unnormalized outputs $(l_c)_{c\in[1:C]}$ called logits. The final output $\hat{\mathbf{y}}$ of the MLP is computed from logits via the $\mathsf{SoftMax}$ function, that is, $\hat{y}_c = \mathsf{SoftMax}(l_c) := \frac{e^{l_c}}{\sum_{j=1}^{C} e^{l_j}} \in [0, 1]$, so that this final output can be interpreted as a vector of probabilities. However, most neural networks are uncalibrated, meaning these probabilities do not accurately approximate the frequencies of class outcomes [119].

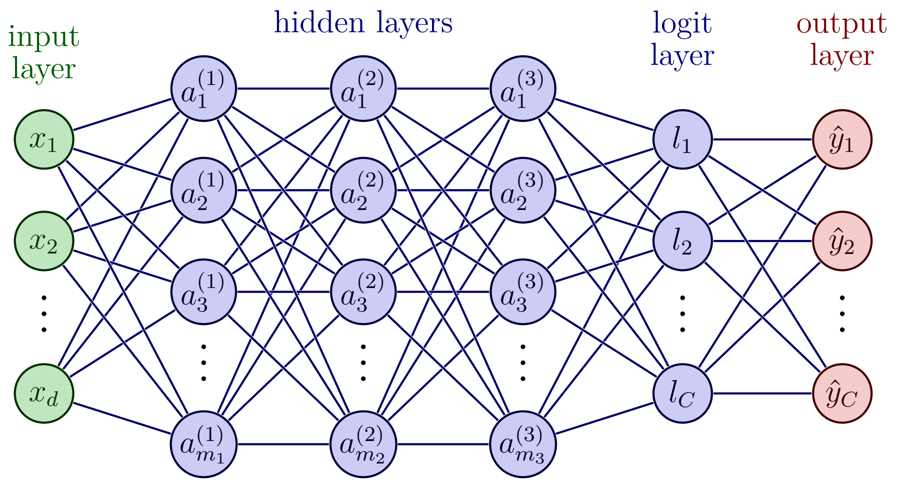
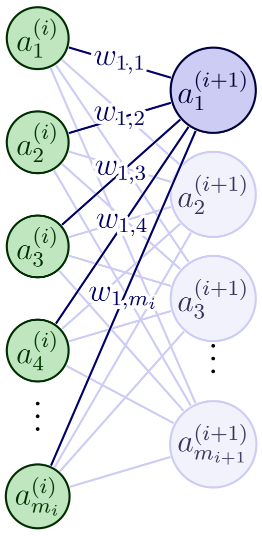


Figure 3.5 – Schematic of a Multi-Layer Perceptron with 3 hidden layers.



$$
\begin{aligned}
a_1^{(i+1)} &= \sigma\left(w_{1,1}a_1^{(i)} + w_{1,2}a_2^{(i)} + \ldots + w_{1,m_i}a_{m_i}^{(i)} + b_1^{(i)}\right) \\
&= \sigma\left(\sum_{i=1}^{m_i} w_{1,i}a_i^{(i)} + b_1^{(i)}\right)
\end{aligned}
$$

$$
\begin{pmatrix} a_1^{(i+1)} \\ a_2^{(i+1)} \\ \vdots \\ a_{m_{i+1}}^{(i+1)} \end{pmatrix} = \sigma\left[\begin{pmatrix} w_{1,1} & w_{1,2} & \ldots & w_{1,m_i} \\ w_{2,1} & w_{2,2} & \ldots & w_{2,m_i} \\ \vdots & \vdots & \ddots & \vdots \\ w_{m_{i+1},1} & w_{m_{i+1},2} & \ldots & w_{m_{i+1},m_i} \end{pmatrix}\begin{pmatrix} a_1^{(i)} \\ a_2^{(i)} \\ \vdots \\ a_{m_i}^{(i)} \end{pmatrix} + \begin{pmatrix} b_1^{(i)} \\ b_2^{(i)} \\ \vdots \\ b_{m_{i+1}}^{(i)} \end{pmatrix}\right]
$$

$$
\mathbf{a}^{(i+1)} = \sigma\left(\mathbf{W}^{(i)}\mathbf{a}^{(i)} + \mathbf{b}^{(i)}\right)
$$

Figure 3.6 – Matrix expression of neural activations at an intermediary layer.

An important result, proved by Hornik, Stinchcombe, and White [95], states that MLPs with one hidden layer, are universal approximators [2], i.e. any continuous function $f : \mathbb{R}^d \mapsto \mathbb{R}^C$ can be approximated, on a *compact* set, with arbitrary accuracy, by an MLP.

3.1.3 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) [96] are specialized for processing data with a grid-like topology, such as images. CNNs employ a mathematical operation called convolution, which is a specialized kind of linear operation, as illustrated in Figure 3.7. Convolutional networks are neural networks that use convolution in place of general matrix multiplication in at least one of their layers, but they also contain dense layers (see Figure 3.6) and MaxPool layers that sum-up features patches by their maximum, see Figure 3.8.

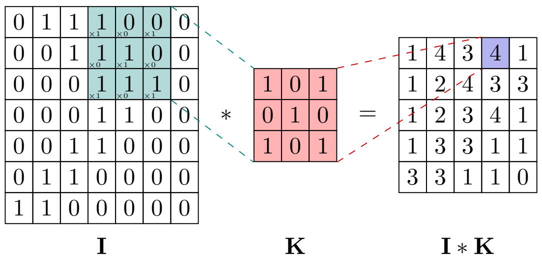


Figure 3.7 – Illustration of a 2D convolution with an input matrix **I** and a kernel **K**.

2. So are decision trees and polynomials. Thus, while this is a nice theoretical result it does *not* explain NNs success in high dimension and it does not explain the benign overfitting phenomenon.

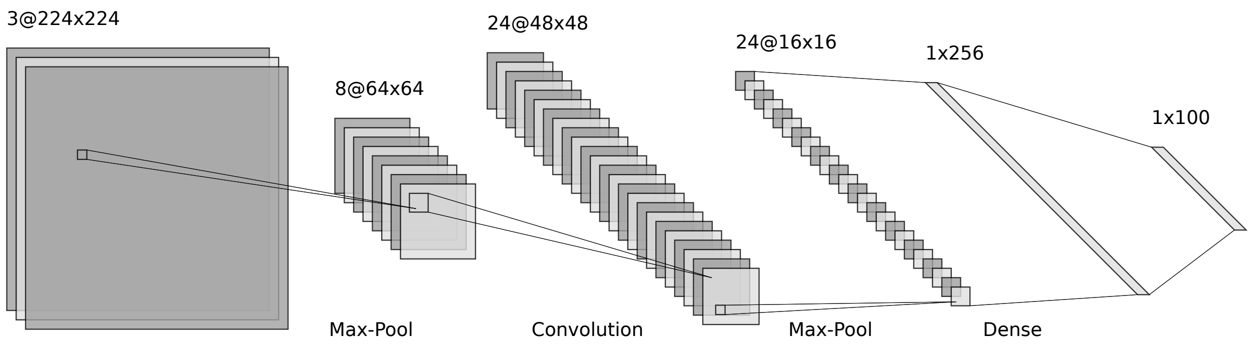


Figure 3.8 – Schematic view of a CNN taking as input a 224 × 224 pixels image and outputting a prediction vector of size 100.

3.1.4 Other Neural Network Architectures

In addition to MLPs and CNNs, several other neural network architectures cater to specific types of data and tasks. These include:

- **Recurrent Neural Networks (RNNs) [120]:** Designed for sequential data, such as time series or natural language data.
- **Long Short-Term Memory (LSTM) Networks [121]:** A type of RNN that is capable of learning long-term dependencies, commonly used in language modeling and other sequence tasks.
- **Generative Adversarial Networks (GANs) [122]:** Composed of two networks, a generator, and a discriminator, that compete against each other, often used for image generation.
- **Transformer Networks [123]:** Based on self-attention mechanisms, transformers have become the model of choice for a variety of natural language processing tasks.

Each of these architectures has unique characteristics and applications, contributing to the versatility and power of deep learning methods.

In Conclusion: Deep learning's power lies in its ability to learn intricate patterns from large volumes of data. Its diverse applications across various fields have made it a cornerstone of modern artificial intelligence research.

3.2 Adversarial Robustness of Neural Networks

Adversarial robustness is a critical facet of deep learning, aiming to study and enhance a neural network's resilience against adversarial attacks and perturbations. As discussed below, this concept can be related to reliability engineering, introduced in chapter 1.

3.2.1 Adversarial Attacks

Adversarial attacks on Deep Neural Networks (DNNs) exploit inherent vulnerabilities, leading to incorrect predictions or classifications. These attacks consist of adding imperceptible perturbations to a correctly classified input to craft an *adversarial example*, i.e. a misclassified input, as illustrated in Figure 3.9. After exploring the concepts of white-box and black-box attacks, we introduce the mathematical framework of adversarial robustness and discuss two common adversarial attacks: the Fast Gradient Sign Method (FGSM) and Projected Gradient Descent (PGD). Finally, we draw a parallel between adversarial examples and the concept of MPFP, previously introduced in chapter 1 (see section 1.3.1).

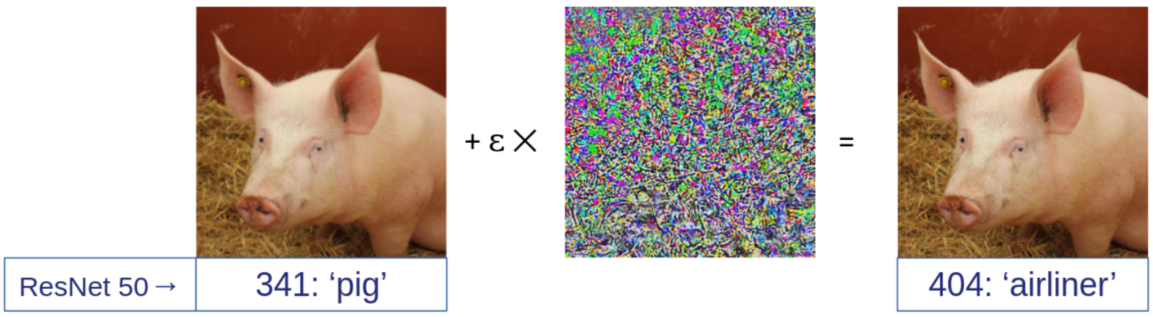


Figure 3.9 – An adversarial example crafted for a ResNet50 model trained on ImageNet. The imperceptibly modified image of a pig (on the right) is classified as an airliner.

White-Box vs. Black-Box Attacks

Adversarial attacks are categorized based on the attacker's knowledge of the model:

- **White-Box Attacks**: In white-box settings, attackers have complete knowledge of the model, including its architecture, parameters, and training data. This allows for crafting more effective adversarial examples using gradient-based methods [124].
- **Black-Box Attacks**: Contrarily, black-box attacks occur when attackers have no access to the model's internals. Attackers typically use trial-and-error methods or train surrogate models to approximate the target model's behavior [125].

While white-box attacks are more powerful by design, efficient black-box attacks have been designed [126] and both attack types pose significant threats to DNNs security.

Mathematical Formulation of Adversarial Attacks

Adversarial attacks on DNNs are formulated as optimization problems where the objective is to find minimal perturbations that lead to misclassification. These perturbations are often constrained by different norms to ensure they remain imperceptible or within a realistic bound.

ℓ_p Norms for Adversarial Attacks

Before defining the optimization problem, we first introduce various norms used to quantify the size of perturbations:

- ℓ_2**-norm**: Measures the Euclidean distance between two points. In the context of adversarial attacks, it represents the root-mean-square of the perturbation and is defined as $\|\boldsymbol{\delta}\|_2 = \sqrt{\sum_{i=1}^{d} \delta_i^2}$. It is useful for capturing the overall energy of the perturbation.
- ℓ_∞**-norm**: Measures the maximum change to any component of a vector. For a vector in $\mathbb{R}^d$, it given by $\|\boldsymbol{\delta}\|_\infty = \max_{i\in[1:d]} |\delta_i|$ for adversarial perturbations. This norm is commonly used as it limits the maximum perturbation at any pixel in an image, maintaining visual similarity.
- ℓ_1**-norm**: Represents the sum of absolute values of the vector components and is defined as $\|\boldsymbol{\delta}\|_1 = \sum_i |\delta_i|$. The ℓ_1 norm is less common in adversarial settings but can be useful for inducing sparse perturbations.
- ℓ_0**-"norm"**: Represents the number values of the vector which are non-zero and is defined as $\|\boldsymbol{\delta}\|_0 = \sum_i |\delta_i|^0$. While it is not technically a norm (it is not homogeneous), it can still be useful to create even sparser perturbations than the ℓ_1 norm. For this reason, ℓ_0 attacks are also called, in computer vision, few-pixel attacks.

Choosing a particular norm to model adversarial perturbation is an open problem in general and will depend on the data type and threat model. The ℓ_2 and ℓ_∞ norms are the most common choices and are generally easier to work with. However, works on sparse adversarial attacks often focus either on ℓ_0 [127] or ℓ_1 [128] attacks.

Finding Adversarial Examples

The problem of finding an adversarial example given a distortion budget ε can be formulated as:

$$\textit{Find}\, \boldsymbol{\delta} \in \mathbb{R}^d \text{ s.t. } \mathsf{cl}(f(\mathbf{x}_0+\boldsymbol{\delta})) \neq \mathsf{cl}(f(\mathbf{x}_0)) \quad \text{subject to} \quad \|\boldsymbol{\delta}\|_p \leq \varepsilon, \quad \mathbf{x}_0+\boldsymbol{\delta} \in [0,1]^d \quad \text{(P1)}$$

where f represents the DNN, $\mathbf{x}$ is the original input, $\boldsymbol{\delta}$ is the adversarial perturbation, $\|\cdot\|_p$ refers to the norm (ℓ_2, ℓ_∞, ℓ_1 or ℓ_0), ε is the distortion budget pertaining to this norm and $\mathsf{cl} : \mathbb{R}^C \to [1:C]$ is class the operator defined as $\mathsf{cl}(\mathbf{y}) = \operatorname{argmax}_{i\in[1:C]} y_i$. Additionally, it is useful to define an adversarial loss $\mathcal{L}_{\mathsf{adv}}$ as a surrogate for the condition "$\mathsf{cl}(f(\mathbf{x}_0+\boldsymbol{\delta})) \neq \mathsf{cl}(f(\mathbf{x}_0))$". Though different choices are possible, see [129], the most common choice is given by:

$$\mathcal{L}_{\mathsf{adv}}(f(\mathbf{x}), y) := f(\mathbf{x})_y - \max_{c\in[1:C]\setminus\{y\}} f(\mathbf{x})_c$$

where $y = \mathsf{cl} f(\mathbf{x}_0)$ is the true label of $\mathbf{x}_0$.

Finding Minimal Norm Adversarial Attacks

A related, but more ambitious objective is that of finding an adversarial attack with *minimum* distortion norm, which can be expressed as:

$$\min_{\boldsymbol{\delta}} \|\boldsymbol{\delta}\|_p \quad \text{subject to} \quad \mathsf{cl}(f(\mathbf{x}_0+\boldsymbol{\delta})) \neq \mathsf{cl}(f(\mathbf{x}_0)), \quad \mathbf{x}_0+\boldsymbol{\delta} \in [0,1]^d. \qquad \text{(P2)}$$

The choice of ℓ_p-norm affects the nature of the perturbation and is selected based on the desired properties of the adversarial examples (sparsity, imperceptibility, etc...).

We next introduce simple, yet commonly used, adversarial attacks to solve P1.

Fast Gradient Sign Method (FGSM)

The FGSM, introduced by Goodfellow et al. [124], generates adversarial examples by adding a small, gradient-sign-based perturbation to the input:

$$\mathbf{x}_{\text{FGSM}} = \mathbf{x}_0 + \boldsymbol{\delta}_{\text{FGSM}} = \mathbf{x}_0 + \varepsilon \cdot \operatorname{sign}(\nabla_{\mathbf{x}} \mathcal{L}_{\mathsf{adv}}(f(\mathbf{x}_0), y))$$

where ε controls the maximum perturbation magnitude, and $\nabla_x \mathcal{L}_{\mathsf{adv}}$ denotes the gradient of the loss function $\mathcal{L}_{\mathsf{adv}}$ w.r.t. to input $\mathbf{x}$. FGSM is an efficient method that does not require fine-tuning parameters and produces an adversarial perturbation $\boldsymbol{\delta}_{\text{FGSM}}$ in the ℓ_∞ sense, as one can readily check that $\|\mathbf{x}^{\text{adv}}_{\text{FGSM}} - \mathbf{x}_0\|_\infty = \|\boldsymbol{\delta}_{\text{FGSM}}\|_\infty \leq \varepsilon$.

Projected Gradient Descent (PGD)

A more powerful attack, Projected Gradient Descent (PGD) [130], iteratively applies small adversarial loss descent steps, while ensuring the adversarial example remains in the feasible set $\mathcal{S} = \mathcal{B}_p(\mathbf{x}_0, \varepsilon) := \{\mathbf{x} \in [0,1]^d : \|\mathbf{x} - \mathbf{x}_0\|_p \leq \varepsilon\}$, using a projection on that space, that is:

$$\forall t, \mathbf{x}_{t+1} = \text{Proj}_{\mathcal{S}}\left(\mathbf{x}_t + \alpha \cdot \nabla_{\mathbf{x}} \mathcal{L}_{\mathsf{adv}}(f(\mathbf{x}_t), y)\right).$$

After a number T of iterations, the final adversarial example is defined as $\mathbf{x}^{\text{adv}}_{\text{PGD}} = \mathbf{x}_T$. Note that, by construction, $\mathbf{x}^{\text{adv}}_{\text{PGD}}$ is in the set $\mathcal{S}$. As in all descent algorithms, the choice of α is crucial for convergence. This attack serves as the basis for the more refined FMNA [131] attack, designed to solve problem P2, and that we use extensively in chapter 6.

Connections in Adversarial Attacks and MPFP Search

The formulation of finding minimal norm adversarial attacks in deep neural networks and identifying the Maximal Probability Failure Point (MPFP) in statistical reliability engineering showcases an interesting parallel. Both problems involve optimization techniques but are applied in different contexts with distinct objectives.

Optimization Formulations

- **Minimal Norm Adversarial Attack**: The objective is to find the smallest perturbation that leads to misclassification for a given neural network. Mathematically, it be rewritten as:

$$\mathbf{x}^* = \underset{\mathbf{x} \in [0,1]^d}{\text{argmin}} \|\mathbf{x} - \mathbf{x}_0\|_p \quad \text{subject to} \quad \mathcal{L}_{\mathsf{adv}}(f(\mathbf{x}), y) \leq 0$$

  This problem focuses on minimizing an ℓ_p-norm under the constraint of achieving misclassification.

- **Maximal Probability Failure Point (MPP or MPFP)**: In reliability engineering, the MPP or MPFP, also known as the Design Point, is defined as the point in the transformed standard normal space where the system's failure is most likely.

Mathematically, it is located by maximizing the joint probability density function ϕ_d subject to the performance function $G(\mathbf{u}) = 0$. The MPFP $\mathbf{u}^*$ can thus expressed as:

$$\mathbf{u}^* = \operatorname*{argmin}_{\mathbf{u}\in\mathbb{R}^d} \|\mathbf{u}\|_2 \quad \text{subject to} \quad G(\mathbf{u}) \leq 0$$

This point has the smallest ℓ_2-norm amongst failure points.

Discussion of Unexplored Links

The optimization formulations in adversarial robustness in deep learning and MPFP identification in reliability engineering share conceptual similarities. The integration of methodologies from one domain could potentially inform and enhance the other. Well-established techniques from reliability engineering, particularly those considering input noise distributions, may offer new perspectives for developing robust DNNs against adversarial attacks. Conversely, adversarial attack strategies could provide innovative approaches for probabilistic failure analysis in complex systems. To the best of our knowledge, this interconnection remains largely unexplored in the literature, offering a promising avenue for future research, which we explore in detail in chapter 6.

3.2.2 Adversarial Defenses

Adversarial defenses are strategies employed to make deep neural networks more robust against adversarial attacks. These defenses aim to either prevent successful attacks or mitigate their impact. This section discusses various defense mechanisms and their underlying principles.

Types of Adversarial Defenses

- **Detection [132]**: These defenses use statistical properties of adversarial data to detect it at inference time. They rely on the hypothesis that adversarial data and correct data are drawn from different probability distributions and use empirical methods, such as the Mean Maximum Discrepancy, to perform detection.
- **Denoising [133]**: These defenses modify the input data before the neural network processes it. Techniques include image smoothing, cropping, compression, and Deep learning-based denoising techniques. Input transformation aims to remove adversarial perturbations while retaining the original content of the input.

- **Adversarial Training (Model Hardening) [134]**: This approach involves training the neural network to be inherently more robust against adversarial attacks. Techniques include adversarial training, where the training data set $\mathcal{D}_{\text{train}}$ is augmented with adversarial examples, random training where it's augmented with randomly corrupted inputs and finally regularization methods that penalize sensitivity to small perturbations, e.g. via control of its Lipschitz constant [135], [136].
- **Certified Defenses [137]**: Certified defenses provide theoretical robustness guarantees within certain constraints. One common approach is interval-bound propagation, which computes bounds on the output of each layer, given bounded input perturbations [138]. Another approach is the use of abstract interpretation techniques [139] to over-approximate the set of possible activations in response to input variations. See section 3.3 for more details on certified bounds.

Challenges and Trade-offs

While adversarial defenses increase the robustness of neural networks, they often come with trade-offs. For instance, adversarial training can lead to a decrease in the accuracy of DNNs on clean data. Similarly, input transformations may degrade the quality of the input, potentially impacting the model's performance. This seemingly unavoidable trade-off between accuracy on clean data has always been studied theoretically under the name of "No-free-Lunch Theorem" [140]. In practice, it is crucial to balance robustness with other performance metrics including accuracy and computational efficiency.

Recent Advances

Recent research has focused on developing more effective and efficient adversarial defenses. This includes work on adaptive defenses that dynamically adjust based on the detected threat level and research into understanding the theoretical limits of adversarial robustness [141]. Hybrid approaches combining adversarial training and certified defense strategies are also being explored for enhanced robustness [142].

Adversarial defenses remain an active area of research, with ongoing efforts to improve their effectiveness and to understand the fundamental principles governing adversarial robustness in deep neural networks.

3.3 Certified Robustness of Neural Networks

Certified robustness in the context of neural networks is about developing and applying methodologies that can provide guarantees on a network's ability to withstand adversarial attacks within certain bounds. This section delves into the concepts, methods, and challenges associated with certified robustness in deep learning.

3.3.1 Understanding Certified Robustness

Certified robustness refers to the ability of a neural network to consistently classify inputs correctly within a specified range of perturbations. Unlike empirical robustness, presented above, certified robustness provides theoretical assurances, ensuring that the network's predictions remain unchanged for perturbations below a certain magnitude.

More generally, formal verification of NNs [143] involves proving input-output properties: given an input set $\mathcal{I} \subset \mathbb{R}^d$, an output set $\mathcal{O} \subset \mathbb{R}^C$, and a neural classifier f, the goal is to *prove* that the image of $\mathcal{I}$ under f is included in $\mathcal{O}$, i.e. that $f(\mathcal{I}) \subset \mathcal{O}$ or equivalently:

$$\forall \mathbf{x} \in \mathbb{R}^d, \quad \mathbf{x} \in \mathcal{I} \Longrightarrow f(\mathbf{x}) \in \mathcal{O}. \qquad (\mathcal{P}(\mathcal{I}, \mathcal{O}, f))$$

A formal verification algorithm (FVA) is a formal procedure that can, ideally, either prove this property whenever it is true or disprove it otherwise, e.g. by providing an example of property violation. This problem is further illustrated in Figures 3.10 and 3.11.

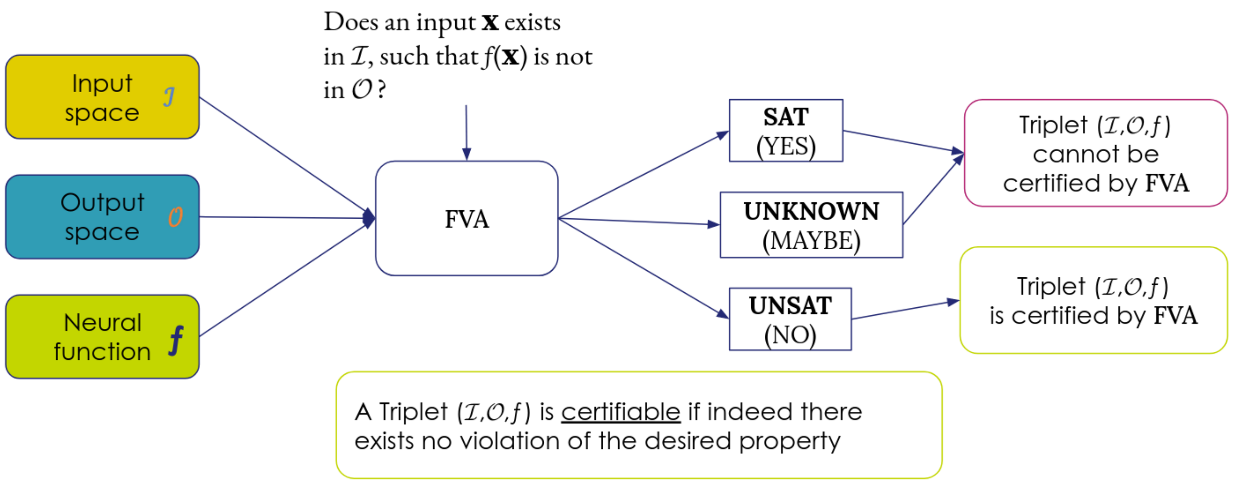

Input space $\mathcal{I}$
Output space $\mathcal{O}$
Neural function f
Does an input $\mathbf{x}$ exists in $\mathcal{I}$, such that $f(\mathbf{x})$ is not in $\mathcal{O}$?
FVA
SAT (YES)
UNKNOWN (MAYBE)
UNSAT (NO)
Triplet $(\mathcal{I},\mathcal{O},f)$ cannot be certified by FVA
Triplet $(\mathcal{I},\mathcal{O},f)$ is certified by FVA
A Triplet $(\mathcal{I},\mathcal{O},f)$ is certifiable if indeed there exists no violation of the desired property

Figure 3.10 – Schematic Illustration of formal verification algorithms.

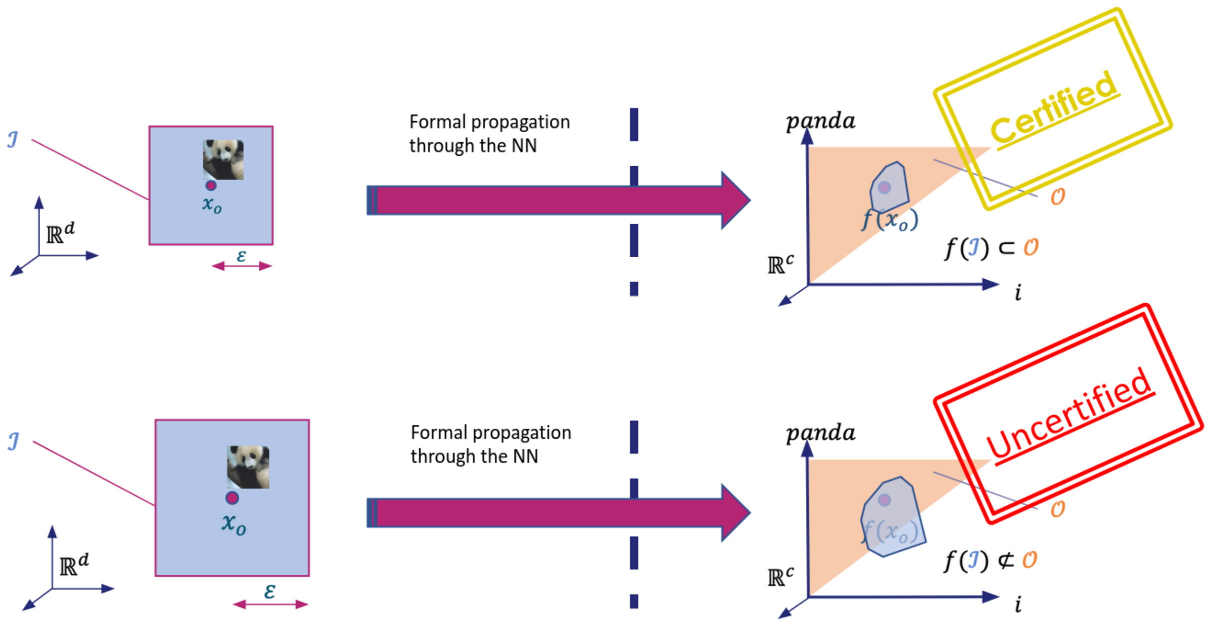

Formal propagation through the NN
panda
Certified
Uncertified
$\mathcal{I}$
$\mathbb{R}^d$
x_o
ε
$f(x_o)$
$\mathcal{O}$
$f(\mathcal{I}) \subset \mathcal{O}$
$f(\mathcal{I}) \not\subset \mathcal{O}$
$\mathbb{R}^c$
i

Figure 3.11 – Formal verification algorithm for Neural Networks robustness.

We now introduce the vocabulary used to characterize different FVAs:

- **Local vs. Global**: A FVA is *local* if it considers input sets $\mathcal{I}$ defined locally around an input $\mathbf{x}_0$. In local adversarial robustness for example, $\mathcal{I}$ is typically an ℓ_p-norm ball around a given input $\mathbf{x}_0$, i.e. $\mathcal{I} = \mathcal{B}_p(\mathbf{x}_0, \varepsilon) = \{\mathbf{x} \in \mathcal{X}, \|\mathbf{x} - \mathbf{x}_0\|_p \leq \varepsilon\}$. Most algorithms considered here are local, as global certification is a much more ambitious objective. However, a recent work of Levy, Yerushalmi, and Katz [144] aims to tackle this problem by aggregating local bounds of robustness using statistical methods.

- **Sound vs. Unsound**: This is perhaps the most important characteristic. A FVA is sound if it *only* certifies triplets $(\mathcal{I}, \mathcal{O}, f)$ such that the property $\mathcal{P}(\mathcal{I}, \mathcal{O}, f)$ is true. In contrast, it is *unsound* if there exists a triplet $(\mathcal{I}_1, \mathcal{O}_1, f_1)$ that would be certified by the FVA even though there exists at least one violation of the desired property: i.e. there exists $\mathbf{x} \in \mathcal{I}_1$ such that $f_1(\mathbf{x}_1) \notin \mathcal{O}_1$.

- **Complete vs. Incomplete**: A FVA is complete if it always certifies a triplet $(\mathcal{I}, \mathcal{O}, f)$, whenever the property $\mathcal{P}(\mathcal{I}, \mathcal{O}, f)$ is true. On the contrary it is *incomplete* if there exists at least one triplet $(\mathcal{I}, \mathcal{O}, f)$ such that $\mathcal{P}(\mathcal{I}, \mathcal{O}, f)$ is true, that cannot be certified by the algorithm (i.e. it can neither prove nor disprove the property).

A fourth, less formal, characteristic is that of scalability: an FVA satisfies this property if its computational cost grows reasonably w.r.t. with the size of the problem.

The certification methods of local adversarial robustness generally belong to one of the following classes of algorithms:

- **Complete Verification Methods**: Also called, *exact* verification methods provide formal guarantees of robustness by exhaustively analyzing all possible perturbations within a given range. Katz, Barrett, Dill, *et al.* [143] proposed the first exact verification method for Neural Networks, using tools from Satisfiability Modulo Theories (SMT). Notably, they prove in the same work that complete verification of Neural Networks is an NP-complete problem. See section 3.3.2 for more details.
- **Incomplete Verification Methods**: These methods use conservative approximations to provide robustness guarantees. They offer computationally efficient, but incomplete, verification algorithms. They essentially work by relaxing the problem of exact verification using different set approximations. Interval bound propagation and abstract interpretation are prominent examples [139]. See section 3.3.3.
- **Probabilistic Verification**: This approach combines probabilistic models and formal methods to obtain certified bounds on the likelihood of adversarial examples under a certain noise distribution. Weng, Chen, Nguyen, *et al.* [145] use incomplete certification methods to get local upper and lower *linear* bounds on neural networks around a given input $\mathbf{x}_0$, denoted respectively $f^{(\text{up})}$ and $f^{(\text{low})}$. These functional bounds are combined with a probabilistic method to produce bounds $[P_{\text{F}}^{(low)}(\mathbf{x}_0), P_{\text{F}}^{(up)}(\mathbf{x}_0)]$ on the probability of errors $P_{\text{F}}(\mathbf{x}_0)$. The main problem with this approach is that the linear upper and lower bounds are generally very *loose*. As show multiple experiments reported in Webb, Rainforth, Teh, *et al.* [146], these methods generally output vacuous bounds, i.e. $[P_{\text{F}}^{(low)}, P_{\text{F}}^{(up)}] \approx [0, 1]$.

3.3.2 Complete Formal Verification of Neural Networks

Complete formal verification aims to provide absolute, mathematical guarantees about the behavior of a neural network, within a specified input range. This process involves rigorously proving that, under specified conditions, the network will always behave as expected, without any possibility of adversarial exploitation.

Approaches and Techniques. Approaches like Satisfiability Modulo Theories (SMT) and Mixed Integer Linear Programming (MILP) are commonly used in complete formal verification [143]. These techniques exhaustively analyze the network's response to all possible inputs within a given perturbation set, ensuring that no adversarial examples can lead to incorrect classifications.

Limitations and Challenges. While offering the strongest guarantees, complete formal verification is computationally expensive and often impractical for large-scale or deep neural networks. The complexity of these methods scales exponentially with the size of the network, making them suitable primarily for smaller, simpler models.

3.3.3 Incomplete Formal Verification of Neural Networks

Incomplete formal verification methods provide robustness guarantees without needing to analyze every possible input. These methods are more scalable than complete verification techniques but offer weaker assurances.

Approximation-based Methods. Incomplete verification often involves approximation techniques, such as abstract interpretation or zonotope-based methods, which estimate the network's behavior under perturbations [139]. These methods balance between computational efficiency and the precision of robustness guarantees.

Trade-offs and Applicability. The primary trade-off with incomplete formal verification is between computational tractability and the strength of the robustness guarantee. While these methods can be applied to larger networks, the guarantees they provide are less strong compared to complete verification.

