Convex Optimization Street Fighting

Quill Healey

June 2, 2024

Chapter 1

Introduction

1.1 What are These Notes?

[BV04]

Part I

Theory

Chapter 2

Linear Algebra

to adds:

- incidence matrix page 132 VMLS (networks there too)
- bidiagonal matrix page 312 [BV04]

2.1 Eigenvalues and Eigenvectors

2.1.1 Stability

Continuous Time Systems

Discrete Time Systems

[PG24] **HW0 Q1**. *Discrete-time LTI stability.* Consider the system $x_{t+1} = Ax_t + Bu_t$, where

$$A = \begin{bmatrix} 4/5 & 0 & 0 \\ 0 & \sqrt{3} & 1 \\ 0 & -1 & \sqrt{3} \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 \\ 1 & 1 \\ 1 & 0 \end{bmatrix}.$$

(a) Explain whether or not this sytem is "open-loop stable" (asymptotically stable for $u_t \equiv 0$).
Response. This sytem is unstable. To see this, note $|\lambda_1| = |\lambda_2| \approx 2$. (Most likely they equal two, the numerical computation via Python yields a magnitude of 1.99... with 15 trailing 9s). Recall that for a discrete time LTI system to be open-loop stable, the magnitude of all eigenvalues must be less than one.
(b) Design a linear feedback controller $u_t = Kx_t$ with fixed gain matrix $K \in \mathbf{R}^{2\times 3}$ such that the closed-loop system is asymptotically stable.
Response. This is a **state feedback control** problem where K is the **state-feedback gain matrix**. In this setting, we can rewrite the LTI system as

$$x_{t+1} = Ax_t + Bu_t = Ax_t + B(Kx_t) = (A + BK)x_t, \quad t = 1, 2, \ldots$$

2.2 Linear Transformations

Chapter 3

Matrix Calculus

3.1 Notation

3.2 Rethinking the Derivative

3.3 Some Analysis

3.4 Some Geometry

3.5 Calculus Composition Rules

- be careful with composing differentials versus derivatives versus gradients
- Move right to "beyond high school calculus," most importantly now must be careful with commuting.
- We are assuming differentiability throughout.

Sum Rule

The generalized matrix calculus sum rule is exactly what you would think it would be. Given $f : \mathbf{R}^n \to \mathbf{R}^m$ where $f(x) = g(x) + h(x)$,

$$df = dg + dh,$$

which can be expanded (according to the definition of a differential) as

$$Df(x)dx = Dg(x)dx + Dh(x)dx = (Dg(x) + Dh(x))dx.$$

Therefore,

$$Df(x) = Dg(x) + Dh(x).$$

Product Rule

Consider $f : \mathbf{R}^n \to \mathbf{R}^m$ defined as $f(x) = g(x)h(x)$. The product rule derivation is as follows.

$$df = f(x + dx) - f(x) = g(x + dx)h(x + dx) - g(x)h(x).$$

Using that $g(x + dx) = g(x) + Dg(x)dx$ and $h(x + dx) = h(x) + Dh(x)dx$, the differential df can be expanded as

$$\begin{aligned} df &= (g(x) + Dg(x)dx)\,(h(x) + Dh(x)dx) - g(x)h(x) \\ &= g(x)h(x) + g(x)Dh(x)dx + Dg(x)dxh(x) + Dg(x)dxDh(x)dx - g(x)h(x), \end{aligned}$$

ignoring the higher order terms, removing like terms, and being mindful *to not commute terms*, we are left with

$$df = g(x)Dh(x)dx + Dg(x)dxh(x).$$

This of course should look like the typical product rule seen for Calculus I derivatives, but we have mixed differentials and derivatives.

$$df = g(dh) + (dg)h \quad \text{and} \quad Df(x)dx = g(x)\,(Dh(x)dx) + (Dg(x)dx)\,h(x).$$

- Notice that we have not written that $Df(x) = g(x)Dh(x) + Dg(x)h(x)$.

3.6 Calculus Atoms

3.6.1 Vector Functions

Linear and Affine Functions

Take $f : \mathbf{R}^n \to \mathbf{R}^m$ defined as $f(x) = Ax - b$ for some $A \in \mathbf{R}^{m\times n}$ and $b \in \mathbf{R}^m$.

$$df = d(Ax - b) = A(x + dx) - b - (Ax - b) = Adx,$$

so

$$Df(x) = A \quad \text{and} \quad \nabla f(x) = A^T.$$

Euclidean Inner Product

Take $f : \mathbf{R}^n \to \mathbf{R}$ defined as $f(x) = x^T x$.

$$\begin{aligned} df = d(x^T x) &= (x + dx)^T (x + dx) - x^T x \\ &= x^T x + x^T dx + (dx)^T x + (dx)^2 - x^T x \\ &= 2x^T dx, \end{aligned}$$

where the last equality holds because a is always equal to a^T when $a \in \mathbf{R}$. Therefore,

$$Df(x) = 2x^T \quad \text{and} \quad \nabla f(x) = 2x.$$

Quadratic Form

Consider $f : \mathbf{R}^n \to \mathbf{R}$ defined as $f(x) = x^T Ax$ for some $A \in \mathbf{R}^{n \times n}$.

$$\begin{aligned} df = d(x^T Ax) &= (x + dx)^T A(x + dx) - x^T Ax \\ &= x^T Ax + x^T Adx + (dx)^T Ax + (dx)^T Adx - x^T Ax \\ &= x^T Adx + x^T A^T dx \\ &= x^T (A + A^T) dx, \end{aligned}$$

so

$$Df(x) = x^T (A + A^T) \quad \text{and} \quad \nabla f(x) = \left(x^T (A + A^T)\right)^T = (A + A^T)x.$$

(Note that $(A + A^T)^T = (A + A^T) \Leftrightarrow A + A^T \in \mathbf{S}^n$.)

