Vignesh Sivaramakrishnan

Candidate

Electrical and Computer Engineering

Department

This dissertation is approved, and it is acceptable in quality and form for publication:

Approved by the Dissertation Committee:

Meeko M.K. Oishi, Chairperson

Rafael Fierro

Ali Bidram

Claus Danielson

Panagiotis Tsiotras

Sean Phillips

Theory and algorithms to learn, propagate, and exploit uncertainty for stochastic optimal control of dynamical systems

by

Vignesh Sivaramakrishnan

B.S. Mechanical Engineering, The University of Utah, 2017

DISSERTATION

Submitted in Partial Fulfillment of the

Requirements for the Degree of

Doctorate of Philosophy

Engineering

The University of New Mexico

Albuquerque, NM

December 2024

Dedication

In memory of my first guru and Appa (father),

Veda Acharya Agoram Sivaramakrishnan

(1968-2024)

Acknowledgments

This excursion would have never started if my advisor, Meeko Oishi, was not willing to take me on as her student. I am truly grateful for her support, enthusiasm, and patience. There are countless number of mentors I must thank who made time for me in their busy schedules. Their feedback is a cornerstone for making many contributions possible. Professor Panagiotis Tsiotras, thank you for all the feedback that has made this work possible. My deepest thanks goes to Professor Cristina Pereyra for solidifying my appreciation for mathematics, especially Fourier and Harmonic Analysis.

To my lab-mates and collaborators, I could not have asked for a better cohort. I am indebted to their insights and camaraderie. To list them all here would be disservice, especially if I forget anyone. So, it is only right I thank everyone in person.

To my mentors at JPL, Dr. Jeffrey Umland, Dr. Stuart Shaklan, and Dr. K. Balasubramanian: I am thankful for their early mentorship and steering me onto this path. Without them this journey would not have even started.

Dr. Michael A. Kapamajian, thank you for treating me over the past four years. You have been a contributing factor not only to my health but my success. I do not take my eyesight for granted now.

To friends, thank you for being kind, understanding, and supportive. I am forever in your debt for the countless conversations, shared meals, car rides, coffee runs, and doctor visits. There are so many of you that have helped me and my family so it's only right I thank you in person and I do so countless times over.

Paati, thank you for helping Appa while Karthik and I were studying or Amma was at work. He was very thankful that you cooked food and had his things ready as he rushed out the door. Karthik, you're awesome...that's all I can say. Amma, thank you for always being there for Karthik, Appa, and me, no matter the hour, and for always listening. Keerthika, I really don't know where I'd be without you, but I hope you continue scolding me, making me cry, and being my voice of reason.

Last but not least, Appa, I don't know why you had to go so soon, but your endless kindness and ability to solve most, if not all, of my problems is a debt I can never repay, no matter how much I earn or how much I help others. You gave me the assurance that things will get better. Sure enough, it always does, one way or another.

This material is based upon work supported by the National Science Foundation under NSF Grant Number CNS-1836900, CMMI-2105631, by the Air Force Research Lab under Grant No. FA9453-23-C-A025, and by NASA under the University Leadership Initiative award #80NSSC20M0163. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation, the Air Force, or NASA.

Theory and algorithms to learn, propagate, and exploit uncertainty for stochastic optimal control of dynamical systems

by

Vignesh Sivaramakrishnan

B.S. Mechanical Engineering, The University of Utah, 2017

Ph.D Electrical Engineering, The University of New Mexico, 2024

Abstract

Non-Gaussian uncertainty frequently arises in learning and control problems involving stochastic dynamical systems, particularly in autonomous vehicles, UAVs, satellites, and robotics. In this dissertation, we propose a new framework that leverages characteristic functions that provides a frequency-domain representation of random variables. The dissertation is structured into three key areas. First, we address model-based stochastic optimal control for linear systems with non-Gaussian noise, demonstrating that characteristic functions can be used to enforce chance constraints and control systems toward desired distributions. Second, we explore data-driven stochastic control, utilizing empirical characteristic functions to handle systems with unknown disturbances. Additionally, we derive several metrics of the cost distribution through characteristic functions, facilitating further exploration in reinforcement learning. Finally, we utilize characteristic functions in neural network verification by propagating by propagating distributions through ReLU activation functions. While this analytical propagation shows promise, we reveal its limitations in higher dimensions and propose a sampling-based approach to verification that maintains guarantees. The core novelty of this dissertation is the creation of a set of mathematical tools and methods that can be used to address difficult problems in stochastic optimal control, neural net verification, and reinforcement learning. These methods and tools are designed to facilitate learning, propagation, and exploitation of uncertainty in autonomous dynamical system, well beyond state-of-the-art approaches.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

The growing complexity of modern systems leads inevitably to increased uncertainty, despite a concurrent, growing need for assurances of safe and effective operation. Consider self-driving cars, autonomous satellite navigation, hypersonic vehicle guidance, UAV swarms, and other scenarios, in which uncertainty in the environment is non-trivial. Uncertainty can arise not only due to a lack of analytic characterization (such as in the dynamics of hypersonic vehicles [8, 9]), but also due to poor sensing (i.e., navigation in cislunar space [10]), unmodeled disturbances (pedestrian interaction with self-driving cars), and human input, amongst many other sources. Irrespective of the source of the uncertainty, there is a clear need for methodological approaches to accommodate uncertainty in both analysis and controller design.

This thesis posits that it is imperative that we design algorithms to control stochastic systems but not limit ourselves in the information we can extract from the underlying uncertainty. However, numerous challenges exist in both theory and computation for learning, propagating, and ultimately, exploiting uncertainty in dynamical systems. For example, consider a typical stochastic optimal control problem,

$$\begin{aligned}
\underset{\theta}{\text{minimize}} \quad & \mathbb{E}\left[\sum_{k=0}^{N-1} c(\mathbf{x}_k, \mathbf{u}_k) + g(\mathbf{x}_N, \mathbf{u}_N)\right] && \text{cost on state and input,,} && \text{(1.1a)}\\
\text{subject to} \quad & \mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k, \mathbf{w}_k) && \text{system dynamics,} && \text{(1.1b)}\\
& \mathbf{u}_k = g_\theta(\mathbf{x}_k),\ k \in \{0, \cdots, N-1\} && \text{controller,} && \text{(1.1c)}\\
& \mathbb{P}\left(\bigcap_{k=1}^{N} \mathbf{x}_k \in X_k\right) \geq 1 - \Delta_{\mathbf{x}} && \text{state constraints,} && \text{(1.1d)}\\
& \mathbb{P}\left(\bigcap_{k=0}^{N-1} \mathbf{u}_k \in U_k\right) \geq 1 - \Delta_{\mathbf{u}} && \text{input constraints,} && \text{(1.1e)}
\end{aligned}$$

where we must minimize some cost (1.1a), subject to the system dynamics (1.1b) and be mindful of constraints on the state (1.1d) and input (1.1e) of the system. One can achieve this minimization by modifying the parameters, θ, of the controller (1.1c). The controller can be a function of the state, a camera input [11], or remain open-loop. The key takeaway is the number of quantities that the uncertainty in (1.1b) spawns. For example, both (1.1d) and (1.1e) require enforcement and evaluation of probabilistic quantities of state and input respectively. That is, does the probability of the state and input constraint being satisfied exceed some tolerance $1-\Delta_{\mathbf{x}}$ and $1-\Delta_{\mathbf{u}}$ respectively. The cost in (1.1a) requires one to minimize the average of the cost. To be successful in solving (1.1), we must efficiently compute and optimize over the quantities the uncertainty spawns. A major challenge in reasoning about uncertainty lies in the choice of representation. We seek a representation which has the following properties:

- Always exists
- Tractable propagation of uncertainty through linear operations where $\mathbf{x} \in \mathbb{R}^n,\ A \in \mathbb{R}^{m\times n},\ \mathbf{y}, b \in \mathbb{R}^m$:

$$\mathbf{y} = A\mathbf{x} + b$$

- Tractable computation of expectations (where $\psi_{\mathbf{x}}$ is a probability density function):

$$\mathbb{E}[\mathbf{x}] = \int x\psi_{\mathbf{x}}(x)\mathrm{d}x$$

- Tractable computation of probabilities (where $1_X(x)$ is an indicator function):

$$\mathbb{P}(\mathbf{x} \in X) = \int 1_X(x)\psi_{\mathbf{x}}(x)\mathrm{d}x$$

- Limit use of quadrature to one-dimensional problems.

Take probability density functions and moment generating functions, for example. Both of these representations are problematic, in that they do not satisfy all of the above properties. Consider a probability density function, ψ_x.

– Don't always exist.

– Linear transformations involve convolution integrals.

$$\mathbf{x} = \mathbf{x}_1 + \mathbf{x}_2 \Leftrightarrow \psi_{\mathbf{x}}(x) = \int \psi_{\mathbf{x}_1}(y)\psi_{\mathbf{x}_2}(x-y)\mathrm{d}y$$

+ Expression for probabilities are one dimensional integrals if $\mathbf{z} \in \mathbb{R}$.

$$\mathbb{P}(\mathbf{z} \leq z) = \int_{-\infty}^{z} \psi_{\mathbf{z}}(z)\mathrm{d}z$$

Now consider a moment generating function $M_x(s)$, which can be rewritten as $\mathbb{E}[\exp(s^\intercal \mathbf{x})] = \int \exp(s^\intercal x)\psi_{\mathbf{x}}(x)\mathrm{d}s$:

– don't always exist either

+ Linear transformations are product operations, in contrast to needing convolution integrals.

$$\mathbf{y} = A\mathbf{x} + b \Leftrightarrow \varphi_{\mathbf{y}}(t) = M_{\mathbf{x}}(A^{\intercal}t)\exp(s^{\intercal}b)$$

+ We can compute Expectations from its derivatives via automatic differentiation []:

$$\mathbb{E}[\mathbf{x}] = \frac{\mathrm{d}M_{\mathbf{x}}}{\mathrm{d}s}\bigg|_{s=0}$$

In short, the two primary existing methods of describing uncertainty are fraught with respect to the properties necessary for efficient computation and therefore efficient control. In order to make headway on the problems we seek to solve, a new representation is necessary. We focus in particular on the characteristic function, which provides clear advantages for each of the four desired properties. The characteristic function is a Fourier transform of a probability density function,

$$\varphi_{\mathbf{x}}(t) = \mathbb{E}[\exp(\mathrm{i}t^{\intercal}\mathbf{x})] = \int \exp(\mathrm{i}t^{\intercal}x)\psi_{\mathbf{x}}(x)\mathrm{d}t. \tag{1.2}$$

The characteristic function of a probability density function meets all of the criteria we have specified:

+ It always exists.

+ Linear transformations are also product operations, in contrast to needing convolution integrals.

$$\mathbf{y} = A\mathbf{x} + b \Leftrightarrow \varphi_{\mathbf{y}}(t) = \varphi_{\mathbf{x}}(A^{\intercal}t)\exp(\mathrm{j}t^{\intercal}b),$$

+ We can compute Expectations from its derivatives via automatic differentiation []:

$$\mathbb{E}[\mathbf{x}] = \frac{\mathrm{d}\varphi_{\mathbf{x}}}{\mathrm{d}t}\Big|_{t=0},$$

+ Expressions for probabilities are one dimensional integrals [12].

$$\mathbb{P}(\mathbf{z} \leq z) = \frac{1}{2} + \frac{1}{2\pi}\int_0^\infty \frac{e^{\mathrm{j}tz}\varphi_{\mathbf{z}}(-t) - e^{-\mathrm{j}tz}\varphi_{\mathbf{z}}(t)}{\mathrm{j}t}\mathrm{d}t.$$

However, beyond stochastic optimal control, we have found that these properties of characteristic functions make them amenable solutions to other related problems. Specifically, we consider 1) neural net verification, and 2) reinforcement learning via cost distributions. In neural net verification, we can conceive of the input to a neural net as a *distribution*, instead of a single sample. Neural network verification (Figure 1.1) is

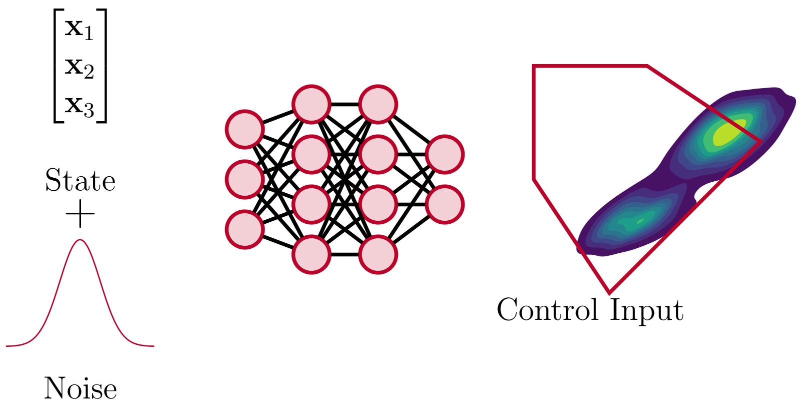

$\begin{bmatrix}\mathbf{x}_1\\\mathbf{x}_2\\\mathbf{x}_3\end{bmatrix}$
State
+
Noise
Control Input

Figure 1.1: A pictorial representation of the neural network verification problem where we provide a noisy esimtate of the state into a neural network controller and we wish to determine with what probability the control output from the neural network resides within a set.

then focused on the question of whether the output distribution of the neural net will fall within some desirable set, with at least a desired likelihood. The advantage of such an approach is that it is computationally efficient, and does not rely upon brute-force methods, such as sampling, which make trade-offs between fidelity of the result and computational complexity. These same properties also make evaluation of cost functions in reinforcement learning more responsive to the underlying uncertainty than existing approaches, particularly when the cost function represents value-at-risk or conditional-value-at-risk (metrics which involve more than the mean). In contrast to these approaches, the framework posed in this thesis relies on the representation of uncertainty through characteristic functions. Here, we show that we can compute and optimize the expressions necessary for difficult problems in stochastic optimal control, Further, we demonstrate how these same methods can be applied to timely and relevant problems in autonomous systems, including neural net verification and distributional reinforcement learning.

1.2 Summary of research contributions

This dissertation has three parts. The first part focuses on model-based stochastic optimal control, with a known system model, state, and control constraints.

- Chapter 3 presents a sampling-free approach to obtaining open-loop control solutions for constrained, stochastic optimal control problems for linear dynamical systems subject to log-concave, non-Gaussian noise.
- Chapter 4 presents a method for distributional steering of linear dynamical systems subject to general additive noise in addition to state and input constraints.

The second part focuses on data-driven, stochastic control in which the disturbance or the system even the entire dynamical is unknown.

- Chapter 5 presents a method for stochastic optimal control for linear dynamical systems with state and input constraints when the disturbance is unknown, with sample theoretic guarantees.
- Chapter 6 derives various performance metrics for reinforcement learning, utilizing a distributional representation of the cost via characteristic functions. This chapter

derives not only the mean but also Value-at-Risk, Conditional-Value-at-Risk, and expectiles.

The third part focuses on probabilistic verification of neural networks through analytical and sample-based approaches.

- Chapter 7 presents results on analytically propagating distributions through a ReLU activation function and applies the results to neural network verification of feedforward neural network with ReLU activation functions.
- Chapter 8 overviews the curse of dimensionality with analytic approaches and proposes a sample-based approach for verifying neural networks, determining the number of samples necessary to validate the probability of satisfying a specification.

1.3 Summary of publications

The content of Chapter 3, which I have contributed as a first author, appears in:

> [13] V. Sivaramakrishnan, A. P. Vinod, and M. M. K. Oishi, "Convexified open-loop stochastic optimal control for linear systems with log-concave disturbances," *IEEE Transactions on Automatic Control*, vol. 69, no. 2, pp. 1249–1256, 2024

The content of Chapter 4, which I have contributed as a first author, appears in:

> [14] V. Sivaramakrishnan, J. Pilipovsky, M. Oishi, and P. Tsiotras, "Distribution steering for discrete-time linear systems with general disturbances using characteristic functions," in *the Proceedings of the 2022 American Control Conference (ACC)*, pp. 4183–4190, 2022

The content of Chapter 5, which I have contributed as a first author, appears in:

> [15] V. Sivaramakrishnan and M. M. K. Oishi, "Fast, convexified stochastic optimal open-loop control for linear systems using empirical characteristic functions," *IEEE Control Systems Letters*, vol. 4, no. 4, pp. 1048–1053, 2020

The content of Chapter 7, for which I was primarily responsible for the theoretical contributions, appears in:

> [16] J. Pilipovsky, V. Sivaramakrishnan, M. Oishi, and P. Tsiotras, "Probabilistic verification of relu neural networks via characteristic functions," in *Proceedings of The 5th Annual Learning for Dynamics and Control Conference*, vol. 211 of *the Proceedings of Machine Learning Research*, pp. 966–979, PMLR, 15–16 Jun 2023

Other papers that I have co-authored during my graduate studies, but which are not covered in this dissertation, include:

> [17] V. Sivaramakrishnan, R. A. Devonport, M. Arcak, and M. M. K. Oishi, "Forward reachability for discrete-time nonlinear stochastic systems via mixed-monotonicity and stochastic order," 2024. (To appear the Proceedings of the 2024 Conference on Decision and Control (CDC))
>
> [18] K. Sivaramakrishnan, V. Sivaramakrishnan, R. A. Devonport, and M. M. K. Oishi, "Stochastic reachability of uncontrolled systems via probability measures: Approximation via deep neural networks," 2024. (To appear the Proceedings of the 2024 Conference on Decision and Control (CDC))

[19] I. Pacula, A. Vinod, V. Sivaramakrishnan, C. Petersen, and M. Oishi, "Stochastic multi-satellite maneuvering with constraints in an elliptical orbit," in *2021 American Control Conference (ACC)*, pp. 4261–4268, 2021

[20] A. J. Thorpe, V. Sivaramakrishnan, and M. M. K. Oishi, "Approximate stochastic reachability for high dimensional systems," in *the Proceedings of the 2021 American Control Conference (ACC)*, pp. 1287–1293, 2021

[21] V. Sivaramakrishnan, O. Thapliyal, A. Vinod, M. Oishi, and I. Hwang, "Predicting mode confusion through mixed integer linear programming," in *the Proceedings of the 2019 IEEE 58th Conference on Decision and Control (CDC)*, pp. 2442–2448, 2019

[22] A. P. Vinod, V. Sivaramakrishnan, and M. M. Oishi, "Piecewise-affine approximation-based stochastic optimal control with gaussian joint chance constraints," in *the Proceedings of the 2019 American Control Conference (ACC)*, pp. 2942–2949, 2019

[23] A. P. Vinod, V. Sivaramakrishnan, and M. M. K. Oishi, "Sampling-free enforcement of non-gaussian chance constraints via fourier transforms," Proceedings of the Fifth International Workshop on Symbolic-Numeric methods for Reasoning about CPS and IoT, p. 9–11, Association for Computing Machinery, 2019

[24] A. Abate, H. Blom, N. Cauchi, S. Haesaert, A. Hartmanns, K. Lesser, M. Oishi, V. Sivaramakrishnan, S. Soudjani, C.-I. Vasile, and A. P. Vinod, "Arch-comp18 category report: Stochastic modelling," in *ARCH18. 5th International Workshop on Applied Verification of Continuous and Hybrid Systems*, vol. 54 of *EPiC Series in Computing*, pp. 71–103, EasyChair, 2018

Chapter 2

Preliminaries

This chapter provides a broad overview of notation, probability theory, discrete-time stochastic systems, and optimization. rIt also covers elements common across chapters, such as characteristic functions from probability theory and convex, difference-of-convex, and non-convex optimization from optimization theory. Each subsequent chapter includes additional preliminaries as needed for further exposition.

2.1 Notation

Real-valued vectors are lowercase $u \in \mathbb{R}^m$, matrices with uppercase $V \in \mathbb{R}^{n\times m}$, and random vectors are in bold case $\mathbf{w} \in \mathbb{R}^p$. The n-dimension identity matrix is denoted by I_n, and the $m \times n$ dimensional zero matrix is denoted by $0_{m,n}$. We define a diagonal matrix as $V = \mathrm{diag}(u)$ and a block diagonal matrix as $V = \mathrm{diag}(V_1, \cdots, V_i \cdots, V_n)$. The imaginary unit is denoted by i; given a complex vector $\varphi \in \mathbb{C}^p$, its conjugate is denoted by $\overline{\varphi}$. I denote intervals with $\mathbb{N}_{[a,b]}$ where $a, b \in \mathbb{N}$, $a < b$. The vector $e_{i,d} = [0\ \cdots\ 1\ \cdots\ 0]^\intercal \in \mathbb{R}^d$ is a basis vector for $\mathbb{R}^d$ and isolates the ith component of a vector $\psi \in \mathbb{R}^d$ by $\psi_i = e_{i,d}^\intercal \psi$.

2.2 Probability

Let $(\Omega, \mathcal{M}(\Omega), P)$ be a probability tuple. The set Ω is the set of all possible outcomes, $\mathcal{M}(\Omega)$ is the set of events, i.e. σ-algebra, where each event is a set of outcomes, and a function $P : \mathcal{M}(\Omega) \to [0,1]$ which assigns a probability to each set in the σ-algebra. A measurable space is a tuple $(\mathcal{X}, \mathcal{M}(\mathcal{X}))$ consisting of a set and the σ-algebra of that set. A random variable is a measurable function, $\mathbf{s} : \Omega \to \mathcal{X}$ where the probability that $\mathbf{s}$ will take on a value in S we represent by a probability measure, $\mathbb{P}_{\mathbf{s}}(S) = P\left(\{\omega \in \Omega : \mathbf{s}(\omega) \in S\}\right)$ for $S \in \mathcal{M}(\mathcal{X})$. We denote a conditional probability measure as a mapping $P : \mathcal{M}(\mathcal{X}) \times \mathcal{X} \times \mathcal{U}$. The probability measure is $P(S|s, a)$ for $S \in \mathcal{M}(\mathcal{X})$ conditioned on $s \in \mathcal{X}$ and $u \in \mathcal{U}$, where $\mathcal{X}$ and $\mathcal{U}$ are sets. An expectation is the Lebesgue integral over the probability measure, i.e. $\mathbb{E}[g(\mathbf{s})] = \int_{\mathcal{X}} g(s) \mathrm{d}\mathbb{P}_{\mathbf{s}}(s)$. For continuous random variables, $\mathbf{w}$, with probability measure $\mathbb{P}\left(\{\mathbf{w} \in \mathcal{W}\}\right) = \int_{\mathcal{W}} \psi_{\mathbf{w}}(z)\, \mathrm{d}z$ for $\mathcal{X} \in \mathscr{B}(\Omega)$, and probability density function (pdf) $\psi_{\mathbf{w}}$ that satisfies $\psi_{\mathbf{w}} \geq 0$ almost everywhere (a.e.) and $\int_{\mathbb{R}} \psi_{\mathbf{w}}(z)\, \mathrm{d}z = 1$. For a random variable, e.g. $\mathbf{y} = a^\intercal \mathbf{w}$, $a \in \mathbb{R}^p$, we denote $\mathbb{P}\{a^\intercal \mathbf{w} \leq \alpha\}$ by the cumulative distribution function (cdf) $\Phi_{a^\intercal \mathbf{w}} : \mathbb{R} \to [0,1]$ via $\mathbb{P}\{a^\intercal \mathbf{w} \leq \alpha\} = \Phi_{a^\intercal \mathbf{w}}(\alpha)$, which follows by definition [25, Sec. 14]. We write $\mathbf{w} \sim \psi_{\mathbf{w}}$ or $\mathbf{w} \sim \mathbb{P}_{\mathbf{w}}$ to denote the fact that $\mathbf{w}$ is distributed according to the pdf, $\psi_{\mathbf{w}}$, or probability measure, $\mathbf{w} \sim \mathbb{P}_{\mathbf{w}}$, respectively. We define the Lebesgue space of measurable pdfs with bounded d-norm by $L^d(\mathbb{R}^n)$ where $1 \leq d < \infty$. The Lebesgue norm of a probability density function $\psi_{\mathbf{w}}$ is $\|\psi_{\mathbf{w}}\|_d = \left(\int_{\mathbb{R}^p} |\psi_{\mathbf{w}}(z)|^d \,\mathrm{d}z\right)^{1/d}$. The space of all (continuous) probability density functions forms a subset of $L_1(\mathbb{R}^p)$ since $\psi_{\mathbf{w}} \geq 0$ a.e. and $\int_{\mathbb{R}^p} \psi_{\mathbf{w}}\, \mathrm{d}z = 1$.

2.2.1 Characteristic Functions

One way to represent the underlying system stochasticity is via characteristic functions.

Definition 2.1. *For a random vector $\mathbf{w} \in \mathcal{W}$ such that $\mathbf{w} \sim \mathbb{P}_{\mathbf{w}}$ or $\mathbf{w} \sim \psi_{\mathbf{w}}$, the* *characteristic function is defined by the Fourier transform $\mathcal{F}\{\psi_{\mathbf{w}}\}(t)$ of its pdf,*

$$\varphi_{\mathbf{w}}(t) = \mathbb{E}_{\mathbf{w}}[\exp(\mathrm{i}t^{\intercal}\mathbf{w})] = \int_{\mathcal{W}} e^{\mathrm{i}t^{\intercal}z}\mathrm{d}\mathbb{P}_{\mathbf{w}}(z), \tag{2.1a}$$

$$= \int_{\mathcal{W}} e^{\mathrm{i}t^{\intercal}z}\psi_{\mathbf{w}}(z)\,\mathrm{d}z, \tag{2.1b}$$

where $t, z \in \mathbb{R}^p$.

The characteristic function has the following properties [26, 27]:

- It is uniformly continuous.
- $\varphi_{\mathbf{w}}(0) = 1$.
- It is bounded, i.e., $|\varphi_{\mathbf{w}}(t)| \leq 1$, for all $t \in \mathbb{R}^p$.
- It is Hermitian, i.e., $\varphi_{\mathbf{w}}(-t) = \overline{\varphi}_{\mathbf{w}}(t)$.

Assumption 2.1. *The characteristic function $\varphi_{\mathbf{w}}$ is absolutely integrable, that is, it is an element of $L_1(\mathbb{R}^p)$.*

To recover the pdf from its characteristic function of a continuous random variable, $\psi_{\mathbf{w}}$, we use the following result.

Theorem 2.1 (Inversion Theorem for pdfs, [27, Theorem 1.2.6])**.** *If the characteristic function $\varphi_{\mathbf{w}} \in L_1(\mathbb{R}^p)$, then the probability density function can be recovered via the inverse Fourier transform $\mathcal{F}^{-1}\{\varphi_{\mathbf{w}}\}(z)$,*

$$\psi_{\mathbf{w}}(z) = \left(\frac{1}{2\pi}\right)^p \int_{\mathbb{R}^p} e^{-\mathrm{i}t^{\intercal}z}\varphi_{\mathbf{w}}(t)\,\mathrm{d}t. \tag{2.2}$$

Below, we summarize useful properties of characteristic functions. Let $\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}, \mathbf{z}$ be random vectors of appropriate dimensions.

1. If $\mathbf{z} = \mathbf{w}_1 + \mathbf{w}_2$, then $\psi_{\mathbf{z}}(z) = \big(\psi_{\mathbf{w}_1} * \psi_{\mathbf{w}_2}\big)(z)$ (i.e., convolution of their pdfs), and $\varphi_{\mathbf{z}}(t) = \varphi_{\mathbf{w}_1}(t)\varphi_{\mathbf{w}_2}(t)$ [28, Sec. 21.11].

2. If $\mathbf{z} = F\mathbf{w} + g$ for $F \in \mathbb{R}^{n\times p}$, $g \in \mathbb{R}^n$, then $\varphi_{\mathbf{z}}(t) = \exp(\mathrm{i}t^\intercal g)\varphi_{\mathbf{w}}(F^\intercal t)$ [28, Sec. 22.6].

3. Given $\mathbf{w}_1$ and $\mathbf{w}_2$, then $\mathbf{z} = [\mathbf{w}_1^\intercal, \mathbf{w}_2^\intercal]^\intercal$ has the pdf $\psi_{\mathbf{z}}(z) = \psi_{\mathbf{w}_1}(e_1^\intercal z_1)\psi_{\mathbf{w}_2}(e_2^\intercal z_2)$, $z = [z_1^\intercal, z_2^\intercal]^\intercal$, and characteristic function $\varphi_{\mathbf{z}}(t) = \varphi_{\mathbf{w}_1}(e_1^\intercal t)\varphi_{\mathbf{w}_2}(e_2^\intercal t)$, $t = [t_1^\intercal, t_2^\intercal]^\intercal$, where e_1 and e_2 isolate the first and second component of the vector, respectively [28, Sec. 22.4].

4. If $\mathbf{z} = [\mathbf{z}_1 \ \cdots \ \mathbf{z}_i \ \cdots \ \mathbf{z}_p]^\intercal \in \mathbb{R}^p$ is a vector of scalar random variables $\mathbf{z}_i$ with pdfs $\psi_{\mathbf{z}_i}$, then the pdf of $a^\intercal \mathbf{z}$, $a \in \mathbb{R}^p$, is $\psi_{a^\intercal\mathbf{z}}(z) = \prod_{i=1}^p \psi_{\mathbf{z}_i}(e_{i,p}^\intercal a z)$, and the characteristic function is $\varphi_{a^\intercal\mathbf{z}}(t) = \varphi_{\mathbf{z}}(t_{\mathbf{z}}) = \prod_{i=1}^p \varphi_{\mathbf{z}_i}(t_i)$, for $t_{\mathbf{z}} = at$, and $t_i = e_{i,p}^\intercal t_{\mathbf{z}}$ [28, Sec. 22.4].

Remark 2.1. *The characteristic function of a distribution always exists, even when the probability density function or moment-generating function do not exist.*

We can also recover the cdf via the characteristic function using the following theorem.

Theorem 2.2 (Gil-Pelaez Inversion Theorem, [12,27])**.** *Given a random variable* $\mathbf{y}$ *with characteristic function* $\varphi_{\mathbf{y}}$ *and pdf* $\psi_{\mathbf{y}}$ *satisfying the property that* $\int_{\mathbb{R}} \log(1+|x|)\psi_{\mathbf{x}}(z)\mathrm{d}z < \infty$*, then the cumulative distribution function of* $\mathbf{y}$*,* $\Phi_{\mathbf{y}}$*, at each point* y *that is continuous, can be evaluated by*

$$\Phi_{\mathbf{y}}(y) = \frac{1}{2} - \frac{1}{\pi}\int_0^\infty \frac{1}{t}\mathrm{Im}\left[\exp\left(-\mathrm{i}ty\right)\varphi_{\mathbf{y}}(t)\right]\,\mathrm{d}t, \tag{2.3}$$

where $y, t \in \mathbb{R}$*.*

The inversion in (2.3) is computable using quadrature techniques which have well defined error bounds [29].

Remark 2.2. *The requirement* $\int_{\mathbb{R}} \log(1+|x|)\psi_{\mathbf{x}}(z)\mathrm{d}z < \infty$ *is a mild condition which is satisfied by many distributions [30].*

Definition 2.2. *The* d^{th} *moment of* $\mathbf{y} \in \mathbb{R}$ *can be written as*

$$\mathbb{E}[\mathbf{y}^d] = (-i)^d \frac{\partial^d \varphi_{\mathbf{y}}(t)}{\partial t^d}\Big|_{t=0}. \tag{2.4}$$

which is extendable to random vectors and matrices but for sake of exposition, it is not included here.

2.3 Dynamical Systems

Here, a system is something we wish to observe and interact with. We call $x \in \mathcal{X}$ a state of a system where $\mathcal{X}$ is the set of states. We interact with a system via control inputs $u \in \mathcal{U}$ where $\mathcal{U}$ is the set of control inputs. The measurable spaces $(\mathcal{X}, \mathcal{M}(\mathcal{X}))$ and $(\mathcal{U}, \mathcal{M}(\mathcal{U}))$ are the state space and input space respectively. Typically, the state or input space is called discrete if the set of values the space takes is finite. On the other hand, it is called continuous, if it takes values in a continuum. We codify continuous or discrete state and input spaces in the following definitions.

Definition 2.3 (Continuous State and Input Spaces)**.** *We define a state and input space as continuous when the sets* $\mathcal{X} \subseteq \mathbb{R}^n$ *and* $\mathcal{U} \subseteq \mathbb{R}^m$ *are continuous subsets of real numbers. The* σ*-algebras in state and input spaces are respectively* $\mathcal{M}(\mathcal{X}) = \mathcal{B}(\mathcal{X})$ *and* $\mathcal{M}(\mathcal{U}) = \mathcal{B}(\mathcal{U})$*, where* $\mathcal{B}(\cdot)$ *is the Borel* σ*-algebras.*

Definition 2.4 (Discrete State and Input Spaces)**.** *We define a state and action space as discrete when the sets* $\mathcal{X}$ *and* $\mathcal{U}$ *are countably finite, i.e the cardinality of* $\mathcal{X}$ *and* $\mathcal{U}$ *are finite. The* σ*-algebras in state and action spaces are respectively* $\mathcal{M}(\mathcal{X}) = 2^{\mathcal{X}}$ *and* $\mathcal{M}(\mathcal{U}) = 2^{\mathcal{U}}$*, where* $2^{(\cdot)}$ *is a power set.*

We index observations of the state of a system and the actions one imposes by a timestep $k \in \mathbb{N}$, i.e. $x_k \in \mathcal{X}$ and $u_k \in \mathcal{U}$ respectively. Practically, given current state,

x_k, and action, u_k, we observe the state at the next time step, $\mathbf{x}_{k+1}$. This subsequent observation, $\mathbf{x}_{k+1}$, is given through a stochastic system function [31, Sec. 1.4].

Definition 2.5 (Stochastic System Function)**.** *We define the stochastic system,*

$$\mathbf{x}_{k+1} = f(x_k, u_k, \mathbf{w}(x_k, u_k)) = f(x_k, u_k, \mathbf{w}) \tag{2.5}$$

from the current state, action, and disturbance, to the next state.

The random disturbance variable $\mathbf{w}$ is a measurable mapping, conditioned on current state and action, $\mathbf{w} : \Omega \times \mathcal{X} \times \mathcal{U} \to \mathcal{W}$. The probability that $\mathbf{w}$ takes on a value in a set $W \in \mathcal{M}(\mathcal{W})$, given current state and action is,

$$\mathbb{P}_{\mathbf{w}}(W|x_t, u_t) = \mathbb{P}\left(\{\omega \in \Omega | \mathbf{w}(\omega|x_t, u_t) \in W\} | x_t, u_t\right). \tag{2.6}$$

As shorthand, we treat $\mathbf{w}(x_k, u_k) = \mathbf{w}(\omega|x_k, u_k)$ as the same function. In addition, the measurable space $(\mathcal{W}, \mathcal{M}(\mathcal{W}))$ we call the disturbance space. We now define the probability that the next state, $\mathbf{x}_{k+1}$ takes on a value in a set $X \in \mathcal{M}(\mathcal{X})$, given current state, x_k, action, u_k, and disturbance, $\mathbf{w}$.

Definition 2.6 (State Transition Kernel)**.** *The state transition kernel is a conditional probability function,* $\mathbb{P}_{\mathbf{x}_{k+1}} : \mathcal{M}(\mathcal{X}) \times \mathcal{X} \times \mathcal{U} \to [0, 1]$. *Specifically, we represent it by the disturbance* $\mathbf{w}$*,*

$$\mathbb{P}_{\mathbf{x}_{k+1}}(X|x_t, u_t) = \mathbb{P}_{\mathbf{w}}(\{\mathbf{w_t}(x_k, u_k) \in W | f(x_k, u_k, \mathbf{w}(x_k, u_k)) \in S\} | x_k, u_k), \tag{2.7}$$

$X \in \mathcal{M}(\mathcal{X})$, $W \in \mathcal{M}(\mathcal{W})$. This leads us to the definition of a discrete time stochastic dynamical system.

Definition 2.7 (DTSS)**.** *A discrete time stochastic system is a tuple* $\mathcal{D} = (\mathcal{X}, \mathcal{U}, \mathbb{P}_{\mathbf{x}_{k+1}})$.

Note that there are two properties of a DTSS we need to be careful about and we can extend to, but do not bother since they can be reduced down to a DTSS. The first property, which we implicitly presume, is that the system is Markov. The Markov property, Markov for short, presumes that describing the next state only depends on the current state and action, i.e.

$$\mathbb{P}_{\mathbf{x}_{k+1}}(X|x_k,u_k)=\mathbb{P}_{\mathbf{x}_{k+1}}(X|x_k,u_k,x_{k-1},u_{k-1},\dots,x_1,u_1,x_0,u_0). \tag{2.8}$$

The other property is that the system is stationary. The stationarity property of a system largely depends on the treatment of the underlying state, action, and disturbance sets. If we index the sets with time, i.e. $\mathcal{X}_t$, $\mathcal{U}_t$, and $\mathcal{W}_t$, then there is a non-stationary analog to Definition 2.7. Nonetheless, the stochastic optimal control literature typically recasts a non-stationary problem into one that is stationary via state augmentation [32, Ch. 10]. These properties are typically realized from observations of the system's trajectories, T.

Definition 2.8 (System Trajectory)**.** *A system trajectory is a set of tuples* $\mathrm{T}=\{(x_0,u_0,x_1),(x_1,u_1,x_2),\dots,\ (x_k,u_k,x_{t+1}),\dots,(x_{N-1},u_{N-1},x_N)\}$ *where* $N\in\mathbb{N}$ *is a time horizon,* $x_k,x_{k+1}\in\mathcal{X}$*, and* $u_k\in\mathcal{U}$*.*

2.3.1 Linear Dynamical System

For chapters 3, 4, and 5 we consider the system as a discrete, linear time-varying system,

$$\mathbf{x}_{k+1}=A_k\mathbf{x}_k+B_k\mathbf{u}_k+D_k\mathbf{w}_k,\quad k\in\mathbb{N}_{[0,N-1]}, \tag{2.9}$$

with state $\mathbf{x}_k\in\mathcal{X}_k\subseteq\mathbb{R}^n$, control input $\mathbf{u}_k\in\mathcal{U}_k\subseteq\mathbb{R}^m$, disturbance $\mathbf{w}_k\sim\psi_{\mathbf{w},k}$, and matrices $A_k,\ B_k,\ D_k$ of appropriate dimensions. We assume that the system starts at $\mathbf{x}_0\sim\psi_{\mathbf{x}_0}$. Following the formulations in [33, 34], we can concatenate the dynamics (2.9) as

$$\mathbf{X} = \mathcal{A}\mathbf{x}_0 + \mathcal{B}\mathbf{U} + \mathcal{D}\mathbf{W}, \tag{2.10}$$

where $\mathbf{X} = [\mathbf{x}_0^\intercal, \dots, \mathbf{x}_N^\intercal]^\intercal \in \mathbb{R}^{(N+1)n}$, $\mathbf{U} = [\mathbf{u}_0^\intercal, \dots, \mathbf{u}_{N-1}^\intercal]^\intercal \in \mathbb{R}^{mN}$, $\mathbf{W} = [\mathbf{w}_0^\intercal, \dots, \mathbf{w}_{N-1}^\intercal]^\intercal \in \mathbb{R}^{pN}$. The input can be stochastic, $\mathbf{u}_k$, or deterministic, u_k. The concatenated disturbance follows the distribution $\mathbf{W} \sim \psi_{\mathbf{W}} = \prod_{k=0}^{N-1} \psi_{\mathbf{w}_k}$. The matrices $\mathcal{A} \in \mathbb{R}^{n(N+1)\times n}$, $\mathcal{B} \in \mathbb{R}^{(N+1)n\times Nm}$, and $\mathcal{D} \in \mathbb{R}^{(N+1)n\times Np}$ are structured as follows,

$$\mathcal{A} = \begin{bmatrix} A_0^0 \\ A_0^1 \\ A_0^2 \\ \vdots \\ A_0^N \end{bmatrix} \mathcal{B} = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ A_1^1 B_) & 0 & \cdots & 0 \\ A_1^2 B_0 & A_2^2 B_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ A_1^N B_0 & A_2^N B_1 & \cdots & A_T^N B_{N-1} \end{bmatrix} \mathcal{D} = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ A_1^1 & 0 & \cdots & 0 \\ A_1^2 & A_2^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ A_1^N & A_2^N & \cdots & A_T^T \end{bmatrix}, \tag{2.11}$$

$$\tag{2.12}$$

$$A_\tau^k = A_{k-1} A_{k-2} \cdots A_\tau, \tag{2.13}$$

where $A_1^1 = I$.

