Algorithms for Large-scale Data Analytics and Applications to the COVID-19 Pandemic

by

Michael Lingzhi Li

B.A. (Hon), University of Cambridge (2017)
MBAn., Massachusetts Institute of Technology (2018)

Submitted to the Sloan School of Management
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Operations Research

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2022

Author .....
Sloan School of Management
December 16th, 2021

Certified by .....
Dimitris J. Bertsimas
Boeing Leaders for Global Operations Professor of Management
Associate Dean for Business Analytics
Thesis Supervisor

Accepted by .....
Georgia Perakis
William F. Pounds Professor of Management Science
Co-director, Operations Research Center

Algorithms for Large-scale Data Analytics and Applications to the COVID-19 Pandemic

by

Michael Lingzhi Li

Submitted to the Sloan School of Management
on December 16th, 2021, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Operations Research

Abstract

Operations Research (OR) can be defined as using advanced quantitative tools to make better decisions and create impact. In the modern world, to generate impact, we need both scalable algorithms that allow us to extract insights from an ever-increasing amount of data, and also important applications to apply our insights to the world. In this thesis, we demonstrate both sides of the coin.

In the first part of the thesis, we focus on building scalable algorithms for large-scale data analytics. In Chapter 2, we consider a novel reformulation of the matrix completion problem and developed a projected stochastic gradient descent method, fastImpute, to solve matrix completion $20x$ faster than state-of-the-art methods while providing optimality guarantees. In Chapter 3, we introduce the Interpretable Matrix Completion problem (IMC) to provide meaningful insights for low-rank matrices using side information. We designed an algorithm, OptComplete, based on the novel concept of stochastic cutting planes that enables us to solve extremely large instances. In Chapter 4, we extend OptComplete to general data-driven mixed-integer optimization problems including sparse regression, support vector machines, and the knapsack problem. We show that the algorithm is able to match or exceed state-of-the-art results.

The second part of the thesis revolves around applying large-scale data analytics to the COVID-19 pandemic. In Chapter 5, we introduce a novel policy-driven epidemiological model, DELPHI. We show that DELPHI compares favorably with other top epidemiology models and predicted the large-scale epidemics in US, UK and Russia months before. We demonstrate how the explicit modeling of governmental interventions in DELPHI enabled its use for planning the trial of the Janssen Ad26.Cov2.S vaccine. In Chapter 6, we apply DELPHI to determine COVID-19 mass vaccination centers in the US. We developed an optimization model to allocate the limited vaccine supply and minimize future pandemic deaths, while fully incorporating the nonlinear DELPHI dynamics. We proposed a coordinate descent model to solve the problem at scale, and showed how optimized vaccine allocation can save 20% more individuals while still ensuring equity. Our conclusions directly affected how FEMA allocated its vaccines, increasing its focus on states such as Texas and Florida.

Thesis Supervisor: Dimitris J. Bertsimas
Title: Boeing Leaders for Global Operations Professor of Management
Associate Dean for Business Analytics

Acknowledgments

First, I would like to thank my academic advisor, Dimitris Bertsimas, for his phenomenal support and guidance over over the course of my PhD. Given his highly infeasible schedule, I find the amount of effort that Dimitris dedicated on my academic and personal development almost magical, and I am extremely grateful for this. In fact, without Dimitris, I probably would have never entered research, much less finish a Doctorate. Thank you for showing me how impactful research can be, and for cultivating and influencing me to become a better researcher. I look forward to our future collaborations for years to come.

I would also like to thank Alexander Jacquillat and Nikolaos Trichakis for agreeing to be on my thesis committee and also collaborating with me in research. It has been an extremely pleasant experience working with both of you. I would also like to thank Georgia Perakis for providing extremely helpful guidance throughout my PhD.

When I entered MIT, I was worried about entering a competitive environment. However, such thoughts were quickly dispelled by the vibrant and supportive community at the ORC. This is a testimonial to the work done by the ORC faculty and the students who built up a friendly and collaborative atmosphere. I would like to thank the many friends I made here, including Max Biggs, Leonard Boussioux, Wenyu Chen, Peter Cohen, Ryan Cory-Wright, Kayla Cummings, Vassilis Digalakis, Driss Lahlou Kitane, Xiaoyue Gong, Rim Hariss, Jason Chek Nam Liang, Yu Ma, Irra Na, Agni Orfanoudaki, Ted Papalexopoulos, Ivan Paskov, Matthew Sobiesk, Kimberly Villalobos Carballo, Qingyang Xu, Evan Yao, Cynthia Zeng, Renbo Zhao and El Ghali Zerhouni. I would also like to thank my other close friends and collaborators for their support, including Hamza Tazi Bouardi, Charles Herrmann, Hsin-Hsiao Scott Wang, and Daniel Wintz.

It goes without saying that an important person during this journey has been my girlfriend Yuan Shi. Though we were thousands of miles apart for the majority of my PhD, out relationship still managed to flourish, and I thank you for staying by my side (both figuratively and emotionally) and supporting me throughout this entire journey. I look forward to many many years ahead with you by my side.

Finally, I would like to thank my parents, Angela Zhu and Wanxiang Li, for their uncon-

ditional love and support throughout my entire life. I cannot find enough words to describe how grateful I am for growing up in such a loving family with so much joy and happiness. I would never be in the position I am today if not for everything that you did for me. I dedicate this thesis to you.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Dr. Anthony Fauci once said: "I believe I have a personal responsibility to make a positive impact on society". Operations Research (OR) is a field that embodies such ideals by utilizing advanced quantitative tools to help make better decisions and create impact.

There are two necessary ingredients for delivering impact. The first is algorithms that can scale to satisfy the needs of real-world applications. With increased computational power and connectivity, the amount of collected data is increasing exponentially, with a Forbes report stating that over 90% of all data was generated in the last two years. The vast sea of data necessities that the algorithms utilized to deliver insights from the data also have to improve steadily in order run in reasonable time-frames. The second ingredient is the applications themselves. An algorithm could be theoretically ideal and yet not make an impact due to the lack of fitting applications. Only if both parts exist can the full impact of OR be realized.

This thesis is separated into exactly these two parts. In the first part of the thesis, we develop theory and optimization algorithms designed to meet the demands of modern data-driven analytics. In particular, we develop new stochastic algorithms to solve general Mixed Integer Non-Linear Optimization (MINLO) problems, and demonstrate its ability to improve scaling for the Interpretable Matrix Completion problem by up to $20x$ when compared to state-of-the-art algorithms. We further introduce new stochastic algorithms to solve the standard continuous Matrix Completion problem that improves scalability by up to $50x$. In the second part, we discuss one of the most serious real-world challenges in modern

history - the COVID-19 pandemic. We develop and implement new large-scale machine learning models to solve complex, time-sensitive real world problems in the pandemic, from predicting epidemiological spread, determining optimal non-pharmaceutical interventions, to selecting vaccine trial locations, and optimizing vaccine allocation.

The following sections provide a detailed illustration of the work and contributions in each of the two parts of the thesis.

1.1 Scalable Algorithms for Large Scale Data-Driven Analytics

In Chapter 1, we start with the classical low-rank matrix completion problem, where we aim to fill missing entries in a matrix $\boldsymbol{A} \in \mathbb{R}^{n \times m}$ that only have some entries with known values. To do so, it assumes that $\boldsymbol{A}$ is approximately low-rank with rank $k \ll n, m$: i.e. there exists two matrices $\boldsymbol{U} \in \mathbb{R}^{n \times k}, \boldsymbol{V} \in \mathbb{R}^{m \times k}$ such that $\boldsymbol{U}\boldsymbol{V}^T \approx \boldsymbol{A}$. $\boldsymbol{U}, \boldsymbol{V}$ can be interpreted as the k latent features for the row and column dimensions respectively. This non-convex problem has received a lot of attention as it characterizes a natural setting for generating recommendations to users. If we treat the n rows as users and m columns as products, then each entry A_{ij} indicates the interaction of user i with product j, of which most are "missing" as each user is likely to have only interacted with a small set of products. Furthermore, the low-rank assumption is intuitive as it stipulates the interaction behavior is characterized by a small number of latent factors, which correspond with human intuition. Therefore, there has been many previous works on creating algorithms for the matrix completion problem [e.g. 33, 126, 105, 93, 13, 100, 185]. However, till this date, scalability of matrix completion is still an important roadblock for its applications. The state-of-the-art methods on a modern server can only roughly solve matrices of sizes $10^5 \times 10^5$ in a few hours, while companies such as Netflix and Amazon have 10^7 users and need to send out recommendations in matters of seconds.

We show that although the problem is non-convex, it can be reformulated to an equivalent (non-convex) problem that can be solved directly with first-order methods to global optimal-

ity. We then construct a projected stochastic gradient descent algorithm, fastImpute, that takes advantage of this structure in the reformulated problem. Compared to state-of-the-art algorithms, fastImpute completes the matrices with over 80% smaller error and completes the matrix $50x$ faster in the largest instances. This enables the classical matrix completion problem to be also solved in minutes on a single server for $m \times n$ up to $10^6 \times 10^6$.

In Chapter 2, we then extend our work in matrix completion by introducing the problem of Interpretable Matrix Completion [22]. The Interpretable Matrix Completion problem is an extension of the classical (low-rank) matrix completion problem. Although matrix completion has an intuitive structure, it does not offer any insight on what the latent features of rows and columns mean. Interpretable Matrix Completion rectifies this by stipulating that the latent features need to be selected from a list of known features. This means we have feature matrices $\boldsymbol{B} \in \mathbb{R}^{m \times p}$ and/or $\boldsymbol{C} \in \mathbb{R}^{n \times p}$ such that $\boldsymbol{U}$ and $\boldsymbol{V}$ are submatrices of $\boldsymbol{B}$ and $\boldsymbol{C}$ respectively. As an example, if each row represents an individual, that means the latent individual features $\boldsymbol{U}$ need to be selected from a list of known individual features $\boldsymbol{B}$. This makes the latent features perfectly interpretable, and helps decision-makers understand the most important characteristics in both the row and column dimension that influence the quantity of interest in $\boldsymbol{A}$.

We illustrate that Interpretable Matrix Completion (IMC) can be reformulated as a binary convex optimization problem, which can be solved with the cutting-plane method. The cutting-plane method, also called the outer-approximation method, has been a popular method to solve mixed-integer optimization problems since its original introduction [57, 144]. It tackles problems of the form:

$$\begin{aligned} &\min_{\boldsymbol{z} \in \mathcal{Z}, \boldsymbol{\theta} \in \Theta} f(\boldsymbol{z}, \boldsymbol{\theta}) \\ &\text{subject to } g_i(\boldsymbol{z}, \boldsymbol{\theta}) \leq 0, \quad i \in [m], \end{aligned}$$

where $\boldsymbol{z}$ and $\boldsymbol{\theta}$ are integer and continuous variables constrained to be in sets $\mathcal{Z}$ and Θ, respectively, and f, g_i are convex. The method aims to generate the tangent plane of both the objective and the constraints at an initial feasible solution $(\boldsymbol{z}_0, \boldsymbol{\theta}_0)$, and solve the resultant mixed integer linear optimization (MILO) instead in order to find an optimal solution $(\boldsymbol{z}_1, \boldsymbol{\theta}_1)$.

Then, the process is repeated where the tangent plane around $(\boldsymbol{z}_1, \boldsymbol{\theta}_1)$ is added until the solution stops updating (with a tolerance of ϵ).

Although the cutting plane method can be utilized to solve IMC, as the data sizes grow, the cutting plane method face challenges in scaling as generating the full tangent planes involves evaluating the objective function and its derivative across all samples. We thus introduce a stochastic variant of the cutting plane algorithm that aims to greatly increase the scalability of the cutting plane problem by multiple orders of magnitude. Specifically, instead of aiming to generate the exact tangent plane in each iteration, we utilize an independent random subset of the data to generate an approximation of the tangent plane. A parallel analogy of this idea in continuous optimization is stochastic gradient descent, where only a subset of the samples is considered to calculate a gradient, resulting in large performance increases that enables the optimization of millions of variables (such as in a convolutional neural network). We illustrate that on both synthetic and real-world experiments, the stochastic cutting-plane algorithm is able to match or exceed current state-of-the-art methods on accuracy, while providing more than $20x$ better scaling behavior, enabling matrix completion to be solved for matrices of sizes up to $10^6 \times 10^6$, reaching the scale required for real-world deployment.

Then, in Chapter 4, we extend the stochastic cutting plane algorithm to a general class of mixed-integer non-linear optimization problems and provide guarantees that it converges to the true optimal solution with high probability and a relatively small penalty on the number of iterations needed. We illustrate that on multiple distinct problems (sparse regression, support vector machines and stochastic knapsack problems), stochastic cutting planes record an order-of-magnitude speedup compared to standard cutting planes and compare competitively with state-of-the art specialized algorithms.

1.2 Application of Data-Driven Analytics in the COVID-19 Pandemic

Starting from the initial Wuhan outbreak in December 2019, COVID-19 quickly spread around the world. As of October 13th, 2021, there is over 238 million confirmed cases of COVID-19 and the disease has taken over 4,500,000 lives. Due to the intensity and speed of how the pandemic hit, many organizations and governments were left under-prepared to plan and manage resources during it. To help institutions plan for the future, we develop a novel epidemiological model, DELPHI, to model the spread of COVID-19. DELPHI (Differential Equations Lead to Predictions of Hospitalizations and Infections), as recorded in Chapter 5, extends a classical SEIR model [99] to include many realistic effects that are critical in this pandemic, including deaths, under-detection, and changing governmental interventions, while ensuring the final model is still scalable [114]. Since its inception in late March, DELPHI has been one of the top 4 models consistently incorporated into the US Centers for Disease Control and Prevention's (CDC) core ensemble forecast [51] and have been utilized by various health and federal agencies including the Federal Reserve for pandemic planning. A major hospital system in the United States, Hartford Healthcare, planned its intensive care unit (ICU) capacity based on our forecasts. Its scalability has also allowed DELPHI to be applied to more than 200 regions worldwide on all 6 populated continents, successfully predicting the exponential rise in cases in areas such as South Africa, Russia, and the United Kingdom weeks before it occur.

One of the key innovations of DELPHI is the explicit and parametric characterization of government interventions. This allows us to understand the effect of different non-pharmaceutical interventions as they have been implemented in various regions while accounting for regional population characteristics including baseline infection rate and mortality percentage. We show that school closings and mass gathering restrictions were among the most effective measures in reducing the rate of infection during the early stages of the pandemic, though they carry a significant social burden. This analysis suggests that despite their extreme cost, these policies were effective in controlling the extreme growth in infec-

tions when other preventative measures (e.g. masks) and treatment options were still being evaluated and developed. As a further illustration of their outsized effect, we are able to demonstrate that had these restrictions been implemented just one week earlier, most – up to 90%– of the deaths in the early stages of the pandemic could have been avoided.

However, non-pharmaceutical interventions, although effective, could not end the pandemic on its own. As the pandemic progressed, thousands of researchers worldwide have focused on developing new medical solutions, which could be broadly separated into three categories: therapeutics, diagnostics, and vaccination. Out of these, vaccinations are the most promising in helping to end the pandemic, as it intervenes at the upstream level to immunize patients against the virus. Unsurprisingly, many pharmaceutical companies and governments have poured considerable resources into creating a COVID-19 vaccine. By Summer 2020, multiple vaccines, from Moderna, BioNTech, AstraZeneca, and Janssen, have passed Phase I/II trials. However, the most cost-intensive and time-consuming part of the development process - the Phase III trials was still lying ahead.

The Phase III trial is a large-scale randomized clinical trial designed to test the efficacy of the vaccine in real-world environments. Participants are recruited and randomly allocated to the placebo group or the experimental group, where only the experimental group is given the vaccine. Infections in both groups are monitored over time, and the vaccine is declared effective if at some point there exists a significant difference between the experimental and placebo group in terms of total infections.

Given the extraordinary humanitarian and economic cost of the pandemic, it is thus imperative for the Phase III trial to be completed as soon as possible. One of the key ways to reduce the trial time is by selecting trial locations with high incidence rates, so that the placebo group develops infections quicker, allowing an earlier comparison with the experimental group. However, trial locations need to be determined many months before the trial actually occurs. Given the frequency of new governmental interventions to control the pandemic, future infection rates are highly uncertain, so selecting good trial locations is an extremely difficult problem for COVID-19 vaccine trials.

Therefore, we augmented DELPHI to provide a scenario analysis tool that predicts how the pandemic would spread based on different possible future interventions. Then, working

with Janssen, we combined the DELPHI predictions and qualtitative insights about likely future policies to select the Phase III trial locations for Janssen's single-dose Ad26.Cov2.S vaccine candidate. In doing so, the final DELPHI-driven location selection achieved 90%+ accuracy in choosing sites with high COVID-19 incidence, and accelerated the Phase III trial time by 6-8 weeks while simultaneously reducing the trial size from 60,000 to 45,000 individuals. Areas of high COVID-19 incidence also are the most likely places to produce variants. Therefore, by utilizing DELPHI to identify hotspots, this enabled Janssen to have the only Phase III trial till date that is directly tested against important variants in Brazil and South Africa. The global applicability of the DELPHI model has also helped Janssen to produce the most location diverse COVID-19 trial till date. The results of these are also included in Chapter 4.

Unfortunately, discovering and developing a vaccine for COVID-19 was just the beginning - it takes months to produce, distribute, and deliver vaccines at scale. Since the vaccines cannot be made available immediately to everyone, policy makers need to make tough decisions on how to distribute the limited vaccines. A global consensus naturally emerged to prioritize to healthcare workers, other front line workers, and vulnerable populations such as older people and people with comorbidities. However, for the remaining general population, there was much more divergence. In the United States, the Biden administration proposed to allow the Federal Emergency Management Agency (FEMA) open up to 100 mass vaccination centers to speed up the vaccination of the general population. A critical question then arose: where to setup the 100 mass vaccination centers, and how do allocate a limited vaccine supply across them?

In Chapter 6, we answer this question with a novel data-driven optimization model that integrates the predictive DELPHI model into a prescriptive optimization model for vaccine allocation [21]. We first proposed an extension to DELPHI, named DELPHI-V, that captures the effect of vaccinations in the epidemiological dynamics. We then formulate an optimization model, referred to as DELPHI–V–OPT, which optimizes the vaccine distribution strategy (that is, the deployment of mass vaccination sites at the strategic level, and the subsequent allocation of vaccines at the tactical level) to minimize the death toll of the pandemic. From a technical standpoint, the DELPHI–V–OPT model relies on time discretization to

embed the system of ordinary differential equations governing the DELPHI–V dynamics into an optimization model. The model is formulated as a bilinear (non-convex) optimization model, due to the SEIR dynamics at the core of DELPHI–V in which the number of new cases is driven by the number of susceptible and infected people. To solve it efficiently in realistic large-scale settings, we propose a coordinate descent algorithm. Starting from a baseline solution, the algorithm iterates, until convergence, between optimizing the vaccine distribution strategy (for given dynamics of the pandemic) and simulating the dynamics of the pandemic (for a given vaccine distribution strategy).

