ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

RSL Robotic Systems Lab
Prof. Dr. Marco Hutter

Semester Thesis

Constrained Path Integral Optimal Control

Spring Term 2019

Supervised by:
Carius, Jan
Farshidian, Farbod Dr.

Author:
Wang, Yize

Declaration of Originality

I hereby declare that the written work I have submitted entitled

Constrained Path Integral Optimal Control

is original work which I alone have authored and which is written in my own words.[1]

Author(s)

Yize	Wang

Student supervisor(s)

Jan Farbod	Carius Farshidian

Supervising lecturer

Marco	Hutter

With the signature I declare that I have been informed regarding normal academic citation rules and that I have read and understood the information on 'Citation etiquette' (`https://www.ethz.ch/content/dam/ethz/main/education/rechtliches-abschluesse/leistungskontrollen/plagiarism-citationetiquette.pdf`). The citation conventions usual to the discipline in question here have been respected.

The above written work may be tested electronically for plagiarism.

Zurich, 2019-10-5
Place and date

Yize Wang
Signature

[1] Co-authored work: The signatures of all authors are required. Each signature attests to the originality of the entire piece of written work in its final form.

Intellectual Property Agreement

The student acted under the supervision of Prof. Hutter and contributed to research of his group. Research results of students outside the scope of an employment contract with ETH Zurich belong to the students themselves. The results of the student within the present thesis shall be exploited by ETH Zurich, possibly together with results of other contributors in the same field. To facilitate and to enable a common exploitation of all combined research results, the student hereby assigns his rights to the research results to ETH Zurich. In exchange, the student shall be treated like an employee of ETH Zurich with respect to any income generated due to the research results.
This agreement regulates the rights to the created research results.

1. Intellectual Property Rights

1. The student assigns his/her rights to the research results, including inventions and works protected by copyright, but not including his moral rights ("Urheberpersönlichkeitsrechte"), to ETH Zurich. Herewith, he cedes, in particular, all rights for commercial exploitations of research results to ETH Zurich. He is doing this voluntarily and with full awareness, in order to facilitate the commercial exploitation of the created Research Results. The student's moral rights ("Urheberpersönlichkeitsrechte") shall not be affected by this assignment.
2. In exchange, the student will be compensated by ETH Zurich in the case of income through the commercial exploitation of research results. Compensation will be made as if the student was an employee of ETH Zurich and according to the guidelines "Richtlinien für die wirtschaftliche Verwertung von Forschungsergebnissen der ETH Zürich".
3. The student agrees to keep all research results confidential. This obligation to confidentiality shall persist until he or she is informed by ETH Zurich that the intellectual property rights to the research results have been protected through patent applications or other adequate measures or that no protection is sought, but not longer than 12 months after the collaborator has signed this agreement.
4. If a patent application is filed for an invention based on the research results, the student will duly provide all necessary signatures. He/she also agrees to be available whenever his aid is necessary in the course of the patent application process, e.g. to respond to questions of patent examiners or the like.

2. Settlement of Disagreements

Should disagreements arise out between the parties, the parties will make an effort to settle them between them in good faith. In case of failure of these agreements, Swiss Law shall be applied and the Courts of Zurich shall have exclusive jurisdiction.

Zurich, 2019-10-5
Place and date

Yize Wang
Signature

Contents

Abstract

Path integral methods have been an important tool for systems with stochastic continuous dynamics. Recent work mainly focuses on unconstrained control problems and constrained ones are rarely studied. This thesis firstly derives path integral formulation for unconstrained cases and shows the optimal cost-to-go and optimal control can be obtained through sampling. Secondly, a theoretical solution is presented to deal with state-input-inequality constraints. We assume the activeness of constraints stays the same as the preceding time step. Based on augmented Lagrangian methods, the optimal cost-to-go and optimal control can be computed given that constraint activeness, which is also the signs of optimal Lagrangian multipliers at current time step. Then, we update the optimal Lagrangian multipliers with the partial derivative of the optimal cost-to-go. Thirdly, a practical solution is introduced as an alternative method, where a sophisticated sampling policy is implemented to respect all constraints, and the optimal control can be obtained just like in the unconstrained cases. Finally, examples are given to illustrate path integrals' characteristics, and to compare unconstrained and constrained cases.

Symbols

Symbols

$\boldsymbol{x}$	system state
$\boldsymbol{u}$	input vector
$\boldsymbol{w}$	Wiener noise
C	cost function
J	optimal cost-to-go
t	time
$\boldsymbol{f}$	function that maps time increment to state increment
$\boldsymbol{G}$	function that maps input vector and time increment to state increment
Φ	terminal cost function
$\boldsymbol{R}$	input cost weight
$\boldsymbol{Q}$	standard deviation
V	state-dependent cost
$\boldsymbol{r}$	state and input coupling cost
ψ	advantage function
$\boldsymbol{\varepsilon}$	sampled noise input
$\boldsymbol{\lambda}$	Lagrangian multiplier
F	cumulative distribution function of standard Gaussian distribution
p	probability density function of standard Gaussian distribution
M	cumulative distribution function of truncated Gaussian distribution
m	probability density function of truncated Gaussian distribution

Indices

k	timestep
n	sample number

Acronyms and Abbreviations

PDF	Probability Density Function
CDF	Cumulative Distribution Function
HJB	Hamilton–Jacobi–Bellman
MPC	Model Predictive Control

DoF Degree of Freedom

Chapter 1

Introduction

Stochastic optimal control methods play an important role in many science and engineering fields, such as chemistry, finance, and robotics. In chemistry, they can be used to compute kinetic isotope effects [1]; In finance, the problem may be to predict stock market behavior [2] or to determine the optimal option price [3]; In robotics, they are applied to find the optimal trajectories in the sense of minimal cost [4].

In the presence of noise, the optimal control can be obtained by solving a stochastic HJB equation. Unfortunately, in general there is no analytic solution to that PDE. Numerical solutions, such as grid-based methods [5, 6], are developed but incur prohibitive computational cost given a high dimensionality. Kappen [7, 8] studied unconstrained path integral formulation for linear-quadratic control problem. He showed that, if the noise meets certain assumptions, and the noise as well as input enter the system dynamics affinely, the non-linear stochastic HJB equation can be log-transformed into a linear PDE and thus be solved by forward simulating a diffusion process according to Feynman-Kac formula.

However, real systems are often limited by constraints, such as actuation limits and state limits. For example, we want the contact force to stay in the friction cone when we plan motions for a legged robot, which leads to a state-input-inequality constraint, as shown in Figure 1.1. Therefore, in this thesis, we investigate how to solve the stochastic HJB equation subject to state-input-inequality constraints and propose two solutions, a theoretical solution with augmented Lagrangian and a practical one with sophisticated sampling policies.

We start with the derivation of unconstrained path integral methods in Chapter 2. We first formulate the stochastic optimal control problem and point out the optimality condition comes as a stochastic HJB equation. With some assumptions on the noise, we apply log transformation for the optimal cost-to-go and turn the non-linear stochastic HJB equation into a linear PDE. Finally, the optimal cost-to-go and optimal control are computed through forward sampling based on Feynman-Kac formula.

In Chapter 3, we incorporate state-input inequality constraints to the system. Then, the optimization problem is reformulated based on game theory and augmented Lagrangian methods. We assume the optimal Lagrangian multipliers at current time step have the same signs as the preceding ones and solve the optimal control and optimal cost-to-go, after which the optimal Lagrangian multiples are updated.

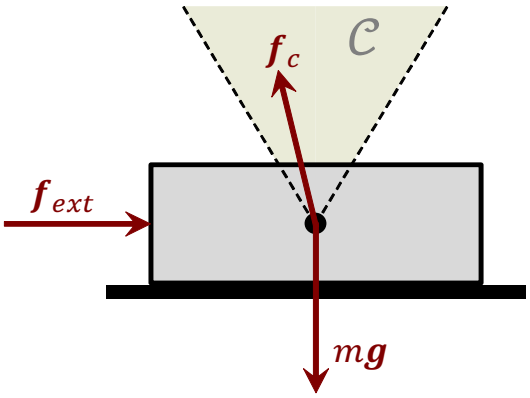


Figure 1.1: Contact force and friction cone

Chapter 4 provides an alternative solution to the stochastic HJB equation. We note that the optimal control computed in the unconstrained case is a convex combination of sampled noise inputs. Instead of going through augmented Lagrangian methods and making more assumptions, we force the optimal control to respect constraints by only accepting constraint-consistent sampled noise inputs.

Chapter 5 presents examples of both unconstrained and constrained cases. Characteristics of path integral methods, such as symmetry breaking and delayed decision, are illustrated and discussed. At last, we show how to formulate the control problem to meet specific criteria.

Finally, Chapter 6 discusses our contribution, some limitations of our methods and the future work.

Chapter 2

Unconstrained path integral control

In this chapter[1], we first introduce the notation with a deterministic system, and then generalize to a stochastic system. By making some assumptions on the noise and applying logarithmic transformation, we are able to find the optimal control and optimal cost through sampling.

2.1 Problem description

We first consider a deterministic system with differential dynamics:

$$\mathrm{d}\boldsymbol{x} = \boldsymbol{f}(\boldsymbol{x}, t)\ \mathrm{d}t + \boldsymbol{G}(\boldsymbol{x})\boldsymbol{u}\ \mathrm{d}t\ , \tag{2.1}$$

where $\boldsymbol{x}$ stands for the system state and $\boldsymbol{u}$ is the input we apply to the system. Function $\boldsymbol{f}$ maps time increment to the state increment, and function $\boldsymbol{G}$ maps time increment as well as input to the state increment.

We define the cost function C as the sum of the terminal cost and the integral of cost accumulated from time t_0 to t_f:

$$C(\boldsymbol{x}_0, t_0, \boldsymbol{u}(t_0 \to t_f)) = \Phi(\boldsymbol{x}_{t_f}) + \int_{t_0}^{t_f} \left(\frac{1}{2}\boldsymbol{u}^\top \boldsymbol{R}\boldsymbol{u} + V(\boldsymbol{x}) + \boldsymbol{r}^\top(\boldsymbol{x})\boldsymbol{u}\right)\ \mathrm{d}\tau\ , \tag{2.2}$$

where $\boldsymbol{u}(t_0 \to t_f)$ denotes input sequence on the time interval $[t_0, t_f]$, function $\Phi(\cdot)$ denotes the terminal cost fully defined by the terminal state $\boldsymbol{x}_{t_f}$, function $V(\cdot)$ is an arbitrary state-dependent cost term, and coupling cost term $\boldsymbol{r}(\boldsymbol{x})$ associates the state with inputs. Note that we assume the input cost weight $\boldsymbol{R}$ to be positive definite and symmetric, and that the cost function is at most quadratic in input $\boldsymbol{u}$.

In practice, control inputs applied to the system are always affected by noise. Therefore, we incorporate the noise into the system dynamics to plan robustly and consider the stochastic differential equation by introducing noise $\boldsymbol{w}$ into the input channel:

$$\mathrm{d}\boldsymbol{x} = \boldsymbol{f}(\boldsymbol{x}, t)\ \mathrm{d}t + \boldsymbol{G}(\boldsymbol{x})(\boldsymbol{u}\ \mathrm{d}t + \boldsymbol{Q}(\boldsymbol{x}, t)\ \mathrm{d}\boldsymbol{w})\ , \tag{2.3}$$

where $\boldsymbol{w}$ is a normalized Brownian motion with zero mean and identity covariance matrix, multiplied by the noise standard deviation $\boldsymbol{Q}$. The equation (2.3) describes

[1] This chapter is based on the original work of Jan Carius (`jan.carius@mavt.ethz.ch`) and not original in this thesis.

a Brownian motion with diffusion $\boldsymbol{GQ}$ and drift $\boldsymbol{f} + \boldsymbol{Gu}$.

The cost function that we want to minimize is now turned into the expectation of the nominal cost (2.2), i.e.,

$$C(\boldsymbol{x}_0, t_0, \boldsymbol{u}(t_0 \to t_f)) = \mathbb{E}_{\boldsymbol{x}_0}\left[\Phi(\boldsymbol{x}_{t_f}) + \int_{t_0}^{t_f}\left(\frac{1}{2}\boldsymbol{u}^\top \boldsymbol{R}\boldsymbol{u} + V(\boldsymbol{x}) + \boldsymbol{r}^\top(\boldsymbol{x})\boldsymbol{u}\right)\,\mathrm{d}\tau\right].$$

The symbol $\mathbb{E}_{\boldsymbol{x}_0}$ thereby denotes the expectation over all trajectories that start at $\boldsymbol{x}_0$ and are governed by the stochastic dynamics (2.3).