2.4 Control Policies

A control policy is a mapping that takes in information such as the state and outputs an input which is fed into the system. It is what modifies the behavior of a system to, for example, minimize fuel usage and avoid obstacles. We presume that a control policy function is a mapping from current state to current input. There are two types of control policy functions we consider: deterministic and stochastic.

Definition 2.9 (Deterministic Control Policy)**.** *A deterministic policy function is a mapping from the set of states to a set of actions $\pi : \mathcal{X} \to \mathcal{U}$, i.e. a function $u_k = \pi(x_k)$.*

Definition 2.10 (Stochastic Control Policy)**.** *A stochastic policy function is a measurable mapping $\pi : \mathcal{X} \times \mathcal{V} \to \mathcal{U}$, i.e. a function*

$$\mathbf{u}_k = \pi(x_k, \mathbf{v}(x_k)) = \pi(x_k, \mathbf{v}), \tag{2.14}$$

where $\mathbf{v}$ is a disturbance variable.

The random disturbance variable, $\mathbf{v}$, takes similar form to $\mathbf{w}$. It is a simpler measurable mapping than $\mathbf{w}$, in that $\mathbf{v} : \Omega \times \mathcal{X} \to \mathcal{V}$, i.e. it is only conditioned on the current state. The probability that $\mathbf{v}$ takes on a value in a set $V \in \mathcal{M}(\mathcal{V})$, given current state is,

$$\mathbb{P}_{\mathbf{v}}(V|x_k) = \mathbb{P}\left(\{\omega \in \Omega | \mathbf{v}(\omega|x_k) \in V\} | x_k\right). \tag{2.15}$$

As shorthand, we treat $\mathbf{v}(x_k) = \mathbf{v}(\omega|x_k)$ as the same function. In addition, the measurable space $(\mathcal{V}, \mathcal{M}(\mathcal{V}))$ we also call a disturbance space. Similar to the stochastic system function in Definition 2.5, we now define the probability that the action, $\mathbf{a}_t$ takes on an action in $U \in \mathcal{M}(\mathcal{U})$ given current state, x_k, and disturbance, $\mathbf{v}$.

Definition 2.11 (Control Transition Kernel)**.** *The action transition kernel is a conditional probability function, $\mathbb{P}_\pi : \mathcal{M}(\mathcal{U}) \times \mathcal{X} \to [0, 1]$. Specifically, we represent it by the disturbance $\mathbf{v}$,*

$$\mathbb{P}_\pi(U|x_k) = \mathbb{P}_{\mathbf{v}}(\{\mathbf{v_k}(x_k) \in V | \pi(x_k, \mathbf{v}(x_k)) \in U\}|x_k), \; U \in \mathcal{M}(\mathcal{U}), \; V \in \mathcal{M}(\mathcal{V}). \tag{2.16}$$

Note that control transition kernel is typically called a stochastic policy as shorthand but that terminology does not clearly define what it represents. There are three additional properties of policies. First, a policy function and an action transition kernel, need not follow the standard information pattern of depending on the current state. For example, it can be depend on belief of the state, i.e. the system is partially observable [32, Ch.9] which appears in Chapter 8 where we validate a pixels to control neural network in Section 8.6.2. Second, similar to stochastic systems, our policy here is presumed Markov,

$$\mathbb{P}_\pi(X|x_k) = \mathbb{P}_\pi(X|x_k, u_k, x_{k-1}, u_{k-1}, \cdots, x_1, u_1, x_0, u_0). \tag{2.17}$$

Third, the policy function we presume is implicitly stationary. Non-stationarity is dependent on the definition of the policy function and the set of actions where the set of action could be time or state dependent, e.g. $\mathcal{U}_k(x_k)$. Both non-Markov and non-stationary polices have careful treatment in the stochastic optimal control literature [32]. We can determine these properties of a policy function or action transition kernel by observing a sequence of actions over a time horizon.

Definition 2.12 (Policy)**.** *A policy is a set of actions inputs by time, $k \in \mathbb{N}$, over a horizon, $N \in \mathbb{N}$, i.e. $\Pi = \{u_0, u_1, \ldots, u_k, \ldots, u_{N-1}\}$ where actions u_k come from a policy function π, given current state, x_k.*

A control policy also allows us to define a closed-loop stochastic system function and a closed loop state transition kernel, from which we can define a closed-loop stochastic system trajectory.

Definition 2.13 (Closed Loop System Function)**.** *The next state, given current state and a stochastic policy function is,*

$$\mathbf{x}_{k+1} = f(x_k, \pi(x_k, \mathbf{v}(x_k)), \mathbf{w}(x_k, \pi(x_k, \mathbf{v}(x_k)))) = f(x_k, \pi(x_k, \mathbf{v}), \mathbf{w})). \tag{2.18}$$

Definition 2.14 (Closed Loop State Transition Kernel)**.** *The closed loop state transition* *kernel is a conditional probability function*

$$\mathbb{P}_{\mathbf{x}_{k+1},\pi}(X|x_k) = \mathbb{E}[\mathbb{P}_{\mathbf{x}_{k+1}}(X|x_k,\mathbf{u}_k)|x_k] = \int_{\mathcal{U}} \mathbb{P}_{\mathbf{x}_{k+1}}(X|x_k,u_k)\mathrm{d}\mathbb{P}_\pi(u_k|x_k). \tag{2.19}$$

Definition 2.15 (Closed Loop System Trajectory)**.** *A closed loop system trajectory is a set of tuples* T *as in Definition 2.8 with actions from a policy as in Definition 2.12.*

2.4.1 Open-Loop Control for Linear Systems

Chapters 3 and 5 utilize an open-loop control policy. An open-loop controller is a deterministic controller does not observe the state and instead produces a deterministic set of value through the time horizon, i.e. $\pi = \{u_0, u_1, \dots, u_k, \dots, u_{N-2}, u_{N-1}\}$ where $u_k \in \mathcal{U}$. This bears resemblance to the deterministic controller in Definition 2.9 but without the state as an input argument.

2.4.2 Affine-Feedback Control for Linear Systems

Chapter 4 utilizes an affine, linear feedback controller of the following form,

Definition 2.16 (Feedback law)**.** *The controller in* (2.10) *has an affine state feedback structure, given by*

$$\mathbf{u}_k = \sum_{i=0}^{k} L_{k,i}\mathbf{x}_i + g_k. \tag{2.20}$$

Concatenating these vectors yields $\mathbf{u} = L\mathbf{x} + g$*, where* $L \in \mathbb{R}^{Nm\times(N+1)n}$ *is a lower block triangular matrix and* $g = [g_0^\intercal, \dots, g_{N-1}^\intercal]^\intercal \in \mathbb{R}^{Nm}$*.*

Note that this feedback law uses the full state *history* to determine the control input at every time step k, as opposed to just using the current state $\mathbf{x}_k$.

Proposition 2.1 (Affine disturbance feedback [33])**.** *The feedback law in (2.20) results* *in the state and input sequences*

$$\mathbf{x} = (I - \mathcal{B}L)^{-1}(\mathcal{A}\mathbf{x}_0 + \mathcal{D}\mathbf{w} + \mathcal{B}g), \tag{2.21a}$$

$$\mathbf{u} = L(I - \mathcal{B}L)^{-1}(\mathcal{A}\mathbf{x}_0 + \mathcal{D}\mathbf{w} + \mathcal{B}g) + g. \tag{2.21b}$$

Proof. Plugging (2.20) into (2.10) yields (2.21a). Similarly, plugging (2.21a) into (2.20) yields (2.21b). □

Corollary 2.1. *Given the affine disturbance feedback terms,*

$$K = L(I - \mathcal{B}L)^{-1}, \tag{2.22a}$$

$$v = L(I - \mathcal{B}L)^{-1}\mathcal{B}g + g, \tag{2.22b}$$

then the state and input sequences in (2.21) can be equivalently written as

$$\mathbf{X} = (I + \mathcal{B}K)(\mathcal{A}\mathbf{x}_0 + \mathcal{D}\mathbf{W}) + \mathcal{B}v, \tag{2.23a}$$

$$\mathbf{U} = K(\mathcal{A}\mathbf{x}_0 + \mathcal{D}\mathbf{W}) + v, \tag{2.23b}$$

Proof. We substitute (2.22a) and (2.22b) into (2.21b), and substitute $\mathbf{u}$ into (2.10), to obtain

$$\mathbf{X} = \mathcal{A}\mathbf{x}_0 + \mathcal{B}\left(L(I - \mathcal{B}L)^{-1}(\mathcal{A}\mathbf{x}_0 + \mathcal{D}\mathbf{W} + \mathcal{B}g) + g\right) + \mathcal{D}\mathbf{W}, \tag{2.24a}$$

$$\mathbf{U} = (I + \mathcal{B}L(I - \mathcal{B}L)^{-1})(\mathcal{A}\mathbf{x}_0 + \mathcal{D}\mathbf{W}) + L(I - \mathcal{B}L)^{-1}\mathcal{B}g + g. \tag{2.24b}$$

Simplifying (2.24a) and (2.24b) yields the desired result (2.23a) and (2.23b). □

An affine, linear controller is a stochastic controller that observes the state and provides a random input, $\mathbf{u}_k$. This bears resemblance to the stochastic policy in Definition 2.10, where the stochastic input, $\mathbf{u}_k$, at the current timestep arises not from an arbitrary process $\mathbf{v}$ but through the state $\mathbf{x}_k$ at the current timestep.

2.5 Optimization

Numerical optimization is the tool by which we obtain control polices to minimize objectives such as fuel usage or having a dynamical system track a desired trajectory while satisfying constraints. A common objective we minimize, which appears throughout this dissertation, is the expectation of quadratic functions over state and input,

$$J(\mathbf{U}) = \mathbb{E}\left[(\mathbf{X} - X_d)^\intercal \mathcal{Q}(\mathbf{X} - X_d) + \mathbf{U}^\intercal \mathcal{R}\mathbf{U}\right], \tag{2.25}$$

with $\mathcal{Q} = \mathrm{diag}\,(Q_0, \dots, Q_{N-1})$, $\mathcal{R} = \mathrm{diag}\,(R_0, \dots, R_{N-1})$, and $X_d = [x_{d,0} \; \cdots \; x_{d,N}]^\intercal \in \mathbb{R}^{(N+1)n}$. Since $Q_k \succeq 0$ and $R_k \succ 0$, $\forall k \in \mathbb{N}_{[0,N-1]}$, it follows that $\mathcal{Q} \succeq 0$ and $\mathcal{R} \succ 0$. Common constraints include probabilistic constraints are imposed on the state and input, namely,

$$\mathbb{P}_{\mathbf{X}}\left(\left\{X \in \mathbb{R}^{nN} : \bigcap_{k=1}^{N} E_k X \in \mathcal{X}_k\right\}\right) \geq 1 - \Delta_X, \tag{2.26a}$$

$$\mathbb{P}_{\mathbf{U}}\left(\left\{U \in \mathbb{R}^{m(N-1)} : \bigcap_{k=0}^{N-1} F_k U \in \mathcal{U}_k\right\}\right) \geq 1 - \Delta_U, \tag{2.26b}$$

where,

$$\mathcal{X}_k = \cap_{j=1}^{N_X}\{x \in \mathbb{R}^n : \alpha_{j,k}^\intercal x \leq \beta_{j,k}\} \tag{2.27a}$$

$$\mathcal{U}_k = \cap_{j=1}^{N_U}\{u : a_{j,k}^\intercal u \leq b_{j,k}\} \tag{2.27b}$$

are polytopic sets defined as intersecting hyperplanes, and where $E_k = [0_{n\times nk}, I_n, 0_{n(N-k)\times n}]$, and $F_k = [0_{m\times mk}, I_m, 0_{m(N-k-1)\times m}]$ isolate the k^{th} element of the state and input, respectively, and $\Delta_X, \Delta_U \in (0, 1)$ are constraint violation thresholds.

2.5.1 Convex Optimization

Convex optimization problems are problems of the following form,

$$\begin{aligned} \underset{x\in\mathbb{R}^n}{\text{minimize}} \quad & f(x), && (2.28a)\\ \text{subject to} \quad & g(x) \leq 0, && (2.28b)\\ & Sx = s, && (2.28c) \end{aligned}$$

where $f,\ g$ are convex and $S \in \mathbb{R}^{l\times n},\ s \in \mathbb{R}^l$ denote linear equality constraints. Convex optimization arises in stochastic optimal control in various contexts and is central to linear quadratic Gaussian stochastic optimal control problems with and without constraints [35].

2.5.2 Non-Convex and Difference of Convex Optimization

Non-convex optimization problems are problems of the following form [35],

$$\begin{aligned} \underset{x\in\mathbb{R}^n}{\text{minimize}} \quad & h(x), && (2.29a)\\ \text{subject to} \quad & p(x) \leq 0, && (2.29b)\\ & && (2.29c) \end{aligned}$$

where $h,\ p$ are non-convex and $S \in \mathbb{R}^{l\times n},\ s \in \mathbb{R}^l$ denote linear equality constraints. Non-convex optimization arises in equally in stride to the convex formulation of stochastic optimal control problems in areas such as non-linear model predictive control.

Difference of Convex Optimization

Difference of convex programs are a subset of non-convex optimization problems of the form,

$$\begin{aligned} \underset{x\in\mathbb{R}^n}{\text{minimize}} \quad & f(x) - g(x) \\ \text{subject to} \quad & f_i(x) - g_i(x) \leq 0, \quad \forall i \in \mathbb{N}_{[1,M]} \end{aligned} \tag{2.30}$$

where f, g, f_i, and g_i are convex for $i \in \mathbb{N}_{[1,l]}$, $l \in \mathbb{N}$. However, it differs from standard non-convex optimization problem as it can immediately utilize convex optimization tools and solver off-the-shelf. Difference of convex programs can directly optimize non-convex functions h, p as long as they are twice differentiable. The penalty based convex-concave procedure [36] solves (2.30) via sequential convex optimization, and is agnostic to the feasibility of the initial condition [36, 37]. The solution is also guaranteed to converge to a fixed point [38].

Part I

Model-Based, Stochastic Optimal Control

Chapter 3

Open-Loop Control of Linear Systems With Log-Concave Uncertainty

3.1 Introduction

Stochastic optimal control requires enforcement of chance constraints, which permit violation of the state constraints with a probability below a specified threshold [39, 40]. However, in the presence of non-Gaussian disturbances, such constraints are hard to implement in a computationally tractable manner analytically. Existing approaches to accommodate non-Gaussian disturbances involve sampling or moment-based methods [39]. Sampling approaches often result in trade-offs between accuracy, feasibility, and computational complexity [1, 3], even with sample reduction techniques [2, 41]. Moment-based approaches [4, 5, 42] can induce conservatism that significantly reduces the solution space, as well as non-convexities associated with simultaneous risk allocation and controller synthesis [42–44].

This chapter proposes a method for stochastic optimal control of linear systems with log-concave disturbances, that results in a scalable solution, avoiding both moment bounds and sampling. Our approach uses risk allocation, which is employed in moment based methods. Risk allocation uses Boole's inequality to decompose joint chance constraints into simpler, individual chance constraints [42–44]. The creation of new decision variables needed to allocate risk across individual chance constraints yields a non-convex problem. Such problems are typically solved via iterative, coordinate descent methods [42, 44], yielding suboptimal solutions with additional conservatism.

Here, this chapter proposes risk allocation that results in a convex formulation, and enables simultaneous (as opposed to iterative) risk allocation and controller synthesis. The key to this is 1) the use of characteristic functions (the Fourier transform of the probability density function) to enforce chance constraints, and 2) a reformulation of risk allocation as a difference-of-convex program. The former enables straightforward calculation of chance constraints, via simple one-dimensional integrals. The latter enables local solutions via convex optimization with clear convergence guarantees [36], when the disturbance is log-concave. Lastly, since convex constraints that arise may be non-conic, we employ piecewise affine approximations, so that standard conic solvers may be used to solve the stochastic optimal control problem.

3.2 Related Work

The primary limitations of our approach are that a) the difference-of-convex reformulation is tractable only for controllers that are limited to open-loop policies, which are more conservative than affine feedback policies, particularly over long time horizons [45, Ch. 2, Sec. 4], and b) open-loop controllers are notoriously resistant to stability guarantees [45, 46]. Although, reference tracking can prove difficult with open-loop control in the presence of uncertainty, it can be facilitated by adding an extra term in the cost function [47, 48] or employing a reference governor [49]. We addess this limitation by presuming systems that are Schur stable, by using an linear quadratic regulator (LQR) based pre-stabilizing controller that can satisfy input constraints, similarly to [50]. Other approaches to ensure stochastic stability involve augmenting the state vector [51], although extension to non-Gaussian disturbances is unclear [46, 52].

Further, open-loop control has advantages over feedback-based approaches: 1) it admits enforcement of hard input constraints without approximations or additional conservatism [50, 53, 54], 2) it can be used in problems where sensory feedback may be unavailable, such as hypersonic vehicles [8, 55], and 3) it is computationally less expensive than constrained, feedback-based control. Thus, open-loop control synthesis is commonplace in stochastic model predictive control [39] for many of these reasons. In addition, while alternative approaches such as robust control or saturated affine disturbance feedback may yield tractable, convex methodologies for affine controller synthesis, they often require artificially bounding disturbances, or ignoring available information about the stochasticity [42, 50, 53, 56–58]. These approaches are particularly ill-suited to systems with long-tailed or heavy distributions which would truncate or saturate the true nature of the uncertainty.

3.3 Main Contribution and Organization

The main contribution of this chapter is a convex solution for stochastic, open-loop optimal control of linear dynamical systems with log-concave disturbances, that can be solved via conic solvers. The approach is based on preliminary work I contributed to [59], in which we propose a mixed-integer program to solve constrained, stochastic, open-loop optimal control of linear Gaussian systems. This chapter extends the approach in [59] to disturbances with log-concave probability distribution functions, by employing characteristic functions to evaluate chance constraints. The log-concavity property is critical for efficient computation, because it assures convexity of the chance constraints. Further, we show that we can construct a difference-of-convex reformulation of the risk allocation constraint, which, in combination with piecewise approximation, results in a conic program. This approach is superior to [6, 59], as it does not incur additional conservatism in the risk allocation caused by convex restriction.

The organization of the paper is as follows: We present the problem formulation in Section 3.4. Section 3.5 describes the reformulation of the stochastic optimal control problem using risk allocation, piecewise affine approximation, and difference-of-convex programming. We demonstrate our approach on two motion planning examples, and compare performance to state-of-the-art moment based and sampling approaches in Section 3.6, and summarize our contribution in Section 3.7.

3.4 Problem statement

A function $f : \mathbb{R}_{\geq 0} \to \mathbb{R}$ is log-concave, if $\log(f)$ is concave [60, Sec. 3.5.1.]. This chapter follows the convention that $\log(0) \triangleq -\infty$.

Assumption 3.1. *$\psi_{\mathbf{x}_0}$ and $\psi_{\mathbf{W}}$ are log-concave [61, Sec. 2.3].*

Log-concave probability densities include Gaussian and exponential disturbances as well as uniform disturbances over convex sets [62]. Since log-concavity is preserved under products, log-concave ψ_{w_k} yields log-concave ψ_W. The affine transformation of random vectors from log-concave distributions is log-concave [61, Lemma 2.1]. Due to the linearity of (2.10), the mean and the covariance vector of $\mathbf{X}$ admit closed-form expressions,

$$\mu_{\mathbf{X},U} = \mathcal{A}\mu_{\mathbf{x}_0} + \mathcal{B}U + \mathcal{D}\mu_{\mathbf{W}} \tag{3.1a}$$

$$\Sigma_{\mathbf{X},U} = \mathcal{A}\Sigma_{\mathbf{x}_0}\mathcal{A}^\intercal + \mathcal{D}\Sigma_{\mathbf{W}}\mathcal{D}^\intercal. \tag{3.1b}$$

We are interested in solving the following stochastic optimal control problem,

$$\begin{aligned} \underset{U}{\text{minimize}} \quad & \mathbb{E}\left[(\mathbf{X} - X_d)^\intercal \mathcal{Q}(\mathbf{X} - X_d)\right] + U^\intercal \mathcal{R} U && \text{(3.2a)} \\ \text{subject to} \quad & (3.1),\ (2.26a) \\ & \bigcap_{k=0}^{N-1} F_k U \in \mathcal{U}_k, && \text{(3.2b)} \end{aligned}$$

with positive semi-definite matrices as in (2.25), and polytopic state and input constraints as in (2.27). The key difference between this problem and those in [43, 44, 59, 63] is that we consider non-Gaussian, log-concave disturbances $\psi_{\mathbf{W}}$.

For a Gaussian disturbance, risk allocation is an established approach to assure (5c). However, under Assumption 3.1, evaluation of the resulting chance constraints is not straightforward. We propose an approach based in characteristic functions, that is sample and moment-bound free, to solve (3.2). In contrast to moment based approaches, which employ lower order moments, our approach uses *all* moments of the distribution, and does not require sampling. We propose to solve two problems:

Problem 1. *Extend risk allocation to log-concave disturbances without moment-based bounds or sampling.*

Problem 2. *Solve* (3.2) *under Assumption 1 using a convex reformulation that employs the risk allocation technique from Problem 1, in a manner amenable to conic solvers.*

We address problem 1 by using characteristic functions to enforce chance constraints. We address problem 2 through piecewise affine approximations of a reverse convex constraint.

3.5 Convexification of non-Gaussian joint chance constraints

3.5.1 Risk-allocation for log-concave disturbances

The standard risk-allocation approach [43, 44, 59, 63], transforms the joint chance constraints (2.26a) into a set of individual chance constraints via Boole's inequality. That is, given $\mathbf{Z} = \bar{A}\mathbf{x}(0) + \bar{D}\mathbf{W}$,

$$\begin{aligned}
&\mathbb{P}_{\mathbf{X}}\left(\left\{X \in \mathbb{R}^{nN} : \bigcap_{k=1}^{N} E_k X \in \mathcal{X}_k\right\}\right) \geq 1 - \Delta_X && (3.3)\\
\Leftrightarrow &\mathbb{P}_{\mathbf{Z}}\left(\left\{Z \in \mathbb{R}^{nN} : \bigcap_{k=1}^{N}\bigcap_{j=1}^{N_X} \alpha_{j,k}^{\intercal} E_k Z \leq \beta_{j,k} - \alpha_{j,k}^{\intercal} E_k \mathcal{B} U\right\}\right) \geq 1 - \Delta_X\\
\Leftrightarrow &\mathbb{P}_{\mathbf{Z}}\left(\left\{Z \in \mathbb{R}^{nN} : \bigcap_{i=1}^{L} p_i^{\intercal} Z \leq q_i - p_i^{\intercal} \mathcal{B} U\right\}\right) \geq 1 - \Delta_X\\
\Leftrightarrow &\mathbb{P}_{\mathbf{Z}}\left(\left\{Z \in \mathbb{R}^{nN} : \bigcup_{i=1}^{L} p_i^{\intercal} Z > q_i - p_i^{\intercal} \bar{B} U\right\}\right) \leq \Delta_X\\
\Leftarrow &\sum_{i=1}^{L} \mathbb{P}_{\mathbf{Z}}\left(\left\{Z \in \mathbb{R}^{nN} : p_i^{\intercal} Z > q_i - p_i^{\intercal} \mathcal{B} U\right\}\right) \leq \Delta_X\\
\Leftrightarrow &\left\{\begin{array}{l} \mathbb{P}_{\mathbf{Z}}\left(\left\{Z \in \mathbb{R}^{nN} : p_i^{\intercal} Z \leq q_i - p_i^{\intercal} \mathcal{B} U\right\}\right) \geq 1 - \delta_{X,i},\ \forall i \in \mathbb{N}_{[1,L]},\\ \sum_{i=1}^{L}, \delta_{X,i} \leq \Delta,\ \delta_{X,i} \in [0,\Delta],\ \forall i \in \mathbb{N}_{[1,L]}, \end{array}\right. && (3.4)
\end{aligned}$$

where $\alpha_{j,k} = p_i$ for $j \in \mathbb{N}_{[1,N_X]},\ k \in \mathbb{N}_{[1,N]},\ i \in \mathbb{N}_{[1,N_L]}$ and $L = N_X(N-1)$. The risk of violating the constraint $p_i^{\intercal} X \leq q_i,\ i \in \mathbb{N}_{[1,L]}$ is represented by the decision variable $\delta_{X,i} \in [0, 1)$. We have $\delta_{X,i} \leq \Delta$ since $\sum_{i=1}^{L} \delta_{X,i} \leq \Delta$ and $\delta_{X,i}$ are non-negative.

Let $\Phi_{p_i^{\intercal}\mathbf{Z}} : \mathbb{R} \rightarrow [0, 1]$ denote the cumulative distribution function of the random variable $p_i^{\intercal}\mathbf{Z}$,

$$\Phi_{p_i^{\intercal}\mathbf{Z}}\left(q'\right) = \mathbb{P}_{\mathbf{Z}}\left(\left\{Z \in \mathbb{R}^{nN} : p_i^{\intercal} Z \leq q'\right\}\right), \tag{3.5}$$

for any scalar $q' \in \mathbb{R}$. We use $\Phi_{p_i^\intercal \mathbf{Z}}$ to rewrite the constraints (3.4) as

$$\Phi_{p_i^\intercal \mathbf{Z}}\left(q_i - p_i^\intercal \bar{B} U\right) \geq 1 - \delta_{X,i} \qquad \forall i \in \mathbb{N}_{[1,L]}, \tag{3.6a}$$

$$\sum\nolimits_{i=1}^{L} \delta_{X,i} \leq \Delta,\ \ \delta_{X,i} \in [0, \Delta], \qquad \forall i \in \mathbb{N}_{[1,L]}, \tag{3.6b}$$

Any feasible controller (3.2b) with a feasible risk allocation $\delta \triangleq [\delta_1\ \cdots\ \delta_L] \in [0,1]^L$ that satisfies (3.6) also satisfies (2.26a).

3.5.2 Enforcing chance constraints using characteristic functions

The main insight we use in this chapter is that the evaluation of the cumulative distribution function in (3.6a) can be evaluated by the Gil-Pilaez inversion noted in (2.3) and characteristic functions in Section 2.2.1. In doing so, we can enforce the chance constraint in (3.6a) using only characteristic functions of $\mathbf{Z}$.

Lemma 3.1 ([64, Thm. 4.2.1]). *If $\psi_{p_i^\intercal \mathbf{Z}}$ is log-concave, then $\Phi_{p_i^\intercal \mathbf{Z}}$ is log-concave over $\mathbb{R}$.*

Using (3.6) and Lemma 3.1, we approximate (3.2) as follows,

$$\underset{U,t}{\text{minimize}}\ \ (\mu_{\mathbf{X},U} - X_d)^\intercal Q (\mu_{\mathbf{X},U} - X_d) + U^\intercal R U + \operatorname{tr}(Q \Sigma_{\mathbf{X},U}) \tag{3.7a}$$

subject to (3.2b),

$$\forall i \in \mathbb{N}_{[1,L]},\ \ p_i^\intercal \bar{B} U + \Phi^{-1}_{p_i^\intercal \mathbf{Z}}(\epsilon) \leq q_i \tag{3.7b}$$

$$\forall i \in \mathbb{N}_{[1,L]},\ \ \log\left(\Phi_{p_i^\intercal \mathbf{Z}}(q_i - p_i^\intercal \bar{B} U)\right) \geq t_i \tag{3.7c}$$

$$\forall i \in \mathbb{N}_{[1,L]},\ \ t_i \in [\log(1-\Delta), 0] \tag{3.7d}$$

$$\forall i \in \mathbb{N}_{[1,L]},\ \ \log\left(\sum_{i=1}^{L} \exp(t_i)\right) \geq \log(L - \Delta). \tag{3.7e}$$

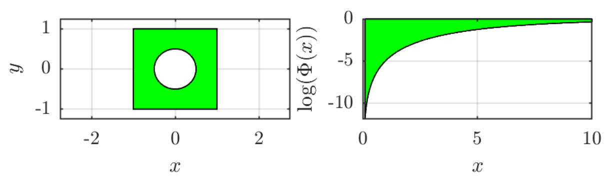


Figure 3.1: Left: $f(x, y) = x^2 + y^2 \geq r^2$ within a unit box. Right: The epigraph of $f(x) = \log(\Phi(x))$ of a log-concave cumulative distribution function. Both functions are reverse convex, meaning that the complements of the inequalities, i.e. $x^2 + y^2 \leq r^2$ and $\log(\Phi(x)) \geq t$, respectively, are convex.

for a small scalar $\epsilon > 0$ and a change of variables

$$t_i \triangleq \log(1 - \delta_{X,i}), \;\; \forall i \in \mathbb{N}_{[1,L]} \tag{3.8}$$

with $t = [t_1 \cdots t_L]^\intercal \in \mathbb{R}^L$. We now establish the relationship between (3.2) and (3.7), and show that (3.7) is a non-convex program with a reverse convex constraint. Recall that reverse-convex constraints are optimization constraints of the form $f(\cdot) \geq 0$, where $f(\cdot)$ is a convex function, as shown in Figure 3.1.

Theorem 3.1. *Under Assumption 3.1, the following statements hold for any* $\Delta \in [0, 1)$ *and any* $\epsilon > 0$*:*

1. *Every feasible solution of* (3.7) *is feasible for* (3.2)*, and*
2. *The cost and the constraints* (3.7b)–(3.7c) *are convex. However,* (3.7e) *is a reverse convex constraint.*

Proof. 1) We need to show that satisfaction of (3.7b)–(3.7e) satisfies (2.26a). Recall that the collection of constraints (3.6) tighten (2.26a). Therefore, it is sufficient to show that the satisfaction of constraints (3.7b)–(3.7e) guarantee satisfaction of (3.6).

The constraint (3.7b) ensures that the constraint (3.7c) is well-defined, since the satisfaction of (3.7b) ensures that $\Phi_{p_i^\intercal \mathbf{Z}}(q_i - p_i^\intercal \bar{B}U)$ is positive. Under (3.8), satisfaction of (3.7c) and (3.7d) implies satisfaction of (3.6a) and $\delta_{X,i} \in [0, \Delta]$, respectively. Finally,

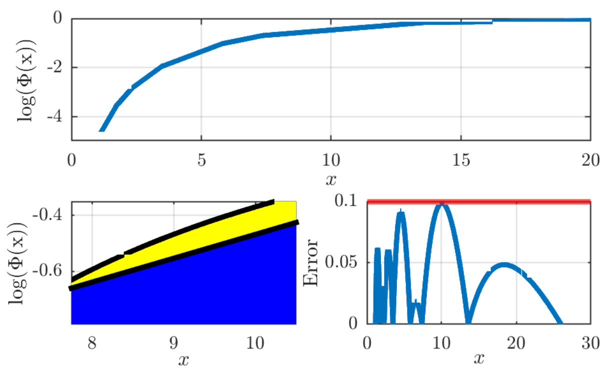


Figure 3.2: Top: Log of the cumulative distribution function of an affine transformation of a random vector $a^\intercal \mathbf{w}_t$, with $\mathbf{w}_t = [\mathbf{w}_1\ \mathbf{w}_2\ \mathbf{w}_3]^\intercal \in \mathbb{R}^3$ and scale parameters $\overline{\lambda}_{\mathbf{w}} = [0.5\ 0.25\ 0.1667]^\intercal$. Bottom Left: A piecewise affine underapproximation (blue) of the log of the cumulative distribution function (yellow). Bottom Right: The difference $f(x) - \ell_f(x)$ as in (17), with $\eta = 0.1$.

we show that (3.7e) and (3.6b) are equivalent via simple algebraic manipulations,

$$\sum_{i=1}^{L} \delta_{X,i} \leq \Delta \Leftrightarrow L - \sum_{i=1}^{L} (1 - \delta_{X,i}) \leq \Delta \tag{3.9a}$$

$$\Leftrightarrow \log\left(\sum_{i=1}^{L} \exp(t_i)\right) \geq \log\left(L - \Delta\right) \tag{3.9b}$$

In other words, every feasible solution (U, t) of (3.7) maps to a feasible solution to (3.6) with $U \in \mathcal{U}^N$, and thereby is feasible for (3.2).

2) The cost (3.7a) is a convex quadratic function of U. By construction, the constraints (3.7b) and (3.7d) are linear constraints in U and t. The convexity of (3.7c) follows from Lemma 3.1 and the definition of log-concavity. Recall that $\log\left(\sum_{i=1}^{L} \exp(t_i)\right)$ is a convex function in t [60, Sec. 3.1.5], hence (3.7e) is a reverse-convex constraint. $\square$

3.5.3 Conic reformulation of (3.7c) via piecewise affine approximation

We now focus on enforcing the convex constraint (3.7c). Although convex, the constraint (3.7c) is not conic, which prevents the use of standard conic solvers. We present tight conic reformulation of (3.7c) via piecewise affine approximations.

Given a concave function $f : \mathcal{D} \to \mathcal{R}$ for bounded intervals $\mathcal{D}, \mathcal{R} \subset \mathbb{R}$, we define its piecewise affine underapproximation as $\ell_f : \mathbb{R} \to \mathbb{R}$ for some $m_j, c_j \in \mathbb{R}$ for $j \in \mathbb{N}_{[1,N_f]}$ and $N_f \in \mathbb{N}$ distinct affine elements,

$$\ell_f(x) \triangleq \min_{j \in \mathbb{N}_{[1,N_f]}} (m_j x + c_j). \tag{3.10}$$

For a user specified approximation error $\eta > 0$, we can find a ℓ_f for a concave f such that

$$\ell_f(x) \leq f(x) \leq \ell_f(x) + \eta, \tag{3.11}$$

with the sandwich algorithm [65]. The sandwich algorithm has convergence guarantees that can be balanced between the user-defined error and the number of affine pieces [6, 59, 65].

In (3.7), we use the piecewise affine underapproximation of the concave functions $f_i = \log\left(\Phi_{p_i^\intercal \mathbf{Z}}\right)$ with $N_i \in \mathbb{N}$ distinct pieces for every $i \in \mathbb{N}_{[1,L]}$ (as shown in Figure 3.2) to enforce (3.7c). The functions f_i have bounded domain and range in $\mathbb{R}$ due to (3.7b). We evaluate f_i using the one-dimensional numerical integration of characteristic functions, as in (2.3). We obtain the following optimization problem,

$$\begin{aligned} \underset{U,t}{\text{minimize}} \quad & (3.7a) \\ \text{subject to} \quad & (3.2b),\ (3.7b),\ (3.7d),\ (3.7e) \\ \substack{\forall i \in \mathbb{N}_{[1,L]} \\ \forall j \in \mathbb{N}_{[1,N_i]}}, \quad & m_{i,j}\left(q_i - p_i^\intercal \bar{B} U\right) + c_{i,j} \geq t_i \end{aligned} \tag{3.12a}$$

By Theorem 3.1 and the use of piecewise affine underapproximations of $\log(\Phi_{p_i^\intercal \mathbf{Z}})$ in (3.7c), every feasible solution of (3.12) is feasible for (3.7), and thereby (3.2).

3.5.4 Solving (3.12) via difference of convex programming

The optimization problem (3.12) has a quadratic cost (3.7a), and linear constraints (3.2b), (3.7b), and (3.12a) in the decision variables U and t, and a reverse-convex constraint (3.7e). We now discuss a tractable solution to (3.12) using difference of convex programming [36].

Given the current estimate for the risk allocation $r = [r_1 \ \cdots \ r_L]^\intercal \in [0,1]^L$ (i.e., the initialization for t), the penalty based convex-concave procedure solves the following convex approximation of (3.12) at every iteration,

$$\begin{aligned}
\underset{U,t,s}{\text{minimize}} \quad & (3.7\text{a}) + \tau_k s && (3.13\text{a})\\
\text{subject to} \quad & (3.2\text{b}),\ (3.7\text{b}),\ (3.7\text{d}),\ (3.12\text{a}) \\
& s \geq 0 && (3.13\text{b})\\
& \log\left(\textstyle\sum_{i=1}^L \exp(r_i)\right) + \tfrac{1}{\sum_{i=1}^L \exp(r_i)} \textstyle\sum_{i=1}^L \exp(r_i)(t_i - r_i) + s \geq \log(L - \Delta) && (3.13\text{c})
\end{aligned}$$

where $\tau_k \geq 0$ for $k \in \mathbb{N}$ are optimization hyperparameters. The constraint (3.13c) corresponds to the first-order approximation of the reverse-convex constraint (3.7e), which is relaxed by a scalar slack variable s. We penalize the slack variable s in the objective (3.13a). The problem (3.13) is convex, since (3.13c) is a linear constraint in t and s, and all other constraints and the objective are convex (Theorem 3.1.b).

Starting with an arbitrary risk allocation $\delta_0 \in [0,1]^L$, we iteratively solve (3.13) with monotonically increasing values of τ_k to promote feasibility. In the numerical experiments, we chose a uniform risk allocation $\delta_0 = \frac{\Delta}{L} I_L$, where I_L is a L-dimensional vector of ones. The corresponding initialization of $\overline{r}$ is therefore $\overline{r}_0 = \log(1 - \delta_0)$. The convex-concave procedure converges to a local fixed-point when a pre-specified violation tolerance η_{viol} is met and the difference in cost between iterations k is less than a pre-specified tolerance η_{dc} [36]. However, the convergence to a local minima remains an open problem [38].