Our results show that optimizing the locations of vaccinations carry significant impact on the vaccination campaign. Compared to reasonable benchmarks based on population count or epidemiological information (e.g. case counts), 100 optimized mass-vaccination locations saves 4,000, or 20% more lives over a three-month period in the United States. The Federal government eventually updated its policy so that Federal facilities take a supporting role while delegating most authority to state-driven vaccination efforts. Therefore, the full plan was not implemented and instead FEMA has opened just 24 facilities in 13 states. However, our recommendations influenced how FEMA allocated its resources, leading to a greater allocation of centers in states like Texas and Florida as compared to a population pro-rata basis, while under-weighting California.

Chapter 2

Fast Exact Matrix Completion: A Unified Optimization Framework for Matrix Completion

2.1 Introduction

Low-rank matrix completion is one of the most studied problems after its successful application in the Netflix Competition. It has been used in computer vision ([32]), signal processing ([95]), and control theory ([27]) to generate a completed matrix from partially observed entries, among several other areas. Given a data matrix $\boldsymbol{A} \in \mathbb{R}^{n \times m}$, the low-rank assumption assumes that rank($\boldsymbol{A}$) is small - in other words there are only a few, but still unknown, common linear factors that affect A_{ij}.

In recent years, as noted by [133], there has been a rise in interest for *inductive matrix completion*, where the common linear factors are chosen from a set of given vectors in the form of side information.

In this paper, we present an optimization based approach that improves upon the state of the art in matrix completion with and without side information. We next review the literature in both the general matrix completion and the inductive matrix completion areas.

Literature

General Matrix Completion

Matrix completion has been applied successfully for many tasks, including recommender systems [105], social network analysis [41] and clustering [37]. After [33] first proved a theoretical guarantee for the retrieval of the exact matrix under the nuclear norm convex relaxation, a lot of methods have focused on the nuclear norm problem (see [126], [13], [93], and [161] for examples). Alternative methods include alternating projections by [148] and Grassmann manifold optimization by [100]. There has also been work where the uniform distributional assumptions required by the theoretical guarantees are violated, such as [134] and [36].

Despite the non-convexity of the problem, many gradient-descent based approaches have also been proposed. Many works, including [105], [94], [185] and [96], utilize the Burer-Monteiro factorization ($\boldsymbol{A} = \boldsymbol{U}\boldsymbol{V}^T$, where $\boldsymbol{U} \in \mathbb{R}^{n\times k}$ and $\boldsymbol{V} \in \mathbb{R}^{m\times k}$) and conduct various forms (projected, stochastic, lifted, etc) of gradient descent on $\boldsymbol{U}$ and $\boldsymbol{V}$. Numerical experiments suggest that such algorithms can often converge to the global optimal solution despite their local nature.

Recently, a line of work, including [38], [75], and [121], investigated the reasons behind this uncanny efficiency of gradient descent on matrix completion. Collectively, they showed that for the case where $\boldsymbol{A}$ is positive semi-definite (PSD), gradient descent on the symmetric factorization $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{U}^T$ can converge linearly to the global optimum under restricted isometry conditions.

This work has also been reproduced in the non-PSD case considered here ($\boldsymbol{A} = \boldsymbol{U}\boldsymbol{V}^T$) if special regularization terms are added (see for example [160]).

Our work differs from previous work in one significant way: After writing $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{V}^T$, instead of performing gradient descent on both $\boldsymbol{U}$ and $\boldsymbol{V}$, we derive $\boldsymbol{U}$ as a function of $\boldsymbol{V}$, $g(\boldsymbol{V})$, and then directly perform gradient updates for $\boldsymbol{V}$, with a projection to a fixed-norm hypersphere $\|\boldsymbol{V}\|_2 = 1$ to ensure scaling invariance.

Using this formulation, we are able to prove convergence guarantees without special regularization terms on the objective. Numerical experiments show that we are able to

retrieve the matrix faster than the top performing methods discussed here.

Inductive Matrix Completion

Interest in inductive matrix completion intensified after [182] showed that given predictive side information, one only needs $O(\log n)$ samples to retrieve the full matrix. Thus, most of this work (see [182], [92], [64], [131]) have focused on the case in which the side information is assumed to be perfectly predictive so that the theoretical bound of $O(\log n)$ sample complexity [182] can be achieved. [40] explored the case in which the side information is corrupted with noise, while [152] and [154] incorporated nonlinear combination of factors into the side information. As pointed out by a recent article [133], there is a considerable lack of effort to introduce sparsity into inductive matrix completion, with [119], [155], [133] as examples. [22] introduced a convex binary formulation of the sparse inductive matrix completion problem, and constructed randomized algorithms for scaling.

Our work differs from the work in [119], [155], and [133] as it does not consider the heuristic convex relaxation of sparsity in the nuclear norm, but rather the exact sparse problem. [22] consider the problem where the low-rank factorization needs to be selected from features in the side information, while we allow the factorization to be any linear combination of the features in the side information. This greatly increases the flexibility of the algorithm, provides higher modeling power, and leads to stronger matrix recovery.

Contributions and Structure

Our contributions in this paper are as follows:

1. We reformulate the low-rank matrix completion problem, both with side information and without, as a separable optimization problem. We show that the general matrix completion problem is in fact a special case of the matrix completion problem with side information.

2. We propose a novel algorithm using projected gradient descent and Nesterov's accelerated gradient to solve the reformulated matrix completion problem. We prove that the algorithm is guaranteed to converge to the optimal solution under mild conditions.

3. We present computational results on both synthetic and real-world datasets that shows the algorithm outperforms current state-of-the-art methods in scalability and accuracy with and without side information. In particular, for the general matrix completion problem, we show the algorithm is about 15 times faster than the fastest algorithm available, while achieving on average a 75% decrease in error of retrieval on synthetic datasets.

The structure of the paper is as follows. In Section 2.2, we introduce the separable reformulation of the low-rank matrix completion problem. In Section 2.3, we introduce the base projected gradient descent method, projImpute. In Section 2.4, we introduce fastImpute, the stochastic version of projImpute designed for use with side information. We report on its computational complexity, derive theoretical results, and compare it with other algorithms. in Section 2.5, we report results on both synthetic and real-world datasets with side information using fastImpute and compare its performance with other algorithms. In Section 2.6, we show how the calculations can be further simplified when there is no side information. We report results on both synthetic and real-world datasets without side information using fastImpute and compare its performance with other algorithms. In Section 2.7, we provide our conclusions.

2.2 Reformulation of Matrix Completion

The classical matrix completion problem considers having a matrix $\boldsymbol{A} \in \mathbb{R}^{n\times m}$ in which $\Omega = \{(i,j) \mid A_{ij} \text{ is known}\}$ is the set of known values. We aim to recover a matrix $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{V}^T$ of rank k that minimizes the distance between $\boldsymbol{X}$ and $\boldsymbol{A}$ on the known entries $\boldsymbol{A}$:

$$\min_{\boldsymbol{X}} \frac{1}{nm} \sum_{(i,j)\in\Omega} (X_{ij} - A_{ij})^2 \quad \text{subject to} \quad \text{Rank}(\boldsymbol{X}) = k.$$

The problem we consider here is that for every column $j = 1, \ldots, m$, we have a given p-dimensional feature vector $\boldsymbol{B}_j$ with $p \geq k$ that contains the information we have on column j. In the Netflix example, column j corresponds to movie j, and thus the feature vector $\boldsymbol{B}_j$ includes information about the jth movie: Budget, Box Office revenue, IMDB rating,

etc. We represent all this side information with a matrix $\boldsymbol{B} \in \mathbb{R}^{m\times p}$. Given side data $\boldsymbol{B}$ we postulate that $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{S}^T\boldsymbol{B}^T$, where $\boldsymbol{U} \in \mathbb{R}^{n\times k}$ is the matrix of feature exposures, and

$$\boldsymbol{S} = \begin{pmatrix} s_{11} & s_{12} & \cdots & s_{1k} \\ s_{21} & s_{22} & \cdots & s_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ s_{p1} & s_{p2} & \cdots & s_{pk} \end{pmatrix} \in \mathbb{R}^{p\times k}.$$

Then, the matrix completion problem with side data $\boldsymbol{B}$ can be written as:

$$\min_{\boldsymbol{S}} \min_{\boldsymbol{U}} \frac{1}{nm} \sum_{(i,j)\in\Omega} (X_{ij} - A_{ij})^2 \quad \text{subject to} \quad \boldsymbol{X} = \boldsymbol{U}\boldsymbol{S}^T\boldsymbol{B}^T,\ \|\boldsymbol{S}\|_2 = 1.$$

The norm constraint on $\boldsymbol{S}$ reduces the space of feasible solutions without loss of generality, as the problem is invariant under the transformation $\boldsymbol{U} \to \boldsymbol{U}\boldsymbol{D}^{-1}$ and $\boldsymbol{S}^T \to \boldsymbol{D}\boldsymbol{S}^T$ for any diagonal matrix $\boldsymbol{D} \in \mathbb{R}^{k\times k}$ that is invertible. Throughout the paper $\|\cdot\|_2$ is the Frobenius norm. We note that since $\boldsymbol{S}$ is a $p \times k$ matrix, the rank of matrix $\boldsymbol{X}$ is indeed k. Further,we note that if $p = m$, and $\boldsymbol{B} = \boldsymbol{I}_m$, then the problem is reduced back to the general low-rank matrix completion problem with no feature information.

Similar to linear regression and for robustness purposes as shown in [19], we address in this paper the problem with a Tikhonov regularization term. Specifically, the matrix completion problem with side information and regularization we address is

$$\min_{\|\boldsymbol{S}\|_2=1} \min_{\boldsymbol{U}} \frac{1}{nm} \left(\sum_{(i,j)\in\Omega} (X_{ij} - A_{ij})^2 + \frac{1}{\gamma}\|\boldsymbol{U}\|_2^2 \right) \quad \text{subject to} \quad \boldsymbol{X} = \boldsymbol{U}\boldsymbol{S}^T\boldsymbol{B}^T, \tag{2.1}$$

where $\gamma > 0$ is a given parameter that controls the strength of the regularization term. We can reformulate Problem (2.1) as followed:

Proposition 1. *Problem (2.1) can be reformulated as a separable optimization problem:*

$$\min_{\|\boldsymbol{S}\|_2=1} c(\boldsymbol{S}) = \frac{1}{nm} \sum_{i=1}^{n} \overline{\boldsymbol{a}}_i \left(\boldsymbol{I}_m - \boldsymbol{V} \left(\frac{\boldsymbol{I}_k}{\gamma} + \boldsymbol{V}^T\boldsymbol{W}_i\boldsymbol{V} \right)^{-1} \boldsymbol{V}^T \right) \overline{\boldsymbol{a}}_i^T, \tag{2.2}$$

where $\boldsymbol{V} = \boldsymbol{BS}$, $\boldsymbol{W}_1, \cdots, \boldsymbol{W}_n \in \mathbb{R}^{m\times m}$ *are diagonal matrices:*

$$(\boldsymbol{W}_i)_{jj} = \begin{cases} 1, & (i,j) \in \Omega, \\ 0, & (i,j) \notin \Omega, \end{cases}$$

and $\overline{\boldsymbol{a}}_i = \boldsymbol{a}_i \boldsymbol{W}_i$, $i = 1 \ldots, n$, *so that* $\overline{\boldsymbol{a}}_i \in \mathbb{R}^{1\times m}$ *is the* i*th row of* $\boldsymbol{A}$ *with unknown entries taken to be 0.*

Proof. With the diagonal projection matrices $\boldsymbol{W}_i$ defined above, we can rewrite the sum in (2.1) over known entries of $\boldsymbol{A}$, $\sum_{(i,j)\in\Omega}(X_{ij} - A_{ij})^2$, as a sum over the rows of $\boldsymbol{A}$:

$$\sum_{i=1}^{n} \|(\boldsymbol{x}_i - \boldsymbol{a}_i)\boldsymbol{W}_i\|_2^2,$$

where $\boldsymbol{x}_i$ is the ith row of $\boldsymbol{X}$. Using $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{S}^T\boldsymbol{B}^T$, then $\boldsymbol{x}_i = \boldsymbol{u}_i\boldsymbol{S}^T\boldsymbol{B}^T$ where $\boldsymbol{u}_i$ is the ith row of $\boldsymbol{U}$. Moreover,

$$\|\boldsymbol{U}\|_2^2 = \sum_{i=1}^{n} \|\boldsymbol{u}_i\|_2^2.$$

Then, Problem (2.1) becomes:

$$\min_{\|\boldsymbol{S}\|_2=1} \min_{\boldsymbol{U}} \frac{1}{nm} \left(\sum_{i=1}^{n} \left(\|(\boldsymbol{u}_i\boldsymbol{S}^T\boldsymbol{B}^T - \boldsymbol{a}_i)\boldsymbol{W}_i\|_2^2 + \frac{1}{\gamma}\|\boldsymbol{u}_i\|_2^2 \right) \right).$$

We then notice that within the sum $\sum_{i=1}^{n}$ each row of $\boldsymbol{U}$ can be optimized separately, leading to:

$$\min_{\|\boldsymbol{S}\|_2=1} \frac{1}{nm} \left(\sum_{i=1}^{n} \min_{\boldsymbol{u}_i} \left(\|(\boldsymbol{u}_i\boldsymbol{S}^T\boldsymbol{B}^T - \boldsymbol{a}_i)\boldsymbol{W}_i\|_2^2 + \frac{1}{\gamma}\|\boldsymbol{u}_i\|_2^2 \right) \right). \tag{2.3}$$

The inner optimization problem $\min_{\boldsymbol{u}_i} \|(\boldsymbol{u}_i\boldsymbol{S}^T\boldsymbol{B}^T - \boldsymbol{a}_i)\boldsymbol{W}_i\|_2^2 + \frac{1}{\gamma}\|\boldsymbol{u}_i\|_2^2$ can be solved in closed form given $\boldsymbol{S}$, as it is a weighted linear regression problem with Tikhonov regularization. The closed form solution is:

$$\boldsymbol{a}_i\boldsymbol{W}_i(\boldsymbol{I}_m + \gamma\boldsymbol{W}_i\boldsymbol{B}\boldsymbol{S}\boldsymbol{S}^T\boldsymbol{B}^T\boldsymbol{W}_i^T)^{-1}\boldsymbol{W}_i\boldsymbol{a}_i^T = \overline{\boldsymbol{a}}_i(\boldsymbol{I}_m + \gamma\boldsymbol{W}_i\boldsymbol{B}\boldsymbol{S}\boldsymbol{S}^T\boldsymbol{B}^T\boldsymbol{W}_i)^{-1}\overline{\boldsymbol{a}}_i^T.$$

So Problem (2.3) can be simplified to:

$$\min_{\|\boldsymbol{S}\|_2=1} \frac{1}{nm}\left(\sum_{i=1}^{n}\overline{\boldsymbol{a}}_i(\boldsymbol{I}_m+\gamma\boldsymbol{W}_i\boldsymbol{B}\boldsymbol{S}\boldsymbol{S}^T\boldsymbol{B}^T\boldsymbol{W}_i^T)^{-1}\overline{\boldsymbol{a}}_i^T\right).$$

Then, let us define $\boldsymbol{V}=\boldsymbol{B}\boldsymbol{S}$, and write:

$$\begin{aligned}
&\frac{1}{nm}\left(\sum_{i=1}^{n}\overline{\boldsymbol{a}}_i(\boldsymbol{I}_m+\gamma\boldsymbol{W}_i\boldsymbol{B}\boldsymbol{S}\boldsymbol{S}^T\boldsymbol{B}^T\boldsymbol{W}_i^T)^{-1}\overline{\boldsymbol{a}}_i^T\right)\\
&=\frac{1}{nm}\left(\sum_{i=1}^{n}\overline{\boldsymbol{a}}_i(\boldsymbol{I}_m+\gamma\boldsymbol{W}_i\boldsymbol{V}\boldsymbol{V}^T\boldsymbol{W}_i^T)^{-1}\overline{\boldsymbol{a}}_i^T\right)\\
&=\frac{1}{nm}\sum_{i=1}^{n}\overline{\boldsymbol{a}}_i\left(\boldsymbol{I}_m-\boldsymbol{V}\left(\frac{\boldsymbol{I}_k}{\gamma}+\boldsymbol{V}^T\boldsymbol{W}_i\boldsymbol{V}\right)^{-1}\boldsymbol{V}^T\right)\overline{\boldsymbol{a}}_i^T,
\end{aligned}$$

which is the needed form in (2.2). The second equality comes from the matrix inversion lemma, as derived in [177]:

Lemma 1. *For matrices $\boldsymbol{U}\in\mathbb{R}^{n\times k}$ and $\boldsymbol{V}\in\mathbb{R}^{k\times n}$, we have the following equivalence:*

$$(\boldsymbol{I}_n+\boldsymbol{U}\boldsymbol{V})^{-1}=\boldsymbol{I}_n-\boldsymbol{U}(\boldsymbol{I}_k+\boldsymbol{V}\boldsymbol{U})^{-1}\boldsymbol{V}.$$

□

2.3 A Gradient Descent Algorithm

In this section, we describe a gradient descent algorithm to solve the separable optimization problem (2.2), and examine its properties. First, we introduce

$$\alpha_i(\boldsymbol{S})=\overline{\boldsymbol{a}}_i\boldsymbol{\gamma}_i(\boldsymbol{S})=\overline{\boldsymbol{a}}_i\left(\boldsymbol{I}_m-\boldsymbol{V}\left(\frac{\boldsymbol{I}_k}{\gamma}+\boldsymbol{V}^T\boldsymbol{W}_i\boldsymbol{V}\right)^{-1}\boldsymbol{V}^T\right)\overline{\boldsymbol{a}}_i^T,\quad i=1,\ldots,n. \tag{2.4}$$

Here $\alpha_i(\boldsymbol{S})$ is a scalar and $\boldsymbol{\gamma}_i(\boldsymbol{S})$ is a $m\times 1$ vector. Then the function $c(\boldsymbol{S})$ in (2.2) can be expressed as

$$c(\boldsymbol{S})=\frac{1}{nm}\sum_{i=1}^{n}\alpha_i(\boldsymbol{S})=\frac{1}{nm}\sum_{i=1}^{n}\overline{\boldsymbol{a}}_i\boldsymbol{\gamma}_i(\boldsymbol{S}).$$

Using this notation, we then can calculate the derivative of $c(\boldsymbol{S})$:

Lemma 2.

$$\nabla c(\boldsymbol{S}) = -\frac{2\gamma}{n}\sum_{i=1}^{n}\boldsymbol{B}^T\boldsymbol{\gamma}_i(\boldsymbol{S})\boldsymbol{\gamma}_i(\boldsymbol{S})^T\boldsymbol{V}. \tag{2.5}$$

Proof. By the standard result of the derivative of the inverse matrix, we have:

$$\begin{aligned}\nabla\alpha_i(\boldsymbol{S}) &= \nabla\overline{\boldsymbol{a}}_i\boldsymbol{\gamma}_i(\boldsymbol{S}) \\ &= -2\gamma\boldsymbol{B}^T\boldsymbol{\gamma}_i(\boldsymbol{S})\overline{\boldsymbol{a}}_i(\boldsymbol{I}_m + \gamma\boldsymbol{W}_i\boldsymbol{V}\boldsymbol{V}^T\boldsymbol{W}_i^T)^{-1}\boldsymbol{B}\boldsymbol{S} \\ &= -2\gamma\boldsymbol{B}^T\boldsymbol{\gamma}_i(\boldsymbol{S})\boldsymbol{\gamma}_i(\boldsymbol{S})^T\boldsymbol{V}.\end{aligned}$$

Then, since $c(\boldsymbol{S}) = \frac{1}{nm}\sum_{i=1}^{n}\alpha_i(\boldsymbol{S})$, we have the required result. □

Using Lemma 2, we apply a projected gradient descent algorithm (as discussed in [16]) on the hypersphere $\|\boldsymbol{S}\|_2 = 1$ and obtain in Algorithm 1. (Note that $\boldsymbol{0}^{p\times k}$ denotes a zero matrix of dimension $p \times k$).