3.3.4 Randomized Smoothing for Neural Network Robustness

Randomized smoothing was introduced by Cohen, Rosenfeld, and Kolter [147] as a probabilistic approach to adversarial robustness certification. It works by averaging the outputs of a neural network across multiple perturbations of the input, leading to smoother and more stable predictions.

Mechanism and Implementation. Randomized smoothing involves producing N perturbed versions of an input by adding Gaussian noise and then aggregating the network's predictions on these perturbed samples, by a vote mechanism, to determine the final output. This process effectively smooths the decision boundary, making it harder for adversarial perturbations to cause misclassification. The main point of the original paper [147] is that this gain of robustness can be *certified*, by using statistical properties of the Gaussian law.

Advantages and Limitations. The primary advantage of randomized smoothing is its simplicity and applicability to any neural network architecture. However, the method relies on *asymptotic* probabilistic guarantees and may not be suitable for applications requiring absolute robustness assurances. Moreover, it introduces a new, random classifier $\tilde{f}_\theta$, whose computational cost is N times larger than that of the initial neural network f_θ.

3.3.5 Challenges in Certified Robustness

Achieving certified robustness is not without challenges. The computational complexity of exact verification methods often limits their applicability to smaller networks. Conservative approximation methods, while scalable, may provide overly cautious estimates, impacting the model's usability. Probabilistic verification methods deal with uncertainties and require careful statistical interpretation. Recent research in certified robustness focuses on balancing scalability and precision. Efforts include developing hybrid methods that combine exact and conservative approaches and enhancing probabilistic models for better accuracy and efficiency [148]. The field is also exploring the use of certified robustness in real-world applications, such as autonomous vehicles and healthcare systems, where safety and reliability are paramount [149].

3.4 Statistical Reliability of Neural Networks

3.4.1 Motivation and Problem Statement

The assessment of statistical reliability in neural networks primarily focuses on their robustness against input perturbations and uncertainties. This involves understanding how neural networks behave under varied conditions and quantifying their propensity towards failure or incorrect outputs.

Challenges and Importance in Safety-Critical Applications

In safety-critical applications, such as autonomous driving, medical diagnostics, and aerospace engineering, the reliability of neural networks is not merely a theoretical concern but a practical necessity. The challenges in these domains stem from several key factors:

- **Complexity of Neural Network Models:** Deep neural networks, with their intricate architectures and large number of parameters, exhibit complex behaviors that are not fully understood, making reliability assessment challenging.
- **Uncertainty and Variability of Inputs:** Safety-critical systems often operate in dynamic environments with high variability and uncertainty in inputs, which can significantly impact the performance and reliability of neural networks.
- **Regulatory and Ethical Considerations:** The deployment of neural networks in safety-critical applications raises regulatory and ethical questions, necessitating robust reliability assessments to ensure compliance and public trust.

These challenges highlight the importance of developing advanced methods for assessing and enhancing the robustness and reliability of neural networks. This thesis aims to address these issues by exploring novel approaches in statistical reliability engineering, particularly focusing on the application of rare event simulation techniques to quantify the robustness of deep neural networks against various types of input perturbations. The goal is not only to advance the theoretical understanding of neural network reliability but also to provide practical tools and methodologies that can be applied in real-world safety-critical systems.

Defining the Limit State Function in Neural Networks

For neural networks, the limit state function g is defined in the context of the network's output and classification behavior. This function quantifies the deviation from correct classification and is pivotal for assessing the reliability of the network.

- **Limit State Function** g: In the context of neural network robustness, the limit state function $g(\mathbf{x})$ is defined as:

  $$g(\mathbf{x}) := f(\mathbf{x})_{y_0} - \max_{k \neq y_0} f(\mathbf{x})_k$$

  where $f(\mathbf{x})$ denotes the logits output of the neural network for input $\mathbf{x}$, y_0 is the true label of the original input $\mathbf{x}_0$. The function g represents the difference between the network's confidence in the correct class and the highest confidence in any incorrect class. Thus it is positive if the neural network classification of $\mathbf{x}$ is correct, and reversely if it's non-negative then $\mathbf{x}$ is misclassified. We note however that this is one possibility out of many options. For example, for any strictly increasing function $\varphi : \mathbb{R} \to \mathbb{R}$ such that $\varphi(0) = 0$, the function $\varphi \circ g$ is a also possible choice of LSF.

- **Transformed Limit State Function** G: The transformed limit state function $G(\mathbf{u})$ in the standard normal space (U-space) is derived from g through an isoprobabilistic transformation. It is defined as:

  $$G(\mathbf{u}) = g(\mathcal{T}^{-1}(\mathbf{u}))$$

  where $\mathcal{T}$ represents the transformation from the original input space to the standard normal space. This function plays a crucial role in applying statistical reliability methods to neural networks. It can obtained as one of the isoprobabilistic transforms presented in section 1.2.2 or as the inverse of a normalizing flow, see section A.4.

The primary aim of statistical reliability assessment in neural networks is to evaluate the probability of failure under uncertain input conditions. This includes scenarios where the network's output deviates from expected behavior due to various input perturbations. The failure probability in the original input space and in the standard normal space is given by:

$$P_{\mathrm{F}} = \mathbb{P}[g(\mathbf{X}) \leq 0] = \int_{\{\mathbf{x}: g(\mathbf{x}) \leq 0\}} p(\mathbf{x})\, d\mathbf{x} \tag{3.9}$$

$$P_{\mathrm{F}} = \mathbb{P}[G(\mathbf{U}) \leq 0] = \int_{\{\mathbf{u}: G(\mathbf{u}) \leq 0\}} \phi_d(\mathbf{u})\, d\mathbf{u}$$

where $p_{\mathbf{X}}$ and ϕ_d denote the pdfs in the original and transformed spaces, respectively.

3.4.2 Prior State-of-the-Art

The growing integration of NNs in critical applications such as autonomous driving and healthcare makes the assessment of their statistical reliability essential. Traditional performance metrics fall short of capturing the probabilistic nature of failures, necessitating the development of robust methodologies that are universally applicable across various NN architectures and domains. Recent advancements have begun addressing these challenges.

Webb, Rainforth, Teh, *et al.* [146] first introduced the problem of estimating the statistical robustness of Neural Networks, i.e. the probability P_{F}, as defined in equation (3.9). Moreover, they proposed to do so by using the Adaptive Multiple Splitting (AMS) method, presented in section 2.4.2. Their approach however comes with no theoretical guarantees and we show in addition that their implementation is inefficient, in chapter 5.

Baluta, Chua, Meel, *et al.* [148], in contrast, use crude Monte Carlo simulations though within the sequential hypothesis testing methodology of Wald [150]. In this approach, the number of samples used is gradually increased to match the difficulty of the estimation problem. This is advantageous, as it does not use too much computing power on easy instances, i.e. when the probability of failure is small. This approach is sound and comes with strong non-asymptotic guarantees. However, it can only be as efficient as CMC and thus is not efficient on hard instances where the probability of failure P_{F} is very small, as we saw in section 2.1.2 of the previous chapter.

In a more recent work Bai, Huang, Lam, *et al.* [151] introduced a novel approach using formal verification methods for neural networks (see section 3.3), combined with Importance Sampling (see section 2.2.2), to obtain strong statistical guarantees. However, as we saw above, exact verification of neural network outputs is an NP-complete problem and the solvers currently used do not scale to large Neural Networks. Indeed, they only applied their method to Neural Networks with 200 hidden neurons at most.

Therefore, significant gaps remain in developing comprehensive methodologies for the statistical reliability assessment of neural networks. This thesis aims to bridge these gaps by introducing novel statistical methodologies and frameworks for NN reliability assessment, contributing to the robust and reliable deployment of NNs in real-world applications. The necessity of statistical reliability assessment in neural networks transcends theoretical importance, especially in critical sectors. In this section, we laid the groundwork for exploring advanced solutions to these challenges in the subsequent chapters.

3.5 Chapter Conclusion

In this chapter, we have explored the critical aspects of deep learning robustness, encompassing adversarial and certified robustness. These concepts and methods lay the foundation for the subsequent chapters in this thesis, which extend the principles of statistical reliability engineering to assess the robustness of neural networks in the presence of rare events.

The scientific contributions of this thesis, presented in Chapters 4, 5, and 6, bridge the gap between traditional statistical reliability engineering and the unique challenges posed by neural network classifiers. Our contributions address the need for efficient and reliable statistical assessment of neural network corruption robustness, emphasizing the importance of estimating reliability in scenarios involving rare events.

Chapter 4, titled "Efficient Statistical Assessment of Neural Network Corruption Robustness," is the first step in this journey. It builds on the foundation established in this chapter, introducing efficient methods for assessing the corruption robustness of neural networks. We will delve into the details of these methods and their contributions to the broader field of deep learning reliability.

Part II

Contributions

CHAPTER 4

Efficient Statistical Assessment of Neural Network Corruption Robustness

4.1 Introduction

Despite state-of-the-art performances on many Computer Vision and NLP tasks, Deep Neural Networks (DNNs) have been shown to be sensitive to both adversarial and random perturbations [7], [8]. Concerns about their safety and reliability have come forth as their applications move to critical fields, such as the defense sector or self-driving vehicles.

Certification A posteriori certification aims at verifying the correct behavior of a trained network $f : \mathbb{R}^d \to \mathbb{R}^C$. This expected property is usually defined locally (a.k.a. instance-wise property): the network performs correctly in the neighborhood $\mathcal{V}(\mathbf{x}_o) \subset \mathbb{R}^d$ of a particular input $\mathbf{x}_o \in \mathbb{R}^d$. Let us denote $\iota(\cdot|\mathbf{x}_o) : \mathbb{R}^d \to \{0, 1\}$ the function indicating a violation of the expected property. The network is locally correct if $\iota(\mathbf{x}|\mathbf{x}_o) = 0$ for any $\mathbf{x} \in \mathcal{V}(\mathbf{x}_o)$.

In classification, the property takes the name of *robustness* and reads as: the output of the network remains unchanged over the neighborhood $\mathcal{V}(\mathbf{x}_o)$. It certifies that the network is robust against inputs corrupted by uncertainties of limited support or adversarial perturbations of constrained distortion.

The certification mechanism has two desired features as defined in [152]:

- Soundness: it does not certify the network when the property does not hold.
- Completeness: it does certify the network whenever the property holds.

Corruption robustness assessment Adversarial robustness corresponds to a worst-case analysis whereas *corruption robustness* considers random perturbations of the inputs.

The key ingredient is the introduction of a statistical model π_0 of epistemic uncertainties occurring along the acquisition chain of the input. For instance, Franceschi, Fawzi, and Fawzi [8] take Gaussian and uniform distributions over the l_p ball $\mathcal{B}_{p,\epsilon}(\mathbf{x}_o)$ of radius ϵ centered on $\mathbf{x}_o$.

This recent trend goes with a *quantitative* assessment gauging to what extent a given property holds or does not hold. For instance, Webb, Rainforth, Teh, *et al.* [146] estimate the probability p that a property is violated under a given statistical model of the inputs. This approach makes no assumption about the network under scrutiny as it is used as a black box. This grants the scalability to tackle deep networks. The main difficulty lies in the efficiency, *i.e.* the computational power needed to estimate weak probabilities. Their lack of soundness stems from the inability to determine if the probability p of violation is exactly zero or too small to be estimated.

This work presents a scalable and efficient procedure assessing corruption robustness under a large panel of statistical models. It provides completeness and theoretical guarantees on the lack of soundness.

4.2 Our approach to corruption robustness assessment

Our approach uses statistical hypothesis testing as a certification *surrogate*. As in [148], the user sets a low critical probability p_c and the test assesses whether p is lower or larger. However, rather than a testing approach powered by crude Monte Carlo simulations, our workhorse is a more efficient Sequential Monte Carlo algorithm [153]. This so-called 'Last Particle' simulation was invented by Guyader, Hengartner, and Matzner-Løber [86] and is a variant of the Adaptive Multi-Level Sampling employed by Webb, Rainforth, Teh, *et al.* [146]. We show that, with a carefully chosen termination condition, it is advantageous both in terms of computational efficiency and theoretical guarantees.

Sect. 4.2.1 presents the 'Last particle' simulation that Sect. 4.2.2 applies to statistical hypothesis testing in the framework of robustness assessment. Alg. 9 gives the pseudo-code of our procedure.

4.2.1 The Last Particle simulation

The goal of the Last Particle simulation is to efficiently generate samples drawn according to a reference probability distribution π_0 but in a region $\mathcal{R} := \{\mathbf{y} : h(\mathbf{y}) > 0\} \subset \mathbb{R}^d$ where $h : \mathbb{R}^d \to \mathbb{R}$, is the so-called the *score function*. Efficiency is the ability to perform this task using few calls to the score function, even when probability $\pi_0(\mathcal{R})$ is small.

The simulation manages a set of N particles (*i.e.* samples) which are initially i.i.d. with respect to π_0. The name 'Last Particle' comes from the fact that the simulation 'kills' the sample with the lowest score at each step. The score of this last particle becomes the intermediate level L_k at iteration k (Alg. 9, line 6). Then, that particle is refreshed by sampling according to π_0 but conditioned on the event $\{h(\mathbf{X}) > L_k\}$. This sampling procedure is performed by $\mathsf{Gen}(L_k, 1)$ in line 11 and is detailed in Sect. 4.3. $\mathsf{Gen}(-\infty, N)$ then simply means sampling N random vectors according to π_0 (line 3).

The algorithm stops when the number of iterations reaches integer m or at any iteration k if the intermediate threshold L_k is positive which means the simulation has generated samples as required.

Algorithm 9 Robustness assessment with Last Particle simulation

```
Require: Number of particles $N$, critical probability level $p_c$, confidence interval level $\alpha$
Ensure: Cert
 1: Initialize: $p \leftarrow 1 - 1/N$,   $k \leftarrow 1$,   Cert $\leftarrow False$,   Stop $\leftarrow False$
 2: $m \leftarrow$ Comp_m$(p_c, \alpha, N)$                          ▷ See Sect. 4.2.3
 3: $\{\mathbf{x}_i\}_{i=1}^N \leftarrow$ Gen$(-\infty, N)$                        ▷ See Sect. 4.3
 4: while $k \le m$ & Stop $= False$ do
 5:     $i^\star \leftarrow \arg\min_{i \in 1:N} h(\mathbf{x}_i)$
 6:     $L_k \leftarrow h(\mathbf{x}_{i^\star})$
 7:     if $L_k > 0$ then
 8:         Stop $\leftarrow True$
 9:         $P_{est} \leftarrow p^{k-1}$
10:     end if
11:     $\mathbf{x}_{i^\star} \leftarrow$ Gen$(L_k, 1)$                           ▷ See Sect. 4.3
12:     $k \leftarrow k + 1$
13: end while
14: if Stop $= False$ then
15:     Cert $\leftarrow True$
16:     $P_{est} \leftarrow p_c$
17: end if
18: return Cert, $P_{est}$
```

Consider the function $\Lambda : \mathbb{R} \to \mathbb{R}_+$ defined as

$$\Lambda(\ell) := -\log \pi_0(h(\mathbf{X}) > \ell). \tag{4.1}$$

This function is unknown in practice, but one can easily see that it is non-decreasing.

During one run of Alg. 9, the intermediate levels are random variables following an increasing order by construction: $L_1 < L_2 < \cdots < L_k$. We here copy the main result of the Last Particle simulation:

Theorem 3 ([86])**.** *The variables $\Lambda(L_1), \Lambda(L_2), \cdots$ are distributed as the successive arrival times of a Poisson process with rate N: $\Lambda(L_k) = 1/N \sum_{j=1}^{k} E_j$, where $E_j \overset{i.i.d.}{\sim} \mathcal{E}(1)$.*

As the sum of i.i.d. exponential random variables is distributed[1] as a Gamma random variable, this theorem states that $\Lambda(L_k) \sim \Gamma(k, N)$ (*i.e.* scale k and rate N).

4.2.2 Corruption robustness assessment as a statistical test

In the framework of robustness assessment of classifiers, the score function is related to the usual loss in the adversarial example literature:

$$h(\mathbf{x}) := \max_{k \neq c(\mathbf{x}_o)} f_k(\mathbf{x}) - f_{c(\mathbf{x}_o)}(\mathbf{x}), \tag{4.2}$$

where $f(\mathbf{x})$ represents the predicted probabilities (or logits) vector and $c(\mathbf{x}) := \arg\max_k f_k(\mathbf{x})$ is the predicted class for input $\mathbf{x}$. Note that $h(\mathbf{x}_o) < 0$ and that the violation indicator function of Sect. 4.1 is simply $\iota(\mathbf{x}|\mathbf{x}_o) = \mathbb{1}(h(\mathbf{x}) > 0)$. The input π_0 models the corruption distribution around $\mathbf{x}_o$. The probability of robustness violation is written as $p := \pi_0(h(\mathbf{X}) > 0)$.

Our approach establishes a hypothesis test parametrized by a low probability p_c given by the user.

- $\mathcal{H}_0$: The probability of robustness violation $p > p_c$. The network should not be certified.
- $\mathcal{H}_1$: The probability of robustness violation $p < p_c$. The network can be certified.

For a given true probability of violation p, we establish the following properties.

1. Here and after, $\sim$ denotes distributional equality between random variables.

Proposition 1. *The probability of false positive* $P_{\mathsf{fp}}(p)$ *equals:*

$$P_{\mathsf{fp}}(p) := \mathbb{P}(\mathsf{Cert} = \mathsf{True} | p > p_c) = \frac{\int_0^{-N\log p} t^{m-1} \mathrm{e}^{-t} dt}{\int_0^{+\infty} t^{m-1}\mathrm{e}^{-t} dt} = \frac{\gamma(m, -N\log p)}{\Gamma(m)}. \tag{4.3}$$

Proof. Certification means that, according to the Alg. 9, even after m loops, the intermediate threshold L_m is still lower than 0. This happens with probability:

$$P_{\mathsf{fp}}(p) = \mathbb{P}(L_m < 0) = \mathbb{P}(\Lambda(L_m) < \Lambda(0)) = \frac{\gamma(m, -N\log p)}{\Gamma(m)}, \tag{4.4}$$

since $\Lambda(L_m) \sim \Gamma(m, N)$ and $\Lambda(0) = -\log p$; $\gamma(s, x)$ being the lower incomplete gamma function. □

Proposition 2. *The probability of false negative* $P_{\mathsf{fn}}(p)$ *equals:*

$$P_{\mathsf{fn}}(p) := \mathbb{P}(\mathsf{Cert} = \mathsf{False} | p < p_c) = \frac{\int_{-N\log p}^{+\infty} t^{m-1} e^{-t} dt}{\int_0^{+\infty} t^{m-1} e^{-t} dt} = \frac{\overline{\gamma}(m, -N\log p)}{\Gamma(m)}. \tag{4.5}$$

Proof. The certification failed because $L_K > 0$ for some $K \leq m$, or equivalently $\Lambda(L_K) > \Lambda(0)$. Note that this was not true at iteration $K-1$ (otherwise the while loop would have be broken earlier). In other words, $K - 1 = \sup\{i : \sum_{j=1}^{i} E_j < -N\log p, E_j \overset{\text{i.i.d.}}{\sim} \mathcal{E}(1)\}$, so that $K-1$ follows the Poisson distribution $\mathcal{P}(-N\log p)$. The probability of false negative is the cdf of $K-1$ at $m-1$:

$$P_{\mathsf{fn}}(p) = \mathbb{P}(K \leq m) = \mathbb{P}(K - 1 \leq m - 1) = \frac{\overline{\gamma}(m, -N\log p)}{\Gamma(m)}, \tag{4.6}$$

where $\overline{\gamma}(s, x)$ is the *upper* incomplete gamma function. □

This shows that $P_{\mathsf{fn}}(p)$ is an increasing function and the worst case happens when p converges to p_c:

$$\forall p < p_c,\ P_{\mathsf{fn}}(p) \leq P_{\mathsf{fn}}(p_c) = 1 - P_{\mathsf{fp}}(p_c). \tag{4.7}$$

Remark that the trade-off between false positive and false negative probabilities is hard at $p = p_c$. Yet, Eq. (4.6) tells that $P_{\mathsf{fn}}(p)$ is quickly vanishing as $p \to 0$, especially when N is large.

4.2.3 Corruption robustness assessment as a certification problem

In the context of certification, we show that i) our procedure is complete but not sound and ii) false positive probability drives the lack of soundness.

A false negative is not a bad event since it prevents us from certifying when the probability p of violation is not zero. At the same time, our procedure always certifies whenever the property holds since $P_{\mathsf{fn}}(0) = 0$. On the contrary, a false positive remains an error since we certify when $p > p_c > 0$. Let us quantify the lack of soundness by

$$P_{\mathsf{ns}}(p) := \mathbb{P}(\mathsf{Not\,Sound}\,|p) = \begin{cases} 1 - P_{\mathsf{fn}}(p) & \text{if } p < p_c \\ P_{\mathsf{fp}}(p) & \text{otherwise} \end{cases} \tag{4.8}$$

Let us recall that in our case it holds simply $1 - P_{\mathsf{fn}}(p) = P_{\mathsf{fp}}(p)$.

Proposition 3. *A suitable choice of the maximum number of iterations m in Alg. 9 can control the lack of soundness by the critical probability p_c and a required significance level $\alpha \in (0,1)$ s.t.*

$$P_{\mathsf{ns}}(p) \leq \alpha, \forall p \geq p_c. \tag{4.9}$$

Proof. This amounts to enforce that $P_{\mathsf{fp}}(p) \leq \alpha$, $\forall p > p_c$. Since $-\log p$ is a decreasing function, the worst case occurs in (4.4) when $p \to p_c$. It is thus safe to ensure $P_{\mathsf{fp}}(p_c) = \alpha$. This is done by carefully selecting m s.t. the α-quantile of the r.v. $\Gamma(m, N)$ equals $-\log p_c$. The routine Comp_m in Alg. 9 solves this numerically with a line search (see Appendix B.1 for some approximations). □

If we assume a Bayesian approach where the pdf of p is denoted by $f_P : [0,1] \to \mathbb{R}_+$, then the probability of not being sound is given by

$$\begin{aligned} \mathbb{P}(\mathsf{Not\,Sound}) &= \int_{0^+}^{p_c} (1 - P_{\mathsf{fn}}(p)) f_P(p) dp + \int_{p_c}^{1} P_{\mathsf{fp}}(p) f_P(p) dp && (4.10) \\ &\leq \int_{0^+}^{p_c} f_P(p) dp + \alpha \int_{p_c}^{1} f_P(p) dp = \alpha + (1-\alpha)\mathbb{P}(p < p_c). && (4.11) \end{aligned}$$

The lack of soundness decreases if both α and p_c are small. This makes the point with the state-of-the-art. Baluta, Chua, Meel, *et al.* [148] are unable to set p_c to a low value because their simulation is based on a crude Monte Carlo, whereas Webb, Rainforth, Teh, *et al.* [146] do not give any guarantee similar to our level α.

Table 4.1 – Maximum number of iterations m and its approximation $\tilde{m}_1$ (see App. B.1)

N	p_c	$\alpha = 0.1$		$\alpha = 0.01$		$\alpha = 0.001$	
		m	$\tilde{m}_1$	m	$\tilde{m}_1$	m	$\tilde{m}_1$
20	10^{-10}	489	489	512	514	529	532
20	10^{-30}	1430	1431	1470	1471	1499	1502
10	10^{-10}	251	251	267	269	280	283
10	10^{-30}	726	726	754	755	774	777
2	10^{-10}	56	56	64	65	69	73
2	10^{-30}	154	155	167	169	177	180

Efficiency Appendix B.1 proposes approximated closed forms outlining that m scales as $\log {}^1\!/\!{}_{p_c}$. This is also visible in the typical values given in Table 4.1. A lower significance level moderately increases the number of iterations. Section 4.3 details how to sample a new particle at each iteration as needed in line 11, Alg. 9. This method consumes a fixed number of calls to the network. In total, the maximum number of calls scales as $O(\log {}^1\!/\!{}_{p_c})$. This is in stark contrast with [148] where the number of calls is proportional to ${}^1\!/\!{}_{p_c}$. Note that this is a maximum: our procedure makes an early stop whenever $L_k > 0$ (line 8, Alg. 9) and outputs $Cert = False$ as well as failure probability estimate P_{est}.

4.3 Sampling procedures

This section details the crucial ingredient of our procedure: sampling a new input $\mathbf{X}$ whose score $h(\mathbf{X})$ is above a given level L. This random generator is called $\mathsf{Gen}(L, 1)$ in line 11, Alg. 9. Appendix B.2 considers a case where the statistical model π_0 and the network are so simple that this sampling is easy. This section details more general scenarios making no assumption about the score function. Our sampling is a rejection procedure relying on *reversible proposals* and *transformations*.

4.3.1 Reversible proposals

We call a (parametric) proposal any a random function $K : \mathbb{R}^d \times \mathbb{R}_+ \to \mathbb{R}^d$. Iterations of i.i.d. proposal generate a Markov chain which is said to be *reversible (detailed balance)* with respect to the distribution π_0 if the following assertion holds:

$$\forall s > 0,\, \mathbf{X} \sim \pi_0 \Rightarrow (\mathbf{X}, K(\mathbf{X}, s)) \sim (K(\mathbf{X}, s), \mathbf{X}). \tag{4.12}$$

A simple example for $\pi_0 = \mathcal{N}(\mathbf{0}_n; \sigma^2\mathbf{I}_n)$ is given in [86]:

$$K(\mathbf{X}, s) := \frac{\mathbf{X} + s\mathbf{N}}{\sqrt{1+s^2}} \quad \text{with} \;\; \mathbf{N} \sim \mathcal{N}(\mathbf{0}_n; \sigma^2\mathbf{I}_n). \tag{4.13}$$

The rejection method described in Alg. 10 takes as input a set $\mathcal{X}$ of particles whose score is larger than L. It randomly picks one particle in $\mathcal{X}$ and applies T times a fresh proposal, followed by a rejection based on the score. If the selected sample is a realization of the distribution π_0 conditioned by a score larger than L, then one application of the proposal keeps π_0 invariant while the rejection ensures that the score remains above L. By induction, iterating maintains these two properties, and in fact leaves invariant the conditional distribution thanks to reversibility (see [86] and App. B.5).

Alg. 9 uses the procedure of Alg. 10 as follows. At iteration k, L is indeed L_k, *i.e.* the score of the 'last' particle $\mathbf{x}_{i^\star}$, and $\mathcal{X} = \{\mathbf{x}_i\}_{i \neq i^\star}$ which contains $(N-1)$ particles whose score is larger than L. The output is one 'fresh' particle and the number of particles equals N from one iteration to another.

The parameter s plays the role of *strength*: $s = 0$ implies that the proposal just copies the input, while $s \to +\infty$ means that $K(\mathbf{x}, s)$ does not depend on $\mathbf{x}$. The proposal strength s is thus important. With a small value, the proposal makes small moves. A large value explores faster but leads to higher rejection rate. Appendix B.4 presents a strategy to automatically control its value depending on the past behavior of the algorithm in order to maintain a given rejection rate.

Theoretically, under some irreducibility assumption, an infinity of iterations in Alg. 10 provides a fresh particle statistically independent of the particles in $\mathcal{X}$ as needed in Alg. 9:

Proposition 4. *Assume that, the proposal $K(\mathbf{x}, s)$ has a density bounded from below uniformly in $\mathbf{x}$ and $s \geq s_0$. Then the distribution of $\Lambda(L_m)$ converges towards the Gamma distribution $\Gamma(N, m)$ exponentially fast with the number T of proposal applications.*

Proof and Remarks. The proof is given in Appendix B.5 and uses a classical probabilistic coupling argument. It requires the lower bound assumption which is a form of strong irreducibility of the proposal. This is compliant with the proposals used in this work. In particular all the formulas given in Prop. 1 and after hold true asymptotically for large

Algorithm 10 Sampling one particle $\mathsf{Gen}(L, 1)$

```
Require: threshold $L$, finite set $\mathcal{X}$ of particles whose score is larger than $L$
Ensure: new particle $\mathbf{X}$
1: $\mathbf{X} \leftarrow \mathcal{U}(\mathcal{X})$                    ▷ Draw uniformly a particle in $\mathcal{X}$
2: for $k = 1 : T$ do
3:     $\mathbf{Z} \leftarrow K(\mathbf{X}, s)$                  ▷ $\pi_0$ reversible proposal. See Sect. 4.3.1
4:     if $h(\mathbf{Z}) > L$ then                    ▷ Rejection
5:         $\mathbf{X} \leftarrow \mathbf{Z}$
6:     end if
7: end for
8: return $\mathbf{X}$
```

T. □

In practice, we choose the number T of iterations approximately proportional to the inverse of the rejection rate, maintained approximately constant by tuning the proposal strength s (see App. B.4).

Refreshing a particle consumes T calls to the score function. This is done *once* per iteration of Alg. 9. Therefore, our method globally consumes $O(T \log {}^1/_{p_c})$ calls. This means that the figures in Table 4.1 are to be multiplied by T. Webb, Rainforth, Teh, *et al.* [146] also manage a sample of size N, but *all* the particles are separately refreshed at *each* iteration by applying T Metropolis-Hasting transitions. Their number of calls per iteration is N times larger than our. Moreover, their typical setup is $N \approx 1000$ and $T \approx 1000$, while ours is $N \approx 2$ and $T \approx 50$. Our complexity is thus smaller by 4 orders of magnitude.

4.3.2 Isoprobabilistic transformation

The proposal (4.13) is simple but reversible only w.r.t. the normal distribution. The isoprobabilistc transformation method is well known in the field of Statistical Reliability Engineering [154], see Sect. 1.2.2. It amounts to working with a latent random vector $\mathcal{G} \sim \mathcal{N}(\mathbf{0}_d; \mathbf{I}_d)$ and to apply the inverse transformation $\mathbf{X} = \mathcal{T}^{-1}(\mathcal{G}, \mathbf{x}_o)$ mapping the normal distribution to the reference model π_0. Some well-known examples are:

- $\mathbf{X} \sim \mathcal{N}(\mathbf{x}_o, \sigma^2 \mathbf{I}_n)$: $D = d$ and $\mathcal{T}^{-1}(\mathcal{G}, \mathbf{x}_o) = \mathbf{x}_o + \sigma\mathcal{G}$
- $\mathbf{X} \sim \mathcal{U}\left(\mathcal{B}_{+\infty,\epsilon}(\mathbf{x}_o)\right)$: $D = d$ and $\mathcal{T}^{-1}(\mathcal{G}, \mathbf{x}_o) = \mathbf{x}_o + \epsilon(2\Phi^{-1}(\mathcal{G}) - 1)$ (component-wise)
- $\mathbf{X} \sim \mathcal{U}(\mathcal{B}_{2,\epsilon}(\mathbf{x}_o))$: $D = d + 2$ and $\mathcal{T}^{-1}(\mathcal{G}, \mathbf{x}_o) = \mathbf{x}_o + \epsilon\mathcal{G}(1 : d)/\|\mathcal{G}\|_2$

More complex examples are inverse Rosenblatt or Nataf transformations [154]. See also section 1.2.2.

This transformation is composed with h to redefine the score function $h_G = h \circ \mathcal{T}^{-1}$ that applies on latent vector $\mathcal{G}$, *i.e.* random vectors suitable for the proposal (4.13). This amounts to use Alg. 9 directly on the latent variable with score function h_G and in conjunction with Alg. 10 and proposal (4.13).

4.4 Experimental evaluation

This section presents experimental results on ACAS Xu, MNIST, and ImageNet datasets with some trained classifications networks listed in App. B.9 together with implementation details. Experiences were run on a laptop PC (CPU=Intel(R) Core(TM) i7-9750H, GPU=GeForce RTX 2070) except for experiences on ImageNet which were run on a Nvidia V100 GPU.

4.4.1 Idealized case

This section considers a setup where $\pi_0 = \mathcal{N}(\mathbf{x}_o; \sigma^2 \mathbf{I}_n)$ and score function h is linear. This setup is ideal because sampling a fresh particle is straightforward (*i.e.* without Alg. 10) as shown in App. B.2.

Fig. 4.1 shows the impact of N. In terms of hypothesis testing (see Sect. 4.2.2), a larger N yields steeper functions: $P_{\mathsf{fp}}(p)$ (resp. $P_{\mathsf{fn}}(p)$) quickly vanishes to zero as p gets larger (resp. smaller) than p_c. In terms of certification (see Sect. 4.2.3), a small N is not a bad choice: the probability $P_{\mathsf{ns}}(p)$ of not being sound takes lower values in the range $p < p_c$. For $p > p_c$, $P_{\mathsf{ns}}(p)$ is lower than α (as stated by Prop. 3) but converges to 0 more slowly. Last but not least, the procedure makes only 167 calls to the score function for $N = 2$, instead of 1470 for $N = 20$.

4.4.2 ACAS Xu

We evaluate our method on the ACAS Xu (Airborne Collision Avoidance System X for unmanned aircrafts) case study [155]. It consists in 45 neural networks used to approximately compress a large lookup table (2GB) containing discrete decisions ('Clear-of-conflicts', 'weak right', 'strong right','weak left', or 'strong left') as well as 5 input/output properties. Each input has dimension $d = 5$ in this case and $45 * 5 = 225$ neural

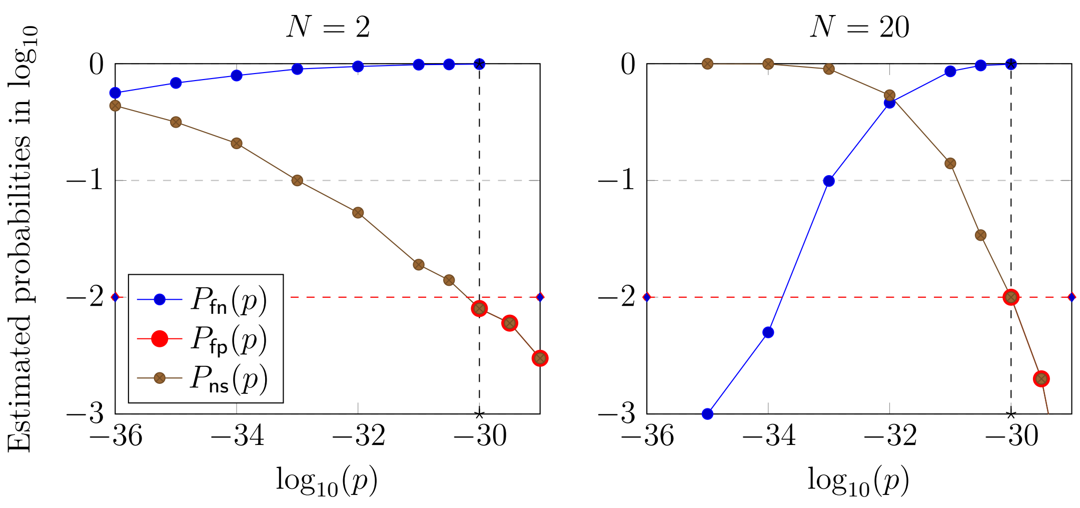


Figure 4.1 – Estimated probabilities of false positive, false negative, and not sound certification, vs. true violation probability p in the ideal setup where $p_c = 10^{-30}$, $\alpha = 0.01$. Estimation over 1000 runs.

network/property pairs must be verified. We compare our method with the complete certification based on DeepPoly [139] and Mixed-Integer Programming from the ERAN benchmark.