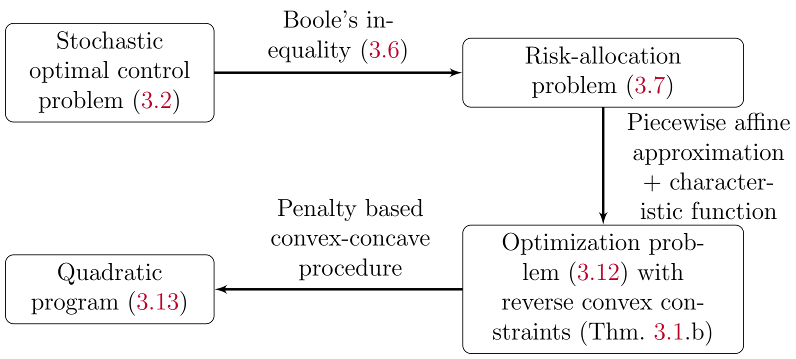

Stochastic optimal control problem (3.2)
Boole's inequality (3.6)
Risk-allocation problem (3.7)
Piecewise affine approximation + characteristic function
Optimization problem (3.12) with reverse convex constraints (Thm. 3.1.b)
Penalty based convex-concave procedure
Quadratic program (3.13)

Figure 3.3: Manipulations that result in a convexified reformulation of (5) that is amenable to conic solvers.

In addition, the procedure may terminate prematurely if τ_k reaches a user specified maximum number of iterations, τ_{max}.

In summary, we have transformed the original stochastic optimal control problem presented in (3.2) into a convex quadratic problem, via the steps shown in Figure 3.3. We first employed risk allocation (3.7), then converted the non-conic convex constraints in (3.7) into conic convex constraints using piecewise affine approximations, as well as the characteristic function. Finally, we utilize difference-of-convex programming to address the remaining reverse convex constraint (3.7e). Thus, our approach solves a convex (quadratic) program (3.13) iteratively to compute a local optimum of (3.2).

3.6 Examples

We apply the proposed approach to two examples: 1) a double integrator, and 2) a quadrotor flying in an environment with a crosswind. We compare the performance of the controller produced by our approach to other open-loop methods: 1) a scenario approach [2], 2) a particle based approach [3], and 3) a moment based approach [4, 5]. We presume that the system is pre-stabilized via LQR using MATLAB's `dlqr`$(\cdot)$ function [66, 67]. When feasible, we also compare performance with the empirical characteristic function (ECF) approach in [15]. The number of samples for the scenario approach is determined by first specifying Δ, as well as a confidence bound $\xi = 1 \times 10^{-16}$ of not achieving Δ [2]. To ensure a fair comparison, we use the same number of samples for the particle and ECF approach. We measure the performance of the controllers based on the computed cost, the probability of constraint satisfaction, and the computation time. For methods which explicitly use samples in the constraints, performance is determined from the average of three runs. For the double integrator, $N_s = 91$ samples were used, and for the quadrotor, $N_s = 143$ samples were used. We used Monte-Carlo simulation with 10^5 samples for validation.

All computations are done with MATLAB on an Intel Core i9-10900K CPU with 3.70GHz and 128GB RAM. We implemented our algorithm, the scenario approach, the particle approach, and the empirical characteristic function approach in YALMIP [68] with MOSEK [69]. We used `fmincon` for the moment approach. For implementation of the proposed approach, we used $\tau_{max} = 10000$, $\tau_0 = 0.1$, and $\eta_{viol} = 1.2$. For the stopping criteria, we used an error tolerance of $\eta_{dc} = 0.1$. For the sandwich algorithm that generates the piecewise affine approximation for our approach and the ECF approach, we chose an absolute error of $\eta = 0.01$ for both examples.

3.6.1 Constrained control of a stochastic double integrator

We first consider a double integrator system,

$$\mathbf{x}_{k+1} = \begin{bmatrix} 1 & T_s \\ 0 & 1 \end{bmatrix} \mathbf{x}_k + \begin{bmatrix} \frac{T_s^2}{2} \\ T_s \end{bmatrix} u_k + \mathbf{w}_k \tag{3.14}$$

with state $\mathbf{x}_k \in \mathbb{R}^2$, input set $\mathcal{U}_k = [-4, 4]$, exponential disturbance $\mathbf{w}_k$ with scale $\overline{\lambda}_{\mathbf{w}_k} \in \mathbb{R}^2_+$, sampling time $T_s = 0.25$s, and initial position $x_0 = [-1\ 0]^\intercal$.

We seek to solve a constrained optimal control problem subject to dynamics (3.14), with quadratic cost (3.2a) that encodes our desire to track $X_d \in \mathbb{R}^{nN}$, penalize high

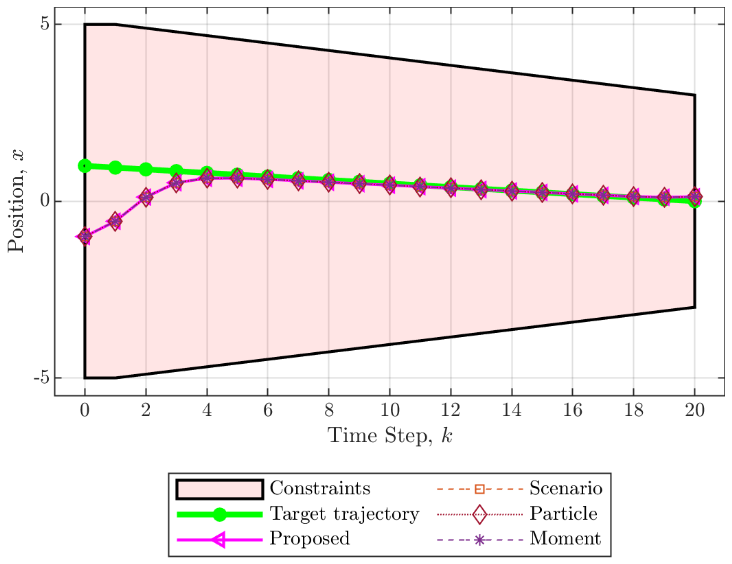


Figure 3.4: Mean trajectories from the proposed approach, scenario approach [1, 2], particle based approach [3], and moment based approach [4, 5]. For all approaches, we presume a constraint violation threshold of $\Delta = 0.1$. Note that all approaches track the reference trajectory.

velocities, and minimize control effort. Specifically, we choose $\mathcal{Q} = \text{diag}([100\ \ 5]) \otimes I_{(nN)\times(nN)}$, $R = 0.5I_{(mN)\times(mN)}$, $(X_d)_k = [m_r k + c_r\ \ 0]^\intercal$, $\forall k \in \mathbb{N}_{[0,N]}$, and problem parameters $m_1, m_2, m_r, c_1, c_2, c_r$ as $0.1, -0.1, -0.05, -5, 5$, and 1 respectively. We define the time varying state constraints as,

$$\mathcal{X}_k = \left\{(k, x) \in \mathbb{N}_{[0,N]} \times \mathbb{R}^2 : m_1 k + c_1 \leq x_1 \leq m_2 k + c_2\right\}$$

and maintain a constraint satisfaction of 95%, i.e. $\Delta = 0.05$.

We compute optimal control trajectories using our approach, the scenario approach, the particle filter approach, and the moment based approach, over a time horizon $N = 20$ and with scale parameter $\overline{\lambda}_{\mathbf{w}} = [10\ 100]^\intercal$. The empirical characteristic function approach cannot be used (Figure 3.6), since the approach requires a concave region of the cumulative distribution function to exist [6, Sec. 3.B.]. Figure 3.4 shows the optimal trajectories for all approaches, where each track the reference trajectory closely while ensuring constraint satisfaction in the presence of uncertainty. Figure 3.5 shows that the stage cost

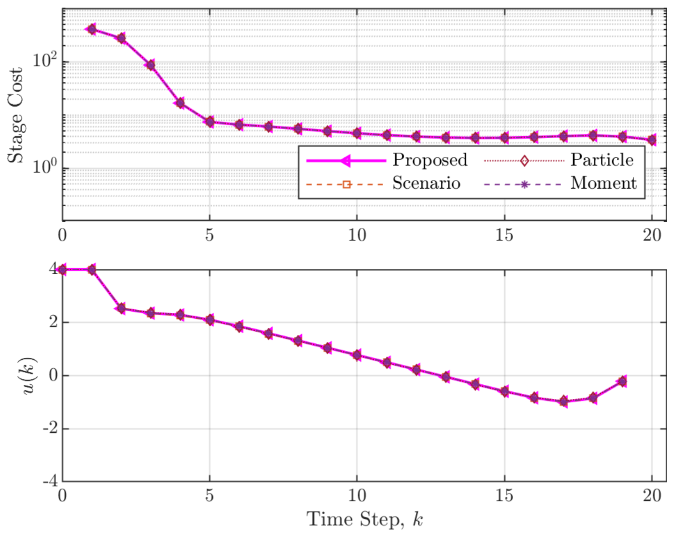


Figure 3.5: Top: Stage cost (the cost incurred at each time step) and control effort over time, for the double integrator. The stage cost of all approaches are similar to highlight that reference tracking is possible for all approaches. Bottom: The optimal input for each approach.

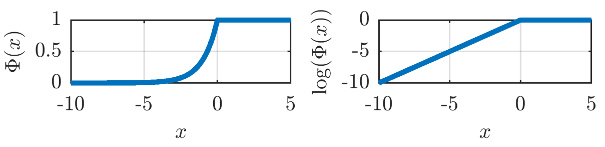


Figure 3.6: The cumulative distribution function (left) and the log of the cumulative distribution function (right) for a negative affine transformation of an exponential random variable with scale parameter $\lambda = 1$. Because the empirical characteristic function approach requires a concave region of the cumulative distribution function to exist [6, Sec. III.B.], it cannot be used to solve the double integrator problem. In contrast, our approach is feasible, since the log of the cumulative distribution function is log-linear.

(the cost at each time step) is similar amongst all approaches except the moment approach. Note that we cannot use the ECF approach here due to the absence of concavity in cumulative distribution function (Figure 3.6).

All methods generate similar trajectories, which track the reference. They also have similar costs and inputs, as shown in Table 3.1, which compares the cost and probability of satisfaction to Monte Carlo estimates for 10^5 simulated trajectories. This example shows that under nominal conditions with non-Gaussian stochasticity, i.e. minimal risk allocation, all methods track the reference trajectory closely.

Table 3.1: Double Integrator example: Cost and constraint satisfaction $(1-\Delta)$ for computed values (Comp) and Monte Carlo (MC) simulation (10^5 samples) for all but the ECF method (See Figure 3.6). We list offline and online computation for all methods where reasonable. Sampling/Particle approaches use $N_s = 91$ samples.

Method	Cost		$1-\Delta$		Time (s)	
	Comp	MC	Comp	MC	Online	Offline
Proposed	863.2	863.2	0.95	1	0.95	56.47
Scenario	862.4	863.2	0.95	1	0.30	N/A
Particle	865.2	863.2	0.95	1	3.50	N/A
Moment	863.2	863.2	0.95	1	3.98	N/A
ECF	N/A	N/A	N/A	N/A	N/A	N/A

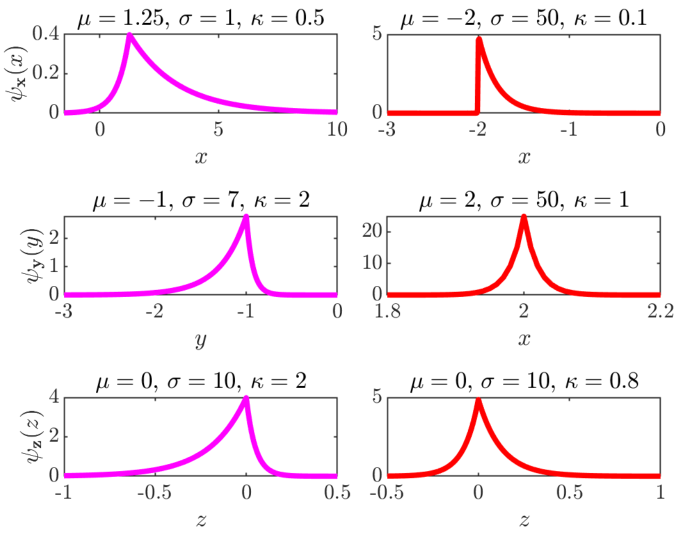


Figure 3.7: The asymmetric Laplace distribution that affects the states representing quadcopter position in x, y, z. The disturbances follow the magenta distributions (left) for the first half of the time horizon, and then follow the red distributions (right) for the second half of the time horizon. The parameters of the distribution are noted above each plot.

3.6.2 Quadrotor in the crosswinds of a harsh environment

We consider a rigid-body quadcopter model [70]. The state is defined as a 12-dimensional vector, $x = [\phi\ \theta\ \psi\ \dot{\phi}\ \dot{\theta}\ \dot{\psi}\ \dot{p}_x\ \dot{p}_y\ \dot{p}_z\ p_x\ p_y\ p_z]^\intercal$, that captures orientation, angular rotation, speed, and position. The net thrust is described by u_1, and the moments around the p_x, p_y, and p_z axes created by the difference in the motor speeds are described by u_2, u_3, and u_4. We presume the mass is $m = 0.478$ kg and the moments of inertia are

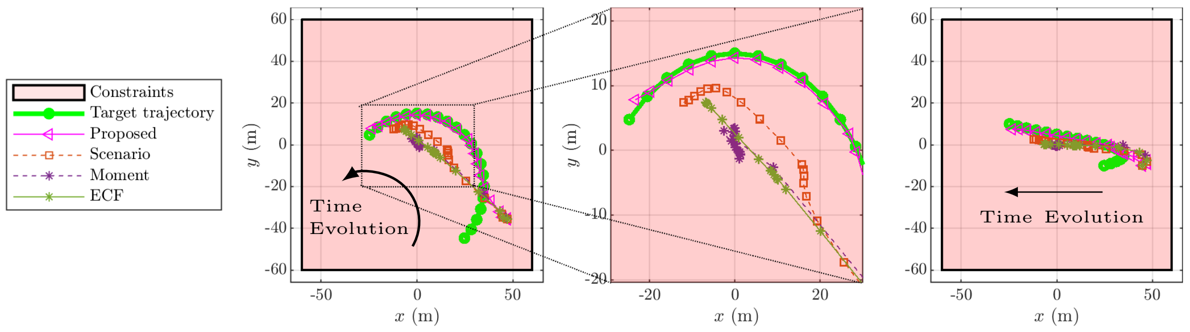


Figure 3.8: Mean trajectories for the quadcopter example. Our approach has the lowest cost, and a probabilistic constraint satisfaction, with a reasonable overall solve time, that is closest (but still above) the desired threshold (Table 3.2). This can be seen in the fact that the trajectory for our approach is close to the reference trajectory (middle plot) compared to the scenario approach which overshoots before recovering to track the reference trajectory. In essence, our approach enables a better trajectory because it can effectively account for the risk of violating the constraint satisfaction in the control optimization process.

$I_{xx} = I_{yy} = 0.0117$ kg m^2 and $I_{zz} = 0.00234$ kg m^2 [71]. We linearize the nonlinear dynamics about a hovering point, and time discretize the dynamics via a zero-order hold with sampling time $T_s = 0.25$.

We incorporate the effect of wind into the quadcopter model as an additive stochastic disturbance, that takes the form of an asymmetric Laplace distribution with characteristic function,

$$\varphi_{\mathbf{w}}(t; \mu, \sigma, \kappa) = \frac{\exp(\mathrm{i}\mu t)}{(1 + \frac{\mathrm{i}t\kappa}{\sigma})(1 - \frac{\mathrm{i}t}{\kappa\sigma})}, \tag{3.15}$$

whose location, shape, and asymmetry suddenly changes halfway through the time horizon, as shown in Figure 3.7, with distribution parameters [72, 73]. That is, the distribution is non-stationary, but independent. We presume the wind directly influences the translational motion, i.e. p_x, p_y, and p_z.

We solve the stochastic optimal control problem (3.2) for a time horizon of $N = 20$ with cost weights $Q = \mathrm{diag}([10 I_{1\times 9N}\ 100 I_{1\times 3N}]) \otimes I_{N\times N}$ and $R = I_{4N\times 4N}$. The desired reference trajectory X_d is defined for the state variables p_x and p_y by generating waypoints

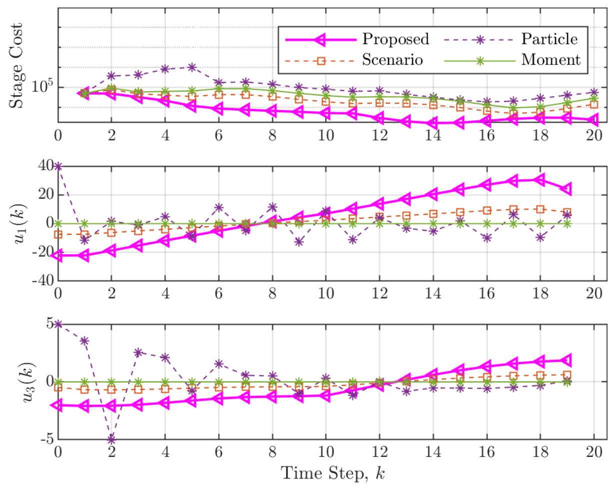


Figure 3.9: The stage cost and input at each time step for all approaches compared in the quadcopter example. Our proposed, moment, and ECF methods have comparable inputs. However, note the scenario approach has differing inputs for u_1 and u_3, and correspondingly higher cost (Table 3.2). The input u_2 and u_4 are not shown because they are quite similar.

via the following functions,

$$X_{\mathrm{d},x,k} = r\sin(\theta_k), \tag{3.16a}$$

$$X_{\mathrm{d},y,k} = r\cos(\theta_k - 20), \tag{3.16b}$$

with $r = 35$, and $\theta_k \in \mathbb{R}$ decreases from $3\pi/4$ to $-\pi/4$. The vertical desired position, $X_{d,x,k} \in \mathbb{R}$ is defined by linearly spaced waypoints from -10 to 10. The input set is $\mathcal{U}_k = [-40, 40] \times [-5, 5]^3$ and the constraint set $\mathcal{X}_k$,

$$\mathcal{X}_k = \left\{x \in \mathbb{R}^{12} : |p_x| \leq 60,\ |p_y| \leq 60,\ |p_z| \leq 60\right\}$$

imposes restrictions on the translational motion. The initial condition is $x(0) = [0\ \cdots\ 0\ \ 44.75\ \ -34.75\ \ -10\ ]^\intercal$.

We choose a constraint satisfaction probability of 95% ($\Delta = 0.05$). Figure 3.8 shows the computed trajectories our approach and those we compare it to. All approaches except for the particle approach find an optimal, open-loop controller, but with varying

Table 3.2: Quadcopter example: Cost and constraint satisfaction $(1-\Delta)$ for computed (Comp) and Monte-Carlo (MC) simulation with 10^5 samples, for all but the particle control method. Sampling and ECF approach use $N_s = 141$ samples.

Method	Cost		$1-\Delta$		Time (s)	
	Comp	MC	Comp	MC	Online	Offline
Proposed	228.4	228.3	0.95	1	0.86	856.8
Scenario	698.4	550.6	0.95	1	1.54	N/A
Particle	N/A	N/A	N/A	N/A	N/A	N/A
Moment	381.9	382.0	0.95	1	266.2	N/A
ECF	932.0	933.1	1	1	0.36	1949.3

conservatism (Table 3.2). The particle approach exceeds our cutoff time of an hour.

In contrast to the first example, only our proposed approach is the only approach which tracks the reference trajectory closely with probabilistic constraint satisfaction (Figure 3.8). As shown in Figure 3.9, our stage cost is the lowest amongst all approaches. Although our approach requires an additional offline calculation for the piecewise under-approximation, the overall cost to solve time is the best out of all approaches relative to online solve time, as shown in Table 3.2.

The piecewise affine approximation for the ECF approach takes longer offline time due to it using a sum of characteristic functions via data to construct the cumulative distribution function. In contrast, since we are given the characteristic function in our approach, the offline time for the piecewise affine approximation is slightly lower. Nonetheless, our approach has a lower cost due to exploiting the log-concavity properties of the distribution. Whereas the ECF approach relies on a conservative concave restriction about the inflection point of the cumulative distribution function (Table 3.2).

3.7 Conclusion

We presented a convex optimization based approach for the constrained, optimal control of a linear dynamical system with additive, log-concave disturbance. Our formulation utilizes a characteristic function based risk allocation technique to assure probabilistic safety for a log-concave disturbance. Our approach solves a tractable difference-of-convex program to synthesize the desired controller. Our reforumulation is amenable to standard conic solvers via the use of piecewise affine approximations that provide tight bounds. Numerical experiments show the efficacy of our approach in comparison to scenario, particle control, and moment based approaches.

Chapter 4

Closed-Loop Steering of Linear Systems With General Uncertainty

4.1 Introduction

In many autonomous systems, the uncertainties that affect the system evolution are quite complex and non-Gaussian. For example, in urban air mobility scenarios, uncertainties may arise from the sensing and perception subsystems, the operating environment (i.e., wind gusts, ground effects), the presence of humans in the loop, or from unmodeled physical phenomena (i.e., higher-order nonlinearities in lift or drag forces) among many others. Methods that ensure robustness to such uncertainties are important for improving the reliability and robustness of autonomous systems. Recently, distribution steering, by which a controller manipulates the stochasticity of the state to guide at states towards a desirable probability distribution, has emerged as a promising approach to *directly* control system uncertainty [74–78]. However, most of these works to date assume Gaussian noise, In this chapter, we describe an approach to synthesize feedback controllers for distribution steering of discrete-time linear systems subject to general (e.g., non-Gaussian) disturbances.

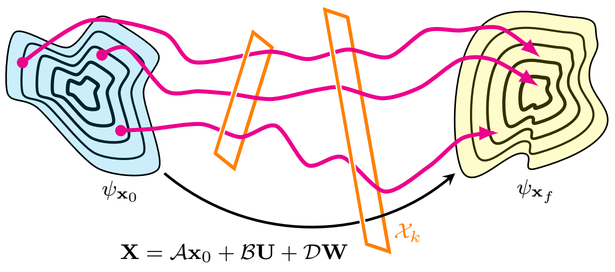

$\psi_{\mathbf{x}_0}$
$\psi_{\mathbf{x}_f}$
$\mathcal{X}_k$
$\mathbf{X} = \mathcal{A}\mathbf{x}_0 + \mathcal{B}\mathbf{U} + \mathcal{D}\mathbf{W}$

Figure 4.1: We seek to steer a stochastic system from an initial distribution to a desired final distribution, subject to probabilistic constraints on the state and input.

4.2 Related Work

Recent work in covariance steering, in which a system is steered from an initial Gaussian distribution to a desired Gaussian distribution, captures the covariance and the mean as extended state variables [76, 79–83]. However, extending this approach to non-Gaussian disturbances is not straightforward. For non-Gaussian disturbances, the distribution is characterized by higher order moments and the computational complexity of the problem increases as the number of moments we need to steer increases. Additionally, when incorporating chance, or probabilistic, constraints into the problem formulation, it is not clear how one can make these constraints tractable (for example, as second order cone constraints through Boole's inequality [84, 85]) with non-Gaussian state evolution, since closed form expressions often do not exist. Methods in optimal transport theory [74, 76] can steer to and from arbitrary distributions, but presume that the disturbance is a Wiener process (i.e., Gaussian increments).

4.3 Main Contribution and Organization

The main contribution of this chapter is the steering of linear systems in the presence of general, non-Gaussian distributions while satisfying state and input constraints with affine feedback. It considers a broad class of disturbances for linear systems, which places very few assumptions on the disturbance probability density function. The approach employs characteristic functions, which circumvent the need to steer all moments individually, and enable straightforward calculation of the absolute distance between distributions. The key insight is that expressions for the chance constraints, previously implemented in an open-loop context [13,15], and the terminal absolute distance constraint can be represented efficiently using characteristic functions and their compositions.

Section 4.4 formulates the problem we wish to solve. We present our analysis of chance constraints within the framework of characteristic functions in Section 4.5. The terminal density matching constraint is presented in 4.6. Section 4.7 presents our reformulation of the constrained stochastic optimal control problem using characteristic functions. Section 4.8 presents an example of a 2D double integrator under various disturbances and initial conditions.

4.4 Problem formulation

Problem 3. *Solve the optimization problem*

$$\begin{aligned} &\underset{\mathbf{U}}{\text{minimize}} \;\; (2.25), \\ &\text{subject to} \;\; (2.10),\; (2.26),\; \text{and} && (4.1a) \\ &\qquad E_0\mathbf{x} \sim \psi_{\mathbf{x}_0},\; E_N\mathbf{x} \sim \psi_{\mathbf{x}_f},\; \mathbf{W} \sim \psi_{\mathbf{W}}. && (4.1b) \end{aligned}$$

The goal of Problem 3 is to minimize the quadratic cost (2.25), satisfy the constraints in (2.10), (2.26), while steering the state of (2.10) from the given initial distribution, $\psi_{\mathbf{x}_0}$, to the desired terminal state distribution, $\psi_{\mathbf{x}_f}$.

4.5 Chance Constraints with Affine Feedback via Characteristic Functions

In this section, we present a decomposition of the chance constraints using Boole's inequality via affine disturbance feedback and represent the probabilistic constraints (also known as chance constraints) using characteristic functions. The resulting constraint is an integral transform over the linear system and the polytopic constraints.

4.5.1 Reformulation of Chance Constraints

State Chance Constraints

The joint state chance constraints in (2.26a) can be transformed into a set of *individual* chance constraints through Boole's inequality [39, 44, 86]

$$\mathbb{P}_{\mathbf{X}}\left(\left\{X \in \mathbb{R}^{nN} : \alpha_{j,k}^{\intercal} E_k X \leq \beta_{j,k}\right\}\right) \geq 1 - \delta_{j,k}^{x}, \quad \sum_{k=1}^{N}\sum_{j=1}^{N_X} \delta_{j,k}^{x} \leq \Delta_X, \tag{4.2}$$

equivalently by the definition of the cumulative distribution function,

$$\Phi_{\alpha_{j,k}^{\intercal} E_k \mathbf{X}}\left(\beta_{j,k}\right) \geq 1 - \delta_{j,k}^{x}, \quad \sum_{k=1}^{N}\sum_{j=1}^{N_X} \delta_{j,k}^{x} \leq \Delta_X, \tag{4.3}$$

where $\delta_{j,k}^{x} \in [0, \Delta_X),\ k \in \mathbb{N}_{[1,N]},\ j \in \mathbb{N}_{[1,N_X]}$. Plugging (2.23a) into (4.3) yields

$$\Phi_{\mathbf{c}_{j,k}}(\beta_{j,k} - \alpha_{j,k}^{\intercal} E_k \mathcal{B} v) \geq 1 - \delta_{j,k}^{x}, \tag{4.4}$$

where $\mathbf{c}_{j,k} = \alpha_{j,k}^{\intercal} E_k (I + \mathcal{B}K)(\mathcal{A}\mathbf{x}_0 + \mathcal{D}\mathbf{W})$.

Input Chance Constraints

Similar to the state chance constraints, we transform the joint chance constraints in (2.26b) into a set of individual chance constraints as follows

$$\mathbb{P}_{\mathbf{U}}\left(\left\{U \in \mathbb{R}^{m(N-1)} : a_{j,k}^{\intercal} F_k U \leq b_{j,k}\right\}\right) \geq 1 - \delta_{j,k}^{u}, \quad \sum_{k=1}^{N}\sum_{j=1}^{N_U} \delta_{j,k}^{u} \leq \Delta_U. \tag{4.5}$$

Equivalently,

$$\Phi_{a_{j,k}^{\intercal} F_k \mathbf{U}}(b_{j,k}) \geq 1 - \delta_{j,k}^{u}, \quad \sum_{k=1}^{N}\sum_{j=1}^{N_U} \delta_{j,k}^{u} \leq \Delta_U, \tag{4.6}$$

where $\delta_{j,k}^{u} \in (0, \Delta_U)$, $k \in \mathbb{N}_{[0,N-1]}$, $j \in \mathbb{N}_{[1,N_U]}$. Using (2.23b) in the chance constraints (4.6) yields

$$\Phi_{\mathbf{d}_{j,k}}(b_j - a_{j,k}^{\intercal} F_k v) \geq 1 - \delta_{j,k}^{u}, \tag{4.7}$$

where $\mathbf{d}_{j,k} = a_{j,k}^{\intercal} F_k K(\mathcal{A}\mathbf{x}_0 + \mathcal{D}\mathbf{W})$.

4.5.2 Encoding Chance Constraints in the Presence of Affine Feedback

To illustrate how to encode the chance constraints (4.4) and (4.7) via characteristic functions, consider first the state chance constraint in (4.4). Expanding the random variable $\mathbf{c}_{j,k}$ we can write

$$\mathbf{c}_{j,k} = \mu_{\mathbf{c},j,k}^{\intercal} \mathbf{x}_0 + \nu_{\mathbf{c},j,k}^{\intercal} \mathbf{W}, \tag{4.8}$$

where $\mu_{\mathbf{c},j,k}^{\intercal} = \alpha_{j,k}^{\intercal} E_k (I + \mathcal{B}K)\mathcal{A}$ and $\nu_{\mathbf{c},j,k}^{\intercal} = \alpha_{j,k}^{\intercal} E_k (I + \mathcal{B}K)\mathcal{D}$ are non-random variables that are linear in the decision variable K. Under Assumption 1, Property 3 of characteristic functions allows us to decompose $\varphi_{\mathbf{c}_{j,k}}$ as

$$\varphi_{\mathbf{c}_{j,k}}(t) = \varphi_{\mu^\intercal_{\mathbf{c},j,k}\mathbf{x}_0}(t)\varphi_{\nu^\intercal_{\mathbf{c},j,k}\mathbf{W}}(t). \tag{4.9}$$

Next, Property 2 yields

$$\varphi_{\mu^\intercal_{\mathbf{c},j,k}\mathbf{x}_0}(t) = \varphi_{\mathbf{x}_0}(t^{\mathbf{c}}_{\mathbf{x}_0}), \tag{4.10a}$$

$$\varphi_{\nu^\intercal_{\mathbf{c},j,k}\mathbf{W}}(t) = \varphi_{\mathbf{W}}(t^{\mathbf{c}}_{\mathbf{W}}). \tag{4.10b}$$

where $t^{\mathbf{c}}_{\mathbf{x}_0} = \mu_{\mathbf{c},j,k}t$ and $t^{\mathbf{c}}_{\mathbf{W}} = \nu_{\mathbf{c},j,k}t$. Finally, Property 4 yields

$$\varphi_{\mathbf{x}_0}(t^{\mathbf{c}}_{\mathbf{x}_0}) = \prod_{i=1}^{n} \varphi_{\mathbf{x}_{0,i}}(t^{\mathbf{c}}_{\mathbf{x}_{0,i}}), \tag{4.11a}$$

$$\varphi_{\mathbf{W}}(t^{\mathbf{c}}_{\mathbf{W}}) = \prod_{i=1}^{pN} \varphi_{\mathbf{w}_i}(t^{\mathbf{c}}_{\mathbf{w}_i}). \tag{4.11b}$$

where $t^{\mathbf{c}}_{\mathbf{x}_{0,i}} = e^\intercal_{i,n}t^{\mathbf{c}}_{\mathbf{x}_0}$ and $t^{\mathbf{c}}_{\mathbf{w}_i} = e^\intercal_{i,pN}t^{\mathbf{c}}_{\mathbf{w}}$. Theorem 1 gives an analytical expression for the cdf of $\mathbf{c}_{j,k}$ evaluated at $\gamma_{j,k} = \beta_{j,k} - \alpha^\intercal_{j,k}E_k\mathcal{B}v$ as

$$\Phi_{\mathbf{c}_{j,k}}(\gamma_{j,k}) = \frac{1}{2} - \frac{1}{\pi}\int_0^\infty \frac{1}{t}\mathrm{Im}\left(e^{-\mathrm{i}t\gamma_{j,k}}\varphi_{\mathbf{c}_{j,k}}(t)\right)\mathrm{d}t, \tag{4.12}$$

where $\varphi_{\mathbf{c}_{j,k}}(t)$ is given in (4.9). Thus, (4.12) provides a means to compute the cdf in the state chance constraints (4.4), and encodes the decision variables K and v through Theorem 1.

Similarly, we can also derive the input chance constraints in (4.7) for the random variable $\mathbf{d}_{j,k}$ by rewriting

$$\mathbf{d}_{j,k} = \mu^\intercal_{\mathbf{d},j,k}\mathbf{x}_0 + \nu^\intercal_{\mathbf{d},j,k}\mathbf{W}, \tag{4.13}$$

where $\mu^\intercal_{\mathbf{d},j,k} = a^\intercal_{j,k}F_kK\mathcal{A}$ and $\nu^\intercal_{\mathbf{d},j,k} = a^\intercal_{j,k}F_kK\mathcal{D}$. From Property 3 of characteristic functions, the characteristic function of $\mathbf{d}_{j,k}$ is therefore

$$\varphi_{\mathbf{d}_{j,k}}(t) = \varphi_{\mu_{\mathbf{d},j,k}^{\intercal}\mathbf{x}_0}(t)\varphi_{\nu_{\mathbf{d},j,k}^{\intercal}\mathbf{W}}(t). \tag{4.14}$$

Further, by Properties 2 and 4, we have

$$\varphi_{\mu_{\mathbf{d},j,k}^{\intercal}\mathbf{x}_0}(t) = \varphi_{\mathbf{x}_0}(t_{\mathbf{x}_0}^{\mathbf{d}}) = \prod_{i=1}^{n} \varphi_{\mathbf{x}_{0,i}}(t_{\mathbf{x}_{0,i}}^{\mathbf{d}}), \tag{4.15a}$$

$$\varphi_{\nu_{\mathbf{d},j,k}^{\intercal}\mathbf{W}}(t) = \varphi_{\mathbf{W}}(t_{\mathbf{W}}^{\mathbf{d}}) = \prod_{i=1}^{pN} \varphi_{\mathbf{w}_i}(t_{\mathbf{w}_i}^{\mathbf{d}}) \tag{4.15b}$$

where $t_{\mathbf{x}_0}^{\mathbf{d}} = \mu_{\mathbf{d},j,k}t$, $t_{\mathbf{W}}^{\mathbf{d}} = \nu_{\mathbf{d},j,k}t$, $t_{\mathbf{x}_{0,i}}^{\mathbf{d}} = e_{i,n}^{\intercal}t_{\mathbf{x}_0}^{\mathbf{d}}$, and $t_{\mathbf{w}_i}^{\mathbf{d}} = e_{i,pN}^{\intercal}t_{\mathbf{W}}^{\mathbf{d}}$. Thus, the expression for the cdf of $\mathbf{d}_{j,k}$ evaluated at $\gamma_{\mathbf{d},j,k} = b_j - a_j^{\intercal}F_k v$ is given by

$$\Phi_{\mathbf{d}_{j,k}}(\gamma_{j,k}) = \frac{1}{2} - \frac{1}{\pi}\int_0^{\infty} \frac{1}{t}\mathrm{Im}\left(e^{-\mathrm{i}t\gamma_{\mathbf{d},j,k}}\varphi_{\mathbf{d}_{j,k}}(t)\right)\mathrm{d}t. \tag{4.16}$$

Note that the constraints encoded by the characteristic function in (4.12) and (4.16) result in *nonlinear* constraints in terms of the decision variables K and v.

4.6 Terminal Density Constraints

Our approach aims at matching probability densities using the machinery of characteristic functions. The benefit of using characteristic functions to match between densities as opposed to other metrics such as KL-divergence or Wasserstein distance is two-fold. First, it can be shown that the largest absolute difference between two pdfs is bounded by the L_1 difference of their characteristic functions. Second, this holds for all distributions (including mixture distributions) which have a characteristic function in $L_1(\mathbb{R}^n)$ and directly results in an explicit integral expression over the frequency domain, and not an integration over the entire state-space [75]. This is convenient, as it is difficult to formulate the terminal constraints analytically in the state-space due to the non-Gaussian state evolution requiring several convolutions at each time step (see Property 1 of operations on characteristic functions).

Next, we first derive a joint distribution representation which results in an n-dimensional integral. We then show that due to the independence property of the disturbances, we can compute this integral using n separate matching constraints at the final time.

4.6.1 Joint Characteristic Function Representation of the Terminal Density

Using Properties 1 and 2 of operations on characteristic functions, the joint characteristic function of the terminal state $\mathbf{x}_N$ is

$$\varphi_{\mathbf{x}_N}(t) = \prod_{i=1}^{n} \exp(\mathrm{i}\sigma_i^{\intercal} t)\varphi_{\mathbf{x}_0}(t_{\mathbf{x}_0}^N)\varphi_{\mathbf{w}}(t_{\mathbf{w}}^N), \tag{4.17}$$

where

$$\varphi_{\mathbf{x}_0}(t_{\mathbf{x}_0}^N) = \prod_{j=1}^{n} \varphi_{\mathbf{x}_{0,j}}(t_{\mathbf{x}_{0,j}}^N), \tag{4.18}$$

$$\varphi_{\mathbf{W}}(t_{\mathbf{W}}^N) = \prod_{j=1}^{pN} \varphi_{\mathbf{w}_j}(t_{\mathbf{w}_j}^N), \tag{4.19}$$

and where $t_{\mathbf{x}_0}^N = \mu_i t$, $t_{\mathbf{x}_{0,j}}^N = e_{j,n}^{\intercal} t_{\mathbf{x}_0}^N, t_{\mathbf{w}}^N = \nu_i t$, $t_{\mathbf{w}_j}^N = e_{j,pN}^{\intercal} t_{\mathbf{w}}^N$, and $\mu_i^{\intercal} = e_{i,n}^{\intercal} E_N(I + \mathcal{B}K)\mathcal{A}$, $\nu_i^{\intercal} = e_{i,p}^{\intercal} E_N(I + \mathcal{B}K)\mathcal{D}$, and $\sigma_i^{\intercal} = e_{i,m}^{\intercal} E_N \mathcal{B} v$. Similarly, the joint characteristic function of the *desired* terminal state is

$$\varphi_{\mathbf{x}_f}(t) = \prod_{i=1}^{n} \varphi_{\mathbf{x}_{f,i}}(t_i). \tag{4.20}$$

We now introduce the L_1 distance as an upper bound on the maximum L_1 deviation between two probability distributions.