2.2 Optimality condition

The optimal cost-to-go J in the stochastic framework is the minimal expected cost that can be achieved when starting from state $\boldsymbol{x}$ at time $t \in [t_0, t_f]$:

$$\begin{aligned} J(\boldsymbol{x}(t), t) &= \min_{\boldsymbol{u}(t\to t_f)} C(\boldsymbol{x}, t_0, \boldsymbol{u}(t_0 \to t_f)) \\ &= \min_{\boldsymbol{u}(t\to t_f)} \mathbb{E}_{\boldsymbol{x}(t)}\left[\Phi(\boldsymbol{x}_{t_f}) + \int_{t}^{t_f}\left(\frac{1}{2}\boldsymbol{u}^\top \boldsymbol{R}\boldsymbol{u} + V(\boldsymbol{x}) + \boldsymbol{r}^\top(\boldsymbol{x})\boldsymbol{u}\right)\,\mathrm{d}\tau\right] \end{aligned} \tag{2.4}$$

Split up the time integration in (2.4) from $\int_t^{t_f}$ to $\int_t^{t'} + \int_{t'}^{t_f}$:

$$\begin{aligned} J(\boldsymbol{x}(t), t) = \min_{\boldsymbol{u}(t\to t_f)} \mathbb{E}_{\boldsymbol{x}(t)}\Bigg[&\Phi(\boldsymbol{x}_{t_f}) + \int_{t}^{t'}\left(\frac{1}{2}\boldsymbol{u}^\top \boldsymbol{R}\boldsymbol{u} + V(\boldsymbol{x}) + \boldsymbol{r}^\top(\boldsymbol{x})\boldsymbol{u}\right)\,\mathrm{d}\tau \\ &+ \int_{t'}^{t_f}\left(\frac{1}{2}\boldsymbol{u}^\top \boldsymbol{R}\boldsymbol{u} + V(\boldsymbol{x}) + \boldsymbol{r}^\top(\boldsymbol{x})\boldsymbol{u}\right)\,\mathrm{d}\tau\Bigg]. \end{aligned}$$

The first integration interval only depends on the input sequence $\boldsymbol{u}(t \to t')$, so the minimization operation can be split into

$$\begin{aligned} J(\boldsymbol{x}(t), t) = \min_{\boldsymbol{u}(t\to t')} \mathbb{E}_{\boldsymbol{x}(t)}\Bigg\{&\int_{t}^{t'}\left(\frac{1}{2}\boldsymbol{u}^\top \boldsymbol{R}\boldsymbol{u} + V(\boldsymbol{x}) + \boldsymbol{r}^\top(\boldsymbol{x})\boldsymbol{u}\right)\,\mathrm{d}\tau \\ &+ \min_{\boldsymbol{u}(t'\to t_f)} \mathbb{E}_{\boldsymbol{x}(t')}\left[\int_{t'}^{t_f}\left(\frac{1}{2}\boldsymbol{u}^\top \boldsymbol{R}\boldsymbol{u} + V(\boldsymbol{x}) + \boldsymbol{r}^\top(\boldsymbol{x})\boldsymbol{u}\right)\,\mathrm{d}\tau + \Phi(\boldsymbol{x}_{t_f})\right]\Bigg\}. \end{aligned}$$

The second minimization is still inside the first minimization (curly braces) because the state $\boldsymbol{x}(t')$ depends on the input sequence $\boldsymbol{u}(t \to t')$. By definition of the optimal cost-to-go, one obtains a recursive formula from principle of optimality [9]:

$$J(\boldsymbol{x}(t), t) = \min_{\boldsymbol{u}(t\to t')} \mathbb{E}_{\boldsymbol{x}(t)}\left[\int_{t}^{t'}\left(\frac{1}{2}\boldsymbol{u}^\top \boldsymbol{R}\boldsymbol{u} + V(\boldsymbol{x}) + \boldsymbol{r}^\top(\boldsymbol{x})\boldsymbol{u}\right)\,\mathrm{d}\tau + J(\boldsymbol{x}(t'), t')\right] \tag{2.5}$$

To transform the recursive formula into a differential equation, we start looking at infinitesimal time intervals and set $t' = t + \mathrm{d}t$. The Taylor expansion law (A.7) allows us to expand the optimal cost-to-go to first order in $\mathrm{d}t$ and second order in $\mathrm{d}x$. To simplify the notation, we denote $J(\boldsymbol{x}(t), t)$ as J:

$$\begin{aligned} &\mathbb{E}_{\boldsymbol{x}(t)}\left[J(\boldsymbol{x}(t'), t')\right] \\ :=&\,\mathbb{E}_{\boldsymbol{x}(t)}\left[J(\boldsymbol{x}(t + \mathrm{d}t), t + \mathrm{d}t)\right] \\ \approx&\,\mathbb{E}_{\boldsymbol{x}(t)}\left[J + (\partial_t J)\,\mathrm{d}t + (\partial_{\boldsymbol{x}} J)^\top\,\mathrm{d}\boldsymbol{x} + \frac{1}{2}\,\mathrm{d}\boldsymbol{x}^\top(\partial_{\boldsymbol{x}\boldsymbol{x}} J)\,\mathrm{d}\boldsymbol{x}\right] \\ =&\,\mathbb{E}_{\boldsymbol{x}(t)}\left[J + (\partial_t J)\,\mathrm{d}t + (\partial_{\boldsymbol{x}} J)^\top\,\mathrm{d}\boldsymbol{x} + \frac{1}{2}\,\mathtt{Tr}\left((\partial_{\boldsymbol{x}\boldsymbol{x}} J)\,\mathrm{d}\boldsymbol{x}\,\mathrm{d}\boldsymbol{x}^\top\right)\right] \end{aligned} \tag{2.6}$$

The last equality results from the property of inner product (A.1) as well as trace of matrix product (A.2), and symmetry of $\partial_{\boldsymbol{xx}} J$:

$$\frac{1}{2}\,\mathrm{d}\boldsymbol{x}^\top(\partial_{\boldsymbol{xx}}J)\,\mathrm{d}\boldsymbol{x} = \frac{1}{2}\,\mathtt{Tr}\left((\partial_{\boldsymbol{xx}}J)\,\mathrm{d}\boldsymbol{x}\,\mathrm{d}\boldsymbol{x}^\top\right)\,. \tag{2.7}$$

Note we can expand the term $\mathrm{d}\boldsymbol{x}\,\mathrm{d}\boldsymbol{x}^\top$ using the system dynamics (2.3):

$$\begin{aligned}\mathrm{d}\boldsymbol{x}\,\mathrm{d}\boldsymbol{x}^\top =\ & [\boldsymbol{f}\,\mathrm{d}t + \boldsymbol{G}(\boldsymbol{u}\,\mathrm{d}t + \boldsymbol{Q}\,\mathrm{d}\boldsymbol{w})]\,[\boldsymbol{f}\,\mathrm{d}t + \boldsymbol{G}(\boldsymbol{u}\,\mathrm{d}t + \boldsymbol{Q}\,\mathrm{d}\boldsymbol{w})]^\top \\ =\ & (\boldsymbol{f}+\boldsymbol{G}\boldsymbol{u})(\boldsymbol{f}+\boldsymbol{G}\boldsymbol{u})^\top\,\mathrm{d}t^2 + (\boldsymbol{f}+\boldsymbol{G}\boldsymbol{u})\,\mathrm{d}t\,\mathrm{d}\boldsymbol{w}^\top(\boldsymbol{G}\boldsymbol{Q})^\top \\ & + \boldsymbol{G}\boldsymbol{Q}\,\mathrm{d}\boldsymbol{w}(\boldsymbol{f}+\boldsymbol{G}\boldsymbol{u})^\top\,\mathrm{d}t + \boldsymbol{G}\boldsymbol{Q}\,\mathrm{d}\boldsymbol{w}\,\mathrm{d}\boldsymbol{w}^\top(\boldsymbol{G}\boldsymbol{Q})^\top\,. \end{aligned} \tag{2.8}$$

Since we assume $\mathrm{d}t$ to be an infinitesimal time interval, any terms above first order of $\mathrm{d}t$ can be ignored, which leads to

$$\mathrm{d}\boldsymbol{x}\,\mathrm{d}\boldsymbol{x}^\top \approx \boldsymbol{G}\boldsymbol{Q}\,\mathrm{d}\boldsymbol{w}\,\mathrm{d}\boldsymbol{w}^\top(\boldsymbol{G}\boldsymbol{Q})^\top\,. \tag{2.9}$$

Substitute the approximation (2.9) and system dynamics (2.3) into the expectation:

$$\begin{aligned}&\mathbb{E}_{\boldsymbol{x}(t)}\left[J(\boldsymbol{x}(t+\ \mathrm{d}t), t+\ \mathrm{d}t)\right] \\ =&\mathbb{E}_{\boldsymbol{x}(t)}\left[J + (\partial_t J)\,\mathrm{d}t + (\partial_{\boldsymbol{x}}J)^\top(\boldsymbol{f}+\boldsymbol{G}\boldsymbol{u})\,\mathrm{d}t + (\partial_{\boldsymbol{x}}J)^\top\boldsymbol{G}\boldsymbol{Q}\,\mathrm{d}\boldsymbol{w}\right. \\ &\left. + \frac{1}{2}\,\mathtt{Tr}\left((\partial_{\boldsymbol{xx}}J)\boldsymbol{G}\boldsymbol{Q}\,\mathrm{d}\boldsymbol{w}\,\mathrm{d}\boldsymbol{w}^\top(\boldsymbol{G}\boldsymbol{Q})^\top\right)\right] \end{aligned} \tag{2.10}$$

We know that $\mathbb{E}\left[\,\mathrm{d}\boldsymbol{w}\,\mathrm{d}\boldsymbol{w}^\top\right] = \mathbb{I}\,\mathrm{d}t$ and $\mathbb{E}\left[\,\mathrm{d}\boldsymbol{w}\right] = \boldsymbol{0}$ from the definition of Brownian motion, which leads to

$$\begin{aligned}&\mathbb{E}_{\boldsymbol{x}(t)}\left[J(\boldsymbol{x}(t+\ \mathrm{d}t), t+\ \mathrm{d}t)\right] \\ =\ & J + (\partial_t J)\,\mathrm{d}t + (\partial_{\boldsymbol{x}}J)^\top(\boldsymbol{f}+\boldsymbol{G}\boldsymbol{u})\,\mathrm{d}t + \frac{1}{2}\,\mathtt{Tr}((\partial_{\boldsymbol{xx}}J)\boldsymbol{G}\boldsymbol{Q}\boldsymbol{Q}^\top\boldsymbol{G}^\top)\,\mathrm{d}t\,. \end{aligned} \tag{2.11}$$

Regarding the first term of (2.5), the integral disappears in the limit $\mathrm{d}t \to 0$ and the expectation can be dropped because $\boldsymbol{x}(t)$ is the known starting state. The recursive formula (2.5) is turned into a deterministic equation:

$$\begin{aligned}J = \min_{\boldsymbol{u}(t)}\Bigg[&\left(\frac{1}{2}\boldsymbol{u}^\top\boldsymbol{R}\boldsymbol{u} + V(\boldsymbol{x}) + \boldsymbol{r}^\top(\boldsymbol{x})\boldsymbol{u}\right)\,\mathrm{d}t \\ &+ J + (\partial_t J)\,\mathrm{d}t + (\partial_{\boldsymbol{x}}J)^\top(\boldsymbol{f}+\boldsymbol{G}\boldsymbol{u})\,\mathrm{d}t + \frac{1}{2}\,\mathtt{Tr}((\partial_{\boldsymbol{xx}}J)\boldsymbol{G}\boldsymbol{Q}\boldsymbol{Q}^\top\boldsymbol{G}^\top)\,\mathrm{d}t\Bigg]\,. \end{aligned} \tag{2.12}$$

Divide both sides of (2.12) with $\mathrm{d}t$, and cancel J to obtain a stochastic HJB equation:

$$\begin{aligned}-(\partial_t J) = \min_{\boldsymbol{u}(t)}\Bigg[&\frac{1}{2}\boldsymbol{u}^\top\boldsymbol{R}\boldsymbol{u} + \boldsymbol{r}^\top(\boldsymbol{x})\boldsymbol{u} + (\partial_{\boldsymbol{x}}J)^\top(\boldsymbol{G}\boldsymbol{u}) \\ &+ V(\boldsymbol{x}) + (\partial_{\boldsymbol{x}}J)^\top\boldsymbol{f} + \frac{1}{2}\,\mathtt{Tr}((\partial_{\boldsymbol{xx}}J)\boldsymbol{G}\boldsymbol{Q}\boldsymbol{Q}^\top\boldsymbol{G}^\top)\Bigg] \end{aligned} \tag{2.13}$$

The optimal input sequence $\boldsymbol{u}$ of (2.13) can be found by setting the gradient of right-hand side to zero:

$$\nabla_{\boldsymbol{u}}\left(\frac{1}{2}\boldsymbol{u}^\top\boldsymbol{R}\boldsymbol{u} + \boldsymbol{r}^\top(\boldsymbol{x})\boldsymbol{u} + (\partial_{\boldsymbol{x}}J)^\top(\boldsymbol{G}\boldsymbol{u})\right) = 0\,, \tag{2.14}$$

which leads to the optimal control $\boldsymbol{u}^*$,

$$\boldsymbol{u}^* = -\boldsymbol{R}^{-1}\left(\boldsymbol{r}(\boldsymbol{x}) + \boldsymbol{G}^\top(\partial_{\boldsymbol{x}} J)\right) . \tag{2.15}$$

The optimal control (2.15) turns the stochastic HJB equation (2.13) into

$$\begin{aligned} -(\partial_t J) = &\left(\boldsymbol{r}(\boldsymbol{x}) + \boldsymbol{G}^\top(\partial_{\boldsymbol{x}} J)\right)^\top \left[-\frac{1}{2}\boldsymbol{R}^{-1}\right] \left(\boldsymbol{r}(\boldsymbol{x}) + \boldsymbol{G}^\top(\partial_{\boldsymbol{x}} J)\right) \\ &+ V(\boldsymbol{x}) + (\partial_{\boldsymbol{x}} J)^\top \boldsymbol{f} + \frac{1}{2}\,\mathtt{Tr}((\partial_{\boldsymbol{x}\boldsymbol{x}} J)\boldsymbol{G}\boldsymbol{Q}\boldsymbol{Q}^\top\boldsymbol{G}^\top) . \end{aligned} \tag{2.16}$$

However, in general there does not exist an analytic solution to the resulting nonlinear partial differential equation (2.16). Although numerical methods are developed to approximate the solution, the computational cost is too high and even suffers from the curse of dimensionality. Fortunately, for the system dynamics (2.3) where inputs enter the system affinely and white noise affects the system only through the input channel, we can apply logarithmic transformation to turn the nonlinear PDE into a linear one [10].

We now apply the log transformation (change of variable) for the optimal cost-to-go,

$$J(\boldsymbol{x}, t) = -\gamma \log \psi(\boldsymbol{x}, t) , \tag{2.17}$$

and we want to maximize the advantage function $\psi(\boldsymbol{x}, t)$.

The matrix derivation rules (A.3)-(A.6) lead to

$$\partial_{\boldsymbol{x}} J = -\gamma \frac{1}{\psi}\partial_{\boldsymbol{x}}\psi , \tag{2.18}$$

$$\partial_{\boldsymbol{x}\boldsymbol{x}} J = -\gamma\left(-\frac{1}{\psi^2}(\partial_{\boldsymbol{x}}\psi)(\partial_{\boldsymbol{x}}\psi)^\top + \frac{1}{\psi}\partial_{\boldsymbol{x}\boldsymbol{x}}\psi\right) , \tag{2.19}$$

$$\partial_t J = -\gamma \frac{1}{\psi}(\partial_t \psi) . \tag{2.20}$$

Plug (2.17)-(2.20) back into (2.16):

$$\begin{aligned} \gamma\frac{1}{\psi}(\partial_t\psi) = &-\frac{1}{2}\boldsymbol{r}^\top\boldsymbol{R}^{-1}\boldsymbol{r} + V(\boldsymbol{x}) - \frac{\gamma}{\psi}(\partial_{\boldsymbol{x}}\psi)^\top(\boldsymbol{f} - \boldsymbol{G}\boldsymbol{R}^{-1}\boldsymbol{r}) \\ &- \frac{1}{2}\frac{\gamma^2}{\psi^2}(\partial_{\boldsymbol{x}}\psi)^\top\boldsymbol{G}\boldsymbol{R}^{-1}\boldsymbol{G}^\top(\partial_{\boldsymbol{x}}\psi) \\ &+ \frac{1}{2}\,\mathtt{Tr}\left[-\gamma\left(-\frac{1}{\psi^2}(\partial_{\boldsymbol{x}}\psi)(\partial_{\boldsymbol{x}}\psi)^\top + \frac{1}{\psi}\partial_{\boldsymbol{x}\boldsymbol{x}}\psi\right)\boldsymbol{G}\boldsymbol{Q}\boldsymbol{Q}^\top\boldsymbol{G}^\top\right] . \end{aligned} \tag{2.21}$$

In order to eliminate the terms quadratic in $\partial_{\boldsymbol{x}}\psi$ such that the HJB equation becomes a linear partial differential equation, we choose γ and $\boldsymbol{R}$ to satisfy

$$-\gamma\boldsymbol{G}\boldsymbol{R}^{-1}\boldsymbol{G}^\top + \boldsymbol{G}\boldsymbol{Q}\boldsymbol{Q}^\top\boldsymbol{G}^\top = \boldsymbol{0} . \tag{2.22}$$

A sufficient condition for (2.22) is

$$\boldsymbol{Q}\boldsymbol{Q}^\top = \gamma\boldsymbol{R}^{-1} , \tag{2.23}$$

which requires the noise to be inversely proportional to the input cost weight.