Quadratic Form Restricted Case

Consider the same function $f : \mathbf{R}^n \to \mathbf{R}$ where $f(x) = x^T Ax$, but now $A \in \mathbf{S}^n$. The above differential derivation still holds:

$$df = x^T (A + A^T) dx,$$

but because $A = A^T$, we can further simplify the differential to

$$df = 2x^T Adx.$$

Consequently,

$$Df(x) = 2x^T A \quad \text{and} \quad \nabla f(x) = 2A^T x = 2Ax,$$

where we again use that $A^T = A$.

3.6.2 Matrix Functions

3.6.3 Summary

3.7 (Bonus) Automatic Differentiation

Chapter 4

Probability Theory

About:

- chapter will examine probability theory through the lense of convex optimization. Probability theory itself is not the main focus.

Todos

- Where to add chebychev
- simple chebychev example page 54 VMLS
- correlation coefficient page 60 VMLS
-

Outline

- Basics (what you need probabilistically to proceed) subsection 1: the probability theory. subsection 2: Viewing the probabilistic objects through the lense of convex optimization
- Sets: Exercise 2.15
- Functions: Exercise 3.24.
- Basic Problems

4.1 Basics

4.1.1 Probabilistics Objects and Notation

I will adopt the notation used by the source material.

- Unlike in statistical texts, random variables (RVs) will not by default be a capital letter. That is, never assume that a capital lettered variable (or any other variable, for that matter) is a random variable, even in a statistical context. If x or X is a random variable, it will be declared as such.

- A random variable, x, may be real valued or it may be vector valued. That is to say, the notation for a random variable or a *random vector* will amost always be the same. However, it will always be specified upon instantiation (using computer science/software engineering dialect) whether x is a random variable or whether it is a random vector.

- In this chapter, we will primarily work with random variables which takes on discrete values. That is, $x \sim (\Omega_x, E_x, \mathbf{Prob}_x)$ where $\Omega_x = \{a_1, a_2, \ldots, a_n\} \subset \mathbf{R}^n$ and it is assumed that $a_1 < a_2 < \cdots < a_n$.

- The fundamental probability space $(\Omega_\omega, E_\omega, \mathbf{Prob}_\omega)$ which induces the random variable's probability space via the relation $x : \Omega_\omega \to \Omega_x$ is irrelevant to our analysis.

- The probability of an outcome a_i in Ω_x ocurring will be denoted in the following ways:

$$p_x(a_i) = \mathbf{Prob}_x(a_i) = \mathbf{Prob}(x = a_i) = p_i.$$

  We'll refer to $p_x(\cdot)$ as the *probability mass function* (PMF) for the random variable x and to $\{p_i\}_{i=1}^n$ as the RV's distribution.

- We'll let $F_x(\alpha) = \mathbf{Prob}(x \leq \alpha)$ be the *cumulative distribution function* (CDF) for the random variable x.

4.1.2 Discrete Probabilistic Objects through a Convex Lens

Before proceeding to exercises and, it is helpful to collect the above objects

Before proceeding to exercises it is helpful to collect some objects. We take x to be a discrete random variable on a finite sample space, as defined above.

- The CDF of x can be expanded as

$$F_x(\alpha) = \mathbf{Prob}(x \leq \alpha) = \mathbf{Prob}(x \leq \alpha_k) = \sum_{i=1}^{k} p_i,$$

  where $k = \max\{j \mid a_j \leq \alpha\}$. It is **critical** to note that k is a fixed integer, **independent of** p. Furthermore, the CDF is just a linear function of p.

- The *expectation* (or weighted average) of x, $\mathbf{E}\, x = \sum_{i=1}^n p_i a_i = p^T a$, is a linear function of p.

- When we transform a random variable, $x \to f(x)$, the expectation becomes

$$\mathbf{E}\, f(x) = \sum_{i=1}^{n} p_i f(a_i),$$

which **is still a linear function of** p. (A curious reader may wonder what types of transformations are valid and how this works. Since probability theory is not the focus of these notes, accept the above statement as true since it will be true for all examples and exercises in these notes. However, to answer this question I recommend Murphy and Lay).

- The *variance* of x, which can be thought of as a measure of how much x typically deviates from its expectation, is defined as $\mathbf{Var}\, x = \mathbf{E}\,[(x - \mathbf{E}\, x)^2]$. However, we can rewrite the variance using common properties of the expectation operator as

$$\begin{aligned}\mathbf{Var}\, x &= \mathbf{E}\left[(x - \mathbf{E}\, x)^2\right] \\ &= \mathbf{E}\left[x^2 - 2x\mathbf{E}\, x + (\mathbf{E}\, x)^2\right] \\ &= \mathbf{E}\, x^2 - 2(\mathbf{E}\, x)(\mathbf{E}\, x) + (\mathbf{E}\, x)^2 = \mathbf{E}\, x^2 - (\mathbf{E}\, x)^2.\end{aligned}$$

To remember this more tractable expression, firstly note that the variance of a random variable is nonnegative. This can be seen mathematically from the definition of **Var**, and it should also be intuitive when thinking of the variance *as a measure*. Furthermore, recall the general form of Jensen's inequality: $f(\mathbf{E}\, x) \leq \mathbf{E}\, f(x)$, where f is a convex function. Moving the left hand side to the right hand side and using $x \xrightarrow{f} x^2$ yields our desired expression.