Algorithm 1 Gradient Descent algorithm for matrix completion with side information.

procedure PROJIMPUTE($\boldsymbol{A}, \boldsymbol{B}, k, \theta, t_{max}$) ▷ $\boldsymbol{A} \in \mathbb{R}^{n\times m}$ the masked matrix, $\boldsymbol{B} \in \mathbb{R}^{p\times m}$ the feature matrix, k the desired rank, θ the step size, and t_{max} the number of steps
  $\boldsymbol{S}_1 \leftarrow$ random initial matrix with $\|\boldsymbol{S}_1\|_2 = 1$ ▷ Randomized Start
  $\eta_0 \leftarrow \infty$ ▷ Initialize objective value
  $\eta_1, \boldsymbol{G}_1 \leftarrow c(\boldsymbol{S}_1), \nabla c(\boldsymbol{S}_1)$ ▷ Initialize objective value and gradient
  $\nabla\tilde{\boldsymbol{S}}_1 \leftarrow \boldsymbol{0}^{p\times k}$ ▷ Initialize accelerated gradient
  while $t < t_{max}$ **do** ▷ While we have not reached t_{max} iterations
    $\nabla\tilde{\boldsymbol{S}}_{t+1} = \boldsymbol{G}_t + \frac{t-1}{t+2}\nabla\tilde{\boldsymbol{S}}_t$ ▷ Nesterov accelerated gradient update step
    $\overline{\nabla\boldsymbol{S}_{t+1}} = -\nabla\tilde{\boldsymbol{S}}_{t+1} + (\nabla\tilde{\boldsymbol{S}}_{t+1}\cdot\boldsymbol{S}_t)\boldsymbol{S}_t$ ▷ Project gradient to tangent plane of $\boldsymbol{S}_t$
    $\boldsymbol{S}_{t+1} \leftarrow \boldsymbol{S}_t\cos\theta + \frac{\overline{\nabla\boldsymbol{S}_{t+1}}}{\|\overline{\nabla\boldsymbol{S}_{t+1}}\|_2}\sin\theta$ ▷ Update $\boldsymbol{S}_t$ by projected gradient
    $\eta_{t+1}, \boldsymbol{G}_{t+1} \leftarrow c(\boldsymbol{S}_{t+1}), \nabla c(\boldsymbol{S}_{t+1})$ ▷ Update the new cost and derivative.
    $t \leftarrow t+1$
  end while
  $\boldsymbol{S}^* \leftarrow \boldsymbol{S}_t$
  $i \leftarrow 1$
  for $i < n$ **do**
    $\boldsymbol{a}_i \leftarrow \overline{\boldsymbol{a}}_i\boldsymbol{B}\boldsymbol{S}^*(\boldsymbol{B}\boldsymbol{S}^*\boldsymbol{W}_i\boldsymbol{B}\boldsymbol{S}^{*T}\boldsymbol{B}^T)^{-1}\boldsymbol{S}^{*T}\boldsymbol{B}^T$ ▷ Calculate the final $\boldsymbol{A}$ matrix
  end for
  return $\boldsymbol{A}$ ▷ Return the filled matrix $\boldsymbol{A}$
end procedure

We explain the reasoning behind some key steps of Algorithm 1.

Nesterov Step This is Step 7 of Algorithm 1. We update the gradient with the accelerated formula in [135] by adding the gradient of the current step to $\frac{t-1}{t+2}$ times the previous gradient, which introduces damping in the resulting gradient and enables faster convergence.

Projection and Update Steps This is Step 8 and 9 of Algorithm 1. There are two reasons why we are optimizing over the $\|\boldsymbol{S}\|_2 = 1$: (a) as explained, rescaling $\boldsymbol{S}$ does not change the objective, so the restriction can guarantee a smaller set of feasible solutions; (b) such restriction enables us to effectively approximate the objective value, and its derivative using random sampling, see Theorem 1.

Because we are optimizing on the hypersphere $\|\boldsymbol{S}\|_2 = 1$, our gradient updates need to be projected to the tangent plane of the sphere at the current point $\boldsymbol{S}_t$. Thus, we project the raw gradient, $\nabla\tilde{\boldsymbol{S}}_{t+1}$ onto the tangent plane to get $\overline{\nabla\boldsymbol{S}_{t+1}}$. Then the update step ensures the norm of the updated $\boldsymbol{S}$ is renormalized to 1.

Note that updating the gradient on the hypersphere is a rotation on the great circle formed by $\boldsymbol{S}_t$ and $\overline{\nabla\boldsymbol{S}_{t+1}}$, so the update formula is $\boldsymbol{S}_t\cos\theta + \frac{\overline{\nabla\boldsymbol{S}_{t+1}}}{\|\overline{\nabla\boldsymbol{S}_{t+1}}\|_2}\sin\theta$.

2.3.1 Discussion on Computational Complexity

There are two key computational steps of the algorithm - Step 10, where the cost and the derivative is calculated in every gradient update, and Step 16 where the final matrix is calculated. We next derive the asymptotic complexity of such steps.

Proposition 2. *The computation complexity of Step 10 in projImpute is*

$$O\left(|\Omega|\left(p + k^2 + \frac{pk + k^3}{m\alpha}\right)\right),$$

where $|\Omega|$ *is the number of samples given in the original matrix* $\boldsymbol{A}$ *and* $\alpha = \frac{|\Omega|}{mn}$ *is the percentage not missing. The computational complexity of Step 16 is*

$$O\left(|\Omega|\left(k^2 + \frac{k}{\alpha} + \frac{k^3}{m\alpha}\right)\right).$$

Proof. We analyze Steps 10 and 16 separately.

Computational Complexity of Step 10

Recall we have that:

$$\alpha_i(\boldsymbol{S}) = \overline{\boldsymbol{a}}_i \boldsymbol{\gamma}_i(\boldsymbol{S}) = \overline{\boldsymbol{a}}_i \left(\boldsymbol{I}_m - \boldsymbol{V} \left(\frac{\boldsymbol{I}_k}{\gamma} + \boldsymbol{V}^T \boldsymbol{W}_i \boldsymbol{V} \right)^{-1} \boldsymbol{V}^T \right) \overline{\boldsymbol{a}}_i^T, \quad i = 1, \ldots, n,$$

First let us denote m_i as the number of non-zero entries of $\boldsymbol{W}_i$. This is the number of known entries per row. Then define $\boldsymbol{V}_{W_i} \in \mathbb{R}^{m_i \times k}$ as the matrix of $\boldsymbol{W}_i \boldsymbol{V}$ after removing the all-zero columns, as illustrated below:

$$\begin{aligned} \boldsymbol{W}_i \boldsymbol{V} &= \mathrm{Diag}(1, 0, 1, \cdots, 0) \times \begin{pmatrix} — & \boldsymbol{v}_1 & — \\ — & \boldsymbol{v}_2 & — \\ — & \boldsymbol{v}_3 & — \\ \vdots & \vdots & \vdots \\ — & \boldsymbol{v}_k & — \end{pmatrix} \\ &= \begin{pmatrix} — & \boldsymbol{v}_1 & — \\ — & \boldsymbol{0} & — \\ — & \boldsymbol{v}_3 & — \\ \vdots & \vdots & \vdots \\ — & \boldsymbol{0} & — \end{pmatrix} \rightsquigarrow \begin{pmatrix} — & \boldsymbol{v}_1 & — \\ — & \boldsymbol{v}_3 & — \\ \vdots & \vdots & \vdots \end{pmatrix} := \boldsymbol{V}_{W_i}. \end{aligned} \tag{2.6}$$

Note that $\boldsymbol{V}_{W_i}$ can be created efficiently through subsetting, and its creation does not impact the asymptotic running time. Then similarly, we denote $\boldsymbol{a}_{W_i} \in \mathbb{R}^{1 \times m_i}$ as $\boldsymbol{a}\boldsymbol{W}_i$ with all the zero elements removed. Then we have the following equality, using (2.4):

$$\begin{aligned} c(\boldsymbol{S}) &= \frac{1}{nm} \sum_{i=1}^{n} \overline{\boldsymbol{a}}_i \boldsymbol{\gamma}_i(\boldsymbol{S}) \\ &= \frac{1}{nm} \sum_{i=1}^{n} \overline{\boldsymbol{a}}_i \left(\boldsymbol{I}_m - \boldsymbol{V} \left(\frac{\boldsymbol{I}_k}{\gamma} + \boldsymbol{V}^T \boldsymbol{W}_i \boldsymbol{V} \right)^{-1} \boldsymbol{V}^T \right) \overline{\boldsymbol{a}}_i^T \end{aligned}$$

$$= \frac{1}{nm} \sum_{i=1}^{n} \boldsymbol{a}_{W_i} \left(\boldsymbol{I}_{m_i} - \boldsymbol{V}_{W_i} \left(\frac{\boldsymbol{I}_k}{\gamma} + \boldsymbol{V}_{W_i}^T \boldsymbol{V}_{W_i} \right)^{-1} \boldsymbol{V}_{W_i}^T \right) \boldsymbol{a}_{W_i}^T \tag{2.7}$$
$$= \frac{1}{nm} \sum_{i=1}^{n} \boldsymbol{a}_{W_i} \boldsymbol{a}_{W_i}^T - \boldsymbol{a}_{W_i} \boldsymbol{V}_{W_i} \left(\frac{\boldsymbol{I}_k}{\gamma} + \boldsymbol{V}_{W_i}^T \boldsymbol{V}_{W_i} \right)^{-1} \boldsymbol{V}_{W_i}^T \boldsymbol{a}_{W_i}^T.$$

The equality (2.7) follows directly from the definition of $\boldsymbol{VW}_i$ and $\boldsymbol{a}_{W_i}$ as all the removed elements/columns are zero.

Then, the complexity of each term in the sum (2.7) is $O(m_i k^2 + k^3)$. Summing over n terms and noting that:

$$\sum_{i=1}^{n} n m_i = |\Omega|,$$

we obtain that the full complexity of the objective function is thus

$$O(|\Omega| k^2 + n k^3) = O\left(|\Omega| \left(k^2 + \frac{1}{m\alpha} k^3 \right) \right).$$

Now let us further define:

$$\boldsymbol{\gamma}_{W_i}(\boldsymbol{S}) = \left(\boldsymbol{I}_{m_i} - \boldsymbol{V}_{W_i} \left(\frac{\boldsymbol{I}_k}{\gamma} + \boldsymbol{V}_{W_i}^T \boldsymbol{V}_{W_i} \right)^{-1} \boldsymbol{V}_{W_i}^T \right) \boldsymbol{a}_{W_i}^T, \tag{2.8}$$

and $\boldsymbol{B}_{W_i}$ in the same fashion as defined in (2.6). Then we can similarly show that the derivative (2.5) is equivalent to the following expression:

$$\nabla c(\boldsymbol{S}) = -\frac{2\gamma}{n} \sum_{i=1}^{n} \boldsymbol{B}^T \boldsymbol{\gamma}_i(\boldsymbol{S}) \boldsymbol{\gamma}_i(\boldsymbol{S})^T \boldsymbol{V} \tag{2.9}$$
$$= -\frac{2\gamma}{n} \sum_{i=1}^{n} \boldsymbol{B}_{W_i}^T \boldsymbol{\gamma}_{W_i}(\boldsymbol{S}) \boldsymbol{\gamma}_{W_i}(\boldsymbol{S})^T \boldsymbol{V}_{W_i}. \tag{2.10}$$

The matrix multiplication in the final expression (2.10) has computational complexity of $O(m_i(p+k) + pk)$ (every individual term has been previously calculated), so summing over n terms the computational complexity is

$$O\left(|\Omega| \left(p + k + \frac{pk}{m\alpha} \right) \right).$$

Thus, the computational complexity of the entire step is:

$$O\left(|\Omega|\left(p + k^2 + \frac{pk + k^3}{m\alpha}\right)\right).$$

Computational Complexity of Step 16

Recall that Step 16 of projImpute does the following calculation for every $i \in \{1, \cdots, n\}$:

$$\boldsymbol{a}_i \leftarrow \overline{\boldsymbol{a}}_i \boldsymbol{B}\boldsymbol{S}^*(\boldsymbol{B}\boldsymbol{S}^*\boldsymbol{W}_i\boldsymbol{B}\boldsymbol{S}^{*T}\boldsymbol{B}^T)^{-1}\boldsymbol{S}^{*T}\boldsymbol{B}^T, \tag{2.11}$$

where $\boldsymbol{S}^*$ is the imputed $\boldsymbol{S}$. Now define $\boldsymbol{V}^* = \boldsymbol{B}\boldsymbol{S}^*$, and define $\boldsymbol{V}^*_{W_i}$ in the same fashion as defined in (2.6). Then we have that:

$$\begin{aligned}\overline{\boldsymbol{a}}_i \boldsymbol{B}\boldsymbol{S}^*(\boldsymbol{B}\boldsymbol{S}^*\boldsymbol{W}_i\boldsymbol{B}\boldsymbol{S}^{*T}\boldsymbol{B}^T)^{-1}\boldsymbol{S}^{*T}\boldsymbol{B}^T &= \overline{\boldsymbol{a}}_i \boldsymbol{V}^*(\boldsymbol{V}^{*T}\boldsymbol{W}_i\boldsymbol{V}^*)^{-1}\boldsymbol{V}^{*T} \\ &= \boldsymbol{a}_{W_i}\boldsymbol{V}^*(\boldsymbol{V}^{*T}_{W_i}\boldsymbol{V}^*_{W_i})^{-1}\boldsymbol{V}^{*T}_{W_i}. \end{aligned} \tag{2.12}$$

The final expression (2.12) has a matrix multiplication complexity of $O(km_i + k^2m_i + k^3 + mk)$, so summing over n samples the total computational complexity of Step 16 is:

$$O\left(|\Omega|\left(k^2 + \frac{k}{\alpha} + \frac{k^3}{m\alpha}\right)\right).$$

□

Using these results we can easily arrive at the following corollary:

Corollary 1. *The projImpute algorithm terminates in t_{max} number of steps. Furthermore, it has complexity*

$$O\left(|\Omega|\left(p + k^2 + \frac{pk + k^3}{m\alpha}\right)\right).$$

Proof. The computational complexity follows immediately from Proposition 2. □

Note that with $p \gg k$ sufficiently large, the computational complexity of Step 10 dominates that of Step 16. In particular, such step scales scales linearly in m, n, and p, which is

undesirable when all of these are large, as it is commonly in real-world settings. In the next section, we design an algorithm that reduces the computational complexity on Step 10.

2.4 fastImpute: An Adaptive Stochastic Projected Gradient Descent Algorithm for Matrix Completion

In the previous section, we introduced the projImpute algorithm, which conducted full projected gradient updates for every step $t \in \{1, \cdots, t_{max}\}$ using all n rows and m columns. However, it is computationally prohibitively expensive, especially when n, m, p is large.

In this section, we introduce fastImpute that uses random sampling of rows and columns to estimate the gradient update at step t. Specifically, at each step, we randomly select n_0 rows and *for each row*, we randomly select m_0 columns to calculate the sampled derivative for that row. Let $[n_t]$ denote the random set of numbers from $\{1, \cdots, n\}$ of size n_0 at step t, and $[m_t^i]$ denote a set of numbers from $\{1, \cdots, m\}$ of size m_0 at step t corresponding to the random sample for row i. Then the objective function evaluated with rows $[n_t]$ and columns $[m_t^i]$ is:

$$c_t(\boldsymbol{S}) = \frac{1}{n_0 m_0} \sum_{i \in [n_t]} \overline{\boldsymbol{a}}_{i m_t^i} \left(\boldsymbol{I}_{m_t^i} - \boldsymbol{V}_{m_t^i} \left(\frac{\boldsymbol{I}_k}{\gamma} + \boldsymbol{V}_{m_t^i}^T \boldsymbol{W}_{i m_t^i} \boldsymbol{V}_{m_t^i} \right)^{-1} \boldsymbol{V}_{m_t^i}^T \right) \overline{\boldsymbol{a}}_{i m_t^i}^T,$$

where $\boldsymbol{V}_{m_t^i}$ is the submatrix of $\boldsymbol{V}$ formed with $[m_t^i]$ columns, and similarly with other terms. We use the notation m_t^i to explicitly indicate that the columns are selected for each row $i \in [n_t]$ and independent across rows. Similarly, using Lemma 2, the derivative evaluated with rows $[n_t]$ and columns $[m_t^i]$ is:

$$\begin{aligned} \nabla c_t(\boldsymbol{S}) = \frac{1}{n_0 m_0} \sum_{i \in [n_t]} & -2\gamma \boldsymbol{B}_{m_t^i}^T \left(I_{m_t^i} - \boldsymbol{V}_{m_t^i} \left(\frac{\boldsymbol{I}_k}{\gamma} + \boldsymbol{V}_{m_t^i}^T \boldsymbol{W}_{i m_t^i} \boldsymbol{V}_{m_t^i} \right)^{-1} \boldsymbol{V}_{m_t^i}^T \right) \overline{\boldsymbol{a}}_{i m_t^i}^T \\ & \times \overline{\boldsymbol{a}}_{i m_t^i} \left(I_{m_t^i} - \boldsymbol{V}_{m_t^i} \left(\frac{\boldsymbol{I}_k}{\gamma} + \boldsymbol{V}_{m_t^i}^T \boldsymbol{W}_{i m_t^i} \boldsymbol{V}_{m_t^i} \right)^{-1} \boldsymbol{V}_{m_t^i}^T \right)^T \boldsymbol{V}_{m_t^i}. \end{aligned}$$

Algorithm 2 Gradient Descent algorithm for matrix completion with side information.