The elaborately chosen γ and $\boldsymbol{R}$ turn the nonlinear one (2.21) into a linear PDE:

$$\begin{aligned} (\partial_t\psi) = &-(\partial_{\boldsymbol{x}}\psi)^\top(\boldsymbol{f} - \boldsymbol{G}\boldsymbol{R}^{-1}\boldsymbol{r}) + \frac{\psi}{\gamma}\left(-\frac{1}{2}\boldsymbol{r}^\top\boldsymbol{R}^{-1}\boldsymbol{r} + V(\boldsymbol{x})\right) \\ &+ \frac{1}{2}\,\mathtt{Tr}\left[-(\partial_{\boldsymbol{x}\boldsymbol{x}}\psi)\boldsymbol{G}\boldsymbol{Q}\boldsymbol{Q}^\top\boldsymbol{G}^\top\right] . \end{aligned} \tag{2.24}$$

2.3 Optimal cost-to-go and optimal control

Next, we show how to compute optimal cost-to-go and optimal control through sampling. According to the Feynman-Kac formula, the solution of the linear partial differential equation (2.24) can be obtained by computing an expectation over all trajectories starting from $\boldsymbol{x}$ at time t, governed by the diffusion:

$$\mathrm{d}\boldsymbol{x} = (\boldsymbol{f} - \boldsymbol{G}\boldsymbol{R}^{-1}\boldsymbol{r})\,\mathrm{d}t + \boldsymbol{G}\boldsymbol{Q}\,\mathrm{d}\boldsymbol{w}\,, \tag{2.25}$$

and the obtained solution is

$$\psi(\boldsymbol{x}, t) = \mathbb{E}_{\boldsymbol{x}(t)}\, e^{-\frac{1}{\gamma}\left(\int_t^{t_f}\left(-\frac{1}{2}\boldsymbol{r}^\top \boldsymbol{R}^{-1}\boldsymbol{r} + V(\boldsymbol{x})\right)\mathrm{d}\tau + \Phi(\boldsymbol{x}_{t_f})\right)}\,. \tag{2.26}$$

For a discrete-time simulation, we choose a *sufficiently small* time interval Δt and forward simulate paths according to

$$\boldsymbol{x}[k+1] = \boldsymbol{x}[k] + (\boldsymbol{f} - \boldsymbol{G}\boldsymbol{R}^{-1}\boldsymbol{r})\Delta t + \boldsymbol{G}\boldsymbol{Q}\Delta\boldsymbol{w}\,, \tag{2.27}$$

where all state-dependent quantities are evaluated at $\boldsymbol{x}[k]$. The noise increment is sampled from $\Delta\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \Delta t\mathbb{I})$.

After solving the advantage function $\psi(\boldsymbol{x}, t)$ through sampling, we can retrieve the optimal cost-to-go from (2.17).

Logarithmic transformation (2.17) and advantage function (2.26) allow us to compute the optimal cost-to-go for a given state $\boldsymbol{x}$ and time t. The optimal control $\boldsymbol{u}^*$ can be recovered via (2.15) which requires us to take the derivative of the optimal cost-to-go, i.e. $\partial_{\boldsymbol{x}} J$. Kappen and Ruiz [11], Buchli et al. [12] further showed that we can directly derive the optimal input $\boldsymbol{u}_s^*$ through sampling:

$$\begin{aligned}
\boldsymbol{u}_s^*(\boldsymbol{x}, t) &= \mathbb{E}_{\boldsymbol{x}(t)}\left\{\boldsymbol{\varepsilon}\frac{e^{-\frac{1}{\gamma}\left(\int_t^{t_f}\left(-\frac{1}{2}\boldsymbol{r}^\top \boldsymbol{R}^{-1}\boldsymbol{r} + V(\boldsymbol{x})\right)\mathrm{d}\tau + \Phi(\boldsymbol{x}_{t_f})\right)}}{\mathbb{E}_{\boldsymbol{x}(t)}\left\{e^{-\frac{1}{\gamma}\left(\int_t^{t_f}\left(-\frac{1}{2}\boldsymbol{r}^\top \boldsymbol{R}^{-1}\boldsymbol{r} + V(\boldsymbol{x})\right)\mathrm{d}\tau + \Phi(\boldsymbol{x}_{t_f})\right)}\right\}}\right\} \\
&= \frac{1}{\psi(\boldsymbol{x}, t)}\,\mathbb{E}_{\boldsymbol{x}(t)}\left\{\boldsymbol{\varepsilon}\, e^{-\frac{1}{\gamma}\left(\int_t^{t_f}\left(-\frac{1}{2}\boldsymbol{r}^\top \boldsymbol{R}^{-1}\boldsymbol{r} + V(\boldsymbol{x})\right)\mathrm{d}\tau + \Phi(\boldsymbol{x}_{t_f})\right)}\right\}
\end{aligned} \tag{2.28}$$

where $\boldsymbol{\varepsilon} \sim \mathcal{N}(\boldsymbol{0}, \Delta t\boldsymbol{Q}\boldsymbol{Q}^\top)$ was the noisy input on the system at the first time step.

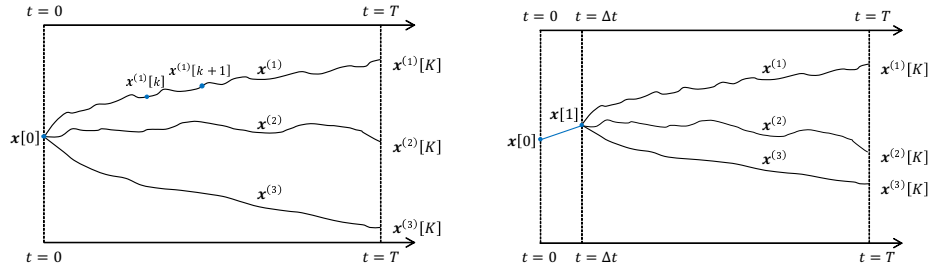
The optimal input obtained through sampling (2.28) can be interpreted as a form of soft-max operation on the negative trajectory cost with γ as the temperature.

As other methods in model predict control framework, we also compute a sequence of inputs at current time step and update the sequence at the next time step when new measurement available, shown in Figure 2.1. Since $\mathrm{d}\boldsymbol{w}$ in the stochastic dynamics (2.3) is zero-mean noise[2], we use the deterministic dynamics (2.1) to approximately predict the next state,

$$\hat{\boldsymbol{x}}_{k+1} = \hat{\boldsymbol{x}}_k + \boldsymbol{f}\left(\hat{\boldsymbol{x}}_k, k\right)\Delta t + \boldsymbol{G}\left(\hat{\boldsymbol{x}}_k\right)\boldsymbol{u}_s^*\left(\hat{\boldsymbol{x}}_k, k\right)\Delta t\,, \tag{2.29}$$

Note the difference in notation here: predicted state is denoted as $\hat{\boldsymbol{x}}_k$ instead of $\boldsymbol{x}[k]$, which is the actual state. We assume the initial state is known, $\hat{\boldsymbol{x}}_0 = \boldsymbol{x}[0]$.

[2] In case of nonzero-mean noise, we can write it as the sum of its mean and a zero-mean noise with the same variance.



(a) Rollouts at current time step (b) Rollouts at next time step

Figure 2.1: Schematic of model predictive control framework. The superscript stands for the sample number and $\boldsymbol{x}[k]$ is the actual state at time step k. For example, $\boldsymbol{x}^{(n)}[k]$ means the state of the n^{th} sample at time $t = k\Delta t$.

Once we have a prediction for the next state $\hat{\boldsymbol{x}}_{n+1}$, we can compute a new optimal control $\hat{\boldsymbol{u}}^*_{k+1}(\hat{\boldsymbol{x}}_{k+1}, k+1)$ with (2.28). We can therefore recursively compute a sequence of optimal control input and a predicted trajectory. Since we approximate the next state by ignoring the noise coming into the input channel, the trajectory predicted at current time step will drift away from the actual one as time goes on. But for the next a few states, the prediction is a valid approximation.

Chapter 3

Constrained path integral control with augmented Lagrangian

3.1 Problem description

In Chapter 2, we reviewed how path integral algorithm solves unconstrained stochastic control problems. In this chapter, we move on to consider stochastic systems with state-input inequality constraints. The optimization problem (2.13) should be solved while constraints are respected:

$$-(\partial_t J) = \min_{\substack{\boldsymbol{u}(t) \\ \text{s.t. (3.2)}}} \Bigg[\frac{1}{2}\boldsymbol{u}^\top \boldsymbol{R}\boldsymbol{u} + \boldsymbol{r}^\top(\boldsymbol{x})\boldsymbol{u} + (\partial_{\boldsymbol{x}} J)^\top (\boldsymbol{G}\boldsymbol{u}) + V(\boldsymbol{x}) + (\partial_{\boldsymbol{x}} J)^\top \boldsymbol{f} + \frac{1}{2}\,\mathtt{Tr}\left((\partial_{\boldsymbol{x}\boldsymbol{x}} J)\boldsymbol{G}\boldsymbol{Q}\boldsymbol{Q}^\top \boldsymbol{G}^\top\right) \Bigg], \tag{3.1}$$

where the constraints are a state-input inequality

$$\boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u}) = \boldsymbol{e}(\boldsymbol{x}) + \boldsymbol{F}(\boldsymbol{x})\boldsymbol{u} \geq \boldsymbol{0}\,. \tag{3.2}$$

3.2 Optimal control and Lagrangian multipliers

According to the unconstrained formulation presented in [13], the two optimization problems (3.1) and (3.3) are equivalent.

$$\begin{aligned} -(\partial_t J) = \min_{\boldsymbol{u}(t)} \max_{\boldsymbol{\lambda}(t) \geq \boldsymbol{0}} \Bigg[& \frac{1}{2}\boldsymbol{u}^\top \boldsymbol{R}\boldsymbol{u} + \boldsymbol{r}^\top(\boldsymbol{x})\boldsymbol{u} + (\partial_{\boldsymbol{x}} J)^\top (\boldsymbol{G}\boldsymbol{u}) - \boldsymbol{\lambda}^\top \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u}) \\ & + V(\boldsymbol{x}) + (\partial_{\boldsymbol{x}} J)^\top \boldsymbol{f} + \frac{1}{2}\,\mathtt{Tr}\left((\partial_{\boldsymbol{x}\boldsymbol{x}} J)\boldsymbol{G}\boldsymbol{Q}\boldsymbol{Q}^\top \boldsymbol{G}^\top\right) \Bigg] \end{aligned} \tag{3.3}$$

Next, want to swap the *min* and *max* operators per the von Neumann's minimax theorem [14] so that we can optimize over unconstrained input $\boldsymbol{u}$ first:

Theorem 1 (Minimax Theorem) *Let $\mathcal{X} \subset \mathbb{R}^n$ and $\mathcal{Y} \subset \mathbb{R}^m$ be compact convex sets. If $f : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ is a continuous function that is concave-convex, i.e., $f(\cdot, \boldsymbol{y}) : \mathcal{X} \to \mathbb{R}$ is concave for fixed $\boldsymbol{y}$, and $f(\boldsymbol{x}, \cdot) : \mathcal{Y} \to \mathbb{R}$ is convex for fixed $\boldsymbol{x}$, then we have that*

$$\max_{\boldsymbol{x} \in \mathcal{X}} \min_{\boldsymbol{y} \in \mathcal{Y}} f(\boldsymbol{x}, \boldsymbol{y}) = \min_{\boldsymbol{y} \in \mathcal{Y}} \max_{\boldsymbol{x} \in \mathcal{X}} f(\boldsymbol{x}, \boldsymbol{y})$$

Although in our case feasible input set $\mathcal{U} = \mathbb{R}^{n_u}$ and feasible Lagrangian multiplier set $\mathcal{A} = \mathbb{R}_+^{n_\lambda}$ are not compact (they are unbounded), we can artificially construct bounds to make the sets large enough so that the optimal solutions can be obtained without explicitly putting constraints on the optimization variables. With such construction, we can swap the *min* and *max* operators in optimization problem (3.3):

$$-(\partial_t J) = \max_{\boldsymbol{\lambda}(t)\geq \mathbf{0}} \min_{\boldsymbol{u}(t)} \Big[\frac{1}{2}\boldsymbol{u}^\top \boldsymbol{R}\boldsymbol{u} + \boldsymbol{r}^\top(\boldsymbol{x})\boldsymbol{u} + (\partial_{\boldsymbol{x}} J)^\top(\boldsymbol{G}\boldsymbol{u}) - \boldsymbol{\lambda}^\top \boldsymbol{g}(\boldsymbol{x},\boldsymbol{u}) + V(\boldsymbol{x}) + (\partial_{\boldsymbol{x}} J)^\top \boldsymbol{f} + \frac{1}{2}\,\mathtt{Tr}\left((\partial_{\boldsymbol{x}\boldsymbol{x}} J)\boldsymbol{G}\boldsymbol{Q}\boldsymbol{Q}^\top \boldsymbol{G}^\top\right) \Big] . \quad (3.4)$$

Set the gradient of the object function with respect to $\boldsymbol{u}$ to 0, i.e.,

$$\nabla_{\boldsymbol{u}} \left(\frac{1}{2}\boldsymbol{u}^\top \boldsymbol{R}\boldsymbol{u} + \boldsymbol{r}^\top \boldsymbol{u} + (\partial_{\boldsymbol{x}} J)^\top(\boldsymbol{G}\boldsymbol{u}) - \boldsymbol{\lambda}^\top \boldsymbol{g}(\boldsymbol{x},\boldsymbol{u}) \right) = \mathbf{0} . \quad (3.5)$$

The optimal control input can be easily computed by

$$\boldsymbol{u}^* = -\boldsymbol{R}^{-1}\left(\boldsymbol{r} + \boldsymbol{G}^\top \partial_{\boldsymbol{x}} J - \boldsymbol{F}^\top \boldsymbol{\lambda}\right) . \quad (3.6)$$