- The quartile of x

4.2 Sets

With these objects at our disposal, most of the solutions to problems in **Exercise 2.15** are immediately apparent. Furthermore, we just address the following.

(a-e) Note that each operator on x is linear in p. Furthermmore, the mixing of linear expressions in p with inequalities ($\leq$ or $\geq$) does not result in nonconvex conditions on P.

(f-g) Firstly, let's utilize Chapter Three composition techniques:

- $\mathbf{E}\, x^2$, as already established, is linear in p.
- $f : \mathbf{R} \to \mathbf{R}$ defined as $f(x) = x^2$ is convex. When $g : \mathbf{R} \to \mathbf{R}$ is also convex, $f(g(x)) = (g(x))^2$ is convex. Furthermore, $(\mathbf{E}\, x)^2$ is a convex quadratic in p.
- The expression (*linear function* $-$ *convex quadratic function*) is a *concave quadratic expression.*

Using $f(p) = \mathbf{Var}\, x = \mathbf{E}\, x^2 - (\mathbf{E}\, x)^2$ to emphasize that the variance is a concave quadratic function of p, (f) and (g) follow from the recollection that $\{p \in P \mid f(p) \geq \alpha\}$ is a convex set while $\{p \in P \mid f(p) \leq \alpha\}$ is not. Additonally, note that the expression in the answer key for $(\mathbf{E}\, x)^2$ can be derived as follows:

$$(\mathbf{E}\, x)^2 = (p^T a)^2 = (p^T a)(p^T a) = (p^T a)(p^T a)^T = (p^T a)(a^T p) = p^T A p.$$

(h) The picture in the solutions posted to EE364a's Stanford Engineering Everywhere page is very useful.

Part II

Applications Part I

Chapter 5

Approximation and Fitting

First use that

$$\text{minimize } f_0(x) = \|Ax - b\|_2$$

is equivalent to (since f_0 is nonnegative)

$$\text{minimize } f_0(x)^2 = \|Ax - b\|_2^2,$$

where $f_0(x) = \|Ax - b\|_2^2 = (Ax - b)^T (Ax - b)$. Important observations

- f_0 is convex.
- The problem is unconstrained.

Furthermore, we know that the globally optimal solution, x^*, satisfies $\nabla f_0(x^*) = 0$. We use this as an opportunity to practice matrix calculus. Specifically, we will use differentials to derive ∇f_0. Start by writing out

$$df_0 = d\left((Ax - b)^T (Ax - b)\right) = (A(x + dx) - b)^T (A(x + dx) - b) - (Ax - b)^T (Ax - b).$$

Recall that we wish to transform this equation into the form $df_0 = Df_0(x)dx$. Or rather, we want to re-arrange the expression for df_0 so that the dx terms are collected in one place and there's a clear expression for $Df_0(x)$. (Remember that depending on the complexity of the function, it's not always the case that the dxs will be collected on the right, as implied above.) Of course, because we've already developed matrix calculus atoms and compositions, we can take a shortcut. Rewrite $f_0(x) = g(h(x))$ where $g : \mathbf{R}^m \to \mathbf{R}$ and $h : \mathbf{R}^n \to \mathbf{R}^m$ are defined as

$$g(x) = x^T x \quad \text{and} \quad h(x) = Ax - b,$$

with corresponding matrix calculus atoms

$$Dg(x) = 2x^T \quad \text{and} \quad Dh(x) = A.$$

Also observe what this implies about f_0. Instead of being a mapping from n-dimensional Euclidean space to the real line, it is actually the composition mapping $f_0 : \mathbf{R}^n \to \mathbf{R}^m \to \mathbf{R}$. This isn't profound, but it can be helpful to keep in mind when performing a derivation such as this one since it reinforces the danger of absentmindedly commuting terms. Employing the chain rule, $Df_0(x) = Dg(h(x)Dh(x))$,

$$Df_0(x) = 2(Ax - b)^T A, \quad \text{so} \quad \nabla f_0(x) = \big(2(Ax - b)^T A\big)^T = 2A^T(Ax - b).$$

Equating $\nabla f_0(x) = 0$, we arrive at the normal equations

$$A^T Ax = A^T b,$$

with associated solution

$$x^* = (A^T A)^{-1} A^T b.$$

5.1 Forecasting

Sources:

- AR Model (page 28 VMLS)
- RMS Prediction Error (page 50 VMLS)
- Dirichlet energy page 66 VMLS
- Finance example page 67 VMLS
- Time series auto-correlation page 67
- back-test timing page 127
- Time series smoothing page 138 VMLS (convolution)
- Markov model page 164
- Regularizing time series (also Laplacian regularization) page 317 VMLS
- Estimating a periodic time series page 318 VMLS (example of regularization)
- Example of periodic time series to work on page 15.10 VMLS
- de meaned return time series page 378 VMLS
- feature matrix page 257 VMLS

5.2 Least Squares Data Fitting

- "Least squares is widely used to construct a mathematical model from some data, experiments, or observations."
- Note how least squares is approached before fitting to data.
- How to relate to a statistical approach
- Is least squares part of the approximation and fitting chapter?