Theorem 4.1 ([27, Sec. 1.4]). *If the joint pdf for the terminal state of the system is $\psi_{\mathbf{x}_N}$ with characteristic function (4.17) and the desired joint pdf is $\psi_{\mathbf{x}_f}$ with characteristic function (4.20), then*

$$\Delta_{\psi_{\mathbf{x}_N}}(K,v) = \sup_{z\in\mathbb{R}^n} |\psi_{\mathbf{x}_N}(z;K,v) - \psi_{\mathbf{x}_f}(z)| \leq D(K,v), \tag{4.21}$$

where

$$D(K,v) = \left(\frac{1}{2\pi}\right)^n \|\varphi_{\mathbf{x}_N} - \varphi_{\mathbf{x}_f}\|_1 = \left(\frac{1}{2\pi}\right)^n \int_{\mathbb{R}^n} \left|\varphi_{\mathbf{x}_N}(t) - \varphi_{\mathbf{x}_f}(t)\right| \mathrm{d}t. \tag{4.22}$$

Proof. Since

$$\begin{aligned} &\Big|\Big|\int_{\mathbb{R}^n} \exp(\mathrm{i}t^\intercal z)\varphi_{\mathbf{x}_N}(t)\mathrm{d}t\Big| - \Big|\int_{\mathbb{R}^n} \exp(\mathrm{i}t^\intercal z)\varphi_{\mathbf{x}_f}(t)\mathrm{d}t\Big|\Big| \\ &\qquad\qquad \leq \int_{\mathbb{R}^n} \left|\exp(\mathrm{i}t^\intercal z)\varphi_{\mathbf{x}_N}(t) - \exp(\mathrm{i}t^\intercal z)\varphi_{\mathbf{x}_f}(t)\right| \mathrm{d}t, \end{aligned} \tag{4.23}$$

by multiplying both sides by $(1/2\pi)^n$ and using (2.2), yields

$$|\psi_{\mathbf{x}_N}(z) - \psi_{\mathbf{x}_f}(z)| \leq \left(\frac{1}{2\pi}\right)^n \|\varphi_{\mathbf{x}_N} - \varphi_{\mathbf{x}}\|_1. \tag{4.24}$$

Lastly, since this holds *for all* $z \in \mathbb{R}^n$, we get (4.21). □

Corollary 4.1. *Let $\epsilon > 0$ such that $D(K,v) < \epsilon$ for some K and v. Then,* $\sup_{z\in\mathbb{R}^n} |\psi_{\mathbf{x}_N}(z; K,v) - \psi_{\mathbf{x}_f}(z)| \leq \epsilon$.

4.6.2 Matching Densities

We derive a simpler representation of the $n-$dimensional integral of the joint representation in (4.21). Specifically, we construct n separate density matching expressions with respect to each terminal state variable, that is, for all $i \in \mathbb{N}_{[1,n]}$,

$$\Delta_{\psi_{\mathbf{x}_N},i}(K, v) = \sup_{z_i \in \mathbb{R}} \left| \psi_{\mathbf{x}_N,i}(z_i; K, v) - \psi_{\mathbf{x}_{f,i}}(z_i) \right| \leq D_i(K, v), \tag{4.25}$$

where

$$D_i(K, v) = \frac{1}{2\pi} \int_{\mathbb{R}} \left| \varphi_{\mathbf{x}_N,i}(t_i) - \varphi_{\mathbf{x}_f,i}(t_i) \right| \mathrm{d}t, \tag{4.26}$$

which follows from Theorem 4.1. The next result provides a relationship between (4.22) and (4.26).

Theorem 4.2. *Suppose there exists (K, v) such that, for all $i \in \mathbb{N}_{[1,n]}$, $D_i(K, v) \leq \epsilon_i$. Then, $D(K, v) \leq (1/2\pi)^{n-1}\epsilon$, where $\epsilon = \sum_i \epsilon_i$.*

Proof. By definition of the L_1 distance, for each $i \in \mathbb{N}_{[1,N]}$,

$$\frac{1}{2\pi} \int_{\mathbb{R}} |\varphi_{e_{i,n}^{\intercal}\mathbf{x}_N}(t_i) - \varphi_{\mathbf{x}_{f,i}}(t_i)| \, \mathrm{d}t_i \leq \epsilon_i. \tag{4.27}$$

Let $a_i = \varphi_{e_{i,n}^{\intercal}\mathbf{x}_N}(t_i)$ and $b_i = \varphi_{\mathbf{x}_{f,i}}(t_i)$. Then, since $|\varphi_{e_{i,n}^{\intercal}\mathbf{x}_N}(t_i)|$, $|\varphi_{\mathbf{x}_{f,i}}(t_i)| \leq 1$, it follows that $|\varphi_{\mathbf{x}_N}(t) - \varphi_{\mathbf{x}_f}(t)| = |\prod_i \varphi_{e_{i,n}^{\intercal}E_N\mathbf{x}}(t_i) - \prod_i \varphi_{\mathbf{x}_{f,i}}(t_i)| \leq \sum_i |\varphi_{e_{i,n}^{\intercal}\mathbf{x}_N}(t_i) - \varphi_{\mathbf{x}_{f,i}}(t_i)|$, where we have used the fact that for $a_i,\ b_i \in \mathbb{C},\ i \in \mathbb{N}_{[1,n]}$ where $|a_i|, |b_i| \leq 1$, $|\prod_i a_i - \prod_i b_i| \leq \sum_i |a_i - b_i|$. The result now follows immediately from the definition of $D(K, v)$. □

4.7 Resulting optimization problem

With the elements derived for both the chance constraints and the terminal distribution constraint, we present the resulting optimization problem.

Problem 4. *Solve the optimization problem*

$$\begin{aligned}
\underset{K,v,\delta^x,\delta^u}{\text{minimize}} \quad & J(K,v)+\sum_{i=1}^{n}\lambda_i D_i(K,v) && \text{(4.28a)}\\
\text{subject to} \quad & \Phi_{\mathbf{c}_{j,k}}(\beta_{j,k}-\alpha_{j,k}^\intercal E_k \mathcal{B} v) \geq 1-\delta^x_{j,k}, && \text{(4.28b)}\\
& \Phi_{\mathbf{d}_{j,k}}(b_j - a_{j,k}^\intercal F_k v) \quad \geq 1-\delta^u_{j,k}, && \text{(4.28c)}\\
& \sum_{k=1}^{N}\sum_{j=1}^{N_X}\delta^x_{j,k}\leq \Delta_X, \quad \sum_{k=1}^{N}\sum_{j=1}^{N_U}\delta^u_{j,k}\leq \Delta_U, && \text{(4.28d)}
\end{aligned}$$

where $J(K,v)$ is the expanded quadratic cost (2.25) *and is defined as*

$$\begin{aligned}
J(K,v) = & \left[(I+\mathcal{B}K)(\mathcal{A}\mathbb{E}[\mathbf{x}_0]+\mathcal{D}\mathbb{E}[\mathbf{W}])+\mathcal{B}v-X_d\right]^\intercal \mathcal{Q}\cdot\\
& \left[(I+\mathcal{B}K)(\mathcal{A}\mathbb{E}[\mathbf{x}_0]+\mathcal{D}\mathbb{E}[\mathbf{W}])+\mathcal{B}v-X_d\right]\\
& +\left[K(\mathcal{A}\mathbb{E}[\mathbf{x}_0]+\mathcal{D}\mathbb{E}[\mathbf{W}])+v\right]^\intercal \mathcal{R}\left[K(\mathcal{A}\mathbb{E}[\mathbf{x}_0]+\mathcal{D}\mathbb{E}[\mathbf{W}])+v\right]\\
& +\operatorname{tr}\left[\left((I+\mathcal{B}K)^\intercal\mathcal{Q}(I+\mathcal{B}K)+K^\intercal\mathcal{R}K\right)\Sigma\right], \qquad \text{(4.29)}
\end{aligned}$$

where $\Sigma=\mathcal{A}\Sigma_{\mathbf{x}_0}\mathcal{A}^\intercal+\mathcal{D}\Sigma_{\mathbf{W}}\mathcal{D}^\intercal$.

We treat the matching constraint as a soft constraint, as in [75]. By penalizing this L_1 distance in the cost, we provide flexibility to the underlying nonlinear program solver and enable increased feasibility.

4.8 Examples

We demonstrate our approach on a 2D double integrator with different disturbances and initial conditions. Consider the system (2.9) with state $x=[x\ \dot{x}\ y\ \dot{y}]^\intercal$. The expressions for the system matrices A_k, B_k, and D_k are given in [83] with $\Delta T=1$ and $N=5$. We assume polytopic state constraints $\mathcal{X}_k$, with $\alpha_{1k}=\begin{bmatrix}1 & 1 & 0 & 0\end{bmatrix}$, $\beta_{1k}=12.75$, $\alpha_{2k}=\begin{bmatrix}1 & 0.1 & 0 & 0\end{bmatrix}$, $\beta_{2k}=8.75$ for $k\in\mathbb{N}_{[1,N]}$, and assume $\mathbf{x}_0\sim\psi_{x_0}$ must be within the state polytopic constraints, as well. We let $\mathcal{U}_k = [-4, 4]^2$ for $k \in \mathbb{N}_{[0,N-1]}$. The desired trajectory X_d is interpolated from waypoints $(4, 5)$ to $(8, 5)$ for $k \in \mathbb{N}_{[0,2]}$ and from $(9, 5)$ to $(7.875, 3)$ for $k \in \mathbb{N}_{[3,5]}$. We seek to drive the final state to $\mathbf{x}_N \sim \psi_{\mathbf{x}_f} = \mathcal{N}(\mu_{\mathbf{x}_f}, \Sigma_{\mathbf{x}_f})$ with mean $\mu_{\mathbf{x}_f} = [7.75\ 2\ 0\ 0]^\intercal$ and variance $\Sigma_{\mathbf{x}_f} = \mathrm{diag}([0.06\ 0.006\ 0.6\ 0.006])$. We choose $\Delta_X = 0.1$ and $\Delta_U = 0.1$, and $Q_k = \mathrm{diag}([10\ 1\ 10\ 1])$, $R_k = \mathrm{diag}([1\ 1])$ for $i \in \mathbb{N}_{[0,N-1]}$. The weighting of the distance metrics in the cost is $\lambda = [10\ 1\ 10\ 1]^\intercal$.

All computations were done in MATLAB with an Intel Core i9-10900K processor and 64GB RAM. The optimization problems were solved using `fmincon`. The characteristic function inversion (4.12) uses CharFunTool [30] and the density matching constraint in (4.26) was implemented using trapezoidal quadrature. We used 10^4 Monte-Carlo samples to verify average state and input constraint violation (denoted as $\Delta_{\mathrm{X,MC}}$ and $\Delta_{\mathrm{U,MC}}$, respectively) and cost (denoted as $J_{\mathrm{MC}}(K, v)$).

4.8.1 Double Integrator: Standard Gaussian Distribution

To validate our approach, we first considered $\mathbf{x}_0 \sim \psi_{\mathbf{x}_0} = \mathcal{N}(\mu_{\mathbf{x}_0}, \Sigma_{\mathbf{x}_0})$ with $\mu_{\mathbf{x}_0} = [4\ 0\ 5\ 0]^\intercal$ and $\Sigma_{\mathbf{x}_0} = \mathrm{diag}([0.18\ 0.002\ 0.18\ 0.002])$; the disturbance is $\mathcal{N}(\mu_{\mathbf{w}_k}, \Sigma_{\mathbf{w}_k})$ with mean $\mu_{\mathbf{w}_k} = [0\ 0]^\intercal$ and variance $\Sigma_{\mathbf{w}_k} = \mathrm{diag}([1\ 1])$ for the entire horizon. As shown in Figure 4.2a, our method drives the system to follow the reference trajectory, while not significantly violating the state constraints (Table 4.2). Likewise, input violation is minimal, as shown in Figure 4.2b and Table 4.2. Since we steer the state from an initial Gaussian distribution to a final Gaussian distribution, the maximum deviation between the final and desired pdfs and the corresponding L_1 distances are small (Table 4.1).

4.8.2 Double Integrator: Long Tail - Laplace Distribution

Heavy-tailed distributions are of interest as they decay much more slowly than Gaussians, but with a similar mean and variance. The Laplace distribution has the pdf $\mathcal{L}_{\mu,\beta}(x) = \exp\left(-|x - \mu|/\beta\right)/2\beta$. We assume the initial condition $\mathbf{x}_0 \sim \mathcal{L}(\mu_{\mathbf{x}_0}, \beta_{\mathbf{x}_0})$ with location

$\mu_{\mathbf{x}_0} = [4\ 0\ 5\ 0]^\intercal$ and scale $\beta_{\mathbf{x}_0} = [0.3\ 0.01\ 0.3\ 0.01]^\intercal$. The disturbance also follows a Laplace distribution $\mathbf{w}_k \sim \mathcal{L}(\mu_{\mathbf{w}_k}, \beta_{\mathbf{w}_k})$ with location $\mu_{\mathbf{w}_k} = [0\ 0]^\intercal$ and scale $\beta_{\mathbf{w}_k} = [1\ 1]^\intercal$. Although the Laplace distribution is not smooth (Figure 4.3a), our method is able to steer to the final desired density with little constraint violation (Table 4.2). The input in Figure 4.3b shows that there is some violation of the bounds, but it is within the violation threshold (Table 4.2). The larger deviation between the final and desired pdfs (Table 4.1) reflects the fact that we modify a random variable that is not Gaussian so as to behave like a Gaussian one.

4.8.3 Double Integrator: Mixture Distributions - Normal Mixture

Lastly, we consider a Gaussian mixture with $\mathbf{x}_0 \sim \psi_{\mathbf{x}_0} = 0.5\mathcal{N}(\mu_{\mathbf{x}_{0,1}}, \Sigma_{\mathbf{x}_{0,1}}) + 0.5\mathcal{N}(\mu_{\mathbf{x}_{0,2}}, \Sigma_{\mathbf{x}_{0,2}})$ with means $\mathbf{x}_{0,1} = [4\ 0\ 5\ 0]^\intercal$, $\mathbf{x}_{0,2} = [3.5\, 0.1\, 3.5\, 0.1]^\intercal$ and covariances $\Sigma_{\mathbf{x}_{0,1}} = \mathrm{diag}([0.3\ \ 0.01\, 0.3\, 0.01])$, $\Sigma_{\mathbf{x}_{0,2}} = \mathrm{diag}([0.1\ 0.01\ 0.5\ 0.01])$. The disturbance is a Gaussian mixture, $\mathbf{w} \sim \psi_{\mathbf{w}_k} = 0.5\mathcal{N}(\mu_{\mathbf{w}_{1,k}}, \Sigma_{\mathbf{w}_{1,k}}) + 0.5\mathcal{N}(\mu_{\mathbf{w}_{2,k}}, \Sigma_{\mathbf{w}_{2,k}})$, with means $\mu_{\mathbf{w}_{1,k}} = [0\ 0.1]^\intercal$, $\mu_{\mathbf{w}_{2,k}} = [0.1\ \ 0]^\intercal$ and covariances $\Sigma_{\mathbf{w}_{1,k}} = \mathrm{diag}([1\ \ 1])$, $\Sigma_{\mathbf{w}_{2,k}} = \mathrm{diag}([1\ \ 1])$. The affine controller steers the Gaussian mixture to a single Gaussian (Figure 4.4a) with minimal violation of the state constraints (Table 4.2). The input remains within acceptable limits (Table 4.2) despite the multi-modal nature of the noise (Figure 4.4b). The deviation between final and desired pdfs are much smaller than seen with the Laplace pdf (Table 4.1). This is likely because the controller alters the weights of the multi-modal Gaussian elements to match the desired, final Gaussian density.

4.9 Conclusion

We have formulated a tractable solution of the distribution steering problem under general, not necessarily Gaussian, disturbances. We showed that using Boole's inequality

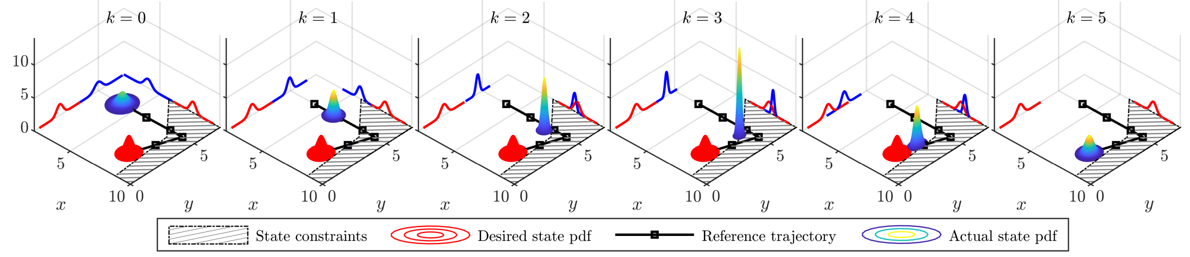
$\mathbf{W}$	$\Delta_{\psi_{\mathbf{x}_N},1}$	D_1	$\Delta_{\psi_{\mathbf{x}_N},2}$	D_2	$\Delta_{\psi_{\mathbf{x}_N},3}$	D_3	$\Delta_{\psi_{\mathbf{x}_N},4}$	D_4
(4.8.1)	0.000383	0.0026	1.69	10.63	3.36E − 6	2.11E − 5	4.29	33.38
(4.8.2)	0.037	0.346	0.077	0.6838	0.096	0.270	4.46	33.16
(4.8.3)	0.002	0.016	0.001	0.011	$2.94E-4$	0.031	4.94	31.87

Table 4.1: The largest deviation between the actual and desired pdfs (4.25), compared to the L_1 distance (4.26) for each scenario. The controller from (4.28) yields an L_1 distance that upper bounds the largest deviation in each case.

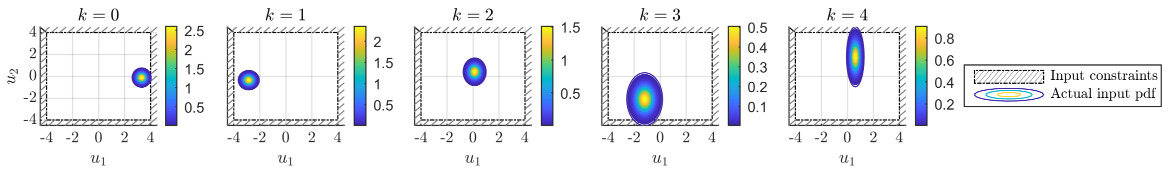
$\mathbf{W}$	$J(K,v)$	$J_{\text{MC}}(K,v)$	Δ_X	$\Delta_{X,\text{MC}}$	Δ_U	$\Delta_{U,\text{MC}}$
(4.8.1)	53.17	49.36	0.098	0.052	0.0975	7E − 4
(4.8.2)	41.84	41.81	0.0983	0.0572	0.0977	0.0103
(4.8.3)	49.74	46.18	0.098	0.009	0.098	0.015

Table 4.2: Cost and risk allocation (for the state and the input) when solving (4.28), and averaged values from 10^4 Monte-Carlo (MC) samples for validation. The MC average cost is consistent with the computed cost, and the MC state and input constraint violations are lower than the computed violations.

and characteristic functions, we can turn the problem into a nonlinear optimization problem. Future work will aim to further utilize the structure of the characteristic functions to obtain faster, real-time solutions and extend the approach to nonlinear systems.

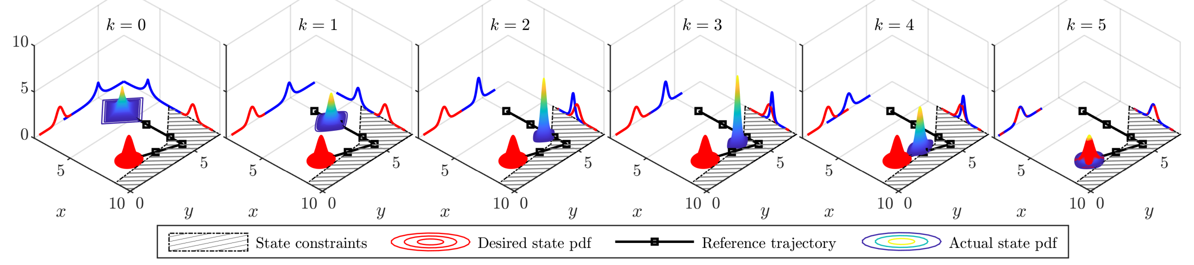


(a) State evolution (x and y) over 5 timesteps, subject to state chance constraints and terminal density constraints. The system is steered from the initial density to the final, desired density without collision, even though the reference trajectory violates the constraints.



(b) Inputs u_1 and u_2 satisfy input chance constraints with violation less than Δ_U.

Figure 4.2: Distribution steering from one Gaussian distribution to another Gaussian distribution.


(a) State evolution (x and y) over 5 time steps, subject to state chance and terminal density constraints. The Laplace distribution is non-smooth at its peak, and is heavy-tailed. Our approach drives the system from a Laplace distribution to a Gaussian distribution, while maintaining state constraint violation below Δ_X.

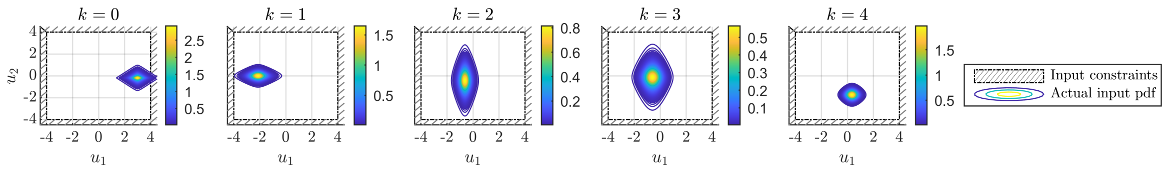
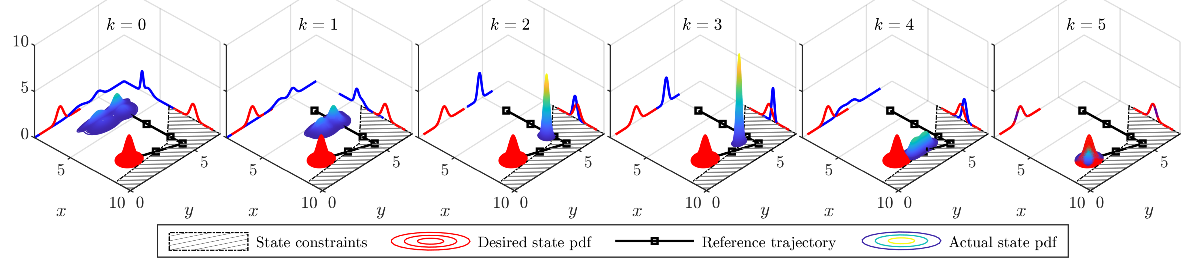

(b) Inputs u_1 and u_2 satisfy input chance constraints with violation less than Δ_U.

Figure 4.3: Distribution steering from a Laplace distribution to a Gaussian distribution.


(a) State evolution (x and y) over timesteps, subject to the state chance and terminal density constraints. The multi-modal Gaussian is transformed into a Gaussian with a single mode with minimal state constraint violation.

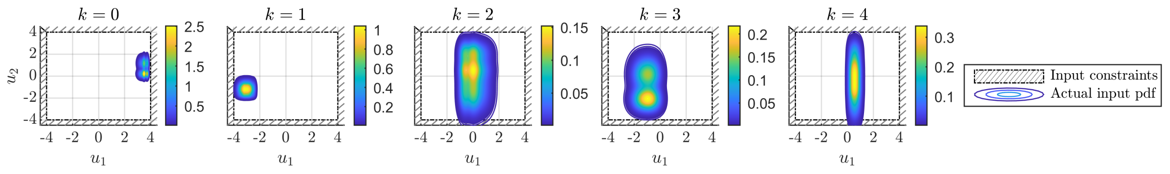

(b) Inputs u_1 and u_2 satisfy input chance constraints with violation less than Δ_U.

Figure 4.4: Distribution steering from a Gaussian mixture to a Gaussian distribution.

Part II

Data-Driven, Stochastic Optimal Control

Chapter 5

Open-Loop Control of Linear Systems With Unknown Uncertainty

5.1 Introduction

Stochastic optimal control typically presumes accurate models of the underlying dynamics and stochastic processes [39, 87, 88]. However, in many circumstances, accurate characterization of uncertainty is difficult. Further, inaccurate characterization of stochastic processes may have unexpected impacts [89, 90], as optimal control actions are typically dependent upon the first and second moments of the stochastic processes [88]. Such inaccuracies could be particularly problematic when the unknown stochastic processes is asymmetric, multimodal, or heavy-tailed. For example, in hypersonic vehicles, excessive turbulence makes aerodynamic processes difficult to model accurately, and their fast time-scale means that erroneous control actions could result in catastrophic failure.

5.2 Related Work

We consider the case in which the dynamics are known, but the noise process is not known, and focus on the problem of data-driven stochastic optimal control in a chance constrained setting, in which probabilistic constraints must be satisfied with at least a desired likelihood. Some approaches, such as distributional stochastic optimal control, seek robustness to ill-defined distributions with finite samples [90,91]. Other approaches construct piecewise-affine over-approximations of value functions by solving a chance-constrained problem [92]. Researchers have also employed kernel density estimation [93,94] to approximate individual chance constraints in nonlinear optimization problems.

5.3 Main Contribution and Organization

One tool to characterize uncertainty through observed data is the *empirical characteristic function* [95], which is often employed in economics and statistics to characterize models where maximum-likelihood estimation can struggle. The empirical characteristic function generates an approximation of the true characteristic function, and has known convergence properties [96,97]. The advantage of this approach is that it enables direct, closed-form approximation of the cumulative distribution function and the moments of the underlying stochastic process [95], both of which are typically necessary for stochastic optimal control problems. However, the main challenge then becomes one of finding computationally efficient under-approximations of the resulting cumulative distribution function, which may be non-convex.

We propose to employ empirical characteristic functions to characterize unknown disturbance processes in a linear, time-invariant dynamical system with a quadratic cost function. We construct a conic, convex reformulation of the resulting stochastic optimal control problem, that ensures computational tractability [98]. Our approach employs a piecewise under-approximation of the approximate cumulative distribution function, with a user-specified trade-off between accuracy and the number of piecewise elements. We use confidence intervals on the approximate cumulative distribution function to provide probabilistic bounds on the solution to the data-driven stochastic optimal control problem. *The main contribution of this paper is the construction of a convex, conic reformulation of a stochastic optimal control problem in the presence of an unknown, additive disturbance, via empirical characteristic functions, with confidence bounds on the optimal solution.*

The outline of the paper is as follows. We first formulate the problem in Section 5.4. Section 5.5 presents algorithms to convexify the problem and proofs of its convergence properties. In Section 5.6, we demonstrate our approach on two examples.

5.4 Problem Statement

Problem 5. *Solve the optimization problem similar to* (3.2),

$$\begin{aligned}
\underset{U}{\text{minimize}} \quad & \mathbb{E}\left[(\mathbf{X}-X_d)^\intercal \mathcal{Q}(\mathbf{X}-X_d)\right] + U^\intercal \mathcal{R} U & (5.1\text{a})\\
\text{subject to} \quad & (3.1),\ (2.26\text{a}) & \\
& \bigcap_{k=0}^{N-1} F_k U \in \mathcal{U}_k, & (5.1\text{b})
\end{aligned}$$

with positive semi-definite matrices as in (2.25)*, and polytopic state and input constraints as in* (2.27)*.* without *direct knowledge of the cumulative distribution function or moments of* $\mathbf{w}$*, but* with *observations of* N_s *samples* $\{\mathbf{w}_j\}_{j=1}^{N_s}$.

The standard approach to solving (5.1) when the disturbance process is well characterized is to tighten the joint chance constraint (2.26a) via individual chance constraints [99, 100]. However, two main challenges then arise: 1) reliance of (2.25) and (2.26a) upon moments and the cumulative distribution function, respectively, of the unknown noise process, and 2) non-convexity of the individual chance constraints. The former can be seen from expanding (2.25),

$$(\mu_{\mathbf{X},U}-X_d)^\intercal Q(\mu_{\mathbf{X},U}-X_d) + U^\intercal R U + \mathrm{tr}(Q\Sigma_{\mathbf{X},U}) \tag{5.2}$$

with mean and covariance of the state from (3.1).

Characteristic functions provide a means to obtain moments, via Definition 2.2, as well as the cumulative distribution function, via Theorem 2.2.

Since we have no direct knowledge of $\mathbf{w}$, the empirical characteristic function can be used to compute the cumulative distribution function and moments from samples of $\mathbf{w}$.

Definition 5.1 (Empirical Characteristic Function [95,97])**.** *Let $\{\mathbf{w}_j\}_{j=1}^{N_s}$ be the sequence of N_s observations of the random vector, $\mathbf{w}$. The empirical characteristic function is*

$$\hat{\varphi}_{\mathbf{w}}(t) = \sum_{j=1}^{N_s} \alpha_j(\mathbf{w}) K_{\mathbf{w}_j}(t) \tag{5.3a}$$

$$K_{\mathbf{w}_j}(t) = \exp\left(it^\intercal \mathbf{w}_j\right) \exp\left(-\tfrac{1}{2}(t^\intercal \Sigma t)\right) \tag{5.3b}$$

for some smoothing parameter matrix $\Sigma \in \mathbb{R}^{p\times p}$ and weighting function $\alpha_j(\mathbf{w}) > 0$, with $\sum_{j=1}^{N_s} \alpha_j(\mathbf{w}) = 1$.

A variety of approaches can be used to find a suitable Σ, to avoid over-smoothing and under-smoothing [101]. The smoothing in (5.3b) is important for ensuring continuity in the cumulative distribution function [26, Eq. 1.2.1] approximated via Theorem 2.2 from the empirical characteristic function.

Hence to solve Problem 1, we first solve the following.

Problem 6. *Using the empirical characteristic function, 1) construct a concave under-approximation of the approximate cumulative distribution function $\hat{\Phi}_{\mathbf{w}}(x)$, and 2) approximate the first two moments of $\overline{\mathbf{w}}$.*

Problem 7. *Reformulate (3) into a convex, conic stochastic optimal control problem, so that feasible solutions of the convex program are feasible solutions of (3).*

5.5 Method

We first transform (2.26a) into a series of individual chance constraints, similar to (3.4), each with a risk δ_i. Similar to the risk allocation derivation in Section 3.5.1, the i^{th} constraint as $p_i^\intercal \mathbf{X} \leq q_i$, where we do not presume the initial condition is random,

$$\begin{aligned} \mathbb{P}_{\mathbf{W}}\left(\left\{W \in \mathbb{R}^{pN} : p_i^\intercal \mathcal{D} W \leq q_i - p_i^\intercal(\mathcal{A}x_0 + \mathcal{B}U)\right\}\right) \geq 1 - \delta_i & \quad (5.4a)\\ \Leftrightarrow \Phi_{p_i^\intercal \mathcal{D}\mathbf{W}}(q_i - p_i^\intercal(\mathcal{A}x_0 + \mathcal{B}U)) \geq 1 - \delta_i & \quad (5.4b)\\ \sum_{i=1}^{L} \delta_i \leq \Delta,\ \delta_i \geq 0,\ \Delta \in [0,1],\ \forall i \in\ \mathbb{N}_{[1,L]}. & \quad (5.4c) \end{aligned}$$

Then solutions of the optimization problem

$$\begin{aligned} \underset{U,\delta}{\text{minimize}} \quad & \mathbb{E}\left[(\mathbf{X} - X_d)^\intercal \mathcal{Q} (\mathbf{X} - X_d)\right] + U^\intercal \mathcal{R} U & (5.5a)\\ \underset{\forall i \in \mathbb{N}_{[1,L]}}{\text{subject to}} \quad & \left\{ \begin{aligned} \Phi_{p_i^\intercal \mathcal{D}\mathbf{W}}(q_i - p_i^\intercal(\mathcal{A}x_0 + \mathcal{B}U)) &\geq\ 1 - \delta_i & (5.5b)\\ q_i - p_i^\intercal(\mathcal{A}x_0 + \mathcal{B}U) &\geq\ x_i^{lb} & (5.5c)\\ \sum_{i=1}^{L} \delta_i \leq \Delta,\ \delta_i \geq 0,\ \Delta \in&\ [0,1] & (5.5d) \end{aligned} \right.\\ & (2.10),\ (5.1b) \end{aligned}$$

are also feasible solutions of (3.2). This is because the joint chance constraint (2.26a) is enforced by (5.5b) and (5.5d) with the additional constraint (5.5c), which restricts the domain of the i^{th} chance constraint by some lower bound x_i^{lb}.

However, several difficulties arise. Note that (5.5) is non-convex due to (5.5b). The constraint (5.5c) ensures a restriction to the concave region of $\Phi_{p_i^\intercal \mathcal{D}\mathbf{W}}(x)$. For unimodal distributions, the inflection point, x_i^{lb}, occurs about the mode [61, Def. 1.1], but for arbitrary distributions, this may not be true.

In addition, (5.5a) is dependent upon the first two moments of $\mathbf{W}$ and (5.5b) is dependent upon the cumulative distribution function of $p_i^\intercal \mathcal{D}\mathbf{W}$, $\forall i \in \mathbb{N}_{[1,l]}$. Hence we seek empirical characteristic functions to approximate the cumulative distribution function and moments based on samples $\mathbf{w}_j$. In addition, we also seek a method to reformulate (5.5b) using its approximation from the empirical characteristic function with a concave restriction (5.5c) by finding x_i^{lb} to solve a convex problem.

5.5.1 Approximating the cumulative distribution function and moments from the empirical characteristic function

Applying Definition 5.1, we obtain

$$\begin{aligned}\hat{\varphi}_{p_i^\intercal \mathcal{D}\mathbf{W}}(t) &= \sum_{j=1}^{N_s} \alpha_j(\mathbf{W}) \exp\left(itp_i^\intercal \mathcal{D}\mathbf{W}_j\right)\cdot \\ &\qquad \exp\left(-\tfrac{1}{2}((p_i^\intercal \mathcal{D})\overline{\Sigma}(p_i^\intercal \mathcal{D})^\intercal t^2)\right) \qquad (5.6a)\\ \hat{\varphi}_{\mathbf{W}}(t) &= \sum_{j=1}^{N_s} \alpha_j(\mathbf{W}) \exp\left(\mathrm{i}t^\intercal \mathbf{W}_j\right) \exp\left(-\tfrac{1}{2}(t^\intercal \overline{\Sigma} t)\right) \qquad (5.6b)\end{aligned}$$

where $\overline{\Sigma} = \mathrm{diag}([\Sigma_0 \cdots \Sigma_N]) \in \mathbb{R}^{pN\times pN}$, $\overline{t} = [t_0 \;\; \cdots t_N]^\intercal \in \mathbb{R}^{pN}$ and $\alpha_j(\mathbf{W}) = 1/N_s$. To approximate $\Phi_{p_i^\intercal \mathcal{D}\mathbf{W}}$ in (5.5b), we use (2.3) to obtain $\hat{\Phi}_{p_i^\intercal \mathcal{D}\mathbf{W}}$. For the moments in the cost (5.5a), we use (2.4) to obtain the approximate moments of $\mathbf{W}$.

5.5.2 Constructing a Convex Restriction for (5.5b)

We seek a conic representation of (5.5b) with a restriction for which it is concave [61, Def 1.1]. For a user-defined error, ϵ, and desired number of affine terms, N_{dr}, we construct a piecewise affine under-approximation [35, Sec. Sec. 4.3.1],

$$\hat{\Phi}^l_{p_i^\intercal \mathcal{D}\mathbf{W}} = \min_{r\in\mathbb{N}_{[1,z^i]}} \{a_{i,r}x + c_{i,r}\} \qquad (5.7)$$

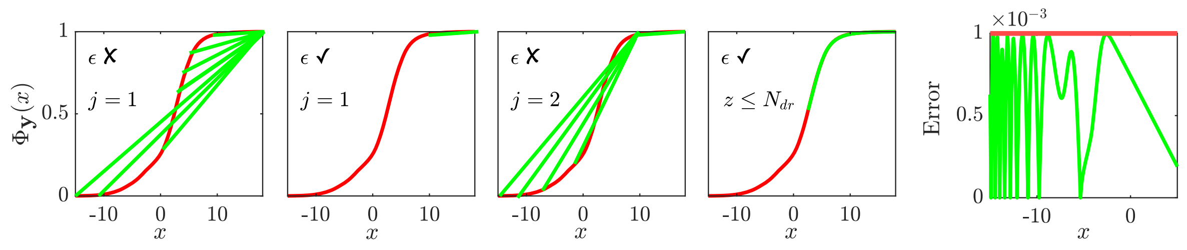


Figure 5.1: (Left to Right) Algorithm 1 under-approximates the cumulative distribution function, $\hat{\Phi}_{\mathbf{y}}(x)$ (red), with $\hat{\Phi}^l_{\mathbf{y}}(x)$ (green), for some user-defined error, ϵ. We use 1000 samples of $\mathbf{y} = \mathbf{f}\mathbf{y}_1 + (1-\mathbf{f})\mathbf{y}_2$, with Bernoulli random variable $\mathbf{f}$, $\mathbf{y}_1$ a Gaussian $\mathcal{N}(0, 0.2)$, and $\mathbf{y}_2$ a Weibull distribution Weib$(k=4, \theta=2)$. The error $\hat{\Phi}_{\mathbf{y}}(x) - \hat{\Phi}^l_{\mathbf{y}}(x) \leq \epsilon$ is depicted on the far right.

such that

$$0 \leq \hat{\Phi}_{p_i^\intercal \mathcal{D}\mathbf{W}}(x) - \hat{\Phi}^l_{p_i^\intercal \mathcal{D}\mathbf{W}}(x) \leq \epsilon \tag{5.8}$$

is assured over the domain $\mathcal{C}_i = [x_i^{lb}, \infty]$. We define $a_{i,r}$ and $c_{i,r}$ as the slope and intercept for the r^{th} affine term.

We propose Algorithm 1 to construct the piecewise linear under-approximation of the cumulative distribution function, with a concave restriction x^{lb}, derived from the empirical characteristic function.

Algorithm 1 Computing $\hat{\Phi}^l_{\mathbf{w}}$ from $\hat{\Phi}_{\mathbf{w}}$

Evaluations of cumulative distribution function $\{(x_p, \hat{\Phi}_{\mathbf{w}}(x_p)\}_{p=1}^{N_p}$, desired error ϵ, desired number of affine terms N_{dr}.
Output: affine terms of $\hat{\Phi}^l_{\mathbf{w}}$, $\{(a_j, c_j)\}_{j=1}^{z}$, restriction x^{lb}

```
1: continue ← true, $p \leftarrow N_p$
2: while continue = true do $\forall j \in \mathbb{N}_{[1,p-1]}$, $\forall k \in \mathbb{N}_{[j,p]}$
3:     $a_j \leftarrow \frac{\hat{\Phi}_{\mathbf{w}}(x_p) - \hat{\Phi}_{\mathbf{w}}(x_j)}{x_p - x_j}$
4:     $c_j \leftarrow \hat{\Phi}_{\mathbf{w}}(x_p) - m_j x_j$
5:     $y_{j,k} \leftarrow a_j x_k + c_j$
6:     $error_{j,k} \leftarrow \hat{\Phi}_{\mathbf{w}}(x_k) - y_{j,k}$
7:     $w \leftarrow$ Smallest $j$ such that $\max_j\{error_{j,k}\} < \epsilon$ and $error_{j,k} > 0$
8:     if $w = \emptyset$ or $z > N_{dr}$ or $||w - p|| = 1$ then
9:         continue ← false
10:        $(a_j, c_j)\}_{j=1}^{z} \leftarrow \{(0, \Phi_{\mathbf{w}}(x_{N_p}))\} \bigcup \mathcal{F}$, $x^{lb} \leftarrow x_j$
11:    else, $\mathcal{F} \leftarrow \{(a_j, c_j)\}_{j=w}$, $p = w$
12:    end if
13: end while
```

Algorithm 1 is based on the sandwich algorithm [65], and is demonstrated in Figure 5.1. At each of N_p evaluation points, $\{(x_p, \hat{\Phi}_{\mathbf{w}}(x_p)\}_{p=1}^{N_p}$, the algorithm constructs affine terms, and stores the affine terms which result in largest positive error close to ϵ. This is repeated until the break conditions are met (line 8) with a total of z piecewise affine terms. We choose an upper bound $\Phi_{\mathbf{w}}(x_{N_p})$ (line 10), as it is unreasonable to infer the probability of an event beyond $\max_{j\in\mathbb{N}_{[1,N_s]}} (p_i^\intercal \mathcal{D}\mathbf{W}_j)$, and it assures (5.8) holds on $\mathcal{C}_i$. This solves Problem 6.