Then, we plug the optimal control (3.6) back into (3.4):

$$-(\partial_t J) = \max_{\boldsymbol{\lambda}(t)\geq \mathbf{0}} \Big[-\frac{1}{2}\big(\boldsymbol{r} + \boldsymbol{G}^\top \partial_{\boldsymbol{x}} J - \boldsymbol{F}^\top \boldsymbol{\lambda}\big)^\top \boldsymbol{R}^{-1}\left(\boldsymbol{r} + \boldsymbol{G}^\top \partial_{\boldsymbol{x}} J - \boldsymbol{F}^\top \boldsymbol{\lambda}\right) - \boldsymbol{\lambda}^\top \boldsymbol{e} + V(\boldsymbol{x}) + (\partial_{\boldsymbol{x}} J)^\top \boldsymbol{f} + \frac{1}{2}\,\mathtt{Tr}\left((\partial_{\boldsymbol{x}\boldsymbol{x}} J)\boldsymbol{G}\boldsymbol{Q}\boldsymbol{Q}^\top \boldsymbol{G}^\top\right) \Big] . \quad (3.7)$$

Rearrange terms into the standard quadratic form in $\boldsymbol{\lambda}$:

$$-(\partial_t J) = \max_{\boldsymbol{\lambda}(t)\geq \mathbf{0}} \Big[-\frac{1}{2}\boldsymbol{\lambda}^\top \boldsymbol{F}\boldsymbol{R}^{-1}\boldsymbol{F}^\top \boldsymbol{\lambda} - \boldsymbol{\lambda}^\top\left(\boldsymbol{e} - \boldsymbol{F}\boldsymbol{R}^{-1}\left(\boldsymbol{r} + \boldsymbol{G}^\top \partial_{\boldsymbol{x}} J\right)\right) - \frac{1}{2}\big(\boldsymbol{r} + \boldsymbol{G}^\top \partial_{\boldsymbol{x}} J\big)^\top \boldsymbol{R}^{-1}\left(\boldsymbol{r} + \boldsymbol{G}^\top \partial_{\boldsymbol{x}} J\right) + V(\boldsymbol{x}) + (\partial_{\boldsymbol{x}} J)^\top \boldsymbol{f} + \frac{1}{2}\,\mathtt{Tr}\left((\partial_{\boldsymbol{x}\boldsymbol{x}} J)\boldsymbol{G}\boldsymbol{Q}\boldsymbol{Q}^\top \boldsymbol{G}^\top\right) \Big] . \quad (3.8)$$

Let $\boldsymbol{A}$ and $\boldsymbol{b}$ denote the quadratic and linear term coefficients of $\boldsymbol{\lambda}$:

$$\boldsymbol{A} = \boldsymbol{F}\boldsymbol{R}^{-1}\boldsymbol{F}^\top , \quad (3.9)$$

$$\boldsymbol{b} = \boldsymbol{e} - \boldsymbol{F}\boldsymbol{R}^{-1}\left(\boldsymbol{r} + \boldsymbol{G}^\top \partial_{\boldsymbol{x}} J\right) , \quad (3.10)$$

and note that $\boldsymbol{A}$ is symmetric, and positive definite given a row-full-rank $\boldsymbol{F}$.

Now, the optimization problem can be rewrote as

$$-(\partial_t J) = \max_{\boldsymbol{\lambda}(t)\geq \mathbf{0}} \Big[-\frac{1}{2}\boldsymbol{\lambda}^\top \boldsymbol{A}\boldsymbol{\lambda} - \boldsymbol{\lambda}^\top \boldsymbol{b} - \frac{1}{2}\big(\boldsymbol{r} + \boldsymbol{G}^\top \partial_{\boldsymbol{x}} J\big)^\top \boldsymbol{R}^{-1}\left(\boldsymbol{r} + \boldsymbol{G}^\top \partial_{\boldsymbol{x}} J\right) + V(\boldsymbol{x}) + (\partial_{\boldsymbol{x}} J)^\top \boldsymbol{f} + \frac{1}{2}\,\mathtt{Tr}\left((\partial_{\boldsymbol{x}\boldsymbol{x}} J)\boldsymbol{G}\boldsymbol{Q}\boldsymbol{Q}^\top \boldsymbol{G}^\top\right) \Big] . \quad (3.11)$$

Take terms independent of $\boldsymbol{\lambda}$ out of the *max* operator and note $\boldsymbol{\lambda}^\top \boldsymbol{b}$ is a scalar:

$$-(\partial_t J) = \max_{\boldsymbol{\lambda}(t)\geq \mathbf{0}} \Big[-\frac{1}{2}\boldsymbol{\lambda}^\top \boldsymbol{A}\boldsymbol{\lambda} - \boldsymbol{b}^\top \boldsymbol{\lambda} \Big] - \frac{1}{2}\big(\boldsymbol{r} + \boldsymbol{G}^\top \partial_{\boldsymbol{x}} J\big)^\top \boldsymbol{R}^{-1}\left(\boldsymbol{r} + \boldsymbol{G}^\top \partial_{\boldsymbol{x}} J\right) + V(\boldsymbol{x}) + (\partial_{\boldsymbol{x}} J)^\top \boldsymbol{f} + \frac{1}{2}\,\mathtt{Tr}\left((\partial_{\boldsymbol{x}\boldsymbol{x}} J)\boldsymbol{G}\boldsymbol{Q}\boldsymbol{Q}^\top \boldsymbol{G}^\top\right) , \quad (3.12)$$

which is equivalent to

$$\begin{aligned}-(\partial_t J) = &- \min_{\boldsymbol{\lambda}(t)\geq \mathbf{0}} \left[\frac{1}{2}\boldsymbol{\lambda}^\top \boldsymbol{A}\boldsymbol{\lambda} + \boldsymbol{b}^\top \boldsymbol{\lambda}\right] \\ &- \frac{1}{2}\left(\boldsymbol{r} + \boldsymbol{G}^\top \partial_{\boldsymbol{x}} J\right)^\top \boldsymbol{R}^{-1}\left(\boldsymbol{r} + \boldsymbol{G}^\top \partial_{\boldsymbol{x}} J\right) \\ &+ V(\boldsymbol{x}) + (\partial_{\boldsymbol{x}} J)^\top \boldsymbol{f} + \frac{1}{2}\,\mathtt{Tr}\left((\partial_{\boldsymbol{x}\boldsymbol{x}} J)\boldsymbol{G}\boldsymbol{Q}\boldsymbol{Q}^\top \boldsymbol{G}^\top\right) . \end{aligned} \tag{3.13}$$

In order to efficiently update $\boldsymbol{\lambda}$ without solving an expensive stochastic HJB equation, we require $\boldsymbol{A}$ to be a diagonal matrix, i.e.,

$$\boldsymbol{A} = \begin{bmatrix} a_1 & & \\ & \ddots & \\ & & a_n \end{bmatrix} , \tag{3.14}$$

otherwise there is no way to decouple $\boldsymbol{\lambda}$. In case of a diagonal matrix $\boldsymbol{A}$, the Lagrangian multiplier $\boldsymbol{\lambda}$ can be computed element-wise,

$$\lambda_i^* = \max\left\{-\frac{b_i}{a_i}, 0\right\} . \tag{3.15}$$

We define the *active operator* $\mathbb{1}(x)$ to simplify further notation

$$\mathbb{1}(x) = \begin{cases} 1 & x > 0 \\ 0 & x \leq 0 \end{cases} . \tag{3.16}$$

The element-wise optimal values of the minimization part can be computed with multiplicative update introduced in [15]:

$$\min_{\boldsymbol{\lambda}(t)\geq \mathbf{0}} \left[\frac{1}{2}\boldsymbol{\lambda}^\top \boldsymbol{A}\boldsymbol{\lambda} + \boldsymbol{b}^\top \boldsymbol{\lambda}\right] = \frac{1}{2}\begin{bmatrix}\mathbb{1}(\lambda_1^{k*}) & \cdots & \mathbb{1}(\lambda_n^{k*})\end{bmatrix} \begin{bmatrix} -\frac{b_1^2}{a_1} \\ \vdots \\ -\frac{b_n^2}{a_n}\end{bmatrix} , \tag{3.17}$$

with non-negative optimizer

$$\boldsymbol{\lambda}^{k*} = \begin{bmatrix}\lambda_1^{k*} \\ \vdots \\ \lambda_n^{k*}\end{bmatrix} = \arg\min_{\boldsymbol{\lambda}(t)\geq \mathbf{0}} \left[\frac{1}{2}\boldsymbol{\lambda}^\top \boldsymbol{A}\boldsymbol{\lambda} + \boldsymbol{b}^\top \boldsymbol{\lambda}\right] = \begin{bmatrix}\max\left\{-\frac{b_1}{a_1}, 0\right\} \\ \vdots \\ \max\left\{-\frac{b_n}{a_n}, 0\right\}\end{bmatrix} . \tag{3.18}$$

Next, we express the optimal value in matrix form,

$$\begin{aligned} \min_{\boldsymbol{\lambda}(t)\geq \mathbf{0}} \left[\frac{1}{2}\boldsymbol{\lambda}^\top \boldsymbol{A}\boldsymbol{\lambda} + \boldsymbol{b}^\top \boldsymbol{\lambda}\right] &= \frac{1}{2}\begin{bmatrix}\mathbb{1}(\lambda_1^{k}) & \cdots & \mathbb{1}(\lambda_n^{k})\end{bmatrix} \begin{bmatrix} -\frac{b_1^2}{a_1} \\ \vdots \\ -\frac{b_n^2}{a_n}\end{bmatrix} \\ &= \frac{1}{2}\boldsymbol{b}^\top \begin{bmatrix} -\frac{\mathbb{1}(\lambda_1^k)}{a_1} & & \\ & \ddots & \\ & & -\frac{\mathbb{1}(\lambda_n^k)}{a_n}\end{bmatrix} \boldsymbol{b} \\ &= -\frac{1}{2}\boldsymbol{b}^\top \begin{bmatrix} \mathbb{1}(\lambda_1^k) & & \\ & \ddots & \\ & & \mathbb{1}(\lambda_n^k)\end{bmatrix} \boldsymbol{A}^{-1}\boldsymbol{b} \\ &\coloneqq -\frac{1}{2}\boldsymbol{b}^\top \mathbb{1}(\boldsymbol{\lambda}^k)\boldsymbol{A}^{-1}\boldsymbol{b} . \end{aligned} \tag{3.19}$$

To summarize,

$$\max_{\boldsymbol{\lambda}(t)\geq\mathbf{0}}\left[-\boldsymbol{\lambda}^\top\boldsymbol{b}-\frac{1}{2}\boldsymbol{\lambda}^\top\boldsymbol{A}\boldsymbol{\lambda}\right]=\frac{1}{2}\boldsymbol{b}^\top\mathbb{1}(\boldsymbol{\lambda}^k)\boldsymbol{A}^{-1}\boldsymbol{b}\,. \tag{3.20}$$

Plug (3.20) back into the HJB equation (3.13):

$$\begin{aligned}-(\partial_t J)=&\frac{1}{2}\boldsymbol{b}^\top\mathbb{1}(\boldsymbol{\lambda}^k)\boldsymbol{A}^{-1}\boldsymbol{b}\\&-\frac{1}{2}\left(\boldsymbol{r}+\boldsymbol{G}^\top\partial_{\boldsymbol{x}}J\right)^\top\boldsymbol{R}^{-1}\left(\boldsymbol{r}+\boldsymbol{G}^\top\partial_{\boldsymbol{x}}J\right)\\&+V(\boldsymbol{x})+(\partial_{\boldsymbol{x}}J)^\top\boldsymbol{f}+\frac{1}{2}\,\mathtt{Tr}\left((\partial_{\boldsymbol{x}\boldsymbol{x}}J)\boldsymbol{G}\boldsymbol{Q}\boldsymbol{Q}^\top\boldsymbol{G}^\top\right)\,.\end{aligned} \tag{3.21}$$

Expand $\boldsymbol{A}$ and $\boldsymbol{b}$ in (3.21), and rearrange terms associated with $\partial_{\boldsymbol{x}}J$:

$$\begin{aligned}-(\partial_t J)=&\frac{1}{2}\left(\left[\boldsymbol{e}-\boldsymbol{F}\boldsymbol{R}^{-1}\boldsymbol{r}\right]^\top\mathbb{1}(\boldsymbol{\lambda}^k)(\boldsymbol{F}\boldsymbol{R}^{-1}\boldsymbol{F}^\top)^{-1}\left[\boldsymbol{e}-\boldsymbol{F}\boldsymbol{R}^{-1}\boldsymbol{r}\right]-\boldsymbol{r}^\top\boldsymbol{R}^{-1}\boldsymbol{r}\right)\\&+(\partial_{\boldsymbol{x}}J)^\top\left(\left[-\boldsymbol{F}\boldsymbol{R}^{-1}\boldsymbol{G}^\top\right]^\top\mathbb{1}(\boldsymbol{\lambda}^k)(\boldsymbol{F}\boldsymbol{R}^{-1}\boldsymbol{F}^\top)^{-1}\left[\boldsymbol{e}-\boldsymbol{F}\boldsymbol{R}^{-1}\boldsymbol{r}\right]\right)\\&+(\partial_{\boldsymbol{x}}J)^\top\left(\boldsymbol{f}-\boldsymbol{G}\boldsymbol{R}^{-1}\boldsymbol{r}\right)\\&+\frac{1}{2}(\partial_{\boldsymbol{x}}J)^\top\left(\left[\boldsymbol{F}\boldsymbol{R}^{-1}\boldsymbol{G}^\top\right]^\top\mathbb{1}(\boldsymbol{\lambda}^k)(\boldsymbol{F}\boldsymbol{R}^{-1}\boldsymbol{F}^\top)^{-1}\left[\boldsymbol{F}\boldsymbol{R}^{-1}\boldsymbol{G}^\top\right]\right)(\partial_{\boldsymbol{x}}J)\\&-\frac{1}{2}(\partial_{\boldsymbol{x}}J)^\top\left(\boldsymbol{G}\boldsymbol{R}^{-1}\boldsymbol{G}^\top\right)(\partial_{\boldsymbol{x}}J)\\&+V(\boldsymbol{x})+\frac{1}{2}\,\mathtt{Tr}\left((\partial_{\boldsymbol{x}\boldsymbol{x}}J)\boldsymbol{G}\boldsymbol{Q}\boldsymbol{Q}^\top\boldsymbol{G}^\top\right)\,.\end{aligned} \tag{3.22}$$

We introduce the left pseudo-inverse of $\boldsymbol{F}$ weighted on $\boldsymbol{R}^{-1}$ for simplification:

$$\boldsymbol{F}^\dagger=\boldsymbol{R}^{-1}\boldsymbol{F}^\top\left(\boldsymbol{F}\boldsymbol{R}^{-1}\boldsymbol{F}^\top\right)^{-1}\,. \tag{3.23}$$