procedure FASTIMPUTE($\boldsymbol{A}, \boldsymbol{B}$,$k$,$\theta$, t_{max}) ▷ $\boldsymbol{A} \in \mathbb{R}^{n \times m}$ the masked matrix, $\boldsymbol{B} \in \mathbb{R}^{p \times m}$ the feature matrix, k the desired rank, θ the step size, and t_{max} the number of gradient steps
  $t \leftarrow 1$
  $\alpha \leftarrow \frac{|\Omega|}{mn}$ ▷ Define existing percentage of $\boldsymbol{A}$. $|\Omega|$ is the set of non-zero entries in $\boldsymbol{A}$
  $\boldsymbol{S}_1 \leftarrow$ random initial matrix with $\|\boldsymbol{S}_1\|_2 = 1$ ▷ Randomized Start
  $\eta_0 \leftarrow \infty$ ▷ Initialize objective value
  $q_1 \leftarrow 0$ ▷ Initialize counter for non-improving steps
  $m_0 \leftarrow \min(2p, m)$ ▷ Define initial gradient update size for columns
  $n_0 \leftarrow \left\lfloor \frac{k\sqrt{nm}\log(\sqrt{nm})}{8m_0\alpha} \right\rfloor$ ▷ Define initial gradient update size for rows
  $[n_0] \subset \{1, \cdots, n\}$ ▷ Initialize rows selected
  for $i \in [n_0]$ **do**
    $[m_0^i] \subset \{1, \cdots, m\}$ ▷ Initialize columns selected
  end for
  $\eta_1, \boldsymbol{G}_1 \leftarrow c_t(\boldsymbol{S}_1), \nabla c_t(\boldsymbol{S}_1)$ ▷ Initialize objective value and gradient
  $\nabla\tilde{\boldsymbol{S}}_1 \leftarrow \boldsymbol{0}^{k \times p}$ ▷ Initialize accelerated gradient
  while $t < t_{max}$ **do** ▷ While we have not reached t_{max} iterations
    $[n_t] \subset \{1, \cdots, n\}$ ▷ Select new rows
    for $i \in [n_t]$ **do**
      $[m_t^i] \subset \{1, \cdots, m\}$ ▷ Select new columns
    end for
    $\nabla\tilde{\boldsymbol{S}}_{t+1} = \boldsymbol{G}_t + \frac{t-1}{t+2}\nabla\tilde{\boldsymbol{S}}_t$ ▷ Nesterov accelerated gradient update step
    $\overline{\nabla \boldsymbol{S}_{t+1}} = -\nabla\tilde{\boldsymbol{S}}_{t+1} + (\nabla\tilde{\boldsymbol{S}}_{t+1} \cdot \boldsymbol{S}_t)\boldsymbol{S}_t$ ▷ Project gradient to the tangent plane of $\boldsymbol{S}_t$
    $\boldsymbol{S}_{t+1} \leftarrow \boldsymbol{S}_t \cos\theta + \frac{\overline{\nabla \boldsymbol{S}_{t+1}}}{\|\overline{\nabla \boldsymbol{S}_{t+1}}\|_2}\sin\theta$ ▷ Update $\boldsymbol{S}_t$ based on projected gradient
    $\eta_{t+1}, \boldsymbol{G}_{t+1} \leftarrow c_t(\boldsymbol{S}_{t+1}), \nabla c_t(\boldsymbol{S}_{t+1})$ ▷ Update the cost and derivative.
    $t \leftarrow t + 1$
  end while
  $\boldsymbol{S}^* \leftarrow \boldsymbol{S}_t$
  $i \leftarrow 1$
  for $i < n$ **do**
    $\boldsymbol{a}_i \leftarrow \overline{\boldsymbol{a}}_i \boldsymbol{B}\boldsymbol{S}^*(\boldsymbol{B}\boldsymbol{S}^*\boldsymbol{W}_i\boldsymbol{B}\boldsymbol{S}^{*T}\boldsymbol{B}^T)^{-1}\boldsymbol{S}^{*T}\boldsymbol{B}^T$ ▷ Calculate the final $\boldsymbol{A}$ matrix
  end for
  return $\boldsymbol{A}$ ▷ Return the filled matrix $\boldsymbol{A}$
end procedure

We then present the algorithm as Algorithm 2. It is the same procedure as projImpute, except with stochastic gradient updates of n_0 rows and m_0 columns. For this algorithm to work, we need the stochastic gradient to be "close" to the true gradient, which is true using a theorem adapted from [22]:

Theorem 1. *Let $\boldsymbol{A}$ be a partially known matrix, $\boldsymbol{B}$ a known feature matrix, and $\boldsymbol{W}_i$ as defined in Proposition 1. Then, with probability at least $1-\epsilon$, for any t, and any $\boldsymbol{S}$ satisfying $\|\boldsymbol{S}\| = 1$, we have:*

$$|c_t(\boldsymbol{S}) - c(\boldsymbol{S})| \leq \sqrt{\frac{Ak \log\left(\frac{k}{\epsilon}\right)}{n_0}},$$
$$\|\nabla c_t(\boldsymbol{S}) - \nabla c(\boldsymbol{S})\|_2 \leq \sqrt{\frac{Bk \log\left(\frac{k}{\epsilon}\right)}{n_0}}.$$

Inverting the statements, we have that:

$$\mathbb{P}(|c_t(\boldsymbol{S}) - c(\boldsymbol{S})| > \epsilon) \leq k e^{-\frac{n_0 \epsilon^2}{Ak}},$$
$$\mathbb{P}(\|\nabla c_t(\boldsymbol{S}) - \nabla c(\boldsymbol{S})\| > \epsilon) \leq k e^{-\frac{n_0 \epsilon^2}{Bk}}.$$

Theorem 1 shows that the probability the stochastic objective value and its derivative is ϵ away from the true value is exponentially vanishing with increasing n_0. This property affects how we choose n_0 and m_0, which is discussed in Section 2.4.2.

In the next section, we show that fastImpute recovers the true solution for matrix completion under relatively mild conditions.

2.4.1 Convergence Rates and Guarantees

In this section, we derive results that show Algorithm 2 converges to the global minimum of $c(\boldsymbol{S})$. for the case $1/\gamma = 0$ and without side information ($p = m$ and $\boldsymbol{B} = \boldsymbol{I}_m$). Under such assumptions, our objective can be written as:

$$c(\boldsymbol{S}) = \frac{1}{nm} \sum_{i=1}^{n} \overline{\boldsymbol{a}}_i \left(\boldsymbol{I}_m - \boldsymbol{S} \left(\boldsymbol{S}^T \boldsymbol{W}_i \boldsymbol{S} \right)^{-1} \boldsymbol{S}^T \right) \overline{\boldsymbol{a}}_i^T.$$

The analysis can be easily extended to the case where $\boldsymbol{B}$ is a general feature matrix, at the expense of increased notational complexity. The analysis below also holds for $\frac{1}{\gamma} < \epsilon_0$ for some sufficiently small ϵ_0 with minimal changes. To facilitate the theoretical analysis, we would further assume that the final projected gradient is perturbed by some asymptotically

vanishing Gaussian noise (immediately before Step 22):

$$\overline{\nabla \boldsymbol{S}_{t+1}} \leftarrow \overline{\nabla \boldsymbol{S}_{t+1}} + O\left(\frac{1}{\sqrt{n_0 m}}\right) \boldsymbol{E}$$

Where $\boldsymbol{E} \in \mathbb{R}^{k \times p}$ has iid entries with the standard normal distribution. As n_0 and m is large, this does not significantly change the resultant gradient, and numerical experiments suggest that this has no appreciable impact on the final solution.

Over the last few years, various researchers have proven related results using different formulations for matrix completion (see [75, 121] for the case where the recovered matrix is semi-definite, and [185] for the general case using a lifting formulation). The proofs follow a two-step process:

- Prove the proposed algorithm converges to a local minimum of the objective function.
- Prove all local minima of the objective function are global minima (equivalently, the objective function has "no spurious local minima"). This condition has been shown to be true for various formulations of nonconvex low rank problems, as explored in e.g. [74].

In our new formulation of matrix completion, we show how proving the first result is equivalent to checking the "strict saddle" condition on the objective function introduced in [73]. We then prove the objective function $c(\boldsymbol{S})$ does indeed satisfy the strict saddle condition, and moreover has no spurious local minima.

Before we layout our detailed results, we need to specify a random sampling model for the known entries of $\boldsymbol{A}$. Generally in the literature (see e.g. [34, 96, 75] for examples), the uniform sampling model is used, where every element is assumed to be present independently with a probability r_0. To simplify the proof, we specify a row-sampling model: For every row $\boldsymbol{a}_i$ of $\boldsymbol{A}$, we randomly sample (without replacement) l out of m entries to be known. Thus, each entry has a probability of $r = \frac{l}{m}$ being selected. By results in [34] and [81], it is known that convergence results under row-sampling models with sampling rate r hold true under the uniform sampling model with a rate $r_0 = O(\log n)r$, where the factor $\log n$ is essentially due to the coupon collector effect (for more discussion please see [34]).

We next show that proving Algorithm 2 converges to a local minimum is equivalent to verifying $c(\boldsymbol{S})$ satisfies the "strict saddle" condition. We first formally define such concept, as introduced in [73]:

Definition 1. *A function $f : \mathbb{R}^d \to \mathbb{R}$ is (θ, ζ, η)-strict saddle if for every $\boldsymbol{x}$, at least one of the following hold for constants $\theta, \zeta, \eta > 0$:*

1. $\|\nabla f(\boldsymbol{x})\| \geq \theta$.
2. $\lambda_{min}(\nabla^2 f(\boldsymbol{x})) \leq -\zeta$.
3. *There exists a local minimum $\boldsymbol{x}^*$ such that $\|\boldsymbol{x} - \boldsymbol{x}^*\| \leq \eta$.*

Using this definition, the following theorem from [63] (see also [50, 73]) establishes that a stochastic gradient algorithm converges to a local minimum if the function f being optimized over satisfies the strict saddle condition:

Theorem 2. *Assume that $f(\boldsymbol{x})$ is a function of $\boldsymbol{x} \in \mathbb{R}^d$ that is a (θ, ζ, η)-strict saddle. Then, a stochastic gradient descent algorithm with stochastic gradient $\tilde{\nabla} f(\boldsymbol{x})$ and added Gaussian noise $\frac{\sigma}{\sqrt{d}}\boldsymbol{\epsilon}$ where $\boldsymbol{\epsilon} \sim N(0, \boldsymbol{I}_d)$ converges in $O\left(\text{poly}\left(\frac{1}{\delta}\right)\right)$ iterations to a point δ close to a local minimum, where we have:*

$$\mathbb{P}(\|\tilde{\nabla} f(\boldsymbol{x}) - \nabla f(\boldsymbol{x})\| > \epsilon) \leq e^{-\frac{\epsilon^2}{2\sigma^2}}.$$

In the current problem, Theorem 1 shows that a valid bound on the standard deviation is $\sqrt{\frac{Bk\log(k)}{n_0}}$. Therefore, as $d = mk$, a stochastic gradient descent algorithm with noise of $O(\frac{1}{\sqrt{n_0 m}})$ would converge to a local minimum of $c(\boldsymbol{S})$. However, Algorithm 2 is a *projected* stochastic gradient descent algorithm (with such added perturbation), so we cannot immediately apply Theorem 2. Thus, we next connect Algorithm 2 with the standard stochastic gradient algorithm. Note that for the case $1/\gamma = 0$, the stochastic gradient is always orthogonal to $\boldsymbol{S}$:

Lemma 3. *Let $1/\gamma = 0$. Then for all $\boldsymbol{S}$ and any random sample $[n_t], [m_t]$ of rows and columns, we have*

$$\boldsymbol{S}^T \nabla c_t(\boldsymbol{S}) = 0.$$

The statement follows immediately from Lemma 4 in Appendix A.1. Therefore, the projection step (Step 21 in Algorithm 2) is trivial and reduces to:

$$\overline{\nabla \boldsymbol{S}_{t+1}} = -\nabla \tilde{\boldsymbol{S}}_{t+1}$$

In other words, the projected gradient is exactly the original gradient, and the projection step does not change the derivative. Thus, Algorithm 2 under $1/\gamma = 0$ reduces to a stochastic gradient algorithm.

Thus, if we can establish that $c(\boldsymbol{S})$ is a (θ, ζ, η)-strict saddle, we can indeed use Theorem 2 to prove that Algorithm 2 converges to a local minimum. Therefore, to establish the global convergence of Algorithm 2, we now equivalently need to establish the following two results:

1. The function $c(\boldsymbol{S})$ is a (θ, ζ, η)-strict saddle.

2. All local minima of $c(\boldsymbol{S})$ are global minima.

These properties and Theorem 2 will then establish that Algorithm 2 converges to a point δ-close to the global minimum in $O\left(\text{poly}\left(\frac{1}{\delta}\right)\right)$ iterations. We prove these two results simultaneously. To do so, we introduce a few regularity conditions that are necessary for these statements to be true. First, the matrix $\boldsymbol{A}$ we want to recover cannot be the following:

$$\boldsymbol{A} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix}. \tag{2.13}$$

This is because to recover such matrix truthfully, we need to have to know A_{11} as knowing any other element would give us no information about the 1 in the top left corner. To know A_{11} we would require basically all entries to be known under a random sampling model, which is undesirable; see [33] for further discussion. To prevent this from happening, we introduce the following concept:

Definition 2. *We define a matrix $\boldsymbol{S} \in \mathbb{R}^{m \times k}$, for $m > k$, to be (α, p)-submatrix full rank if every $p \times k$ submatrix of $\boldsymbol{S}$ is full column rank and the minimum singular value is at least α.*

We introduce the following assumption:

Assumption 1. *The desired matrix to be recovered, $\boldsymbol{A}$, is of rank k and admits a decomposition $\boldsymbol{A} = \boldsymbol{U}^* \boldsymbol{S}^{*T}$ where $\boldsymbol{U}^* \in \mathbb{R}^{n \times k}, \boldsymbol{S}^* \in \mathbb{R}^{m \times k}$, $\|\boldsymbol{S}^*\| = 1$, and $\boldsymbol{U}^*$, $\boldsymbol{S}^*$ is (α, l)-submatrix full rank, where l/m is the sampling rate and $\alpha > 0$.*

Qualitatively, this assumption requires the decomposition of $\boldsymbol{A}$ to be "spread out" enough so that no particular row and column is special. In particular, it excludes the pathological matrices like the one presented in Eq. (2.13), as $\boldsymbol{A} = (1, 0, \cdots, 0)(1, 0, \cdots, 0)^T = \boldsymbol{U}^* \boldsymbol{S}^{*T}$, and most of the submatrices of $\boldsymbol{U}^* = \boldsymbol{S}^* = (1, 0, \cdots, 0)$ have rank 0, save for those that includes the first element. This is because the first element is special in the decomposition of such $\boldsymbol{A}$, and if the first element is missing, no information for $\boldsymbol{A}$ can be deduced. Assumption 1 excludes these matrices so that no element is special enough such that if it is missing, we fail to recover $\boldsymbol{A}$.

We now introduce a second technical assumption. If $\boldsymbol{W_i S}$ has rank $< k$, then the inverse $(\boldsymbol{S}^T \boldsymbol{W}_i \boldsymbol{S})^{-1}$ in the objective function $c(\boldsymbol{S})$ would not be well-defined, so the objective function is not defined. Therefore, the second assumption is that we restrict our domain to the set of $\boldsymbol{S}$ where the objective function is defined:

Assumption 2. *The set $\boldsymbol{S}$ is restricted to matrices that are (α, l)-submatrix full rank, where l/m is the sampling rate and $\alpha > 0$.*

In real-world situations this assumption is almost never violated as numerical matrices almost never have *perfectly* singular submatrices. If it is, any small random perturbation around $\boldsymbol{S}$ would almost surely make the condition true.

Now, with these regularity conditions, we establish that for r sufficiently large, the two results we aim to prove are true:

Theorem 3. *Assume $\boldsymbol{A}$ follows the row-sampling model, and each element is known with probability $r = \frac{l}{m} > \frac{Ck}{m}$ for sufficiently large C. Further assume Assumptions 1-2 hold. Then, with probability $1 - O(\frac{1}{n})$, for $1/\gamma = 0$ and $\boldsymbol{B} = \boldsymbol{I}_m$, $c(\boldsymbol{S})$ is $(\epsilon, O(\frac{\epsilon}{k}), O(\sqrt{\epsilon}))$-strict saddle for sufficiently small ϵ. Furthermore, all local minima of $c(\boldsymbol{S})$ are global minima.*

The proof is contained in Appendix A.1. Then combining Theorem 3 with Theorem 2, we have the desired global convergence result:

Corollary 2. *Consider the minimization problem defined in (2.2) under* $1/\gamma = 0$ *and* $\boldsymbol{B} = \boldsymbol{I}_m$*. Assume that Assumptions 1-2 hold. Then, under the row-sampling model, for* $r > \frac{Ck}{m}$ *with* C *sufficiently large, Algorithm 2 (with added perturbation) outputs a point that is* δ*-close to a global minimum in* $\text{poly}(\frac{1}{\delta})$ *iterations with probability* $1 - O(\frac{1}{n})$*.*

As explained above, we can transfer results in the row-sampling model to the commonly utilized uniform sampling (Bernoulli) model:

Corollary 3. *Consider the minimization problem defined in (2.2) under* $1/\gamma = 0$ *and* $\boldsymbol{B} = \boldsymbol{I}_m$*. Assume that Assumptions 1-2 hold. Then, under the uniform sampling model, for a sampling rate* $r_0 > \frac{Cnk\log(n)}{mn}$ *with* C *sufficiently large, Algorithm 2 (with added perturbation) outputs a point that is* δ*-close to a global minimum in* $\text{poly}(\frac{1}{\delta})$ *iterations with probability* $1 - O(\frac{1}{n})$*.*

The conclusion is consistent with other algorithms which require $O(nk^{\alpha}\log(n))$ samples under the uniform sampling model.

Necessity of Regularization

The previous results deal with the case when there is no regularization ($1/\gamma = 0$) though it can be readily extended to any sufficiently small regularization term. Therefore, it is natural to question the purpose of the ℓ_2 regularization term in the formulation. The answer is two-fold:

1. The proof above is for the case when the matrix $\boldsymbol{A}$ is perfectly low rank. However, in real-world cases, the data might be corrupted in unseen ways, and it is shown in [19] that ℓ_2 regularization provides robustness properties that could guard against such corruption.

2. There are cases in the real-world in which the the number of known entries in a row $\boldsymbol{A}_i$ of the partially-known matrix is less than the desired rank k, and in such cases the

regularization becomes necessary to recover the true solution, as $\text{Rank}(\boldsymbol{W}_i) < k$, so $\boldsymbol{S}\boldsymbol{W}_i\boldsymbol{S}^T$ is not invertible. Adding a regularization term guards against such cases.

2.4.2 Computational Complexity of fastImpute

Here we first discuss how m_0 and n_0 is chosen. By Theorem 1 on the approximation of the stochastic gradient and objective, the error in such approximation drops exponentially with increasing n_0. Therefore, we want to ensure n_0 is never too small.