5.5.3 Underapproximative, Conic Optimization Problem

We replace the individual chance constraints in (5.5b) and the lower bounds in (5.5c) with a conic, convex reformulation, obtained from Algorithm 1, resulting in the following.

$$\begin{aligned}
\min_{\overline{u},\delta} \quad & \mathbb{E}\left[(\overline{\mathbf{x}} - \overline{x}_d)^\intercal Q (\overline{\mathbf{x}} - \overline{x}_d) + \overline{u}^\intercal R \overline{u}\right] && \text{(5.9a)}\\
\text{s.t.} \quad & \left\{ a_{i,r}(q_i - p_i^\intercal(\mathcal{A}x_0 + \mathcal{B}U)) + c_{i,r} \geq 1 - \delta_i \right. && \text{(5.9b)}\\
\forall i \in \mathbb{N}_{[1,l]} \quad & q_i - p_i^\intercal(\mathcal{A}x_0 + \mathcal{B}U) \geq x_i^{lb} && \text{(5.9c)}\\
\forall r \in \mathbb{N}_{[1,z^i]} \quad & \sum_{i=1}^{l} \delta_i \leq \Delta, \ \delta_i \geq 0, \ \Delta \in [0,1] && \text{(5.9d)}\\
& \overline{u} \in \mathcal{U}^N && \text{(5.9e)}
\end{aligned}$$

The optimization problem in (5.9) can be posed as a second-order cone program [35, Sec. 4.4]. Algorithm 2 summarizes how the methods described in this section solve (5.9).

5.5.4 Convergence and Confidence Intervals

While (5.9) is convex and conic, its relationship to (3.2) is not clear, as it utilizes an under-approximation of the *approximate* cumulative distribution function, $\hat{\Phi}_{p_i^\intercal \mathcal{D}\mathbf{W}}(x)$ and approximate moments of $\mathbf{W}$. We first establish asymptotic convergence, then construct

Algorithm 2 Underapproximative, conic optimization (5.9)

Time horizon N, T_s, polytopic set $\{P, q\}$, samples $\{\mathbf{w}_j\}_{j=1}^{N_s}$, (5.6a), (5.6b), evaluation points N_p, desired error ϵ, desired number of affine terms N_{dr}, smoothing matrix $\overline{\Sigma}$.
Output: Open loop input $\overline{u}$, risk allocation $\overline{\delta}$

```
1: for $i \in \mathbb{N}_{[1,l]}$ do
2:     Let $\mathcal{C} = [\min_{j\in\mathbb{N}_{[1,N_s]}} (p_i^\intercal \mathcal{D}\mathbf{W}_j), \max_{j\in\mathbb{N}_{[1,N_s]}} (p_i^\intercal \mathcal{D}\mathbf{W}_j)]$
3:     $\{(x_p, \Phi_{p_i^\intercal \mathcal{D}\mathbf{W}}(x_p))\}_{p=1}^{N_p} \leftarrow$ Using (2.3) and (5.6a).
4:     $\{(a_{i,r}, c_{i,r})\}_{r=1}^{z^i}$ and $x_i^{lb} \leftarrow$ From Algorithm 1
5:     Let $\mathcal{C}_i \leftarrow [x_i^{lb}, \infty]$
6: end for
7: $\mathbb{E}[\mathbf{W}]$, $\mathbb{E}[\mathbf{W}^2] \leftarrow$ Using (2.4) and (5.6b).
8: $C_{\overline{w}} \leftarrow \mathbb{E}[\mathbf{W}^2] - (\mathbb{E}[\mathbf{W}])^2$
9: $\{\overline{u}, \overline{\delta}\} \leftarrow$ Solve (5.9).
```

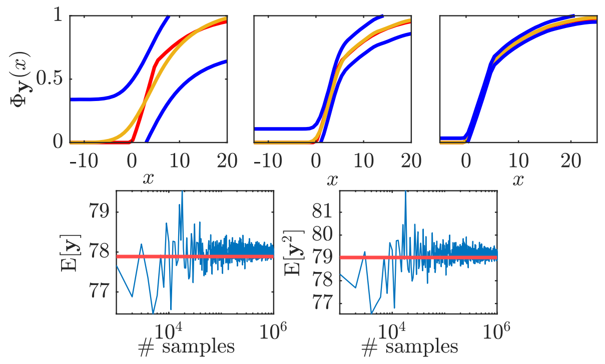


Figure 5.2: (Top) Approximation $\hat{\Phi}_{\mathbf{y}}(x)$ (yellow) of $\Phi_{\mathbf{y}}(x)$ (red) with 80% confidence interval bands (blue) for 10, 100, and 1000 samples. (Bottom) Convergence of $\mathbb{E}[\mathbf{y}]$ and $\mathbb{E}[\mathbf{y}^2]$. We presume $\mathbf{y} = \mathbf{f}\mathbf{y}_1 + (1 - \mathbf{f})\mathbf{y}_2$ for a Bernoulli random variable $\mathbf{f}$, with $\mathbf{y}_1$, $\mathbf{y}_2$, drawn from a gamma distribution Gam$(k = 2, \theta = 5)$, and a uniform distribution Unif$[0, 5]$, respectively.

confidence intervals to describe a relationship to (3.2).

Theorem 5.1. *If $\hat{\varphi}_{\mathbf{W}}(t)$ converges in probability to $\varphi_{\mathbf{W}}(t)$ as $N_s \to \infty$, every feasible solution of (5.9) is feasible for (3.2).*

Proof. By [97, Thm 2.1] $\hat{\varphi}_{\mathbf{W}}(t)$ converges to $\varphi_{\mathbf{W}}(t)$ as $N_s \to \infty$. By the Portmanteau theorem, the cumulative distribution function converges [102, Thm. 2.1]. For $\hat{\varphi}_{\mathbf{W}}(t)$ that is differentiable at zero, then by (5.6b), the moments converge [26, Thm. 2.3.2]. □

Remark 5.1. *The ECF converges at a rate $\sqrt{N_s}$ [95, Sec. 3].*

Asymptotic convergence establishes the relationship between our convex formulation and the original problem, but it is not practical in order to solve the reformulation quickly nor does it guarantee that (5.9b) is an under-approximation. We provide confidence intervals on the cumulative distribution function, a worst-case under-approximation.

Definition 5.2 (Dvoretzky–Kiefer–Wolfowitz Inequality [103])**.** *Given an empirical cumulative distribution function, $\hat{\Phi}^E_{p_i^\intercal \mathcal{D}\mathbf{W}}(x)$, from N_s samples, the probability that the worst deviation is above some ϵ_E is*

$$\mathbb{P}\left\{\sup_{x\in\mathbb{R}}\left(|\hat{\Phi}^E_{p_i^\intercal \mathcal{D}\mathbf{W}}(x) - \Phi_{p_i^\intercal \mathcal{D}\mathbf{W}}(x)| > \epsilon_E\right)\right\} \leq \alpha \tag{5.10}$$

for $\alpha = 2e^{-2N_s\epsilon_E^2}$.

Hence for a desired confidence level α, using N_s samples, we have $\epsilon_E = ((2N_s)^{-1} \ln(2/\alpha))^{1/2}$. To make use of (5.10) for $\hat{\Phi}$, we make the following assumption.

Assumption 5.1. *For $x \in \mathcal{C}_i$, $|\hat{\Phi}^E_{p_i^\intercal \mathcal{D}\mathbf{W}}(x) - \hat{\Phi}_{p_i^\intercal \mathcal{D}\mathbf{W}}(x)| \leq \epsilon_D$.*

Assumption 5.1 is dependent upon $\overline{\Sigma}$ and N_s, and reasonable for Σ chosen to avoid under- or over-smoothing. Both terms converge to $\Phi_{p_i^\intercal \mathcal{D}\mathbf{W}}(x)$ as $N_s \to \infty$, so their difference tends to zero [25, Thm. 20.6].

Theorem 5.2 (Confidence Interval for $\hat{\Phi}_{p_i^\intercal \mathcal{D}\mathbf{W}}(x)$)**.** *Given Def. 5.2 and Assumption 5.1, we have that with probability $1-\alpha$,*

$$|\hat{\Phi}_{p_i^\intercal \mathcal{D}\mathbf{W}}(x) - \Phi_{p_i^\intercal \mathcal{D}\mathbf{W}}(x)| \leq \epsilon_E + \epsilon_D \tag{5.11}$$

Proof. For $x \in \mathcal{C}_i$, by Def. 5.2 and by the least upper bound property [104, Def. 5.5.5], we have that $|\hat{\Phi}^E_{p_i^\intercal \mathcal{D}\mathbf{W}}(x) - \Phi_{p_i^\intercal \mathcal{D}\mathbf{W}}(x)| \leq \epsilon_E$ is satisfied with probability $1-\alpha$. By the properties of absolute value [104, Prop. 4.3.3],

$$\hat{\Phi}^E_{p_i^\intercal \mathcal{D}\mathbf{W}}(x) - \epsilon_E \leq \Phi_{p_i^\intercal \mathcal{D}\mathbf{W}}(x) \leq \hat{\Phi}^E_{p_i^\intercal \mathcal{D}\mathbf{W}}(x) + \epsilon_E \tag{5.12}$$

By Assumption 5.1 and the properties of absolute value,

$$\hat{\Phi}_{p_i^\intercal \mathcal{D}\mathbf{W}}(x) - \epsilon_D \leq \hat{\Phi}^E_{p_i^\intercal \mathcal{D}\mathbf{W}}(x) \leq \hat{\Phi}_{p_i^\intercal \mathcal{D}\mathbf{W}}(x) + \epsilon_D \tag{5.13}$$

Since $\hat{\Phi}_{p_i^\intercal \mathcal{D}\mathbf{W}}(x)$, $\hat{\Phi}^E_{p_i^\intercal \mathcal{D}\mathbf{W}}(x)$, and $\Phi_{p_i^\intercal \mathcal{D}\mathbf{W}}(x)$ are positive, bounded, right-hand continuous functions [25], we combine (5.12) and (5.13), so that $\hat{\Phi}_{p_i^\intercal \mathcal{D}\mathbf{W}}(x) - \epsilon_E - \epsilon_D \leq \Phi_{p_i^\intercal \mathcal{D}\mathbf{W}}(x) \leq \hat{\Phi}_{p_i^\intercal \mathcal{D}\mathbf{W}}(x) + \epsilon_E + \epsilon_D$. Thus, we have (5.11) by the properties of absolute value. □

Corollary 5.1. *Given $\hat{\Phi}^l_{p_i^\intercal \mathcal{D}\mathbf{W}}(x)$, which under-approximates $\hat{\Phi}_{p_i^\intercal \mathcal{D}\mathbf{W}}(x)$ according to (5.8) on $\mathcal{C}_i$, and the confidence interval $\epsilon_D + \epsilon_E$ in (5.11) with likelihood $1-\alpha$, we have $\hat{\Phi}^l_{p_i^\intercal \mathcal{D}\mathbf{W}}(x) - \epsilon - \epsilon_E - \epsilon_D \leq \Phi_{p_i^\intercal \mathcal{D}\mathbf{W}}(x)$ with likelihood $1-\alpha$.*

Proof. Follows directly from (5.8) and (5.11). □

Corollary 5.1 establishes a worst-case under-approximation to the true cumulative distribution function. A similar approach can be taken for $\mathbb{E}[\mathbf{W}]$ and $\mathbb{E}[\mathbf{W}^2]$, using results from [105] and [106], respectively. However, because the approximate moments are cheap to compute (i.e., 3.22 seconds for 10^6 samples), numerical approximations can be quite accurate (Figure 5.2). In contrast, the computational cost of sampling is high for the chance constraint under-approximation.

Algorithm 2 and the optimization reformulation (5.9), along with convergence results and confidence intervals in this section, solve Problem 7.

5.6 Examples

We demonstrate our approach on two examples. We presume $N_s = 1000$, $N_p = 1000$, $\epsilon = 1 \times 10^{-3}$, $N_{dr} = 20$, and $\Delta = 0.2$. In each case, we compare our method to a mixed-integer particle control approach [7], which uses disturbance samples (we chose 50) to compute an open-loop controller. To do so, we used Monte-Carlo simulation with 10^5 disturbance sequences. All computations were done in MATLAB with a 3.80GHz Xeon processor and 32GB of RAM. The optimization problems were formulated in CVX [107] and solved with Gurobi [108]. The inversion (2.3) uses CharFunTool [30] and system formulations are implemented in SReachTools [109]. We use [110], which employs linear diffusion and a plug-in method, to compute $\overline{\Sigma}$.

5.6.1 Double Integrator

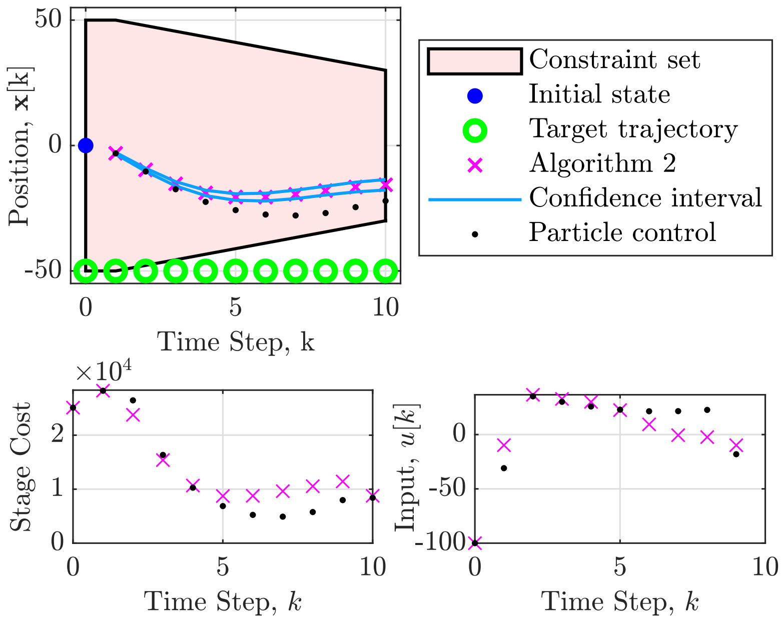


Figure 5.3: (Top) Mean trajectories for the double integrator. Algorithm 2 satisfies the desired constraint satisfaction likelihood, while particle control [7] does not. The reference trajectory is chosen to test constraint violation. (Bottom) Mean stage cost and control input. Algorithm 2 has higher stage cost due to constraint satisfaction.

Consider a double integrator

$$\mathbf{x}_{k+1} = \begin{bmatrix} 1 & T_s \\ 0 & 1 \end{bmatrix} \mathbf{x}_k + \begin{bmatrix} \frac{T_s^2}{2} \\ T_s \end{bmatrix} u_k + \mathbf{w}_k \tag{5.14}$$

with state $\mathbf{x}_k \in \mathbb{R}^2$, disturbance $\mathbf{w}_k \in \mathbb{R}^2$, input $u_k \in \mathcal{U}_k = [-100, 100] \subset \mathbb{R}$, sampling time $T_s = 0.25$, and time horizon $N = 10$. Disturbance samples are drawn independently for each dimension, from a uniform distribution Unif$[-5, 5]$ on $\mathbf{w}_1$, and

Table 5.1: Empirical evaluation of the constraint satisfaction likelihood and mean computation time, based on 10^5 samples.

	Algorithm 2		Particle Control	
Example	$1-\Delta$	Time (s)	$1-\Delta$	Time (s)
Double Integrator	0.912	2.502	0.697	144.6
Hypersonic Vehicle	0.889	5.395	0.639	31.563

from a scaled gamma distribution $0.005 \cdot \text{Gam}(k = 8, \theta = 0.5)$ on $\mathbf{w}_2$. The cost function has $\mathcal{Q}10I_{22\times 22}$, $\mathcal{R} = 10^{-2}I_{10\times 10}$. The time-varying constraint set is $\mathcal{X}_k = \left\{k \in \mathbb{N}_{[0,N]} \times \mathbb{R}^2 : p_1 k + q_1 \leq \mathbf{x}_1 \leq p_2 k + q_2\right\}$ with $p_1 = -p_2 = -2$, $q_1 = -q_2 = -50$. The reference trajectory, $X_d = [50\ 0]^\intercal$, was chosen intentionally to be outside of the constraint set, to test constraint violation.

While the mean state trajectories from Algorithm 2 and from particle control are similar (Figure 5.3), the stage cost, i.e. the cost at each time, and the control trajectories differ. Algorithm 2 exceeds the constraint satisfaction likelihood of 0.8, while particle control falls well below (Table 5.1). This is due to the fact that Algorithm 2 is based on 1000 disturbance samples, while particle control is based on only 50 (from inherent undersampling due to computational cost). The higher cost for Algorithm 2 is incurred because of constraint satisfaction.

5.6.2 One-way Hypersonic Vehicle

Consider a hypersonic vehicle with longitudinal dynamics

$$\begin{aligned}
\dot{h} &= V\sin(\theta-\alpha) \\
\dot{V} &= \tfrac{1}{m}(T(\Psi,\alpha)\cos\alpha - D(\alpha,\delta_e)) - g\sin(\theta-\alpha) \\
\dot{\alpha} &= \tfrac{1}{mV}(-T(\Psi,\alpha)\sin\alpha - L) + Q + \tfrac{g}{V}\cos(\theta-\alpha) \\
\dot{\theta} &= Q \\
\dot{Q} &= M(\alpha,\delta_e,\Psi)/I_{yy}
\end{aligned} \tag{5.15}$$

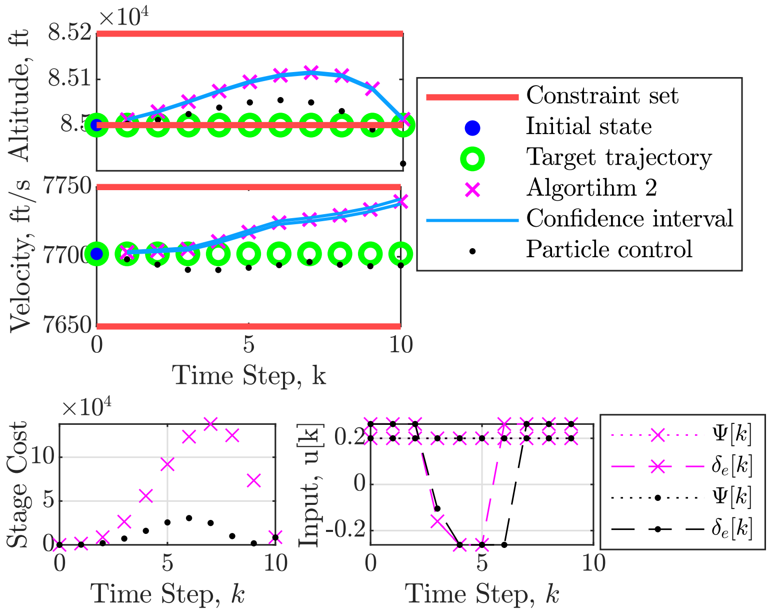


Figure 5.4: (Top) Mean trajectories for the hypersonic vehicle. Constraint satisfaction is above the desired likelihood with Algorithm 2, but not with particle control [7]. (Bottom) Mean stage cost and input. The particle control cost is low because constraints are not satisfied.

with state $\mathbf{x} = [h \ V \ \alpha \ \theta \ Q]^\intercal$ and input $u = [\Psi \ \delta_e]^\intercal$, that includes fuel-to-air ratio Ψ and elevator deflection δ_e [8]. We linearize (5.15) about the trim condition, $x_d = [85000 \text{ ft}, 7702 \text{ ft/s}, 0.026 \text{ rad}, 0.026 \text{ rad}, 0 \text{ rad}]$, which is also the reference trajectory, and $u_d = [0.25, 0.2 \text{ rad}]$, and add a disturbance $\mathbf{w} \in \mathbb{R}^2$, which affects $\dot{h}$ and $\dot{V}$ only, with $\mathbf{w}_1$, $\mathbf{w}_2$ drawn from a scaled Weibull distribution, $2 \cdot \text{Weib}(\text{shape} = 5, \theta = 4)$, and a gamma distribution, $\text{Gam}(\text{shape} = 5, \theta = 1)$, respectively. We discretize in time with $T_s = 0.25$, $N = 10$. The cost function has $\mathcal{Q}10I_{55\times 55}$ and $\mathcal{R} = 10^{-2}I_{20\times 20}$. The constraint set, $\mathcal{X}_k = \{k \in \mathbb{N}_{[0,N]} \times \mathbb{R}^5 : h \in [85000 \text{ ft}, 85200 \text{ ft}], V \in [7650 \text{ ft/s}, 7750 \text{ ft/s}]\}$, and input constraints $\Psi \in [0.2, 1.2]$, $\delta_e \in [-0.26 \text{ rad}, 0.26 \text{ rad}]$ arise from the flight envelope and the operational mode [111–113].

Comparing Algorithm 2 to the particle filter approach, mean trajectories (Figure 5.4) show a similar trend as in Section 5.6.1. While constraints are satisfied under Algorithm 2 with at least the desired likelihood, particle control violates the altitude constraint, and is excessively conservative with respect to the speed constraint. The constraint satisfaction likelihood is 0.889 for Algorithm 2, but only 0.639 for particle control (Table 5.1).

5.7 Conclusion

In conclusion, this chapter presented a novel approach for solving stochastic optimal control problems for linear systems in the presence of unknown uncertainty. By leveraging empirical characteristic functions, we were able to compute both the cumulative distribution function and moments from sampled data, constructing a convex, conic reformulation of the control problem. This approach allows for probabilistic bounds on the solution, offering a flexible, sample-driven method that balances accuracy and computational efficiency. The methods were demonstrated on practical examples, showing that our algorithm provides reliable performance with probabilistic guarantees on constraint satisfaction.

Chapter 6

Distributional Representation of Value Functions for Reinforcement Learning

6.1 Introduction

Reinforcement learning in stochastic settings is limited by what is easy to compute, e.g. the average cost [31, 114]. However, being able to compute other metrics of interest such as Value-at-Risk, Conditional-Value-at-Risk, expectiles, etc. is important to understand the performance of the system through the cost distribution [115, 115, 116]. For example, it would be of interest to understand how large the tails are for the cost distribution, despite the cost distribution having high average performance [117]. In this chapter, we first propose that characteristic functions are an effective representation of the cost distribution as we can construct them from empirical observations. Second, we show that we can derive metrics of the cost distribution, such as Value-at-Risk, Conditional-Value-at-Risk, and expectiles from the cost distribution.

6.2 Related Work

In distributional reinforcement learning, authors utilize performance metrics to improve a controller's performance across the entire distribution [115, 115, 116]. We emphasize also a distributional representation of the cost is not new and the author in [118] explores characteristic functions in the value iteration context. Policy gradient methods are popular within reinforcement learning for their wide applicability and commercial success, of the most popular being proximal policy gradient (PPO) [119]. With such policy gradient methods, many only optimize the control policy over the average cost [114, 119, 120]. In both stochastic optimization and optimal control, metrics other than the average cost are commonplace. For example, in stochastic optimal control, many authors study risk-sensitive stochastic optimal control for the quadratic regulator [121], model predictive control [122–124], and reachability [125]. Works in reinforcement learning also look into risk-sensitivity to improve the controller's, i.e. agent's, ability to handle rare situations [126].

6.3 Main Contribution and Organization

The main contribution of this chapter is a distributional representation of the cost from which we can derive its cost in closed form, but also other performance metrics including, but not limited to, Value-at-Risk, Conditional-Value-at-Risk, and expectiles. We represent the distribution of the cost via characteristic functions, from which we derive closed form expressions for the above performance metrics. We do not explore the entire pipeline to conduct reinforcement learning via policy gradients or value iteration but utilize the derivations as a starting point for future work.

The chapter is organized as follows. Section 6.4 overviews the preliminaries specifically needed for reinforcement learning and where cost arises along with the problem formulation. Section 6.5 covers the derivation of the following performance metrics:

Value-at-Risk, Conditional-Value-at-Risk, and expectiles from a characteristic function representation of the cost. As a proof of concept, Section 6.6 presents the derivation of various performance metrics through a simple, toy examples.

6.4 Reinforcement Learning Preliminaries and Problem Statements

6.4.1 Costs

A stage cost is a mapping from state and/or action to the set of real numbers.

Definition 6.1 (Stage Cost)**.** *A stage cost under stochastic policies is a mapping $g : \mathcal{X} \to \mathbb{R}$,*

$$c_t = g(x_k), \tag{6.1}$$

where $c_t \in \mathbb{R}$ denotes the cost incurred for state at timestep $t \in \mathbb{N}$.

The stage cost can also be a mapping $g : \mathcal{X} \times \mathcal{U} \times \mathcal{W} \to \mathbb{R}$, i.e. it depends on the current action, u_k, and disturbance, $\mathbf{w}$ but we defer adding complexity into the exposition. Nonetheless, with a stage cost, we can formulate a total cost over a time horizon, N, presuming a fixed policy π.

Definition 6.2 (Total Expected Cost)**.** *The total expected cost is the expected value of sum of stage costs, conditioned on the state at initial time, $t = 0$,*

$$J_{0,\pi}(x) = \mathbb{E}\left[\sum_{t=0}^{N-1} \gamma^t g(\mathbf{x}_k) \middle| \mathbf{x}_0 = x\right] \tag{6.2a}$$

$$= \gamma^0 g(x) + \mathbb{E}\left[\sum_{t=0}^{N-1} \gamma^t g(f(\mathbf{x}_k, \pi(\mathbf{x}_k), \mathbf{w})) \middle| \mathbf{x}_0 = x\right], \tag{6.2b}$$

where $\gamma \in (0, 1)$ is a discount factor for each stage and $\mathbf{x}_0 = x$ and, as a result, is a deterministic variable.

We can expand the expected total cost in terms of integrals over the state and action transition kernels,

$$J_{0,\pi}(x) = \int_{\mathcal{X}^{N-1}} \int_{\mathcal{U}^{N-1}} \sum_{t=0}^{N-1} \gamma^t g(x_t) \, \mathrm{d}P_{\mathbf{x}_{N-1}}(x_{N-1}|x_{N-2}, u_{N-2}) \mathrm{d}P_\pi(u_{N-2}|x_{N-2}) \\ \cdots \mathrm{d}P_{\mathbf{x}_1}(x_1|x_0 = x, u_0) \mathrm{d}P_\pi(u_0|x_0 = x). \quad (6.3)$$

The integrals are over state and action and, as a result, amount to $2(N-1)$ integrals. The integral form also elucidates a decomposition known as a value function or cost-to-go.

Definition 6.3 (Cost-To-Go). *A value function is a mapping $J_t : \mathbb{N} \times \mathcal{X} \to \mathbb{R}$,*

$$J_{N-k,\pi}(x) = g(x) + \gamma \mathbb{E}[J_{N-k+1,\pi}(f(s, \pi(x), \mathbf{w}))|s], \; k = \{2, \ldots, N\} \quad (6.4)$$

Another, intuitive way to define the cost-to-go is in terms of a value function, where $V_k(x) = J_{N-k}(x)$, thus

$$V_{k+1,\pi}(x) = g(x) + \gamma \mathbb{E}[V_{k,\pi}(f(s, \pi(s, \mathbf{v}), \mathbf{w}))|s], \; k = \{1, \ldots, N\}. \quad (6.5)$$

Note, $V_{1,\pi}(x) = J_{N-1,\pi}(x) = g(x)$ and $V_{N,\pi}(x) = J_{N-N,\pi}(x) = J_{0,\pi}(x)$. Should we need to extend the time horizon for example by 1, then $V_{k=N+1,\pi} = J_{N-(N+1),\pi} = J_{-1,\pi}$, thereby allowing us to continue the backward recursion indefinitely.

While there are variations in total expected costs, such as infinite and finite horizon costs, we will only focus on infinite, discounted costs. Discounted, infinite horizon total expected costs presume $N = \infty$, which is shorthand for

$$J_\pi(x) = \lim_{N\to\infty} J_{0,\pi}(x), \tag{6.6a}$$

$$= \lim_{N\to\infty} \gamma^0 g(x) + \mathbb{E}\left[\sum_{t=1}^{N-1} \gamma^t g(\mathbf{x}_k) \middle| \mathbf{x}_0 = x\right], \tag{6.6b}$$

$$= \lim_{N\to\infty} g(x) + \gamma\mathbb{E}[J_{1,\pi}(f(s,\pi(s,\mathbf{v}),\mathbf{w}))|s]. \tag{6.6c}$$

We could use $\limsup$ here to avoid cases where the limit does not exist [127, Ch.1]. However, we will just acknowledge when the limit may not exist. For the value function in (6.5), $\lim_{k\to\infty} V_k(x) = J(x),\ \forall s \in \mathcal{X}$. We can make this argument a bit cleaner by introducing the Bellman operator.

Definition 6.4 (Bellman Operator)**.** *Let $\mathcal{R}(\mathcal{X})$ be the set of functions $J : \mathcal{X} \to \mathbb{R}$. Given a policy, π, the Bellman operator is an abstract operator, $\mathcal{N}_\pi : \mathcal{R}(\mathcal{X}) \times \mathcal{X} \to \mathbb{R}$,*

$$\mathcal{N}_\pi(J_\pi)(x) = g(x) + \gamma\mathbb{E}[J_\pi(f(s,\pi(s,\mathbf{v}),\mathbf{w}))|s]. \tag{6.7}$$

With this operator, we will arrive at the value function convergence by the following steps. First, we note that the Bellman operator is monotone. Second, we show that is a contraction mapping. Finally, that this contraction mapping has a unique fixed point via the Banach fixed point theorem.

Lemma 6.1 (Bellman Operator is Monotone)**.** *Suppose we are given a policy π. If $J_{1,\pi},\ J_{2,\pi} \in \mathcal{R}(\mathcal{X})$, and $J_{1,\pi}(x) \le J_{2,\pi}(x),\ \forall s \in \mathcal{X}$, then*

$$\mathcal{N}_\pi(J_{1,\pi})(x) \le \mathcal{N}_\pi(J_{2,\pi})(x),\ \forall s \in \mathcal{X}. \tag{6.8}$$

Definition 6.5 (Contraction Mapping)**.** *Let $(\mathcal{Z}, d)$ be a metric space. A mapping $L : \mathcal{Z} \to \mathcal{Z}$ is a contraction mapping when there exists a constant $a \in [0,1)$ such that for all*

$z_1,\ z_2 \in \mathcal{Z}$, we have

$$d(L(z_1), L(z_2)) \leq d(z_1, z_2). \tag{6.9}$$

Lemma 6.2 (Bellman Operator is a $\gamma-$Contraction)**.** *Suppose we have a policy π and operating on a metric space $(\mathcal{Z}, d_\infty)$. If the Bellman operator, $\mathcal{N}_\pi$, is monotone, then the Bellman operator is a γ-contraction mapping. That is,*

$$||\mathcal{N}_\pi(J_{1,\pi})(x) - \mathcal{N}_\pi(J_{2,\pi})(x)||_\infty \leq \gamma||J_{1,\pi} - J_{2,\pi}||_\infty,\ \forall s \in \mathcal{X}. \tag{6.10}$$

Definition 6.6 (Fixed Point)**.** *Suppose we are given a mapping $L : \mathcal{R} \to \mathcal{Z}$. For a point $z \in \mathcal{Z}$ such that,*

$$L(z) = z, \tag{6.11}$$

is a fixed point of L.

Theorem 6.1 (Banach Fixed Point Theorem)**.** *If $L : \mathcal{Z} \to \mathcal{Z}$ is a contraction mapping on a complete metric space $(\mathcal{Z}, d)$, then there exists a unique $\hat{z} \in \mathcal{Z}$ such that $L(\hat{z}) = \hat{z}$. In addition, with initial point $z_0 \in \mathcal{Z}$ and $z_{k+1} = L(z_k)$ where $k \in \mathbb{N}$, then $\lim_{k\to\infty} z_k = \hat{z}$.*

Proposition 6.1 (Bellman Operator has a Unique Fixed Point)**.** *Suppose $(\mathcal{Z}, d_\infty)$ is a complete metric space and $g(x),\ \forall s \in \mathcal{X}$ is bounded. If the Bellman operator, $\mathcal{N}_\pi$, is a contraction mapping, then the Bellman operator has a unique fixed point*

$$J_\pi(x) = \mathcal{N}_\pi(J_\pi)(x),\ \forall s \in \mathcal{X}. \tag{6.12}$$

Thus, from an initial point $V_0 \in \mathcal{R}(\mathcal{X})$ and $V_{k+1,\pi} = \mathcal{N}(V_{k,\pi})$ where $k \in \mathbb{N}$, $\lim_{k\to\infty} V_{k,\pi} = J_\pi$ (i.e. $\lim_{k\to\infty} V_{k,\pi}(x) = J_\pi(x),\ \forall s \in \mathcal{X}$).

6.4.2 Optimal Costs and Policy

The optimal solution of the stochastic optimal control problem in finite time is the infimum of (6.2),

$$J^*_{0,\pi}(x) = \inf_{\pi} \left(\gamma^0 g(x) + \mathbb{E}\left[\sum_{t=0}^{N-1} \gamma^t g(f(\mathbf{x}_k, \pi(\mathbf{x}_k), \mathbf{w})) \middle| \mathbf{x}_0 = x \right] \right). \tag{6.13}$$

The backward recursion via cost-to-go and value function take similar modifications where the optimal cost-to-go and value function are infimums of the expressions, which we denote by $J^*N - k$ and V^*_k respectively.

Thus for the discounted, infinite time stochastic optimal control problem, the optimal solution is,

$$J^*_{\pi}(x) = \inf_{\pi} \lim_{N\to\infty} \left(\gamma^0 g(x) + \mathbb{E}\left[\sum_{t=0}^{N-1} \gamma^t g(f(\mathbf{x}_k, \pi(\mathbf{x}_k), \mathbf{w})) \middle| \mathbf{x}_0 = x \right] \right). \tag{6.14}$$

We obtain a convergence of the optimal value function similar to the value function, i.e. $J^*(x) = \lim_{k\to\infty} V^*_k(x), \forall s \in \mathcal{X}$. With this, we directly introduce the Bellman optimality operator,

Definition 6.7 (Bellman Optimality Operator)**.** *Let $\mathcal{R}(\mathcal{X})$ be the set of functions $J_\pi : \mathcal{X} \to \mathbb{R}$. Given a policy, π, the Bellman optimality operator is an abstract operator, $\mathcal{N}_{\pi^*} : \mathcal{R}(\mathcal{X}) \times \mathcal{X} \to \mathbb{R}$,*

$$\mathcal{N}_{\pi^*}(J_\pi)(x) = \inf_{\pi} \left(g(x) + \gamma \mathbb{E}[J_\pi(f(s, \pi(s, \mathbf{v}), \mathbf{w}))|s] \right). \tag{6.15}$$

We can show that this operator is also a γ-contraction mapping and has a unique fixed point, i.e. $\lim_{k\to\infty} T_{\pi^*}(V^*_{k,\pi}) = \lim_{k\to\infty} V^*_{k,\pi} = J^*_\pi$.

6.4.3 Markov Decision Process Formalism

A Markov decision process encapsulates the problem data for a stochastic optimal control problem.

Definition 6.8 (Markov Decision Process)**.** *A Markov decision process is a tuple* $(\mathcal{X}, \mathcal{U}, f, g, \gamma)$*.*

Note that the dynamics, f, are included here rather than the state transition kernel. In addition, some Markov decision processes only mention the state transition kernel as opposed to the dynamics as above, yielding the tuple $(\mathcal{X}, \mathcal{U}, P_{\mathbf{x}_{k+1}}, g, \gamma)$ In addition, some Markov decision processes include an initial state distribution, i.e. where the system starts from, as part of the tuple making it $(\mathcal{X}, \mathcal{U}, f, g, \gamma, P_{\mathbf{x}_0})$.

6.4.4 Reinforcement Learning

Reinforcement learning is a study of efficient and scalable algorithms to solve Markov decision processes. There is a large taxonomy of approaches labeled as reinforcement learning such as value iteration and policy gradients [31, 114]. Value iteration follows the exposition in Section 6.4.1 with variations should a controller/policy be given [118]. However, policy gradients requires additional steps. First make the following assumptions of the policy function.

Definition 6.9 (Parameterized Stochastic Policy Function)**.** *A parametrized policy function is a function* π_θ *where* $\theta \in \mathbb{R}^p$ *modify the behavior of the mapping* $\pi_\theta : \mathcal{X} \times \mathcal{V} \to \mathcal{U}$*.*

For simplicity, we will derive everything in terms of the discounted finite horizon case, which will also hold for the discounted, infinite horizon case. Policy gradient solves the optimization problems in (6.2), starting from some $s \in \mathcal{X}$ by gradient descent,

$$\theta_{k+1} = \theta_k + \nabla_{\theta_k} J_{0,\pi_{\theta_k}}(x), \ k \in \mathbb{N}. \tag{6.16}$$

To make the gradient descent step computationally practical, i.e. to not take the gradient of J_0, the following theorem uses the log-probability trick.

Theorem 6.2 (Policy Gradient Theorem). *Given a parameterized stochastic policy function π_θ and a cost function, J_{0,π_θ}, the gradient of the cost in (6.16) is reformulated as,*

$$\nabla_\theta J_{0,\pi_\theta}(x) = \mathbb{E}\left[C(x)\nabla_\theta\left(\sum_{t=0}^{N-1}\log(P_{\pi_\theta}(A|\mathbf{x}_k))\right)\middle|\mathbf{x}_0 = s\right],\ \forall A \in \mathcal{M}(\mathcal{U}) \tag{6.17a}$$

$$= \mathbb{E}\left[\sum_{t=0}^{N-1}\nabla_\theta\log(P_{\pi_\theta}(A|\mathbf{x}_k))C(\mathbf{x}_k)\middle|\mathbf{x}_0 = s\right] \tag{6.17b}$$

where

$$C(x_k) = \sum_{\tau=0}^{N-1}\gamma^\tau g(x_{t+\tau}), \tag{6.18}$$

is the finite horizon, discounted cost as before but we evaluate from timestep t onward.