Table 4.2 contains the confusion matrix taking into account the cases for which the ERAN complete certification fails because the Gurobi optimizer either outputs an 'infeasible' status or reaches a timeout (set to 600 seconds). Unsurprisingly, our method is complete in the sense that it certifies all cases certified by ERAN. It is not sound as it admits 9 false positives. This is due to the critical probability p_c which is not low enough (the decisions were the same over 10 runs). Yet, our method takes a decision on the 6 unsolved cases by ERAN. In addition our method is faster for all ACAS Xu properties except for the property 4, confer figure 4.2.

Table 4.2 – ACAS Xu – Confusion matrix comparing ERAN [DeepPoly+MILP] and Last Particle $[N = 2, p_c = 10^{-50}, T = 40]$

		ERAN			
		Certified	Uncertified	Infeasible	TimeOut
Last Particle	Certified	107	9	1	1
	Uncertified	0	103	4	0

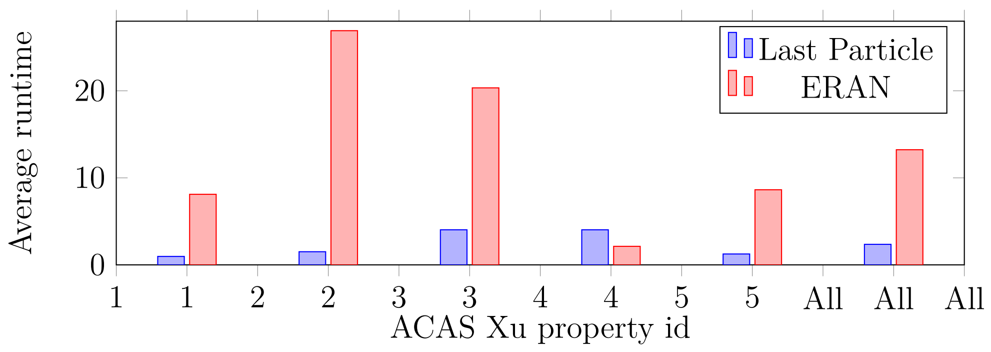


Figure 4.2 – ACAS Xu – runtimes in sec. of ERAN (Deep Zonotope) and Last Particle algorithm $[N = 2, p_c = 10^{-50}, T = 40]$

4.4.3 MNIST

We compare our procedure with the DeepPoly *incomplete* certification on MNIST [156] with 4 neural networks from the ERAN benchmark (see App. B.9). Each input in MNIST is a 28×28 pixels black and white picture of a digit, so in this case the dimension of the problem is $d = 784$. We focus on L_∞ uniform robustness since the implementation provided for DeepPoly cannot deal with L_2 norms. We run our algorithm with $N = 2, p_c = 10^{-35}$ and $T = 40$. As in ACAS Xu experiment, our method runs faster than the ERAN method as shown in table 4.3. Interestingly, the average runtime of our method decreases with larger ϵ since the probability p of violation is bigger, whereas DeepPoly computation time increases with the size of the input space tested. On the one hand, DeepPoly provides an efficient lower bound to both corruption and adversarial robustness, on the other hand, our method provides a fast upper bound. 10 independent LP simulations (runs) on the same image always give the same output and the standard deviation is thus empirically negligible in our setting.

4.4.4 ImageNet

For the last experiment, our method analyses 2 neural networks (ResNet50 et MobileNet) with 100 test images from ImageNet dataset [157] correctly classified by each network. Inputs in this dataset are (224×224)-pixels RGB pictures, thus the dimension of the problem in this case is $d = 150528$. These experiments were run on a Nvida V100 GPU. The average number of calls reported is rounded up and the average runtime is for a pass over one image. The robustness is again defined against noise uniformly distributed over

Table 4.3 – MNIST – Comparison ERAN [DeepPoly] and Last Particle [$N = 2, p_c = 10^{-35}, T = 40$]

	ERAN		Last Particle	
ϵ	Certified (%)	runtime (sec.)	Certified (%)	runtime (sec.)
0.015	82	5.69	99	1.04 ± 0.005
0.03	62	5.92	97	1.03 ± 0.01
0.06	28	8.13	93	1.00 ± 0.01
0.1	22	8.84	85	0.96 ± 0.02

L_∞ of radius ε. As one can notice, the compute time increases reasonably the input space dimension and network size.

Table 4.4 – ImageNet - Last Particle [$N = 2, p_c = 10^{-15}, T = 20$]

Network	ϵ	Avg. runtime (in sec. $\pm std$)	Avg. number of calls	Certified (%)
MobileNet	0.02	20.78 ± 0.74	1388	71
	0.03	18.74 ± 0.18	1274	64
	0.06	14.5 ± 0.11	1037	50
ResNet50	0.02	33.86 ± 1.14	1537	81
	0.03	31.38 ± 0.48	1434	71
	0.06	25.51 ± 0.67	1160	59

4.5 Conclusion and Limitations

The paper proposes a statistical simulation to assess corruption robustness. It looks at this problem from a hypothesis testing (false positive/ false negative) and from a certification (completeness / soundness) points of view. The procedure is scalable, efficient, complete and comes with guarantees on the lack of soundness. There are two limitations:

- 1) The Last Particle simulation is sequential, which is not GPU-friendly. Yet, we provide a code processing several inputs $\mathbf{x}_o$ in parallel.
- 2) Our procedure is general as it uses the network as a black-box classifier. However,

it does not exploit its gradient easily computed thanks to backpropagation. More sophisticated mixing kernels using gradient information (e.g. Langevin Monte Carlo, Hamiltonian Monte Carlo [9]) can accelerate convergence.

Moreover, the statistical guarantees we obtained, though non-asymptotic w.r.t. to the number of particles N, apply exactly only for the *idealized* algorithm introduced above, which corresponds by ergodicity to an asymptotic guarantee for the original algorithm as T goes to $+\infty$. In practice, we observed that taking $T \approx 40$ was sufficient to obtain convergence, as the results did not change drastically with a higher number of kernel iterations. However, obtaining non-asymptotic guarantees, under reasonable hypothesis on the score function s, with both a finite number of samples N and a finite number of kernel iterations would be a crucial addition, allowing for a more reliable local robustness testing algorithm.

CHAPTER 5

Gradient-Informed Neural Network Statistical Robustness Estimation

5.1 Introduction

In many machine learning applications, test data are captured by a sensor, then quantized or compressed, and finally transmitted to the model. In each of these steps, uncertainties may corrupt the pieces of data. For instance, in autonomous driving, capturing the scene at night implies increasing the ISO parameter of the camera whence a photo-site noise increase [158]. Statistical (a.k.a. Corruption) robustness is defined as the ability to perform correctly on test data corrupted by a given random noise distribution [159].

Statistical robustness is a related but different concept from adversarial robustness, which is the ability of a model to perform correctly over maliciously perturbed data [124]. Gilmer, Ford, Carlini, *et al.* [7] study in detail the relationship between these concepts. Moreover, Franceschi, Fawzi, and Fawzi [160] provide theoretical and empirical evidence that, for input data of dimension d, there is a ratio of order $\frac{1}{\sqrt{d}}$ between the typical power of an adversarial signal and that of a random perturbation to trigger a misclassification in a neural network. Neural network classifiers are thus robust against random noise to a certain extent. Yet, safety requirements in critical applications, e.g. in autonomous driving [149], ask for very weak probabilities of failure under a typical noise power [161].

Webb, Rainforth, Teh, *et al.* [146] quantify the local statistical robustness of a model by its probability of failure. The major difficulty of this quantitative approach is the estimation of weak probabilities with accuracy and low complexity. The above-mentioned work resorts to a Sequential Monte Carlo technique (SMC, [162], [163]), dedicated to rare event analysis: the Adaptive Multi-Level Splitting (AMLS) procedure [86]. Querying the model is assumed to be time-consuming and the number of calls gauges the complexity of the estimator. Although their work proposes experiments on neural networks, it is not specific to deep learning as they use the model as a black box function. Our paper proposes an alternative gradient-informed SMC algorithm.

The proposed algorithm trades off the universality of the aforementioned work against a lower complexity, by leveraging a key feature of neural networks: the computation of the gradient of the model (here w.r.t. its input) is easily implemented thanks to auto-differentiation and cheaply run thanks to back-propagation.

5.2 Sketch of the Algorithm

5.2.1 Hamiltonian Monte Carlo

The Hamiltonian Monte Carlo (HMC) is a type MCMC sampling method [9] making use of gradient computations w.r.t. a given differentiable potential $U : \mathbb{R}^d \mapsto \mathbb{R}$ (see also appendix A) to generates samples asymptotically distributed according to $\pi_U \propto e^{-U}$. We define (up to a constant) the joint probability measure μ: $\mu(q,p) \propto e^{-H(q,p)} dqdp$, with

$$H(q,p) \triangleq U(q) + \|p\|^2/2,$$

and the related Hamiltonian dynamics in continuous time:

$$\begin{cases} dQ_t = P_t dt, \\ dP_t = -\nabla U(Q_t) dt. \end{cases} \tag{5.1}$$

Hamiltonian Monte Carlo method is based on the symplectic and time-reversible *discretization* of this dynamics, also known as the Verlet scheme:

$$\begin{cases} P_{i+1/2} = P_i - \frac{1}{2}\nabla U(Q_i)\Delta t, \\ Q_{i+1} = Q_i + P_{i+1/2}\Delta t, \\ P_{i+1} = P_{i+1/2} - \frac{1}{2}\nabla U(Q_{i+1})\Delta t. \end{cases} \tag{5.2}$$

After L iterations of the Verlet scheme, the deterministic map $(Q_0, P_0) \mapsto (Q_L, -P_L)$ is reversible in time (an involution), and it conserves the flat phase-space measure $dqdp$. One can then apply the standard Metropolis accept-reject rule, which consists in accepting the proposal $(Q_L, -P_L)$ with probability:

$$\min(1, e^{-H((Q_L, -P_L))+H(Q_0,P_0)}). \tag{5.3}$$

The result is denoted $(\tilde{Q}_L; \tilde{P}_L) \in \{(Q_0, P_0), (Q_L, -P_L)\}$. The obtained Markovian kernel has stationary joint distribution μ, whose first marginal is the target π_U. When $L = 1$ the HMC kernel becomes a Metropolis-adjusted discretization of a diffusion with a drift $-\nabla U$, called the Metropolis Adjusted Langevin Algorithm (MALA, see [164]).

First, we define a one-parameter family of smooth densities, which enables sampling with the gradient-informed HMC kernel defined above. Second, we recast the rare event problem as the normalization with respect to π_0 of the final density of this family. Finally, we construct our algorithm as a SMC approach [163] integrating the gradient-informed HMC kernel.

5.2.2 A Family of Distributions: Gibbs Measures

Let us consider the potential function:

$$\mathbf{x} \in \mathbb{R}^d \mapsto V(\mathbf{x}) \triangleq |-h(\mathbf{x})|_+ \geq 0, \tag{5.4}$$

where $|a|_+ = a$ if $a \geq 0$ and 0 otherwise, and function h defined in Sect. 4.2.2 of the previous chapter. Here the set $\{\mathbf{x} : V(\mathbf{x}) = 0\}$ is the set of latent vectors that give birth to violations of the local robustness property since the corresponding inputs are classified with a different label than that of of the original input $\mathbf{x}_0$. Roughly speaking, points with a low potential value are 'close' to being misclassified, whereas points with a high potential are 'far' from the decision boundary.

Given in addition a tempering parameter $\beta \geq 0$ (called the inverse temperature in statistical physics) and the reference measure π_0 modelling uncertainties in the inputs, we construct the following family of probabilities:

$$\pi_\beta(d\mathbf{x}) \triangleq \frac{e^{-\beta V(\mathbf{x})}}{Z_\beta} \pi_0(d\mathbf{x}), \tag{5.5}$$

$$Z_\beta = \int e^{-\beta V(\mathbf{x})} \pi_0(d\mathbf{x}) = \mathbb{E}_{\pi_0}\left[e^{-\beta V(\mathbf{x})}\right], \tag{5.6}$$

where $Z_\beta < +\infty$ is the normalizing constant assumed to be finite.

This family of distributions, called the Gibbs measures, ranges from the reference measure π_0 for $\beta = 0$ to the same distribution but conditioned on the event of failure when $\beta \to \infty$. Indeed, it happens that

$$\forall \mathbf{x} \in \mathbb{R}^d \exp(-\beta V(\mathbf{x})) \overset{\beta\to\infty}{\to} \mathbb{1}_{V(\mathbf{x})=0} \tag{5.7}$$

In other words, $\pi_{+\infty}(d\mathbf{x}) = \mathbb{1}_{s(\mathbf{x})\geq 0}\pi_0(d\mathbf{x})/Z_{+\infty}$.

We denote the expectation of a test function t over the distribution π_β by $\mathbb{E}_{\pi_\beta}[t(\mathbf{x})]$. An interesting property used to construct our estimator in section 5.2.3 is that for any couple $(\alpha, \beta) \in \mathbb{R}_+^2$, the ratio Z_β/Z_α can be expressed as an expectation over the measure π_α:

$$Z_\beta = Z_\alpha \mathbb{E}_{\pi_\alpha}\left[e^{-(\beta-\alpha)V(\mathbf{x})}\right]. \tag{5.8}$$

Therefore, if one can sample from measure π_α, then one can estimate the ratio Z_β/Z_α by an empirical average.

Connection to the Rare Event Normalizing constants are in general analytically intractable and hard to integrate numerically. In the family of distributions we consider (5.5), we simply have $Z_0 = 1$, while for $\beta = +\infty$ the normalizing constant is indeed the probability we want to estimate:

$$\begin{aligned} Z_{+\infty} = \mathbb{E}_{\pi_0}\left[\mathbb{1}_{s(\mathbf{x})\geq 0}\right] = \mathcal{R}[\theta, \rho_0].Z_{+\infty} &= \int \mathbb{1}_{V(\mathbf{x})=0}\pi_0(d\mathbf{x}) = \langle \mathbb{1}_{V=0}\rangle_{\pi_0} \\ &= \int \mathbb{1}_{s(\mathbf{x})\geq 0}\pi_0(d\mathbf{x}) \qquad = \langle \mathbb{1}_{s\geq 0}\rangle_{\pi_0} = \mathcal{I}[\pi_0, s, \theta, \phi]. \end{aligned}$$

that is the probability that the perturbed input is classified with a different label.

$$\begin{aligned} Z_{+\infty,0} &= \int \mathbb{1}_{V(\mathbf{x})=0}\pi(d\mathbf{x}) \\ &= \mathbb{P}[predict(x_0 + U) \neq predict(x_0)], U \sim \pi_0 \end{aligned}$$

The problem becomes the evaluation of the ratio $Z_{+\infty}/Z_0$, known to be (in statistical physics or Bayesian statistics) more tractable than the direct evaluation of normalizations.

5.2.3 Key Ideas Supporting the Algorithms

Our algorithm is built upon the following key ideas. The first step is to split the estimation problem into several easier intricate ones like in multilevel splitting algorithms [165]. This is driven by a series of increasing parameters $0 = \beta_0 \leq \beta_1 \leq \cdots \leq \beta_{K-1} \leq \beta_K = +\infty$. Indeed, using equation (5.8) recursively, we have

$$\begin{aligned} Z_{\beta_K} &= \mathbb{E}_{\pi_{\beta_{K-1}}}\left[e^{-(\beta_K-\beta_{K-1})V(\mathbf{x})}\right] \times Z_{\beta_{K-1}} \\ &= \prod_{k=0}^{K-1} \mathbb{E}_{\pi_{\beta_k}}\left[e^{-(\beta_{k+1}-\beta_k)V(\mathbf{x})}\right]. \end{aligned} \tag{5.9}$$

The main idea is then to estimate each expectation term of the product by an empirical average over a sample of N particles $\{X_k^{(n)}\}_{n=1}^N$ with empirical distribution $\pi_{\beta_k}^N \triangleq \frac{1}{N}\sum_{n=1}^N \delta_{\mathbf{x}_k^{(n)}}$, and then to use a Sequential Monte Carlo (SMC) strategy [162] to sample $\pi_{\beta_{k+1}}^N$ from $\pi_{\beta_k}^N$. In the present context, we use the following simple iteration of selection and mutation steps:

Selection For each particle $X_k^{(n)}$ in the sample defining $\pi_{\beta_k}^N$, kill it with probability $e^{-(\beta_{k+1}-\beta_k)[V(\mathbf{x}_k^{(n)})-\min_m V(\mathbf{x}_k^{(m)})]}$. Let K_k be the number of killed particles. Draw K_k new particles uniformly among the $N-K_k$ survivors to keep the sample size equal to N.

Mutation Independently mutate particles using a Markov kernel $X_{k+1}^{(n)} = \mathtt{Ker}(X_k^{(n)}, \beta_{k+1})$ which leaves invariant the distribution $\pi_{\beta_{k+1}}$ (see Sect. 5.3.1).

The following proposition establishes the soundness of this simulation method.

Proposition 5. *The following estimator of $\mathcal{R}[\theta, \rho_0] = Z_{+\infty}$ is unbiased and consistent:*

$$\widehat{Z}_{+\infty} = \prod_{k=0}^{K-1} \mathbb{E}_{\pi_{\beta_k}^N}\left[e^{-(\beta_{k+1}-\beta_k)V(\mathbf{x})}\right] = \prod_{k=0}^{K-1} \frac{\widehat{Z}_{\beta_{k+1}}}{\widehat{Z}_{\beta_k}}.$$

See App. C.2 for the proof.

In a similar way, if one denote

$$X_1(X_0) = Ker(X_0, V, \beta_1, \Delta t)$$

one random step of discretized Langevin scheme leaving π_{β_1} invariant, then for any test function ϕ, one has :

$$\begin{aligned}\langle \phi \rangle_{\pi_{\beta_1}} &= \langle \mathbb{E}\left[\phi(X_1)\right]\rangle_{\pi_{\beta_1}} \\ &=\frac{Z_{\beta_0}}{Z_{\beta_1}}\left\langle \mathbb{E}\left[\phi(X_1)e^{-(\beta_1-\beta_0)V}\right]\right\rangle_{\pi_{\beta_0}}. \end{aligned} \tag{5.10}$$

The next step is to split the estimation problem into several easier intricate ones like in multilevel splitting algorithms [165]. This is driven by a series of increasing inverse temperatures:
$\beta_0 \le \beta_1 \le \cdots \le \beta_{K-1} \le \beta_K$,

$$\begin{aligned} Z_{\beta_K} &= \frac{Z_{\beta_K}}{Z_{\beta_{K-1}}} \times Z_{\beta_{K-1}} \\ &= \left\langle e^{-(\beta_K-\beta_{K-1})V}\right\rangle_{\pi_{\beta_{K-1}}} Z_{\beta_{K-1}} \\ &= \left\langle \mathbb{E}\left[e^{-(\beta_K-\beta_{K-1})V(X_{K-1}(x_{K_2})-(\beta_{K-1}-\beta_{K-2})V(x_{K-2}}\right]\right\rangle_{\pi_{\beta_{K-2}}(dx_{K_2})} Z_{\beta_{K-2}}, \\ &= \left\langle \mathbb{E}\left[e^{-\sum_{k=1}^{K}(\beta_k-\beta_{k-1})V(X_{k-1}(x_0))}\right]\right\rangle_{\pi_0(dx_0)} Z_{\beta_0}, \end{aligned}$$

where in the last line, we have defined $X_k(x_0) = Ker(X_{k-1}(x_0), V, \beta_k, \Delta t)$ for $k \ge 1$ and $X_0(x_0) = x_0 \sim \pi_0$; and then have used (5.10) iteratively on k. Thus we devise a first simple estimator of $\mathcal{I}$,

$$\widehat{\mathcal{I}} = \frac{1}{N}\sum_{n=1}^{N} \mathbb{1}_{V(\mathbf{x}_{K-1}^{(n)})=0} \times e^{-\sum_{k=1}^{K-1}(\beta_k-\beta_{k-1})V(\mathbf{x}_{k-1}^{(n)})}, \tag{5.11}$$

where $\mathbf{x}^{(n)}, n = 1 \dots N$ denotes N i.i.d. copies with initial $X_0 \sim \pi_0$.

Estimators of normalizing constants based on (5.11) have been extensively studied in the Sequential Monte Carlo literature [162], [166]. In statistical physics, it is referred to as 'Jarzinsky's equality' for non-equilibrium [1] processes [167]–[169]. In the present work, we also adaptively tune the choice of the cooling schedule denoted now $\widehat{\beta}_k^N$ at step k. The latter must be a function of the empirical distribution of the $\mathbf{x}_{k'}^{(n)}$ for $k' \le k$ that mitigate the weight degeneracy in (5.11) per unit computation cost. This idea has been for instance analyzed in [163].

1. Non-equilibrium refers to the fact that the parameter β_k defining the chain X_k, may evolve slightly faster than the typical mixing time of the associated Markov chain.

5.3 Proposed Algorithm

This section details the algorithm whose pseudo-code is described in Alg. 11.

5.3.1 Sampling Kernel

Given a particle $X_k^{(n)}$ issued from the selection step, the mutation step in our main method uses the Hamiltonian Monte Carlo method of Sect. 5.2.1. We set $U = \beta_k V - \log \pi_0$ ($= -\log \pi_{\beta_k}$ up to an additive constant), $Q_0 = X_k^{(n)}$, and we sample P_0 independently according to a standard Gaussian law. We apply L steps of the Verlet scheme (5.2) and accept the outcome with probability given in (5.3). The result of this Metropolis step denoted $\tilde{Q}_L$ in Sect. 5.2.1 defines the sampling kernel:

$$\texttt{Ker}_{\Delta t, L}(X_k^{(n)}, \beta_k) \triangleq \tilde{Q}_L. \tag{5.12}$$

This kernel is invariant for the distribution π_{β_k} which is already approximated by the sample $(X_k^{(n)})_n$ after the selection step. We can adaptively tune the number of times the kernel is applied by applying the statistical stopping condition, based on auto-correlations, proposed by [170]. This corresponds to the `AutoCorrCond` function used at line 20 in the algorithm 11. In the experiments section 6.3 we chose T_{max} such that the condition is never reached, i.e. $T = T_{max}$. The auto-correlations condition is used to make sure that the proposed sample has low correlation to the initial value $X_k^{(n)}$. Similarly the time step Δt and the number L of Verlet iterations of the scheme (5.2) are tuned for each particle using the algorithm 5 of [170], based on previous work from [171] in the context of Bayesian Analysis. This is a genetic algorithm adapting at each iteration of the kernel the value of Δt and L parameters using the ESJD (expectd square jump distance) performance metric [170]. We also use in the experiments section the MALA (Metropolis-Adjusted Langevin Algorithm) method, in which case L is fixed to 1 and only Δt is adapted, using the same genetic algorithm. The `BatchKer` notation in line 18 outlines the batch processing of the particles through the Verlet scheme and the Metropolis steps.

Note that the potential in the Verlet scheme need not be equal to the target potential $\tilde{U} \neq U$ in order for the kernel to exactly leave invariant π_U. In particular, if $U = -\log \pi_0(q)$ with π_0 a standard isotropic Gaussian, it is possible to choose:

$$\tilde{U} = -\kappa_{\Delta t} \log \pi_0(q)$$

with $-\kappa_{\Delta t} = \frac{2}{1+\sqrt{1-\Delta t^2}}$ (notice that $\kappa_{\Delta t} \overset{\Delta t \to 0}{\to} 1$) which implies the exact energy conservation $H(Q_L, -P_L) = H(Q_0, P_0)$ leading to an identically one acceptance rate.

We now address the issue of sampling according to π_{β_k} in order to estimate each term of product (5.9) with Sequential Monte Carlo simulations. We assume that we know at least how to sample according to the reference measure π_0. This gives a first set of inputs denoted by $\{x_0^{(n)}\}_{n=1}^N$. We define the following independent Markov chains based on (5.12): for $k \in [0 : K-1]$ and $n \in [1 : N]$,

$$X_{k+1}^{(n)} = Ker(X_k^{(n)}, \beta_k V - log(\pi_0), \Delta t), \tag{5.13}$$

and $X_0^{(n)} = x_0^{(n)} \sim \pi_0$.

5.3.2 Adaptive Tempering Parameters

The proposed algorithm also tunes the choice of the cooling schedule $(\beta_k)_k$. A too rapidly increasing schedule increases the variance of the terms in product (5.11). A too-slow schedule increases the number of steps to reach the rare event, whence a bigger complexity. This trade-off has been studied in [163] for instance, where it is recommended to maintain the Efficient Sample Size constant across the steps. It amounts to set the next tempering parameter such that

$$\texttt{ESS}(\beta_{k+1}) = \alpha N \tag{5.14}$$

for a user-chosen $\alpha \in (1/N, 1)$ where

$$\texttt{ESS}(\beta) \triangleq \frac{\left(\sum_{n=1}^N e^{-(\beta-\beta_k)V(X_k^{(n)})}\right)^2}{\left(\sum_{n=1}^N e^{-2(\beta-\beta_k)V(X_k^{(n)})}\right)}. \tag{5.15}$$

Note that `ESS` is a continuous and decreasing function. The value of $\beta_{k+1} > \beta_k$ is numerically found using the bisection method. This process is called `AdaptESS` in lines 2 and 23. In practice the choice of parameter α in eq. 5.14 is crucial to obtain an efficient schedule. In the absence of a theoretically founded rule for selecting this parameter, we propose to fine-tune it for each Neural Network architecture.

5.4 Experiments

This section compares the efficiency of four Sequential Monte Carlo methods:

H-SMC This is our gradient-informed algorithm, *i.e.* the SMC Alg. 11 with the HMC kernel at line 18. The main parameters are T and α, introduced in Sect. 5.3.

MALA-SMC A variant of our method, where the HMC kernel at line 18 is constrained with $L = 1$ (instead of being adaptively tuned, see Sect. 5.3.1).

RW-SMC A black-box version of our method, the HMC at line 18 being replaced by a Metropolis-Hastings kernel [172] with Gaussian proposals Q: $Q(X, s) \triangleq \frac{X+sG}{\sqrt{1+s^2}}$ with $G \sim \mathcal{N}(\mathbf{0}; \mathbf{I})$. The parameter s is called the kernel strength and is tuned adaptively at each iteration. RW stands for "Random Walk".

MLS-SMC This is the Adaptive Multi-Level Splitting SMC used in [146]. It works on a different family of (non-smooth) distributions $\pi_\tau(dx) \triangleq \mathbb{1}_{s(x)>\tau}\pi_0(dx)$ and uses a Metropolis-Hastings kernel with uniform noise proposals. The main parameters are T and the survival rate of particles denoted ρ in [146]. Throughout the experiments, we use directly their implementation and take $\rho = 0.1$ following the authors' choice in their paper.

Note that the HMC and MALA kernels only work with smooth densities, as those defined in Sect. 5.2.2. Thus it could not be applied directly to those used in MLS-SMC.
The comparison of H-SMC and RW-SMC allows us to single out the efficiency gain of adding the gradient information in the kernel. Comparing H-SMC and MALA-SMC (which are both gradient-based methods) we can evaluate the relative performance of Hamiltonian Monte Carlo vs. Langevin Monte Carlo kernels for robustness estimation.

All variants of our method are fairly adaptive but still require the user to choose parameters α, T, and N. In practice, we noticed a good trade-off between speed and accuracy for α in the range $[0.8, 0.95]$. Throughout the experience, we let N and T vary and compare algorithms based on their accuracy and variance for a given computational cost. All experiments run on a cluster using various GPUs, hence we report the number of calls to the model and its gradient rather than compute times. We count each call to the gradient as two calls to the model function to reflect the cost of the back-propagation. Our code is available on GitHub at: https://github.com/karimtito/stat_reliability_measure.

The tendency to underestimate rare event probabilities in Monte Carlo estimation is a well-known phenomenon and shall be interpreted (when assessing the quality of algorithms) as a manifestation of variance due to a heavy tail towards larger values.

5.4.1 Toy Model

We begin our analysis by a sanity check of the algorithms on a simple problem where we know the true probability of failure. We consider a linear model $s(x) = u^\top x - \tau$, where u is a fixed unit norm vector in $\mathbb{R}^d$ and τ a bias such that the true probability is $p_\tau = \Phi(-\tau)$.

We evaluate the quality of the estimation by the Mean Relative Error (`MRE`). For an estimator run n_{run} times, this metric is given by $\texttt{MRE} = n_{run}^{-1} \sum_{i=1}^{n_{run}} |\hat{p}_c^{(i)}/p_\tau - 1|$. In this section, we used $n_{run} = 200$ for all the experiments. All the estimators are run with several numbers of particles. A bigger number provides a better estimation quality while demanding a larger number of calls.

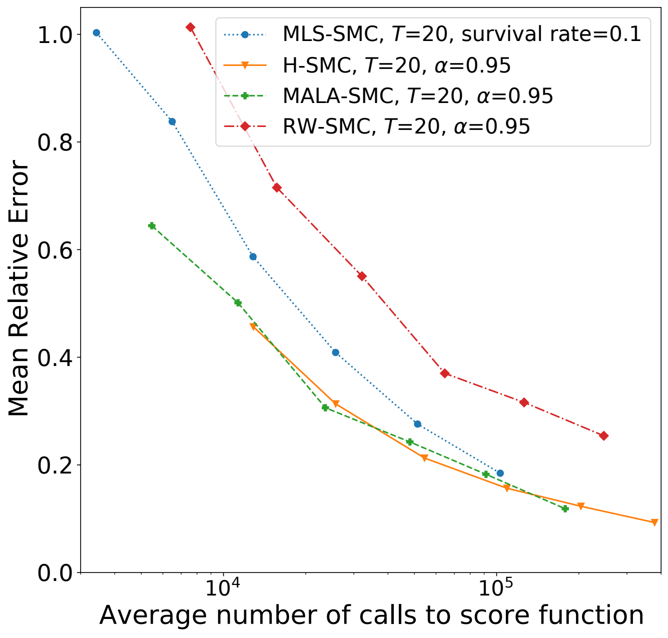


Figure 5.1 – Mean Relative Error *vs.* the average number of calls, for target probability $p_\tau = 10^{-6}$, and number of particles $N \in \{32, 64, 128, 256, 512, 1024\}$. Gradient-informed methods (H/MALA-SMC) outperform black box methods (MLS/RW-SMC) on a linear toy model.

Figure 5.1 and 5.2 compare the efficiency of the four algorithms by plotting the Mean Relative Error as a function of the average number of calls. Gradient-informed methods outperform black box methods, especially for a lower number of calls to the score function. However for the relatively easier instance of rare event simulation ($p_\tau = 10^{-6}$ on Fig. 5.1), the difference in performance is smaller and vanishes for a higher number of calls. On the opposite, for the same problem with a higher threshold ($p_\tau = 10^{-12}$ on Fig. 5.2), gradient-informed methods outperform black box methods even at a higher average number of calls. In particular, the RW-SMC algorithm displays bad performance on this harder estimation problem, which shows that integrating gradient information in the proposal kernel can highly increase the efficiency of an SMC method. Finally, on these two problems, we do not notice a real performance gap between the H-SMC and the MALA-SMC algorithms. Simply, for a given number of particles N and kernel iteration T the MALA-SMC makes fewer calls to the gradients, since it only makes one call to the gradient function per proposal.

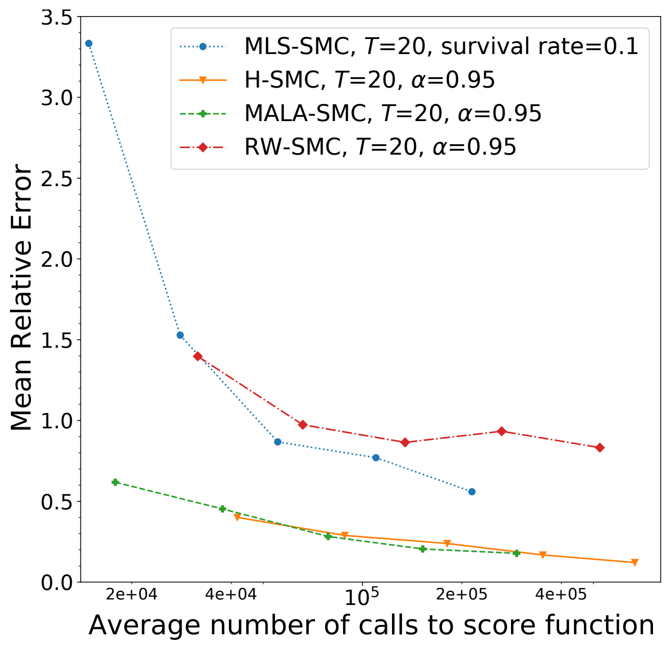


Figure 5.2 – Mean Relative Error *vs.* the average number of calls, for target probability $p_\tau = 10^{-12}$, and number of particles $N \in \{64, 128, 256, 512, 1024\}$. Gradient-informed methods (H/MALA-SMC) outperform black box methods (MLS/RW-SMC) on a linear toy model.

5.4.2 MNIST

We next compare the aforementioned methods on a robustness estimation problem for a multilayer perceptron trained on the MNIST dataset [156]. Its architecture is given in Appendix A. The input dimension is $d = 784$. We consider a uniform perturbation $E \sim \mathcal{U}([-\varepsilon, \varepsilon]^d)$ added to a correctly classified image from the test set. To map the

Gaussian distribution, presupposed by our methods, to this uniform distribution, we use the transform $\mathcal{T}_\varepsilon : x \mapsto \varepsilon(2\Phi(x) - 1)$, see Sect. 4.3.2. In contrast, the MLS-SMC does not use such a transform and works directly with the uniform distribution.