Now, the PDE (3.22) can be simplified into

$$\begin{aligned}-(\partial_t J)=&\frac{1}{2}\left(\left[\boldsymbol{e}-\boldsymbol{F}\boldsymbol{R}^{-1}\boldsymbol{r}\right]^\top\mathbb{1}(\boldsymbol{\lambda}^k)\left[\left(\boldsymbol{F}\boldsymbol{R}^{-1}\boldsymbol{F}^\top\right)^{-1}\boldsymbol{e}-{\boldsymbol{F}^\dagger}^\top\boldsymbol{r}\right]-\boldsymbol{r}^\top\boldsymbol{R}^{-1}\boldsymbol{r}\right)\\&+(\partial_{\boldsymbol{x}}J)^\top\left(\left[-\boldsymbol{F}\boldsymbol{R}^{-1}\boldsymbol{G}^\top\right]^\top\mathbb{1}(\boldsymbol{\lambda}^k)\left[\left(\boldsymbol{F}\boldsymbol{R}^{-1}\boldsymbol{F}^\top\right)^{-1}\boldsymbol{e}-{\boldsymbol{F}^\dagger}^\top\boldsymbol{r}\right]+\boldsymbol{f}-\boldsymbol{G}\boldsymbol{R}^{-1}\boldsymbol{r}\right)\\&+\frac{1}{2}(\partial_{\boldsymbol{x}}J)^\top\left(\left[\boldsymbol{F}\boldsymbol{R}^{-1}\boldsymbol{G}^\top\right]^\top\mathbb{1}(\boldsymbol{\lambda}^k){\boldsymbol{F}^\dagger}^\top\boldsymbol{G}^\top-\boldsymbol{G}\boldsymbol{R}^{-1}\boldsymbol{G}^\top\right)(\partial_{\boldsymbol{x}}J)\\&+V(\boldsymbol{x})+\frac{1}{2}\,\mathtt{Tr}\left((\partial_{\boldsymbol{x}\boldsymbol{x}}J)\boldsymbol{G}\boldsymbol{Q}\boldsymbol{Q}^\top\boldsymbol{G}^\top\right)\,.\end{aligned} \tag{3.24}$$

The quadratic terms in $\partial_{\boldsymbol{x}}\psi$ would disappear under the condition:

$$\begin{aligned}0&=\gamma\left(\left[\boldsymbol{F}\boldsymbol{R}^{-1}\boldsymbol{G}^\top\right]^\top\mathbb{1}(\boldsymbol{\lambda}^k){\boldsymbol{F}^\dagger}^\top\boldsymbol{G}^\top-\boldsymbol{G}\boldsymbol{R}^{-1}\boldsymbol{G}^\top\right)+\boldsymbol{G}\boldsymbol{Q}\boldsymbol{Q}^\top\boldsymbol{G}^\top\\&=\gamma\left(\boldsymbol{G}\boldsymbol{R}^{-1}\boldsymbol{F}^\top\mathbb{1}(\boldsymbol{\lambda}^k){\boldsymbol{F}^\dagger}^\top\boldsymbol{G}^\top-\boldsymbol{G}\boldsymbol{R}^{-1}\boldsymbol{G}^\top\right)+\boldsymbol{G}\boldsymbol{Q}\boldsymbol{Q}^\top\boldsymbol{G}^\top\\&=\boldsymbol{G}\left[\gamma\boldsymbol{R}^{-1}\left(\boldsymbol{F}^\top\mathbb{1}(\boldsymbol{\lambda}^k){\boldsymbol{F}^\dagger}^\top-\mathbb{I}\right)+\boldsymbol{Q}\boldsymbol{Q}^\top\right]\boldsymbol{G}^\top\,.\end{aligned} \tag{3.25}$$

One sufficient condition for (3.25) to hold is

$$\begin{aligned}\boldsymbol{Q}\boldsymbol{Q}^\top=\left(\boldsymbol{Q}\boldsymbol{Q}^\top\right)^\top&=\gamma\boldsymbol{R}^{-1}\left[\mathbb{I}-\boldsymbol{F}^\top\mathbb{1}(\boldsymbol{\lambda}^k){\boldsymbol{F}^\dagger}^\top\right]\\&=\gamma\left[\mathbb{I}-\boldsymbol{F}^\dagger\mathbb{1}(\boldsymbol{\lambda}^k)\boldsymbol{F}\right]\boldsymbol{R}^{-1}\,.\end{aligned} \tag{3.26}$$

Interpretation: If all constraints are inactive at the optimum, the optimal solution can also be obtained by solving the unconstrained problem. We can verify this argument by setting $\boldsymbol{\lambda}^k$ to $\mathbf{0}$ in the sufficient condition (3.26) and observe that it is now exactly the same as the sufficient condition (2.23) in the unconstrained case. On the other hand, if all constraints are active at the optimizer, we can equivalently derive the optimal solution by solving state-input-equality constrained problem, subject to

$$\boldsymbol{h}(\boldsymbol{x}, \boldsymbol{u}) = \boldsymbol{e}(\boldsymbol{x}) + \boldsymbol{F}(\boldsymbol{x})\boldsymbol{u} = \mathbf{0} \,, \tag{3.27}$$

and the sufficient condition would be

$$\boldsymbol{Q}\boldsymbol{Q}^\top = \gamma(\boldsymbol{I} - \boldsymbol{F}^\dagger \boldsymbol{F})\boldsymbol{R}^{-1} \,. \tag{3.28}$$

The linearized and log-transformed HJB equation then reads

$$\begin{aligned}(\partial_t \psi) = \frac{\psi}{2\gamma} \Big(\left[\boldsymbol{e} - \boldsymbol{F}\boldsymbol{R}^{-1}\boldsymbol{r}\right]^\top \mathbb{1}(\boldsymbol{\lambda}^k) \left[\left(\boldsymbol{F}\boldsymbol{R}^{-1}\boldsymbol{F}^\top\right)^{-1} \boldsymbol{e} - {\boldsymbol{F}^\dagger}^\top \boldsymbol{r} \right] - \boldsymbol{r}^\top \boldsymbol{R}^{-1} \boldsymbol{r} + V(\boldsymbol{x}) \Big) \\ - (\partial_{\boldsymbol{x}} \psi)^\top \left(\left[-\boldsymbol{F}\boldsymbol{R}^{-1}\boldsymbol{G}^\top\right]^\top \mathbb{1}(\boldsymbol{\lambda}^k) \left[\left(\boldsymbol{F}\boldsymbol{R}^{-1}\boldsymbol{F}^\top\right)^{-1} \boldsymbol{e} - {\boldsymbol{F}^\dagger}^\top \boldsymbol{r} \right] + \boldsymbol{f} - \boldsymbol{G}\boldsymbol{R}^{-1}\boldsymbol{r} \right) \\ + \frac{1}{2} \mathtt{Tr}\left[-(\partial_{\boldsymbol{x}\boldsymbol{x}} \psi) \boldsymbol{G}\boldsymbol{Q}\boldsymbol{Q}^\top \boldsymbol{G}^\top \right] \,. \end{aligned} \tag{3.29}$$

The forward simulating process should respect the stochastic dynamics

$$\begin{aligned} \mathrm{d}\boldsymbol{x} = \left(\left[-\boldsymbol{F}\boldsymbol{R}^{-1}\boldsymbol{G}^\top\right]^\top \mathbb{1}(\boldsymbol{\lambda}^k) \left[\left(\boldsymbol{F}\boldsymbol{R}^{-1}\boldsymbol{F}^\top\right)^{-1} \boldsymbol{e} - {\boldsymbol{F}^\dagger}^\top \boldsymbol{r} \right] + \boldsymbol{f} - \boldsymbol{G}\boldsymbol{R}^{-1}\boldsymbol{r} \right) \mathrm{d}t \\ + \boldsymbol{G}\boldsymbol{Q}\, \mathrm{d}\boldsymbol{w} \end{aligned} \tag{3.30}$$

Since the optimal multiplier ${\boldsymbol{\lambda}^k}^*$ and the optimal cost-to-go J at time step k are coupled, we must know one of them to compute the other one. Practically, we assume the active constraints at the preceding time step $k-1$ remain the same for the current time step k, and approximate ${\boldsymbol{\lambda}^k}^*$ with ${\boldsymbol{\lambda}^{k-1}}^*$. Then we solve the optimal cost-to-go J and update ${\boldsymbol{\lambda}^k}^*$.

Note that the sampling dynamics (3.30) requires ${\boldsymbol{\lambda}^{k+1}}^*, \ldots, {\boldsymbol{\lambda}^N}^*$ in the future. Remember how we look into the future and predict the future optimal control in (2.29). Similarly, we predict the optimal Lagrangian multipliers and use them in the sampling dynamics.

As we already have a reasonable approximation of optimal Lagrangian multiplier sequence, we can compute the optimal cost-to-go $J(\boldsymbol{x}, k)$ by (2.26) and the optimal control input $\boldsymbol{u}_s^*(\boldsymbol{x}, k)$ by (2.28). Once we have the optimal cost-to-go J, we can work out $\boldsymbol{b}$ from (3.10) by taking derivative of J and then update the optimal Lagrangian multiplier ${\boldsymbol{\lambda}^k}^*$ from (3.18).

Consequently, an optimal trajectory will be achieved if we proceed recursively.

Chapter 4

Practical solution to constrained path integral problem

Besides the theoretical solution we introduced in Chapter 3, we can also respect the constraints with an elaborately designed sampling policy. In this chapter, three sampling policies are discussed and compared. With such sampling policies, we can directly compute a constraint-consistent control input as presented in Chapter 2 without explicitly dealing with the constraints.

4.1 Optimal control revisit

Before introducing the sampling policies, we rewrite the optimal control $\boldsymbol{u}_s^*$ (2.28) in unconstrained cases as a convex combination of the sampled noise inputs,

$$\boldsymbol{u}_s^* = \sum_{n=1}^{N} \alpha_1 \boldsymbol{\varepsilon}_1 + \cdots + \alpha_N \boldsymbol{\varepsilon}_N \ , \tag{4.1}$$

where

$$\alpha_n = \frac{e^{-\frac{1}{\gamma}\left(\int_t^{t_f}\left(-\frac{1}{2}\boldsymbol{r}^\top \boldsymbol{R}^{-1}\boldsymbol{r} + V(\boldsymbol{x})\right)\mathrm{d}\tau + \Phi(\boldsymbol{x}_{t_f})\right)}}{\sum_{n=1}^{N} e^{-\frac{1}{\gamma}\left(\int_t^{t_f}\left(-\frac{1}{2}\boldsymbol{r}^\top \boldsymbol{R}^{-1}\boldsymbol{r} + V(\boldsymbol{x})\right)\mathrm{d}\tau + \Phi(\boldsymbol{x}_{t_f})\right)}} \in [0,1] \tag{4.2}$$

is the weight of the sampled noise input $\boldsymbol{\varepsilon}_n$.

Note that $\alpha_1, \ldots, \alpha_N$ sum up to 1, i.e.,

$$\sum_{n=1}^{N} \alpha_n = 1 \ . \tag{4.3}$$

Now, we consider the feasible input set $\mathcal{U}(\boldsymbol{x})$ of state $\boldsymbol{x}$,

$$\mathcal{U}(\boldsymbol{x}) = \{\boldsymbol{u} \in \mathbb{R}^{n_u} \mid \boldsymbol{e}(\boldsymbol{x}) + \boldsymbol{F}(\boldsymbol{x})\boldsymbol{u} \geq \boldsymbol{0}\} \ . \tag{4.4}$$

Per Theorem 2 [16] and (4.1)-(4.3), we know that if $\boldsymbol{\varepsilon}_1, \ldots, \boldsymbol{\varepsilon}_N \in \mathcal{U}$, the convex combination $\boldsymbol{u}_s^*$ also belongs to $\mathcal{U}$ since $\mathcal{U}$ defines a convex set (polyhedron).

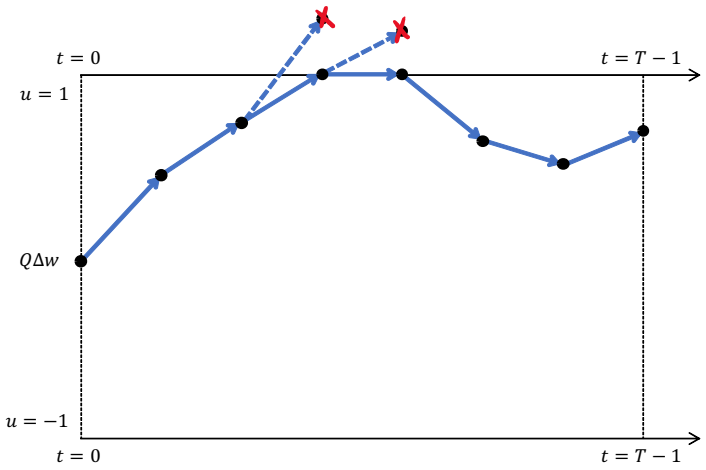


Figure 4.1: Projection sampling scheme

Theorem 2 (Convex Combination) *Given N points $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_N$ in a convex set $\mathcal{U}$, and N non-negative numbers $\alpha_1, \ldots, \alpha_N$ such that $\alpha_1 + \cdots + \alpha_N = 1$, the affine combination*

$$\sum_{n=1}^{N} \alpha_k \boldsymbol{u}_n$$

belongs to $\mathcal{U}$.

4.2 Sampling policies

From the last section, we know that if all sampled noise inputs are constraint-consistent, the resulting optimal control will satisfy all constraints automatically. This fact leads an alternative way to solve the constrained problem, that is, force all sampled noise inputs to stay in the feasible region, which can be achieved by elaborately designing a sampling policy. Next, we will introduce and discuss three sampling policy candidates.

4.2.1 Projection sampling

As shown in Figure 4.1, we first generate an unconstrained sample and determine whether constraints are satisfied. If it is constraint-consistent, we keep this sample. Otherwise, we project this infeasible sample onto the boundary of the feasible region $\mathcal{U}$ by solving the QP problem (4.5). The basic idea is that the projection should not result in too much deviation from the originally sampled noise input, and we can use quadratic programming to find the closest feasible point to replace the infeasible one:

$$\begin{aligned} \min_{\boldsymbol{u}} \ & \frac{1}{2}(\boldsymbol{u} - \boldsymbol{u}_s)^\top \boldsymbol{R}(\boldsymbol{u} - \boldsymbol{u}_s) \\ \text{s.t. } & \boldsymbol{e}(\boldsymbol{x}) + \boldsymbol{F}(\boldsymbol{x})\boldsymbol{u} \geq \boldsymbol{0} \ , \end{aligned} \tag{4.5}$$

where $\boldsymbol{u}_s$ stands for the infeasible sample generated originally.