5.3 Recurrent Neural Networks

Source: Intro Statistical Learning with Python

- Used when data arise as sequences (so we are considering time series data)
- RNNs build models that take into account this sequential nature of the data and build a memory of the past
- The feature for each observation is a sequence of vectors

$$x_t \in \mathbf{R}^n, \quad t = 1, \ldots, L$$

Chapter 6

Statistical Estimation

Unlike in the probability chapter, we

6.1 Parametric Distribution Estimation

My notation

- likelihood function will use the same notation as the probability density function: $p_\theta(x)$.

6.1.1 Maximum Likelihood

Poisson Maximum Likelihood

Suppose we are a store owner. It is obviously desireable to have an estimate for the number of customers that might come to our store on any given day. In the Approximation and Fitting chapter, we addressed this type of problem with **forecasting**. More specifically, if we had a dataset containing the number of customers who came into our store each day, we could fit a time series model (perhaps an AR model or a model with trend and seasonal components) to attempt to predict the customer demand over some time horizon (the next week, perhaps).

In this chapter, we take an alternative approach. Specifically, $y \approx f(x)$. (Although it is worth mentioning that more often than not, forecasting problems)

While **predicting or forecasting demand** is a difficult problem with complications such as seasonality, perhaps we do not yet have any demand data and we just want some baseline model which can help inform our inventory ordering. Furthermore, for the next N days we count the number of customers who visit our store *each day*. We can pose our estimation problem as follows:

- Let $x \in \mathbf{R}$ be a random variable representing the number of customers who visit the store on any given day.
- Since the Poisson distribution is often used to count the number of random arrivals to a system within a given amount of time, we assume that $x \sim \text{Poisson}(\lambda)$.

- Having collected N instances of x

$$p_\lambda(x) = \prod_{t=1}^{T} \frac{e^{-\lambda}\lambda_t^x}{x_t!}$$

The corresponding log-likelihood function

$$\begin{aligned} l(\lambda) &= \left(\sum_{t=1}^{T} -\lambda + x_t \log \lambda - \log(x_t!) \right) \\ &= -T\lambda + \log \lambda \left(\sum_{t=1}^{T} x_t \right) - \left(\sum_{t=1}^{T} \log(x_t!) \right) \end{aligned}$$

$$\text{maximize } -T\lambda + \log \lambda \sum_{t=1}^{T} x_t - \sum_{t=1}^{T} \log(x_t!)$$

Equivalent to (and have written λ under maximize to emphasize). Also note that unlike in other contexts, we have not specified that $\lambda > 0$. I want to stress the reason: we have adopted the notation of CVX and use problem domain. $\lambda > 0$ is not really a constraint on our maximization problem, if you were to give this problem a non-positive lambda, we couldn't even evaluate the problem.

$$\underset{\lambda}{\text{maximize}} \; -T\lambda + \left(\sum_{t=1}^{T} x_t \right) \log \lambda.$$

$$\hat{\lambda} = \text{argmax}_\lambda \; l(\lambda) = \frac{1}{T} \sum_{t=1}^{T} x_t$$

which of course is just the sample mean of the T observations in the sample.

Part III

Applications Part II

Chapter 7

Control

7.1 Overview

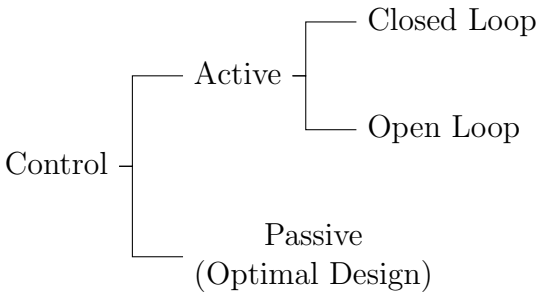

Figure 7.1: Control Strategies

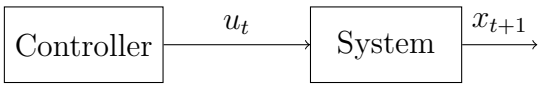

Figure 7.2: Open Loop Control Strategy

Source: Control bootcamp (Brunton)

- (Dynamical Systems) Systems of ODEs describing state of system has been a successful modeling framework.
- Often want to actively make changes to the system.
- In control theory, you have a dynamical system of interest, you write down the system of equations which describes the behavior of the system, and then you create a control theory to create a more "desireable" system behavior.
- Passive control (Boyd would call optimal design): design an upfront solution (ex: minimizing drag on a truck)

- Active control: pump energy into the system to actively manipulate its behavior.