In the absence of ground truth, we obtain a reference probability by running an expensive simulation with the H-SMC algorithm taking $N = 4096, T = 200$, and 200 repetitions. We plot it on each figure (as a blue dotted line) for reference only and it is not used to compute any errors. However, it can be seen that all methods converge to this value, for T large enough, as the number of particles N increases. Figures 5.3, 5.4, 5.5 and 5.6 show the convergence of estimators for different values of T. Those were obtained with 100 runs and taking $N \in \{32, 64, 128, 256, 512, 1024\}$ (except for $T \geq 200$ in Fig. 5.3 where N only takes value in $\{8, 16\}$). Figures 5.3 and 5.4 correspond to the same value of epsilon ($\varepsilon = 0.15$), whereas Fig. 5.5 and 5.6 use a higher value ($\varepsilon = 0.18$).

Looking at figures 5.3 and 5.5, one sees that the failure probabilities estimated by MLS-SMC are very low for lower values of T. They increase steadily with T and come closer to the reference but with a higher average number of calls. In figure 5.3, MLS-SMC at last converges to the same mean log probability for $T \geq 500$ whereas the H-SMC only needs $T = 10$ and offers a much smaller standard deviation. This confirms the idea that the slow mixing of the Metropolis-Hastings kernel used by [146] makes it difficult to estimate low probabilities efficiently. On the contrary, our gradient-informed algorithms (H-SMC and MALA-SMC) seem to benefit from a better mixing kernel.

This is further highlighted in figures 5.4 and 5.6, where we compare the convergence of the RW-SMC and H/MALA-SMC. These algorithms are in fact identical except for the transition kernels proposals, which are gradient-informed only for H/MALA-SMC. Figure 5.4 shows that the RW-SMC estimates are low compared to the reference value for low values of T. Figure 5.6 corresponds to an easier estimation problem where this negative bias is less evident but gradient-informed again display noticeably lower variances for a given number of calls to the neural network.

Figures 5.4 and 5.6 also allow to compare the convergence of the MALA-SMC and H-SMC algorithms. Figure 5.4 shows that both algorithms underestimate the probability of failure for $T = 1$, with MALA-SMC having a lower bias. However both algorithms mean estimates are close to the reference probability for $T = 5$, and MALA-SMC is still slightly more efficient than H-SMC. This may seem somewhat surpising as MALA-SMC is only a constrained version of H-SMC. This might mean that our adaptive tuning of the parameter L, see section 5.3.2, is far from optimal.

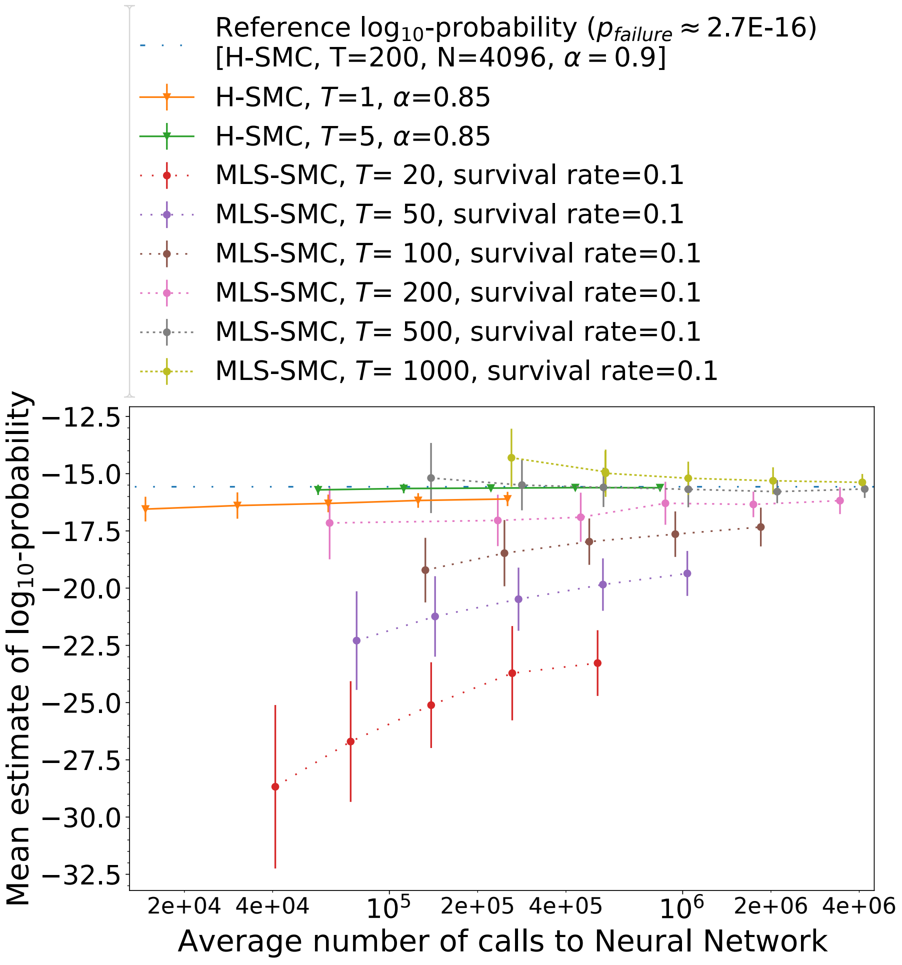


Figure 5.3 – Mean estimate of $\log_{10}(p)$ *vs* Average number of calls, comparing MLS-SMC and H-SMC on MNIST data, for uniform random noise on hyper-rectangle of radius $\varepsilon = 0.15$. Error bars show empirical standard deviations over 100 runs.

5.4.3 ImageNet

We conclude our empirical analysis with experiments on two convolutional neural networks (CNN) architectures trained on ImageNet [99], namely MobileNetV2 [173] and ResNet18 [100]. The former architecture contains 53 layers and around 3.4 million parameters, while the latter contains 18 layers and around 11 million parameters. Though rather light-weight models, they still pose an important challenge for probability of failure estimation. Note that no experiment was done over ImageNet in [146]. We consider an additive uniform noise $E \sim \mathcal{U}([-\varepsilon, \varepsilon]^d)$ on two images from the ImageNet validation dataset correctly classified repspectively by the pre-trained MobileNetV2 model and the

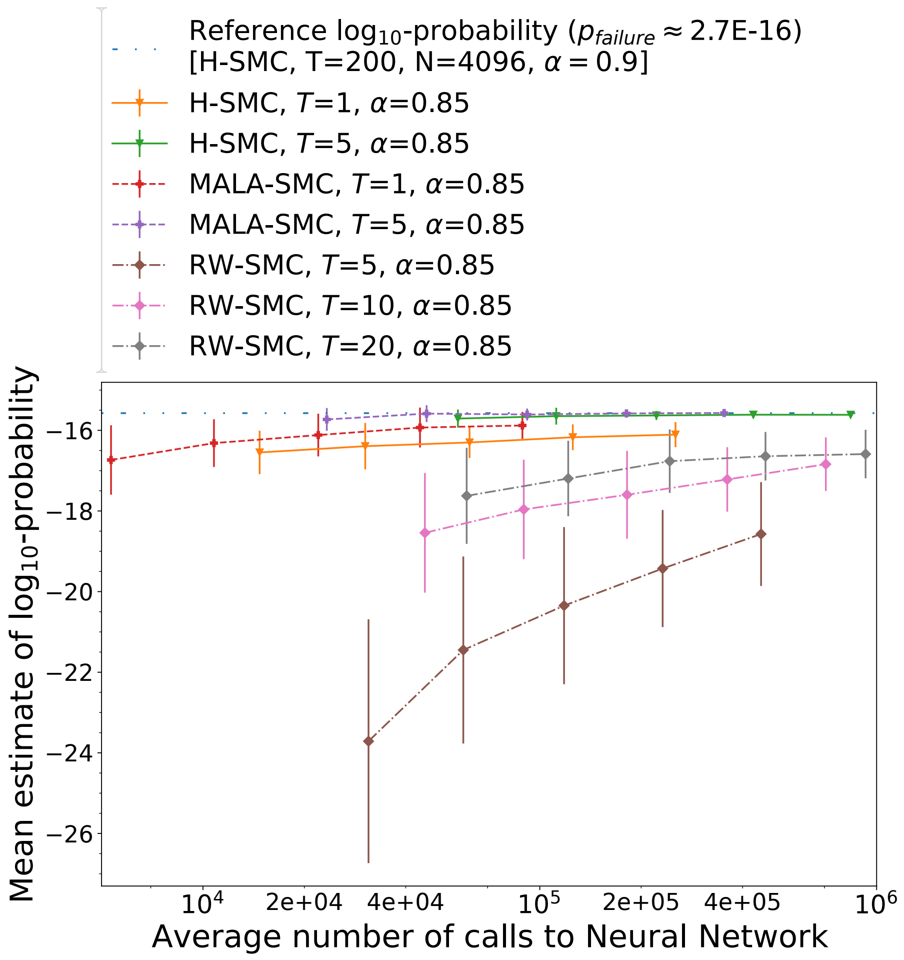


Figure 5.4 – Mean estimate of $\log_{10}(p)$ *vs* the Average number of calls, comparing RW-SMC, MALA-SMC and H-SMC on MNIST data, for uniform random noise on hyper-rectangle of radius $\varepsilon = 0.15$. Error bars show empirical standard deviations over 100 runs.

pre-trained ResNet18 model.

Figures 5.7 and 5.8 show the convergence of estimators for different values of T. Those were obtained over 50 runs and taking $N \in \{64, 128, 256, 512\}$. In figure 5.7 a value for MLS-SMC is missing from the graph for $T = 20, N = 64$ (red-dotted line): this configuration gave a mean estimation of $\log_{10}(p)$ equal to $-\infty$ (a zero probability). For the ResNet18 network the MLS-SMC gave a null estimation of the failure probability, even for large values T and N, thus it is not represented on figure 5.8. As in MNIST experiments, the MLS-SMC tends to underestimate the probability of failure, especially for low values of T. Similarly, we see on figure 5.7 that the RW-SMC clearly underestimates the probability of failure. On figure 5.8, the negative bias of the RW-SMC is less striking, however it

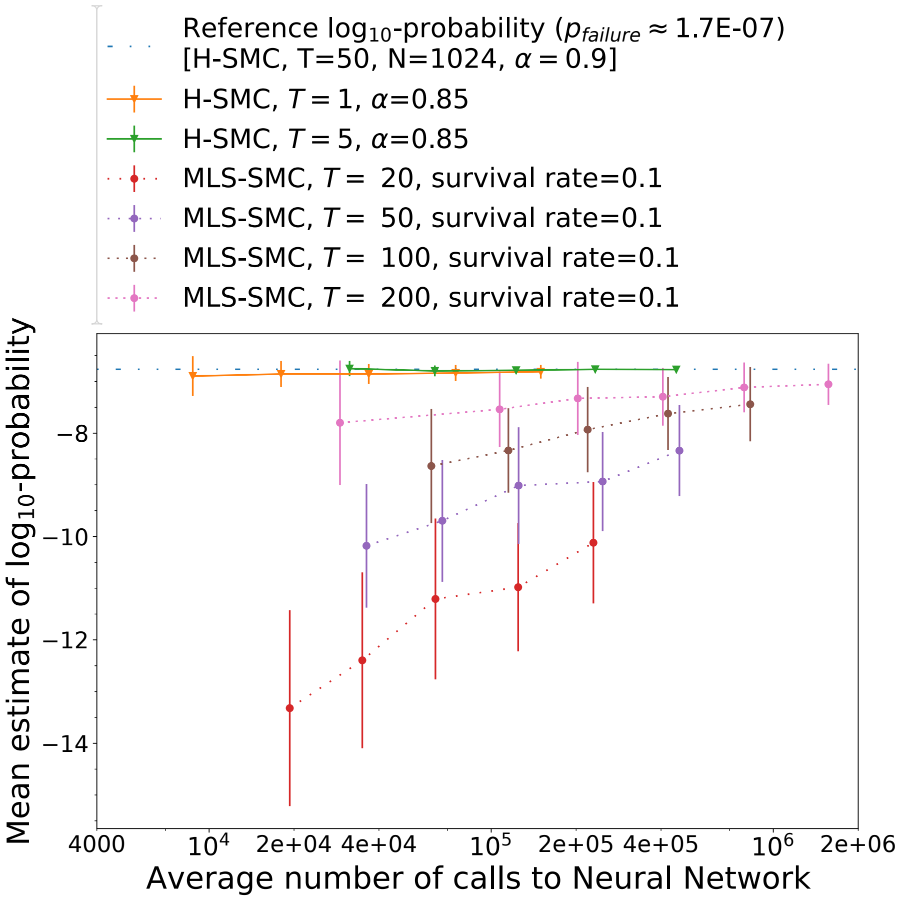


Figure 5.5 – Mean estimate of $\log_{10}(p)$ *vs* Average number of calls, comparing MLS-SMC and H-SMC on MNIST data, for uniform random noise on hyper-rectangle of radius $\varepsilon = 0.18$. Error bars show empirical standard deviations over 100 runs.

displays higher variance than the H/MALA-SMC. This highlight a lower efficiency of kernels based on Random Walk proposals (RW-SMC and MLS-SMC) in comparison to the gradient-informed kernels used in the H-SMC and MALA-SMC algorithms.

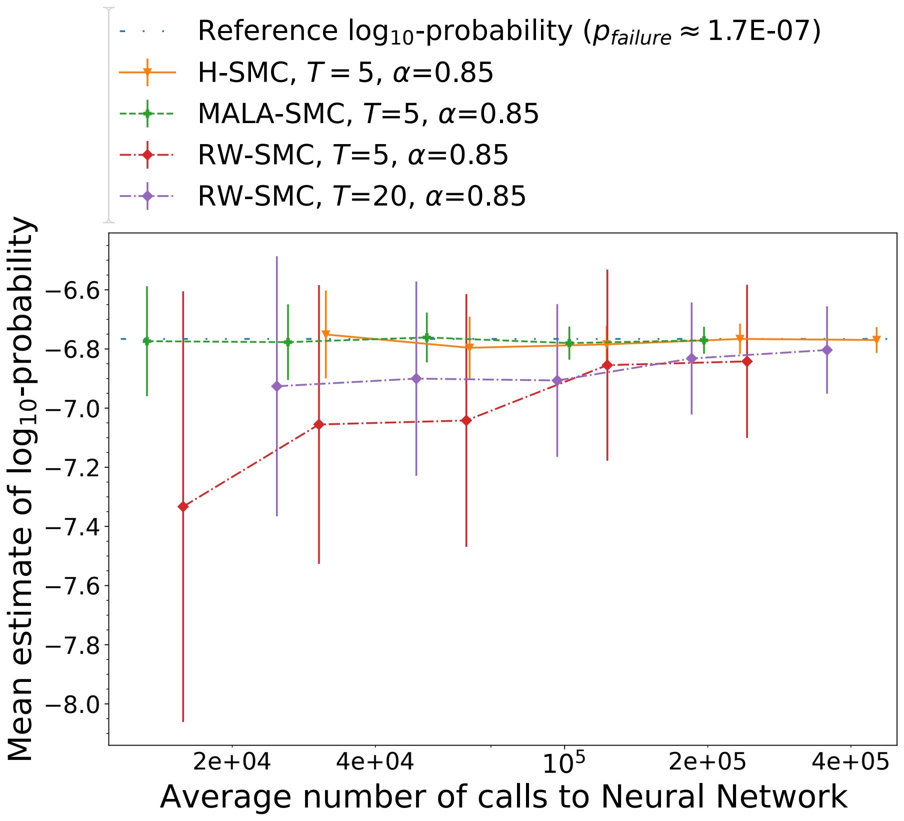


Figure 5.6 – Mean estimate of $\log_{10}(p)$ *vs* the Average number of calls, comparing RW-SMC, MALA-SMC and H-SMC on MNIST data, for uniform random noise on hyper-rectangle of radius $\varepsilon = 0.18$. Error bars show empirical standard deviations over 100 runs.

Algorithm 11 SMC Estimator of the probability of failure

```
Require: Number of particles $N$, Generator of i.i.d. reference measure samples Gen, Kernel Ker, Potential function $V$, Maximum number of iterations of the kernel $T$, Maximum number of iterations $n_{max}$, Minimum relative effective sample size $\alpha$, Minimum rare event rate $r$.
Ensure: $P_{est}$
 1: $X_0 \sim \texttt{Gen}(N)$                                   ▷ $X$ is a $[N \times d]$ array
 2: $g \leftarrow 1,\, k \leftarrow 1, \beta_1 \leftarrow \texttt{AdaptESS}(V(X_0), 0, \alpha)$
 3:                                                               ▷ see Sect. 5.3.2
 4: while $\frac{1}{N}\sum_{n=1}^{N} \mathbb{1}_{V(X_{k-1}^{(n)})=0} < r \;\;\&\;\; k \leq n_{max}$ do
 5:     $G \leftarrow \exp(-(\beta_k - \beta_{k-1})V(X_{k-1}))$
 6:     $g \leftarrow g \times \frac{1}{N}\sum_{n=1}^{N} G^{(n)}$
 7:     $X_k \leftarrow X_{k-1}$                                  ▷ Initially copying previous particles
 8:     $U \sim \mathcal{U}([0,1])^{\otimes N}$
 9:     $\mathcal{I}_{killed} \leftarrow \{i : U^{(i)} > (G^{(i)} / \max_{1\leq n \leq N} G^{(n)})\}$
10:                                                               ▷ Selection phase
11:     for $i$ in $\mathcal{I}_{killed}$ do
12:         $X_k^{(i)} \sim \mathcal{U}(\{X_{k-1}^{(j)}\}_{j \notin \mathcal{I}_{killed}})$
13:                                  ▷ Resampling particles uniformly amongst survivors
14:     end for
15:     $\texttt{StopFlag} \leftarrow \texttt{False}, t \leftarrow 1, Hist_1 \leftarrow \{X_k\}$
16:     while $t \leq T \;\;\&\;\; \texttt{StopFlag} = \texttt{False}$ do
17:         $X_k \leftarrow \texttt{BatchKer}((X_k^{(n)})_{1\leq n \leq N}, \beta_k)$
18:                                                  ▷ Mutations, see Sect. 5.3.1
19:         $Hist_{t+1} \leftarrow Hist_t \times \{X_k\}$
20:         $\texttt{StopFlag} \leftarrow \texttt{AutoCorrCond}(Hist_{t+1})$
21:         $t \leftarrow t + 1$
22:     end while
23:     $\beta_{k+1} \leftarrow \texttt{AdaptESS}(V(X_k), \beta_k, \alpha)$     ▷ see Sect. 5.3.2
24:     $k \leftarrow k + 1$
25: end while
26: $P_{est} \leftarrow g \times \frac{1}{N}\sum_{n=1}^{N} \mathbb{1}_{V(X_{k-1}^{(n)})=0}$
```

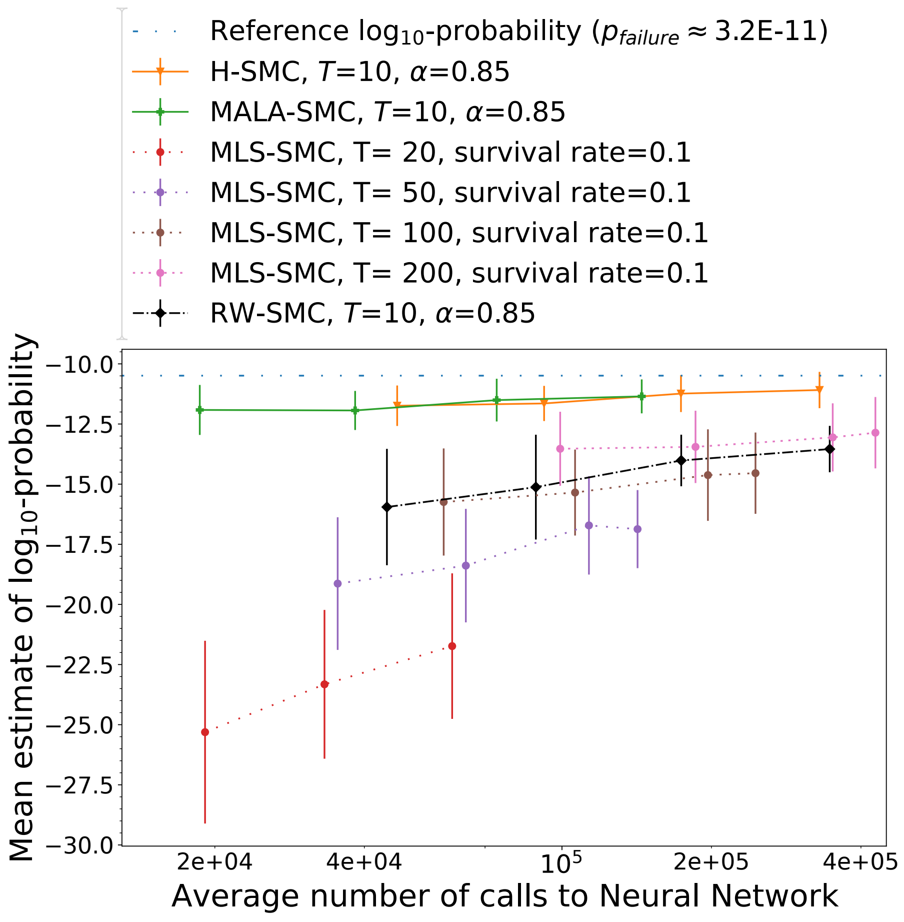


Figure 5.7 – Mean estimate of $\log_{10}(p)$ vs Average number of calls to MobileNetV2, with uniform noise over a hyper-rectangle of radius $\varepsilon = 0.13$, on ImageNet data. Error bars represent empirical standard deviation over 50 runs.

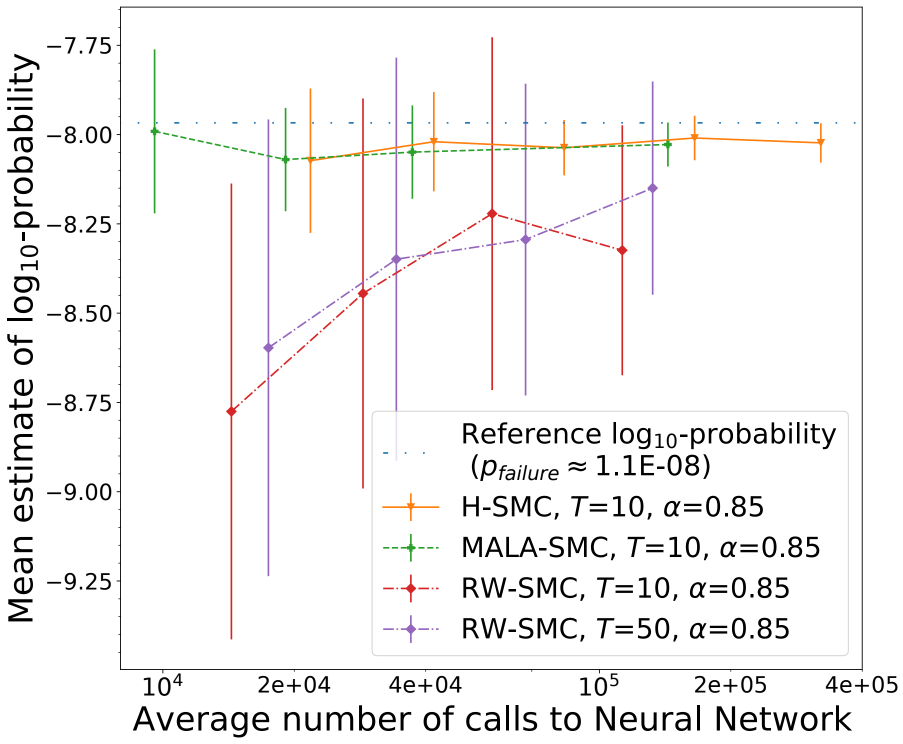


Figure 5.8 – Mean estimate of $\log_{10}(p)$ vs Average number of calls to ResNet18, with uniform noise over a hyper-rectangle of radius $\varepsilon = 0.01$, on ImageNet data. Error bars represent empirical standard deviation over 50 runs.

5.5 Limitations

Our gradient-informed method can be applied in principle to any machine learning model that is (almost everywhere) differentiable w.r.t. its input. However, unlike black-box methods, it cannot be applied to other machine learning models such as random forests and k-nearest neighbors algorithms. An other issue is that, as for the method from Webb, Rainforth, Teh, *et al.* [146], we are currently limited to asymptotic guarantees, see Prop. 5. Deriving non-asymptotic statistical guarantees, as provided in [1], is left for future work. Yet another point of contention is the use of the Effective Sample Size (ESS) metric for inverse temperature scheduling, as presented in 5.3.2. In deed, as pointed out recently by [61], this metric is based on an approximation relying on multiple assumptions which are hard to verify in practice. Therefore, developing a better, theoretically-founded, scheduling method could be a worthy direction for future research.

5.6 Societal Impact

With the development of DNN-based cyber-physical systems, a rigorous quantitative assessment of their reliability becomes of utmost importance. This paper has proposed and evaluated examples of state-of-the-art Monte Carlo methods to perform this task, especially to quickly detect unsatisfactory classifiers whose probabilities of failure are too large.

5.7 Conclusion

This paper shows how to estimate robustness to random noise in neural network classifiers more efficiently using a gradient-informed sampling technique. Our technique plugs MALA and Hamiltonian Monte Carlo methods within Sequential Monte Carlo to this rare event probability estimation problem. The performance of these gradient-informed methods is first checked on a linear toy model. We then compare these methods to a state-of-the-art black-box method for robustness estimation, and a black-box variant of our algorithm, on deep neural network models trained with MNIST and ImageNet data. We observe a faster mixing of our algorithm compared to black box methods, which in turn leads to more efficient estimation highlighted by lower variances for a given number of calls to the Neural Network functions. Future works include the investigation of novel non-asymptotic guarantees on error rates, and the extension of our methods to statistical hypothesis testing.

CHAPTER 6

Fast Reliability Estimation of Neural Networks with Adversarial Attack-Driven Importance Sampling

6.1 Introduction

In the fast-evolving landscape of Deep Learning, ensuring the robustness and reliability of Neural Networks (NNs) is paramount, particularly for critical decision-making applications. This chapter introduces a unique approach for estimating the reliability of trained Neural Networks, with a focus on their performance in the vicinity of clean inputs. We propose a method that amalgamates adversarial attacks with Importance Sampling (IS), an established technique in reliability engineering, to refine the efficiency and precision of reliability assessments in NNs.

Adversarial attacks, traditionally aimed at uncovering NN vulnerabilities, are repurposed in our methodology as a strategic guide for the IS process. This approach enables the identification of the most error-prone regions in the input space, thus directing the sampling process more efficiently than traditional Monte Carlo methods.

A key contribution of this research is the comparative analysis of our method with classical techniques from Statistical Reliability Engineering, as introduced in the chapter 1. These techniques include the First Order Reliability Method (FORM), Second Order Reliability Method (SORM), and Line Sampling (LS), which have not been extensively applied to DNNs in very high-dimensional spaces, a gap our study aims to fill.

In addition, we compare this IS estimator to other rare event simulation algorithms introduced in chapter 2. These include Cross-entropy-based Adaptive Importance Sampling (CE-AIS) and Adaptive Multilevel Splitting (AMS). We show, for various architectures and datasets, that the proposed method is more efficient and faster than these techniques.

However, this novel estimator is not without limitations. Its effectiveness is inherently tied to the efficiency of adversarial attacks; it can only be as good as the adversarial attacks it relies on. Moreover, the occurrence of weight degeneracy in extremely high-dimensional data, such as ImageNet data where $d = 150528$, restricts the applicability of this method. These constraints highlight the need for continued refinement and adaptation of the methodology, especially in dealing with large-scale, complex data structures.

In this chapter, we delve into the intricacies of integrating adversarial attack strategies within the IS framework, addressing both the algorithmic challenges and the theoretical aspects. We focus on adapting these strategies for high-dimensional reliability analysis in NNs, confronting computational and conceptual hurdles.

Through empirical studies and experiments using real-world datasets and NN models, we validate our approach. These evaluations demonstrate the method's efficacy in rapidly estimating NN reliability and shed light on failure patterns in high-dimensional models.

We conclude the chapter by discussing the broader impact of our findings, their potential applications across various domains, and directions for future research. By leveraging adversarial phenomena within neural networks, this work contributes a novel perspective to the field of reliability assessment in complex machine learning models.

6.2 Proposed Method

This paper introduces a simple yet innovative approach to speed up the reliability estimation of Neural Networks by integrating adversarial attacks into the framework of Importance Sampling (IS). This method is built upon the foundations of Statistical Reliability Engineering and especially MPFP-based Importance Sampling [154], introduced in section 2.2.2. Here, we leverage the strengths of specific adversarial attacks to construct a biased distribution for more effective sampling. The key lies in using these attacks to shift the focus of the sampling process towards regions in the input space where the NN is most vulnerable, thus allowing for a more accurate estimation of the model's reliability.

6.2.1 Adversarial Examples

Adversarial examples are considered a vulnerability of machine learning classifiers (see Sect. 3.2.1). Given an input $\mathbf{x}_0$, in $\mathbb{R}^d$, well classified by a classifier $f : \mathbb{R}^d \mapsto \mathbb{R}^C$, the optimal adversarial example is defined as the nearest misclassified input:

$$\mathbf{x}^\star = \underset{\mathbf{x}\in[0,1]^d:\mathsf{cl}(f(\mathbf{x}))\neq\mathsf{cl}(f(\mathbf{x}_0))}{\operatorname{argmin}} \mathsf{dist}(\mathbf{x},\mathbf{x}_0), \tag{6.1}$$

where $\mathsf{dist}(\mathbf{x},\mathbf{x}_0)$ is a distance between $\mathbf{x}$ and $\mathbf{x}_0$, and $\mathsf{cl}:\mathbb{R}^C\to[1:C]$ is class the operator defined as $\mathsf{cl}(\mathbf{y}) = \operatorname{argmax}_{i\in[1:C]} y_i$. In the case where the classifier is a neural network, the event $\{\mathsf{cl}(f(\mathbf{x}))\neq\mathsf{cl}(f(\mathbf{x}_0))\}$ can be rephrased as $\{h(\mathbf{x})\geq 0\}$, where h is the score function defined in section 4.2.2 of chapter 4.

If distance dist is the Euclidean norm and in the case of Gaussian noise perturbations, then the adversarial example (6.1) is indeed a design point, or MPFP, as defined in Sect 1.3.1. More generally, as in the previous chapters, we now suppose the existence of an isoprobablistic transform $\mathcal{T}$ so that it is possible to work in the U-space (see Sect 1.2.2) and define accordingly the transformed LSF function as: $G := -h\circ\mathcal{T}^{-1}$. As far as we know, this connection between adversarial examples and statistical reliability engineering has never been made before. This implies that algorithms from this later domain, like the HL-RF method designed in the 70s (see Sect. 1.3.1), could find ℓ_2 adversarial examples. This is not necessarily the case due to the high dimensionality of the input space in modern classification problems and one might expect that the recent attacks find adversarial examples more efficiently.

The Carlini and Wagner [129] (CW) attack is known for its precision in scouting adversarial examples with minimal perturbation. In the U-space, it amounts to solving the Lagrangian formulation of (1.21):

Define $J(\mathbf{u},\lambda) := \|\mathbf{u}\|^2 + \lambda G(\mathbf{u}), \forall\lambda\leq 0$ and the optimization problem:

$$\mathbf{u}^\star_\lambda := \arg\min_{\mathbf{x}\in[0,1]^d} J(\mathbf{u},\lambda). \tag{6.2}$$

This is done with a numerical solver. On top of it, a line search finds $\lambda^\star$ s.t. $G(\mathbf{u}^\star_{\lambda^\star}) = 0$. This attack requires a fair amount of function G gradient computations. Of note, we have the following property: $2\mathbf{u}^\star_\lambda + \lambda\nabla G(\mathbf{u}^\star_\lambda) = \mathbf{0}$, or:

$$\cos\left(\mathbf{u}^\star_\lambda, \nabla G(\mathbf{u}^\star_\lambda)\right) = -1. \tag{6.3}$$

The FMNA attack [131] (abbreviation for "Fast Minimum-norm Adversarial Attack"), focuses on finding the shortest path to the decision boundary, iteratively refining the input to project it onto the decision boundary. This is done by using a modified version of the

PGD attack, presented in section 3.2.1, where at each iteration the distortion budget ε is chosen adaptively. This method is much faster and almost as precise as CW.

6.2.2 Constructing the Biased Distribution

Utilizing these adversarial attacks, we construct a shifted Gaussian distribution in the U-space (standard normal space), where the mean of the distribution is adjusted based on the insights gained from the attack. This results in a biased distribution that is centered around the region of high failure probability. The steps for constructing this distribution are as follows:

Mapping to the U-Space: Design an isoprobabilistic transform, as explained in Sect. 1.2.2, that maps inputs from the X-space to the U-space. In the U-space, the uncertainties are distributed according to the standard normal law on $\mathbb{R}^d$.

Generating Adversarial Examples: Employ attacks described in Sect. 6.2.1 to find the adversarial example $\mathbf{u}^\star$ that highlights the NN's vulnerable point. Select an attack efficient in high-dimensional spaces and designed to find adversarial examples of *minimal* norm, like CW or FMNA.

Creating the Biased Distribution: Formulate a Gaussian distribution in the U-space centered around the adversarial example, ensuring that the sampling process is concentrated around the most vulnerable regions of the NN. Run the Importance Sampling procedure with $\mathbf{Y}_i \overset{i.i.d.}{\sim} \mathcal{N}(\mathbf{u}^\star, \mathbf{I})$. This means that the likelihood ratio function is given by

$$w(\mathbf{Y}_i) = \frac{\phi(\mathbf{Y}_i)}{f_Y(\mathbf{Y}_i)} = \exp\left(\|\mathbf{u}^\star\|^2/2 - \mathbf{Y}_i^\top \mathbf{u}^\star\right). \tag{6.4}$$

6.2.3 Assumptions

This method relies on the following assumptions:

A1. The design point is unique. This means that $\mathbf{u}^\star$ is a global minimum of $J(\mathbf{u}, \lambda^\star)$. If existing, local minima lie further away from the origin. This means that the probability of failure is dominated by the probability of sampling $\mathbf{U}$ around this unique design point.

A2. The attack finds this design point.

A3. The frontier locally around the design point $\mathbf{u}^\star$ is not so curved.