Note that there are a number of functions $C : \mathcal{X} \to \mathbb{R}$ which improve the total cost minimization. This is due to the fact that C is evaluated through repeated simulations of the system with a policy function, making it unstable. Such functions include the Q-function,

$$Q_{\pi_\theta}(x_k, u_k) = \mathbb{E}\left[\sum_{\tau=0}^{N-1}\gamma^\tau g(\mathbf{x}_{t+\tau})\middle|\mathbf{x}_t = x_k, \mathbf{a}_t = u_k\right], \tag{6.19}$$

as well as the advantage function,

$$A_{\pi_\theta}(x_k, u_k) = Q_{\pi_\theta}(x_k, u_k) - J_{0,\pi}(x_k). \tag{6.20}$$

To evaluate such functions, an estimate of the total cost. We can use either empirical sampling, $\hat{J}_{0,\pi}$, or an approximation, $\hat{J}_{0,\pi,\phi}$ both based on the current policy iteration. The approximation uses empirical samples to obtain an estimate that could potentially have better properties, such as smoothness. This approximation is typically done with a neural network in the literature, with a mean squared error,

$$\phi_k \in \arg\inf_{\phi} \mathbb{E}\left[(\hat{J}_{0,\pi}(x) - \hat{J}_{0,\pi,\phi}(x))^2\right], \tag{6.21}$$

where ϕ_k are the updated parameters for the approximation of the total cost $\hat{J}_{0,\pi,\phi}$.

Thus, at a high-level, the steps to find an optimal policy in the reinforcement learning context are:

1. Evaluate the parameterized stochastic policy function (random weights at $k = 0$) and compute samples of the state to approximate $\hat{J}_{0,\pi,\phi_k}$.

2. Iterate the policy gradient step in (6.16) to obtain, $\pi_{\theta k+1}$.

3. Iterate k and repeat from step 1 until convergence criteria is met.

There are variations on the steps above which seek to improve optimality. For example, ensuring monotonic improvement in the policy optimization [120]. Others utilize clipping to stabilize the optimization process [119].

6.4.5 Problem Statement

Problem 8. *Given a Markov Decision Process $(\mathcal{X}, \mathcal{U}, P_{\mathbf{x}_{k+1}}, g, \gamma)$ where the stationary state transition kernel, $P_{\mathbf{x}_{k+1}}$, does not have a closed form or is intractable to compute, and a cost objective,*

$$J^{\rho}_{0,\pi}(x) = \rho\left(\sum_{k=0}^{N-1} \gamma^k g(\mathbf{x}_t); \mathbf{x}_0 = x\right), \tag{6.22}$$

where $\rho : \mathcal{C} \times \mathcal{X} \to \mathbb{R}$ is a performance metric and $\mathcal{C}$ is the space of random variables, $\mathbf{c}$, find closed form expressions for performance metrics such as: i) Expectation, ii) Value-at-Risk/quantiles [128], iii) Conditional-Value-at-risk [126], or iv) expectiles [129].

6.5 Method

We utilize characteristic functions to represent the underlying distribution of the total cost within the function ρ in (6.22),

$$\mathbf{c}(x) = g(x) + \tilde{\mathbf{c}}, \tag{6.23a}$$

$$\tilde{\mathbf{c}} = \sum_{k=1}^{N-1} \gamma^k g(\mathbf{x}_k). \tag{6.23b}$$

The total cost has a direct representation when we use characteristic functions,

$$\varphi_{\mathbf{c}}(t) = \exp(itg(x))\varphi_{\tilde{\mathbf{c}}}(t) \tag{6.24}$$

where the complex exponential represents the first part of (6.23a) and $\varphi_{\tilde{c}}$ represents the random variable in (6.23b).

From this, deriving expectation is straightforward, as it is merely the first derivative of the characteristic function evaluated at zero as in Definition 2.2.

$$\mathbb{E}[\mathbf{c}] = \frac{\mathrm{d}\varphi_{\mathbf{c}}}{\mathrm{d}t}(0) \tag{6.25}$$

Value-at-Risk is simply a re-framing of the quantile function, in terms of a "risk" parameter, Δ.

Definition 6.10. *The quantile function $Q : [0,1] \to \mathbb{R}$ associated with a cumulative distribution function is the inverse of the cumulative distribution function, provided that $\Phi_{gC(f(\mathbf{x}))}(x)$ is continuous and non-decreasing. Formally, for $p \in [0,1]$, the quantile function is given by*

$$Q(\Delta) = \inf\{c \in \mathbb{R} : \Phi_{\mathbf{c}}(c) \geq \Delta\}. \tag{6.26}$$

In other words, $Q(\Delta)$ returns the value c such that the probability of a random variable being less than or equal to c is at least Δ.

Note that the quantile function requires the cumulative distribution function. We can compute the cumulative distribution function directly from the characteristic function through Theorem 2.2. Thus, both performance metrics are attainable with known properties of the characteristic functions.

Conditional-Value-at-Risk is a measure of tail behavior beyond Value-at-Risk [130].

Definition 6.11. *Suppose* $\mathbf{c}$ *is a random variable which has a mean. The Conditional-Value-at-Risk of the random variable,* $\mathbf{c}$*, is*

$$\mathrm{CVaR}_{\Delta}(\mathbf{c}) = \inf_{c\in\mathbb{R}} \left\{ c + \frac{1}{\Delta}\mathbb{E}[\max(0, \mathbf{c} - c)] \right\} \tag{6.27}$$

We derive the following theorem that enables the derivation of the metric directly from the characteristic function.

Theorem 6.3. *Given the cost as in* (6.23a)*, then the Conditional-Value-at-Risk via the cost characteristic function is,*

$$\mathrm{CVaR}_{\Delta}(\mathbf{c}) = \inf_{c\in\mathbb{R}} \left\{ c + \frac{1}{\Delta}\frac{\mathrm{d}\varphi_{\max(0,\mathbf{c}-c)}(0;c)}{\mathrm{d}t} \right\}, \tag{6.28a}$$

$$\varphi_{\max(0,\mathbf{c}-c);c}(t) = \frac{1}{2}[1 + \varphi_{\mathbf{z}}(t)] + \frac{\mathrm{i}}{2}[\mathcal{H}(\varphi_{\mathbf{z}})(t) - \mathcal{H}(\varphi_{\mathbf{z}})(0)], \tag{6.28b}$$

$$\varphi_{\mathbf{z}}(t) = \varphi_{\mathbf{c}}(t)\exp(-\mathrm{i}tc), \tag{6.28c}$$

where we evaluate the derivative of the characteristic function at zero while we can still evaluate the variable c *in* (6.28a)*.*

Proof. The characteristic function in (6.28c) arises from the properties of the characteristic function in Section 2.2.1. This characteristic function is then fed into the characteristic function for a max function in (6.28b) [131]. From Definition (2.2), we can substitute for the expectation in (6.27) with the derivative of the characteristic function as in Definition 2.2 evaluated at zero. □

Theorem 6.3 utilizes the average, (2.4), from the derivative of the characteristic function in Definition 2.2 as well as the derivation of the characteristic function of the positive part of a random variable [131].

Lastly, we investigate the expectile, which is a re-framing of the mean similar to how the quantile in Definition 6.10 is a parameterized representation of the median [132].

Definition 6.12. *Suppose* $\mathbf{c}$ *is a random variable which as a mean. The expectile is a value* $c \in \mathbb{R}$ *found by determining when, for* $\Delta \in (0,1)$*,*

$$\Delta \mathbb{E}[\max(0, \mathbf{c} - c)] = (1 - \Delta)\mathbb{E}[\max(0, c - \mathbf{c})], \tag{6.29}$$

where the expectations are in terms of the random variable $\mathbf{c}$*.*

Note that finding c such that (6.29) holds is a root finding problem [133]. As with the Conditional-Value-at-Risk, we can compute the expectile from the characteristic function. This involves us computing the expectations in (6.29).

Theorem 6.4. *Given the cost as in (6.23a), then the expectations in finding the expectile in (6.29) are,*

$$\mathbb{E}[\max(0, c - \mathbf{c})] = \frac{\mathrm{d}\varphi_{\max(0,\mathbf{c}-c)}(0; c)}{\mathrm{d}t} \tag{6.30a}$$

$$\mathbb{E}[\max(0, \mathbf{c} - c)] = \frac{\mathrm{d}\varphi_{\max(0,c-\mathbf{c})}(0; c)}{\mathrm{d}t} \tag{6.30b}$$

we evaluate the derivative of the characteristic function at zero while it is parameterized by c *in (6.29). Both (6.30a) and (6.30b) are*

Proof. Follows similarly to Theorem 6.3. □

Thus, with Theorems 6.3 and 6.4 show that a single distributional representation is sufficient to derive various distributional metrics without needing to empiricially calculate them separately.

6.5.1 Computational and Representation Considerations

In most approaches to reinforcement learning, the total cost is approximatied via function approximation [31, 114]. As a result, poor approximation of the characteristic function will result in poor representations of the performance metrics. For example, since the average of the total cost in (6.23a) is the derivative of the characteristic function of the total cost evaluated at zero, i.e. (6.25), an approximate characteristic function can potentially have a poor approximation of the derivative. For Value-at-Risk, Conditional-Value-at-Risk, and expectiles, the approximation of the characteristic function, which informs the quality of the integral transforms involved will dictate how accurate the metric is relative to the true value.

Another concern is the representation of the performance metrics. For risk metrics such as conditional-value-at-risk, risk violations as a Markov decision process evolves over time are not accounted for [134]. This distinguishes approaches that consider risk over time, preventing paradoxes [135, 136], versus static assessments of risk, as we present here. Here, we only consider the static assessment of the performance metrics for sake of addressing the problem statement. Static assessments of risk nonetheless have been used successfully on benchmarks in distributional reinforcement learning despite its limitations [116, 129].

6.6 Example

We empirically demonstrate the calculation of various performance metrics via characteristic functions through a simple toy example. We presume a scalar linear system with state $\mathbf{x} \in \mathbb{R}$

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{w}_k, \tag{6.31}$$

Table 6.1: The proposed characteristic function approach closely reflects the empirical result. While this example is simple, the proof of concept shows promise for application to reinforcement learning.

Performance Metric	Proposed	Empirical
Value-at-Risk	8.35	8.35
Conditional-Value-at-Risk	7.56	7.42
expectile	6.09	6.43

with Gaussian additive noise, $\mathbf{w}_k \sim \mathcal{N}(0,1)$ and the initial state be deterministic, $\mathbf{x}_0 = 1$. Let the cost function, g, be a linear function,

$$g(x) = x. \tag{6.32}$$

Note that by the properties of the standard Gaussian, the sum of Gaussians increases the variance by one as the time horizon grows, $k \in \mathbb{N}$, but the average will remain the same,

$$\mu_{\mathbf{x}_{k+1}} = \mu_{\mathbf{x}_k} + \mu_{\mathbf{w}_k} \tag{6.33a}$$

$$\sigma^2_{\mathbf{x}_{k+1}} = \sigma^2_{\mathbf{x}_k} + \sigma^2_{\mathbf{w}_k}. \tag{6.33b}$$

We conduct all experiments on a Macbook Pro, with an M3 Pro CPU and 16GB of RAM in MATLAB. Computations with the characteristic functions are done using the CharFun toolbox [30]. Table 6.1 presents a comparison of the characteristic function approach and the empirical computation via 1,000,000 samples. With a time horizon of $N = 10$, the characteristic function derived results are near the empirically computed quantity, but not exact for both Conditional Value-at-Risk and expectiles. This is likely due to the fact that we have to employ numerical quadrature for the Hilbert transforms in (6.28b) and take a numerical derivative atop this quadrature.

6.7 Conclusion

This chapter has presented a novel starting point for reinforcement learning to be performance metric agnostic by leveraging the distributional representation of value functions. Specifically, we introduced computational and representation techniques that account for the full distribution of returns, rather than just the expected value via the characteristic function. As part of future work, we will focus on the dynamic enforcement of risk over time as well as efficient computation via the characteristic function. Additionally, we plan to explore a characteristic function variant of the policy gradient method to provide a performance metric agnostic means of improving the controller. Finally, we aim to validate the proposed approach by applying it to standard reinforcement learning benchmarks, ensuring its scalability and robustness in real-world applications [137]. These future directions will further strengthen the integration of distributional representations of the cost into reinforcement learning, enriching algorithms which already exist and spawn exploration for new algorithms that improve performance.

Part III

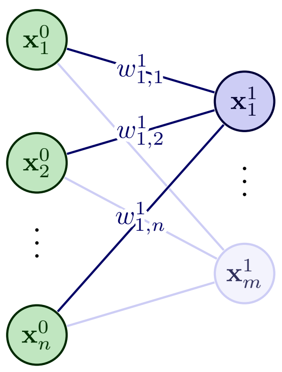
Probabilistic Verification of Neural Networks

Chapter 7

Analytic Distribution Propagation Through ReLUs

7.1 Introduction

Neural networks have become a powerful tool in recent years for a large class of applications, including image classification [138], speech recognition [139], autonomous driving [140], drone acrobatics [141], and many others. The formal verification of neural networks is crucial for their wider adoption in safety-critical scenarios. The main difficulty with the use of (deep) neural network for safety-critical applications lies in the demonstrated sensitivity of deep neural networks to input uncertainties and/or adversarial attacks. For example, in the context of image classification, adding even a small amount of noise to the input set can greatly change the network output [142, 143]. For safety-critical applications, deep neural networks should be robust or insensitive to input uncertainties, a property that be tested by verifying that the network prescribes to certain output specifications subject to various inputs.



$$
\begin{aligned}
\mathbf{x}^1 &= \left[\mathbf{x}_1^1 = \mathrm{ReLU}\left(\mathbf{y}_1^1\right) \quad \cdots \quad \mathbf{x}_m^1 = \mathrm{ReLU}\left(\mathbf{y}_m^1\right)\right]^\intercal \\
\mathbf{y}^1 &= W^1\mathbf{x}^0 + b^1 \\
&\Updownarrow \\
\varphi_{\mathbf{x}_1^1}(t) &= \frac{1}{2}(1 + \varphi_{\mathbf{y}_1^1}(t)) + \frac{\mathrm{i}}{2}\left[\mathcal{H}(\varphi_{\mathbf{y}_1^1})(t) - \mathcal{H}(\varphi_{\mathbf{y}_1^1})(0)\right] \\
&\vdots \\
\varphi_{\mathbf{x}_m^1}(t) &= \frac{1}{2}(1 + \varphi_{\mathbf{y}_m^1}(t)) + \frac{\mathrm{i}}{2}\left[\mathcal{H}(\varphi_{\mathbf{y}_m^1})(t) - \mathcal{H}(\varphi_{\mathbf{y}_m^1})(0)\right] \\
\varphi_{\mathbf{y}^1}(t) &= \exp(\mathrm{i}t^\intercal b^1)\varphi_{\mathbf{x}^0}((W^1)^\intercal t)
\end{aligned}
$$

Figure 7.1: The characteristic function of the input data can be propagated through a ReLU network analytically. This enables one to query the characteristic function of the network to answer out-of-distribution questions at the output. The use of characteristic functions also circumvents difficulties in cases where the underlying distributions do not have any moments or moment-generating functions (e.g., Cauchy distribution).

7.2 Related Work

Verification frameworks for deep neural networks can be classified as either *deterministic* or *probabilistic*. In exact verification, a deterministic input set is mapped to an output set; if any output falls outside the safety set, the verification fails. This is referred to as *worst-case* safety verification since the input set can be treated as an uncertainty set centered around some nominal input. Given some input x_0 and a neural network $f : x \mapsto y$, deterministic verification can be posed as a nonlinear program (NLP), with the objective function quantifying satisfaction of some safety rule $y \in \mathcal{S}$. In general, though, the resulting NLP is intractable using standard off-the-shelf solvers. Several works have used mixed-integer linear programming (MILP) [144,145], Satisfiability Modulo Theories (SMT) [146,147], or semi-definite programming (SDP) [148–153], to recast and solve this NLP problem. In recent work, given an input or an output polytope, one can generate the respective output or input polytope through the ReLU neural network [154].

In probabilistic verification, the input set itself is uncertain and potentially unbounded. Random uncertainties naturally arise in practical applications, for example, from signal processing, environmental noise, and other exogenous disturbances. For example, impulse noise from Cauchy distributions arise in varioius sensing and imaging domains [155, 156]. In this context, the uncertainties are modeled in terms of probability distributions, and the verification problem is to find the *probability* that the output is contained in a safety set given a random input from the input set. Given a random input vector $\mathbf{x}_0$ and a neural network f, the probability that the output random vector $\mathbf{y} = f(\mathbf{x}_0)$ lies in some safety set $\mathcal{X}$ is greater than some threshold $1 - \Delta$ is given by the chance constraint

$$\mathbb{P}_{\mathbf{y}}(\mathcal{X}) \geq 1 - \Delta. \tag{7.1}$$

Relatively few works have studied the verification of deep neural networks in a probabilistic setting; most of the existing approaches involve under- or over-approximations. In [157], an output confidence ellipsoid is estimated via an SDP that is an affine and quadratic relaxation, and then equivalence between confidence sets and chance constraints is used to solve the verification problem. PROVEN [158] accommodates bounded disturbances, using linear approximations of activation functions and concentration inequalities to generate bounds on (7.1). In [159], a similar approach is taken with Cramer-Chernoff concentration inequalities, but because it is based on sampling, a linear approximation of the activation functions is not needed. Generative deep neural networks are considered in [160, 161], which formulates an upper bound on the chance constraint via duality. Lastly, a scenario optimization approach in [162] constructs a lower bound on (7.1) that depends upon the number of samples.

7.3 Main Contribution and Organization

In this chapter, the main contribution is the interpretion of a deep neural network as a dynamical system [163–165] that shapes distributions of data and view the verification problem as one of propagating a distribution through a linear stochastic system to form an output distribution that needs to meet the safety constraints. We focus on deep neural networks with rectified linear units (ReLU) activation functions, as the piece-wise linear nonlinearity of ReLU have *known* integral operators which allow us to propagate a given input distribution. Specifically, as we denote in Figure 7.1, given an input distribution's characteristic function , we can recover the characteristic function of the output of a ReLU deep neural network with, known error accuracy, from which we can verify the output chance constraint (7.1). Therefore, we can provide rigorous statistical guarantees for the performance of any given ReLU neural network for any input distribution.

The chapter is organized as follows. Section 7.4 introduces the preliminaries and problem formulation. Section 7.5 presents the main properties of characteristic functions we use in our work and states the main result that allows us to propagate a characteristic function through a ReLU neural network. Section 7.5 Section 7.7 presents the safety verification algorithm given the machinery developed in the previous section applied to output polytopes. Examples demonstrating the theory are given in Section 7.8, and we provide some concluding remarks and avenues for future work in Section 7.9.

7.4 Preliminaries and Problem Statement

We consider an L-layer ReLU deep neural network with input $\mathbf{x}^0 \in \mathbb{R}^{h_0}$ and output $\mathbf{y} = f(\mathbf{x}^0) = \mathbf{x}^l \in \mathbb{R}^{h_L}$, with f being the composition of L layers, that is, $f = f_{L-1} \circ \cdots \circ f_0$. The kth layer of the ReLU network corresponds to a function $f_k : \mathbb{R}^{h_k} \to \mathbb{R}^{h_{k+1}}$ of the form

$$\mathbf{x}^{k+1} = f_k(\mathbf{x}^k) = \sigma(W^k \mathbf{x}^k + b^k), \tag{7.2}$$

where $W^k \in \mathbb{R}^{h_{k+1} \times h_k}$ is the weight matrix, $b^k \in \mathbb{R}^{h_{k+1}}$ is the bias, and $\sigma(x_j^k) := \max(0, x_j^k)$ is the component-wise ReLU function, where x_j^k is the jth component of $x^k \in \mathbb{R}^{h_k}$. We assume that the last layer is an affine transformation, that is, $\mathbf{x}^L = W^{L-1}\mathbf{x}^{L-1} + b^{L-1}$. Note that convolution layers can be captured by this framework, as they correspond to linear layers W^k endowed with a particular matrix structure.

Let the mapping $f : \mathcal{X} \mapsto \mathcal{Y}$ with $\mathcal{X}$ and $\mathcal{Y}$ subsets of Euclidean spaces of given dimensions, and let $\mathcal{S} \subset \mathcal{Y}$ denote the output safety set. We would like to answer the following problems:

Problem 9. *Given a random sample from the input set $x = \mathbf{x}(\omega) \in \mathcal{X}$, where $\omega \in \Omega$, what is the probability that the output $y = f(x) \in \mathcal{Y}$ lies in the output set $\mathcal{S}$? Equivalently, given some verification threshold $p \in (0, 1]$, is the chance constraint (7.1) satisfied for all $x \in \mathcal{X}$?*

Problem 10. *Given the numerically computed output distribution $\hat{\psi}_{\mathbf{y}}$, what is the relative error in the probability of satisfaction of the output chance constraint compared to that of the true output distribution $\psi_{\mathbf{y}}$?*

To solve the above problems, we use the machinery of characteristic functions to propagate a distribution through a ReLU network allowing us to perform the verification task.

7.5 Propagation of a Characteristic Function through a ReLU Network

Given an initial characteristic function φ^0 that represents the input distribution, we compute the output characteristic function φ^L. At an arbitrary layer k this propagation can be split into a two-step process: (i) propagate the characteristic function through the affine layer to obtain $\varphi_{\mathbf{y}^k}$, where $\mathbf{y}^k = W^k\mathbf{x}^k + b^k$, and (ii) propagate the intermediate characteristic function through the ReLU layer to obtain the output $\varphi_{\mathbf{x}^{k+1}}$. Using Property P5 of characteristic functions, it is straightforward to compute

$$\varphi_{\mathbf{y}^k}(t) = \exp(\mathrm{i}t^\intercal b^k)\varphi_{\mathbf{x}^k}((W^k)^\intercal t). \tag{7.3}$$

Given the intermediate characteristic function $\varphi_{\mathbf{y}^k}$, we can compute the component-wise characteristic function after the ReLU, based on the work of [166], which is summarized below.

Corollary 7.1. *The characteristic function of the random variable* $\mathbf{x}_+ := \max(0, \mathbf{x})$ *is given by*

$$\varphi_{\mathbf{x}_+}(t) := \mathbb{E}[e^{\mathrm{i}t\mathbf{x}_+}] = \frac{1}{2}\left[1 + \varphi_{\mathbf{x}}(t)\right] + \frac{\mathrm{i}}{2}\left[\mathcal{H}(\varphi_{\mathbf{x}})(t) - \mathcal{H}(\varphi_{\mathbf{x}})(0)\right]. \tag{7.4}$$

Let $\mathbf{x} \in \mathbb{R}$ be a scalar random variable, and introduce the operator

$$J_a(\varphi)(t) := \frac{1}{2\pi\mathrm{i}} \int_{\mathbb{R}} e^{-\mathrm{i}a\eta} \varphi(t+\eta)\, \frac{\mathrm{d}\eta}{\eta}. \tag{7.5}$$

Using the change of variables $u \mapsto -u$, one may equivalently write (7.5) as

$$J_a(\varphi)(t) = \frac{1}{4\pi\mathrm{i}} \int_{\mathbb{R}} [e^{-\mathrm{i}a\eta}\varphi(t+\eta) - e^{\mathrm{i}a\eta}\varphi(t-\eta)]\, \frac{\mathrm{d}\eta}{\eta}. \tag{7.6}$$

Proposition 7.1. *Let* $\mathbf{x} \in \mathbb{R}$ *be a real-valued random variable with characteristic function* $\varphi_{\mathbf{x}}$*. Then,*

$$J_a(\varphi_{\mathbf{x}})(t) = \frac{1}{2}\mathbb{E}[e^{\mathrm{i}t\mathbf{f}x} \operatorname{sgn}(\mathbf{x} - a)]. \tag{7.7}$$

Proof. The proof is straightforward using the definition of J_a in (7.5) and Fubini's theorem. See [166] for details. □

Next, consider the ReLU operator $\mathrm{ReLU}(x) = \max(0, x)$. Using the identity

$$2e^{\mathrm{i}t\max(0,x)} = 1 + e^{\mathrm{i}tx} + e^{\mathrm{i}tx}\operatorname{sgn}(x) - \operatorname{sgn}(x), \tag{7.8}$$

we can derive the CF of the ReLU operator [166, Equation 8]. From the identity in (7.8), take the expectation of both sides, which yields

$$2\mathbb{E}[e^{\mathrm{i}t\mathbf{x}_+}] = 1 + \mathbb{E}[e^{\mathrm{i}t\mathbf{x}}] + \mathbb{E}[e^{\mathrm{i}t\mathbf{x}}\operatorname{sgn}(\mathbf{x})] - \mathbb{E}[\operatorname{sgn}(\mathbf{x})]. \tag{7.9}$$

Using (7.7) from Proposition 1 with $a = 0$ we get the desired result. Comparing the alternative definition of J_a in (7.6) with the definition of the HT, we can identify $J_0 = \frac{\mathrm{i}}{2}\mathcal{H}$, which implies that

$$\varphi_{\mathbf{x}_j^{k+1}}(t_j) = \frac{1}{2}(1 + \varphi_{\mathbf{y}_j^k}(t_j)) + \frac{\mathrm{i}}{2}\left[\mathcal{H}(\varphi_{\mathbf{y}_j^k})(t_j) - \mathcal{H}(\varphi_{\mathbf{y}_j^k})(0)\right], \tag{7.10}$$

where $t_j = e_{j,h_{k+1}}^\intercal t$ isolates the jth element of the frequency variable t. Applying the characteristic function update (7.4) to each neuron j for each layer k results in the update (7.10).

7.6 Complexity of Propagation

Given the machinery of how to propagate characteristic functions through a ReLU network via (7.3) and (7.10), we would like to know how many computations are actually being done per layer. Exact neural network verifiers face an exponential run-time barrier due to the coupling between all neurons in the spacial domain [148]. Inexact verifiers still have to solve an associated convex program, which is bottlenecked by the solver speed and tolerances. The accuracy of our method, on the other hand, is solely due to the refinement of the frequency grid for the characteristic function evaluations and the parameters used to numerically compute the HT, that is, no optimization is needed for verification.

7.6.1 Frequency Domain Gridding

In order to numerically propagate the characteristic function through the ReLU network, one needs to properly setup the bounds for computing the characteristic function. Since the characteristic function is defined for all points $t \in \mathbb{R}$, we need to setup a grid $\{t_m\}_{m=1}^N$ with some cutoffs $-\infty < d_- < d_+ < \infty$ and evaluate the characteristic function at these grid points. As we shall see, our method is only linearly complex in the total number of grid points used to compute the characteristic function. Given that the characteristic function reduces down to zero at its tails, we can heuristically find the cutoff points with a convergence-type condition of the form

$$d_- := \mathrm{argmax}_t |\varphi(t) - \varphi(t-\epsilon)| \le \epsilon, \tag{7.11a}$$

$$d_+ := \mathrm{argmin}_t |\varphi(t+\epsilon) - \varphi(t)| \le \epsilon. \tag{7.11b}$$

7.6.2 Affine Layer Propagation

To analyze the complexity of computations in our frequency domain-based formalism, we can break up the computations between the affine and max layers. The propagation of the joint characteristic function is given in (7.3), however in practice, we propagate each component $\varphi_k^{(j)}$ individually, then parallelize over each marginal characteristic function. In the spacial domain, breaking up the affine layer propagation into components yields

$$\mathbf{y}_j^k = \sum_{\ell=1}^{h_k} W_{j,\ell}^k \mathbf{x}_\ell^k + b_\ell^k, \quad j \in \{1, \dots, h_{k+1}\}. \tag{7.12}$$

Since this is just an affine transformation of the random variables $\mathbf{x}_\ell^k$, we can use properties P4 and P5 of characteristic functions to get

$$\varphi_{\mathbf{y}_\ell^k}(t_m) = \exp(\mathrm{i} t_m b_\ell^k) \prod_{\ell=1}^{h_k} \varphi_{\mathbf{x}_\ell^k}(W_{j,\ell}^k t_m), \tag{7.13}$$

where $t_m \in [d_-, d_+]$ is a grid point in the frequency domain. From (7.13), there are $(h_k + 1)$ terms in the product for each grid point and each component, which results in a complexity of $\mathcal{O}(h_k h_{k+1} N)$. Since the number of grid points $N \gg h_k$ for all layers k, this essentially becomes $\mathcal{O}(N)$, which is *linear* in the resolution of the grid. As a result, this implies we can construct a very fine grid - hence capturing the data very well - without major losses in computational speed.

7.6.3 Max Layer Propagation

The propagation of the characteristic function through the max layer requires the computation of two Hilbert transforms, as per (7.10), for each grid point and neuron. The discrete HT requires $2M+1$ terms in the sum, which implies a complexity of $\mathcal{O}(h_{k+1}MN) \sim \mathcal{O}(MN)$ across all neurons for one layer [167].

Consider the set $\mathcal{C}_{(d_-,d_+)} := \{z \in \mathbb{C} : \mathrm{Im}(z) \in (d_-, d_+)\}$, for some $-\infty < d_- < 0$ and $0 < d_+ < \infty$. A function f is in $H(\mathcal{C}_{(d_-,d_+)})$ if it is analytic in $\mathcal{C}_{(d_-,d_+)}$ and satisfies

$$\int_{d_-}^{d_+} |f(x + \mathrm{i}y)|\,\mathrm{d}y \to 0, \quad x \to \pm\infty,$$
$$\|f\|^{\pm} := \lim_{\epsilon \to 0^+} \int_{\mathbb{R}} |f(x + \mathrm{i}(d_\pm \mp \epsilon))|\,\mathrm{d}x < +\infty.$$

From [168], we can approximate the Hilbert transform of a function $f \in H(\mathcal{C}_{(d_-,d_+)})$ via

$$H_{h,\infty}(f,0)(x) = \sum_{m=-\infty}^{\infty} f(mh)\frac{1-\cos(\pi(x-mh)/h)}{\pi(x-mh)/h}, \tag{7.14}$$

with the error bound

$$|\mathcal{H}f(x) - H_{h,\infty}(f,0)(x)| \le \frac{e^{-\pi d_0/h}}{\pi d_0(1-e^{-\pi d_0/h})}(\|f\|^- + \|f\|^+), \tag{7.15}$$

where $d_0 = \min(-d_-, d_+)$. Note that the error decays exponentially in $1/h$, which is an attractive property. The truncated approximation of (7.14) is given by

$$H_{h,M}(f)(x) = \sum_{m=-M}^{M} f(mh) \frac{1-\cos(\pi(x-mh)/h)}{\pi(x-mh)/h},$$

which can be equivalently written as

$$H_{h,M}(f)(x) = \sum_{m=-M}^{M} f(mh) \operatorname{sinc}\left(\frac{x-mh}{2h}\right) \sin\left(\frac{x-mh}{2h}\right), \tag{7.16}$$

where $\operatorname{sinc}(x) := \sin(\pi x)/(\pi x)$. The truncation error of (7.16) depends on the tail behaviour of $f(\cdot + \mathrm{i}a)$. In the context of characteristic functions, the following holds in general:

$$|f(x+\mathrm{i}a)| \leq \kappa |x|^n \exp(-c|x|^\nu), \quad x \in \mathbb{R}, \tag{7.17}$$

for some $\kappa, \nu, c > 0$, and $n \in \mathbb{R}$. If f satisfies (7.17), then the truncation error is bounded by (7.15) plus the additional term

$$\mathcal{T}_{h,M} := \frac{2\kappa}{\nu c^{(n+1)/\nu} h} \Gamma\left(\frac{n+1}{\nu}, c(Mh)^\nu\right), \tag{7.18}$$

where $\Gamma(s,b) := \int_b^\infty e^{-t} t^{s-1} \, \mathrm{d}t$ is the incomplete Gamma function. The dominant term in the truncation error of level M is thus $\exp(-c(Mh)^\nu$ and thus decays exponentially in Mh. To this end, we can choose $h = h(M)$ so that the discretization errors from (7.15), which decay according to $\exp(-\pi d_0/h)$, and the truncation errors from (7.18) decay at the same rate, i.e.,

$$\exp(-\pi d_0/h) = \exp(-c(Mh)^\nu) \tag{7.19}$$

$$h(M) = (\pi d_0/c)^{\frac{1}{1+\nu}} M^{-\frac{\nu}{1+\nu}}. \tag{7.20}$$

Algorithm 3 ReLU Network Verification

```
$\varphi_{\mathbf{x}}$, $\{c, d\}$, $N, M, h, p$
Output: $\hat{\Delta}$, pass/fail
1: $\varphi_{\mathbf{x}_j^0} \leftarrow$ Compute initial characteristic function components on grid.
2: $\varphi_{\mathbf{x}_j^L} \leftarrow$ Propagate through ReLU network using (7.13) and (7.10)
3: $\varphi_{\mathbf{y}} \leftarrow$ Compute characteristic function of output r.v. $\mathbf{y} := c^\intercal \mathbf{x}_L$
4: $\Phi_{\mathbf{y}} \leftarrow$ Compute cumulative distribution function using (2.3)
5: $\hat{\Delta} := \mathbb{P}(\mathbf{y} \in \mathcal{S}) = \Phi_{\mathbf{y}}(d)$
```

7.7 Probabilistic Deep Neural Network Verification

With the developed characteristic function machinery outlined in Section 7.5, we can verify ReLU networks to a prescribed degree of accuracy. For example, if $p = 0.05$, then a neural network passes verification if *at least* 95% of the input samples belong in the desired output set $\mathcal{S}$. We presume that the output set $\mathcal{Y} \subseteq \mathbb{R}^{h_L}$ can be represented by a convex polytope, that is, an intersection of halfspaces. For notational simplicity, we consider an output set that can be written as

$$\mathcal{S} = \{y \in \mathbb{R}^{h_l} \mid c^\intercal y \leq d\}. \tag{7.21}$$

and note that generalization to convex polytopes follows easily by analyzing each half-space independently.

The first three parameters the algorithm accepts are the characteristic function of the input, $\varphi_{\mathbf{x}}$, and the parameters that define the half-space, c, d. The last two design choices are the HT resolution, specified by N, h, M, and the cutoff probability for verification, p. The initial characteristic function is then propagated through the network, which yields the final characteristic function. Since the output set is a half-space, the probability for the output $\mathbf{x}^L$ to be in the half-space is given by Theorem 2.2, which can also be written as a Hilbert transform [169],

$$\mathbb{P}_{\mathbf{x}^L}(\mathcal{X}) = \Phi_{\mathbf{y}}(d) = \frac{1}{2} - \frac{\mathrm{i}}{2}\mathcal{H}(e^{-\mathrm{i}td}\varphi_{\mathbf{y}}(t))(0), \tag{7.22}$$

where $\mathbf{y} := c^\intercal \mathbf{x}^L$ and $\varphi_{\mathbf{y}}(t) = \prod_j \varphi_{\mathbf{x}_j^L}(c_j t)$. Thus, Steps 3-4 in Algorithm 1 compute the associated characteristic function and cumulative distribution function of the constraint (7.21). The cumulative distribution function evaluated at $x = d$ represents the probability of the event $\{x \in \mathbb{R}^m : c^\intercal x \leq d\}$; if this value is less than $1 - \Delta$, this is below the cutoff for verification. As an example, if $\Phi_{c^\intercal \mathbf{x}^L}(d) = 0.7$ but $\Delta = 0.1$, then only 70% of samples from the output set lie in the safety set, which is less than the cutoff of 90%; hence the verification test fails in this case.

7.8 Examples

We provide two examples that illustrate the proposed verification algorithm. Both examples use ReLU feedfoward neural networks from the verification literature. All simulations were run on a 32 GB Intel i7-10750H @ 2.60 GHz computer. For computations and memory storage, we use python with JAX [170]. JAX was run on CPU-only mode but can be run on GPUs or TPUs. All trials of the verification algorithm were compared to an empirical truth computed by brute-force propagation of 10^4 samples through the ReLU networks for each example.

7.8.1 Small Toy Neural Network With Cauchy Noise Input

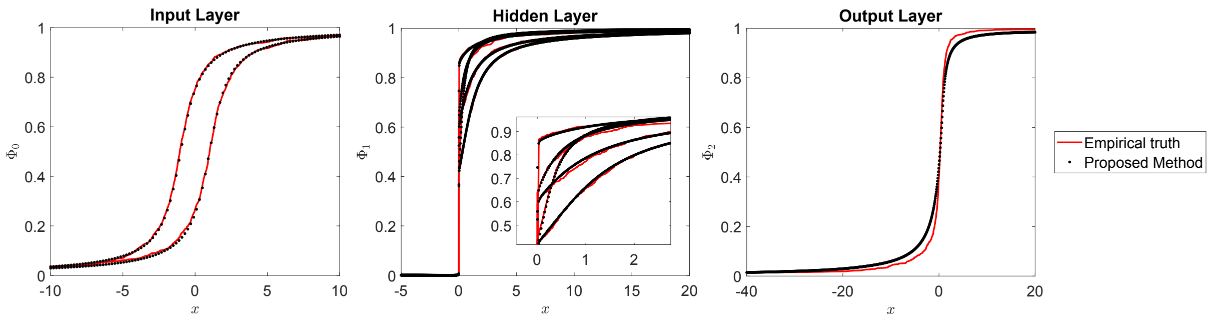


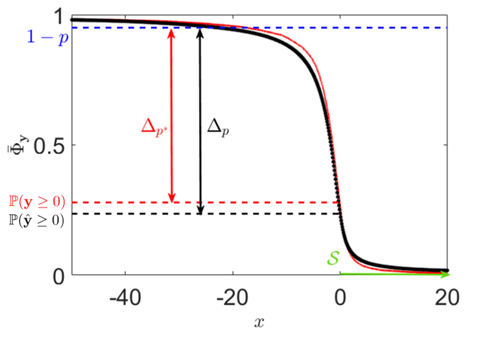
Figure 7.2: The characteristic function and cumulative distribution function for each layer in the ReLU network. The cumulative distribution function computed using the proposed method (black dot) using (2.3) closely resembles the empirical cumulative distribution function computed from brute-force propagation of 10^4 input samples (red line).

To showcase the proposed verification scheme and its advantages, we simulate the following scenario, adapted from [148], and compare against the bounds from a scenario-based approach in [162]. We run a set of 1000 trials, where the ReLU network weights and biases are uniformly sampled from $U[-1,1]$ for various parameters affecting the accuracy of the characteristic function propagation. The network architecture has two inputs, one output, and one hidden layer with 10 neurons. The output safety set is $\mathcal{S} = \{x_L : x_L \geq 0\}$. The maximum probability of lying outside the safety set is $p = 0.05$. Lastly, we also generated (an approximation of) the true output set $\mathcal{Y}$ by propagating 1 million samples from the input set through the network. The inputs are modeled as Cauchy distributions with characteristic function

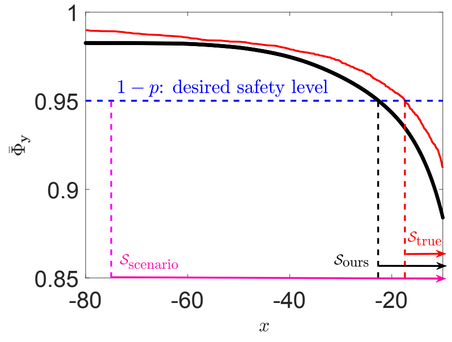
$$\varphi_0(t) = \exp(x_0 \mathrm{i} t - \gamma |t|), \tag{7.23}$$

with locations $x_0^{(1)} = 1, x_0^{(2)} = -1$ and scale $\gamma^{(1)} = \gamma^{(2)} = 1$. The characteristic function of a distribution allows one to easily compute its moments from the derivatives of the characteristic function via Definition 2.2. We emphasize that the derivatives of the Cauchy characteristic function do *not* exist at zero, hence this distribution does not have any standard moments nor does it have a moment generating function. As a result, the methods proposed in [157–159] would not work in this case.