Active Control

- Open Loop

7.2 Basic Examples

[BV04] **Exercise 4.16**. *Minimum fuel optimal control.* Consider the LTI dynamical system with state $x_t \in \mathbf{R}^n$, $t = 0, \ldots, N$, and actuator or input signal $u_t \in \mathbf{R}$, for $t = 0, \ldots, N-1$. The dynamics of the system are governed by the linear recurrence

$$x_{t+1} = Ax_t + bu_t, \quad t = 0, \ldots N-1,$$

where $A \in \mathbf{R}^{m \times n}$ and $b \in \mathbf{R}^n$ are given. Assume the initial state is $x_0 = 0$. The *minimum fuel optimal control problem* is to choose the inputs $u_0, \ldots u_{N-1}$ so as to minimize the total fuel consumed, which is given by

$$F = \sum_{t=0}^{N-1} f(u_t),$$

subject to the constraint that $x_N = x_{\text{des}}$, where N is the (given) time horizon and $x_{\text{des}} \in \mathbf{R}^n$ is the (given) desired final or target state. The function $f : \mathbf{R} \to \mathbf{R}$ is the *fuel use map* for the actuator, and gives the amount of fuel used as a function of the actuator signal amplitude. In this problem we use

$$f(a) = \begin{cases} |a| & |a| \leq 1 \\ 2\,|a| - 1 & |a| > 1. \end{cases}$$

Formulate the minimum fuel control problem as an LP.

Response. Firstly, a few notes about the problem itself:

- [Rea20] What is an actuator (broadly)?
  - A device that makes something move or operate. As a trivial example, an actuator moves the sliding doors at a grocery store.
  - Two types: straight line movement (linear) and circular movement (rotary).
  - An actuator converts a source of energy into a physical, mechanical motion. A silly example of this: a handwheel can be used to feed energy into a rotary actuator. In industrial applications, there are three typical sources of energy: *Electric* uses electricty (duh), *hydraulic* use a variety of liquids, *pneumatic* are operated by compressed air.
  - Common industrial actuators are electric motors, hydraulic motors, and pneumatic control valves.

- Example use of a pneumatic actuator: "PLC analog output card* produces a 4-20 mA current to move the valve from fully open to fully closed. The 4-20 mA current will be converted to pneumatic pressure, which becomes the source of energy to operate the actuator."
- *A Programmable Logic Controller (PLC) analog output card is a component used to convert digital signals from the PLC's processor into analog signals (digital-to-analog conversion, DAC that can be used to control external devices.

- Actuator in this problem.
  - u_t is a scalar valued signal that directly affects the state of our system of interest (since it's in the linear recurrence equation).
  - Physically, this signal is being used to direct the actuator, which in turn is turning energy into mechanical motion.
  - Therefore, the fuel use map for the actuator is a mathematical relation between the amplitude of the signal directing the actuator and the fuel being used. In other words, we can imagine a larger actuator signal corresponds to greater actuator motion, and in this instance, the source of energy needed to produce this motion is "fuel."

Naive formulation of this problem

$$\begin{array}{ll} \text{minimize} & \sum_{t=0}^{N-1} f(u_t) \\ \text{subject to} & x_{t+1} = Ax_t + bu_t \qquad t = 0, \ldots, N-1 \\ & x_0 = 0, \quad x_N = x_{\text{des}}. \end{array}$$

Consider the graph of the fuel use map 7.3

$$\begin{array}{ll} \text{minimize} & \sum_{t=0}^{N-1} \max\left\{|u_t|, 2|u_t| - 1\right\} \\ \text{subject to} & x_{t+1} = Ax_t + bu_t \qquad t = 0, \ldots, N-1 \\ & x_0 = 0, \quad x_N = x_{\text{des}}. \end{array}$$

$$\begin{array}{ll} \text{minimize} & \mathbf{1}^T t \\ \text{subject to} & x_{t+1} = Ax_t + bu_t \qquad t = 0, \ldots, N-1 \\ & x_0 = 0, \quad x_N = x_{\text{des}}, \\ & \max\left\{|u_i|, 2|u_i| - 1\right\} \le t_i, \quad i = 0, \ldots, N-1 \end{array}$$

$$\begin{array}{ll} \text{minimize} & \mathbf{1}^T t \\ \text{subject to} & x_{t+1} = Ax_t + bu_t \qquad t = 0, \ldots, N-1 \\ & x_0 = 0, \quad x_N = x_{\text{des}}, \\ & |u_i| \le t_i, \qquad i = 0, \ldots, N-1 \\ & 2|u_i| - 1 \le t_i \qquad i = 0, \ldots, N-1 \end{array}$$