Now, by our convergence result in Theorem 3, we need $O(nk \log n)$ samples to recover the true solution under the uniform sampling model, when n is sufficiently large. Since α is the empirical probability that any element is known, we need to have:

$$m_0 n_0 \alpha = O(nk \log n).$$

The complexity required to discern the p factors we can choose from is $O(p)$, thus, our selection of n_0 and m_0 is:

$$m_0 = O(p), \qquad n_0 = \max\left\{O\left(\frac{nk\log(n)}{m_0\alpha}\right), \underline{n}\right\}.$$

where $\underline{n}$ is some lower bound on n_0 to ensure that Theorem 3 can be applied. We now show that such m_0 and n_0 reduces the computational complexity of the gradient update step asymptotically:

Corollary 4. *The computational complexity of Step 10 in Algorithm 2 is*

$$O\left(\frac{(pk + k^3)n\log(n)}{\alpha}\right).$$

Proof. For the gradient update step, we replace m with $O(p)$ and n with $O\left(\frac{kn\log(n)}{p\alpha}\right)$ in the original formula for Step 10 in projImpute to get that its computational complexity is:

$$O\left(\frac{kn\log(n)}{\alpha}\left(p + k^2 + \frac{k}{\alpha} + \frac{k^3 + pk}{p\alpha}\right)\right)$$

$$= O\left(\frac{(pk + k^3)n\log(n)}{\alpha}\right).$$

Where we have suppressed the last two terms as they are dominated by the first two. □

We note that the dependence of m is completely removed (for sufficiently large n) when compared with Algorithm 1. This, however, does not reduce the asymptotic complexity with regards to m for the full algorithm as eventually the step to fill the matrix $\boldsymbol{A}$ dominates, and that scales linearly with m. Our dependence on k is cubic, but k is usually small in real-world applications, so it is not a large concern.

In the next section, we discuss experiments for fastImpute.

2.5 Experiments on fastImpute with Side Information

In this section, we compare fastImpute with other inductive matrix completion algorithms on both synthetic datasets and real-world datasets to explore its performance and scaling behavior.

2.5.1 Synthetic Data Experiments

For synthetic data experiments, we assume that the underlying matrix satisfies the form $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{S}^T\boldsymbol{B}^T$, where $\boldsymbol{U} \in \mathbb{R}^{n\times k}$, $\boldsymbol{S} \in \mathbb{R}^{p\times k}$, and $\boldsymbol{B} \in \mathbb{R}^{m\times p}$. The elements of $\boldsymbol{U}$, $\boldsymbol{S}$, and $\boldsymbol{B}$ are selected from a uniform distribution of $[0, 1]$, where a fraction μ is missing. We report statistics on various combinations of (m, n, p, k, μ).

All algorithms are tested on a server with 16 CPUs. For each combination (m, n, p, k, μ), we ran 10 tests and report the average value for every statistic. The algorithms tested are:

- **fastImpute**: We use the sampling parameters:

$$m_0 = \min(2p, m), \qquad n_0 = \max\left\{\frac{nk\log(n)}{8m_0\alpha}, 100\right\}.$$

  with $t_{max} = 50$, and $\theta = \frac{\pi}{64}$, and regularization parameter $\gamma = 10^6$. We explicitly stress here that no parameter tuning is done on fastImpute as we intend to show that the

algorithm is not parameter sensitive. We implement our algorithm in Julia 0.6 with only the base packages.

- **IMC**: This algorithm is a well-accepted benchmark for testing Inductive Matrix Completion algorithms developed by [131]. For each combination of (m, n, p, k, μ), we tune the regularization parameter λ by imputing random matrices with such combination and find the λ that gives the best results. We utilize the implementation provided by the authors in Matlab.

To further understand the benefits of the stochastic algorithm, we also compare against projImpute, the full gradient descent algorithm. We report the following statistics for each algorithm:

- n, m - the dimensions of $\boldsymbol{A}$.
- p - the number of features in the feature matrix.
- k - the true number of features.
- μ - The fraction of missing entries in $\boldsymbol{A}$.
- T - the total time of algorithm execution.
- MAPE - the Mean Absolute Percentage Error (MAPE) for the retrieved matrix $\hat{\boldsymbol{A}}$:

$$\text{MAPE} = \frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} \frac{|\hat{A}_{ij} - A_{ij}|}{|A_{ij}|}.$$

We have the following observations:

- fastImpute significantly improves the scaling behavior of projImpute in n and m, and reduces the running time by over 10 times on real-world datasets.
- For small n, m, fastImpute scales roughly linearly in n and is not sensitive towards m. Eventually it scales as $O(nm)$ when the matrix filling step dominates, as predicted. IMC scales similarly with slightly worse accuracy.

	n	m	p	k	μ%	projImpute		fastImpute		IMC	
						T	MAPE	T	MAPE	T	MAPE
n	10^3	10^3	100	5	95%	5.1s	0.2%	3.1s	0.4%	4.4s	2.9%
	10^4	10^3	100	5	95%	56.7s	0.1%	12.0s	0.1%	18s	2.5%
	10^5	10^3	100	5	95%	580s	0.2%	54.8s	0.2%	173s	1.0%
	10^6	10^3	100	5	95%	6035s	0.2%	390s	0.2%	1400s	0.3%
m	10^4	10^3	100	5	95%	56.7s	0.1%	12.0s	0.1%	18s	2.5%
	10^4	10^4	100	5	95%	605s	0.2%	39s	0.4%	144s	0.8%
	10^4	10^5	100	5	95%	5874s	0.1%	360s	0.2%	1265s	0.4%
	10^4	10^6	100	5	95%	N/A	0.1%	3056s	0.1%	13891s	0.2%
p	10^4	10^3	100	5	95%	56.7s	0.1%	12.0s	0.1%	18s	2.5%
	10^4	10^3	200	5	95%	104s	0.05%	23s	0.1%	30s	2.7%
	10^4	10^3	500	5	95%	230s	0.06%	40s	0.05%	87s	2.1%
	10^4	10^3	1000	5	95%	417s	0.02%	85s	0.01%	160s	1.9%
k	10^4	10^3	100	5	95%	56.7s	0.1%	12.0s	0.1%	18s	2.5%
	10^4	10^3	100	10	95%	194s	0.1%	37.6s	0.2%	32s	2.1%
	10^4	10^3	100	20	95%	460s	0.2%	90s	0.4%	47s	2.2%
	10^4	10^3	100	30	95%	1055s	0.3%	290s	0.5%	55s	2.6%
μ	10^4	10^3	100	5	20%	107.4s	0.5%	25.1s	0.7%	6.2s	0.01%
	10^4	10^3	100	5	50%	92.6s	0.2%	19.2s	0.4%	8.9s	0.03%
	10^4	10^3	100	5	80%	70.4s	0.1%	15.7s	0.1%	14s	0.7%
	10^4	10^3	100	5	95%	56.7s	0.1%	12.0s	0.1%	18s	2.5%
	10^4	10^3	100	5	95%	56.7s	0.1%	12.0s	0.1%	18s	2.5%
	10^5	10^3	100	10	95%	1095s	0.1%	84s	0.1%	203s	0.8%
	10^5	10^4	200	10	95%	14506s	0.1%	560s	0.2%	2430s	0.4%
	10^5	10^5	200	10	95%	N/A	N/A	3170s	0.2%	25707s	0.2%
	10^6	10^4	500	20	95%	N/A	N/A	6516s	0.2%	N/A	N/A

Table 2.1: Comparison of fastImpute and IMC on synthetic data. N/A means the algorithm did not complete running in 20 hours, corresponding to 72000 seconds.

- Both algorithms roughly exhibit linear scaling in p, but IMC retrieves a matrix that has much higher MAPE than fastImpute.

- fastImpute roughly shows the $O(k^3)$ dominant behavior as expected, while IMC seems to scale linearly in k.

- IMC's running time increases with more missing data while it decreases for fastImpute. Both algorithms achieve roughly the same performance, with IMC dropping

significantly as the number of missing entries increased.

2.5.2 Real-World Experiments

For real-world experiments, we utilize the Netflix Prize Dataset. This dataset was released in a competition to predict ratings of customers on unseen movies, given over 10 million ratings scattered across $500,000$ people and $16,000$ movies. Thus, when presented in a matrix $\boldsymbol{A}$ where A_{ij} represents the rating of individual i on movie j, the goal is to complete the matrix $\boldsymbol{A}$ under a low-rank assumption.

For this experiment, we included movies where people who had at least 5 ratings present. This gives a matrix of $471,268$ people and $14,538$ movies. To observe the scalability of fastImpute, we created five data sets (in similar format to [22]):

1. Base - $\boldsymbol{A}_1$ has dimensions $3,923 \times 103$.
2. Small - $\boldsymbol{A}_2$ has dimensions $18,227 \times 323$.
3. Medium - $\boldsymbol{A}_3$ has dimensions $96,601 \times 788$.
4. Large - $\boldsymbol{A}_4$ has dimensions $471,268 \times 1760$.
5. Full - $\boldsymbol{A}$ has dimensions $471,268 \times 14,538$.

These sizes are constructed such that the total number of elements in $\boldsymbol{A}$ in the successive sizes are approximately different by approximately an order of magnitude.

The feature matrix $\boldsymbol{B}$ is constructed using data from the TMDB Database, and covers 59 features that measure geography, popularity, top actors/actresses, box office, runtime, genre and more. The full list of 59 features is contained in Appendix A.3.

For comparison, we test against IMC. We split the training set in 80%/20%, where the latter group is used for validation of the rank in IMC and fastImpute. We then report the time taken, T, the MAPE, and the optimal chosen rank k^* for each algorithm: We see that fastImpute is able to outperform IMC on the Netflix dataset across the different n and m values, while enjoying competitive scalability.

n	m	p	$\mu\%$	fastImpute			IMC		
				T	k^*	MAPE	T	k^*	MAPE
3,923	103	59	92.6%	1.7s	5	32.9%	0.8s	5	34.1%
18,227	323	59	94.8%	11	6	28.0%	7.5s	6	29.0%
96,601	788	59	94.2%	75s	7	25.7%	49s	8	28.5%
471,268	1,760	59	93.6%	460s	8	22.9%	870s	10	24.1%
471,268	14,538	59	94.1%	2934s	8	20.7%	7605s	10	21.0%

Table 2.2: Comparison of methods on Netflix data for fastImpute.

2.6 fastImpute without Side Information

In this section, we explore fastImpute in the special case where there is no side information ($p = m$, and $\boldsymbol{B} = \boldsymbol{I}_m$). In such case, the objective function becomes

$$c(\boldsymbol{S}) = \frac{1}{nm} \sum_{i=1}^{n} \overline{\boldsymbol{a}}_i \left(\boldsymbol{I}_m - \boldsymbol{S} \left(\boldsymbol{S}^T \boldsymbol{W}_i \boldsymbol{S} \right)^{-1} \boldsymbol{S}^T \right) \overline{\boldsymbol{a}}_i^T.$$

This special cases give rises to two additional optimizations:

- Define $d = \min\{m, n\}$. Since the problem is now symmetric in n and m, instead of requiring $O(nk \log(n))$ samples, we only require $O(dk \log(d))$ as we can similarly perform the matrix completion on $\boldsymbol{A}^T$.
- The multiplications involving $\boldsymbol{B}$ are no longer needed as $\boldsymbol{B}$ is the identity matrix and thus does not contribute.

With these two observations, we can update our corollary about the computational complexity of the gradient update step:

Corollary 5. *For $m = p$, and $\boldsymbol{B} = \boldsymbol{I}_m$, the computational complexity of Step 10 in Algorithm 2 is*

$$O\left(\frac{k^3 d \log(d)}{\alpha} \right).$$

Using this further optimization, we compare fastImpute with multiple general matrix completion algorithms on both synthetic datasets and real-world datasets to explore its

performance and scaling behavior.

2.6.1 Synthetic Data Experiments

For synthetic data experiments, we assume that the underlying matrix satisfies the form $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{S}^T$, where $\boldsymbol{U} \in \mathbb{R}^{n\times k}$, $\boldsymbol{S} \in \mathbb{R}^{m\times k}$. Then the elements of $\boldsymbol{U}$ and $\boldsymbol{S}$ are selected from a uniform distribution of $[0, 1]$, where a fraction μ is missing. We report statistics on various combinations of (m, n, k, μ).

The algorithms tested are:

- **fastImpute**: We use the sampling parameters:

$$m_0 = m, \qquad n_0 = \max\left\{\frac{nk\log(n)}{4m_0\alpha}, 100\right\}.$$

  with $t_{max} = 50$, and $\theta = \frac{\pi}{64}$, and regularization parameter $\gamma = 10^6$. We explicitly stress here that no parameter tuning is done on fastImpute as we intend to show that the algorithm is not parameter sensitive. We implement our algorithm in Julia 0.6 with only the base packages.

- **softImpute-ALS (SIALS)**: Developed by [86], this is widely recognized as a state-of-the-art matrix completion method without feature information. It has among the best scaling behavior across all classes of matrix completion algorithms as it utilizes fast alternating least squares to achieve scalability. For each combination of (m, n, μ, k), we tune the regularization parameter λ by imputing random matrices with such combination and find the λ that gives the best results. We utilize the implementation in the softImpute package in R for testing.

- **softImpute-SVD (SISVD)**: Developed by [126], this is the original softImpute algorithm that utilizes truncated SVDs and spectral regularization to impute the matrix. This method is used as a fast benchmark for SVD-type methods. For each combination of (m, n, μ, k), we tune the regularization parameter λ by imputing random matrices with such combination and find the λ that gives the best results. We utilize the implementation in the softImpute package in R for testing.

- **Matrix Factorization Stochastic Gradient Descent (MFSGD)**: This is a popular stochastic gradient descent algorithm (discussed in [96]) which separates $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{V}$, where $\boldsymbol{U} \in \mathbb{R}^{n\times k}$ and $\boldsymbol{V} \in \mathbb{R}^{k\times m}$), and perform gradient updates for $\boldsymbol{U}$ and $\boldsymbol{V}$. We utilize the implementation in the Fancyimpute package of python that utilizes Tensorflow and the latest available speed optimizations for such algorithm.

To further illustrate the favorable scaling behavior of fastImpute, we compare fastImpute against *online* matrix completion algorithms. Online matrix completion algorithms complete the matrix by updating the factorization $\boldsymbol{U}$ and $\boldsymbol{V}$ with sequential data input from $\boldsymbol{A}$ over time. By their sequential nature, they are usually much faster than offline matrix completion algorithms which consider all the data jointly. The specific online algorithm we compare to is:

- **GROUSE**: This is a popular and efficient online matrix completion algorithm based on sequential gradient updates on the Grassmann manifold, as set out in [9]. We utilize the official implementation in MATLAB along with the latest optimizations in MATLAB R2020a.

We note that online matrix completion is usually used in different settings than offline algorithms and such comparison is only to note the favorable scaling behavior of fastImpute.

All algorithms are executed on a server with 16 CPUs. Each combination (m, n, k, μ) was ran 10 times, and we report the average value of every statistic. The statistics reported are as followed:

- n, m - the dimensions of $\boldsymbol{A}$.
- k - the true number of features.
- μ - The fraction of missing entries in $\boldsymbol{A}$.
- T - the total time of algorithm execution.
- MAPE - the Mean Absolute Percentage Error (MAPE) for the retrieved matrix $\hat{\boldsymbol{A}}$:

$$\text{MAPE} = \frac{1}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m}\frac{|\hat{A}_{ij} - A_{ij}|}{|A_{ij}|}.$$

	n	m	k	μ%	fastImpute		SIALS		SISVD		MFSGD		GROUSE	
					T	MAPE	T	MAPE	T	MAPE	T	MAPE	T	MAPE
n	10^3	10^3	5	95%	0.9s	3.5%	2.3s	19.9%	213s	21.3%	9.4s	2.1%	1.2s	5.0%
	10^4	10^3	5	95%	4.2s	2.4%	25.5s	12.7%	1780s	17.5%	72.1s	2.8%	9.6s	2.6%
	10^5	10^3	5	95%	18.9s	2.2%	443s	8.1%	23650s	12.1%	709s	6.1%	153s	3.1%
	10^6	10^3	5	95%	104s	2.1%	6270s	6.7%	N/A	N/A	7605s	7.0%	1480s	2.7%
m	10^4	10^3	5	95%	4.2s	2.4%	25.5s	12.7%	1780s	17.5%	72.1s	2.8%	9.6s	2.6%
	10^4	10^4	5	95%	19s	4.0%	227s	6.2%	15070s	8.9%	840s	5.4%	75s	5.1%
	10^4	10^5	5	95%	140s	3.1%	3170s	9.1%	N/A	N/A	8010s	7.5%	748s	3.6%
	10^4	10^6	5	95%	1052s	3.5%	30542s	8.0%	N/A	N/A	N/A	N/A	6309s	3.1%
k	10^4	10^3	5	95%	4.2s	2.4%	25.5s	12.7%	1780s	17.5%	72.1s	2.8%	9.6s	2.6%
	10^4	10^3	10	95%	14.1s	2.8%	41.0s	9.8%	3120s	11.7%	80.7s	2.6%	17.6s	2.5%
	10^4	10^3	20	95%	29.8s	3.7%	80.4s	8.2%	3609s	14.6%	83.7s	5.9%	80.2s	1.9%
	10^4	10^3	30	95%	49.5s	5.0%	122s	9.3%	3670s	20.9%	82.1s	4.9%	196s	1.1%
μ	10^4	10^3	5	20%	8.4s	1.1%	3.6s	0.6%	204s	0.6%	140s	4.9%	21s	1.9%
	10^4	10^3	5	50%	7.5s	1.6%	6.7s	1.0%	370s	1.1%	109s	1.2%	15.8s	2.1%
	10^4	10^3	5	80%	5.7s	2.4%	13.1s	4.0%	1340s	2.7%	85s	1.9%	12.5s	2.4%
	10^4	10^3	5	95%	4.2s	2.6%	25.5s	12.7%	1780s	17.5%	72.1s	2.8%	9.6s	2.6%
	10^4	10^3	5	95%	4.2s	2.5%	25.5s	12.7%	1780s	17.5%	72.1s	2.8%	9.6s	2.6%
	10^5	10^3	10	95%	29.0s	2.5%	403s	10.7%	25049s	14.6%	708s	6.0%	270s	2.9%
	10^5	10^4	10	95%	317s	2.1%	4470s	8.9%	N/A	N/A	8215s	7.3%	2076s	2.4%
	10^5	10^5	10	95%	3260s	2.0%	52690s	4.1%	N/A	N/A	N/A	N/A	27043s	2.2%
	10^6	10^4	20	95%	5070s	1.9%	N/A	N/A	N/A	N/A	N/A	N/A	48740s	2.9%

Table 2.3: Comparison of fastImpute, SIALS, SISVD, MFSGD, and GROUSE on synthetic data. N/A means the algorithm did not complete running in 20 hours, corresponding to 72000 seconds.