To show how the distribution of the inputs propagates throughout the ReLU network, we take a snapshot of the characteristic functions and CDFs for a few random trials. The plots in images 7.2-7.3 correspond to the parameters $\{N, h, M, d\} = \{10000, 0.05, 5000, 50\}$, namely, 10001 terms in the HT for each of the 10^4 grid points in the domain $\mathcal{C} = \{t : t \in [-50, 50]\}$. Figure 7.2 shows the cumulative distribution function at each layer in the network, as computed from the characteristic function through the HT. It closely resembles the *ground-truth* cumulative distribution function computed via sampling.



(a) For this random trial, our method accurately determines the violation of the output safety set. Our results (black) are very close ($|\tilde{\Delta}| = 0.049$) to the empirically obtained cumulative distribution function and likelihood (red).



(b) Our estimated safety set at the desired probability threshold (black) is much closer to the empirically determined safety set (red) as compared to other SoTA methods (magenta).

Figure 7.3: Comparison of ReLU network safety verification.

The accuracy of this propagation depends on the grid resolution in the frequency domain and the numerical accuracy of the HT used to propagate the characteristic function through the max layer. A finer grid in the frequency domain with a large number of terms in the HT summation yields better results than a coarser grid with fewer terms in the summation. To illustrate this, the fourth column in Table 7.1 computes the average error in probability across all trials for various values of the grid resolution and HT parameters, where $\tilde{\Delta}$ represents the difference in the computed probability of success with the given probability threshold, i.e.,

$$\tilde{\Delta} := \bar{\Phi}_{\mathbf{y}}(0) - (1 - \Delta), \tag{7.24}$$

where $\bar{\Phi}_{\mathbf{y}}(0) := \mathbb{P}(\mathbf{x}^L > 0) = 1 - \Phi_{\mathbf{x}^L}(0)$ is known as the *complementary* cumulative distribution function. See Figure 7.3 for a visual representation of these differences. Thus, the difference in these deltas is a metric for how accurate the characteristic function propagation is — if the numerics were exact ($M, L \to \infty$), then $\tilde{\Delta}_{p^*} = \tilde{\Delta}_p$. Note that the trial for Figure 7.3 fails verification because $\tilde{\Delta} < 0$, which implies that the probability of

Table 7.1: Average verification times, approximation errors for different values of Hilbert transform terms (h, M), and grid resolution (N).

h	N	M	$\mathbb{E}[\lvert\tilde{\Delta}_p\rvert]$	Time (s)	h	N	M	$\mathbb{E}[\lvert\tilde{\Delta}_p\rvert]$	Time (s)
1.0	10^4	5,000	0.0259	17.78	0.7	10^4	2,000	0.0254	7.26
0.6	10^4	5,000	0.0238	17.52	0.7	10^4	10^3	0.0303	3.55
0.5	10^4	5,000	0.0209	18.69	0.7	10^3	10^3	0.1007	0.29
0.1	10^4	5,000	0.0228	18.55	0.7	10^3	100	0.1009	0.03

being in the safety set is *less* than $1 - p = 0.95$.

For the trial in Figure 7.3(a), the estimated probability of being in the safety set is approximately 23.6%, whereas the true probability is 27.9%, giving $|\tilde{\Delta}\tilde{\Delta}_p| = 4.3\%$. In comparison, [162, Appendix D], uses scenario optimization to solve the reverse problem; namely, that of finding the maximal safety set $\mathcal{X} = \bar{r}(p) = \sup_r\{r \in \mathbb{R} : \mathbb{P}(\mathbf{y} > r) \geq 1 - p\}$. Running the method by choosing samples according to $N \geq \frac{2}{\epsilon}(\log(\frac{1}{\tilde{\Delta}}) + 1)$ where $\tilde{\Delta}$ is a confidence parameter such that $\mathbb{P}_{\tilde{r}}\{\mathbb{P}(\mathbf{y} > r) \geq 1 - p\} \geq 1 - \tilde{\Delta}$. With $\tilde{\Delta} = 10^{-5}$, we require 501 samples. Over 500 trials, we generated 501 samples and propagated them through the network. The best $\tilde{r} = -74.99$ with the average over 500 trials is $\mathbb{E}[\tilde{r}] = -3297.92$, whereas the *true* 95% quantile occurs at $x^* = -17.43$ while our estimated quantile is at $x = -22.67$. We mark the quantile values with vertical lines on Figure 7.3(b). The Cauchy distribution has a longer tail than the normal distribution, thus sampling from it produces more outliers. This does not bode well for sampling-based verification methods, causing large over-approximations of the safety set.

Table 7.1 also shows the average time it takes to complete verification for one trial for various parameters. For a grid resolution of 10^4 points and 10^3 HT computations per grid point, we can get verification results in approximately 3 seconds for a two-layer network. Naturally, the accuracy of the propagation degrades with lower values for the parameters, but the computation time decreases, so there is a trade-off between accuracy and speed.

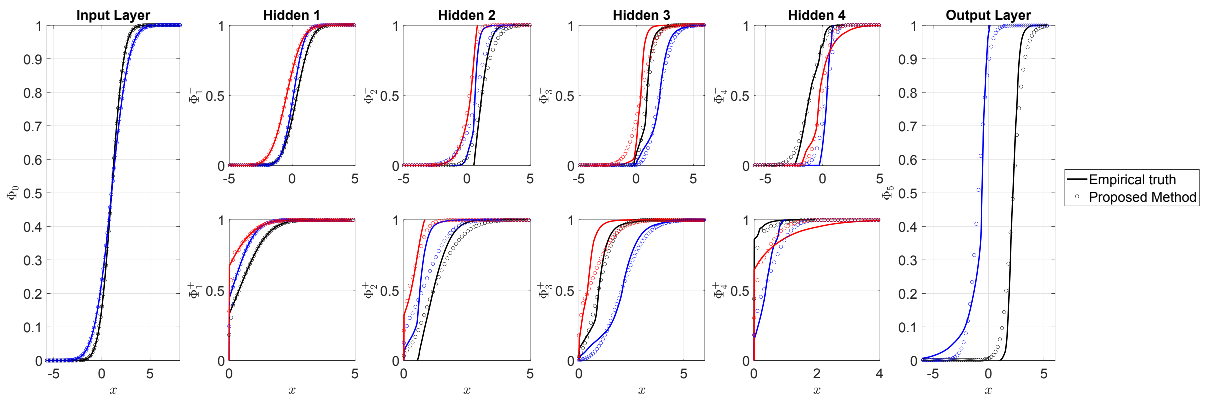


Figure 7.4: Comparison of empirical truth and estimated CDFs for each layer of the ReLU network where only the cumulative distribution function of the first three neurons are plotted before activation, φ^-, and after activation, φ^+. The cumulative distribution function as calculated from the characteristic function via (2.3) (circles) closely matches the empirically calculated cumulative distribution function from the propagation of 10^4 samples (solid lines) even for 50 neuron hidden layer deep networks.

7.8.2 Larger Toy Neural Network with Gaussian Input Noise

We now consider a more complex network based on [157]. In this example, we have 2 inputs, 5 hidden layers, 50 neurons in each of the 5 hidden layers, and 2 outputs. The inputs are normally distributed with mean $\mu_0 = [1, 1]^\intercal$ and covariance $\Sigma = \text{diag}(1, 2)$. We assume the weights and biases for each layer are randomly chosen from $U[-1, 1]$. For the propagation we use $d = 20$ for the frequency cutoffs, $N = 10^4$ grid points for the frequency resolution, and $h = 0.5$ and $M = 5000$ for the HT computations.

Figure 7.4 shows the evolution of the CDFs of each marginal distribution along the network. The labels $\varphi^{+/-}$ denote the cumulative distribution function before and after the ReLU activation layer. We see that the characteristic function propagation is relatively accurate throughout the whole network given the resolution in the characteristic function and HT. The inaccuracies result from the evaluation of the cumulative distribution function at $x = 0$ as can be first seen in Φ_1^+. The sinc method that was used to compute the HT and cumulative distribution function does not perform very well at discontinuity points and these errors propagate after each max layer [168].

7.9 Conclusion

We have presented a probabilistic verification scheme for ReLU neural networks using the machinery of characteristic functions. We show that our method has a clear representation of distribution propagation through a ReLU feedforward (deep) neural network and verification becomes a evaluation of the cumulative distribution function from the network's output characteristic function. One extension of this work could be to optimize the risk level by minimizing p such that $\mathbb{P}_{f(\mathbf{x}^0)}(\mathcal{X}) \geq \Delta$, for some input distribution $\mathbf{x}^0 \sim \psi^0$. Moreover, we can consider the reverse problem of finding the *largest* input set $\mathcal{X}$ such that a network is probabilistically safe for a given risk level Δ [158,162]. Lastly, it might be possible to extend this framework to other activation functions, as long as one can analytically propagate the characteristic function through that activation function.

Chapter 8

Sample-based Verification of Neural Networks

8.1 Introduction

As we emphasize in the previous chapter, safety of neural nets is paramount at autonomy becomes pervasive. This has been particularly evident in autonomous vehicles [171]. Therefore, the development of tools to verify neural networks and prevent costly mishaps is paramount, especially as they percolate into high-stakes systems such as those in aerospace [172]. While analytical forms of ReLU activation functions exist to propagate distributions via characteristic function as in Chapter 7, they do note scale to higher dimensions. Consider a two dimensional random vector $\mathbf{x} \in \mathbb{R}^2$ through a ReLU activation function,

$$\begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix} = \begin{bmatrix} \text{ReLU}(\mathbf{x}_1) \\ \text{ReLU}(\mathbf{x}_2) \end{bmatrix} \tag{8.1}$$

Using the characteristic function of a ReLU activation function in (7.4), we can extend it to higher dimension as follows for (8.1),

$$\begin{aligned}
&\mathbb{E}[e^{\mathrm{i}(t_1 \max(\mathbf{x}_1,0)+t_2\max(\mathbf{x}_2,0))}] = \\
&\qquad \frac{1}{4}\mathbb{E}[(1 + e^{it_1\mathbf{x}_1} + e^{it_2\mathbf{x}_2} + e^{i(t_1\mathbf{x}_1+t_2\mathbf{x}_2)} + e^{it_1\mathbf{x}_1}\mathrm{sign}(\mathbf{x}_1) + e^{it_2\mathbf{x}_2}\mathrm{sign}(\mathbf{x}_2) \\
&\qquad + e^{i(t_1\mathbf{x}_1+t_2\mathbf{x}_2)}\mathrm{sign}(\mathbf{x}_1) + e^{i(t_1\mathbf{x}_1+t_2\mathbf{x}_2)}\mathrm{sign}(\mathbf{x}_2) + e^{it_1\mathbf{x}_1}\mathrm{sign}(\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_2) \\
&\qquad + e^{it_2\mathbf{x}_2}\mathrm{sign}(\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_2) + e^{i(t_1\mathbf{x}_1+t_2\mathbf{x}_2)}\mathrm{sign}(\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_2) \\
&\qquad - \mathrm{sign}(\mathbf{x}_1) - \mathrm{sign}(\mathbf{x}_2) - \mathrm{sign}(\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_2))]. \qquad (8.2)
\end{aligned}$$

With this multivariate definition, we already see that there are a large number of terms for which we need to integrate over. Should we increase the dimension of the input random vector, $\mathbf{x}$, by one,

$$\begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \mathbf{y}_3 \end{bmatrix} = \begin{bmatrix} \mathrm{ReLU}(\mathbf{x}_1) \\ \mathrm{ReLU}(\mathbf{x}_2) \\ \mathrm{ReLU}(\mathbf{x}_3) \end{bmatrix}, \qquad (8.3)$$

thus characteristic function of the multivariate ReLU is,

$$\begin{aligned}
&\mathbb{E}[e^{\mathrm{i}(t_1 \max(\mathbf{x}_1,0)+t_2\max(\mathbf{x}_2,0)+t_3\max(\mathbf{x}_3,0))}] = \\
&\quad \frac{1}{6}\mathbb{E}[(\exp(it_1\mathbf{x}_1) + \exp(it_2\mathbf{x}_2) + \exp(it_3\mathbf{x}_3) - \mathrm{sign}(\mathbf{x}_1) - \mathrm{sign}(\mathbf{x}_2) - \mathrm{sign}(\mathbf{x}_3) \\
&\quad + \mathrm{sign}(\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_2) + \mathrm{sign}(\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_3) + \mathrm{sign}(\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_3) \\
&\quad + \exp(it_1\mathbf{x}_1)\exp(it_2\mathbf{x}_2) + \exp(it_1\mathbf{x}_1)\exp(it_3\mathbf{x}_3) + \exp(it_2\mathbf{x}_2)\exp(it_3\mathbf{x}_3) \\
&\quad + \exp(it_1\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_1) - \exp(it_1\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_2) - \exp(it_1\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_3) \\
&\quad - \exp(it_2\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_1) + \exp(it_2\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_2) - \exp(it_2\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_3) \\
&\quad - \exp(it_3\mathbf{x}_3)\mathrm{sign}(\mathbf{x}_1) - \exp(it_3\mathbf{x}_3)\mathrm{sign}(\mathbf{x}_2) + \exp(it_3\mathbf{x}_3)\mathrm{sign}(\mathbf{x}_3)
\end{aligned}$$

$$
\begin{aligned}
&- \mathrm{sign}(\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_3) + \exp(it_1\mathbf{x}_1)\exp(it_2\mathbf{x}_2)\exp(it_3\mathbf{x}_3)\\
&+ \exp(it_1\mathbf{x}_1)\exp(it_2\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_1) + \exp(it_1\mathbf{x}_1)\exp(it_2\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_2)\\
&- \exp(it_1\mathbf{x}_1)\exp(it_2\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_3)\\
&+ \exp(it_1\mathbf{x}_1)\exp(it_3\mathbf{x}_3)\mathrm{sign}(\mathbf{x}_1) - \exp(it_1\mathbf{x}_1)\exp(it_3\mathbf{x}_3)\mathrm{sign}(\mathbf{x}_2)\\
&+ \exp(it_1\mathbf{x}_1)\exp(it_3\mathbf{x}_3)\mathrm{sign}(\mathbf{x}_3)\\
&- \exp(it_2\mathbf{x}_2)\exp(it_3\mathbf{x}_3)\mathrm{sign}(\mathbf{x}_1) + \exp(it_2\mathbf{x}_2)\exp(it_3\mathbf{x}_3)\mathrm{sign}(\mathbf{x}_2)\\
&+ \exp(it_2\mathbf{x}_2)\exp(it_3\mathbf{x}_3)\mathrm{sign}(\mathbf{x}_3)\\
&- \exp(it_1\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_2) - \exp(it_1\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_3)\\
&+ \exp(it_1\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_3)\\
&- \exp(it_2\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_2) + \exp(it_2\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_3)\\
&- \exp(it_2\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_3)\\
&+ \exp(it_3\mathbf{x}_3)\mathrm{sign}(\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_2) - \exp(it_3\mathbf{x}_3)\mathrm{sign}(\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_3)\\
&- \exp(it_3\mathbf{x}_3)\mathrm{sign}(\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_3) + \exp(it_1\mathbf{x}_1)\exp(it_2\mathbf{x}_2)\exp(it_3\mathbf{x}_3)\mathrm{sign}(\mathbf{x}_1)\\
&+ \exp(it_1\mathbf{x}_1)\exp(it_2\mathbf{x}_2)\exp(it_3\mathbf{x}_3)\mathrm{sign}(\mathbf{x}_2) + \exp(it_1\mathbf{x}_1)\exp(it_2\mathbf{x}_2)\exp(it_3\mathbf{x}_3)\mathrm{sign}(\mathbf{x}_3)\\
&+ \exp(it_1\mathbf{x}_1)\exp(it_2\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_2) - \exp(it_1\mathbf{x}_1)\exp(it_2\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_3)\\
&- \exp(it_1\mathbf{x}_1)\exp(it_2\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_3) - \exp(it_1\mathbf{x}_1)\exp(it_3\mathbf{x}_3)\mathrm{sign}(\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_2)\\
&+ \exp(it_1\mathbf{x}_1)\exp(it_3\mathbf{x}_3)\mathrm{sign}(\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_3) - \exp(it_1\mathbf{x}_1)\exp(it_3\mathbf{x}_3)\mathrm{sign}(\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_3)\\
&- \exp(it_2\mathbf{x}_2)\exp(it_3\mathbf{x}_3)\mathrm{sign}(\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_2) - \exp(it_2\mathbf{x}_2)\exp(it_3\mathbf{x}_3)\mathrm{sign}(\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_3)\\
&+ \exp(it_2\mathbf{x}_2)\exp(it_3\mathbf{x}_3)\mathrm{sign}(\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_3) + \exp(it_1\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_3)\\
&+ \exp(it_2\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_3) + \exp(it_3\mathbf{x}_3)\mathrm{sign}(\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_3)\\
&+ \exp(it_1\mathbf{x}_1)\exp(it_2\mathbf{x}_2)\exp(it_3\mathbf{x}_3)\mathrm{sign}(\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_2)\\
&+ \exp(it_1\mathbf{x}_1)\exp(it_2\mathbf{x}_2)\exp(it_3\mathbf{x}_3)\mathrm{sign}(\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_3)\\
&+ \exp(it_1\mathbf{x}_1)\exp(it_2\mathbf{x}_2)\exp(it_3\mathbf{x}_3)\mathrm{sign}(\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_3)\\
&- \exp(it_1\mathbf{x}_1)\exp(it_2\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_3)
\end{aligned}
$$

$$
\begin{aligned}
&- \exp(it_1\mathbf{x}_1)\exp(it_3\mathbf{x}_3)\mathrm{sign}(\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_3) \\
&- \exp(it_2\mathbf{x}_2)\exp(it_3\mathbf{x}_3)\mathrm{sign}(\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_3) \\
&\qquad + \exp(it_1\mathbf{x}_1)\exp(it_2\mathbf{x}_2)\exp(it_3\mathbf{x}_3)\mathrm{sign}(\mathbf{x}_1)\mathrm{sign}(\mathbf{x}_2)\mathrm{sign}(\mathbf{x}_3) + 1)]. \qquad (8.4)
\end{aligned}
$$

We can see that the number of terms scales exponentially. This makes the analytical propagation of the distribution, proposed in Chapter 7, infeasible in higher dimensions. Hence, we require alternative approaches to verify... neural networks at scale. This chapter explores sampling-based approaches to minimize the curse of dimensionality.

8.2 Related Work

Sampling-based verification is not new and many works exist in the scenario optimization literature. For example, the authors in [173] employ scenario optimization for forward reachability problems of dynamical systems where the goal is to determine sets for which some $1-\Delta$ of the probability mass of the state resides within it. Convex scenario optimization arises in the verification of neural networks in [174], similar to Chapter 7, where the authors validate the neural network's robustness through the propagation of samples. To improve the set characterizations for forward reachability, the authors in [175] employ scenario optimization guarantees amenable for non-convex optimization problems. The sampling guarantees in this chapter follow Chapter 5 where we use the Dvoretzky–Kiefer–Wolfowitz inequality [103].

8.3 Main Contribution and Organization

The main contribution of this chapter is a sampling-based approach in verifying neural networks through samples via empirical characterizations of the cumulative distribution function. We enable this through the characterization of high dimensional sets via signed distance functions. In doing so, it naturally allows us to utilize the structure of cumulative distribution functions and empirically compute them via samples with guarantees, as done in the context of stochastic optimal control in Chapter 5.

The chapter is structured as follows. Section 8.4 states two problem statements. The first being for the verification task and the second poses the problem of modifying the specification in the event that the verification task fails. Section 8.5 lays out the procedure to empirically estimate the empirical characteristic function as well as how many samples are necessary for the estimate to be accurate. It also presents a method to modify the set defined specification in the event we cannot assure that guarantee the probability of satisfaction. The conclusion in Section 8.7 outlines future directions of sample-based verification, including how we can employ samples for the same utility characteristic functions provide.

8.4 Problem Statement

We denote a constraint set $C \in \mathcal{M}(\mathcal{Y})$ through a measurable signed-distance function $g_C : \mathcal{Y} \to \mathbb{R}$,

$$g_C(y) := \begin{cases} \inf_{p \in C} \|p - y\|, \ y \in \mathcal{Y} \backslash C, \\ -\inf_{p \in \mathcal{Y} \backslash C} \|p - y\|, \ y \in C. \end{cases} \tag{8.5}$$

Put simply, the signed distance function says that the point $y \in \mathcal{Y}$ is within the set if its output is negative or zero and positive otherwise. Union of two sets are a minimum of between two signed distance functions, intersection of two sets are a maximum two signed distance function, and the subtraction of two signed distance functions is the maximum of one signed distance and the negative of the other.

Problem 11. *Given a nonlinear function $f : \mathcal{X} \to \mathcal{Y}$, e.g. a system with an neural network in the loop, a signed distance function $g_C : \mathcal{Y} \to \mathbb{R}$, and a probability of satisfaction*

$\epsilon \in (0,1)$, determine if

$$\mathbb{P}\left(\{\omega : g_C(f(\mathbf{x}(\omega))) \leq 0\}\right) \geq 1 - \Delta, \tag{8.6}$$

where $1-\Delta \in (0,1)$ is the probability of satisfaction.

Problem 12. *Given a nonlinear function $f : \mathcal{X} \to \mathcal{Y}$, e.g. a system with an neural network in the loop, a parameterized signed distance function, $g_{C,\theta} : \mathcal{Y} \times \mathbb{R} \to \mathbb{R}$, and a probability of satisfaction $\Delta \in (0,1)$, solve*

$$\underset{\theta \in \mathbb{R}}{\text{minimize}} \quad \theta, \tag{8.7a}$$

$$\text{subject to} \quad \mathbb{P}\left(\{\omega : g_C(f(\mathbf{x}(\omega))) - \theta \leq 0\}\right) \geq 1 - \Delta. \tag{8.7b}$$

Solving Problem 11 is crucial for determining whether neural network or a system with a neural network in the loop satisfies specifications with a given probability. In addition, Problem 12 serves two purposes:

1. It provides a tighter sets for the required satisfaction probability, Δ should the original specification be satisfied with high probability.

2. It relaxes the specification to reach the satisfaction probability, Δ.

Regardless, for both, the most basic way to represent the left-hand side of (8.6) is via a cumulative distribution function,

$$\Phi_{g_C(f(\mathbf{x}))}(0) = \mathbb{E}[\mathbf{1}_{g_C(f(\mathbf{x}))\leq 0}], \tag{8.8a}$$

$$= \mathbb{P}(\{\omega \in \Omega |\ g_C(f(\mathbf{x}(\omega))) \leq 0\}), \tag{8.8b}$$

which we denote via the function $\Phi : \mathbb{R} \to [0,1]$. Note that (8.8b) is the original probability constraint and since it is a expectation of the indicator function, i.e. (8.8),

$$\hat{\Phi}_{g_C(f(\mathbf{x}))}(0) = \frac{1}{M}\sum_{i=1}^{M} \mathbf{1}_{g_C(f(\mathbf{x}^{(i)}))\leq 0}. \tag{8.9}$$

However, having $M \to \infty$ is computationally irresponsible to solve both problem statements.

8.5 Method

We can determine how many samples are needed to estimate (8.8) via (8.9). The Dvoretzky-Kiefer-Wolfowitz inequality, which we utilize in Chapter 5, gives a bound on the difference between the empirical distribution function $\hat{\Phi}_{g_C(f(\mathbf{x}))}$ and the true cumulative distribution function $\Phi_{g_C(\mathrm{NN}(\mathbf{x}))}$. It can be used to derive sample complexity estimates, which indicate how many samples are needed to achieve a given confidence level $1-\beta$ and error tolerance ϵ. Formally, given M i.i.d. samples, the Dvoretzky-Kiefer-Wolfowitz inequality states that for any error tolerance $\epsilon > 0$,

$$\mathbb{P}\left(\sup_x |\hat{\Phi}_{g_C(f(\mathbf{x}))}(x) - \Phi_{g_C(f(\mathbf{x}))}(x)| > \epsilon\right) \leq 2e^{-2M\epsilon^2}, \tag{8.10}$$

where $\hat{\Phi}_{g_C(\mathrm{NN}(\mathbf{x}))_n}(x)$ is the empirical cumulative distribution function, $\Phi_{g_C(\mathrm{NN}(\mathbf{x}))_n}(x)$ is the true cumulative distribution function, and N is the sample size. Given a confidence level $1-\beta$, we set:

$$M \geq \left\lceil -\frac{1}{2\epsilon^2}\ln\left(\frac{\beta}{2}\right)\right\rceil, \tag{8.11}$$

following [103].

Therefore, solving Problem 11 is merely a matter of using (8.11) to compute the number of samples via a user specified $\epsilon,\ \beta \in (0,1)$. For Problem 12, we require more exposition. Note that the optimization problem in (8.7) is equivalent to Definition 6.10

of the quantile function,

$$Q(p) = \inf\{x \in \mathbb{R} : \Phi_{g_C(f(\mathbf{x}))}(x) \geq p\}. \tag{8.12}$$

If we can evaluate the cumulative distribution function, which we can even for the empirical cumulative distribution function, we can utilize root-finding, such as the bisection algorithm [133], to find the smallest θ such that,

$$\hat{\Phi}_{g_C(f(\mathbf{x}))}(\theta^*) = 1 - \Delta. \tag{8.13}$$

8.6 Examples

We provide two examples that illustrate the proposed verification algorithm. Both examples use ReLU feedfoward neural networks from the verification literature. All simulations were run on a Intel Core i9-10900K processor and 128 GB RAM. For computations and randomness generation, we use Python with NumPy and SciPy [176, 177]. For neural network evaluations in the toy example, we utilize PyTorch for all evaluations [178].

8.6.1 Toy Neural Network with Cauchy Input

In this example, we replicate the toy neural networks similar to Section 7.8 where we feed a Cauchy input into to feedforward neural network with ReLU activation functions. The network size is 3 inputs, 10 layers, and two outputs. The location, x, and scale, γ parameters of the input Cauchy distribution are 0 and 1 respectively. The weights and biases are randomly initialized using PyTorch's neural network module. We require the neural network's output to reside within a 2-norm ball of size 10,

$$g_C(p) = ||p||_2 - 10. \tag{8.14}$$

We require the specification in (8.14) to be satisfied with $1 - \Delta = 0.99$ probability. To determine the number of samples to empirically confirm the probability of satisfaction, we specify $\epsilon = 0.001$ and $\beta = 0.001$ for (8.11). This results in us needing 3800452 samples.

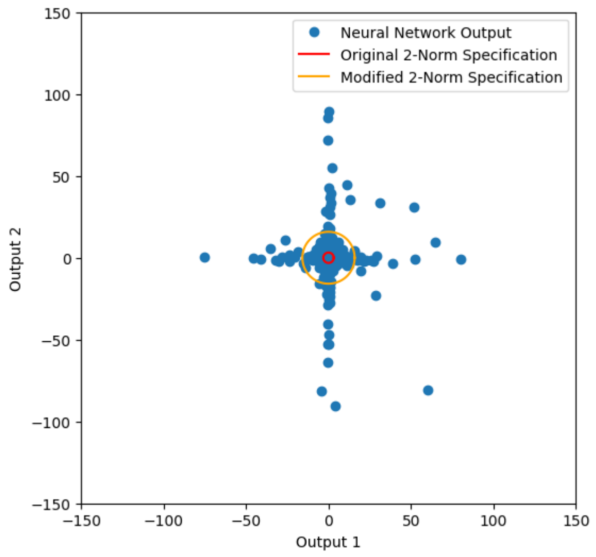


Figure 8.1: The output samples of the toy neural network which is fed with a Cauchy input. The original specification of the 2-Norm with value 10 (in red) fails the probability of satisfaction, $1 - \Delta = 0.99$. Thus, we solve Problem 12 to find a modification of the specification which results in $\theta^* = 241.37$. Thus, to ensure $1 - \Delta = 0.99$ we get larger set, i.e. $||p||_2 - 10 - \theta^*$.

Figure 8.1 shows the output samples of the neural network in blue, the original specification in orange. By the difference in the specification and the modification of the specification we plot in orange, we see that Problem 11 is not satisfied and in solving Problem 12, we obtain $\theta^* = 241.37$ to guarantee $1 - \Delta = 0.99$, taking 5.61 seconds to solve. Figure 8.2 shows the resulting empirical cumulative distribution function plotted using samples along with the probability of satisfaction we require, and the resulting modification to the specification necessary to satisfy it. Note that if we satisfy $1 - \Delta = 0.99$, then the the evaluation at zero would be 0.99, which it is not here.

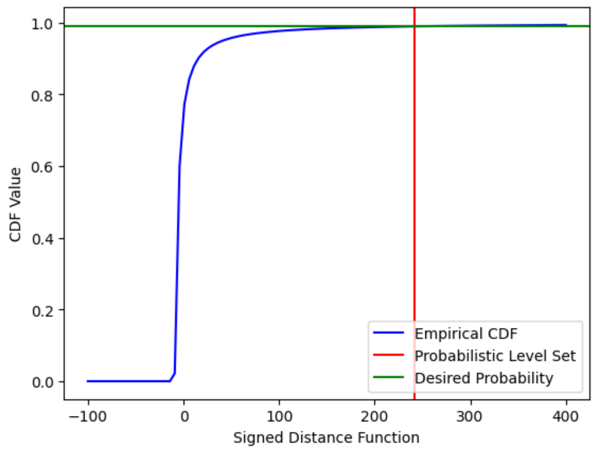


Figure 8.2: The empirical cumulative distribution function of the neural network output from 3800452 samples. We wish to satisfy (8.14) with $1 - \Delta = 0.99$ probability. We do not satisfy the specification, thus we find the level set scaling needed to satisfy $1 - \Delta$ by solving Problem 12 and obtain $\theta^* = 241.37$.

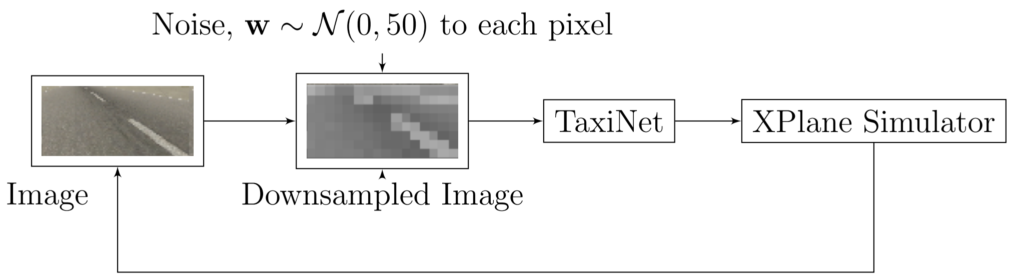


Figure 8.3: The TaxiNet simulator consists of a Cessna 208B Grand Caravan where the camera is placed under the right wing of the aircraft. The camera image is downsampled, where we have added Gaussian noise, and fed into a feedforward neural network, thereby causing deviations in the aircraft's crosstrack position down the runway.

8.6.2 TaxiNet: Pixels to Control Input

We validate the sampling-based approach on the TaxiNet benchmark shown in Figure 8.3 [11]. TaxiNet consists of a neural network which predicts heading angle and the cross track position from images from a camera attached on the right wing of a Cessna 208B Grand Caravan taxiing at 5 m/s down runway 04 of Grant County International Airport. The crosstrack and heading angle are fed into a proportional controller,

$$\phi = -0.74p - 0.44\theta, \tag{8.15}$$

where ϕ is the steering angle of the aircraft. Gaussian noise to each pixel, i.e. $\mathbf{w} \sim \mathcal{N}(0, 50)$, corrupts the downsampled image that is fed into TaxiNet.

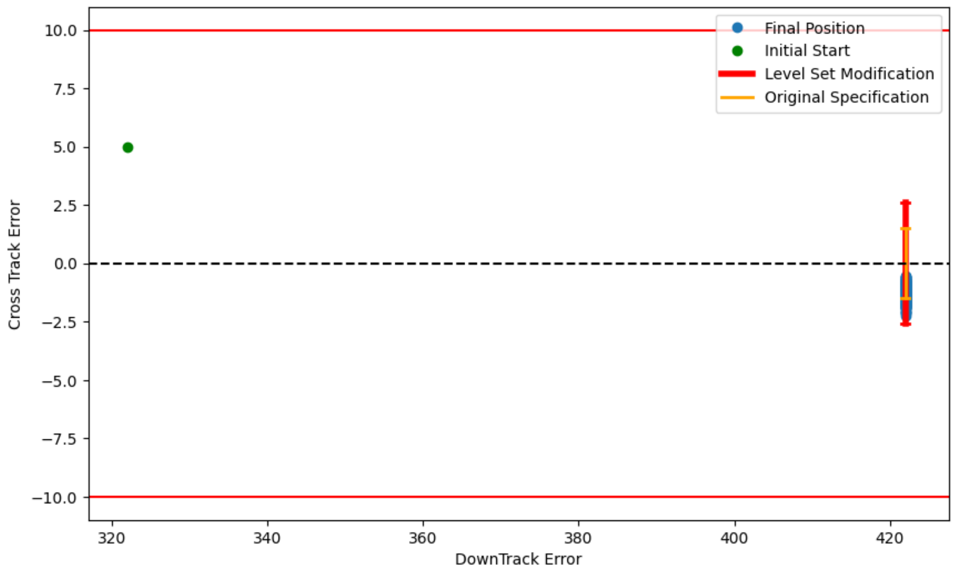


Figure 8.4: Top down visual of the TaxiNet experiment. The Cessna 208B Grand Caravan starts at the cross track position of 5 meters and attempts to get centerline. However, because the downsampled image is corrupted, the aircraft veers off. Solving Problem 12 to ensure $1 - \Delta = 0.99$, results in a larger set than the original specification dictates, i.e. $|p| - 1 - \theta^*$, where $\theta^* = 1.1$.

The aircraft starts at crosstrack position $p_0 = 5$ meters, heading angle $\theta_0 = 10$ degrees, and runway downtrack position of 322 meters. At 422 meters, wish to validate whether the crosstrack position is within 1 meters of the centerline of the runway, i.e.

$$g_C(p) = |p| - 1, \tag{8.16}$$

where we have taken the 1-Norm deviation from the aircraft's crosstrack position. Figure 8.4 visually overviews the experiment and specification, (8.16), in orange. After we solve the root finding problem that attempts to compute the quantile in (8.12), solving Problem 12 to find (8.13).

We require that the specification in (8.16) be satisfied to with $1-\Delta = 0.99$ probability. To determine if the specification is satisfied, we specify $\epsilon = 0.1$ and $\beta = 0.001$ for (8.11),

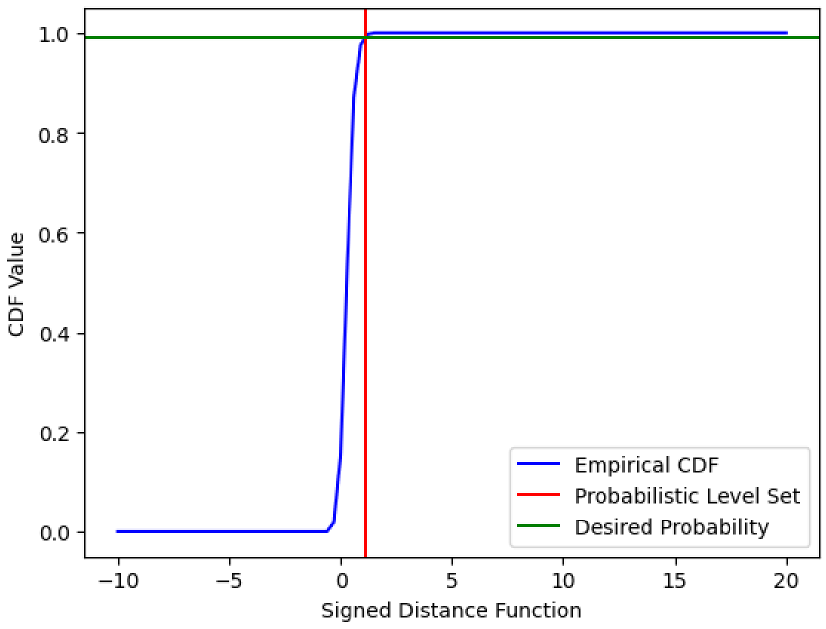


Figure 8.5: The empirical cumulative distribution function of the resulting TaxiNet example computed from 381 samples. We wish to satisfy (8.16) with $1 - \Delta = 0.99$ probability. Since the specification is not satisfied, we can find the level set scaling necessary to satisfy $1 - \Delta$ (green) by solving Problem 12 and obtain $\theta^* = 3.6$ (red).

resulting in requiring 381 samples. Thus we ran the simulator for 381 times, over a span of two hours. Note, by Problem 11, we did not satisfy the specification in (8.16). Thus, we solve Problem 12, to determine a θ that satisfies the probability of satisfaction $1 - \Delta$. In doing so $\theta^* = 1.1$ to ensure $1 - \Delta = 0.99$, which we compute in 589 milliseconds.

8.7 Conclusion

In this chapter, we addressed the challenge of verifying high-dimensional neural networks by exploring sample-based approaches. Given the limitations posed by the curse of dimensionality, especially with purely analytic methods, the introduction of sample-based empirical characteristic functions offers a practical solution. These methods allow for more computationally efficient estimation of the probability of specification satisfaction. The results demonstrate that sample-based techniques hold promise in overcoming the challenges inherent in high-dimensional verification problems. Furthermore, the empirical approaches outlined in this chapter provide both accuracy and computational feasibility, even when traditional analytic methods fail. Future work should focus on refining these sample-based techniques, enhancing their robustness and scalability, especially in safety-critical applications. Additionally, integrating hybrid methods that combine both analytic and sample-based approaches could further improve the reliability and efficiency of neural network verification.

Part IV

Conclusion

Chapter 9

Conclusion

In this dissertation, we have advanced the field of stochastic optimal control by developing new theoretical frameworks and algorithms that address the challenge of controlling dynamical systems under uncertainty. The work is structured into three primary parts, each contributing to a distinct aspect of stochastic control and expanding into neural network verification. Through the combination of these approaches, this dissertation offers a comprehensive set of tools for learning, propagating, and exploiting uncertainty in stochastic control systems, showing promise for developing more tools for stochastic optimal control and neural network verification. The work not only enhances the understanding of stochastic optimal control but also sets the stage for future research into more complex, data-driven, and real-time control systems.