Once the attack produces a point $\mathbf{u}^\star$, it is easy to check that it lies on the boundary, i.e. $G(\mathbf{u}^\star) = 0$, and it is a local minimum because (6.3) holds. However, this does not prove that $\mathbf{u}^\star$ is the true global minimum. As for assumption A3, if too many random vectors $\mathbf{Y}_i$

drawn for the IS lead to $G(\mathbf{Y}_i) > 0$, i.e. are not in the rare event region, the Importance Sampling estimation will be zero or dominated by too few samples. Statisticians say that the *efficient* number of samples is too small which provokes a non-reliable estimation. In conclusion, we have means for controlling that assumption A3 holds and assumption A2 is partly fulfilled. Yet, it is impossible to ensure that A1 holds.

6.3 Experimental results

6.3.1 Experimental Setup

We compare the convergence of several Rare Event Simulation methods: our Adversarial-Attack Driven IS of Sect 6.2 (which we abbreviate by ADV-IS), the Line Sampling (LS) estimator (see Sect. 1.3.4), the Cross-Entropy Importance Sampling (CE-IS) (see Sect. 2.3.2) using the Gaussian parametric family $\mathcal{Q}^{\text{U-Gauss}}$ defined in Sect.2.3.1, and two estimators based on Sequential Monte Carlo (SMC) techniques, the Multilevel Splitting [63] and the Langevin Monte Carlo within an SMC scheme [174] introduced in the previous chapter, that we note respectively MLS-SMC and MALA-SMC (MALA stands for Metropolized Langevin Algorithm). As we saw in the previous chapter, an important parameter for these SMC methods, in addition to the number of samples N, is the number T of applications of a transition kernel, which reduces the dependence between samples. Theoretical guarantees are derived under the perfect independence ($T = \infty$). In practice, $T < \infty$ has a huge impact on the number of calls to the NN.

We consider three models across two datasets and apply uniform noise to different instances. For each instance, we compute a reference probability of failure $\hat{P}_{\text{F}}^{\text{Ref}}$ by using an expensive IS compute using the FMNA search method mentioned above and taking N of the order 10^6. Crucially, we check a posteriori that all methods converge towards the same value. Still, this reference probability is not a ground truth, and therefore we also report metrics independent of its value, as explained below. In addition to benchmarking the rare event simulation methods, we compute both the FORM estimate $P_{\text{F}}^{\text{FORM}}$ and, whenever possible, the SORM estimate $P_{\text{F}}^{\text{SORM}}$, as defined above, using different search methods. These estimators are quantitively compared thanks to two metrics:

- The coefficient of variation $\Delta[\cdot]$, defined for an estimator $\hat{P}_{\text{F}}$ as, $\Delta[\hat{P}_{\text{F}}] = \frac{\sqrt{\mathbb{V}[\hat{P}_{\text{F}}]}}{\mathbb{E}[\hat{P}_{\text{F}}]}$.
- The relative mean absolute error, note RE$[\cdot]$, define as: $\text{RE}[\hat{P}_{\text{F}}] = \mathbb{E}[|P_{\text{F}} - \hat{P}_{\text{F}}|] \cdot P_{\text{F}}^{-1}$.

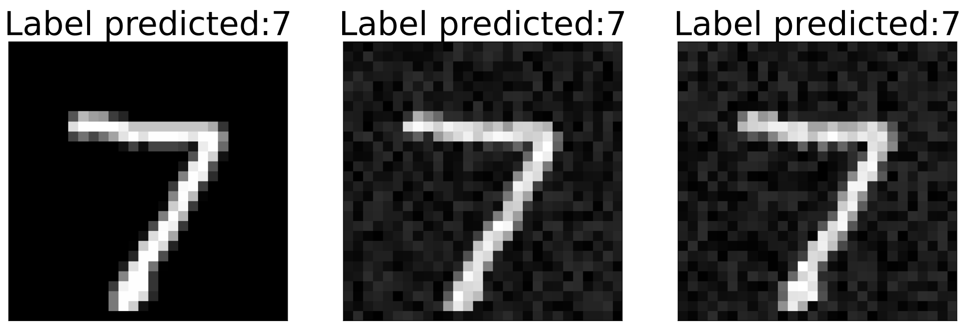


Figure 6.1 – Input $\mathbf{x}_{0,1}$ (on the left) and examples of perturbations with uniform noise $\varepsilon = 0.18$.

In practice, we have to estimate these metrics by their empirical counterpart. Moreover, as RE explicitly involves the failure probability, we will use the reference probability $\hat{P}_{\mathrm{F}}^{\mathrm{Ref}}$ as a surrogate. Crucially, for a fair comparison, these metrics and the complexity of an estimator (gauged by the number of calls) are measured over the same runs. All experiments were run on a personal laptop, with a 4060RTX GPU.

6.3.2 MNIST

MLP with two hidden layers

We first compare these methods via experiments on a simple Multi-Layer Perceptron (MLP) with only 2 hidden layers (each containing 200 neurons) trained on the MNIST dataset [156], which will be referred to as model M_1, and on a first instance we note $\mathbf{x}_{0,1}$. This dataset consists of 28×28, black-and-white, images of hand-written digits. We consider additive noise perturbations, uniform on the ℓ_∞ ball of radius $\varepsilon = 0.18$ and centered on the input $\mathbf{x}_{0,1}$, see Figure 6.3. This distribution can be mapped to the standard Gaussian law via an isoprobabilistic transform. At this level of noise, the probability of misclassification is low. Running an expensive simulation we find that $\hat{P}_{\mathrm{F}}^{\mathrm{Ref}} \approx 1.95 \cdot 10^{-6}$.

We apply the FORM and SORM methods with three adversarial attacks, the Carlini-Wagner attack, FMNA attack, and HLRF attack. Indeed, the dimension is $d = 784$ for this dataset and it is possible to manipulate matrices of size $d \times d$ and in particular to evaluate, via auto-differentiation, the Hessian of G. Table 6.1 presents the results. At a glance, it is clear that FORM significantly overestimates the probability of failure when the FMNA and HLRF attacks approximately locate the MPFP, but underestimates it with the CW attack. This indicates that the decision boundary at $\mathbf{u}^*$ is not "flat" enough

Table 6.1 – FORM/SORM estimations of $\hat{P}_{\mathrm{F}}^{\mathrm{Ref}} \approx 1.95 \cdot 10^{-6}$ for model M_1 and input $\mathbf{x}_{0,1}$, with uniform noise ($\varepsilon = 0.18$).

Attack	$P_{\mathrm{F}}^{\mathrm{FORM}}$	$P_{\mathrm{F}}^{\mathrm{SORM}}$	$\cos(\tilde{u}^*, \nabla G(\tilde{u}^*))$
CW	$7.2 \cdot 10^{-8}$	$6.39 \cdot 10^{-6}$	-0.69
FMNA	$1.17 \cdot 10^{-4}$	$6.49 \cdot 10^{-6}$	-0.995
HLRF	$7.53 \cdot 10^{-5}$	$6.65 \cdot 10^{-6}$	-0.977
	$\lVert\tilde{u}^*\rVert_2$	$G(\tilde{u}^*)$	Time (in sec.)
CW	5.26	$-4.1 \cdot 10^{-5}$	0.19
FMNA	3.68	$-1.4 \cdot 10^{-5}$	0.16
HLRF	3.79	$-2.0 \cdot 10^{-2}$	0.01

for a linear approximation to hold. This idea is further reinforced by observing that the SORM estimators are indeed closer to the actual probability of failure. In addition, we note that, here, the CW attack performed poorly, as its norm is higher in comparison with that of the two other attacks. Moreover, the Hessian $\nabla^2 h = -\nabla^2 G$ has both positive and negative eigenvalues at the CW point, whereas it only has non-positive eigenvalues at the other attack points. We ran all these search algorithms with a limited budget of 100 iterations, thus the failure of the CW attack should be interpreted in this light.

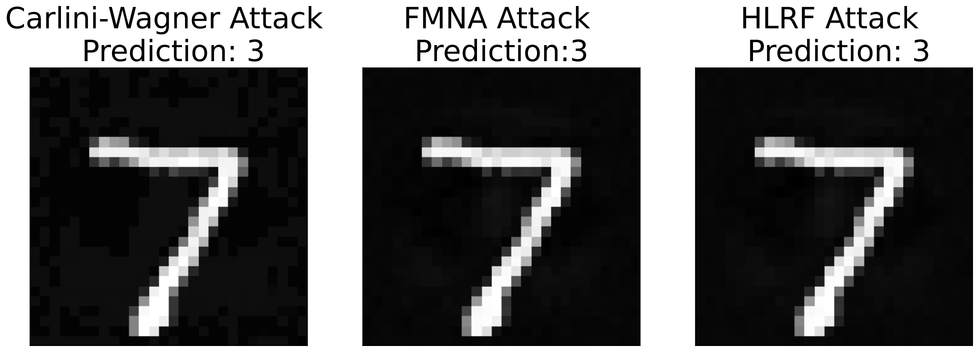


Figure 6.2 – Adversarial attacks for model M_1 on input $\mathbf{x}_{0,1}$.

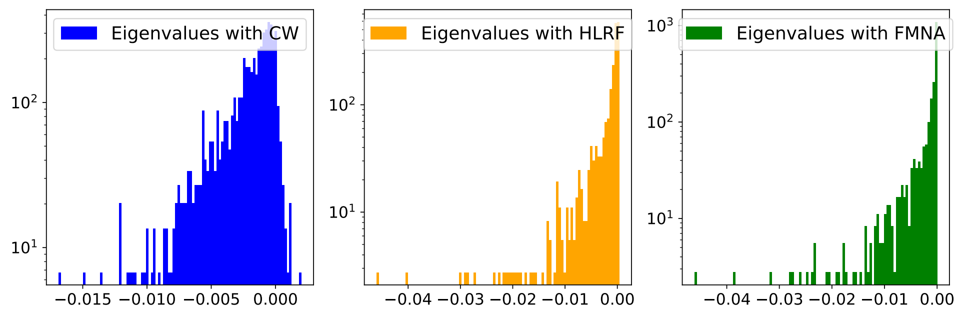


Figure 6.3 – Eigenvalues of the Hessian of score function h at the CW attack (on the left), at the FMNA attack (in the center), and the HLRF attack (on the right).

We next, look at the convergence of the statistical methods w.r.t. the average number of calls, noted $\bar{N}_{\text{calls}}$. In Figure 6.4 we see that all methods seem to converge towards the reference probability as the average number of calls increases, though their convergence rate differs. In particular, the Sequential Monte Carlo methods, MALA-SMC and MLS-SMC, converge noticeably slower than the LS and ADV-IS methods. The cross-entropy (CE) AIS method has a significant overhead as it must first converge towards a good parameter $\boldsymbol{\theta}$ in the parameter space Θ, as explained in Sect. 2.3.2, before exploiting its final distribution to estimate P_{F}. We focus on the IS and LS methods in Figure 6.5, comparing their speed of convergence for different adversarial attacks. These figures are obtained by: running each method 400 times (with different random seeds to obtain standard errors) using a given number of samples N and repeating the same operation for increasing values of N. For example, we ran the ADV-IS for values of N in the range $\{100, 1000, 10000, 50000, 100000\}$.

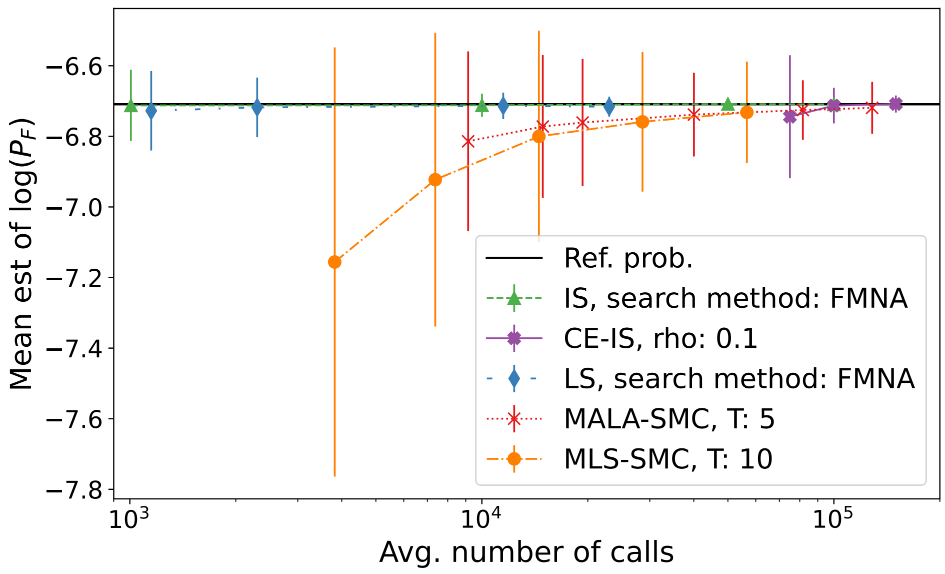


Figure 6.4 – Convergence of different estimators w.r.t. the number of calls to the model M_1.

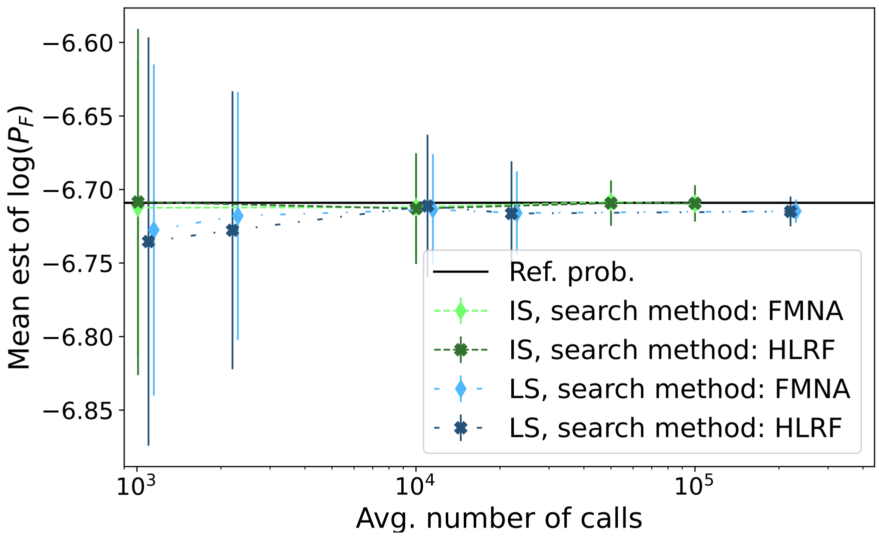


Figure 6.5 – Convergence of IS and LS with different attacks.

Finally, we give the best performance of each algorithm (w.r.t. the number of samples used) in terms of the coefficient of variation multiplied by a measure of the computational burden. In practice, we use either the number of calls to the model $\bar{N}_{\text{calls}}$ (i.e. the metric

Table 6.2 – Best performance of estimators of P_{F} for the model M_1 and input $\mathbf{x}_{0,1}$, with uniform noise ($\varepsilon = 0.18$).

Method	N_{best}	time (sec.)	$\text{RE}[\hat{P}_{\text{F}}]$
ADV-IS	$5 \cdot 10^4$	$5 \cdot 10^{-2}$	$2.5 \cdot 10^{-2}$
CE-IS	$3 \cdot 10^4$	$2.3 \cdot 10^{-1}$	$4.3 \cdot 10^{-2}$
LS	50	$4.3 \cdot 10^{-2}$	$2.1 \cdot 10^{-1}$
MALA	256	$2.0 \cdot 10^{-1}$	$2.1 \cdot 10^{-1}$
MLS	1024	$2.5 \cdot 10^{-2}$	$2.6 \cdot 10^{-1}$
	$\hat{\Delta}^2[\hat{P}_{\text{F}}] \times \bar{N}_{\text{calls}}$	$\hat{\Delta}^2[\hat{P}_{\text{F}}] \times \text{time}$	$\bar{N}_{\text{calls}}$
ADV-IS	48	$4.8 \cdot 10^{-5}$	$5 \cdot 10^4$
CE-IS	460	$7 \cdot 10^{-4}$	$1.5 \cdot 10^5$
LS	77	$2.9 \cdot 10^{-3}$	1200
MALA	3000	$1.5 \cdot 10^{-2}$	$4 \cdot 10^4$
MLS	6200	$2.7 \cdot 10^{-3}$	$5.7 \cdot 10^4$

$\hat{\Delta}^2[\hat{P}_{\text{F}}] \times \bar{N}_{\text{calls}}$), or the duration of the simulation in seconds (i.e. the metric $\hat{\Delta}^2[\hat{P}_{\text{F}}] \times \text{time}$). Table 6.2 reports the results where N_{best} denotes the number of samples that gave the best performance in terms of the metric $\hat{\Delta}^2[\hat{P}_{\text{F}}] \times \bar{N}_{\text{calls}}$. All metrics reported in this table pertain to the ADV-IS method outperforms all other methods, for both metrics mentioned above. The CE-IS method also obtains good performance, for a relatively low number of samples N_{best} used for estimation. However, the *total* number of calls needed for CE-IS is in the order of *hundreds of thousands.*

MLP with four hidden layers

We now consider a similar MLP architecture with four hidden layers (each hidden layer containing 200 neurons), denoted M_2. Simulation results for the FORM and SORM algorithms are given in the Appendix. Overall, these results support the idea that the decision boundaries of neural networks do not appear to be (locally) flat enough to be accurately approximated by hyperplanes, as the FORM method tends to overestimate the probability by an order of 10 or more. In contrast, the SORM method shows promising results, with the caveat that it systematically underestimates the probability of failure, which can be problematic when considering safety-critical applications. Focusing now on statistical estimators, we study their empirical convergence, for two images $\mathbf{x}_{0,1}$ and $\mathbf{x}_{0,2}$, with similar perturbations as in the previous section, i.e. uniform noise on ℓ_∞ balls of radius $\varepsilon = 0.18$. Simulation results are reported in Figure 6.7.

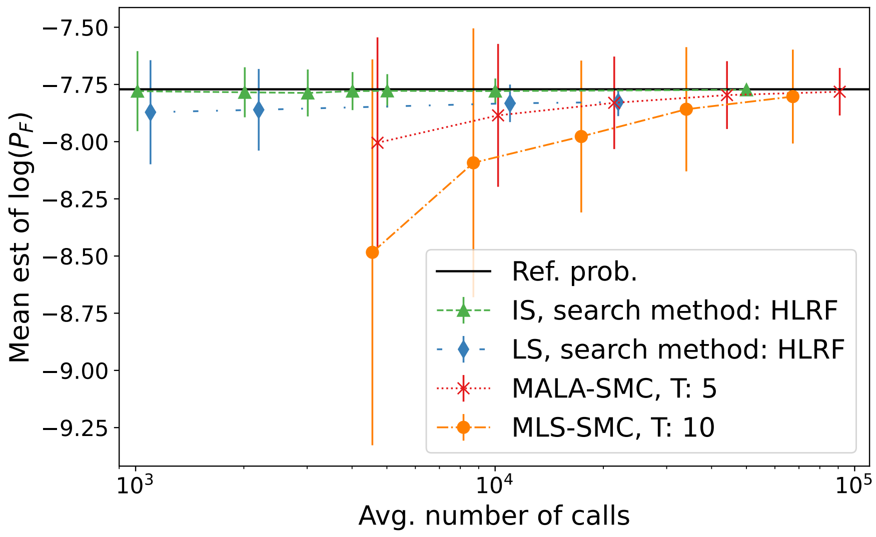


Figure 6.6 – Convergence of the estimators w.r.t. the number of calls to the model M_2, on the input $\mathbf{x}_{0,2}$

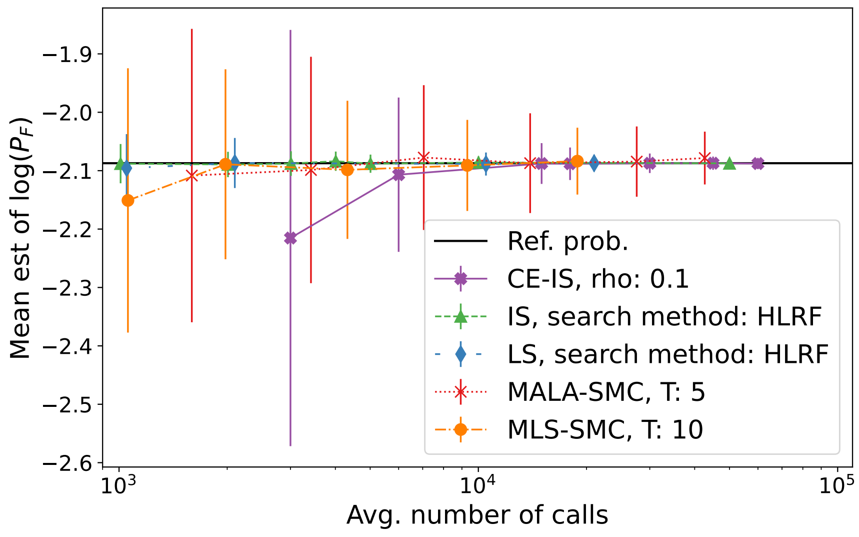


Figure 6.7 – Convergence of different estimators w.r.t. the number of calls to the model M_2, on the input $\mathbf{x}_{0,3}$

Like in previous experiments, the SMC-based algorithms converge much slower than both LS and the adversarial-attack-driven IS algorithm, though the gap is slightly less important in the case of input $\mathbf{x}_{0,3}$, which has a higher probability of failure, leading in

Table 6.3 – FORM/SORM estimations of $P_{\mathrm{F}} \approx 2.4 \cdot 10^{-7}$ for the custom CNN model, with uniform noise ($\varepsilon = 0.03$).

Attack	$P_{\mathrm{F}}^{\mathrm{FORM}}$	$P_{\mathrm{F}}^{\mathrm{SORM}}$	$\cos(\tilde{u}^*, \nabla G(\tilde{u}^*))$
CW	$3.91 \cdot 10^{-5}$	NA	-0.97
FMNA	$5.22 \cdot 10^{-5}$	NA	-0.985
HLRF	$2.16 \cdot 10^{-5}$	NA	-0.965
	$\|\tilde{u}^*\|_2$	$G(\tilde{u}^*)$	Time (in sec.)
CW	3.95	$-1.2 \cdot 10^{-4}$	1.49
FMNA	3.88	$-8.0 \cdot 10^{-5}$	0.23
HLRF	4.09	$-8.1 \cdot 10^{-2}$	0.03

particular to less dramatic underestimation of the MLS algorithm when using a smaller number of samples. Interestingly, in this example, the MLS algorithm, which is a black-box method, seems to slightly outperform the MALA-SMC algorithm, introduced in the last chapter, that uses gradient information [174]. This can be explained again by the relatively high probability of failure, as $P_{\mathrm{F}} \approx 10^{-2}$.

6.3.3 CIFAR10

We move on to the CIFAR10 dataset, which is more challenging for rare event simulation as the dimension of each input is $d = 32^2 \times 3 = 3072$. We run experiments on a custom convolutional neural network, which contains four convolutional layers, followed by two dense layers and contains in total of 476 278 scalar parameters.

As before, we applied the FORM algorithm using different adversarial attacks, and the associated results are reported in Table 6.3. However, it is not possible to apply the SORM algorithm, as it requires too much memory capacity and computing power.

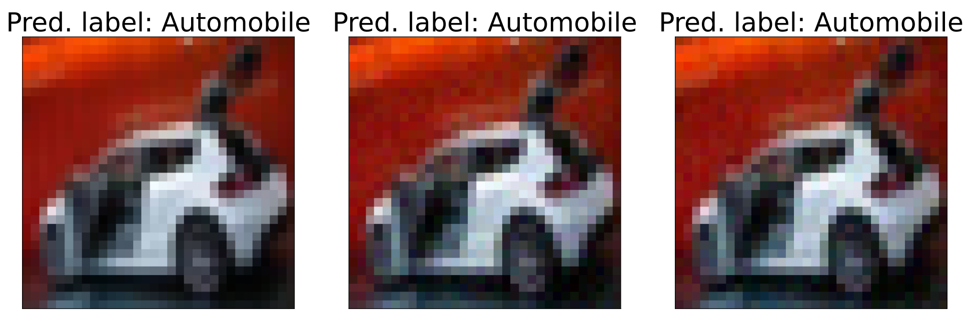


Figure 6.8 – Clean input of the CIFAR10 dataset (on the left) and copies perturbed with Gaussian noise ($\sigma = 0.02$).

We next focus on the simulation algorithms' performance. Again, we primarily compare the LS and adversarial-attack-driven IS algorithm to sequential Monte Carlo methods used in the literature [146], [174]. The associated results are reported in Figure 6.9 below.

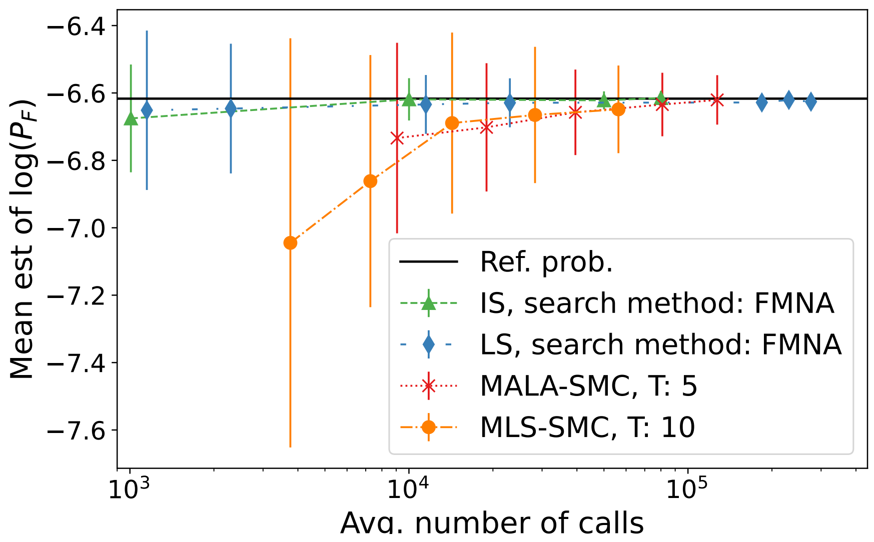


Figure 6.9 – Convergence of different estimators w.r.t. the number of calls to the CNN.

We obtain similar results to that obtained for MNIST data: Our method and Line Sampling converge in a few thousand calls whereas state-of-the-art SMC algorithms require a few *hundreads* thousands of calls to obtain similar standard errors. That being said, the performance gap is somewhat smaller, a fact we attribute to the curse of dimension (COD), leading to weight degeneracy in Importance Sampling [57].

Figure 6.10 compares the performance of the adversarial attacks. We notice again very small differences in terms of performance for the FMNA and HLRF algorithms. This means that the HLRF algorithm we have implemented for Neural Networks proves to be a powerful adversarial attack.

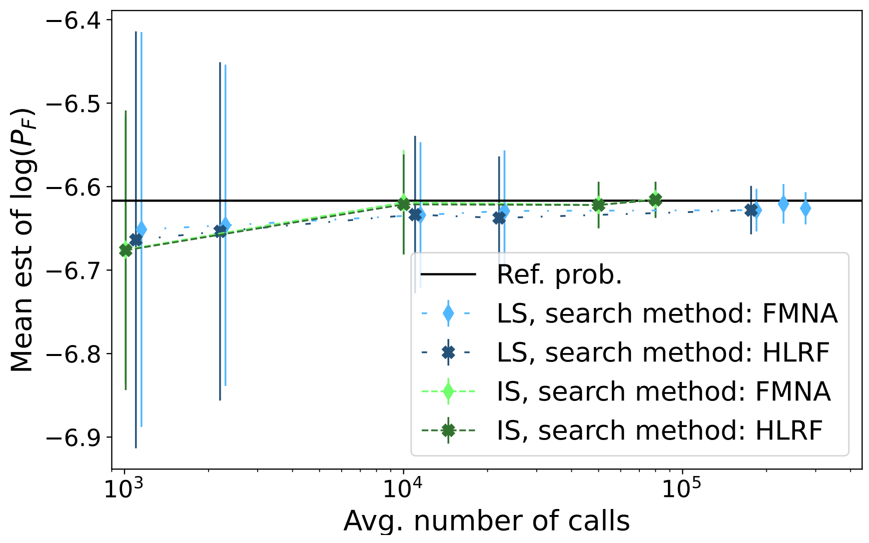


Figure 6.10 – Convergence of different estimators w.r.t. the number of calls to the CNN.

6.3.4 ImageNet Results

Finally, we conclude this section with experimental results obtained on the ImageNet [99] dataset, where $d = 224^2 \times 3 = 150528$. We test the probabilistic robustness of a pre-trained ResNet-18 model [100] under uniform noise of size $\varepsilon = 0.055$, around a clean image. Figure 6.11 illustrates the convergence of ADV-IS, MALA, and MLS estimation methods. In contrast to previous experiments, we see that the convergence rate of ADV-IS is worse than SMC-based methods. We attribute this poor performance to the high dimension of the problem, leading to catastrophic weight degeneracy, as mentioned above. In this case, it seems that SMC methods are more reliable than the proposed adversarial attack-based Importance Sampling. Thus, proposing a method that is both highly efficient for moderately high-dimensional data and reliable even for very high-dimensional data remains an important direction for future research in probabilistic robustness assessment.

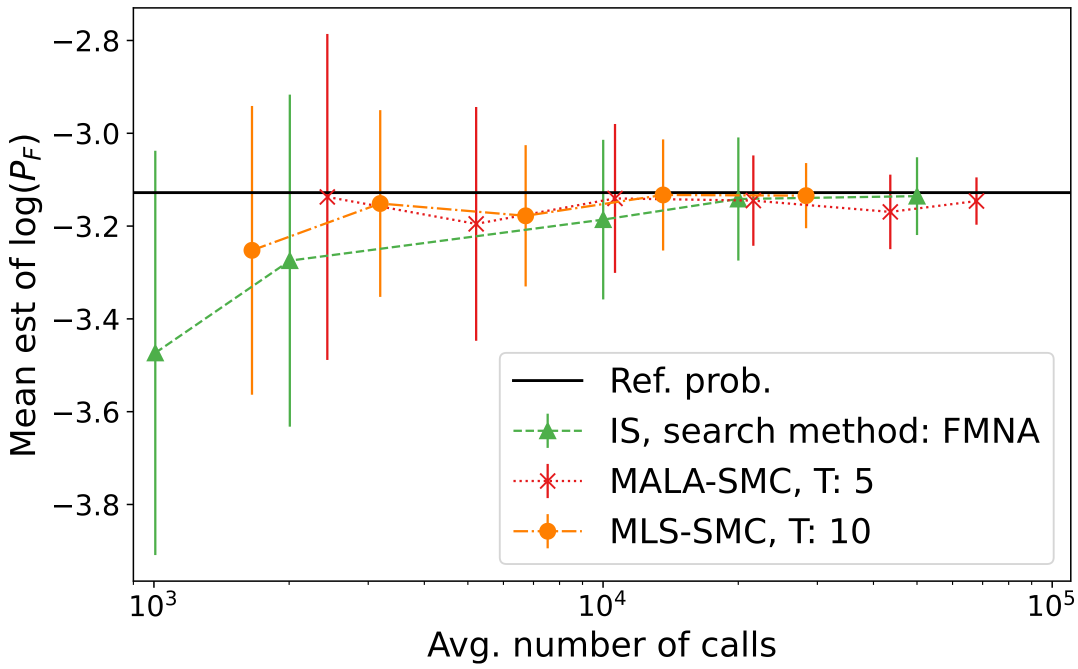


Figure 6.11 – Convergence of different estimators w.r.t. the number of calls to a ResNet-18 model pre-trained on ImageNet.

6.4 Conclusion

In conclusion, through extensive empirical analysis, we showed that the proposed algorithm outperforms, in terms of speed and computational efficiency, state-of-the-art methods for Neural Network reliability assessment, for moderately high dimensional datasets such as MNIST and CIFAR10. However, as mentioned above, a crucial limitation of our approach, in comparison with the sequential Monte Carlo approach, is the inability to handle very high-dimensional data. Indeed, while the algorithm we introduced in the previous chapter is slower, we showed that it still can efficiently estimate probabilities of failure on the ImageNet dataset. This limitation of the proposed approach is directly linked to the phenomenon of weight degeneracy, which becomes very difficult to handle when the problem dimension, d, is of the order of hundreds of thousands or more. Developing a hybrid approach between this method and splitting techniques, which has been done for another type of reliability problem [175], is a promising avenue for future research.

Final Conclusion and Perspectives

To sum up, after reviewing the necessary concepts and methods from the fields of Statistical Reliability Engineering, Rare Event Simulation, and Deep Learning Robustness, we have presented three original contributions dedicated to the estimation of the probability of misclassification of a Deep Neural Network around a clean input. We next recall the essential results of our work and the limitations that are inherent to the methodologies we have proposed. Finally, we conclude this text with a short tour of promising avenues for future research in the field of Deep Learning reliability and robustness.

Results and Limitations

The primary goal of our thesis was to explore the potential of rare event simulation methodologies as applied to Deep Learning models. We have first shown how to use the Last Particle algorithm to emulate DNN certification with a statistical hypothesis testing approach. Even though our numerical experiments can be convincing, a more thorough analysis is still necessary to obtain reliable bounds on the probability of false positives. Indeed, the statistical guarantees we provided, while non-asymptotic with respect to the number of samples, are only applicable as such to an idealized version of the algorithm, as described in the work of [86] and more succinctly in chapter 4.

From another angle, the algorithm described in chapter 4 constitutes a 'black-box' methodology, meaning that it uses direct outputs (typically the logits) of the neural network. This can be advantageous as it can be applied to various classifiers, with the caveat of requiring that their primary output be a continuous function of the inputs, as opposed to a hard label (e.g. in binary classification it should output a score between 0, and 1, instead of directly outputting a label prediction). Indeed, we presented applications of this method to various classification algorithms, including Random Forests and Gradient Boosting, in a work we presented orally at the Conference on Artificial Intelligence for Defense [176], in 2021. However, in the specific case of Neural Networks, gradients of the model are readily accessible thanks to back-propagation. Therefore, the work we introduced in chapter 5 came as a natural extension of the work that began in the paper of

Webb, Rainforth, Teh, *et al.* [146], where they used random walks kernels instead. Indeed, we showed that this modified version of the Splitting algorithm was more efficient in most cases, though it does require defining new target densities, similar to what is done in Improved CE [72], in the context of Cross-Entropy based IS (see section 2.3.2). Again, a crucial issue is the lack of non-asymptotic statistical guarantees, as in this case, we only have asymptotic guarantees, valid when the number of samples, N, goes to $+\infty$.