The results are separated into sections in Table 2.3. The first four sections investigate fastImpute's scalability with respect to each of the 4 parameters m, n, μ, k, with the parameter under investigation denoted in the leftmost column.The final section of the results compares the different algorithms' performance on large realistic combinations of (m, n, p, k).

We see that on the final set of large realistic combinations, fastImpute outperforms all comparison algorithms in all cases. Table 2.4 records the average difference in time and MAPE between fastImpute and the other algorithms, on the final set of combinations. On average fastImpute takes 10% of the time of comparison while achieving $\sim 40-70\%$ reduction in MAPE at the same time.

Even when compared to the online algorithm GROUSE, fastImpute is on average over 80% faster on the final set of combinations.

For scaling behavior, we have the following observations:

	SIALS		SISVD		MFSGD		GROUSE	
	ΔT	ΔMAPE	ΔT	ΔMAPE	ΔT	ΔMAPE	ΔT	ΔMAPE
fastImpute vs.	-90%	-71%	-99%	-95%	-96%	-45%	-81%	-15%

Table 2.4: Average performance of fastImpute, SIALS, SISVD, MFSGD and GROUSE on synthetic trials. Percentages are computed by averaging over the set of realistic combinations.

- n, m - We see that fastImpute scales sublinearly for low n, m as the gradient descent step dominates, and as we move to $n \sim 10^6$ it starts to scale as $O(nm)$ as the step of completing the final matrix starts to dominate. The MAPE steadily decreases as we have more entries. In contrast, SIALS, SISVD, and MFSGD all roughly scale linearly with n, m from the start. Interestingly, the MFSGD algorithm has increasing error with increasing number of entries - we hypothesize this may be due to the gradient descent in factorized form $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{V}^T$ failing to capture non-linear dynamics of the interactions between $\boldsymbol{U}$ and $\boldsymbol{V}$ at high levels.

- k - Somewhat surprisingly, fastImpute scales as $O(k)$ even though theoretically it scales at $O(k^3)$. We believe this is due to the small constant factor in front of the k^2 and k^3 terms.

- μ - We see that in accordance to the linear dependence on $|\Omega|$, the number of known entries, fastImpute runs slower as we have more filled elements, while in contrast SIALS and SISVD both run faster. MFSGD is similar to fastImpute in that it runs slower with more filled elements (a construct of the gradient descent method).

2.6.2 Real-World Experiments

For real-world experiments, we again utilize the Netflix datasets created in Section 2.5.2 without the feature matrix.

We test against SIALS as it is the only algorithm capable of scaling to such size. We split the training set in 80%/20%, where the latter group is used for validation of the rank in SIALS and fastImpute. We then report the time taken, T, the MAPE, and the optimal chosen rank k^* for each algorithm: We see that fastImpute is able to outperform SIALS on the Netflix dataset across the different n and m values, while enjoying superior scalability

n	m	μ%	fastImpute			SIALS		
			T	k^*	MAPE	T	k^*	MAPE
3,923	103	92.6%	2s	5	23.5%	4s	5	30.6%
18,227	323	94.8%	18s	8	19.8%	47s	7	27.5%
96,601	788	94.2%	109s	10	18.2%	620s	10	24.0%
471,268	1,760	93.6%	370s	10	16.5%	2837s	12	22.5%
471,268	14,538	94.1%	2098s	12	13.8%	38256s	14	20.1%

Table 2.5: Comparison of methods on Netflix data for fastImpute

especially as we approach the full matrix.

2.7 Conclusion

In conclusion, we have designed a unified optimization framework that is able to conduct state-of-the-art matrix completion with and without side information. Using the factorization approach $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{S}^T\boldsymbol{B}^T$, we wrote $\boldsymbol{U} = f(\boldsymbol{S})$ as a function of $\boldsymbol{S}$, and derived the cost and gradient expressions with respect to $\boldsymbol{S}$ through a separable reformulation of the problem. By then conducting non-convex gradient descent on $\boldsymbol{S}$, our synthetic and real-world data experiments show the competitiveness of the method in both scalability and accuracy against a multitude of comparison algorithms.

Chapter 3

Interpretable Matrix Completion: A Discrete Optimization Approach

3.1 Introduction

The classical low-rank matrix completion problem considers the following problem: Given a matrix $\boldsymbol{A} \in \mathbb{R}^{n\times m}$ with entries only partially known (denote $\Omega \subset \{1,\cdots,n\}\times\{1,\cdots,m\}$ as the set of known entries), we aim to recover a matrix $\boldsymbol{X} \in \mathbb{R}^{n\times m}$ of rank k that minimizes a certain distance metric between $\boldsymbol{X}$ and $\boldsymbol{A}$ on the known entries of $\boldsymbol{A}$:

$$\min_{\boldsymbol{X}} \frac{1}{nm} \sum_{(i,j)\in\Omega} \|X_{ij} - A_{ij}\|^2 \quad \text{subject to} \quad \text{Rank}(\boldsymbol{X}) = k,$$

where we normalized the objective so that it is $O(1)$. The rank k constraint on $\boldsymbol{X}$ can be equivalently formulated as the existence of two matrices $\boldsymbol{U} \in \mathbb{R}^{n\times k}$, $\boldsymbol{V} \in \mathbb{R}^{m\times k}$ such that $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{V}^T$. Therefore, the problem can be restated as:

$$\min_{\boldsymbol{U}} \min_{\boldsymbol{V}} \frac{1}{nm} \sum_{(i,j)\in\Omega} \|X_{ij} - A_{ij}\|^2 \quad \text{subject to} \quad \boldsymbol{X} = \boldsymbol{U}\boldsymbol{V}^T. \tag{3.1}$$

Low-rank matrix completion has attracted much attention after the successful application in the Netflix Competition. It is now widely utilized in far-reaching areas such as computer vision ([32]), signal processing ([95]), and control theory ([27]) to generate a completed matrix

from partially observed entries.

In many applications for matrix completion, it is customary for each row of the data to represent an individual and each column a product or an item of interest, and A_{ij} being the response data of individual i on item j. Therefore, the matrices $\boldsymbol{U}$ and $\boldsymbol{V}$ are commonly interpreted as the "user matrix" and "product matrix", respectively.

Let us denote each row of $\boldsymbol{U}$ as $\boldsymbol{u}_i$ and each column as $\boldsymbol{u}^i$ (similarly for $\boldsymbol{V}$). Then, $\boldsymbol{u}^i$ ($\boldsymbol{v}^i$) represents a "latent feature" for users (products), and in total there are k latent features for users (products). The goal of matrix completion is thus to discover such latent features of the users and the products, so that the dot product of such features on user i and product j, $X_{ij} = \boldsymbol{u}_i^T \cdot \boldsymbol{v}_j$, is the intended response of user i on product j.

While this interpretation is intuitive, it does not offer any insight on what the latent features of users and products mean. Inductive Matrix Completion, first considered in [52], aims to rectify such problem by asserting that each of the k latent features is a linear combination of $p > k$ known features. We focus on the *one-sided information* case, where only one of the two matrices are subject to such constraint. Specifically, the one sided information constraint assumes (without loss of generality) that the matrix $\boldsymbol{V}$ satisfies:

$$\boldsymbol{V} = \boldsymbol{B}\boldsymbol{S},$$

for some feature matrix $\boldsymbol{B} \in \mathbb{R}^{m \times p}$ containing p features for each column/product, and $\boldsymbol{S} \in \mathbb{R}^{p \times k}$. As an example, if the items are movies, then $\boldsymbol{b}^j$ represents feature j for a movie (actors, budget, running time, etc), and each product feature $\boldsymbol{v}^j$ needs to be linear combination of such features. The *two-sided information* case, on the other hand, further assumes that the matrix $\boldsymbol{U}$ satisfies:

$$\boldsymbol{U} = \boldsymbol{C}\boldsymbol{T},$$

for some feature matrix $\boldsymbol{C} \in \mathbb{R}^{n \times p'}$ containing p' features for each row/user, and $\boldsymbol{T} \in \mathbb{R}^{p' \times k}$.

The *one-sided information* case is more relevant as features related to users ($\boldsymbol{U}$) are often fragmented and increasingly constrained by data privacy regulations, while features about products ($\boldsymbol{V}$) are easy to obtain.

The inductive version of the one-sided information problem in (3.1) can be written as:

$$\min_{\boldsymbol{U}} \min_{\boldsymbol{S}} \frac{1}{nm} \sum_{(i,j)\in\Omega} \|X_{ij} - A_{ij}\|^2 \quad \text{subject to} \quad \boldsymbol{X} = \boldsymbol{U}\boldsymbol{S}^T\boldsymbol{B}^T. \tag{3.2}$$

Although the inductive version of the problem adds more interpretability to the product latent features, they are still far from fully interpretable. For example, if we take the items to be movies, and base features ($\boldsymbol{B}$) to be running time, budget, box office, and number of top 100 actors, then the generated linear combinations ($\boldsymbol{B}\boldsymbol{S}$) could be such as:

$$5.6 \times \text{running time} - 0.00067 \times \text{budget} + 12 \times \#\text{ of Top 100 actors}$$
$$0.25 \times \text{box office} - 5 \times \#\text{ of Top 100 actors}$$

These linear combinations present two difficulties for interpretability. First, there is no restriction in the formulation that limits the total number of features used in the linear combinations. As we show in Section 3.5, on real-world data, it is common for this formulation to produce linear combinations with high number of features. When $p \gg k$, the resulting linear combinations could contain $O(p)$ features that is too large for decision makers to understand what is important. Additionally, the signs ofin this example, the number of Top 100 actors appears twice with different signs.

Furthermore, these generated linear combinations are not very interpretable themselves due to the involvement of multiple factors with different units. In particular, it is difficult to understand what it means for the feature $0.25 \times$ box office $- 5 \times \#$ of Top 100 actors to be important. The features can also be used with different signs across different linear combinations, as in this example with "# of Top 100 actors", making it challenging to understand how the feature affects the final outcome.

Therefore, we argue that instead of supposing the k features are linear combinations of the p known features, we should assume that the k features are *selected* from the p known features. This formulation alleviates the two previous problems mentioned: it guarantees the latent features to be interpretable (as long as the original features are), and it prevents any duplicating features in the selected k latent features. We denote this *Interpretable Matrix*

Completion. We use the term interpretable, as opposed to inductive, to highlight that our approach, like sparse linear regression, give actionable insights on what are the important features for the matrix $\boldsymbol{A}$. We note that being a discrete problem, Interpretable Matrix Completion faces additional difficulties compared to the Inductive or the classical matrix completion problem as it needs to select k out of p factors, which is generally a NP-hard problem.

In this paper, we show that the Interpretable Matrix Completion problem can be written as a mixed integer convex optimization problem. Inspired by [23] for sparse linear regression, we reformulate the interpretable matrix completion problem as a binary convex optimization problem. We then introduce a new algorithm OptComplete, based on stochastic cutting planes, to enable scalability for matrices of sizes on the order of $(n, m) = (10^6, 10^6)$. In addition, we provide empirical evidence on both synthetic and real-world data that OptComplete is able to match or exceed current state-of-the-art methods for inductive and general matrix completion on both speed and accuracy, while providing significantly better scaling behavior than state-of-the-art exact solvers.

The structure of the paper is as follows. In Section 3.2, we introduce the binary convex reformulation of the low-rank interpretable matrix completion problem, and how it can be solved through a cutting plane algorithm, which we denote CutPlanes. In Section 3.3, we introduce OptComplete, a stochastic cutting planes method designed to scale the CutPlanes algorithm in Section 3.2, and show that it recovers the optimal solution of CutPlanes with exponentially vanishing failure probability. In Section 3.4, we report on computational experiments with synthetic data that compare OptComplete to Inductive Matrix Completion (IMC) introduced in [131] and SoftImpute-ALS by [86], two state-of-the-art matrix completion algorithms for inductive and general completion. We also compare OptComplete to CutPlanes, and Pavito ([46]), a general convex MINLO solver, to demonstrate the 20x to 60x speedup of the stochastic algorithm. In Section 3.5, we report computational experiments on the Netflix Prize dataset that shows the favorable scaling behavior carries over to real-world datasets. In Section 3.6, we provide our conclusions.

Literature

Matrix completion has been applied successfully to many tasks, including recommender systems [105], social network analysis [41] and clustering [37]. After [34] proved a theoretical guarantee for the retrieval of the exact matrix under the nuclear norm convex relaxation, a lot of methods have focused on the nuclear norm problem (see [126], [13], [93], and [161] for examples). Alternative methods include alternating projections by [148] and Grassmann manifold optimization by [100]. There has also been work where the uniform distributional assumptions required by the theoretical guarantees are violated, such as [134] and [36].

Matrix completion with side information was first introduced in the context of inductive matrix completion. [182] showed that given predictive side information, one only needs $O(\log n)$ samples to retrieve the full matrix. Thus,many works (see [182], [92], [64], [131] for examples) have focused on the case in which the side information is assumed to be perfectly predictive so that the theoretical bound of $O(\log n)$ sample complexity [182] can be achieved. [40] explored the case in which the side information is corrupted with noise, while [152] and [154] incorporated nonlinear combination of factors into the side information.

Our work differs from prior work in that previous attempts mainly focus on choosing latent features which are linear combinations of the given features. *Interpretable Matrix Completion*, however, aims to *select* exactly k features from the known features. Surprisingly, as pointed out by a recent article [133], there is a considerable lack of effort to introduce sparsity/interpretability into inductive matrix completion, with [119], [155] and [133] being among the only works that attempt to do so by considering sparse linear combinations of the known features, and thus different from the setting of this paper.

3.2 Interpretable Matrix Completion

In this section, we present the mathematical formulation of Interpretable Matrix Completion and how it can be reformulated as a binary convex problem that is based on [23]. We show how this naturally leads to a cutting plane algorithm, and discuss its computational complexity. We also discuss the two-sided information case, and how that reduces to the

sparse regression problem.

3.2.1 Binary Convex Reformulation of Interpretable Matrix Completion

The (one-sided) interpretable matrix completion problem can be written as a mixed binary optimization problem:

$$\min_{\boldsymbol{U}} \min_{\boldsymbol{s}\in S_k^p} \frac{1}{nm} \sum_{(i,j)\in\Omega} \|X_{ij} - A_{ij}\|^2 \quad \text{subject to} \quad \boldsymbol{X} = \boldsymbol{USB}^T.$$

Where $\boldsymbol{U} \in \mathbb{R}^{n\times p}$, $S_p^k = \{\boldsymbol{s} \in \{0,1\}^p \mid \sum_{i=1}^p s_i = k\}$, and $\boldsymbol{S} = \text{Diag}\{s_1, \cdots, s_p\}$. We note that given that $\sum_{i=1}^p s_i = k$, the rank of matrix $\boldsymbol{X}$ is indeed k. We further note that the coefficients of $\boldsymbol{S}$ can be taken to be binary without loss of generality, since if they are not and $\boldsymbol{S} = \text{Diag}(s_1, \ldots, s_p)$, then by applying the transformation:

$$\boldsymbol{U} \to \boldsymbol{UD} \qquad \boldsymbol{S} \to \boldsymbol{SD}^{-1} \tag{3.3}$$

for $\boldsymbol{D} = \text{Diag}(1/s_1, \ldots, 1/s_p)$, (where we take $1/0 = 0$) results in an equivalent problem with the coefficients of $\boldsymbol{S}$ being binary.

For this paper, we consider the squared norm, and for robustness purposes (see [23] and [20]), we add a Tikhonov regularization term to the original problem. Specifically, the (one-sided) interpretable matrix completion problem with regularization we address is

$$\min_{\boldsymbol{U}} \min_{\boldsymbol{s}\in S_k^p} \frac{1}{nm} \left(\sum_{(i,j)\in\Omega} (X_{ij} - A_{ij})^2 + \frac{1}{\gamma}\|\boldsymbol{U}\|_2^2 \right) \quad \text{subject to} \quad \boldsymbol{X} = \boldsymbol{USB}^T. \tag{3.4}$$

In this section, we show how that problem (3.4) can be reformulated as a binary convex optimization problem, and can be solved to optimality using a cutting plane algorithm. The main theorem and proof is presented below:

Theorem 4. *Problem (3.4) can be reformulated as a binary convex optimization problem:*

$$\min_{\boldsymbol{s}\in S_k^p} c(\boldsymbol{s}) = \frac{1}{nm}\sum_{i=1}^{n}\overline{\boldsymbol{a}}_i\left(\boldsymbol{I}_m+\gamma\boldsymbol{W}_i\left(\sum_{j=1}^{p}s_j\boldsymbol{K}_j\right)\boldsymbol{W}_i\right)^{-1}\overline{\boldsymbol{a}}_i^T,$$

where $\boldsymbol{W}_1,\ldots,\boldsymbol{W}_n\in\mathbb{R}^{m\times m}$ *are diagonal matrices:*

$$(\boldsymbol{W}_i)_{jj}=\begin{cases}1, & (i,j)\in\Omega,\\ 0, & (i,j)\notin\Omega,\end{cases}$$

$\overline{\boldsymbol{a}}_i=\boldsymbol{a}_i\boldsymbol{W}_i$, $i\in[n]=\{1,\ldots,n\}$, *where* $\boldsymbol{a}_i\in\mathbb{R}^{1\times m}$ *is the* i*th row of* $\boldsymbol{A}$ *with unknown entries taken to be 0, and* $\boldsymbol{K}_j=\boldsymbol{b}^j(\boldsymbol{b}^j)^T\in\mathbb{R}^{m\times m}$, $j\in[p]$ *with* $\boldsymbol{b}^j\in\mathbb{R}^{m\times 1}$ *the* j*th column of* $\boldsymbol{B}$.