9.1 Summary of Contributions

In the first part, we addressed model-based stochastic optimal control, focusing on systems where the underlying dynamics are known but are subject to uncertainties, particularly those with non-Gaussian disturbances. A significant contribution in this part is the development of a scalable, convex solution for open-loop control of linear systems with log-concave disturbances. By leveraging characteristic functions, we bypass the need for sampling or moment-based methods, which often introduce conservatism or computational inefficiency. Our method allows for the efficient enforcement of chance constraints through convex optimization, offering a more tractable approach to handling non-Gaussian uncertainties. We then turn to utilizing a central representation of the stochasticity via characteristic functions to steer linear systems with affine disturbance feedback controllers. We show that we can steer linear systems in the presence of general disturbances through measuring the distance between characteristic functions of the state and the desired distribution.

The second part of the dissertation turns towards data-driven stochastic optimal control, where the system's disturbances or even the system itself are unknown. We proposed a novel methodology for solving stochastic control problems with unknown disturbances by utilizing empirical characteristic functions derived from observed data. This approach allows us to construct a convex reformulation of the control problem while providing theoretical guarantees through confidence intervals. This framework opens new possibilities for practical applications where model uncertainties are prevalent, offering a robust alternative to traditional approaches that rely on assumed distributions. Moving into the state-of-the-art, we lay the groundwork of utilizing a characteristic function representation of the cost to derive not only the average, but other performance metrics such at Value-at-Risk, Conditional-Value-at-Risk, and expectiles.

Finally, in the third part, we explored the probabilistic verification of neural networks. We presented a framework for propagating characteristic functions through neural networks, particularly those utilizing ReLU activation functions, allowing for the analytical verification of network outputs under probabilistic constraints. Additionally, we show how an analytical approach can fail in higher dimensions and propose a sample-based method with guarantees for neural network verification, providing a means to quantify the probability of satisfying a given specification.

9.2 Future Work

There are many directions to extend work in this dissertation.

1. *Further exploring properties of characteristic functions*: Properties such as convexity would be crucial for optimizing in the presence of uncertainty. For example, Pólya identified some properties of a function to qualify as a characteristic function, one of which is convexity of the positive half plane of a characteristic function [179]. Should it be possible to exploit convexity or generalizations such as quasiconvexity [35], invexity [180], or exploiting pseudoconvex domains [181] of characteristic functions, we may be improve the efficiency and relax the assumptions made within the stochastic optimal control problems in this dissertation.

2. *Working with learned approximations of characteristic functions*: While the empirical characteristic function has proven useful, they are not ideal to work with to extract the various elements that characteristic functions have to offer. Chapter 5 addressed this by using a smoothing factor with the empirical characteristic function. Nonetheless, it would relevant to use convex approaches as a proxy for the empirical characteristic function to extract elements such as moments and probabilities. We emphasize convex here as a starting point as there are a multitude of tools such as kernel methods [182] and convex relaxations of neural networks [183], thereby retaining the convex nature of the stochastic optimal control problems.

3. *Expanding reinforcement learning using characteristic functions:* Chapter 6 laid the groundwork for utilizing the characteristic function to represent the cost distribution. Further explorations in using characteristic functions for reinforcement learning may prove useful in not only risk sensitive reinforcement learning [134] and distributional reinforcement learning [115], but also safe reinforcement learning [184] where we cannot allow the systems to violate safety constraints even while learning.

References

[1] G. Calafiore and M. Campi, “The scenario approach to robust control design,” *IEEE Trans. Autom. Control*, vol. 51, no. 5, pp. 742–753, 2006.

[2] M. C. Campi and S. Garatti, “A sampling-and-discarding approach to chance-constrained optimization: Feasibility and optimality,” *J. Optim Theory Appl.*, vol. 148, no. 2, pp. 257–280, 2011.

[3] L. Blackmore, M. Ono, and B. Williams, “Chance-constrained optimal path planning with obstacles,” *IEEE Trans. Robot.*, vol. 27, no. 6, pp. 1080–1094, 2011.

[4] G. Calafiore and L. Ghaoui, “On distributionally robust chance-constrained linear programs,” *J. Optim Theory Appl.*, vol. 130, no. 1, pp. 1–22, 2006.

[5] A. Nemirovski and A. Shapiro, “Convex approximations of chance constrained programs,” *J. Optimization*, vol. 17, pp. 969–996, 2006.

[6] V. Sivaramakrishnan and M. Oishi, “Fast, convexified stochastic optimal open-loop control for linear systems using empirical characteristic functions,” *IEEE Control Sys. Lett.*, vol. 4, no. 4, pp. 1048–1053, 2020.

[7] L. Blackmore, M. Ono, A. Bektassov, and B. Williams, “A probabilistic particle-control approximation of chance-constrained stochastic predictive control,” *IEEE Transactions on Robotics*, vol. 26, no. 3, pp. 502–517, 2010.

[8] J. Parker, A. Serrani, S. Yurkovich, M. Bolender, and D. Doman, “Control-oriented modeling of an air-breathing hypersonic vehicle,” *J. Guid. Control Dyn.*, vol. 30, no. 3, pp. 856–869, 2007.

[9] H. Buschek and A. J. Calise, “Uncertainty modeling and fixed-order controller design for a hypersonic vehicle model,” *Journal of Guidance, Control, and Dynamics*, vol. 20, no. 1, pp. 42–48, 1997.

[10] M. J. Holzinger, C. C. Chow, and P. Garretson, “A primer on cislunar space,” 2021.

[11] S. M. Katz, A. Corso, S. Chinchali, A. Elhafsi, A. Sharma, M. J. Kochenderfer, and M. Pavone, “NASA ULI Aircraft Taxi Dataset.” Stanford Digital Repository, 2021. Available at: https://purl.stanford.edu/zz143mb4347.

[12] J. Gil-Pelaez, “Note on the inversion theorem,” *Biometrika*, vol. 38, no. 3-4, pp. 481–482, 1951.

[13] V. Sivaramakrishnan, A. P. Vinod, and M. M. K. Oishi, “Convexified open-loop stochastic optimal control for linear systems with log-concave disturbances,” *IEEE Transactions on Automatic Control*, vol. 69, no. 2, pp. 1249–1256, 2024.

[14] V. Sivaramakrishnan, J. Pilipovsky, M. Oishi, and P. Tsiotras, “Distribution steering for discrete-time linear systems with general disturbances using characteristic functions,” in *the Proceedings of the 2022 American Control Conference (ACC)*, pp. 4183–4190, 2022.

[15] V. Sivaramakrishnan and M. M. K. Oishi, “Fast, convexified stochastic optimal open-loop control for linear systems using empirical characteristic functions,” *IEEE Control Systems Letters*, vol. 4, no. 4, pp. 1048–1053, 2020.

[16] J. Pilipovsky, V. Sivaramakrishnan, M. Oishi, and P. Tsiotras, “Probabilistic verification of relu neural networks via characteristic functions,” in *Proceedings of*

The 5th Annual Learning for Dynamics and Control Conference, vol. 211 of *the Proceedings of Machine Learning Research*, pp. 966–979, PMLR, 15–16 Jun 2023.

[17] V. Sivaramakrishnan, R. A. Devonport, M. Arcak, and M. M. K. Oishi, "Forward reachability for discrete-time nonlinear stochastic systems via mixed-monotonicity and stochastic order," 2024. (To appear the Proceedings of the 2024 Conference on Decision and Control (CDC)).

[18] K. Sivaramakrishnan, V. Sivaramakrishnan, R. A. Devonport, and M. M. K. Oishi, "Stochastic reachability of uncontrolled systems via probability measures: Approximation via deep neural networks," 2024. (To appear the Proceedings of the 2024 Conference on Decision and Control (CDC)).

[19] I. Pacula, A. Vinod, V. Sivaramakrishnan, C. Petersen, and M. Oishi, "Stochastic multi-satellite maneuvering with constraints in an elliptical orbit," in *2021 American Control Conference (ACC)*, pp. 4261–4268, 2021.

[20] A. J. Thorpe, V. Sivaramakrishnan, and M. M. K. Oishi, "Approximate stochastic reachability for high dimensional systems," in *the Proceedings of the 2021 American Control Conference (ACC)*, pp. 1287–1293, 2021.

[21] V. Sivaramakrishnan, O. Thapliyal, A. Vinod, M. Oishi, and I. Hwang, "Predicting mode confusion through mixed integer linear programming," in *the Proceedings of the 2019 IEEE 58th Conference on Decision and Control (CDC)*, pp. 2442–2448, 2019.

[22] A. P. Vinod, V. Sivaramakrishnan, and M. M. Oishi, "Piecewise-affine approximation-based stochastic optimal control with gaussian joint chance constraints," in *the Proceedings of the 2019 American Control Conference (ACC)*, pp. 2942–2949, 2019.

[23] A. P. Vinod, V. Sivaramakrishnan, and M. M. K. Oishi, "Sampling-free enforcement of non-gaussian chance constraints via fourier transforms," Proceedings of the Fifth International Workshop on Symbolic-Numeric methods for Reasoning about CPS and IoT, p. 9–11, Association for Computing Machinery, 2019.

[24] A. Abate, H. Blom, N. Cauchi, S. Haesaert, A. Hartmanns, K. Lesser, M. Oishi, V. Sivaramakrishnan, S. Soudjani, C.-I. Vasile, and A. P. Vinod, "Arch-comp18 category report: Stochastic modelling," in *ARCH18. 5th International Workshop on Applied Verification of Continuous and Hybrid Systems*, vol. 54 of *EPiC Series in Computing*, pp. 71–103, EasyChair, 2018.

[25] P. Billingsley, *Probability and Measure.* Wiley, 2008.

[26] E. Lukacs, *Characteristic functions.* London: Griffin, 2nd ed., revised & enlarged ed., 1970.

[27] N. G. Ushakov, *Selected Topics in Characteristic Functions.* No. 4 in Modern probability and statistics, Utrecht: VSP, 1999.

[28] H. Cramér, *Mathematical Methods of Statistics.* Princeton Landmarks in Mathematics and Physics, Princeton: Princeton University Press, 1999.

[29] R. Davies, "Numerical inversion of a characteristic function," *Biometrika*, vol. 60, no. 2, pp. 415–417, 1973.

[30] V. Witkovsky, "Numerical inversion of a characteristic function: An alternative tool to form the probability distribution of output quantity in linear measurement models," *ACTA IMEKO*, vol. 5, no. 3, pp. 32–44, 2016.

[31] D. Bertsekas, *Reinforcement learning and optimal control*, vol. 1. Athena Scientific, 2019.

[32] D. Bertsekas and S. E. Shreve, *Stochastic optimal control: the discrete-time case*, vol. 5. Athena Scientific, 1996.

[33] K. Okamoto, M. Goldshtein, and P. Tsiotras, "Optimal covariance control for stochastic systems under chance constraints," *IEEE Control Systems Letters*, vol. 2, no. 2, pp. 266–271, 2018.

[34] J. Skaf and S. Boyd, "Design of affine controllers via convex optimization," *IEEE Transactions on Automatic Control*, vol. 55, no. 11, pp. 2476–2487, 2010.

[35] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[36] T. Lipp and S. Boyd, "Variations and extension of the convex–concave procedure," *Optim. and Engg.*, vol. 17, pp. 263–287, 2016.

[37] R. Horst, P. Pardalos, and N. Van Thoai, *Introduction to Global Optimization*. Springer, 2000.

[38] B. Sriperumbudur and G. Lanckriet, "On the convergence of the concave-convex procedure," in *NeurIPS*, pp. 1759–1767, 2009.

[39] A. Mesbah, "Stochastic model predictive control: An overview and perspectives for future research," *IEEE Ctrl. Syst. Mag.*, vol. 36, no. 6, pp. 30–44, 2016.

[40] A. Shapiro, D. Dentcheva, and A. Ruszczyński, *Lectures on stochastic programming: modeling and theory*. SIAM, 2014.

[41] A. Carè, S. Garatti, and M. Campi, "Fast—fast algorithm for the scenario technique," *Operations Research*, vol. 62, no. 3, pp. 662–671, 2014.

[42] J. A. Paulson, E. A. Buehler, R. D. Braatz, and A. Mesbah, "Stochastic model predictive control with joint chance constraints," *Int'l J. Ctrl.*, vol. 93, no. 1, pp. 126–139, 2020.

[43] F. Oldewurtel, C. Jones, A. Parisio, and M. Morari, “Stochastic model predictive control for building climate control,” *IEEE Trans. Control Syst. Technol.*, vol. 22, no. 3, pp. 1198–1205, 2014.

[44] M. Ono and B. Williams, “Iterative risk allocation: A new approach to robust model predictive control with a joint chance constraint,” in *Conf. on Dec. Ctrl*, pp. 3427–3432, 2008.

[45] P. Kumar and P. Varaiya, *Stochastic systems: Estimation, identification, and adaptive control*, vol. 75. SIAM, 1986.

[46] D. Mayne and P. Falugi, “Stabilizing conditions for model predictive control,” *International Journal of Robust and Nonlinear Control*, vol. 29, no. 4, pp. 894–903, 2019.

[47] A. D. Bonzanini, D. B. Graves, and A. Mesbah, “Learning-based smpc for reference tracking under state-dependent uncertainty: An application to atmospheric pressure plasma jets for plasma medicine,” *IEEE Trans. on Ctrl Sys. Tech.*, vol. 30, no. 2, pp. 611–624, 2022.

[48] D. Limon, I. Alvarado, T. Alamo, and E. Camacho, “Mpc for tracking piecewise constant references for constrained linear systems,” *Automatica*, vol. 44, no. 9, pp. 2382–2387, 2008.

[49] P. K. Mishra, S. S. Diwale, C. N. Jones, and D. Chatterjee, “Reference tracking stochastic model predictive control over unreliable channels and bounded control actions,” *Automatica*, vol. 127, p. 109512, 2021.

[50] P. Hokayem, E. Cinquemani, D. Chatterjee, F. Ramponi, and J. Lygeros, “Stochastic receding horizon control with output feedback and bounded controls,” *Automatica*, vol. 48, no. 1, pp. 77–88, 2012.

[51] M. Cannon, B. Kouvaritakis, and X. Wu, "Model predictive control for systems with stochastic multiplicative uncertainty and probabilistic constraints," *Automatica*, vol. 45, no. 1, pp. 167–172, 2009.

[52] R. D. McAllister and J. B. Rawlings, "Nonlinear stochastic model predictive control: Existence, measurability, and stochastic asymptotic stability," *IEEE Transactions on Automatic Control*, vol. 68, no. 3, pp. 1524–1536, 2023.

[53] P. Hokayem, D. Chatterjee, and J. Lygeros, "On stochastic receding horizon control with bounded control inputs," in *Conf. on Dec. Ctrl*, pp. 6359–6364, 2009.

[54] M. Farina, L. Giulioni, L. Magni, and R. Scattolini, "An approach to output-feedback mpc of stochastic linear discrete-time systems," *Automatica*, vol. 55, pp. 140–149, 2015.

[55] S. Chan, P. Cheng, D. Pitt, T. Myers, D. Klyde, R. Magdaleno, and D. McRuer, "Aeroservoelastic stabilization techniques for hypersonic flight vehicles," Tech. Rep. 187614, NASA, 1991.

[56] P. J. Goulart, E. C. Kerrigan, and J. M. Maciejowski, "Optimization over state feedback policies for robust control with constraints," *Automatica*, vol. 42, no. 4, pp. 523–533, 2006.

[57] J. Lofberg, "Approximations of closed-loop minimax mpc," in *Conf. on Dec. Ctrl*, vol. 2, pp. 1438–1442, 2003.

[58] D. van Hessem and O. Bosgra, "A conic reformulation of model predictive control including bounded and stochastic disturbances under state and input constraints," in *Conf. on Dec. Ctrl*, vol. 4, pp. 4643–4648, 2002.

[59] A. P. Vinod, V. Sivaramakrishnan, and M. Oishi, "Piecewise-affine approximation-based stochastic optimal control with gaussian joint chance constraints," in *Amer. Ctrl. Conf.*, pp. 2942–2949, 2019.

[60] S. Boyd and L. Vandenberghe, *Convex optimization.* Cambridge Univ. Press, 2004.

[61] S. Dharmadhikari and K. Joag-Dev, *Unimodality, convexity, and applications.* Elsevier, 1988.

[62] M. Bagnoli and T. Bergstrom, "Log-concave probability and its applications," *Economic theory*, vol. 26, no. 2, pp. 445–469, 2005.

[63] M. Vitus, Z. Zhou, and C. Tomlin, "Stochastic control with uncertain parameters via chance constrained control," *IEEE Trans. Autom. Control*, vol. 61, no. 10, pp. 2892–2905, 2016.

[64] A. Prékopa, *Stochastic programming.* Springer, 1995.

[65] G. Rote, "The convergence rate of the sandwich algorithm for approximating convex functions," *Computing*, vol. 48, no. 3-4, pp. 337–361, 1992.

[66] A. Bemporad, "Reducing conservativeness in predictive control of constrained systems with disturbances," in *Conf. on Dec. Ctrl*, vol. 2, pp. 1384–1389 vol.2, 1998.

[67] D. Mayne, M. Seron, and S. Raković, "Robust model predictive control of constrained linear systems with bounded disturbances," *Automatica*, vol. 41, no. 2, pp. 219–224, 2005.

[68] J. Löfberg, "Yalmip : A toolbox for modeling and optimization in matlab," in *CACSD Conf.*, (Taipei, Taiwan), 2004.

[69] M. ApS, *The MOSEK optimization toolbox for MATLAB manual. Version 9.3.*, 2021.

[70] F. Sabatino, *Quadrotor control: modeling, nonlinearcontrol design, and simulation.* Master's thesis, KTH Royal Institute of Technology, 2015.

[71] A. Vinod, B. HomChaudhuri, C. Hintz, A. Parikh, S. Buerger, M. Oishi, G. Brunson, S. Ahmad, and R. Fierro, "Multiple pursuer-based intercept via forward stochastic reachability," in *Amer. Ctrl. Conf.*, pp. 1559–1566, 2018.

[72] I. Mathiasson, "Wind power. statistical analysis of wind speed," *Chalmers University of Technology*, 2015.

[73] S. Kotz, T. Kozubowski, and K. Podgórski, *The Laplace distribution and generalizations: a revisit with applications to communications, economics, engineering, and finance.* No. 183, Springer, 2001.

[74] K. F. Caluya and A. Halder, "Reflected Schrödinger bridge: Density control with path constraints," in *American Control Conference*, (New Orleans, LA), pp. 1137–1142, 2021.

[75] I. M. Balci and E. Bakolas, "Covariance steering of discrete-time stochastic linear systems based on wasserstein distance terminal cost," *IEEE Control Systems Letters*, vol. 5, no. 6, pp. 2000–2005, 2020.

[76] Y. Chen, T. T. Georgiou, and M. Pavon, "Optimal steering of a linear stochastic system to a final probability distribution – Part I," *IEEE Trans. Automatic Control*, vol. 61, no. 5, pp. 1158–1169, 2016.

[77] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou, "Information-theoretic model predictive control: Theory and applications to autonomous driving," *IEEE Transactions on Robotics*, vol. 34, no. 6, pp. 1603–1622, 2018.

[78] G. Williams, A. Aldrich, and E. A. Theodorou, "Model predictive path integral control: From theory to parallel computation," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 2, pp. 344–357, 2017.

[79] M. Goldshtein and P. Tsiotras, "Finite-horizon covariance control of linear time-varying systems," in *56th IEEE Conference on Decision and Control*, (Melbourne, Australia), pp. 3606–3611, Dec 12–15 2017.

[80] K. Okamoto and P. Tsiotras, "Optimal stochastic vehicle path planning using covariance steering," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2276–2281, 2019.

[81] E. Bakolas, "Finite-horizon covariance control for discrete-time stochastic linear systems subject to input constraints," *Automatica*, vol. 91, pp. 61–68, 2018.

[82] J. Pilipovsky and P. Tsiotras, "Chance-constrained optimal covariance steering with iterative risk allocation," in *American Control Conference*, (New Orleans, LA), pp. 2011–2016, 2021.

[83] J. Ridderhof, J. Pilipovsky, and P. Tsiotras, "Chance-constrained covariance control for low-thrust minimum-fuel trajectory optimization," in *AAS/AIAA Astrodynamics Specialist Conference*, (Lake Tahoe, CA), Aug 9–13 2020.

[84] L. Blackmore, H. X. Li, and B. C. Williams, "A probabilistic approach to optimal robust path planning with obstacles," in *American Control Conference*, (Minneapolis, MN), pp. 1–7, June 14–16, 2006.

[85] A. Prékopa, "Boole-Bonferroni inequalities and linear programming," *Operations Research*, vol. 36, no. 1, pp. 145–162, 1988.

[86] M. Farina, L. Giulioni, and R. Scattolini, “Stochastic linear model predictive control with chance constraints – a review,” *J. of Process Control*, vol. 44, pp. 53–67, Aug. 2016.

[87] D. Bertsekas and S. Shreve, *Stochastic optimal control: The discrete time case.* Academic Press, 1978.

[88] R. Stengel, *Optimal control and estimation.* Dover, 1994.

[89] A. Nilim and L. El Ghaoui, “Robust control of Markov decision processes with uncertain transition matrices,” *Operations Res.*, vol. 53, no. 5, pp. 780–798, 2005.

[90] S. Samuelson and I. Yang, “Data-driven distributionally robust control of energy storage to manage wind power fluctuations,” in *IEEE Conf. on Ctrl. Technol. and Appl.*, pp. 199–204, 2017.

[91] I. Yang, “A Convex Optimization Approach to Distributionally Robust Markov Decision Processes With Wasserstein Distance,” *IEEE Contr. Syst. Lett.*, vol. 1, no. 1, pp. 164–169, 2017.

[92] G. Darivianakis, A. Eichler, R. Smith, and J. Lygeros, “A data-driven stochastic optimization approach to the seasonal storage energy management,” *IEEE Contr. Syst. Lett.*, vol. 1, no. 2, pp. 394–399, 2017.

[93] B. Calfa, I. Grossmann, A. Agarwal, S. Bury, and J. Wassick, “Data-driven individual and joint chance-constrained optimization via kernel smoothing,” *Comput. & Chem. Eng.*, vol. 78, pp. 51–69, 2015.

[94] J. Caillau, M. Cerf, A. Sassi, E. Trélat, and H. Zidani, “Solving chance constrained optimal control problems in aerospace via kernel density estimation,” *Optim Control Appl Methods*, vol. 39, no. 5, pp. 1833–1858, 2018.

[95] J. Yu, "Empirical Characteristic Function Estimation and its Applications," *Econom. Rev.*, vol. 23, no. 2, pp. 93–123, 2004.

[96] S. Csorgo, "Limit Behaviour of the Empirical Characteristic Function," *Ann. Probab.*, vol. 9, pp. 130–144, Feb. 1981.

[97] A. Feuerverger and R. Mureika, "The Empirical Characteristic Function and Its Applications," *Ann. Stat.*, vol. 5, no. 1, pp. 88–97, 1977.

[98] L. Blackmore, B. Açikmeşe, and D. Scharf, "Minimum-landing-error powered-descent guidance for mars landing using convex optimization," *J. Guid. Control Dyn.*, vol. 33, no. 4, pp. 1161–1171, 2010.

[99] E. Cinquemani, M. Agarwal, D. Chatterjee, and J. Lygeros, "Convexity and convex approximations of discrete-time stochastic control problems with constraints," *Automatica*, vol. 47, no. 9, pp. 2082–2087, 2011.

[100] M. Vitus, Z. Zhou, and C. Tomlin, "Stochastic Control With Uncertain Parameters via Chance Constrained Control," *IEEE Transactions on Automatic Control*, vol. 61, no. 10, pp. 2892–2905, 2016.

[101] B. W. Silverman, *Density estimation for statistics and data analysis.* Chapman and Hall, 1986.

[102] P. Billingsley, *Convergence of probability measures.* Wiley, 2013.

[103] P. Massart, "The Tight Constant in the Dvoretzky-Kiefer-Wolfowitz Inequality," *Ann. Probab.*, pp. 1269–1283, 1990.

[104] T. Tao, *Analysis*, vol. 1. Springer, 2006.

[105] T. Anderson, "Confidence limits for the expected value of an arbitrary bounded random variable with a continuous distribution function," tech. rep., Stanford Dept. Of Statistics, 1969.

[106] J. Romano and M. Wolf, "Explicit nonparametric confidence intervals for the variance with guaranteed coverage," *Commun. Stat. - Theory Methods*, vol. 31, no. 8, pp. 1231–1250, 2002.

[107] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming." http://cvxr.com/cvx, Mar. 2014.

[108] Gurobi Optimization LLC, "Gurobi optimizer reference manual," 2018.

[109] A. Vinod, J. Gleason, and M. Oishi, "SReachTools: Stochastic reachability toolbox for MATLAB," in *Hybrid Systems Control Conference*, 2019. https://unm-hscl.github.io/SReachTools.

[110] Z. Botev, J. Grotowski, D. Kroese, *et al.*, "Kernel density estimation via diffusion," *Ann. Stat.*, vol. 38, no. 5, pp. 2916–2957, 2010.

[111] J. Hicks, "Flight testing of airbreathing hypersonic vehicles," tech. rep., NASA, Office of Management, 1993.

[112] D. Dalle, S. Torrez, J. Driscoll, and M. Bolender, "Flight envelope calculation of a hypersonic vehicle using a first principles-derived model," *17th AIAA International Space Planes and Hypersonic Systems and Technologies Conference*, 2011.

[113] L. Fiorentini, A. Serrani, M. Bolender, and D. B. Doman, "Nonlinear robust adaptive control of flexible air-breathing hypersonic vehicles," *J. Guid. Control Dyn.*, vol. 32, no. 2, pp. 402–417, 2009.

[114] R. S. Sutton, "Integrated modeling and control based on reinforcement learning and dynamic programming," *Advances in neural information processing systems*, vol. 3, 1990.

[115] M. G. Bellemare, W. Dabney, and M. Rowland, *Distributional reinforcement learning*. MIT Press, 2023.

[116] W. Dabney, M. Rowland, M. Bellemare, and R. Munos, "Distributional reinforcement learning with quantile regression," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, 2018.

[117] Y. Wang and M. P. Chapman, "Risk-averse autonomous systems: A brief history and recent developments from the perspective of optimal control," *Artificial Intelligence*, vol. 311, p. 103743, 2022.

[118] A.-m. Farahmand, "Value function in frequency domain and the characteristic value iteration algorithm," in *Advances in Neural Information Processing Systems* (H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché Buc, E. Fox, and R. Garnett, eds.), vol. 32, Curran Associates, Inc., 2019.

[119] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[120] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International conference on machine learning*, pp. 1889–1897, PMLR, 2015.

[121] A. Tsiamis, D. S. Kalogerias, L. F. Chamon, A. Ribeiro, and G. J. Pappas, "Risk-constrained linear-quadratic regulators," in *2020 59th IEEE Conference on Decision and Control (CDC)*, pp. 3040–3047, IEEE, 2020.

[122] J. Yin, Z. Zhang, and P. Tsiotras, "Risk-aware model predictive path integral control using conditional value-at-risk," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7937–7943, IEEE, 2023.

[123] Y. Chen, U. Rosolia, W. Ubellacker, N. Csomay-Shanklin, and A. D. Ames, "Interactive multi-modal motion planning with branch model predictive control," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 5365–5372, 2022.

[124] A. Hakobyan, G. C. Kim, and I. Yang, “Risk-aware motion planning and control using cvar-constrained optimization,” *IEEE Robotics and Automation letters*, vol. 4, no. 4, pp. 3924–3931, 2019.

[125] M. P. Chapman, R. Bonalli, K. M. Smith, I. Yang, M. Pavone, and C. J. Tomlin, “Risk-sensitive safety analysis using conditional value-at-risk,” *IEEE Transactions on Automatic Control*, vol. 67, no. 12, pp. 6521–6536, 2021.

[126] A. Tamar, Y. Chow, M. Ghavamzadeh, and S. Mannor, “Policy gradient for coherent risk measures,” in *Advances in Neural Information Processing Systems* (C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, eds.), vol. 28, Curran Associates, Inc., 2015.

[127] D. Bertsekas, *Abstract dynamic programming.* Athena Scientific, 2022.

[128] W. Dabney, G. Ostrovski, D. Silver, and R. Munos, “Implicit quantile networks for distributional reinforcement learning,” in *International conference on machine learning*, pp. 1096–1105, PMLR, 2018.

[129] M. Rowland, R. Dadashi, S. Kumar, R. Munos, M. G. Bellemare, and W. Dabney, “Statistics and samples in distributional reinforcement learning,” in *International Conference on Machine Learning*, pp. 5528–5536, PMLR, 2019.

[130] R. T. Rockafellar, S. Uryasev, *et al.*, “Optimization of conditional value-at-risk,” *Journal of risk*, vol. 2, pp. 21–42, 2000.

[131] I. Pinelis, “Characteristic function of the positive part of a random variable and related results, with applications,” 2015.

[132] C. S. Phillips, “Interpreting expectiles,” Working Paper 2022-01, United States Air Force Academy, Department of Economics and Geosciences, 2354 Fairchild Drive, Suite 6k-110, United States Air Force Academy, CO 80840, January 2022.

[133] J. Nocedal and S. J. Wright, *Numerical optimization.* Springer, 1999.

[134] A. Tamar, Y. Chow, M. Ghavamzadeh, and S. Mannor, "Policy gradient for coherent risk measures," *Advances in neural information processing systems*, vol. 28, 2015.

[135] A. Ruszczyński, "Risk-averse dynamic programming for markov decision processes," *Mathematical programming*, vol. 125, pp. 235–261, 2010.

[136] D. A. Iancu, M. Petrik, and D. Subramanian, "Tight approximations of dynamic risk measures," *Mathematics of Operations Research*, vol. 40, no. 3, pp. 655–682, 2015.

[137] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, "Stable-baselines3: Reliable reinforcement learning implementations," *Journal of Machine Learning Research*, vol. 22, no. 268, pp. 1–8, 2021.

[138] X. Yang, Y. Ye, X. Li, R. Y. K. Lau, X. Zhang, and X. Huang, "Hyperspectral image classification with deep learning models," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 9, pp. 5408–5423, 2018.

[139] C.-C. Chiu, T. N. Sainath, Y. Wu, R. Prabhavalkar, and et al., "State-of-the-art speech recognition with sequence-to-sequence models," *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4774–4778, 2018.

[140] Y. Huang and Y. Chen, "Survey of state-of-art autonomous driving technologies with deep learning," in *IEEE 20th International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, pp. 221–228, 2020.

[141] Y. Song, M. Steinweg, E. Kaufmann, and D. Scaramuzza, "Autonomous drone racing with deep reinforcement learning," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1205–1212, 2021.

[142] J. Su, D. V. Vargas, and K. Sakurai, "One pixel attack for fooling deep neural networks," *IEEE Transactions on Evolutionary Computation,*, vol. 23, no. 5, pp. 828–841, 2019.

[143] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 86–94, 2017.

[144] A. Lomuscio and L. Maganti, "An approach to reachability analysis for feed-forward relu neural networks," 2017.

[145] C.-H. Cheng, G. Nührenberg, and H. Ruess, "Maximum resilience of artificial neural networks," in *Automated Technology for Verification and Analysis* (D. D'Souza and K. Narayan Kumar, eds.), (Cham), pp. 251–268, Springer International Publishing, 2017.

[146] 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*, pp. 97–117, Springer, 2017.

[147] K. Scheibler, L. Winterer, R. Wimmer, and B. Becker, "Towards verification of artificial neural networks," in *MBMV*, 2015.

[148] R. A. Brown, E. Schmerling, N. Azizan, and M. Pavone, "A unified view of SDP-based neural network verification through completely positive programming," in *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics* (G. Camps-Valls, F. J. R. Ruiz, and I. Valera, eds.), vol. 151 of *Proceedings of Machine Learning Research*, pp. 9334–9355, PMLR, 28–30 Mar 2022.

[149] M. Fazlyab, M. Morari, and G. J. Pappas, "Safety verification and robustness analysis of neural networks via quadratic constraints and semidefinite programming," *IEEE Transactions on Automatic Control*, vol. 67, no. 1, pp. 1–15, 2022.

[150] K. Dvijotham, R. Stanforth, S. Gowal, C. Qin, S. De, and P. Kohli, “Efficient neural network verification with exactness characterization,” in *Proceedings of The 35th Uncertainty in Artificial Intelligence Conference* (R. P. Adams and V. Gogate, eds.), vol. 115 of *Proceedings of Machine Learning Research*, pp. 497–507, 22–25 Jul 2020.

[151] S. Dathathri, K. Dvijotham, A. Kurakin, A. Raghunathan, J. Uesato, R. R. Bunel, S. Shankar, J. Steinhardt, I. Goodfellow, P. S. Liang, and P. Kohli, “Enabling certification of verification-agnostic networks via memory-efficient semidefinite programming,” in *Advances in Neural Information Processing Systems*, vol. 33, pp. 5318–5331, Curran Associates, Inc., 2020.

[152] E. Wong and J. Z. Kolter, “Provable defenses against adversarial examples via the convex outer adversarial polytope,” in *Proceedings of the 35th International Conference on Machine Learning, (ICML)*, vol. 80, (Stockholm, Sweden), pp. 5283–5292, July 10-15, 2018.

[153] A. Raghunathan, J. Steinhardt, and P. S. Liang, “Semidefinite relaxations for certifying robustness to adversarial examples,” in *Advances in Neural Information Processing Systems*, vol. 31, Curran Associates, Inc., 2018.

[154] J. A. Vincent and M. Schwager, “Reachable polyhedral marching (rpm): A safety verification algorithm for robotic systems with deep neural network components,” in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 9029–9035, 2021.

[155] F. Sciacchitano, Y. Dong, and T. Zeng, “Variational approach for restoring blurred images with cauchy noise,” *SIAM Journal on Imaging Sciences*, vol. 8, no. 3, pp. 1894–1922, 2015.

[156] J.-J. Mei, Y. Dong, T.-Z. Huang, and W. Yin, "Cauchy Noise Removal by Nonconvex ADMM with Convergence Guarantees," *Journal of Scientific Computing*, vol. 74, pp. 743–766, Feb. 2018.

[157] M. Fazlyab, M. Morari, and G. J. Pappas, "Probabilistic verification and reachability analysis of neural networks via semidefinite programming," in *2019 IEEE 58th Conference on Decision and Control (CDC)*, pp. 2726–2731, 2019.

[158] 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*, vol. 97 of *Proceedings of Machine Learning Research*, pp. 6727–6736, 09–15 Jun 2019.

[159] M. Pautov, N. Tursynbek, M. Munkhoeva, N. Muravev, A. Petiushko, and I. Oseledets, "Cc-cert: A probabilistic approach to certify general robustness of neural networks," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, pp. 7975–7983, Jun. 2022.

[160] K. Dvijotham, M. Garnelo, A. Fawzi, and P. Kohli, "Verification of deep probabilistic models," in *Advances in Neural Information Processing Systems, SecML workshop*, 2018.

[161] L. Berrada, S. Dathathri, K. Dvijotham, R. Stanforth, R. R. Bunel, J. Uesato, S. Gowal, and M. P. Kumar, "Make sure you're unsure: A framework for verifying probabilistic specifications," in *Advances in Neural Information Processing Systems* (M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, eds.), vol. 34, pp. 11136–11147, Curran Associates, Inc., 2021.

[162] B. G. Anderson and S. Sojoudi, "Data-driven certification of neural networks with random input noise," *IEEE Transactions on Control of Network Systems*, pp. 1–12, 2022.

[163] K. S. Narendra and K. Parthasarathy, "Neural networks and dynamical systems," *International Journal of Approximate Reasoning*, vol. 6, no. 2, pp. 109–131, 1992.

[164] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Transactions on Neural Networks*, vol. 1, no. 1, pp. 4–27, 1990.

[165] E. Weinan, "A proposal on machine learning via dynamical systems," *Communications in Mathematics and Statistics*, vol. 5, pp. 1–11, 3 2017. Dedicated to Professor Chi-Wang Shu on the occasion of his 60th birthday.

[166] I. Pinelis, "Characteristic function of the positive part of a random variable and related results, with applications," *Statistics & Probability Letters*, vol. 106, pp. 281–286, 2015.

[167] J. Pilipovsky, V. Sivaramakrishnan, M. M. K. Oishi, and P. Tsiotras, "Probabilistic verification of relu neural networks via characteristic functions," 2023.

[168] L. Feng and X. Lin, "Inverting analytic characteristic functions and financial applications," *SIAM Journal on Financial Mathematics*, vol. 4, no. 1, pp. 372–398, 2013.

[169] L. Feng and X. Lin, "Inverting analytic characteristic functions and financial applications," *SIAM Journal on Financial Mathematics*, vol. 4, no. 1, pp. 372–398, 2013.

[170] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang, "JAX: composable transformations of Python+NumPy programs," 2018.

[171] Forbes Advisor, "Perception of self-driving cars," 2024. Accessed: 2024-05-30.

[172] K. Dunlap, N. Hamilton, Z. Lippay, M. Shubert, S. Phillips, and K. L. Hobbs, "Demonstrating reinforcement learning and run time assurance for spacecraft inspection using unmanned aerial vehicles," *arXiv preprint arXiv:2405.06770*, 2024.

[173] A. Devonport and M. Arcak, "Estimating reachable sets with scenario optimization," in *Learning for dynamics and control*, pp. 75–84, PMLR, 2020.

[174] B. G. Anderson and S. Sojoudi, "Data-driven certification of neural networks with random input noise," *IEEE transactions on control of network systems*, vol. 10, no. 1, pp. 249–260, 2022.

[175] E. Dietrich, A. Devonport, and M. Arcak, "Nonconvex scenario optimization for data-driven reachability," in *6th Annual Learning for Dynamics & Control Conference*, pp. 514–527, PMLR, 2024.

[176] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, "Array programming with NumPy," *Nature*, vol. 585, pp. 357–362, Sept. 2020.

[177] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern,

E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[178] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.

[179] G. Pólya *et al.*, "Remarks on characteristic functions," in *Proc. First Berkeley Conf. on Math. Stat. and Prob*, pp. 115–123, 1949.

[180] A. Ben-Israel and B. Mond, "What is invexity?," *The Journal of the Australian Mathematical Society. Series B. Applied Mathematics*, vol. 28, no. 1, p. 1–9, 1986.

[181] R. M. Range, "What is a pseudoconvex domain," *Notices of the AMS*, vol. 59, no. 2, 2012.

[182] K. Muandet, K. Fukumizu, B. Sriperumbudur, B. Schölkopf, *et al.*, "Kernel mean embedding of distributions: A review and beyond," *Foundations and Trends® in Machine Learning*, vol. 10, no. 1-2, pp. 1–141, 2017.

[183] T. Ergen and M. Pilanci, "Revealing the structure of deep neural networks via convex duality," in *International Conference on Machine Learning*, pp. 3004–3014, PMLR, 2021.

[184] R. Mitta, H. Hasanbeig, J. Wang, D. Kroening, Y. Kantaros, and A. Abate, "Safeguarded progress in reinforcement learning: Safe bayesian exploration for control policy synthesis," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, pp. 21412–21419, 2024.