Finally, while these works improve on the state-of-the-art for DNNs reliability estimation, they ignored important methodologies from the field of statistical reliability engineering. Our latest work, presented in chapter 6, remedies this shortcoming, and introduces a new methodology using elements from both adversarial robustness and the field of structural reliability. We also applied FORM, SORM, and CE-AIS to the DNN's reliability estimation problems, which had not been done yet in the deep learning literature. The proposed method proved to be more efficient than splitting methods for moderately high-dimensional data. However, we were not able to reproduce these results in very high-dimension, as exemplified by ImageNet. We link this failure to the phenomenon of weight degeneracy, as was studied in the paper of Li, Bengtsson, and Bickel [57]. Proposing an improvement of this method for such high-dimensional data, perhaps by combining it with an SMC-based method, is an important direction for future research.

Future Directions

Building on this foundation, future research can take multiple trajectories. We next focus on three of those:

1. Broadening reliability assessment to include other models of Deep Learning. In particular, Reinforcement Learning [177], as the framework is especially relevant for critical systems, such as autonomous vehicles and it has an inherently non-stationary structure. Time-related issues were inexistent in this thesis, as we only studied static models. However, both reliability engineering and rare event simulation methods introduced in this thesis have natural extensions to dynamic models.
2. Advancing rare event simulation: bridging the gap between IS and the importance of Splitting methods, eventually by combining these approaches, e.g., as done for dynamic models in the work of Jacquemart-Tomi, Morio, and Le Gland [175].
3. Applying these methodologies in real-world contexts, particularly in critical domains like autonomous driving and medical diagnostics, to enhance safety and reliability.

Reliability Assessments in Reinforcement Learning

The exploration of reliability in the context of Reinforcement Learning (RL) represents an exciting frontier for future research. RL systems, particularly those deployed in dynamic and potentially hazardous environments, demand rigorous reliability assessments [178]. The challenge lies in the inherent variability and non-stationarity of RL environments, which complicates the application of traditional reliability assessment methods. Future work should focus on adapting and extending the methodologies developed for static models to accommodate the temporal complexities of RL. This may involve the development of novel simulation techniques or the application of sequential decision-making frameworks to model the evolving state of RL systems accurately. Addressing these challenges will be critical for ensuring the safety and dependability of RL applications in fields such as autonomous navigation and interactive robotics.

Bridging the Gap between Importance Sampling and Splitting

The dichotomy between Importance Sampling (IS) and Splitting methods presents another promising area for future exploration. While each approach has its strengths, a hybrid methodology that leverages the advantages of both could significantly enhance the efficiency and accuracy of rare event simulations. The integration of IS's ability to focus computational resources on high-probability failure regions with Splitting's structured exploration of the state space offers a compelling strategy for tackling complex reliability problems. Future research should investigate algorithmic frameworks that combine these methods, drawing inspiration from successful hybridizations in dynamic models [175]. Such advancements could lead to breakthroughs in the simulation of rare events, with broad applications across engineering and science.

Applications in the Real-World

The ultimate test of any theoretical advancement lies in its applicability to real-world problems. The methodologies developed in this thesis hold considerable potential for improving the safety and reliability of deep learning-based critical systems in domains such as autonomous driving [179] and medical diagnostics [180]. Future research should focus on implementing and validating these methods in practical settings, working closely with industry partners to tailor the approaches to specific challenges. For instance, the integration of reliability assessments into the design and testing phases of autonomous vehicle development could significantly reduce the risk of system failures. Similarly, in medical diagnostics, enhancing the reliability of Deep Learning models could lead to more accurate and trustworthy decision-making tools. By bridging the gap between theoretical research and practical application, future efforts can contribute to the creation of safer, more reliable technology that benefits society as a whole.

In conclusion, the research presented in this thesis lays the groundwork for a wide range of future investigations into the reliability of Deep Learning models. As the field continues to evolve, it will be imperative to explore these and other directions to ensure the development of robust and trustworthy AI systems.

List of Publications

First-author Publications

[3] - Fast Neural Networks Reliability Estimation with Attack-Driven Importance Sampling.
K.T., T. Furon- 40th Conference on Uncertainty in Artificial Intelligence (UAI), Jul 2024, Barcelona, Spain.

[2] - Gradient-Informed Neural Network Statistical Robustness Estimation.
K.T., T. Furon, M. Rousset. - 26th International Conference on Artificial Intelligence and Statistics (AISTATS), Apr 2023, Valencia, Spain.

[1] - Efficient Statistical Assessment of Neural Network Corruption Robustness.
K.T., T. Furon, M. Rousset. - 35th Conference on Neural Information Processing Systems, Dec 2021, Sydney (virtual), Australia.

[176] - Évaluation statistique efficace de la robustesse de classifieurs.
K.T., T. Furon, M. Rousset, L.-M. Traonouez. - Conference on Artificial Intelligence for Defense, DGA, DGNUM, Ministère des Armées, Nov 2021, Rennes, France.

Other Publications

- Three Bricks to Consolidate Watermarks for Large Language Models.
P. Fernandez, A. Chaffin, K.T., V. Chappelier, T. Furon- IEEE 15th International Workshop on Information Forensics and Security, Dec 2023, Nuremberg, Germany.

List of Figures

List of Tables

APPENDIX A

Non-Uniform Random Variables Generation

This appendix provides methodologies for generating non-uniform random variables, a fundamental task in simulation and probabilistic modeling. We first discuss three prevalent methods: Inverse Transform Sampling, Rejection Sampling, and Monte Carlo Markov Chains (MCMC), each catering to different scenarios and distribution characteristics. Whereas Inverse Transform and Rejection Sampling are exact methods, MCMC is only approximate although it can come with asymptotic convergence guarantees.

A.1 Inverse Transform Sampling

Inverse Transform Sampling is an effective technique for generating samples of a random variable or vector, particularly when the components of the random vector are independent and each of their individual cdfs are known.

Given a random vector $\mathbf{Z} = (Z_1, Z_2, \ldots, Z_d)$ taking values in $\mathbb{R}^d$, assume that the components Z_i are independent with known cdfs $F_{Z_i}(z_i)$. The aim is to simulate random variables that follow the distribution of $\mathbf{Z}$.

The Inverse Transform Sampling procedure can be broken down as follows:

1. Generate d independent uniform random samples $V_1, V_2, \ldots, V_d$ in the interval [0, 1].
2. For each component $i \in [1 : d]$, compute the inverse of the corresponding cdf to transform the uniform sample: $Z_i = F_{Z_i}^{-1}(V_i)$.
3. Combine the transformed components to form the random vector $\mathbf{Z} = (Z_1, Z_2, \ldots, Z_d)$.

The resulting random vector $\mathbf{z}$ simulates the distribution of $\mathbf{Z}$ under the assumption that the components of $\mathbf{Z}$ are independent. The independence assumption is critical for the validity of this method; it allows the joint distribution of $\mathbf{Z}$ to be expressed as the product of its marginal distributions.

Inverse Transform Sampling is powerful due to its simplicity and precision in generating samples from the target distribution through the cdf inversion. However, it requires that the cdfs of the individual components are invertible and that the inverse can be computed efficiently, which may not be feasible for complex distributions. For distributions with dependencies among components or where the cdf is not easily invertible, alternative methods such as Rejection Sampling or Monte Carlo Markov Chains (MCMC) might be more appropriate.

A.2 Rejection Sampling

Rejection Sampling, also known as Accept-Reject Method, is a technique for generating samples from a target distribution $p_{\mathbf{Z}}$ when direct sampling is challenging. It is particularly useful when the probability density function (pdf) $p_{\mathbf{Z}}(\mathbf{z})$ of $\mathbf{Z}$ is known only up to a normalization constant or is otherwise difficult to sample from directly.

The method involves using a proposal distribution $q(\mathbf{z})$ from which it is easy to sample, and a scaling constant k such that $kq(\mathbf{z})$ dominates $p_{\mathbf{Z}}(\mathbf{z})$ for all $\mathbf{z}$. The steps for Rejection Sampling are as follows:

1. Identify a proposal distribution $q(\mathbf{z})$ and determine a scaling constant k such that $kq(\mathbf{z}) \geq p_{\mathbf{Z}}(\mathbf{z})$ for all $\mathbf{z}$ in the support of $\mathbf{Z}$.
2. Generate a sample $\mathbf{z}'$ from the proposal distribution $q(\mathbf{z})$.
3. Generate a uniform random sample U from the interval [0, 1].
4. Compute the acceptance ratio $r = \frac{p_{\mathbf{Z}}(\mathbf{z}')}{kq(\mathbf{z}')}$. Accept the sample $\mathbf{z}'$ if $U \leq r$; otherwise, reject $\mathbf{z}'$.
5. Repeat the process until a sufficient number of samples are accepted.

The efficiency of Rejection Sampling depends heavily on how well the proposal distribution $q(\mathbf{z})$ approximates the target distribution $p_{\mathbf{Z}}(\mathbf{z})$ and on the choice of the scaling constant k. A poorly chosen proposal distribution or scaling constant can lead to a high rejection rate, making the method inefficient. Therefore, it's crucial to select $q(\mathbf{z})$ and k judiciously to ensure a high acceptance rate and efficient sampling.

While Rejection Sampling is versatile and applicable to a wide range of problems, it is most efficient when the proposal distribution is close to the target distribution, and the dimensionality of the problem is not excessively high. In cases where the dimensionality is high or the target distribution is complex, alternative methods such as Monte Carlo Markov Chains (MCMC) may be more suitable.

A.3 Markov Chains Monte Carlo (MCMC)

Markov Chains Monte Carlo (MCMC) methods are robust techniques for sampling from probability distributions, particularly when direct sampling is challenging. These methods are crucial in scenarios involving high-dimensional spaces or distributions known only up to a normalization constant.

A.3.1 General Principles of MCMC

MCMC algorithms are centered around the construction of a Markov chain that converges to the target distribution as its stationary distribution. The steps typically involved in an MCMC method include:

1. Initializing the chain with an arbitrary starting point.
2. Defining a proposal distribution that governs the transitions between states in the Markov chain.
3. Employing an acceptance criterion to decide whether to move to the proposed state, ensuring convergence to the target distribution.
4. Iterating the process for a sufficient number of steps to allow the chain to converge and to collect a representative set of samples.

The convergence to the target distribution and the efficiency of the sampling process depends on the choice of the proposal distribution and the acceptance criterion.

A.3.2 Metropolis-Hastings Algorithm

The Metropolis-Hastings algorithm is a versatile and widely used MCMC method. It provides a framework for sampling from complex distributions, accommodating a broad range of proposal distributions.

1. Initialize the chain with a starting point x_0.
2. Define a proposal distribution $q(x|x_{\text{current}})$.
3. Generate a candidate state x' from $q(x|x_{\text{current}})$.

4. Calculate the acceptance probability $\alpha = \min\left(1, \frac{p(x')q(x_{\text{current}}|x')}{p(x_{\text{current}})q(x'|x_{\text{current}})}\right)$.
5. Accept the candidate state with probability α, otherwise remain in the current state.
6. Repeat the process to generate a sequence of states.

The algorithm's effectiveness hinges on the appropriate choice of the proposal distribution and the accuracy of the acceptance probability calculation.

A.3.3 Application of Metropolis-Hastings in the U-Space

The U-space, characterized by the standard normal distribution $\mathbf{U} \sim \mathcal{N}(\mathbf{0}_d, \mathbf{I}_d)$, simplifies the application of MCMC methods, including the Metropolis-Hastings algorithm. In this space, sampling from conditional distributions such as $\mathcal{L}(\mathbf{U}|S(\mathbf{U}) > \gamma)$ becomes more straightforward.

Sampling from Conditional Distributions in the U-Space

Using the Metropolis-Hastings algorithm in the U-space to sample from a conditional distribution involves:

1. Initializing the chain with a starting point u_0 from the standard normal distribution $\mathcal{N}(\mathbf{0}_d, \mathbf{I}_d)$.
2. Selecting a proposal distribution $q(u|u_{\text{current}})$ that is suitable for the standard normal characteristics of the U-space.
3. Generating a candidate state u' from $q(u|u_{\text{current}})$.
4. Calculating the acceptance probability α, ensuring it reflects the condition $S(u) > \gamma$ in the U-space.
5. Accepting the candidate state with probability α, otherwise remaining in the current state.
6. Repeating the process to generate a sequence of states, focusing on the region where $S(u) > \gamma$.

This application of the Metropolis-Hastings algorithm in the U-space is particularly effective for exploring regions of interest defined by a continuous function S and a threshold γ, enabling detailed analysis of system behavior under specified conditions.

A.3.4 Hamiltonian Monte Carlo (HMC) and Gibbs Measures

Hamiltonian Monte Carlo (HMC) is an advanced MCMC method that draws inspiration from Hamiltonian dynamics in classical mechanics, making it highly effective for sampling from complex probability distributions. It is particularly well-suited for Gibbs measures, characterized by $\pi_\beta \propto \exp^{-\beta V(\mathbf{u})} \phi_d(\mathbf{u})$, where $V(\mathbf{u})$ is the potential energy function, β is the inverse temperature parameter, and $\phi_d(\mathbf{u})$ is the pdf of the standard normal distribution in $\mathbb{R}^d$.

Principles of Hamiltonian Monte Carlo

HMC augments the state space with auxiliary momentum variables and defines a Hamiltonian function that combines the potential energy, associated with the Gibbs measure, and the kinetic energy of the momentum variables. The evolution of the system is governed by Hamilton's equations and is discretized using numerical integration methods such as the Verlet Integration scheme. This approach allows HMC to propose new states that can efficiently explore the target distribution, especially in high-dimensional spaces.

Verlet Integration Scheme

The Verlet Integration scheme is crucial for simulating the Hamiltonian dynamics in HMC. It ensures the stability and energy conservation of the simulated dynamics, which are essential for maintaining the detailed balance and ergodicity of the Markov chain. The Verlet Integration scheme for HMC with Gibbs measures involves the following steps:

Algorithm 12 Verlet Integration for HMC with Gibbs Measures

```
Require: Initial position $q_0$, Initial momentum $p_0$, Step size $\epsilon$, Number of steps $L$,
  Potential energy function $V(q)$
Ensure: New state $(q_L, p_L)$
  Initialize $q \leftarrow q_0$, $p \leftarrow p_0$
  for $i = 1$ to $L$ do
      $p \leftarrow p - \frac{\epsilon}{2} \cdot \nabla V(q)$        ▷ Half-step update for momentum
      $q \leftarrow q + \epsilon \cdot p$                              ▷ Full-step update for position
      $p \leftarrow p - \frac{\epsilon}{2} \cdot \nabla V(q)$        ▷ Half-step update for momentum
  end for
  Optionally negate $p$ to make the proposal symmetric
```

The leapfrogging nature of the updates ensures the reversibility and symplectic nature of the integration, both of which are crucial for the accuracy and efficiency of HMC.

Application of HMC to Gibbs Measures in the U-Space

When applied to Gibbs measures in the U-space, HMC can efficiently explore complex energy landscapes characterized by the potential energy function V. The use of the Verlet Integration scheme allows for precise simulation of the Hamiltonian dynamics, ensuring that the chain explores the critical regions of the target distribution, such as those defined by high energy barriers or narrow passages.

In the U-space, the standard normal distribution $\phi_d(\mathbf{u})$ simplifies the computation of the kinetic energy term in the Hamiltonian, making HMC particularly well-suited for these settings. The algorithm facilitates sampling from conditional distributions, such as $\mathcal{L}(\mathbf{U}|S(\mathbf{U}) > \gamma)$, enabling the study of rare events and system behavior under extreme conditions.

In summary, HMC offers a powerful approach for sampling from complex distributions, particularly Gibbs measures in high-dimensional spaces. Its foundation in Hamiltonian dynamics and use of the Verlet Integration scheme enable efficient exploration of the state space, making it an invaluable tool in advanced statistical reliability analysis and rare event simulation.

In conclusion, MCMC methods, and specifically the Metropolis-Hastings algorithm, are invaluable in the field of probabilistic modeling and simulation, offering robust solutions for sampling from complex distributions, particularly in the context of high-dimensional spaces and the U-space.

A.4 Normalizing Flows

Normalizing flows were introduced by Rezende and Mohamed [181] as a sophisticated framework for constructing flexible and complex probabilistic models by transforming a simpler base distribution through a series of invertible mappings. Unlike the traditional isoprobabilistic transformations, normalizing flows allow for a more flexible and learnable transformation, making them particularly suitable for applications in statistical reliability engineering and deep learning robustness where the uncertainty distribution has to be learned from data. In this section, we delve into the fundamental principles of $\mathcal{T}^{\mathrm{NF}}$, illustrating their role in modeling complex distributions and their potential applications in enhancing and assessing deep learning models' reliability.

A.4.1 Defining Normalizing Flows

Normalizing flows provide a systematic way of transforming a simple, easy-to-sample distribution π_Z (typically the standard normal law), in the latent space $\mathcal{Z}$ into a complex distribution in the data space $\mathcal{X}$. More precisely, a normalizing flow is a learned transform $\mathcal{T}_{\boldsymbol{\theta}}^{\mathrm{NF}} : \mathcal{Z} \rightarrow \mathcal{X}$ enabling the modeling of a complex data distribution, ν_X, through a sequence of invertible and differentiable mappings, so that $\mathcal{T}_{\boldsymbol{\theta}}^{\mathrm{NF}} \# \pi_Z = \nu_X$.

Let $\mathbf{Z}_0 \sim \pi_Z$ denote a random vector in the latent space $\mathcal{Z}$, following a simple base distribution such as a multivariate normal distribution. We define a sequence of invertible and differentiable mappings $f_1, f_2, \ldots, f_K$, where each mapping transforms the random vector into a new space. The final transformation yields the random vector $\mathbf{X} = \mathbf{Z}_K$ in the data space $\mathcal{X}$, which aims to follow the complex data distribution ν_X of natural data. Formally, this process is described as:

$$\mathbf{Z}_K = \mathcal{T}^{\mathrm{NF}}(\mathbf{Z}_0) = f_K \circ f_{K-1} \circ \ldots \circ f_1(\mathbf{Z}_0) \tag{A.1}$$

The probability density function of $\mathbf{x}$, denoted as p_X, can be derived using the change of variables formula. This involves the determinant of the Jacobian of the inverse mappings, ensuring that the probability volume is preserved during the transformations:

$$p_X(\mathbf{x}) = p_{\mathbf{Z}_0}(f_1^{-1} \circ \ldots \circ f_K^{-1}(\mathbf{x})) \left| \det \frac{d(f_1^{-1} \circ \ldots \circ f_K^{-1})}{dx}(\mathbf{x}) \right| \tag{A.2}$$

In this framework, the complex distribution of natural data ν_x is approximated by the transformed distribution of $\mathbf{x}$. The quality of this approximation depends on the choice of the mappings $f_1, f_2, \ldots, f_K$ and their ability to capture the intricate structure of the data space $\mathcal{X}$.

A.4.2 Learning the Normalizing Flow from Data

The strength of normalizing flows, denoted as $\mathcal{T}_{\boldsymbol{\theta}}^{\mathrm{NF}}$, is encapsulated in their capacity to model intricate distributions through data-driven learning. The mappings $f_1, f_2, \ldots, f_K$, parameterized by neural networks with parameters $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_K$, are learned via maximum likelihood estimation. This process maximizes the likelihood of the observed data under the transformed distribution. Specifically, the learning process involves optimizing the parameters $\boldsymbol{\theta} = \{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_K\}$ by minimizing a loss function $\mathcal{L}(\boldsymbol{\theta})$, typically chosen as the negative log-likelihood:

$$\mathcal{L}(\boldsymbol{\theta}) = -\sum_{i=1}^{N} \log p_{\mathcal{T}^{\mathrm{NF}}}(x_i; \boldsymbol{\theta}), \tag{A.3}$$

where $p_{\mathcal{T}_{\boldsymbol{\theta}}^{\mathrm{NF}}(\mathbf{x})}$ denotes the density of the normalizing flow model parameterized by $\boldsymbol{\theta}$, and $\{x_i\}_{i=1}^{N}$ represents the observed data.

The optimization is performed using gradient-based techniques, such as stochastic gradient descent (SGD) or its variants like Adam. The update rule for a parameter $\boldsymbol{\theta}_k$ at iteration t is given by:

$$\boldsymbol{\theta}_k^{(t+1)} = \boldsymbol{\theta}_k^{(t)} - \eta \cdot \nabla_{\boldsymbol{\theta}_k} \mathcal{L}(\boldsymbol{\theta}^{(t)}), \tag{A.4}$$

where η is the learning rate, and $\nabla_{\boldsymbol{\theta}_k} \mathcal{L}(\boldsymbol{\theta}^{(t)})$ is the gradient of the loss function w.r.t. the parameter $\boldsymbol{\theta}_k$ at iteration t.

By leveraging backpropagation and adaptive learning techniques, the parameters $\boldsymbol{\theta}$ are iteratively adjusted to capture the underlying complex structures of high-dimensional data, rendering normalizing flows a powerful tool for tasks in reliability engineering and deep learning.

A.4.3 Applications in Reliability Analysis and Deep Learning

Normalizing flows serve as powerful tools for modeling intricate distributions, a critical aspect in fields such as reliability engineering and deep learning. In reliability engineering, it could facilitate the modeling of complex dependencies among system components. In the realm of deep learning, they provide a structured approach to understanding data distributions and generating new samples. This capability is invaluable for assessing model robustness and enhancing performance. The adaptability and sophistication of normalizing flows make them an essential asset in these advanced applications.

APPENDIX B

Appendix to Chapter 4

B.1 Approximations for the computation of m

Providing a very low critical probability p_c means that certification occurs when the simulation ends after a large number of iterations m. $\Lambda(L_m)$ follows a Gamma distribution $\Gamma(m, N)$ which can be then approximated by the Gaussian law $\mathcal{N}(m/N; m/N^2)$ (application of the Central Limit Theorem). We introduce ℓ_c the threshold associated to p_c s.t. $p_c = \mathbb{P}(h(\mathbf{X}) > \ell_c)$, and $m_c = \log(p_c)/\log(1 - 1/N)$.

Under this assumption:

$$\mathbb{P}(\Lambda(L_m) < \Lambda(\ell_c)) = \alpha \to \Lambda(\ell_c) = \frac{m}{N} - z_\alpha \frac{\sqrt{m}}{N} \tag{B.1}$$

with $z_\alpha = \Phi^{-1}(1-\alpha) > 0$ for $\alpha < 1/2$ and $\Lambda(\ell_c) = -\log(p_c)$. We find a first approximation of m by solving this second order polynomial in $\sqrt{m}$:

$$m \approx \tilde{m}_1 = \left\lceil \frac{1}{4}\left(z_\alpha + \sqrt{z_\alpha^2 - 4N\log(p_c)}\right)^2 \right\rceil . \tag{B.2}$$

This clearly shows that the dependence on p_c is approximately logarithmic. Table 4.1 shows that this approximation is excellent even for large p_c.

Moreover, if N is large enough, then $N\log(p_c) = N m_c \log(1 - 1/N) \approx m_c$ and m approximately satisfies

$$m - z_\alpha\sqrt{m} - m_c = 0, \tag{B.3}$$

producing

$$m \approx \tilde{m}_2 = \left\lceil \frac{1}{4}\left(z_\alpha + \sqrt{z_\alpha^2 + 4m_c}\right)^2 \right\rceil = \left\lceil m_c\left(\sqrt{1 + z_\alpha^2/4m_c} + \frac{z_\alpha}{2\sqrt{m_c}}\right)^2 \right\rceil . \tag{B.4}$$

This shows that m is a little larger than $m_c = {}^{\log(p_c)}/_{\log(1-1/N)}$.

B.2 Experiments in the idealized case

This appendix details the experimental results of Sect. 4.4.1. This section assumes that $\mathbf{X} = \mathbf{x}_o + \sigma\tilde{\mathbf{X}}$ with $\tilde{\mathbf{X}} \sim \mathcal{N}(\mathbf{0}_n; \mathbf{I}_n)$ and that $h(\mathbf{x}) = \mathbf{x}^\top\mathbf{g} - \tau$ with $\mathbf{g} \in \mathbb{R}^n$ and $\|\mathbf{g}\| = 1$ (w.l.o.g.). In this textbook case, the true probability $p = \pi_0(h(X) > 0)$ depends on τ by

$$p = 1 - \Phi\left(\frac{\tau - \mathbf{x}_o^\top\mathbf{g}}{\sigma}\right). \tag{B.5}$$

We now explain how to 'directly' sample a new particle as required by line 11, Alg. 9 for this particular case, without resorting to Alg. 10.

The projection of $\tilde{\mathbf{X}}$ onto $\mathbf{g}$ is Gaussian distributed. By linearity of the score function, conditioning on the event $\mathcal{E} := \{h(\mathbf{X}) > L\}$ means that the c.d.f of $Z := \tilde{\mathbf{X}}^\top\mathbf{g}$ equals:

$$F_Z(z) = \mathbb{1}(z > L_0).\frac{\Phi(z) - \Phi(L_0)}{1 - \Phi(L_0)} \quad \text{with } L_0 := {}^{(L - \mathbf{x}_o^\top\mathbf{g})}/_{\sigma}. \tag{B.6}$$

On the other hand, the projection of $\tilde{\mathbf{X}}$ onto any other direction orthogonal to $\mathbf{g}$ remains normal distributed. This justifies the following construction:

$$\begin{aligned} Z &= F_Z^{-1}(U) = \sigma\Phi^{-1}\left((1 - \Phi({}^{L_0}/_{\sigma}))U + \Phi({}^{L_0}/_{\sigma})\right) \quad \text{with } U \sim \mathcal{U}_{[0,1]} && \text{(B.7)} \\ \mathbf{X} &= \mathbf{x}_o + \sigma\left(Z\mathbf{g} + (\mathbf{I}_n - \mathbf{g}\mathbf{g}^\top)\mathbf{N}\right) \quad \text{with } \mathbf{N} \sim \mathcal{N}(\mathbf{0}_n; \mathbf{I}_n), && \text{(B.8)} \end{aligned}$$

In a nutshell, $(\mathbf{I}_n - \mathbf{g}\mathbf{g}^\top)$ is the projection onto the $(n-1)$-dimension subspace orthogonal to $\mathbf{g}$. This operator resets the projection of $\mathbf{N}$ onto $\mathbf{g}$, which is then set to Z. Section 4.4.1 uses this toy example to illustrate our procedure in the idealized case.

B.3 Choice of N and T

Most experiences are run with $N = 2$ which is counter-intuitive. In this section we elaborate on the choice of N and T using experiments in case of linear decision function and X follows a Gaussian law. More precisely we take $X \sim \mathcal{N}(0, I_d)$ and the score function $s : \mathbb{R}^d \ni x \mapsto x^T n$ with $n \in \mathbb{R}^d$ defining the normal vector of the decision hyperplane. For simplicity, we take $n = e_1$ i.e. the first vector of the canonical basis of $\mathbb{R}^d$. With this toy model the probability of failure for a threshold level L is given by,

$$p = \mathbb{P}(s(X) > L) = \mathbb{P}(X_1 > L) = 1 - \Phi(L) \tag{B.9}$$

We now apply the last particle algorithm 9 to the statistical test with null hypothesis $\mathcal{H}_0 : p \geq p_c$ and alternative hypothesis $\mathcal{H}_1 : p < p_c$. For numerical experiments below, we take $p = p_c$ and $\alpha = 0.05$. We let vary the number of particles N in the range $\{2, 20, 100\}$ and the parameter T in the range $\{25, 50, 100, 150, 200\}$. For each couple of parameters (N, T) we make 1000 runs and count the number of false positive (i.e. the number of times the algorithm wrongfully asserted that $p < p_c$). The results are presented in the table B.1 below.

Table B.1 – Estimation of false positive rates and number of calls in function of T and N for a toy model

N	T	Estimated false positive rate	Avg. number of calls
2	25	0.038	1.05e+03
2	50	0.041	2.08e+03
2	100	0.033	4.14e+03
2	150	0.026	6.19e+03
2	200	0.040	8.28e+03
20	25	0.034	1.04e+04
20	50	0.050	2.07e+04
20	100	0.048	4.15e+04
20	150	0.043	6.20e+04
20	200	0.043	8.29e+04
100	25	0.036	5.19e+04
100	50	0.052	1.04e+05
100	100	0.049	2.07e+05
100	150	0.033	3.11e+05
100	200	0.050	4.15e+05

Algorithm 13 Adaptive Sampling for one particle AdaptGen$(L, 1)$

```
Require: threshold $L$, finite set $\mathcal{X}$ of particles whose score is larger than $L$, input strength
    parameter $s_{in}$, scaling parameter $\gamma < 1$, acceptance ratio threshold $a_*$, gain threshold
    $g_*$
Ensure: new particle $\mathbf{X}$, new strength parameter $s_{out}$
 1: Initialize Count $\leftarrow 0$, $s_{out} \leftarrow s_{in}$, $\mathbf{X} \leftarrow \mathcal{U}(\mathcal{X})$        ▷ Draw uniformly a particle in $\mathcal{X}$
 2: for $k = 1 : t$ do
 3:     $\mathbf{Z} \leftarrow K(\mathbf{X}, s_{in})$                    ▷ $\pi_0$ reversible proposal. See Sect. 4.3.1
 4:     if $h(\mathbf{Z}) > L$ then                                        ▷ Rejection
 5:         $\mathbf{X} \leftarrow \mathbf{Z}$
 6:         Count $\leftarrow$ Count $+ 1$
 7:     end if
 8: end for
 9: if Count $< t \times a_*$ then
10:     $s_{out} \leftarrow \gamma \times s_{in}$                ▷ Decrease $s$ if acceptation rate is too low
11: else
12:     $L_* \leftarrow \min(h(\mathbf{X}), \min_{x \in \mathcal{X}} h(x))$
13:     Gain $\leftarrow \frac{L_* - L}{|L|}$
14:     if Gain $< g_*$ then
15:         $s_{out} \leftarrow \frac{s_{in}}{\gamma}$                   ▷ Increase $s$ if the progress is too low
16:     end if
17: end if
18: return $\mathbf{X}$, $s_{out}$
```

B.4 Automatic control of kernel strength

In practice the strength parameter s of the kernel is adapted at each iteration using an heuristic. More precisely we choose a acceptance ratio threshold $a_* \in [0, 1]$ and at iteration k, after the line 11 of Algorithm 1, decrease the s by a decay rate $0 < \gamma < 1$. Conversely if the acceptance ratio is high but progress, as measured by the relative gain between the old and the new level, is too slow we increase s by the same parameter γ. This tuning mechanism is further outlined in algorithm 13. Experimentally we find that, with well chosen parameters (a_*,g_*, γ) this adaptive tuning speeds up the algorithm drastically keeping both acceptance ratio and level-wise progress under control.

B.5 Proof of proposition 4

π_0 denotes the reference probability distribution. The proof applies to the last particle algorithm describes in Alg.9 in the case where the refreshed particle state $\mathsf{Gen}(l, 1)$ is given for each $l \in \mathbb{R}$ by Alg.10. We recall that in Alg.10, $\mathsf{Gen}(l, 1)$ is obtained by t iterations of a proposal K with score-based accept /reject; starting from a uniformly chosen other (surviving) particle with score strictly greater than l.

The proof is based on a (instructive and explicit) probabilistic coupling between this last particle algorithm and the 'idealized algorithm' counterpart. The latter is obtained by taking for $\mathsf{Gen}(l, 1)$ the exact conditional distribution $\pi_0(d\mathbf{x}|h(\mathbf{x}) > l)$. The underlying idea (see [86]) is that the Markov chain generated by $\mathsf{Gen}(l, 1)$ in Alg.10 leaves invariant the distribution $\pi_0(d\mathbf{x}|h(\mathbf{x}) > l)$, so that the idealized algorithm is formally the limit of the simulated algorithm when $t \to +\infty$.

Step 0: Checking the lower bound assumption

The lower bound assumption can be rewritten as follows:

$$\exists\, p_* > 0, s_0 > 0, \forall\, \mathbf{x}, s \geq s_0, \quad \mathrm{Law}(K(\mathbf{x}, s)) \geq p_* \pi_0 \tag{Doeblin}$$

where inequality between two measures simply means that their difference is a non-negative measure. (Doeblin) is a well-known irreducibility condition coined 'Doeblin condition' in the probabilistic literature on Markov chain.

Let us check that the lower bound condition is compliant with some very minor variants of the transformation method detailed in Sect. 4.3.2.

Consider for instance the transformation: $\mathbf{X} \sim \mathcal{U}(\mathcal{B}_{2,\epsilon}(\mathbf{x}_o))$, $T(\mathbf{U}, \mathbf{x}_o) = \mathbf{x}_o + \epsilon \mathbf{U}(1 : n)$ where $\mathbf{U}$ is $n + 2$-dimensional with uniform distribution on the unit sphere of $\mathbb{R}^{n+2}$.

On the other hand, consider the proposal on the unit sphere of $\mathbb{R}^{n+2}$ obtained by composing the Gaussian proposal (4.13) in $\mathbb{R}^{n+2}$ with an additional orthogonal projection. This proposal on the sphere has the following two properties: i) it is reversible with respect to the uniform distribution on the sphere (by a symmetry argument), ii) its density satisfies (Doeblin) (by lower bounding (4.13) with initial condition on the unit sphere by a centered Gaussian distribution).

Combining the latter proposal with T we obtain again a proposal reversible w.r.t. $\mathcal{U}(\mathcal{B}_{2,\epsilon}(\mathbf{x}_o))$ and satisfying (Doeblin). See below for possible (slight but technical) generalizations to proposals satisfying weaker versions of (Doeblin).

Step 1: Uniform rejection rate

The acceptance rate of a proposal satisfying (Doeblin) with accept rule given by score $h(\mathbf{x}) > l$ is bounded from below by:

$$p_* \mathbb{P}(h(\mathbf{X}) > l),$$

which is, in turn, uniformly bounded from below if $l \leq l_0$ with $\mathbb{P}(h(\mathbf{X}) > l_0) > 0$.

Note that the proof is thus compliant with the tuning of the proposal strength s w.r.t. a constant rejection rate (App. B.4), since that latter can be carried out while ensuring (Doeblin).