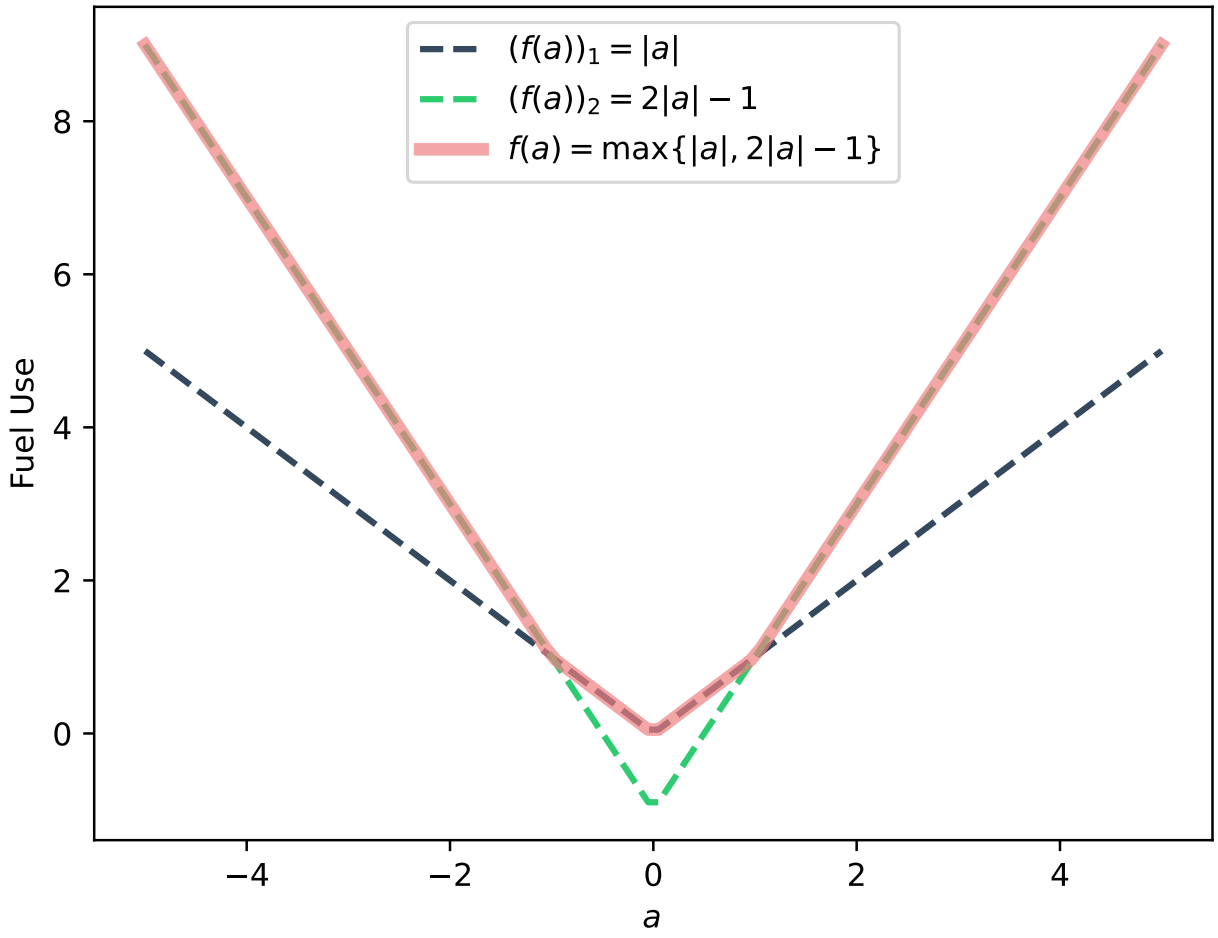

Figure 7.3: Actuator Fuel Use Map.

$$
\begin{array}{lll}
\text{minimize} & \mathbf{1}^T t & \\
\text{subject to} & x_{t+1} = Ax_t + bu_t & t = 0, \ldots, N-1 \\
& x_0 = 0, \quad x_N = x_{\text{des}}, & \\
& y_i \leq t_i, & i = 0, \ldots, N-1 \\
& 2y_i - 1 \leq t_i & i = 0, \ldots, N-1 \\
& |u_i| \leq y_i & i = 0, \ldots, N-1
\end{array}
$$

Important/helpful to remeber the definition of absolute value: $|a| = \max\{a, -a\}$

$$
\begin{array}{lll}
\text{minimize} & \mathbf{1}^T t & \\
\text{subject to} & x_{t+1} = Ax_t + bu_t & t = 0, \ldots, N-1 \\
& x_0 = 0, \quad x_N = x_{\text{des}}, & \\
& y_i \leq t_i, & i = 0, \ldots, N-1 \\
& 2y_i - 1 \leq t_i & i = 0, \ldots, N-1 \\
& -y_i \leq u_i \leq y_i & i = 0, \ldots, N-1
\end{array}
$$

Write more compactly as

$$\begin{array}{ll} \text{minimize} & \mathbf{1}^T t \\ \text{subject to} & x_{t+1} = Ax_t + bu_t \qquad t = 0, \ldots, N-1 \\ & x_0 = 0, \quad x_N = x_{\text{des}}, \\ & y \preceq t \\ & 2y - \mathbf{1} \preceq t \\ & -y \preceq u \preceq y_i \end{array}$$

Still not ideal form because of the linear recurrence. Use control theory result.