Proof. Proof: With the diagonal matrices $\boldsymbol{W}_i$ defined above, we can rewrite the sum in (3.4) over known entries of $\boldsymbol{A}$, $\sum_{(i,j)\in\Omega}(X_{ij}-A_{ij})^2$, as a sum over the rows of $\boldsymbol{A}$:

$$\sum_{i=1}^{n}\|(\boldsymbol{x}_i-\boldsymbol{a}_i)\boldsymbol{W}_i\|_2^2,$$

where $\boldsymbol{x}_i\in\mathbb{R}^{1\times m}$ is the ith row of $\boldsymbol{X}$. Using $\boldsymbol{X}=\boldsymbol{U}\boldsymbol{S}\boldsymbol{B}^T$, then $\boldsymbol{x}_i=\boldsymbol{u}_i\boldsymbol{S}\boldsymbol{B}^T$ where $\boldsymbol{u}_i\in\mathbb{R}^{1\times m}$ is the ith row of U. Moreover,

$$\|\boldsymbol{U}\|_2^2=\sum_{i=1}^{n}\|\boldsymbol{u}_i\|_2^2.$$

Then, Problem (3.4) becomes:

$$\min_{\boldsymbol{s}\in S_k^p}\ \min_{\boldsymbol{U}}\frac{1}{nm}\left(\sum_{i=1}^{n}\left(\|(\boldsymbol{u}_i\boldsymbol{S}\boldsymbol{B}^T-\boldsymbol{a}_i)\boldsymbol{W}_i\|_2^2+\frac{1}{\gamma}\|\boldsymbol{u}_i\|_2^2\right)\right).$$

We then notice that within the sum $\sum_{i=1}^{n}$ each row of $\boldsymbol{U}$ can be optimized separately, leading to:

$$\min_{\boldsymbol{s}\in S_k^p}\ \frac{1}{nm}\left(\sum_{i=1}^{n}\min_{\boldsymbol{u}_i}\left(\|(\boldsymbol{u}_i\boldsymbol{S}\boldsymbol{B}^T-\boldsymbol{a}_i)\boldsymbol{W}_i\|_2^2+\frac{1}{\gamma}\|\boldsymbol{u}_i\|_2^2\right)\right). \tag{3.5}$$

The inner optimization problem $\min_{\boldsymbol{u}_i}\|(\boldsymbol{u}_i\boldsymbol{S}\boldsymbol{B}^T-\boldsymbol{a}_i)\boldsymbol{W}_i\|_2^2+\frac{1}{\gamma}\|\boldsymbol{u}_i\|_2^2$ can be solved in closed

form given $\boldsymbol{S}$, as it is a weighted linear regression problem with Tikhonov regularization, see [23]. The closed form solution is:

$$\min_{\boldsymbol{u}_i} \|(\boldsymbol{u}_i \boldsymbol{S}\boldsymbol{B}^T - \boldsymbol{a}_i)\boldsymbol{W}_i\|_2^2 + \frac{1}{\gamma}\|\boldsymbol{u}_i\|_2^2 = \overline{\boldsymbol{a}}_i(\boldsymbol{I}_m + \gamma \boldsymbol{W}_i \boldsymbol{B}\boldsymbol{S}\boldsymbol{B}^T \boldsymbol{W}_i)^{-1}\overline{\boldsymbol{a}}_i^T. \tag{3.6}$$

So Problem (3.5) can be simplified to:

$$\min_{\boldsymbol{s}\in S_k^p} \frac{1}{nm}\left(\sum_{i=1}^{n} \overline{\boldsymbol{a}}_i(\boldsymbol{I}_m + \gamma \boldsymbol{W}_i \boldsymbol{B}\boldsymbol{S}\boldsymbol{B}^T \boldsymbol{W}_i)^{-1}\overline{\boldsymbol{a}}_i^T\right).$$

Finally, we notice that

$$\boldsymbol{B}\boldsymbol{S}\boldsymbol{B}^T = \sum_{j=1}^{p} s_j \boldsymbol{b}^j (\boldsymbol{b}^j)^T = \sum_{j=1}^{p} s_j \boldsymbol{K}_j,$$

and we obtain the required expression.

To see that this function is convex, one can calculate the Hessian of $c(\boldsymbol{s})$:

$$\begin{aligned}\frac{\partial c(\boldsymbol{s})}{\partial s_q \partial s_r} = 2\frac{\gamma^2}{nm}\sum_{i=1}^{n} \overline{\boldsymbol{a}}_i &\left(\boldsymbol{I}_m + \gamma \boldsymbol{W}_i \left(\sum_{j=1}^{p} s_j \boldsymbol{K}_j\right)\boldsymbol{W}_i\right)^{-1} \\ &\times \boldsymbol{W}_i \boldsymbol{K}_q \boldsymbol{W}_i \left(\boldsymbol{I}_m + \gamma \boldsymbol{W}_i \left(\sum_{j=1}^{p} s_j \boldsymbol{K}_j\right)\boldsymbol{W}_i\right)^{-1} \\ &\times \boldsymbol{W}_i \boldsymbol{K}_r \boldsymbol{W}_i \left(\boldsymbol{I}_m + \gamma \boldsymbol{W}_i \left(\sum_{j=1}^{p} s_j \boldsymbol{K}_j\right)\boldsymbol{W}_i\right)^{-1} \overline{\boldsymbol{a}}_i^T\end{aligned}$$

$\boldsymbol{K}_j$ are positive definite for all j, $\left(\boldsymbol{I}_m + \gamma \boldsymbol{W}_i \left(\sum_{j=1}^{p} s_j \boldsymbol{K}_j\right)\boldsymbol{W}_i\right)^{-1}$ is positive definite, and $\boldsymbol{W}_i$ are positive semidefinite by definition. Therefore, the Hessian of the cost function is a sum of quadratic forms of positive semi-definite matrices and thus is non-negative, so $c(\boldsymbol{s})$ is convex. □ □

With Theorem 4, our original problem can now be restated as:

$$\min_{\boldsymbol{s}\in S_k^p} c(\boldsymbol{s}) = \frac{1}{nm}\sum_{i=1}^{n} \overline{\boldsymbol{a}}_i \left(\boldsymbol{I}_m + \gamma \boldsymbol{W}_i \left(\sum_{j=1}^{p} s_j \boldsymbol{K}_j\right)\boldsymbol{W}_i\right)^{-1} \overline{\boldsymbol{a}}_i^T. \tag{3.7}$$

This can be solved utilizing the cutting plane algorithm first introduced by [58], summarized

as Algorithm 3. The cutting plane algorithm, at iteration t, adds a linear approximation of

Algorithm 3 Cutting-plane algorithm for matrix completion with side information.

procedure CUTPLANES($\boldsymbol{A}, \boldsymbol{B}$) ▷ masked matrix $\boldsymbol{A}$, and feature matrix $\boldsymbol{B}$
  $t \leftarrow 1$
  $\boldsymbol{s}_1 \leftarrow$ random initialization ▷ Random Initialization
  $\eta_1 \leftarrow 0$ ▷ Initialize feasible solution variable
  while $\eta_t < c(\boldsymbol{s}_t)$ **do** ▷ While the current solution is not feasible for iteration t
    $\boldsymbol{s}_{t+1}, \eta_{t+1} \leftarrow \underset{\boldsymbol{s} \in S_k^p, \eta > 0}{\arg\min} \quad \eta \quad \text{s.t.} \quad \eta \geq c(\boldsymbol{s}_i) + \nabla c(\boldsymbol{s}_i)^T (\boldsymbol{s} - \boldsymbol{s}_i) \quad \forall i \in [t]$
    $t \leftarrow t + 1$
  end while
  $\boldsymbol{s} \leftarrow \boldsymbol{s}_{t-1}$
  $i \leftarrow 1$
  for $i < n$ **do** ▷ Fill each row $\boldsymbol{x}_i$ of final output matrix $\boldsymbol{X}$
    $\boldsymbol{x}_i \leftarrow \boldsymbol{B}^{\boldsymbol{s}}((\boldsymbol{B}^{\boldsymbol{s}})^T \boldsymbol{W}_i \boldsymbol{B}^{\boldsymbol{s}})^{-1} (\boldsymbol{B}^{\boldsymbol{s}})^T \overline{\boldsymbol{a}}_i^T$ ▷ $\boldsymbol{B}^{\boldsymbol{s}}$ is submatrix of $\boldsymbol{B}$ with $\boldsymbol{s}$ columns
  end for
  return $\boldsymbol{X}$ ▷ Return the filled matrix $\boldsymbol{X}$
end procedure

$c(\boldsymbol{s})$ at the current feasible solution $\boldsymbol{s}_t$ to the set of constraints:

$$\eta \geq c(\boldsymbol{s}_t) + \nabla c(\boldsymbol{s}_t)^T (\boldsymbol{s} - \boldsymbol{s}_t), \tag{3.8}$$

and we solve the mixed-integer linear optimization problem (defining $[t] = \{1, \cdots, t\}$):

$$\begin{aligned} &\min_{\boldsymbol{s} \in S_k^p, \eta \geq 0} \eta \\ &\eta \geq c(\boldsymbol{s}_t) + \nabla c(\boldsymbol{s}_t)^T (\boldsymbol{s} - \boldsymbol{s}_t), \quad i \in [t] \end{aligned}$$

to obtain $\boldsymbol{s}_{t+1}, \eta_{t+1}$. We see that η_{t+1} is exactly the minimum value of the current approximation of $c(\boldsymbol{s})$, $c_t(\boldsymbol{s})$, defined below:

$$\eta_{t+1} = \min_{\boldsymbol{s}} \max_{i \in [t]} c(\boldsymbol{s}_t) + \nabla x(\boldsymbol{s}_t)^T (\boldsymbol{s} - \boldsymbol{s}_t) = \min_{\boldsymbol{s}} c_t(\boldsymbol{s}).$$

Since $c(\boldsymbol{s})$ is convex, the piecewise linear approximation $c_t(\boldsymbol{s})$ is an outer approximation ($c_t(\boldsymbol{s}) <= c(\boldsymbol{s})\ \forall \boldsymbol{s}$), so $\eta_t \leq c(\boldsymbol{s}_t)\ \forall t$. As the algorithm progresses, the set of linear approximations form an increasingly better approximation of $c(\boldsymbol{s})$, and η_t increases with t. The

algorithm terminates once the previous solution $(\boldsymbol{s}_t, \eta_t)$ is still feasible for the cutting plane problem at the tth iteration (which adds the tth cutting plane $\eta \geq c(\boldsymbol{s}_t) + \nabla c(\boldsymbol{s}_t)^T(\boldsymbol{s} - \boldsymbol{s}_t)$), as this means the new cutting plane does not cutoff the previous solution, implying optimality. Furthermore, we note that this implies the returned solution $(\boldsymbol{s}_T, \eta_T)$ is the optimal solution to the cutting plane problem with T cutting planes and the Tth cutting plane is at $\boldsymbol{s} = \boldsymbol{s}_T$.

Once the optimal solution $\boldsymbol{s}^*$ is reached, we can obtain the optimal $\boldsymbol{U}$ using the closed form solution in (3.6) and recover $\boldsymbol{X}$. In the next section, we discuss how this algorithm can be implemented in the context of $c(\boldsymbol{s})$ in (3.7). We also derive the computational complexity of the part of this algorithm that involves generating cutting planes. Although solving the resulting mixed-integer linear optimization problem (MILO) is a significant part of the cutting-plane algorithm, we do not consider further refinements for this part. We focus on improving the speed of calculating the cutting planes, and we observe in Section 3.4 that this is able to achieve a significant order-of-magnitude speedup. This suggests that the generation of cutting planes also accounts for a significant portion of computational difficulty.

3.2.2 Implementation and Computational Complexity of CutPlanes

The computational complexity of the cutting plane comes from calculating $c(\boldsymbol{s})$ and its derivative $\nabla c(\boldsymbol{s})$. We first introduce the notations $\alpha_i(\boldsymbol{s}) \in \mathbb{R}$ and $\boldsymbol{\tau}_i(\boldsymbol{s}) \in \mathbb{R}^{m \times 1}$.

$$\alpha_i(\boldsymbol{s}) = \frac{1}{m}\overline{\boldsymbol{a}}_i \boldsymbol{\tau}_i(\boldsymbol{s}) = \frac{1}{m}\overline{\boldsymbol{a}}_i \left[\left(\boldsymbol{I}_m + \gamma \boldsymbol{W}_i \left(\sum_{j=1}^{p} s_j \boldsymbol{K}_j \right) \boldsymbol{W}_i \right)^{-1} \overline{\boldsymbol{a}}_i^T \right], \quad i \in [n]. \tag{3.9}$$

Then, the function $c(\boldsymbol{s})$ in (3.7) can be expressed as

$$c(\boldsymbol{s}) = \frac{1}{n}\sum_{i=1}^{n} \alpha_i(\boldsymbol{s}) = \frac{1}{n}\sum_{i=1}^{n} \frac{\overline{\boldsymbol{a}}_i \boldsymbol{\tau}_i(\boldsymbol{s})}{m}. \tag{3.10}$$

To calculate the derivative $\nabla c(\boldsymbol{s})$, it is easier to utilize the expression in Theorem 4 and then utilize the chain rule. After some manipulations, we obtain

$$\nabla c(\boldsymbol{s}) = \frac{1}{n}\sum_{i=1}^{n} -\frac{\gamma \left(\boldsymbol{B}^T \boldsymbol{W}_i \boldsymbol{\tau}_i(\boldsymbol{s}) \right)^2}{m}. \tag{3.11}$$

Therefore, we focus on calculating $\boldsymbol{\tau}_i(\boldsymbol{s})$. First, by the Matrix Inversion Lemma ([176]) we have

$$\begin{aligned}\boldsymbol{\tau}_i(\boldsymbol{s}) &= \left(\boldsymbol{I}_m + \gamma \boldsymbol{W}_i \left(\sum_{j=1}^{p} s_j \boldsymbol{K}_j\right) \boldsymbol{W}_i\right)^{-1} \overline{\boldsymbol{a}}_i^T \\ &= \left(\boldsymbol{I}_m - \boldsymbol{V}\left(\frac{\boldsymbol{I}_k}{\gamma} + \boldsymbol{V}^T \boldsymbol{W}_i \boldsymbol{V}\right)^{-1} \boldsymbol{V}^T\right) \overline{\boldsymbol{a}}_i^T \\ &= \left(\overline{\boldsymbol{a}}_i^T - \boldsymbol{V}\left(\frac{\boldsymbol{I}_k}{\gamma} + \boldsymbol{V}^T \boldsymbol{W}_i \boldsymbol{V}\right)^{-1} \boldsymbol{V}^T \overline{\boldsymbol{a}}_i^T\right), \end{aligned} \tag{3.12}$$

where $\boldsymbol{V} \in \mathbb{R}^{m \times k}$ is the feature matrix formed by the k columns of $\boldsymbol{B}$ such that $s_j = 1$, and we have suppressed the dependency of $\boldsymbol{V}$ on $\boldsymbol{s}$ for notation ease. Note that in order to compute $\gamma_i(\boldsymbol{s})$ using Eq. (3.9) we need to invert an $m \times m$ matrix, while from Eq. (3.12) we need to invert a $k \times k$ matrix $\frac{\boldsymbol{I}_k}{\gamma} + \boldsymbol{V}^T \boldsymbol{W}_i \boldsymbol{V}$, which only requires $O(k^3)$ calculations. Furthermore, note that calculating $\overline{\boldsymbol{a}}_i \overline{\boldsymbol{a}}_i^T$ only requires $|\Omega_i|$ multiplications where Ω_i is the number of known entries in row i of $\boldsymbol{A}$ as we do not need to multiply on the unknown entries. Similarly, we can compute $\boldsymbol{V}^T \overline{\boldsymbol{a}}_i^T$ in $|\Omega_i|k$ multiplications, and $\boldsymbol{V}^T \boldsymbol{W}_i \boldsymbol{V}$ in $|\Omega_i|k^2$ multiplications.

Therefore, we can compute $\gamma_i(\boldsymbol{s})$ in floating point complexity of $O(|\Omega_i|k^2 + k^3)$. Then to calculate $\overline{\boldsymbol{a}}_i \boldsymbol{\tau}_i(\boldsymbol{s})$ in (3.10) and $-\gamma \left(\boldsymbol{B}^T \boldsymbol{W}_i \boldsymbol{\tau}_i(\boldsymbol{s})\right)^2$ (3.11) only requires $O(|\Omega_i|)$ and $O(|\Omega_i|p)$ calculations respectively. Thus, the total complexity of generating a full cutting plane is:

$$\sum_{i=1}^{n} O(|\Omega_i|p + |\Omega_i|k^2 + k^3) = O(|\Omega|(p + k^2) + nk^3). \tag{3.13}$$

3.2.3 Two-sided Information Case

In this section, we briefly discuss the matrix completion problem under the two-sided information case, and how it reduces to the problem of sparse linear regression. The two sided interpretable matrix completion problem with Tikhonov regularization can be stated

as follows:

$$\min_{\boldsymbol{L}} \frac{1}{nm}\left(\sum_{(i,j)\in\Omega}(X_{ij}-A_{ij})^2+\frac{1}{\gamma}\|\boldsymbol{L}\|_2^2\right) \quad \text{subject to} \quad \boldsymbol{X}=\boldsymbol{U}\boldsymbol{L}\boldsymbol{B}^T \quad \|\boldsymbol{L}\|_0=k, \tag{3.14}$$

where $\boldsymbol{U}\in\mathbb{R}^{n\times p_1}$ is a known matrix of p_1 features of each row, $\boldsymbol{B}\in\mathbb{R}^{m\times p_2}$ is a known matrix of p_2 features of each column, and $\boldsymbol{L}\in\mathbb{R}^{p_1\times p_2}$ is a sparse matrix that has k nonzero entries, ensuring that $\text{Rank}(\boldsymbol{X})\leq k$. We note that in Eq. (3.14) we restrict the support of matrix $\boldsymbol{L}$ to be k, rather than forcing the entries of $\boldsymbol{L}$ to be binary. This is because unlike in the one-sided case, both $\boldsymbol{U}$ and $\boldsymbol{B}$ are known, so we cannot apply the scaling transformation in (3.3).

We denote by $\boldsymbol{u}^i\in\mathbb{R}^{n\times 1}$ the ith column of $\boldsymbol{U}$ and $\boldsymbol{b}^j\in\mathbb{R}^{m\times 1}$ the jth column of $\boldsymbol{B}$. We introduce the matrices $\boldsymbol{W}_i$ as in Theorem 4. Using $\boldsymbol{X}=\boldsymbol{U}\boldsymbol{L}\boldsymbol{B}$, we can write

$$X_{ij}=\sum_{q=1}^{p_1}\sum_{\ell=1}^{p_2}L_{q,\ell}D_{ij}^{q,\ell},$$

where $D_{ij}^{q,\ell}=(\boldsymbol{u}^q(\boldsymbol{b}^\ell)^T)_{ij}$ is the (i,j)th entry of the matrix formed by multiplying qth column of $\boldsymbol{U}$ with ℓth column of $\boldsymbol{B}$. Then, Problem (3.14) becomes:

$$\min_{\boldsymbol{L}} \frac{1}{nm}\left(\sum_{(i,j)\in\Omega}\left(\sum_{q=1}^{p_1}\sum_{\ell=1}^{p_2}L_{q,\ell}D_{ij}^{q,\ell}-A_{ij}\right)^2+\frac{1}{\gamma}\|\boldsymbol{L}\|_2^2\right) \quad \text{subject to} \quad \|\boldsymbol{L}\|_0=k. \tag{3.15}$$

As every $\boldsymbol{D}$ matrix is known, this becomes a sparse regression problem where there are p_1p_2 features to choose from (the $\boldsymbol{D}$ matrices), there are $|\Omega|$ samples (the $\boldsymbol{A}$ matrix), the sparsity requirement is k, the regression coefficients are $\boldsymbol{L}$, and we have Tikhonov regularization. Vectorizing $\boldsymbol{D}$, $\boldsymbol{L}$, and $\boldsymbol{A}$ reduces the problem back to the familiar form of sparse linear regression, that can be solved by the algorithm developed in [23] at scale.