Step 2: Coupling of proposals

Let us define the 'local' coupling between proposals that will enable the coupling between algorithms. Let $\mathbf{x}, \mathbf{x}'$ be given, as well as a proposal satisfying (Doeblin). A *coupled proposal* $K((\mathbf{x}, s), K(\mathbf{x}, s))$ is generated as follows: i) with probability p_*, generate a successful coupling $K(\mathbf{x}, s) = K'(\mathbf{x}', s)$ with distribution π_0; ii) else, generate independent proposals $K(\mathbf{x}, s)$ and $K'(\mathbf{x}', s)$ with respective distributions $\mathrm{Law}(K(\mathbf{x}, s)) - p_*\pi_0$ and $\mathrm{Law}(K(\mathbf{x}', s)) - p_*\pi_0$.

Clearly, the associated two marginal distributions of $K(\mathbf{x}, s)$ and $K'(\mathbf{x}', s)$ are respectively $\mathrm{Law}(K(\mathbf{x}, s))$ and $\mathrm{Law}(K(\mathbf{x}', s))$.

Step 3: Coupling of the two algorithms

Let us denote by L_k and L'_k the two levels of the last particle at iteration k in Alg. 9 for the *real and idealized algorithms, respectively.* If $L_k = L'_k$, we sample independently $\mathbf{X}'_k$, the new, refreshed particle of the idealized algorithm, according to the exact conditional distribution $\pi_0(d\mathbf{x}|h(\mathbf{x}) > L_k)$ (this replaces line 1 in Alg. 10). $\mathbf{X}'_k$ is then modified in parallel with the new particle of the real algorithm according to Alg. 10 by iterating t times the coupled proposal transition of Step 2; K being used for the real and idealized algorithms, respectively.

After t iterations one has thus obtained a successful coupling with probability (conditional on L_k) $1 - (1 - p_*\mathbb{P}(h(\mathbf{X}) > L_k))^t \xrightarrow[t \to +\infty]{} 1$.

Moreover, since Alg. 10 leaves invariant the conditional distribution $\pi_0(d\mathbf{x}|h(\mathbf{x}) > L_k)$, it does not modify the distribution of the refreshed particle in the idealized algorithm.

Step 4: Conclusion by induction

Let l_0 be any critical level such that $\pi_0(h(\mathbf{X}) > l_0) > 0$. We consider the following induction hypothesis at iteration k:

H_k On the event, $L_k \leq l_0$, The probability that the two particle systems are equal tends exponentially fast to 1 when $t \to +\infty$.

Assume H_k is true. The probability that the two particle systems are equal at iteration $k+1$ is the probability conditioned by equality at iteration k multiplied by probability of equality at iteration k. If the score level is below l_0, the former conditioned probability is bounded below by $1-(1-p_*\mathbb{P}(h(\mathbf{X})>l_0))^t$ by Step 3 so that using H_k the induction on H_{k+1} is complete.

We deduce that $\mathbb{P}(L_m \leq l_0)$ converges exponentially fast with t large towards $\mathbb{P}(L'_m \leq l_0)$ for each l_0. Using in addition Theorem 3 on the idealized algorithm, we conclude the proof.

Possible Generalizations: It is possible to relax the irreducibility condition (Doeblin) so that it is verified by most practical proposals, see Sec. 4.3.2. This requires using so-called Lyapounov functions, as well as an extra (but mild) assumption on the shape of h 'at infinity'.

For instance, consider the Gaussian proposal (4.13) in $\mathbb{R}^{n+2}$. It satisfies the Doeblin condition (Doeblin), but only locally, for all $\mathbf{x}$ in a ball, p_* depending now of the size of the ball.

The extra assumption on the shape of the score function h at infinity is then necessary to check that the rejection rate is again uniformly bounded from below.

Finally, one can remark that the following so-called Lyapounov condition $\mathbb{E}[|K(\mathbf{x},s)|^2] \leq \rho|x|^2+c$ holds true (with $\rho=\frac{1}{1+s^2}<1$ and $c=\frac{s^2}{1+s^2}<+\infty$). It ensures that the proposal cannot be stuck at infinity, in areas where the 'local' Doeblin condition is poor.

One can then couple proposals using (Doeblin) as above, but only when the coupled initial states are in a given ball, and use the Lyapounov condition (see [182]) to nonetheless obtain a successful coupling with a lower bounded success rate.

The proof then works as above.

Final remarks: Note that the exponential convergence rate obtained in the proof of Proposition 4 is too sub-optimal to be suitable for practical purpose. Practical estimation of this rate is left for future work although estimating the mixing rate of such Markov chain is known to be difficult and widely dependent on the geometry of h.

B.6 Implementation details of the experiments

This section gives further details on the implementation available at
`https://github.com/karimtito/efficient-statistical`

The source code provided can be used to re-run experiments or run different experiments (see README file for more information).

B.7 ACAS Xu

In the experiments on the ACAS Xu DNN compression case study we used the 45 neural networks from the VNNLIB website (in ONNX format), which do not require normalizing the inputs. We only tested the 5 first properties since they apply to all networks. We use an adaptive procedure to tune the strength parameter s as explained in B.4. Experiments main parameters are set to: $N = 2, p_c = 10^{-50}, T = 40, \alpha = 10^{-3}$. We initialize the strength s at 1.5 and use the adaptive sampling procedure of section B.4 with $\gamma = 0.99, a_* = 0.90, g_* = 0.01$. In addition we ran experiments with the ERAN complete certification method using DeepPoly and Mixed Integer programming on the same benchmark.

B.8 MNIST

We selected 4 neural networks from the ERAN benchmark: 3 architectures of varying complexity trained with pytorch named '`convMedGRELU__PGDK_w_0.1`', '`ffnnRELU__Point_6_500`' and '`ffnnRELU__PGDK_w_0.1_6_500`' as well as a simpler model trained with tensorflow '`mnist_relu_9_200`'. We use the batched version of the Last Particle algorithm where we test the local robustnes aroung 100 images in parallel. For each image we create a system of $N(=2)$ particles and we call the score at each iteration (line 6 in Algorithm 1) with a batch consisting of all lower-scored particles. This trick accelerates the computations by taking advantage of the GPU. We also used the adaptive tuning of the strength, initializing s at 1.5 and with $\gamma = 0.999, a_* = 0.90, g_* = 0.01$.

B.9 ImageNet

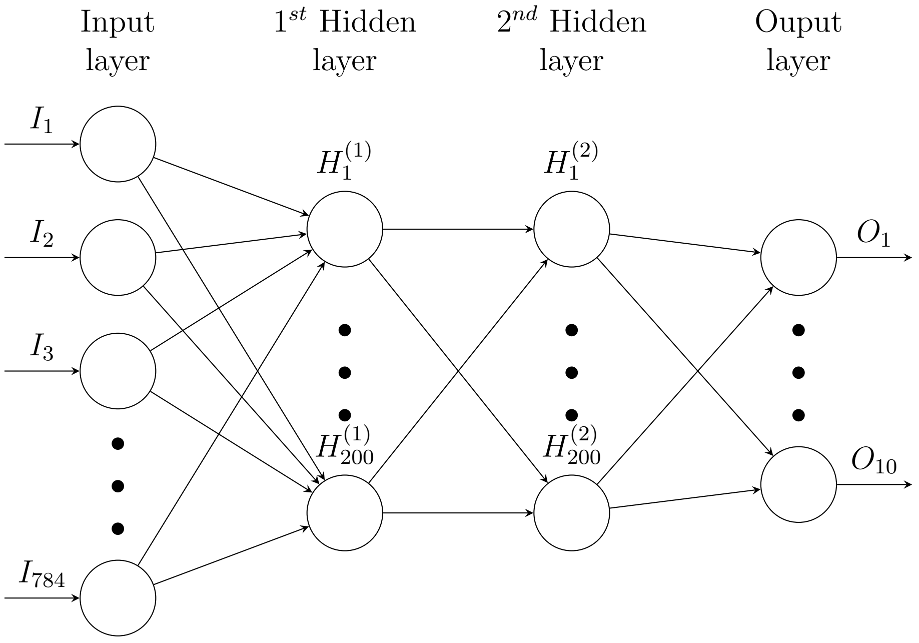
Similarly to MNIST we used a batched version of the Last Particle algorithm presented in section 4.2. Again we also used a automatic control mechanism (see section B.4, initializing s at 1 and taking $\gamma = 0.999, a_* = 0.90, g_* = 0.01$. For ImageNet we could not run the ERAN certification methods unfortunately since these methods barely scale to such high input dimension and management of ImageNet is not implemented for now on the ERAN GitHub repository.

APPENDIX C

Appendix to Chapter 5

C.1 Neural Architecture for MNIST experiments

The architecture used is a fully connected neural network with 2 hidden layers, each containing 200 units. It is illustrated below. Each hidden unit outputs a linear combination of its inputs, composed with the ReLU activation, defined by: $ReLU(x) = x\mathbb{1}_{x\geq 0}$.



C.2 Proof of Proposition 1

First, we show that the selection step is indeed sufficient to obtain an unbiased estimation. Let $\pi_{\beta_k}^N = \frac{1}{N}\sum_{n=1}^N \delta_{X_k^{(n)}}$ denote the empirical distribution of particles at iteration k, and denote by $B_{k+1}^{(n)}$ the number of offsprings of the particle $X_k^{(n)}$ after the selection step. By construction of the selection step, in which surviving particles are duplicated uniformly, one has

$$\begin{aligned}\mathbb{E}\left[B_{k+1}^{(n)} \mid (X_k^{(1)}, \ldots, X_k^{(N)})\right] &= \mathbb{P}\left[(n) \text{ is killed}\right] \times \tilde{C}_k^{-1}, \\ &= e^{-(\beta_{k+1}-\beta_k)V(X_k^{(n)})} C_k^{-1}\end{aligned}$$

where C_k and $\tilde{C}_k$ are constants independent of n, and the expectation is taken conditional on the particles $(X_k^{(1)}, \ldots, X_k^{(N)})$. The latter constants can be simplified by remarking that since the number of particles is kept equal to N, then $\sum_n B_{k+1}^{(n)} = N$, which implies

$$C_k = \frac{1}{N} \sum_{n=1}^{N} e^{-(\beta_{k+1}-\beta_k)V(X_k^{(n)})} = \frac{\widehat{Z}_{k+1}}{\widehat{Z}_k}.$$

As a consequence, since just after the selection step one has

$$\frac{1}{N} \sum_{n=1}^{N} \delta_{X_{k+1}^{(n)}} = \frac{1}{N} \sum_{n=1}^{N} B_{k+1}^{(n)} \delta_{X_k^{(n)}},$$

the following key unbiasedness (consistency) is obtained – the expectation being again conditional on the particles $(X_k^{(1)}, \ldots, X_k^{(N)})$ at iteration k: For any test function t,

$$\widehat{Z}_{k+1} \mathbb{E}\left[\mathbb{E}_{\pi_{\beta_{k+1}}^N}\left[t(X)\right] \mid (X_k^{(1)}, \ldots, X_k^{(N)})\right] = \widehat{Z}_k \mathbb{E}_{\pi_{\beta_k}^N}\left[e^{-(\beta_{k+1}-\beta_k)V(X)} t(X)\right]. \tag{C.1}$$

Second, by induction on k with initialization for $k = 0$ given by $Z_0 \mathbb{E}[\mathbb{E}_{\pi_0^N}[t(X)]] = \mathbb{E}_{\pi_0}(t(X))$, we obtain that (C.1) is unbiased in the sense that its full expectation is indeed given by $Z_{k+1} E_{\pi_{\beta_{k+1}}}(t(X))$.

Third, the mutation step (which is crucial to enforce independence between particles) does not modify the above argument by induction on unbiasedness. Indeed, at each iteration k, the kernel is applied to each particle before the selection step of iteration $k+1$. It is chosen to leave invariant the distribution π_{β_k} (see Sect. 5.1). As a consequence, the full expectation of the right-hand side of (C.1) – which consists of mutated particles – is unchanged.

In the end the quantity $\widehat{Z}_k$ is indeed an unbiased estimator of Z_k for each k. Induction on k can also be applied to the weak law of large numbers to obtain consistency at each iteration. This concludes the proof of Proposition 1.

APPENDIX D

Appendix to Chapter 6

D.1 Additional Results for FORM and SORM

We report below additional results for the FORM and SORM methods.

Table D.1 – FORM/SORM estimations of $P_{\mathrm{F}} \approx 1.69 \cdot 10^{-8}$ for the model M_2 and input $\mathbf{x}_{0,2}$, with uniform noise ($\varepsilon = 0.18$).

Attack	$P_{\mathrm{F}}^{\mathrm{FORM}}$	$P_{\mathrm{F}}^{\mathrm{SORM}}$	$\cos(\tilde{u}^*, \nabla G(\tilde{u}^*))$
CW	$1.74 \cdot 10^{-5}$	$6.88 \cdot 10^{-9}$	-0.95
FMNA	$3.17 \cdot 10^{-5}$	$6.12 \cdot 10^{-9}$	-0.996
HLRF	$1.89 \cdot 10^{-5}$	$7.56 \cdot 10^{-9}$	-0.97
	$\lVert\tilde{u}^*\rVert_2$	$G(\tilde{u}^*)$	Time (in sec.)
CW	4.14	$-2.1 \cdot 10^{-5}$	1.05
FMNA	4.0	$-1.9 \cdot 10^{-5}$	0.25
HLRF	4.12	$-2.3 \cdot 10^{-2}$	0.01

Table D.2 – FORM/SORM estimations of $P_{\mathrm{F}} \approx 8.1 \cdot 10^{-3}$ for the model M_2 and input $\mathbf{x}_{0,3}$, with uniform noise ($\varepsilon = 0.18$)

Attack	$P_{\mathrm{F}}^{\mathrm{FORM}}$	$P_{\mathrm{F}}^{\mathrm{SORM}}$	$\cos(\tilde{u}^*, \nabla G(\tilde{u}^*))$
CW	$3.22 \cdot 10^{-2}$	$5.23 \cdot 10^{-3}$	-0.95
FMNA	$3.84 \cdot 10^{-2}$	$5.37 \cdot 10^{-3}$	-0.988
HLRF	$3.29 \cdot 10^{-2}$	$5.35 \cdot 10^{-3}$	-0.957
	$\lVert\tilde{u}^*\rVert_2$	$G(\tilde{u}^*)$	Time (in sec.)
CW	1.85	$-1.0 \cdot 10^{-5}$	0.9
FMNA	1.77	$-1.1 \cdot 10^{-5}$	0.16
HLRF	1.84	$-1.8 \cdot 10^{-2}$	0.01

Table D.3 – FORM/SORM estimations of $P_{\mathrm{F}} \approx 9.92 \cdot 10^{-6}$ for the model M_2 and input $\mathbf{x}_{0,1}$, with uniform noise ($\varepsilon = 0.18$).

Attack	$P_{\mathrm{F}}^{\mathrm{FORM}}$	$P_{\mathrm{F}}^{\mathrm{SORM}}$	$\cos(\tilde{u}^*, \nabla G(\tilde{u}^*))$
CW	$6.64 \cdot 10^{-4}$	$4.66 \cdot 10^{-6}$	-0.96
FMNA	$8.45 \cdot 10^{-4}$	$3.83 \cdot 10^{-6}$	-0.993
HLRF	$7.36 \cdot 10^{-4}$	$6.53 \cdot 10^{-6}$	-0.96
	$\lVert\tilde{u}^*\rVert_2$	$G(\tilde{u}^*)$	Time (in sec.)
CW	3.21	$-1.9 \cdot 10^{-5}$	1.07
FMNA	3.14	$-2.5 \cdot 10^{-5}$	0.30
HLRF	3.18	$-2.1 \cdot 10^{-2}$	0.05

Table D.4 – FORM/SORM estimations of $P_{\mathrm{F}} \approx 5.7 \cdot 10^{-6}$ for the custom CNN model and gaussian noise ($\sigma = 0.02$).

Attack	$P_{\mathrm{F}}^{\mathrm{FORM}}$	$P_{\mathrm{F}}^{\mathrm{SORM}}$	$\cos(\tilde{u}^*, \nabla G(\tilde{u}^*))$
CW	$3.91 \cdot 10^{-5}$	NA	-0.96
FMNA	$5.22 \cdot 10^{-5}$	NA	-0.985
HLRF	$2.16 \cdot 10^{-5}$	NA	-0.973
	$\lVert\tilde{u}^*\rVert_2$	$G(\tilde{u}^*)$	Time (in sec.)
CW	3.95	$-3.7 \cdot 10^{-5}$	1.49
FMNA	3.88	$-8.0 \cdot 10^{-4}$	0.23
HLRF	4.09	$-1.9 \cdot 10^{-2}$	0.03

Bibliography

[1] K. Tit, T. Furon, and M. Rousset, « Efficient Statistical Assessment of Neural Network Corruption Robustness », *in NeurIPS 2021 - 35th Conference on Neural Information Processing Systems*, ser. Advances in Neural Information Processing Systems proceedings, vol. 34, Sydney (virtual), Australia, Dec. 2021. [Online]. Available: https://hal.archives-ouvertes.fr/hal-03407011.

[2] K. TIT, T. Furon, and M. Rousset, « Gradient-Informed Neural Network Statistical Robustness Estimation », *in Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, F. Ruiz, J. Dy, and J.-W. van de Meent, Eds., ser. Proceedings of Machine Learning Research, vol. 206, PMLR, 2023, pp. 323–334. [Online]. Available: https://proceedings.mlr.press/v206/tit23a.html.

[3] K. Tit and T. Furon, « Fast Reliability Estimation for Neural Networks with Adversarial Attack-Driven Importance Sampling », *in UAI 2024 - 40th Conference on Uncertainty in Artificial Intelligence*, Barcelona, Spain, Jul. 2024, pp. 1–11. [Online]. Available: https://hal.science/hal-04565441.

[4] A. Chojaczyk, A. Teixeira, L. Neves, J. Cardoso, and C. Guedes Soares, « Review and application of Artificial Neural Networks models in reliability analysis of steel structures », *Structural Safety*,

[5] J. Schmidhuber, « Deep learning in neural networks: An overview », *Neural Networks*, vol. 61, pp. 85–117, 2015, ISSN: 0893-6080. DOI: https://doi.org/10.1016/j.neunet.2014.09.003.

[6] I. J. Goodfellow, J. Shlens, and C. Szegedy, « Explaining and Harnessing Adversarial Examples », *CoRR*, vol. abs/1412.6572, 2014. [Online]. Available: https://api.semanticscholar.org/CorpusID:6706414.

[7] J. Gilmer, N. Ford, N. Carlini, and E. Cubuk, « Adversarial Examples Are a Natural Consequence of Test Error in Noise », *in Proceedings of the 36th International Conference on Machine Learning*, K. Chaudhuri and R. Salakhutdinov, Eds., ser. Proceedings of Machine Learning Research, vol. 97, PMLR, 2019, pp. 2280–2289. [Online]. Available: http://proceedings.mlr.press/v97/gilmer19a.html.

[8] J.-Y. Franceschi, A. Fawzi, and O. Fawzi, « Robustness of classifiers to uniform ℓ_p and Gaussian noise », *in Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, A. Storkey and F. Perez-Cruz, Eds.,

ser. Proceedings of Machine Learning Research, vol. 84, PMLR, 2018, pp. 1280–1288. [Online]. Available: http://proceedings.mlr.press/v84/franceschi18a.html.

[9] R. M. Neal, « Handbook of Markov Chain Monte Carlo », *in.* Chapman and Hall/CRC, 2011, ch. MCMC using Hamiltonian dynamics. DOI: 10.1201/b10905. [Online]. Available: https://doi.org/10.1201/b10905.

[10] C. A. R. Hoare, « The Emperor's Old Clothes », *Commun. ACM*, vol. 24, *2*, 75–83, 1981, ISSN: 0001-0782. DOI: 10.1145/358549.358561. [Online]. Available: https://doi.org/10.1145/358549.358561.

[11] O. Ditlevsen and H. Madsen, *Structural Reliability Methods*, English. John Wiley Sons Ltd, 1996, ISBN: 0471960861.

[12] J. Morio and M. Balesdent, *Estimation of Rare Event Probabilities in Complex Aerospace and Other Systems*, J. Morio and M. Balesdent, Eds. Woodhead Publishing, 2016, ISBN: 978-0-08-100091-5. DOI: https://doi.org/10.1016/B978-0-08-100091-5.00001-0. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780081000915000010.

[13] S. Taghipour, *Reliability and Maintenance of Medical Devices.* Jan. 2012, ISBN: 978-3-8454-4184-9.

[14] A. Pradeep, M. Bakoev, and N. Akhroljonova, « A Reliability Analysis of Self-Driving Vehicles: Evaluating the Safety and Performance of Autonomous Driving Systems », *in 2023 15th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*, 2023, pp. 1–5. DOI: 10.1109/ECAI58194.2023.10194188.

[15] W. Feller, *An Introduction to Probability Theory and Its Applications*, ser. An Introduction to Probability Theory and Its Applications v. 1-2. Wiley, 1957, ISBN: 9780471257097. [Online]. Available: https://books.google.fr/books?id=K7kdAQAAMAAJ.

[16] J. Gall, *Measure Theory, Probability, and Stochastic Processes*, ser. Graduate Texts in Mathematics. Springer International Publishing, 2022, ISBN: 9783031142055. [Online]. Available: https://books.google.fr/books?id=Ba2YEAAAQBAJ.

[17] A. Sklar, « Random variables, joint distribution functions, and copulas », eng, *Kybernetika*, vol. 09, *6*, (449)–460, 1973. [Online]. Available: http://eudml.org/doc/28992.

[18] S. Cambanis, S. Huang, and G. Simons, « On the theory of elliptically contoured distributions », *Journal of Multivariate Analysis*, vol. 11, *3*, pp. 368–385, 1981, ISSN: 0047-259X. DOI: `https://doi.org/10.1016/0047-259X(81)90082-8`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/0047259X81900828`.

[19] H. J. Malik and B. Abraham, « Multivariate Logistic Distributions », *The Annals of Statistics*, vol. 1, *3*, pp. 588 –590, 1973. DOI: `10.1214/aos/1176342430`. [Online]. Available: `https://doi.org/10.1214/aos/1176342430`.

[20] B. C. Sutradhar, « On the Characteristic Function of Multivariate Student t-Distribution », *The Canadian Journal of Statistics / La Revue Canadienne de Statistique*, vol. 14, *4*, pp. 329–337, 1986, ISSN: 03195724. [Online]. Available: `http://www.jstor.org/stable/3315191` (visited on 12/26/2023).

[21] A. D. Kiureghian and P. Liu, « Structural Reliability under Incomplete Probability Information », *Journal of Engineering Mechanics*, vol. 112, *1*, pp. 85–104, 1986. DOI: `10.1061/(ASCE)0733-9399(1986)112:1(85)`.

[22] R. K. Y. L. Hossein Rad and R. Faff, « The profitability of pairs trading strategies: distance, cointegration and copula methods », *Quantitative Finance*, vol. 16, *10*, pp. 1541–1558, 2016. DOI: `10.1080/14697688.2016.1164337`.

[23] A. Nataf, « Étude graphique de détermination de distributions de probabilités planes dont les marges sont données », *Annales de l'ISUP*, vol. XI, *3*, [257]–260, 1962. [Online]. Available: `https://hal.science/hal-04095227`.

[24] R. Lebrun and A. Dutfoy, « A generalization of the Nataf transformation to distribution with copula », *Probabilistic Engineering Mechanics*, vol. 24, pp. 172–178, Apr. 2009. DOI: `10.1016/j.probengmech.2008.05.001`.

[25] M. Rosenblatt, « Remarks on a Multivariate Transformation », *The Annals of Mathematical Statistics*, vol. 23, *3*, pp. 470–472, 1952, ISSN: 00034851. [Online]. Available: `http://www.jstor.org/stable/2236692` (visited on 12/05/2023).

[26] R. Lebrun and A. Dutfoy, « Do Rosenblatt and Nataf isoprobabilistic transformations really differ? », *Probabilistic Engineering Mechanics*, vol. 24, *4*, pp. 577–584, 2009, ISSN: 0266-8920. DOI: `https://doi.org/10.1016/j.probengmech.2009.04.006`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S0266892009000307`.

[27] A. M. Hasofer and N. Lind, « Exact and Invariant Second-Moment Code Format », *Journal of Engineering Mechanics-asce*, vol. 100, pp. 111–121, 1974. [Online]. Available: `https://api.semanticscholar.org/CorpusID:118986521`.

[28] R. Rackwitz and B. Flessler, « Structural reliability under combined random load sequences », *Computers Structures*, vol. 9, *5*, pp. 489–494, 1978, ISSN: 0045-7949. DOI: `https://doi.org/10.1016/0045-7949(78)90046-9`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/0045794978900469`.

[29] P.-L. Liu and A. Der Kiureghian, « Optimization algorithms for structural reliability », *Structural Safety*, vol. 9, *3*, pp. 161–177, 1991, ISSN: 0167-4730. DOI: `https://doi.org/10.1016/0167-4730(91)90041-7`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/0167473091900417`.

[30] Y. Zhang and A. Der Kiureghian, *Finite Element Reliability Methods for Inelastic Structures*. University of California, Berkeley, 1994. [Online]. Available: `https://books.google.fr/books?id=0VFHjwEACAAJ`.

[31] L. Armijo, « Minimization of functions having Lipschitz continuous first partial derivatives. », *Pacific Journal of Mathematics*, vol. 16, *1*, pp. 1 –3, 1966.

[32] J. B. Rosen, « The Gradient Projection Method for Nonlinear Programming. Part II. Nonlinear Constraints », *Journal of the Society for Industrial and Applied Mathematics*, vol. 9, *4*, pp. 514–532, 1961, ISSN: 03684245. [Online]. Available: `http://www.jstor.org/stable/2098878` (visited on 01/02/2024).

[33] D. G. Luenberger and Y. Ye, *Linear and Nonlinear Programming*. Springer Publishing Company, Incorporated, 2015, ISBN: 3319188410.

[34] K. Schittkowski, « On the convergence of a sequential quadratic programming method with an augmented lagrangian line search function », *Mathematische Operationsforschung und Statistik. Series Optimization*, vol. 14, *2*, pp. 197–216, 1983. DOI: `10.1080/02331938308842847`. eprint: `https://doi.org/10.1080/02331938308842847`. [Online]. Available: `https://doi.org/10.1080/02331938308842847`.

[35] K. W. Breitung, *Asymptotic Approximations for Probability Integrals*, 1st ed. 1994., ser. Lecture Notes in Mathematics, Berlin, Heidelberg : Springer Berlin Heidelberg : 1994. [Online]. Available: `https://doi.org/10.1007/BFb0073538`.

[36] M. Hohenbichler and R. Rackwitz, « Improvement Of Second-Order Reliability Estimates by Importance Sampling », *Journal of Engineering Mechanics-asce*, vol. 114, pp. 2195–2199, 1988. [Online]. Available: https://api.semanticscholar.org/CorpusID:120401958.

[37] J.-M. Bourinet, « Reliability analysis and optimal design under uncertainty - Focus on adaptive surrogate-based approaches », Habilitation à diriger des recherches, Université Clermont Auvergne, Jan. 2018. [Online]. Available: https://theses.hal.science/tel-01737299.

[38] A. Der Kiureghian and T. Dakessian, « Multiple design points in first and second-order reliability », *Structural Safety*, vol. 20, *1*, pp. 37–49, 1998, ISSN: 0167-4730. DOI: https://doi.org/10.1016/S0167-4730(97)00026-X. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S016747309700026X.

[39] P. Koutsourelakis, H. Pradlwarter, and G. Schuëller, « Reliability of structures in high dimensions, part I: algorithms and applications », *Probabilistic Engineering Mechanics*, vol. 19, *4*, pp. 409–417, 2004, ISSN: 0266-8920. DOI: https://doi.org/10.1016/j.probengmech.2004.05.001. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0266892004000402.

[40] A. Der Kiureghian, *Structural and System Reliability.* Cambridge University Press, 2022, ISBN: 9781108834148. [Online]. Available: https://books.google.fr/books?id=M_JLEAAAQBAJ.

[41] E. J. Gumbel, *Statistics of Extremes.* New York Chichester, West Sussex: Columbia University Press, 1958, ISBN: 9780231891318. DOI: doi:10.7312/gumb92958. [Online]. Available: https://doi.org/10.7312/gumb92958.

[42] G. Rubino and B. Tuffin, *Introduction to Rare Event Simulation.* John Wiley Sons, Ltd, 2009, ISBN: 9780470745403. DOI: https://doi.org/10.1002/9780470745403.ch1. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9780470745403.ch1. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470745403.ch1.

[43] A. B. Owen, *Monte Carlo theory, methods and examples.* https://artowen.su.domains/mc/, 2013.

[44] R. Eckhardt, « Stan ulam, john von neumann, and the monte carlo method », *Los Alamos Science*, vol. 15, pp. 131–136, 1987.

[45] H. Niederreiter, *Random Number Generation and Quasi-Monte Carlo Methods.* Society for Industrial and Applied Mathematics, 1992. DOI: `10.1137/1.9781611970081`. eprint: `https://epubs.siam.org/doi/pdf/10.1137/1.9781611970081`. [Online]. Available: `https://epubs.siam.org/doi/abs/10.1137/1.9781611970081`.

[46] M. Matsumoto and T. Nishimura, « Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator », *ACM Trans. Model. Comput. Simul.*, vol. 8, *1*, 3–30, 1998, ISSN: 1049-3301. DOI: `10.1145/272991.272995`. [Online]. Available: `https://doi.org/10.1145/272991.272995`.

[47] A. Marghescu and P. Svasta, « Into generating True Random Numbers - a practical approach using FPGA », *in 2015 IEEE 21st International Symposium for Design and Technology in Electronic Packaging (SIITME)*, 2015, pp. 319–322. DOI: `10.1109/SIITME.2015.7342346`.

[48] W. J. Morokoff and R. E. Caflisch, « Quasi-Monte Carlo Integration », *Journal of Computational Physics*, vol. 122, *2*, pp. 218–230, 1995, ISSN: 0021-9991. DOI: `https://doi.org/10.1006/jcph.1995.1209`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S0021999185712090`.

[49] C. Andrieu, N. de Freitas, A. Doucet, and M. I. Jordan, « An Introduction to MCMC for Machine Learning », *Machine Learning*, vol. 50, *1–2*, pp. 5–43, 2003.

[50] C. Robert and G. Casella, *Monte Carlo statistical methods.* Springer Verlag, 2004.

[51] A. Owen and Y. Zhou, « Safe and Effective Importance Sampling », *Journal of the American Statistical Association*, vol. 95, *449*, pp. 135–143, 2000, ISSN: 01621459. [Online]. Available: `http://www.jstor.org/stable/2669533` (visited on 01/24/2024).

[52] T. Hesterberg, « Weighted Average Importance Sampling and Defensive Mixture Distributions », *Technometrics*, vol. 37, *2*, pp. 185–194, 1995, ISSN: 00401706. [Online]. Available: `http://www.jstor.org/stable/1269620` (visited on 01/24/2024).

[53] M.-S. Oh and J. O. Berger, « Integration of Multimodal Functions by Monte Carlo Importance Sampling », *Journal of the American Statistical Association*, vol. 88, *422*, pp. 450–456, 1993, ISSN: 01621459. [Online]. Available: `http://www.jstor.org/stable/2290324` (visited on 01/24/2024).

[54] Y. Qiu and X. Wang, « Efficient Multimodal Sampling via Tempered Distribution Flow », *Journal of the American Statistical Association*, vol. 0, *0*, pp. 1–15, 2023. DOI: 10.1080/01621459.2023.2198059. eprint: https://doi.org/10.1080/01621459.2023.2198059. [Online]. Available: https://doi.org/10.1080/01621459.2023.2198059.

[55] C. L. Canonne, « Topics and Techniques in Distribution Testing: A Biased but Representative Sample », *Foundations and Trends® in Communications and Information Theory*, vol. 19, *6*, pp. 1032–1198, 2022, ISSN: 1567-2190. DOI: 10.1561/0100000114. [Online]. Available: http://dx.doi.org/10.1561/0100000114.

[56] R. Melchers, « Importance sampling in structural systems », *Structural Safety*, vol. 6, *1*, pp. 3–10, 1989, ISSN: 0167-4730. DOI: https://doi.org/10.1016/0167-4730(89)90003-9. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0167473089900039.

[57] B. Li, T. Bengtsson, and P. Bickel, « Curse-of-dimensionality revisited: Collapse of importance sampling in very large scale systems », Jan. 2005.

[58] S. Chatterjee and P. Diaconis, « The sample size required in importance sampling », *The Annals of Applied Probability*, vol. 28, *2*, pp. 1099 –1135, 2018. DOI: 10.1214/17-AAP1326. [Online]. Available: https://doi.org/10.1214/17-AAP1326.

[59] A. Kong, « A note on importance sampling using standardized weights. », University of Chicago, Dept. of Statistics, Tech. Rep., 1992.

[60] G. W. Oehlert, « A Note on the Delta Method », *The American Statistician*, vol. 46, *1*, pp. 27–29, 1992. DOI: 10.1080/00031305.1992.10475842. eprint: https://www.tandfonline.com/doi/pdf/10.1080/00031305.1992.10475842. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1080/00031305.1992.10475842.

[61] V. Elvira, L. Martino, and C. P. Robert, « Rethinking the Effective Sample Size », *International Statistical Review*, vol. 90, *3*, pp. 525–550, 2022. DOI: 10.1111/insr.12500. [Online]. Available: https://ideas.repec.org/a/bla/istatr/v90y2022i3p525-550.html.

[62] R. Y. Rubinstein and D. P. Kroese, *Simulation and the Monte Carlo Method*, 3rd. Wiley Publishing, 2016, ISBN: 1118632168.

[63] S.-K. Au and J. L. Beck, « Estimation of small failure probabilities in high dimensions by subset simulation », *Probabilistic Engineering Mechanics*, vol. 16, *4*, pp. 263–277, 2001, ISSN: 0266-8920. DOI: https://doi.org/10.1016/S0266-8920(01)00019-4. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0266892001000194.

[64] F. Uribe, I. Papaioannou, Y. M. Marzouk, and D. Straub, « Cross-Entropy-Based Importance Sampling with Failure-Informed Dimension Reduction for Rare Event Simulation », *SIAM/ASA Journal on Uncertainty Quantification*, vol. 9, *2*, pp. 818–847, 2021. DOI: 10.1137/20M1344585. eprint: https://doi.org/10.1137/20M1344585. [Online]. Available: https://doi.org/10.1137/20M1344585.

[65] M. El Masri, J. Morio, and F. Simatos, « Improvement of the cross-entropy method in high dimension for failure probability estimation through a one-dimensional projection without gradient estimation », *Reliability Engineering System Safety*, vol. 216, p. 107 991, 2021, ISSN: 0951-8320. DOI: https://doi.org/10.1016/j.ress.2021.107991. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0951832021005019.