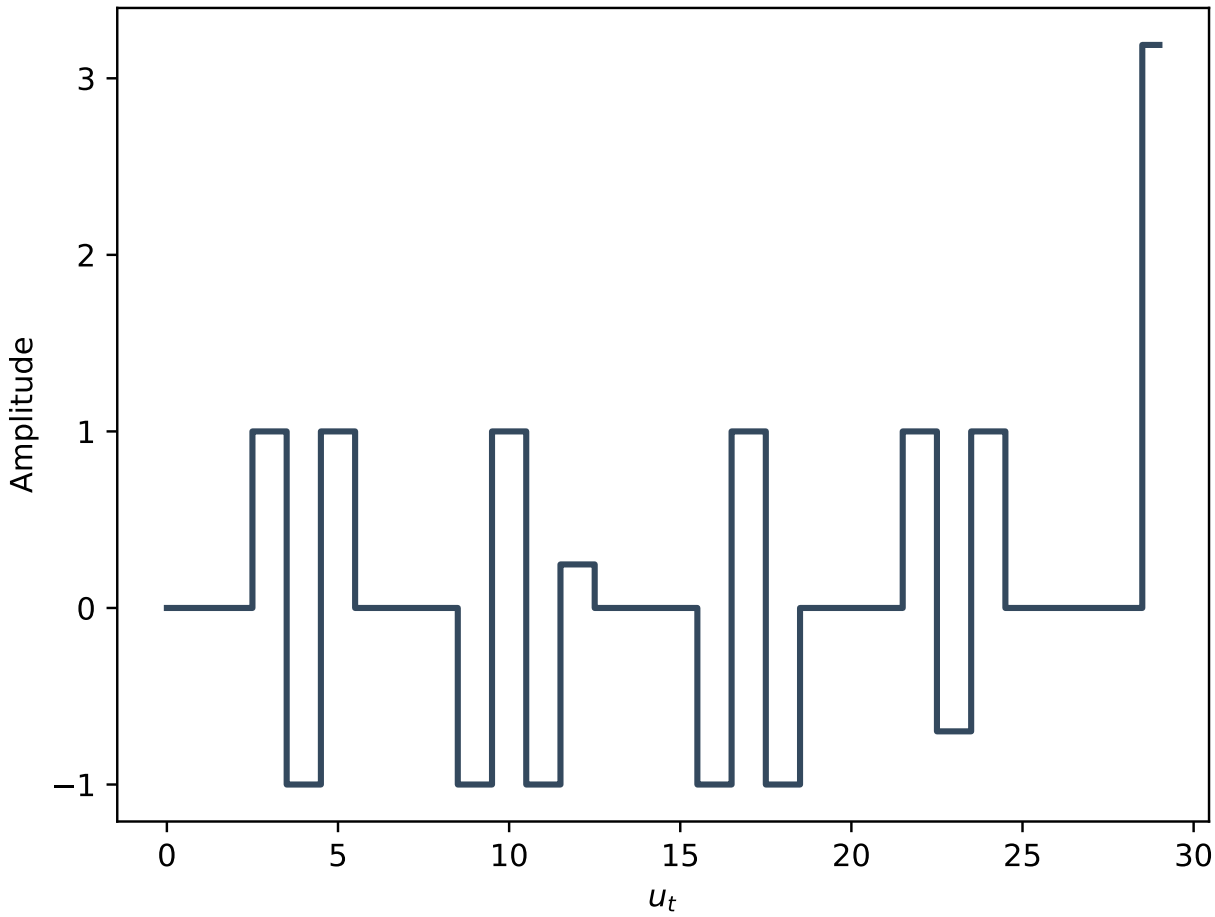

Figure 7.4: Minimum fuel actuator signal.

7.3 Model Predictive Control

7.3.1 Overview and Basic Formulations

7.3.2 Examples

Almost MPC

We apply MPC to the following output tracking example. The purpose of this exercise is simply to show how sequentially solving receeding horizon problems can converge to the agnostic control problem (where the problem is solved with full access to the desired output) as the horizon is increased. Note that this example **does not** demonstrate the power of using MPC in a stochastic setting.

[Boy-8] **HW7 Q1.** *MPC for output tracking.* Consider the linear dynamical system

$$x_{t+1} = Ax_t + Bu_t, \quad y_t = Cx_t, \quad t = 0, \ldots, T-1,$$

with state $x_t \in \mathbf{R}^n$, input $u_t \in \mathbf{R}^m$, and output $y_t \in \mathbf{R}^p$. The matrices A and B are known, and $x_0 = 0$. The goal is to choose the input sequence $u_1, \ldots, u_t$ to minimize the output tracking cost

$$J_{\text{output}} = \sum_{t=1}^{T} \left\| y_t - y_t^{\text{des}} \right\|_2^2,$$

subject to $\|u_t\|_\infty \leq U^{\max}$, $t = 0, \ldots, T-1$.

For the remainder of this problem we work with the specific problem instance with associated data

$$A = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 0.5 \\ 1 \end{bmatrix}, \quad C = \begin{bmatrix} -1 & 0 & 1 \end{bmatrix},$$

$T = 100$, and $U^{\max} = 0.1$. The desired output trajectory is given by

$$y_t^{\text{des}} = \begin{cases} 0 & t < 30, \\ 10 & 30 \leq t < 70 \\ 0 & t \geq 70. \end{cases}$$

(a) Find the optimal input u^* and the associated optimal cost J^*.

Response. This is simply a *linear* (in the dynamics) *time-invariant quadratic tracking* problem. Instead of doing a theoretical analysis of controllability, etc., we can determine the feasibility of the control problem by formulating and attempting to solve the following convex optimization problem:

$$\begin{array}{ll} \text{minimize} & J_{\text{output}} = \sum_{t=1}^{100} \left\| Cx_t - y_t^{\text{des}} \right\|_2^2 \\ \text{subject to} & \|u_t\|_\infty \leq 0.1, \quad x_{t+1} = Ax_t + Bu_t, \quad t = 0, \ldots, 99 \\ & x_0 = 0, \end{array}$$

(with the provided data for A, B, C, and y^{des}, of course.) Using CVXPY, we obtain the optimal cost $J^*_{\text{outout}} = 112.4157$.