We chose $\boldsymbol{R}$ as weight because costly inputs should move less and we want high sensitivity in the direction of important inputs. However, the weight $\boldsymbol{R}$ is not unique. One can choose other weights to meet specific criteria.

Reformulate the problem in the standard form of quadratic programming:

$$\begin{aligned} \min_{\boldsymbol{u}} \ & \frac{1}{2}\boldsymbol{u}^\top \boldsymbol{R}\boldsymbol{u} - \boldsymbol{u}_s^\top \boldsymbol{R}\boldsymbol{u} + \frac{1}{2}\boldsymbol{u}_s^\top \boldsymbol{R}\boldsymbol{u}_s \\ \text{s.t. } & \boldsymbol{e}(\boldsymbol{x}) + \boldsymbol{F}(\boldsymbol{x})\boldsymbol{u} \geq \boldsymbol{0} \ . \end{aligned} \tag{4.6}$$

Algorithm 1 Projection sampling

$\boldsymbol{x} \leftarrow$ current state
$\boldsymbol{u}_s \sim \mathcal{N}\left(\boldsymbol{0}, \Delta t \boldsymbol{Q}\boldsymbol{Q}^\top\right)$
if $\boldsymbol{e}(\boldsymbol{x}) + \boldsymbol{F}(\boldsymbol{x})\boldsymbol{u}_s \geq \boldsymbol{0}$ **then**
  return $\boldsymbol{u}_s$
else
  $\boldsymbol{u} \leftarrow$ solution of (4.6)
end if
return $\boldsymbol{u}$

Advantages

- Since the limits are the best we can do, projection sampling performs very well under certain conditions when we would like the system to go all out. With stringent constraints, projection sampling may give us a higher chance to find a feasible path.

Disadvantages

- The projection sampling is biased since most samples accumulate on the boundaries of the feasible region so that the system is likely to operate around the limits. In the existence of noise, the actuation is prone to saturation, and safety issues and performance failure may occur.
- The projection sampling policy also comes with expensive computation since we need to solve an optimization problem for each infeasible sample at each time step. Even worse, the cost grows quickly with the dimensionality.
- Moreover, the projection sampling does not respect "draw samples from Gaussian distribution" well, which may cause the algorithm to fail.

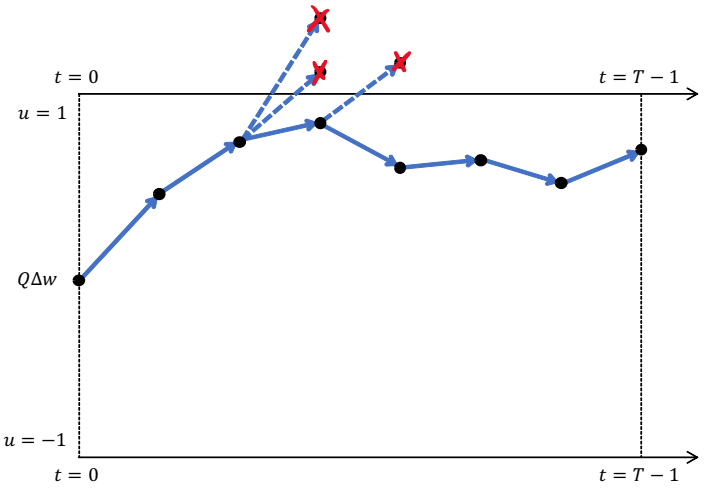


Figure 4.2: Rejection sampling scheme

4.2.2 Rejection sampling

In rejection sampling, we first generate unconstrained samples and only accept feasible ones as shown in Figure 4.2. For infeasible samples, we keep resampling them until they satisfy all constraints. The computational cost depends on the constraints and variance. For example, if $x \sim \mathcal{N}(0, \sigma^2)$ is constrained in $[-\sigma, \sigma]$, the acceptance rate will be approximately 68.3%, which leads to the high efficiency of rejection sampling. However, if the constraints are stringent, it will be much harder to generate a feasible sample. Think about an extreme example, where $0 \leq x \leq 0$ comes as constraints. The algorithm will never find a feasible sample since it is probabilistically impossible.

Algorithm 2 Rejection sampling

$N \leftarrow$ maximum iteration
$\boldsymbol{x} \leftarrow$ current state
$i \leftarrow 1$
while $i \leq N$ **do**
  $\boldsymbol{u}^{(i)} \sim \mathcal{N}\left(\mathbf{0}, \Delta t \boldsymbol{Q}\boldsymbol{Q}^\top\right)$
  if $\boldsymbol{e}(\boldsymbol{x}) + \boldsymbol{F}(\boldsymbol{x})\boldsymbol{u}^{(i)} \geq \mathbf{0}$ **then**
    return $\boldsymbol{u}^{(i)}$
  else
    $i \leftarrow i + 1$
  end if
end while
$\boldsymbol{u}_{\text{proj}} \leftarrow$ projection of $\boldsymbol{u}^{(N)}$ in $\boldsymbol{e}(\boldsymbol{x}) + \boldsymbol{F}(\boldsymbol{x})\boldsymbol{u} \geq \mathbf{0}$
return $\boldsymbol{u}_{\text{proj}}$

Advantages

- Theoretically, rejection sampling can handle any kind of constraints except for a singleton feasible set[1].
- In addition, rejection sampling can be implemented extremely easily.

Disadvantages

- When it comes to a high dimension or stringent constraints, the probability of getting a constraint-consistent sample quickly drops and we may only get out of the loop after generating a huge number of samples.

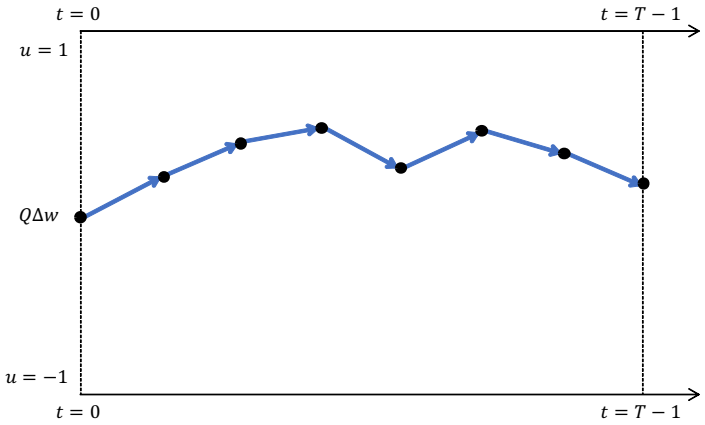
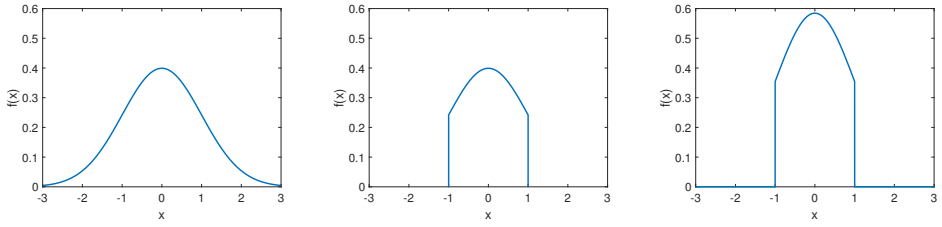


Figure 4.3: Truncated Gaussian sampling scheme



(a) (b) (c)

Figure 4.4: Truncated Gaussian distribution. (a) shows the PDF of a standard Gaussian distribution. (b) visualizes the result after the truncation. Note that it is not a valid PDF since the integration does not equal 1. (c) displays the valid PDF of truncated Gussian after normalizing the remaining part in (b).

[1] We can modify the algorithm to deal with singleton properly: Let the algorithm first determine whether the feasible set is a singleton. If so, return this value and stop.

4.2.3 Truncated Gaussian sampling

Truncated Gaussian distribution is derived from the Gaussian distribution by truncating "tails" of the PDF curve as shown in Figure (4.4). Suppose $x \sim \mathcal{N}(0, \sigma^2)$ lies in $[a, b]$. Then we set the probability of being outside this interval to zero, which makes sure that every generated sample satisfies the bound constraints, and normalize the probability of being inside. The truncated probability density function $m(x)$ can be analytically expressed as

$$m(x; \mu, \sigma, a, b) = \begin{cases} \frac{p\left(\frac{x-\mu}{\sigma}\right)}{\sigma\left(F\left(\frac{b-\mu}{\sigma}\right) - F\left(\frac{a-\mu}{\sigma}\right)\right)} & a \leq u \leq b \\ 0 & \text{otherwise} \end{cases} \tag{4.7}$$

where $p(x)$ and $F(x)$ are the PDF and CDF of the standard Gaussian distribution,

$$p(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2} , \tag{4.8}$$

$$F(x) = \frac{1}{2}\left(1 + \operatorname{erf}\left(\frac{x}{\sqrt{2}}\right)\right) , \tag{4.9}$$

and $\operatorname{erf}(x)$ is the error function,

$$\operatorname{erf}(x) = \frac{1}{\sqrt{\pi}} \int_{-x}^{x} e^{-s^2} \, \mathrm{d}s . \tag{4.10}$$

We now compute the cumulative distribution function $M(x)$ of the truncated Gaussian distribution for $x \in [a, b]$,

$$\begin{aligned} M(x; \mu, \sigma, a, b) &= \int_a^x m(s) \, \mathrm{d}s \\ &= \frac{1}{\sigma\left(F\left(\frac{b-\mu}{\sigma}\right) - F\left(\frac{a-\mu}{\sigma}\right)\right)} \int_a^x p\left(\frac{s-\mu}{\sigma}\right) \, \mathrm{d}s \\ &= \frac{1}{F\left(\frac{b-\mu}{\sigma}\right) - F\left(\frac{a-\mu}{\sigma}\right)} \int_a^x p\left(\frac{s-\mu}{\sigma}\right) \, \mathrm{d}\left(\frac{s-\mu}{\sigma}\right) \\ &= \frac{1}{F\left(\frac{b-\mu}{\sigma}\right) - F\left(\frac{a-\mu}{\sigma}\right)} \int_{\frac{a-\mu}{\sigma}}^{\frac{x-\mu}{\sigma}} p(y) \, \mathrm{d}y \\ &= \frac{F\left(\frac{x-\mu}{\sigma}\right) - F\left(\frac{a-\mu}{\sigma}\right)}{F\left(\frac{b-\mu}{\sigma}\right) - F\left(\frac{a-\mu}{\sigma}\right)} . \end{aligned} \tag{4.11}$$

Therefore, $M(x)$ for $x \in \mathbb{R}$ is

$$M(x; \mu, \sigma, a, b) = \begin{cases} 0 & x < a \\ \frac{F\left(\frac{x-\mu}{\sigma}\right) - F\left(\frac{a-\mu}{\sigma}\right)}{F\left(\frac{b-\mu}{\sigma}\right) - F\left(\frac{a-\mu}{\sigma}\right)} & a \leq x \leq b \\ 1 & x > b \end{cases} \tag{4.12}$$

Next, we introduce how to draw samples from the truncated Gaussian distribution.

1. generate a random number ξ from the uniform distribution over $[0, 1]$
2. let $\bar{x}$ be the solution to

$$M(\bar{x}; \mu, \sigma, a, b) = \xi \tag{4.13}$$

3. Plug (4.12) into (4.13),

$$F\left(\frac{\bar{x}-\mu}{\sigma}\right) = F\left(\frac{a-\mu}{\sigma}\right) + \xi\left(F\left(\frac{b-\mu}{\sigma}\right) - F\left(\frac{a-\mu}{\sigma}\right)\right) \tag{4.14}$$

4. Solve the equation by inverting the error function,

$$\bar{x} = \sqrt{2}\sigma\,\mathrm{erf}^{-1}\left\{2\left[F\left(\frac{a-\mu}{\sigma}\right) + \xi\left(F\left(\frac{b-\mu}{\sigma}\right) - F\left(\frac{a-\mu}{\sigma}\right)\right)\right] - 1\right\} + \mu$$

5. $\bar{x}$ is the sample we draw from the truncated Gaussian distribution

As shown above, sampling from uni-variable truncated Gaussian distribution can be implemented very efficiently. However, in real systems, the noise inputs are often multi-channel and dependent. In this case, the constraints cannot be explicitly expressed as a bound on the random variable. We have to find a way decouple the inputs so that we can break down the multi-variable truncated Gaussian sampling into several uni-variable truncated Gaussian sampling. Let's consider the state-input inequality constraints,

$$\boldsymbol{e} + \boldsymbol{F}\boldsymbol{u} \geq \boldsymbol{0}, \tag{4.15}$$

where sampled noise inputs $\boldsymbol{u}$ are normally distributed,

$$\boldsymbol{u} \sim \mathcal{N}(\boldsymbol{0}, \Delta t \boldsymbol{Q}\boldsymbol{Q}^\top)\ . \tag{4.16}$$

We define an auxiliary variable $\boldsymbol{v}$ which satisfies,

$$\boldsymbol{u} = \sqrt{\Delta t}\boldsymbol{Q}\boldsymbol{v}\ , \tag{4.17}$$

where $\boldsymbol{v}$ has a standard Gaussian distribution $\boldsymbol{v} \sim \mathcal{N}(\boldsymbol{0}, \mathbb{I})$.

The original constraints on $\boldsymbol{u}$ are equivalent to the following constraints on $\boldsymbol{v}$,

$$\boldsymbol{e} + \sqrt{\Delta t}\boldsymbol{F}\boldsymbol{Q}\boldsymbol{v} \leq \boldsymbol{0}\ . \tag{4.18}$$

If we can express (4.18) as bound constraints on $v_1, \ldots, v_{n_u}$ separately, we can directly draw samples $v_{s,1}, \ldots, v_{s,n_u}$ channel-by-channel from uni-variable truncated Gaussian distribution.

Finally, the original samples $\boldsymbol{u}_s$ can be easily recovered by

$$\boldsymbol{u}_s = \sqrt{\Delta t}\boldsymbol{Q}\boldsymbol{v}_s\ . \tag{4.19}$$

Advantages

- Truncated Gaussian sampling is computationally more attractive because we do not need to solve a costly optimization or sample multiple times to achieve a feasible sample.
- Samples are computed with a closed-from function, which dramatically reduces the computational cost.
- Each sample generated with truncated Gaussian sampling will satisfy all constraints by definition.
- Samples respect Gaussian distribution.
- The computational cost grows only linearly with dimensionality.

Disadvantages

- Truncated Gaussian sampling works only for limited kinds of constraints. It particularly cannot handle coupled constraints on the noise inputs.