[66] M. F. Bugallo, V. Elvira, L. Martino, D. Luengo, J. Miguez, and P. M. Djuric, « Adaptive Importance Sampling: The past, the present, and the future », *IEEE Signal Processing Magazine*, vol. 34, *4*, pp. 60–79, 2017. DOI: 10.1109/MSP.2017.2699226.

[67] M. B. Giles, *Multilevel Monte Carlo methods*, 2013. arXiv: 1304.5472 [math.NA].

[68] I. Csiszár and P. Shields, « Information Theory and Statistics: A Tutorial », *Foundations and Trends® in Communications and Information Theory*, vol. 1, *4*, pp. 417–528, 2004, ISSN: 1567-2190. DOI: 10.1561/0100000004. [Online]. Available: http://dx.doi.org/10.1561/0100000004.

[69] H. Robbins and S. Monro, « A Stochastic Approximation Method », *The Annals of Mathematical Statistics*, vol. 22, *3*, pp. 400 –407, 1951. DOI: 10.1214/aoms/1177729586. [Online]. Available: https://doi.org/10.1214/aoms/1177729586.

[70] J. Chan, P. Glynn, and D. Kroese, « A comparison of cross-entropy and variance minimization strategies », *Journal of Applied Probability*, vol. 48, pp. 183–194, Aug. 2011. DOI: 10.1017/S0021900200099216.

[71] T. Nishiyama and I. Sason, « On Relations Between the Relative Entropy and 2-Divergence, Generalizations and Applications », *Entropy*, vol. 22, *5*, p. 563, May 2020, ISSN: 1099-4300. DOI: 10.3390/e22050563. [Online]. Available: http://dx.doi.org/10.3390/e22050563.

[72] I. Papaioannou, S. Geyer, and D. Straub, « Improved cross entropy-based importance sampling with a flexible mixture model », *Reliability Engineering System Safety*, vol. 191, p. 106 564, 2019, ISSN: 0951-8320. DOI: https://doi.org/10.1016/j.ress.2019.106564. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0951832019301528.

[73] M. El Masri, « High dimensional importance sampling through projections on a low dimensional subspace », Theses, ISAE - Institut Supérieur de l'Aéronautique et de l'Espace, Mar. 2022. [Online]. Available: https://hal.science/tel-03927607.

[74] Y. B. Kim, D. S. Roh, and M. Y. Lee, « Nonparametric adaptive importance sampling for rare event simulation », *in 2000 Winter Simulation Conference Proceedings (Cat. No.00CH37165)*, vol. 1, 2000, 767–772 vol.1. DOI: 10.1109/WSC.2000.899849.

[75] J. Morio, « Extreme quantile estimation with nonparametric adaptive importance sampling », *Simulation Modelling Practice and Theory*, vol. 27, pp. 76–89, 2012, ISSN: 1569-190X. DOI: https://doi.org/10.1016/j.simpat.2012.05.008. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1569190X1200072X.

[76] E. Parzen, « On Estimation of a Probability Density Function and Mode », *The Annals of Mathematical Statistics*, vol. 33, *3*, pp. 1065 –1076, 1962. DOI: 10.1214/aoms/1177704472. [Online]. Available: https://doi.org/10.1214/aoms/1177704472.

[77] M. Rosenblatt, « Remarks on Some Nonparametric Estimates of a Density Function », *The Annals of Mathematical Statistics*, vol. 27, *3*, pp. 832 –837, 1956. DOI: 10.1214/aoms/1177728190. [Online]. Available: https://doi.org/10.1214/aoms/1177728190.

[78] Z. Wang and D. W. Scott, « Nonparametric density estimation for high-dimensional data—Algorithms and applications », *WIREs Computational Statistics*, vol. 11, *4*, e1461, 2019. DOI: https://doi.org/10.1002/wics.1461. eprint: https://wires.onlinelibrary.wiley.com/doi/pdf/10.1002/wics.1461. [Online]. Available: https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/wics.1461.

[79] J. Demange-Chryst, F. Bachoc, J. Morio, and T. Krauth, « Variational autoencoder with weighted samples for high-dimensional non-parametric adaptive importance sampling », *Transactions on Machine Learning Research*, 2024, ISSN: 2835-8856. [Online]. Available: `https://openreview.net/forum?id=nzG9KGssSe`.

[80] H. Kahn and T. E. Harris, « Estimation of particle transmission by random sampling », *National Bureau of Standards applied mathematics series*, vol. 12, pp. 27–30, 1951.

[81] M. Amrein and H. R. Künsch, « A variant of importance splitting for rare event estimation: Fixed number of successes », *ACM Trans. Model. Comput. Simul.*, vol. 21, *2*, 2011, ISSN: 1049-3301. DOI: `10.1145/1899396.1899401`. [Online]. Available: `https://doi.org/10.1145/1899396.1899401`.

[82] I. Papaioannou, W. Betz, K. Zwirglmaier, and D. Straub, « MCMC algorithms for Subset Simulation », *Probabilistic Engineering Mechanics*, vol. 41, pp. 89–103, 2015, ISSN: 0266-8920. DOI: `https://doi.org/10.1016/j.probengmech.2015.06.006`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S0266892015300205`.

[83] L. Katafygiotis and K. Zuev, « Geometric insight into the challenges of solving high-dimensional reliability problems », *Probabilistic Engineering Mechanics*, vol. 23, *2*, pp. 208–218, 2008, 5th International Conference on Computational Stochastic Mechanics, ISSN: 0266-8920. DOI: `https://doi.org/10.1016/j.probengmech.2007.12.026`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S0266892007000707`.

[84] F. Cérou and A. Guyader, « Adaptive Multilevel Splitting for Rare Event Analysis », *Stochastic Analysis and Applications*, vol. 25, *2*, pp. 417–443, 2007. DOI: `10.1080/07362990601139628`. eprint: `https://doi.org/10.1080/07362990601139628`. [Online]. Available: `https://doi.org/10.1080/07362990601139628`.

[85] F. Cérou, A. Guyader, and M. Rousset, « Adaptive multilevel splitting: Historical perspective and recent results », *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 29, *4*, p. 043 108, Apr. 2019, ISSN: 1054-1500. DOI: `10.1063/1.5082247`. eprint: `https://pubs.aip.org/aip/cha/article-pdf/doi/10.1063/1.5082247/14620716/043108\_1\_online.pdf`. [Online]. Available: `https://doi.org/10.1063/1.5082247`.

[86] A. Guyader, N. Hengartner, and E. Matzner-Løber, « Simulation and Estimation of Extreme Quantiles and Extreme Probabilities », *Applied Mathematics & Optimization*, vol. 64, pp. 171–196, Oct. 2011. DOI: `10.1007/s00245-011-9135-z`.

[87] A. Rosenblueth and N. Wiener, « The Role of Models in Science », *Philosophy of Science*, vol. 12, *4*, pp. 316–321, 1945. DOI: `10.1086/286874`.

[88] C. Bauckhage, C. Ojeda, J. Schücker, R. Sifa, and S. Wrobel, « Informed Machine Learning Through Functional Composition », Aug. 2018.

[89] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, « Learning representations by back-propagating errors », *nature*, vol. 323, *6088*, pp. 533–536, 1986.

[90] L. Bottou, F. E. Curtis, and J. Nocedal, « Optimization Methods for Large-Scale Machine Learning », *SIAM Review*, vol. 60, *2*, pp. 223–311, 2018. DOI: `10.1137/16M1080173`. eprint: `https://doi.org/10.1137/16M1080173`. [Online]. Available: `https://doi.org/10.1137/16M1080173`.

[91] W. Mcculloch and W. Pitts, « A Logical Calculus of Ideas Immanent in Nervous Activity », *Bulletin of Mathematical Biophysics*, vol. 5, pp. 127–147, 1943.

[92] F. Rosenblatt, *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*, ser. Cornell Aeronautical Laboratory. Report no. VG-1196-G-8. Spartan Books, 1962. [Online]. Available: `https://books.google.fr/books?id=7FhRAAAAMAAJ`.

[93] M. Minsky and S. Papert, *Perceptrons.* Cambridge, MA: MIT Press, 1969.

[94] K.-I. Funahashi, « On the approximate realization of continuous mappings by neural networks », *Neural Networks*, vol. 2, *3*, pp. 183–192, 1989, ISSN: 0893-6080. DOI: `https://doi.org/10.1016/0893-6080(89)90003-8`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/0893608089900038`.

[95] K. Hornik, M. Stinchcombe, and H. White, « Multilayer feedforward networks are universal approximators », *Neural Networks*, vol. 2, *5*, pp. 359–366, 1989, ISSN: 0893-6080. DOI: `https://doi.org/10.1016/0893-6080(89)90020-8`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/0893608089900208`.

[96] Y. L. Cun, B. Boser, J. S. Denker, R. E. Howard, W. Habbard, L. D. Jackel, and D. Henderson, « Handwritten digit recognition with a back-propagation network », *in Advances in Neural Information Processing Systems 2*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1990, 396–404, ISBN: 1558601007.

[97] J. Schmidhuber, « Learning Complex, Extended Sequences Using the Principle of History Compression », *Neural Computation*, vol. 4, *2*, pp. 234–242, Mar. 1992, ISSN: 0899-7667. DOI: 10.1162/neco.1992.4.2.234. eprint: https://direct.mit.edu/neco/article-pdf/4/2/234/812272/neco.1992.4.2.234.pdf. [Online]. Available: https://doi.org/10.1162/neco.1992.4.2.234.

[98] A. Krizhevsky, I. Sutskever, and G. E. Hinton, « ImageNet Classification with Deep Convolutional Neural Networks », *Commun. ACM*, vol. 60, *6*, 84–90, 2017, ISSN: 0001-0782. DOI: 10.1145/3065386. [Online]. Available: https://doi.org/10.1145/3065386.

[99] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, « ImageNet: A large-scale hierarchical image database », *in 2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848.

[100] K. He, X. Zhang, S. Ren, and J. Sun, « Deep Residual Learning for Image Recognition », *CoRR*, vol. abs/1512.03385, 2015. arXiv: 1512.03385. [Online]. Available: http://arxiv.org/abs/1512.03385.

[101] A. Mehrish, N. Majumder, R. Bharadwaj, R. Mihalcea, and S. Poria, « A review of deep learning techniques for speech processing », *Information Fusion*, vol. 99, p. 101 869, 2023, ISSN: 1566-2535. DOI: https://doi.org/10.1016/j.inffus.2023.101869. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1566253523001859.

[102] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, « A survey of deep learning techniques for autonomous driving », *Journal of Field Robotics*, vol. 37, *3*, 362–386, Nov. 2019, ISSN: 1556-4967. DOI: 10.1002/rob.21918. [Online]. Available: http://dx.doi.org/10.1002/rob.21918.

[103] J. M. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Zídek, A. Potapenko, A. Bridgland, C. Meyer, S. A. A. Kohl, A. Ballard, A. Cowie, B. Romera-Paredes, S. Nikolov, R. Jain, J. Adler, T. Back, S. Petersen, D. A. Reiman, E. Clancy, M. Zielinski, M. Steinegger,

M. Pacholska, T. Berghammer, S. Bodenstein, D. Silver, O. Vinyals, A. W. Senior, K. Kavukcuoglu, P. Kohli, and D. Hassabis, « Highly accurate protein structure prediction with AlphaFold », *Nature*, vol. 596, pp. 583 –589, 2021. [Online]. Available: `https://api.semanticscholar.org/CorpusID:235959867`.

[104] R. A. Fisher, *Iris*, UCI Machine Learning Repository, DOI: https://doi.org/10.24432/C56C76, 1988.

[105] V. Vapnik, *The Nature of Statistical Learning Theory*, ser. Information Science and Statistics. Springer New York, 1999, ISBN: 9780387987804. [Online]. Available: `https://books.google.fr/books?id=sna9BaxVbj8C`.

[106] B. K. Natarajan, « On learning sets and functions », *Machine Learning*, vol. 4, pp. 67–97, 2004. [Online]. Available: `https://api.semanticscholar.org/CorpusID:24468766`.

[107] K. L. L. Chao-Ying Joanne Peng and G. M. Ingersoll, « An Introduction to Logistic Regression Analysis and Reporting », *The Journal of Educational Research*, vol. 96, *1*, pp. 3–14, 2002. DOI: `10.1080/00220670209598786`. eprint: `https://doi.org/10.1080/00220670209598786`. [Online]. Available: `https://doi.org/10.1080/00220670209598786`.

[108] C. Cortes and V. N. Vapnik, « Support-Vector Networks », *Machine Learning*, vol. 20, pp. 273–297, 1995. [Online]. Available: `https://api.semanticscholar.org/CorpusID:52874011`.

[109] J. R. Quinlan, « Induction of Decision Trees », *Machine Learning*, vol. 1, pp. 81–106, 1986.

[110] L. Breiman, « Random Forests », English, *Machine Learning*, vol. 45, *1*, pp. 5–32, 2001, ISSN: 0885-6125. DOI: `10.1023/A:1010933404324`. [Online]. Available: `http://dx.doi.org/10.1023/A%3A1010933404324`.

[111] Z. Allen-Zhu, Y. Li, and Y. Liang, « Learning and generalization in overparameterized neural networks, going beyond two layers », *in Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2019.

[112] P. L. Bartlett, P. M. Long, G. Lugosi, and A. Tsigler, « Benign overfitting in linear regression », *Proceedings of the National Academy of Sciences*, vol. 117, *48*, 30063–30070, Apr. 2020, ISSN: 1091-6490. DOI: 10.1073/pnas.1907378117. [Online]. Available: http://dx.doi.org/10.1073/pnas.1907378117.

[113] D. R. Cox, « The Regression Analysis of Binary Sequences (with Discussion) », *J Roy Stat Soc B*, vol. 20, pp. 215–242, 1958.

[114] A. Mao, M. Mohri, and Y. Zhong, *Cross-Entropy Loss Functions: Theoretical Analysis and Applications*, 2023. arXiv: 2304.07288 [cs.LG].

[115] A. Graves, « Generating Sequences With Recurrent Neural Networks », *ArXiv*, vol. abs/1308.0850, 2013. [Online]. Available: https://api.semanticscholar.org/CorpusID:1697424.

[116] D. P. Kingma and J. Ba, « Adam: A Method for Stochastic Optimization », *in 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015. [Online]. Available: http://arxiv.org/abs/1412.6980.

[117] N. Qian, « On the momentum term in gradient descent learning algorithms », *Neural Networks*, vol. 12, *1*, pp. 145–151, 1999, ISSN: 0893-6080. DOI: https://doi.org/10.1016/S0893-6080(98)00116-6. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0893608098001166.

[118] A. Sanyal, P. K. Dokania, V. Kanade, and P. Torr, « How Benign is Benign Overfitting ? », *in International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=g-wu9TMPODo.

[119] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, « On Calibration of Modern Neural Networks », *in Proceedings of the 34th International Conference on Machine Learning*, D. Precup and Y. W. Teh, Eds., ser. Proceedings of Machine Learning Research, vol. 70, PMLR, 2017, pp. 1321–1330. [Online]. Available: https://proceedings.mlr.press/v70/guo17a.html.

[120] M. I. Jordan, « Serial order: a parallel distributed processing approach. Technical report, June 1985-March 1986 », [Online]. Available: https://www.osti.gov/biblio/6910294.

[121] S. Hochreiter and J. Schmidhuber, « Long Short-Term Memory », *Neural Comput.*, vol. 9, *8*, 1735–1780, 1997, ISSN: 0899-7667. DOI: `10.1162/neco.1997.9.8.1735`. [Online]. Available: `https://doi.org/10.1162/neco.1997.9.8.1735`.

[122] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, « Generative Adversarial Nets », *in Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, Eds., vol. 27, Curran Associates, Inc., 2014. [Online]. Available: `https://proceedings.neurips.cc/paper_files/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf`.

[123] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, « Attention is All you Need », *in Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30, Curran Associates, Inc., 2017. [Online]. Available: `https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf`.

[124] I. Goodfellow, J. Shlens, and C. Szegedy, « Explaining and Harnessing Adversarial Examples », *in International Conference on Learning Representations*, 2015. [Online]. Available: `http://arxiv.org/abs/1412.6572`.

[125] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, *Practical Black-Box Attacks against Machine Learning*, 2017. arXiv: `1602.02697 [cs.CR]`.

[126] T. Maho, T. Furon, and E. Le Merrer, « SurFree: A Fast Surrogate-Free Black-Box Attack », *in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 10 430–10 439.

[127] F. Croce and M. Hein, « Sparse and Imperceivable Adversarial Attacks », *in 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Los Alamitos, CA, USA: IEEE Computer Society, 2019, pp. 4723–4731. DOI: `10.1109/ICCV.2019.00482`. [Online]. Available: `https://doi.ieeecomputersociety.org/10.1109/ICCV.2019.00482`.

[128] F. Croce and M. Hein, « Mind the Box: l_1-APGD for Sparse Adversarial Attacks on Image Classifiers », *in Proceedings of the 38th International Conference on Machine Learning*, M. Meila and T. Zhang, Eds., ser. Proceedings of Machine

Learning Research, vol. 139, PMLR, 2021, pp. 2201–2211. [Online]. Available: https://proceedings.mlr.press/v139/croce21a.html.

[129] N. Carlini and D. Wagner, « Towards Evaluating the Robustness of Neural Networks », *in 2017 IEEE Symposium on Security and Privacy (SP)*, 2017.

[130] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, « Towards Deep Learning Models Resistant to Adversarial Attacks », *in International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=rJzIBfZAb.

[131] M. Pintor, F. Roli, W. Brendel, and B. Biggio, « Fast Minimum-norm Adversarial Attacks through Adaptive Norm Constraints », 2021.

[132] R. Gao, F. Liu, J. Zhang, B. Han, T. Liu, G. Niu, and M. Sugiyama, « Maximum Mean Discrepancy Test is Aware of Adversarial Attacks », *in Proceedings of the 38th International Conference on Machine Learning*, M. Meila and T. Zhang, Eds., ser. Proceedings of Machine Learning Research, vol. 139, PMLR, 2021, pp. 3564–3575. [Online]. Available: https://proceedings.mlr.press/v139/gao21b.html.

[133] R. Sahay, R. Mahfuz, and A. E. Gamal, « Combatting Adversarial Attacks through Denoising and Dimensionality Reduction: A Cascaded Autoencoder Approach », *in 2019 53rd Annual Conference on Information Sciences and Systems (CISS)*, 2019, pp. 1–6. DOI: 10.1109/CISS.2019.8692918.

[134] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, « Towards Deep Learning Models Resistant to Adversarial Attacks », *in International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=rJzIBfZAb.

[135] H. Gouk, E. Frank, B. Pfahringer, and M. J. Cree, « Regularisation of neural networks by enforcing Lipschitz continuity », *Mach. Learn.*, vol. 110, *2*, 393–416, 2021, ISSN: 0885-6125. DOI: 10.1007/s10994-020-05929-w. [Online]. Available: https://doi.org/10.1007/s10994-020-05929-w.

[136] M. Serrurier, F. Mamalet, A. González-Sanz, T. Boissin, J.-M. Loubes, and E. del Barrio, *Achieving robustness in classification using optimal transport with hinge regularization*, 2021. arXiv: 2006.06520 [cs.LG].

[137] M. Mirman, T. Gehr, and M. Vechev, « Differentiable Abstract Interpretation for Provably Robust Neural Networks », *in Proceedings of the 35th International Conference on Machine Learning*, J. Dy and A. Krause, Eds., ser. Proceedings of Machine Learning Research, vol. 80, PMLR, 2018, pp. 3578–3586. [Online]. Available: `https://proceedings.mlr.press/v80/mirman18b.html`.

[138] Y. Mao, M. N. Müller, M. Fischer, and M. Vechev, *Understanding Certified Training with Interval Bound Propagation*, 2023. arXiv: `2306.10426 [cs.LG]`.

[139] G. Singh, T. Gehr, M. Püschel, and M. Vechev, « An Abstract Domain for Certifying Neural Networks », *Proc. ACM Program. Lang.*, vol. 3, *POPL*, Jan. 2019. DOI: `10.1145/3290354`. [Online]. Available: `https://doi.org/10.1145/3290354`.

[140] E. Dohmatob, « Generalized No Free Lunch Theorem for Adversarial Robustness », *in Proceedings of the 36th International Conference on Machine Learning*, K. Chaudhuri and R. Salakhutdinov, Eds., ser. Proceedings of Machine Learning Research, vol. 97, PMLR, 2019, pp. 1646–1654. [Online]. Available: `https://proceedings.mlr.press/v97/dohmatob19a.html`.

[141] H. Zhang, Y. Yu, J. Jiao, E. Xing, L. E. Ghaoui, and M. Jordan, « Theoretically Principled Trade-off between Robustness and Accuracy », *in Proceedings of the 36th International Conference on Machine Learning*, K. Chaudhuri and R. Salakhutdinov, Eds., ser. Proceedings of Machine Learning Research, vol. 97, PMLR, 2019, pp. 7472–7482. [Online]. Available: `https://proceedings.mlr.press/v97/zhang19p.html`.

[142] M. Balunovic and M. Vechev, « Adversarial Training and Provable Defenses: Bridging the Gap », *in International Conference on Learning Representations*, 2020. [Online]. Available: `https://openreview.net/forum?id=SJxSDxrKDr`.

[143] G. Katz, C. Barrett, D. L. Dill, K. Julian, and M. J. Kochenderfer, « Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks », *in Computer Aided Verification*, R. Majumdar and V. Kunčak, Eds., Cham: Springer International Publishing, 2017, pp. 97–117, ISBN: 978-3-319-63387-9. [Online]. Available: `https://arxiv.org/abs/1312.6199`.

[144] N. Levy, R. Yerushalmi, and G. Katz, « gRoMA: A Tool for Measuring the Global Robustness of Deep Neural Networks », *in Bridging the Gap Between AI and Reality*, B. Steffen, Ed., Cham: Springer Nature Switzerland, 2024, pp. 160–170, ISBN: 978-3-031-46002-9.

[145] L. Weng, P.-Y. Chen, L. Nguyen, M. Squillante, A. Boopathy, I. Oseledets, and L. Daniel, « PROVEN: Verifying Robustness of Neural Networks with a Probabilistic Approach », *in Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 97, PMLR, 2019, pp. 6727–6736. [Online]. Available: `http://proceedings.mlr.press/v97/weng19a.html`.

[146] S. Webb, T. Rainforth, Y. W. Teh, and M. P. Kumar, « A Statistical Approach to Assessing Neural Network Robustness », *in International Conference on Learning Representations*, 2019.

[147] J. Cohen, E. Rosenfeld, and Z. Kolter, « Certified Adversarial Robustness via Randomized Smoothing », *in Proceedings of the 36th International Conference on Machine Learning*, K. Chaudhuri and R. Salakhutdinov, Eds., ser. Proceedings of Machine Learning Research, vol. 97, PMLR, 2019, pp. 1310–1320. [Online]. Available: `https://proceedings.mlr.press/v97/cohen19c.html`.

[148] T. Baluta, Z. L. Chua, K. S. Meel, and P. Saxena, « Scalable Quantitative Verification For Deep Neural Networks », *in Proc. of Int. Conf. on Software Engineering*, 2021. arXiv: `2002.06864 [cs.LG]`.

[149] P. Koopman and M. Wagner, « Autonomous Vehicle Safety: An Interdisciplinary Challenge », *IEEE Intelligent Transportation Systems Magazine*, vol. 9, *1*, pp. 90–96, 2017. DOI: `10.1109/MITS.2016.2583491`.

[150] A. Wald, « Sequential Tests of Statistical Hypotheses », *The Annals of Mathematical Statistics*, vol. 16, *2*, pp. 117–186, 1945, ISSN: 00034851. [Online]. Available: `http://www.jstor.org/stable/2235829` (visited on 02/09/2024).

[151] Y. Bai, Z. Huang, H. Lam, and D. Zhao, « Rare-event Simulation for Neural Network and Random Forest Predictors », *ACM Trans. Model. Comput. Simul.*, vol. 32, *3*, 2022, ISSN: 1049-3301. DOI: `10.1145/3519385`. [Online]. Available: `https://doi.org/10.1145/3519385`.

[152] G. Singh, T. Gehr, M. Mirman, M. Püschel, and M. Vechev, « Fast and Effective Robustness Certification », *in Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31, Curran Associates, Inc., 2018. [Online]. Available: `https://proceedings.neurips.cc/paper/2018/file/f2f446980d8e971ef3da97af089481c3-Paper.pdf`.

[153] C. A. Naesseth, F. Lindsten, and T. B. Schön, « Elements of Sequential Monte Carlo », *Foundations and Trends® in Machine Learning*, vol. 12, *3*, pp. 307–392, 2019, ISSN: 1935-8237. DOI: 10.1561/2200000074. [Online]. Available: http://dx.doi.org/10.1561/2200000074.

[154] R. E. Melchers and A. T. Beck, *Structural reliability analysis and prediction.* John wiley & sons, 2018.

[155] M. P. Owen, A. Panken, R. Moss, L. Alvarez, and C. Leeper, « ACAS Xu: Integrated Collision Avoidance and Detect and Avoid Capability for UAS », *in 2019 IEEE/AIAA 38th Digital Avionics Systems Conference (DASC)*, 2019, pp. 1–10. DOI: 10.1109/DASC43569.2019.9081758.

[156] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel, « Handwritten Digit Recognition with a Back-Propagation Network », *in Advances in Neural Information Processing Systems*, D. Touretzky, Ed., vol. 2, Morgan-Kaufmann, 1990. [Online]. Available: https://proceedings.neurips.cc/paper/1989/file/53c3bce66e43be4f209556518c2fcb54-Paper.pdf.

[157] J. Deng, K. Li, M. Do, H. Su, and L. Fei-Fei, « Construction and Analysis of a Large Scale Image Ontology », Vision Sciences Society, 2009.

[158] A. Foi, M. Trimeche, V. Katkovnik, and K. Egiazarian, « Practical Poissonian-Gaussian Noise Modeling and Fitting for Single-Image Raw-Data », *IEEE Transactions on Image Processing*, vol. 17, *10*, pp. 1737–1754, 2008. DOI: 10.1109/TIP.2008.2001399.

[159] D. Hendrycks and T. Dietterich, « Benchmarking Neural Network Robustness to Common Corruptions and Perturbations », *in International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=HJz6tiCqYm.

[160] J.-Y. Franceschi, A. Fawzi, and O. Fawzi, *Robustness of classifiers to uniform ℓ_p and Gaussian noise*, 2018. arXiv: 1802.07971 [cs.LG].

[161] M. Arief, Z. Huang, G. Koushik Senthil Kumar, Y. Bai, S. He, W. Ding, H. Lam, and D. Zhao, « Deep Probabilistic Accelerated Evaluation: A Robust Certifiable Rare-Event Simulation Methodology for Black-Box Safety-Critical Systems », *in Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, A. Banerjee and K. Fukumizu, Eds., ser. Proceedings of Machine Learning

Research, vol. 130, PMLR, 2021, pp. 595–603. [Online]. Available: `https://proceedings.mlr.press/v130/arief21a.html`.

[162] P. Del Moral, A. Doucet, and A. Jasra, « Sequential Monte-Carlo Samplers », *J. Roy. Stat. Soc. B*, vol. 68, *3*, pp. 411–436, 2006.

[163] A. Beskos, A. Jasra, N. Kantas, and A. Thiery, « On the convergence of adaptive sequential Monte Carlo methods », *The Annals of Applied Probability*, vol. 26, *2*, pp. 1111–1146, 2016.

[164] G. O. Roberts and R. L. Tweedie, « Exponential convergence of Langevin distributions and their discrete approximations », *Bernoulli*, pp. 341–363, 1996.

[165] H. Kahn and T. E. Harris, « Estimation of particle transmission by random sampling », *National Bureau of Standards applied mathematics series*, vol. 12, pp. 27–30, 1951.

[166] P. Del Moral, *Feynman-Kac Formulae, Genealogical and Interacting Particle Systems with Applications*, ser. Probability and its Applications. Springer, 2004.

[167] C. Jarzynski, « Equilibrium free-energy differences from nonequilibrium measurements: A master-equation approach », *Phys. Rev. E*, vol. 56, *5*, pp. 5018–5035, 1997.

[168] ——, « Nonequilibrium equality for free energy differences », *Phys. Rev. Lett.*, vol. 78, *14*, pp. 2690–2693, 1997.

[169] G. Stoltz, M. Rousset, *et al.*, *Free energy computations: A mathematical perspective.* World Scientific, 2010.

[170] A. Buchholz, N. Chopin, and P. E. Jacob, « Adaptive tuning of hamiltonian monte carlo within sequential monte carlo », *Bayesian Analysis*, vol. 16, *3*, pp. 745–771, 2021.

[171] P. Fearnhead and B. Taylor, « An Adaptive Sequential Monte Carlo Sampler », *Bayesian Analysis*, vol. 8, May 2010. DOI: `10.1214/13-BA814`.

[172] W. K. Hastings, « Monte Carlo Sampling Methods Using Markov Chains and Their Applications », *Biometrika*, vol. 57, *1*, pp. 97–109, 1970, ISSN: 00063444. [Online]. Available: `http://www.jstor.org/stable/2334940` (visited on 09/21/2022).

[173] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, « MobileNetV2: Inverted Residuals and Linear Bottlenecks », *in 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4510–4520. DOI: 10.1109/CVPR.2018.00474.

[174] K. Tit, T. Furon, and M. Rousset, « Gradient-Informed Neural Network Statistical Robustness Estimation », *in Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, F. Ruiz, J. Dy, and J.-W. van de Meent, Eds., ser. Proceedings of Machine Learning Research, vol. 206, PMLR, 2023, pp. 323–334. [Online]. Available: https://proceedings.mlr.press/v206/tit23a.html.

[175] D. Jacquemart-Tomi, J. Morio, and F. Le Gland, « A combined importance splitting and sampling algorithm for rare event estimation », *in 2013 Winter Simulations Conference (WSC)*, 2013, pp. 1035–1046. DOI: 10.1109/WSC.2013.6721493.

[176] K. Tit, T. Furon, M. Rousset, and L.-M. Traonouez, « Évaluation statistique efficace de la robustesse de classifieurs », *in CAID 2021 - Conference on Artificial Intelligence for Defense*, ser. Actes de la conférence CAID 2021, DGA, DGNUM, Ministère des Armées, Rennes, France, Nov. 2021, pp. 1–11. [Online]. Available: https://hal.science/hal-03462156.

[177] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, Second. The MIT Press, 2018. [Online]. Available: http://incompleteideas.net/book/the-book-2nd.html.

[178] S. C. Chan, S. Fishman, A. Korattikara, J. Canny, and S. Guadarrama, « Measuring the Reliability of Reinforcement Learning Algorithms », *in International Conference on Learning Representations*, 2020. [Online]. Available: https://openreview.net/forum?id=SJlpYJBKvH.

[179] M. Arief, Z. Cen, Z. Liu, Z. Huang, H. Lam, B. Li, and D. Zhao, *Test Against High-Dimensional Uncertainties: Accelerated Evaluation of Autonomous Vehicles with Deep Importance Sampling*, 2022. arXiv: 2204.02351 [cs.LG].

[180] G. Mårtensson, D. Ferreira, T. Granberg, L. Cavallin, K. Oppedal, A. Padovani, I. Rektorova, L. Bonanni, M. Pardini, M. G. Kramberger, J.-P. Taylor, J. Hort, J. Snædal, J. Kulisevsky, F. Blanc, A. Antonini, P. Mecocci, B. Vellas, M. Tsolaki, I. Kłoszewska, H. Soininen, S. Lovestone, A. Simmons, D. Aarsland, and E. Westman, « The reliability of a deep learning model in clinical out-of-distribution MRI data: A

multicohort study », *Medical Image Analysis*, vol. 66, p. 101 714, 2020, ISSN: 1361-8415. DOI: `https://doi.org/10.1016/j.media.2020.101714`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S1361841520300785`.

[181] D. J. Rezende and S. Mohamed, « Variational inference with normalizing flows », *in Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ser. ICML'15, Lille, France: JMLR.org, 2015, 1530–1538.

[182] M. Hairer and J. C. Mattingly, « Yet another look at Harris' ergodic theorem for Markov chains », *in Seminar on Stochastic Analysis, Random Fields and Applications VI*, Springer, 2011, pp. 109–117.

Titre : Fiabilité du Deep Learning avec simulation d'événements rares : théorie et pratique.

Mot clés : réseaux de neurones profonds ; simulation d'événements rares ; Monte Carlo séquentiel ; ingénierie statistique de fiabilité

Résumé : Cette thèse étudie la fiabilité des réseaux de neurones profonds en utilisant des algorithmes de simulation d'événements rares dans le cadre de l'ingénierie de la fiabilité statistique. L'objectif est d'évaluer la robustesse de ces réseaux dans des situations peu communes mais cruciales. La recherche se concentre sur le développement de nouvelles méthodes statistiques spécifiquement pour les réseaux de neurones profonds. Ces méthodes sont conçues pour mieux comprendre comment ces réseaux se comportent face à des données inhabituelles ou corrompues. Une réalisation clé est la création de nouveaux algorithmes qui améliore l'applicabilité des techniques de simulations d'événements rares aux classificateurs différentiables, une caractéristique commune dans les modèlesmodernes d'apprentissage profond. L'étude met en évidence les difficultés d'application des méthodes traditionnelles de fiabilité sta tistique aux données complexes et de grande dimension typiques en apprentissage profond. Malgré ces défis, les résultats offrent des outils et des approches qui peuvent être appliqués à divers modèles d'apprentissage profond.

Title: Reliability of Deep Learning with Rare Event Simulation: Theory and Practice.

Keywords: Deep Neural Networks; Rare Event Simulation; Sequential Monte Carlo; Statistical Reliability Engineering

Abstract: This thesis studies the reliability of deep neural networks using rare-event simulation algorithms in the context of statistical reliability engineering. The aim is to assess the robustness of these networks in uncommon but crucial situations. The research focuses on the development of new statistical methods specifically for deep neural networks. These methods are designed to better understand how these networks behave in the face of unusual or corrupted data. A key achievement is the creation of new algorithms that improve the applicability of rare event simulation techniques to differentiable classifiers, a common feature in modern deep learning models. The study highlights the difficulties of applying traditional static reliability methods to the complex, high-dimensional data typical of deep learning. Despite these challenges, the results offer tools and approaches that can be applied to a variety of deep learning models.