(b) *Rolling look-ahead*. Now consider the input obtained using an MPC-like method where at time t, we find the values of $u_t, \ldots, u_{t+N-1}$ that solve the following convex optimization problem

$$\begin{array}{ll} \text{minimize} & J_{\text{output}} = \sum_{\tau=t+1}^{t+N} \left\| Cx_\tau - y_\tau^{\text{des}} \right\|_2^2 \\ \text{subject to} & \|u_\tau\|_\infty \leq 0.1, \quad x_{\tau+1} = Ax_\tau + Bu_\tau, \qquad \tau = t, \ldots, t+N-1 \\ & x_0 = 0. \end{array}$$

The value N is the amount of *look-ahead*, since it dictates how much of the future of the desired output signal we are allowed to access when we decide on the current input.

Find the input signal for look-ahead values $N = 8$, $N = 10$, and $N = 12$. Compare the cost J_{output} obtained in these three instances to the optimal cost J^*_{output} found in part (a).

Response. The Python code used to solve this problem can be found in these note's associated examples folder under 364b, so it is omitted here. The cost obtained by applying MPC with

- $N = 8$ is 379.634,
- $N = 10$ is 128.13,
- and $N = 12$ is 123.62.

Figure 7.5 shows the output trajectories for $N = 8$, $N = 10$, $N = 12$, the optimal output trajectory, and the desired output trajectory.

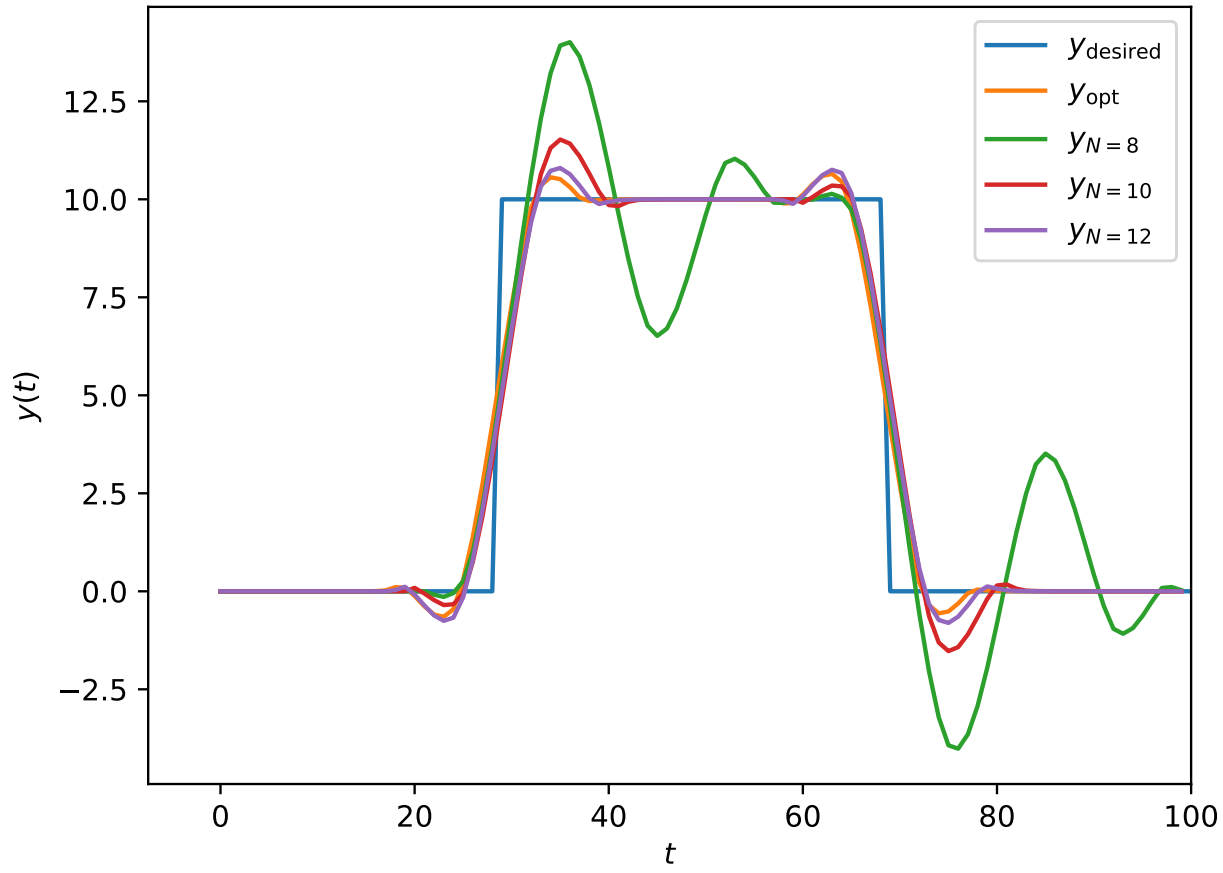

Figure 7.5: Output Trajectories.

Part IV

Algorithms

Bibliography

[Boy-8] Stephen Boyd. *EE36b: Convex Optimization II*. 2007-8. URL: https://see.stanford.edu/Course/EE364B.

[BV04] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge, UK: Cambridge University Press, 2004. ISBN: 978-0521833783.

[PG24] Marco Pavone and Daniele Gammelli. *AA203: Optimal and Learning-Based Control*. 2024. URL: https://stanfordasl.github.io/aa203/sp2324/.

[Rea20] RealPars. *What is an Actuator*. 2020. URL: https://www.youtube.com/watch?v=LHn7O6PUaoY.

Appendix A

Implementation

- For now setting is Python.
- Will include C++ gradually.
- Especially how to generate C optimization solvers from CVXPY.

A.1 CVXPY

A.2 C++