4.3 Summary

In this chapter, we introduced projection sampling, rejection sampling and truncated Gaussian sampling, and explained how they work in detail. However, it is impossible to tell which sampling policy of three outperforms others. They have different advantages and disadvantages. One must delicately consider the actual problem and accordingly choose one sampling policy or a mixture of them. For example, one can apply rejection sampling first and switch to projection sampling or truncated Gaussian sampling if no feasible sample are found during 10 iterations.

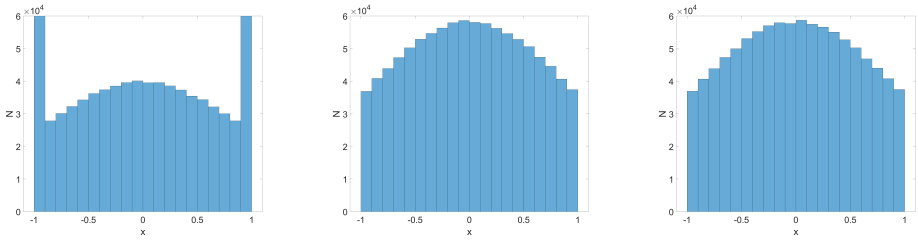
Chapter 5

Numerical examples

In this chapter, we start with a comparison of different sampling policies. Then, we consider a simple 1-DoF double slit example and illustrate the theoretical work presented in previous chapters. Important concepts in path integral, such as symmetry breaking and delayed choice, will be covered. Next, we move on to consider a more difficult case where the system has two states.

5.1 Sampling policies

To compare different sampling policies, we consider a standard Gaussian distribution and constrain it in $[-1, 1]$. Sample distribution is plotted in Figure 5.1.



(a) Projection sampling (b) Rejection sampling (c) Truncated Gaussian

Figure 5.1: Sampling policy comparison

In projection sampling, lots of samples accumulate on the boundaries but those inside the boundaries are normally distributed. Note that Figure 5.1 (a) has been scaled so that the vertical axis range equals the other two. Actually, the two bars on the boundaries are far higher than they look, with an order of 2×10^5. Rejection sampling and truncated Gaussian sampling are equivalent in the sense of probability distribution but yield different computational cost. Since we only need to sample once in truncated Gaussian sampling, we finally choose this method.

5.2 1-DoF simulation

In this section, we consider a simple 1-DoF example where the robot only has a scalar state x.

5.2.1 Problem settings

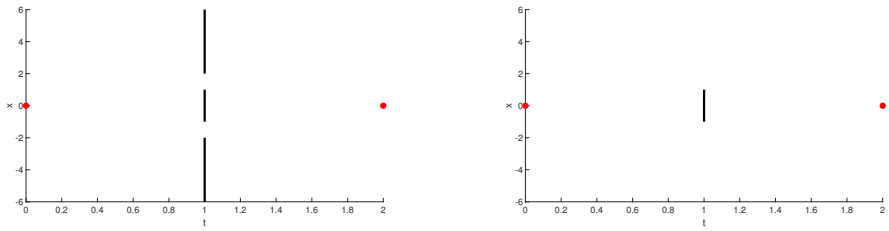
Consider a double-slit problem where the robot moves with constant horizontal velocity from $t = 0$ to $t = 2$. The input and noise affect the robot in x direction with stochastic dynamics,

$$\mathrm{d}x = u\,\mathrm{d}t + \mathrm{d}w\ . \tag{5.1}$$

The cost function is the sum of $V(x,t)$ and $\Phi(x_{t_f})$,

$$\begin{aligned} V(x,t) &= \begin{cases} \infty & x \notin [-2,-1] \cup [1,2] \text{ and } t = 1 \\ 0 & \text{otherwise} \end{cases}\ , \\ \Phi(x_{t_f}) &= \frac{1}{2} x_{t_f}^2\ , \end{aligned} \tag{5.2}$$

where $V(x,t)$ indicates whether the trajectory crashes into the barrier and $\Phi(x_{t_f})$ is the terminal cost.



(a) Double slit example (b) Simplified example

Figure 5.2: 1-DoF example scheme

The problem is illustrated in Figure 5.2 (a). The robot initially starts at the left red point and plans go to the target, marked by the right point. A wall with two traversable slits is placed in between. The slits are positioned at $[-2,-1]$ and $[1,2]$, and the wall blocks any other way at $t = 1$ except the two slits. The time interval $\mathrm{d}t = 0.02$. The robot moves with a constant velocity in t direction and the input and noise affect it in x direction. The standard deviation $q = 1$ and input cost weight $r = 0.1$. We choose $\gamma = 0.1$ according to the sufficient condition (2.23).

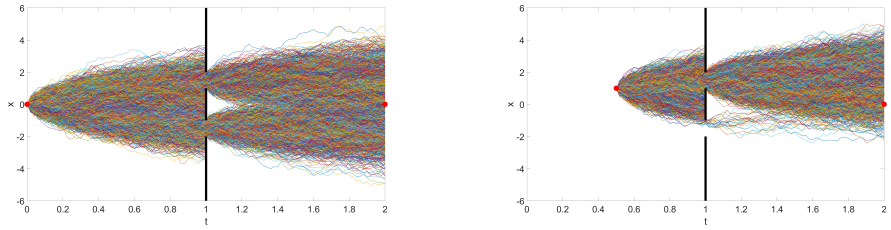
5.2.2 Symmetry breaking and delayed decision

We start with an unconstrained simulation. According to the uncontrolled sampling dynamics (2.27), we forward simulate samples with,

$$x[n+1] = x[n] + \Delta w\ , \tag{5.3}$$

where $\Delta w \sim \mathcal{N}(0, \Delta t)$.

The visualization of samples driven only by the noise is plotted in Figure 5.3 (a). The initial state $x_{t_0} = 0$ and no constraints are imposed on the system. The sampler



(a) At centerline (b) Off centerline

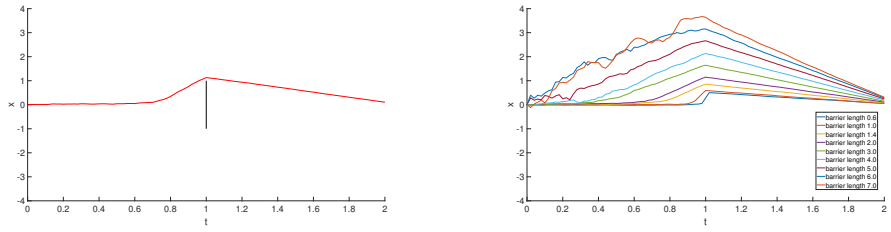
Figure 5.3: Visualization of samples

uses 10,000 trajectories for each x and 2,701 trajectories successfully go through the slits. There are almost the same number of paths go through the upper slit (1,346) and lower slits (1,355) since the initial state is placed on the centerline and the obstacle is symmetric for it. In this case, the optimal control, which is the weighted average, will thus be almost 0. The system will just go straight and delay the action.

However, if the robot is off the centerline, there will be significantly more samples go through one slit (preferred slit) than the other, and thus the optimal control will drive the system to go through the preferred slit. In Figure 5.3 (b), samples start from the left red point $x = 1$, $t = 0.5$. Since the starting point is above the centerline, it is easier for a noise-driven path to go through the upper slit. As a result, there are a total of 4,242 trajectories successfully pass the barrier, and most of them choose the upper slit. Symmetry breaking occurs and the robot chooses a preferred slit and take the corresponding action.

The timing of symmetry breaking closely depends on the barrier length. To better illustrate this, we consider a simplified setting by keeping only the barrier in the middle as shown in Figure 5.2 (b).

The actual paths of the robot are plotted in Figure 5.4. The simulation well demonstrates the delayed decision phenomenon, that is, the robot does not take any action until it is close to the barrier. A longer barrier forces the robot to make decision earlier. If the barrier is long enough, actions must be taken at the very beginning.



(a) Single actual path (b) Multiple actual paths

Figure 5.4: Illustration of delayed decision. (a) shows the actual path resulting from the path integral control, with barrier length $L = 2$. (b) plots the actual paths corresponding to different barrier length. To make comparison easier, barriers are omitted and paths go through the lower slit are mirrored to the upper part.

5.2.3 Discussion

Safety distance

Let's take a closer look at Figure 5.4 (a). The safety distance between the robot and the barrier guarantees that the robot can pass robustly in the existence of noise. Path integral achieves the safety distance automatically because the sampled trajectories which crash into the barrier have zero weights. The optimal control, as a result of sampled feasible trajectories, pushes the robot away from the barrier if it is too close.

Fluctuation

In Figure 5.4 (b), we can observe smooth actual paths corresponding to short barriers since there are plenty of sampled trajectories pass the barrier. However, obvious fluctuation occurs for long barriers because only a few sampled trajectories are feasible and each have a great impact on the optimal control. For example, assume the current state is above the centerline. Theoretically, more sampled trajectories go over the barrier and the system should also go this way. Since the barrier is long and only several trajectories can pass, it can happen that, for example, 3 trajectories go over the barrier but with high terminal cost and only 1 below but with low terminal cost. The controller considers terminal costs as weights and may choose the lower path. The fluctuation also results from the free control inputs since the only goal of the controller is coming back to 0 and it does care which way to go. If we include a properly-chosen cost on inputs, the fluctuation can be alleviated.

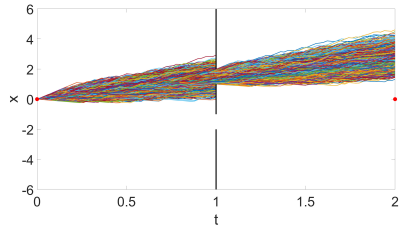
Choice of time interval

In simulation, we have another interesting finding. If we choose a extremely short barrier, for example, $L = 0.1$, the actual path will crash into the barrier finally. Because even at the proceeding time step ($t = 0.98$ for the example above), the symmetry breaking has not happened yet and system continues to go straight forward. Counter-intuitively, with more samples, the robot is more likely to hit the barrier. To explain this, assume we have infinitely many samples. Due to the nature of Gaussian sampling, there should be the *same* number of trajectories go over and below the barrier, and the optimal control remains 0 for all the time.

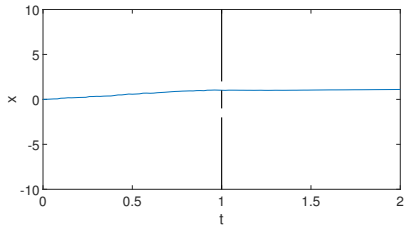
Another way to interpret, remember that we assumed $\mathrm{d}t \to 0$ when deriving the optimality condition. In this 1-DoF simulation, we chose time interval $\Delta t = 0.02$, which is not relatively small enough compared to the barrier length. In order to make path integral get back to work, we must shorten the time interval to stay consistent to the previous assumption $\mathrm{d}t \to 0$.

5.2.4 Constraint satisfaction

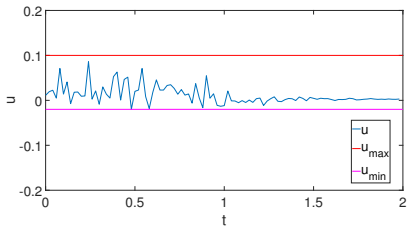
Next we consider an asymmetric bound constraint on the input $-0.02 \leq u \leq 0.10$. Figure 5.5 (a)-(c) show respectively the rolled out samples from the initial state $x = 0$, the actual path and the control sequence. Note that the sampling policy used here is truncated Gaussian sampling. Because of the asymmetric bound constraints, all feasible trajectories from the initial state go through the upper slit. Due to the tight low bound, the robot cannot return to $x_{t_f} = 0$ after passing the barrier. Figure 5.5 (c) and (d) compare the input sequence in constrained and unconstrained cases. Without dealing with the constraints in (d), the input goes below the lower bound while the input always stays in the feasible region by using a sampling policies introduced in Section 4.2.



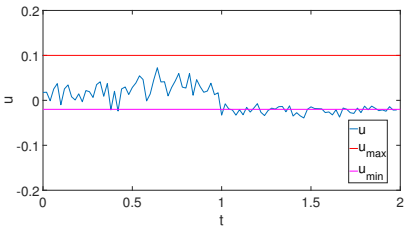
(a) Sample visualization



(b) Actual path



(c) Constrained control sequence



(d) Unconstrained control sequence

Figure 5.5: Constrained path integral results

5.3 2-DoF simulation

In this section, we move on to consider a more difficult case where the robot has two states x and z.

5.3.1 Problem settings

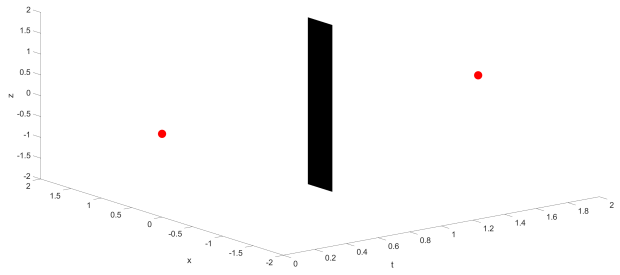


Figure 5.6: 2-DoF example scheme

Consider a 2-DoF problem shown in Figure 5.6 where the robot moves with a constant velocity in the t direction from $t = 0$ to $t = 2$. The input and noise affect the robot in x and z direction with stochastic dynamics,

$$\mathrm{d}\boldsymbol{x} = \boldsymbol{u}\,\mathrm{d}t + \mathrm{d}\boldsymbol{w}\ . \tag{5.4}$$

The cost function is the sum of $V(\boldsymbol{x}, t)$ and $\Phi(\boldsymbol{x}_{t_f})$,

$$V(\boldsymbol{x}, t) = \begin{cases} \infty & x \in [-0.2, 0.2] \text{ and } z \in [-1, 1] \text{ and } t = 1 \\ 0 & \text{otherwise}\ , \end{cases} \tag{5.5}$$

$$\Phi(\boldsymbol{x}_{t_f}) = \frac{1}{2}\left(x_{t_f}^2 + z_{t_f}^2\right)\ , \tag{5.6}$$

where $V(\boldsymbol{x}, t)$ indicates whether the trajectory crashes into the barrier and $\Phi(\boldsymbol{x}_{t_f})$ is the terminal cost.

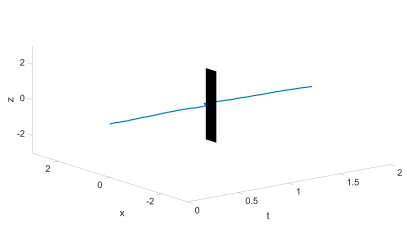
The robot initially stays at the left red point and plans go to the target, marked by the right point. A rectangular obstacle, centered at $[0, 0]$ with side length 0.4 and 2, stays in between. The time interval $\mathrm{d}t = 0.02$. The standard deviation $Q = \mathbb{I}$ and input weight $R = 0.1\mathbb{I}$. We choose $\gamma = 0.1$ according to the sufficient condition (2.23).

Once again, we start with an unconstrained simulation and compare it with the constrained case. We forward simulate samples with uncontrolled sampling dynamics,

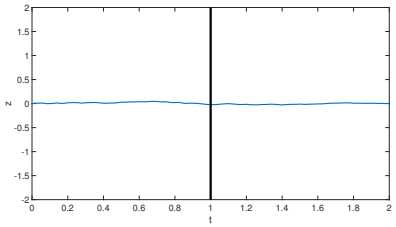
$$\boldsymbol{x}[n+1] = \boldsymbol{x}[n] + \Delta \boldsymbol{w} \ , \tag{5.7}$$

where $\Delta \boldsymbol{w} \sim \mathcal{N}(0, \Delta t \mathbb{I})$.

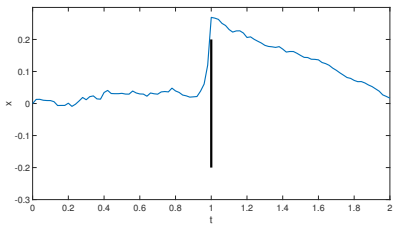
In Figure 5.7 are plotted the results of unconstrained path integral control. From the side view, we clearly observe the robot simply circumvents the barrier horizontally almost without any vertical movement. Figure 5.7 (c) also shows delayed decision behavior. Figure 5.7 (d) displays the control input sequence, which will be compared with Figure 5.8 (d).



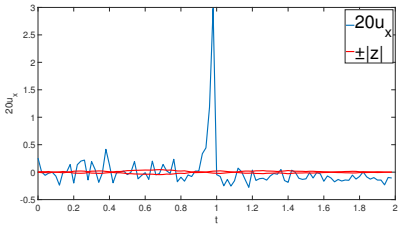
(a) Illustrative diagram



(b) Side view



(c) Top view



(d) Unconstrained control sequence

Figure 5.7: Results of Unconstrained case

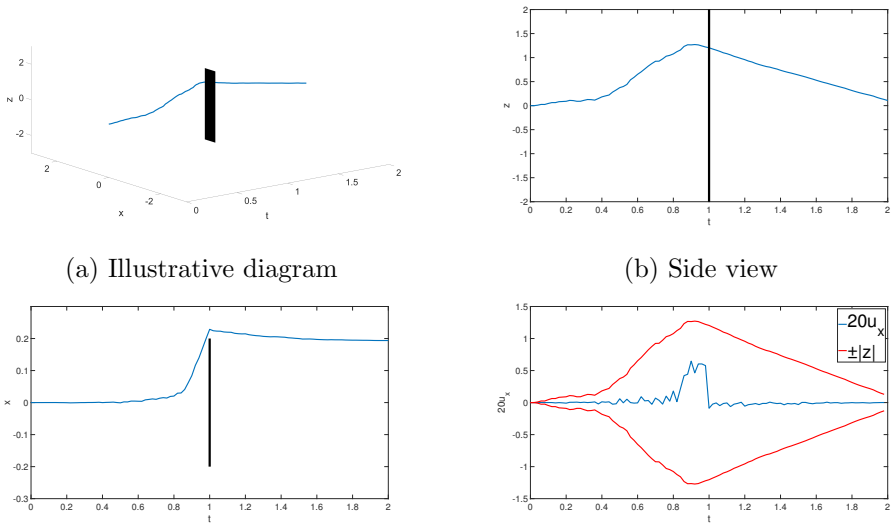
5.3.2 Constraint satisfaction

Now, we require the robot to pass the barrier with enough height. This additional requirement can be modeled as a state-input constraint,

$$|u_x| \leq 0.05\,|z| \ , \tag{5.8}$$

which can be interpreted as, the robot needs to gain enough height to steer.

Results of constrained case are plotted in Figure 5.8. Note that the sampling policy used here is truncated Gaussian sampling. From the illustrative diagram and side view, we note that the robot first goes up, passes the barrier and finally comes back down. We can also observe the delayed decision phenomenon in the side view and top view. Figure 5.8 (d) shows that the constraints are satisfied all the time and the actuation peak corresponds to the sharp turn in (c).



(a) Illustrative diagram

(b) Side view

(c) Top view

(d) Unconstrained control sequence

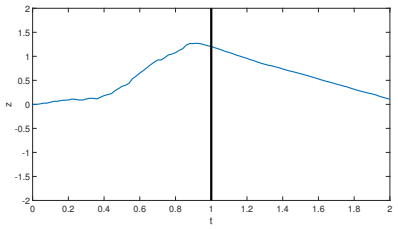
Figure 5.8: Results of constrained case

Note that the terminal state in Figure 5.8, $x_{t_f} = 0.19$ and $z_{t_f} = 0.1$, are not very close to the target $x^*_{t_f} = 0$ and $z^*_{t_f} = 0$, which is the best the controller can do. x does not come back to 0 because z is much greater than x, so the controller prefers to go down with all efforts to quickly reduce the cost.

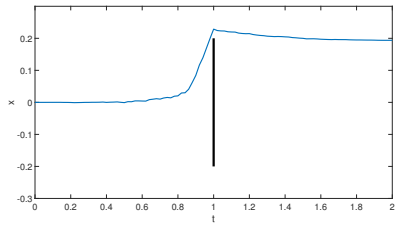
If we want x to come back closer to 0, we can redesign the cost function by increasing the cost weight on x, for example,

$$\Phi(\boldsymbol{x}_{t_f}) = \frac{1}{2}\left(100x^2_{t_f} + z^2_{t_f}\right) \ . \tag{5.9}$$

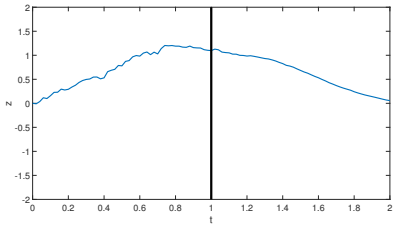
New side view and top view are compared with original ones in Figure 5.9. This time, terminal state, $x_{t_f} = 0.02$ and $z_{t_f} = 0.08$, returns much closer to the origin.



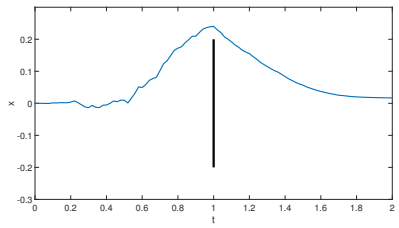
(a) Side view ($\boldsymbol{P} = \text{diag}(1, 1)$)



(b) Top view ($\boldsymbol{P} = \text{diag}(1, 1)$)



(c) Side view ($\boldsymbol{P} = \text{diag}(100, 1)$)



(d) Top view ($\boldsymbol{P} = \text{diag}(100, 1)$)

Figure 5.9: Comparison of different weights

Finally, we conclude this chapter with a comparison of different sampling policies. The simulation (from $t = 0$ to $t = 2$) was run in MATLAB with 1,000 samples at each time step. The result is shown in Table 5.1, where the safety distance stands for the distance between $(x, z)_{t=1}$ and the barrier. With all three sampling policies, constraints were satisfied all the time and the robot successfully passed the barrier.

Table 5.1: Sampling policy comparison in 2D simulation

Sampling Policy	**projection**	**rejection**	**truncated Gaussian**
total cost	7.85×10^{-4}	5.75×10^{-4}	3.04×10^{-4}
terminal state $(x, z)_{t=2}$	$(0.028, 0.046)$	$(0.024, 0.065)$	$(0.017, 0.065)$
computation time	3208.91 s	25.55 s	25.37 s
middle state $(x, z)_{t=1}$	$(0.254, 0.808)$	$(0.232, 0.915)$	$(0.242, 1.335)$
safety distance	0.054	0.032	0.042

Note that in rejection sampling, we chose to do rejection sampling 10 times before switching to truncated Gaussian sampling. Although the computational costs of rejection sampling and truncated Gaussian sampling are almost the same, the costs were spent on different phases: for rejection sampling, the computation was expensive during early stage when the constraints are stringent while for truncated Gaussian sampling, the cost was spent evenly. It is also obvious the projection sampling yields extremely high cost and is hard to be implemented in real time.

Chapter 6

Discussion

Most current studies on path integral methods focus on unconstrained systems while constrained problems are actually more common. In this thesis, we investigated how to deal with state-input inequality constraints in two ways:

- With additional assumptions and reasonable approximation, we used the augmented Lagrangian method to respect constraints.
- We have also demonstrated the success of the practical sampling solution with a 2-DoF example.

It should be note that the augmented Lagrangian solution does not work with all kinds of state-input inequality constraints since we assume $\boldsymbol{A}$ in (3.9) to be diagonal. On the other hand, the practical solution can theoretically be applied to all systems with state-input inequality constraints.

We did not cover state-inequality constraints in this thesis, which can be easily dealt with by incorporating a constraint-related cost $p(\boldsymbol{x})$ into the state-dependent term $V(\boldsymbol{x})$ and defining a new state-dependent cost term $\tilde{V}(\boldsymbol{x})$:

$$\tilde{V}(\boldsymbol{x}) = V(\boldsymbol{x}) + p(\boldsymbol{x}) \tag{6.1}$$

$$p(\boldsymbol{x}) = \begin{cases} 0 & \text{state-inequality constraints satisfied} \\ \infty & \text{otherwise} \end{cases} \tag{6.2}$$

Future work includes a numerical example for the theoretical solution. Due to the inefficiency of Monte Carlo sampling, we will combine more efficient sampling methods, such as importance sampling, with our presented work.

Chapter 7

Acknowledgment

I would like to thank Jan Carius and Dr. Farbod Farshidian for supervising me. They gave me many inspiring ideas and patiently answered my infinite questions. Before I started this project, I knew nothing about path integral control. Without their help during these three months, I cannot push myself here. I would also like to thank Prof. Dr. Marco Hutter, who leads this amazing group so that I have this precious opportunity to work with these wonderful people.

Bibliography

[1] D. T. Major and J. Gao, "An integrated path integral and free-energy perturbation- umbrella sampling method for computing kinetic isotope effects of chemical reactions in solution and in enzymes," *Journal of chemical theory and computation*, vol. 3, no. 3, pp. 949–960, 2007.

[2] G. Paolinelli and G. Arioli, "A model for stocks dynamics based on a non-gaussian path integral," *Physica A: Statistical Mechanics and its Applications*, vol. 517, pp. 499–514, 2019.

[3] V. Linetsky, "The path integral approach to financial modeling and options pricing," *Computational Economics*, vol. 11, no. 1-2, pp. 129–163, 1997.

[4] E. Theodorou, F. Stulp, J. Buchli, and S. Schaal, "An iterative path integral stochastic optimal control approach for learning robotic tasks," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 11 594–11 601, 2011.

[5] M. Boué and P. Dupuis, "Markov chain approximations for deterministic control problems with affine dynamics and quadratic cost in the control," *SIAM Journal on Numerical Analysis*, vol. 36, no. 3, pp. 667–695, 1999.

[6] M. Bardi and I. Capuzzo-Dolcetta, *Optimal control and viscosity solutions of Hamilton-Jacobi-Bellman equations*. Springer Science & Business Media, 2008.

[7] H. J. Kappen, "Path integrals and symmetry breaking for optimal control theory," *Journal of statistical mechanics: theory and experiment*, vol. 2005, no. 11, p. P11011, 2005.

[8] ——, "Linear theory for control of nonlinear stochastic systems," *Physical review letters*, vol. 95, no. 20, p. 200201, 2005.

[9] R. Bellman, "Dynamic Programming," *Science*, vol. 153, no. 3731, pp. 34 LP – 37, jul 1966.

[10] W. H. Fleming, "Logarithmic transformations and stochastic control," in *Advances in Filtering and Optimal Stochastic Control*. Springer, 1982, pp. 131–141.

[11] H. J. Kappen and H. C. Ruiz, "Adaptive importance sampling for control and inference," *Journal of Statistical Physics*, vol. 162, no. 5, pp. 1244–1266, 2016.

[12] J. Buchli, F. Farshidian, A. W. Winkler, T. Sandy, and M. Giftthaler, "Optimal and learning control for autonomous robots," *CoRR*, vol. abs/1708.09342, 2017.

[13] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. New York, NY, USA: Springer, 2006.

[14] J. v. Neumann, "The theory of social games," *Mathematical annals*, vol. 100, no. 1, pp. 295–320, 1928.

[15] F. Sha, L. K. Saul, and D. D. Lee, "Multiplicative updates for nonnegative quadratic programming in support vector machines," in *Advances in neural information processing systems*, 2003, pp. 1065–1072.

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

Appendix A

Mathematical Rules

Inner product to trace
The inner product of two vectors with the same dimension can be rewritten in the form of matrix trace:

$$\boldsymbol{u}^\top \boldsymbol{v} = \sum_i \boldsymbol{u}_i \boldsymbol{v}_i = \mathtt{Tr}\left(\boldsymbol{u}\boldsymbol{v}^\top\right) \ . \tag{A.1}$$

In the case of a quadratic form with symmetric weight $\boldsymbol{R}$, the same idea leads to

$$\boldsymbol{u}^\top \boldsymbol{R}\boldsymbol{u} = \mathtt{Tr}\left(\boldsymbol{R}\boldsymbol{u}\boldsymbol{u}^\top\right) \ . \tag{A.2}$$

Matrix derivative
We use denominator layout convention for the matrix derivatives:

$$\frac{\partial\left(\boldsymbol{A}\boldsymbol{x}\right)}{\partial \boldsymbol{x}} = \boldsymbol{A}^\top \ , \tag{A.3}$$

$$\frac{\partial\left(\boldsymbol{x}^\top \boldsymbol{A}\right)}{\partial \boldsymbol{x}} = \boldsymbol{A} \ , \tag{A.4}$$

$$\frac{\partial\left(\boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x}\right)}{\partial \boldsymbol{x}} = \left(\boldsymbol{A} + \boldsymbol{A}^\top\right)\boldsymbol{x} \ , \tag{A.5}$$

$$\frac{\partial g\left(\psi(\boldsymbol{x})\right)}{\partial \boldsymbol{x}} = \frac{\partial g(\psi)}{\partial \psi} \cdot \frac{\partial \psi(\boldsymbol{x})}{\partial \boldsymbol{x}} \ . \tag{A.6}$$

Taylor expansion
The second-order Taylor approximation is

$$f(\boldsymbol{x} + \delta\boldsymbol{x}) \approx f(\boldsymbol{x}) + \frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}^\top \delta\boldsymbol{x} + \frac{1}{2}\delta\boldsymbol{x}^\top \frac{\partial^2 f(\boldsymbol{x})}{\partial \boldsymbol{x}^2}\delta\boldsymbol{x} \ . \tag{A.7}$$