3.3 OptComplete: Stochastic Cutting Plane Speedup

In this section, we introduce OptComplete, a stochastic version of the cutting plane algorithm introduced in Section 3.2. We present theoretical results to show that the stochastic

algorithm recovers the true optimal solution of the original algorithm with high probability without distributional assumptions. We also include a discussion on the dependence of such probability with various factors and its favorable theoretical computational complexity.

3.3.1 Introduction of OptComplete

In the previous section, we showed that through careful evaluation, we can calculate a cutting plane in $O(|\Omega|(p+k^2)+nk^3)$ calculations. However, in very high dimensions where $|\Omega|, n, m$ are extremely large, the cost of generating the exact cutting plane is still prohibitive. Thus, we consider generating approximations of the cutting plane that enable the algorithm to scale for high values for n and m. Specifically, consider the cutting plane function in (3.10), reproduced below:

$$c(\boldsymbol{s}) = \frac{1}{n}\sum_{i=1}^{n} \alpha_i(\boldsymbol{s}),$$

where:

$$\alpha_i(\boldsymbol{s}) = \frac{1}{m}\left(\overline{\boldsymbol{a}}_i\overline{\boldsymbol{a}}_i^T - \overline{\boldsymbol{a}}_i\boldsymbol{V}\left(\frac{\boldsymbol{I}_k}{\gamma} + \boldsymbol{V}^T\boldsymbol{W}_i\boldsymbol{V}\right)^{-1}\boldsymbol{V}^T\overline{\boldsymbol{a}}_i^T\right).$$

We approximate the inner term $\alpha_i(\boldsymbol{s})$ by choosing $1 \le f < m$ samples from $\{1, \cdots, m\}$ *without replacement*, and denoting the set F. We formulate the submatrix $\boldsymbol{V}_F$ with such selected rows, and similarly with $\overline{\boldsymbol{a}}_{Fi}$ and calculate the approximation:

$$\alpha_i(\boldsymbol{s}) \approx \alpha_i^F(\boldsymbol{s}) = \frac{1}{f}\left(\overline{\boldsymbol{a}}_{Fi}\overline{\boldsymbol{a}}_{Fi}^T - \overline{\boldsymbol{a}}_{Fi}\boldsymbol{V}_F\left(\frac{\boldsymbol{I}_k}{\gamma} + \boldsymbol{V}_F^T\boldsymbol{W}_i\boldsymbol{V}_F\right)^{-1}\boldsymbol{V}_F^T\overline{\boldsymbol{a}}_{Fi}^T\right).$$

We choose $1 \le g < n$ samples from $\{1, \cdots, n\}$ without replacement, with the set denoted G. We next calculate an approximation of $c(\boldsymbol{s})$ using the approximated $\alpha_i^F(\boldsymbol{s})$:

$$c(\boldsymbol{s}) \approx \tilde{c}_G^F(\boldsymbol{s}) = \frac{1}{g}\sum_{i\in G} \alpha_i^F(\boldsymbol{s}),$$

where the set F is chosen independently for every row $i \in G$. Then the derivative of $\tilde{c}_G^F(\boldsymbol{s})$ is:

$$\nabla\tilde{c}_G^F(\boldsymbol{s}) = \frac{1}{g}\sum_{i\in G}\frac{(\boldsymbol{B}_{Fi}^T\boldsymbol{W}_{Fi}\boldsymbol{\tau}_i^F(\boldsymbol{s}))^2}{f},$$

where $\boldsymbol{W}_{Fi}$ and $\boldsymbol{B}_{Fi}$ are subsetted using the set F similar to before. Using such approximations, we can derive a stochastic cutting-plane algorithm, which we call OptComplete, as presented in Algorithm 4 below.

Algorithm 4 Stochastic Cutting-plane algorithm for matrix completion with side information.

procedure OptComplete($\boldsymbol{A}, \boldsymbol{B}$) ▷ masked matrix $\boldsymbol{A}$, and feature matrix $\boldsymbol{B}$
  $t \leftarrow 1$
  $\boldsymbol{s}_1 \leftarrow$ random initialization
  $\eta_1 \leftarrow 0$ ▷ Initialize feasible solution variable
  $G \leftarrow |g|$-sized random sample of $\{1, \ldots, n\}$ without replacement
  for $i \in G$ **do** ▷ Generate F for each row in random sample
    $F_i \leftarrow |f|$-sized random sample of $\{1, \ldots, m\}$ without replacement
  end for
  while $\eta_t < \tilde{c}_G^F(\boldsymbol{s}_t)$ **do** ▷ While the current solution is not feasible for iteration t
    $\boldsymbol{s}_{t+1}, \eta_{t+1} \leftarrow \underset{\boldsymbol{s} \in S_k^p, \eta > 0}{\arg\min} \quad \eta \quad \text{s.t.} \quad \eta \geq \tilde{c}_G^F(\boldsymbol{s}_i) + \nabla \tilde{c}_G^F(\boldsymbol{s}_i)^T(\boldsymbol{s} - \boldsymbol{s}_i) \quad \forall i \in [t]$
    $G \leftarrow |g|$-sized random sample of $\{1, \ldots, n\}$ without replacement
    for $i \in G$ **do** ▷ Generate F for each row in random sample
      $F_i \leftarrow |f|$-sized random sample of $\{1, \ldots, m\}$ without replacement
    end for
    $t \leftarrow t + 1$
  end while
  $\boldsymbol{s} \leftarrow \boldsymbol{s}_{t-1}$
  $i \leftarrow 1$
  for $i < n$ **do** ▷ Fill each row $\boldsymbol{x}_i$ of final output matrix $\boldsymbol{X}$
    $\boldsymbol{x}_i \leftarrow \boldsymbol{B}^{\boldsymbol{s}}((\boldsymbol{B}^{\boldsymbol{s}})^T \boldsymbol{W}_i \boldsymbol{B}^{\boldsymbol{s}})^{-1}(\boldsymbol{B}^{\boldsymbol{s}})^T \overline{\boldsymbol{a}}_i^T$ ▷ $\boldsymbol{B}^{\boldsymbol{s}}$ is submatrix of $\boldsymbol{B}$ with $\boldsymbol{s}$ columns
  end for
  return $\boldsymbol{X}$ ▷ Return the filled matrix $\boldsymbol{X}$
end procedure

For this algorithm to work, we need the approximation $\tilde{c}_G^F(\boldsymbol{s})$ and its derivative to be close to the nominal values. Furthermore, the approximated cutting planes should not cutoff the true solution. In the next section, we show that OptComplete enjoys such properties with high probability. In Section 3.3.3, we discuss how to select the size of f and g.

3.3.2 Main Theoretical Results

We first show that the inner approximation is close to the true term with high probability:

Theorem 5. *Let $\boldsymbol{A}$ be a partially known matrix, $\boldsymbol{B}$ a known feature matrix, and $\boldsymbol{W}_i$ as defined in Theorem 4. Let F be a random sample of size f from the set $\{1,\dots,m\}$, chosen without replacement. With probability at least $1-\epsilon$, we have*

$$|\alpha_i(\boldsymbol{s}) - \alpha_i^F(\boldsymbol{s})| \leq \sqrt{\frac{Mk\log(\frac{k}{\epsilon})}{f}}, \qquad \forall i \in [m], \quad \forall \boldsymbol{s} \in S_k^p,$$

$$\left\| \frac{(\boldsymbol{B}^T\boldsymbol{W}_i\boldsymbol{\tau}_i(\boldsymbol{s}))^2}{m} - \frac{(\boldsymbol{B}_{Fi}^T\boldsymbol{W}_{Fi}\boldsymbol{\tau}_i^F(\boldsymbol{s}))^2}{f} \right\|_2 \leq \sqrt{\frac{M'(p+k)\log(\frac{k}{\epsilon})}{f}}, \qquad \forall i \in [m], \quad \forall \boldsymbol{s} \in S_k^p,$$

where M, M' are absolute constants.

We see that, without assumptions on the data, the inner approximation for both the value and the derivative follows a bound with $O\left(\sqrt{\frac{(p+k)}{f}}\right)$ terms with very high probability. Furthermore, inverting the statements give that, for all $i \in [m]$ and all $\boldsymbol{s} \in S_k^p$:

$$\mathbb{P}\left(|\alpha_i(\boldsymbol{s}) - \alpha_i^F(\boldsymbol{s})| \geq \delta\right) \leq k\exp\left(-\frac{f\delta^2}{Mk}\right),$$

$$\mathbb{P}\left(\left\| \frac{(\boldsymbol{B}^T\boldsymbol{W}_i\boldsymbol{\tau}_i(\boldsymbol{s}))^2}{m} - \frac{(\boldsymbol{B}_{Fi}^T\boldsymbol{W}_{Fi}\boldsymbol{\tau}_i^F(\boldsymbol{s}))^2}{f} \right\|_2 \geq \delta\right) \leq k\exp\left(-\frac{f\delta^2}{M'(p+k)}\right).$$

So the failure probability drops off exponentially with increasing bound δ, reflecting a gaussian tail structure for $\alpha_i(\boldsymbol{s})$ and $(\boldsymbol{B}_{Fi}^T\boldsymbol{W}_{Fi}\boldsymbol{\tau}_i^F(\boldsymbol{s}))^2$. The proof of it relies on Hoeffding and Chernoff-type results, and is contained in Appendix B.1.

Using this result, we are able to prove a tight deviation bound for the approximated cost function $\tilde{c}_G^F(\boldsymbol{s})$ and $\nabla\tilde{c}_G^F(\boldsymbol{s})$:

Theorem 6. *Let $\boldsymbol{A}$ be a partially known matrix, $\boldsymbol{B}$ a known feature matrix, and $\boldsymbol{W}_i$ as defined in Theorem 4. Let G be a random sample of size g from $\{1,\dots,n\}$ chosen without replacement. Then, for each $i \in G$, we let F_i be a random sample of size f from the set $\{1,\dots,m\}$, all chosen without replacement. We have, with probability at least $1-\epsilon$:*

$$|\tilde{c}_G^F(\boldsymbol{s}) - c(\boldsymbol{s})| \leq \sqrt{\frac{Ak\log\left(\frac{k}{\epsilon}\right)}{g}}, \qquad \forall \boldsymbol{s} \in S_k^p,$$

$$\|\nabla \tilde{c}_G^F(\boldsymbol{s}) - \nabla c(\boldsymbol{s})\|_2 \leq \sqrt{\frac{B(p+k)\log\left(\frac{k}{\epsilon}\right)}{g}}, \qquad \forall \boldsymbol{s} \in S_k^p,$$

where A, B are absolute constants.

Similar to the inner approximations, $c_G^F(\boldsymbol{s})$ and $\nabla c_G^F(\boldsymbol{s})$ has Gaussian tails. Furthermore, the scaling here only depends on g and not f: This shows that the error of the inner approximation is dominated by the outer sampling of the rows G. The proof is contained in Appendix B.2.

Then, using this result, we are able to prove our main result for OptComplete. We first introduce a new definition:

Definition 3. *The **convexity parameter** a of the cost function $c(s)$ is defined as the largest positive number for which the the following statement is true:*

$$c(\boldsymbol{s}) \geq c(\boldsymbol{s}_0) + \nabla c(\boldsymbol{s}_0)^T(\boldsymbol{s} - \boldsymbol{s}_0) + \frac{a^2}{2}(\boldsymbol{s} - \boldsymbol{s}_0)^T(\boldsymbol{s} - \boldsymbol{s}_0) \quad \forall \boldsymbol{s}, \boldsymbol{s}_0 \in S_k^p \tag{3.16}$$

We have $c(\boldsymbol{s}) = \sum_{i=1}^n \alpha_i(\boldsymbol{s})$, and by the definition of $\alpha_i(\boldsymbol{s})$ in (3.9), $\alpha_i(\boldsymbol{s})$ is strongly convex in $\boldsymbol{s}$ for all i, so $\alpha > 0$.

Now, we state our main theorem for OptComplete.

Theorem 7. *For the matrix completion problem (3.4), let $\boldsymbol{B} \in \mathbb{R}^{m\times p}$ be a known feature matrix, $\boldsymbol{A} \in \mathbb{R}^{n\times m}$ a matrix with entries partially known, and OptComplete as defined in Algorithm 4. Then OptComplete terminates in a finite number of steps C with probability 1, and finds an optimal solution of (3.4) with probability at least $1 - kC\exp\left(-\frac{Da^4 g}{(p+k)}\right) - k(C+1)\exp\left(\frac{-Eg}{k}\right)$ where D, E are constants independent of C, f, g, k, p, and a is the convexity parameter of the function $c(\boldsymbol{s})$.*

The proof is contained in Appendix B.3. This theorem shows that OptComplete is able to find the optimal solution of the original binary convex optimization problem with exponentially vanishing failure probability that scales at least as fast as $O\left(\exp\left(\frac{-g}{(p+k)}\right)\right)$. The theorem requires no assumptions on the data, and thus applies generally. We again note that the bound does not depend on f and only on g: we discuss how this informs our selection of the size of f and g in the next section.

3.3.3 Sampling Size and Computational Complexity

To select an appropriate f and g, we first note that [34] showed that to complete a square $N \times N$ matrix of rank k, we need at least $O(kN \log N)$ elements. Assume an average known rate of $\alpha = \frac{|\Omega|}{mn}$ in the original matrix $\boldsymbol{A}$, the expected number of known elements under a sampling of f and g is αfg. Using $N^2 = mn$, we need that:

$$\alpha fg \geq c \cdot k\sqrt{nm}\log(\sqrt{nm}) \tag{3.17}$$

for some constant c. Theorem 7 showed that the bound on failure probability scales with $O\left(\exp\left(\frac{-g}{(p+k)}\right)\right)$, and thus we cannot have g too small. Using (3.13), the complexity of the cutting plane with f and g samples are:

$$O(\alpha fg(p + k^2) + gk^3). \tag{3.18}$$

Therefore, if we fix the expected known elements (αfg) constant, it is more advantageous to select a smaller g, as g scales with k^3. Thus, we set:

$$f = \min\left(\frac{ck\sqrt{mn}\log(\sqrt{mn})}{\alpha \min(g_0, n)}, m\right), \qquad g = \min(g_0, n), \tag{3.19}$$

Experimentally, we found $g_0 = 100$, $c = 1$ to generate good results (the results were similar for $\frac{1}{2} \leq c \leq 2$). Therefore, by (3.18), the approximated cutting plane has a computational complexity of:

$$O\left(k\sqrt{mn}\log(\sqrt{mn})(p + k^2)\right). \tag{3.20}$$

This scales in a square root fashion in n and m, rather than linearly in n and m for the exact cutting plane. This allows OptComplete to enjoy a considerable speedup compared to CutPlanes during the calculation of the cutting planes, as we demonstrate in Section 3.4.

3.4 Synthetic Data Experiments

We assume that the matrix $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{V} + \boldsymbol{E}$, where $\boldsymbol{U} \in \mathbb{R}^{n\times k}$, $\boldsymbol{V} \in \mathbb{R}^{k\times m}$, and $\boldsymbol{E}$ is an error matrix with individual elements sampled from $N(0, 0.01)$. We sample the elements of $\boldsymbol{U}$ and $\boldsymbol{V}$ from a uniform distribution of $[0, 1]$, and then randomly select a fraction $\mu = 1 - \alpha$ to be missing. We formulate the feature matrix $\boldsymbol{B}$ by combining $\boldsymbol{V} \in \mathbb{R}^{k\times m}$ with a confounding matrix $\boldsymbol{Z} \in \mathbb{R}^{(p-k)\times m}$ that contains unnecessary factors sampled similarly from the Uniform $[0, 1]$ distribution. We run OptComplete on a server with 16 CPU cores, using Gurobi 8.1.0. For each combination (m, n, p, k, μ), we ran 10 tests and report the median value for every statistic.

We report the following statistics with $\boldsymbol{s}^*$ being the ground-truth factor vector, and $\overline{\boldsymbol{s}}$ the estimated factor vector.

- n, m - the dimensions of $\boldsymbol{A}$.
- p - the number of features in the feature matrix.
- k - the true number of features.
- μ - The fraction of missing entries in $\boldsymbol{A}$.
- T - the total time taken for the algorithm.
- MAPE - the Mean Absolute Percentage Error (MAPE) for the retrieved matrix $\hat{\boldsymbol{A}}$:

$$\text{MAPE} = \frac{1}{|\mathcal{S}|} \sum_{(i,j)\in\mathcal{S}} \frac{|\hat{A}_{ij} - A_{ij}|}{|A_{ij}|},$$

  where $\mathcal{S} = \Omega^c$ is the set of missing data in $\boldsymbol{A}$.

Since the concept of Interpretable Matrix Completion is new, there is a lack of directly comparable algorithms in the literature. Thus, in lieu, we compare OptComplete to state-of-the-art solvers for Inductive Matrix Completion and general matrix completion, which are:

- IMC by [131] - This algorithm is a well-accepted benchmark for testing Inductive Matrix Completion algorithms.

- SoftImpute-ALS (SIALS) by [86] - This is widely recognized as a state-of-the-art matrix completion method without feature information. It has among the best scaling behavior across all classes of matrix completion algorithms as it utilizes fast alternating least squares to achieve scalability.

We use the best existing implementations of IMC (Matlab 2018b) and SIALS (R 3.4.4, package softImpute) with parallelization on the same server.

We further compare our algorithm to CutPlanes, the original cutting plane algorithm developed in Section 3.2 to demonstrate the speedup provided by the stochastic procedure.

To illustrate the performance of the cutting plane algorithm against general purpose convex mixed-integer non-linear optimization (MINLO) solvers, we also compare against Pavito V0.2.0 ([46]), a convex MINLO solver based on outer approximations. A recent review in [108] illustrates the competitive performance of (an earlier version of) Pavito on general convex MINLO problems. We further attempted to compare this algorithm against the proprietary solver Artelys Knitro ([30]), but the solver returned a non-optimal solution even when it claimed optimality was reached, resulting in a large MAPE, and thus the comparisons are omitted.

The Pavito solver requires a specification of a continuous non-linear solver and a mixed-integer solver. To ensure the comparisons between the cutting-plane algorithm and the Pavito solver are fair, we supplied Pavito with Ipopt V3.13.2 as the non-linear solver, and Gurobi V8.1.0 as the mixed-integer solver. Furthermore, instead of utilizing the automatic differentiation capabilities of the Pavito solver to evaluate the derivatives of the cost function $c(\boldsymbol{s})$, we provided access to the same derivative routine that the cutting plane algorithms (OptComplete and CutPlanes) utilize, which provided an average speedup of about $2.5x$ compared to the automatic differentiation procedure.

We randomly selected 20% of those elements masked to serve as a validation set. The regularization parameter γ of OptComplete, the rank parameter of IMC and the penalization parameter λ of IMC and SIALS are selected using the validation set. The results

are separated into sections below. The first five sections modify one single variable out of n, m, p, k, μ to investigate OptComplete's scalability, where the leftmost column indicates the variable modified. The last section compares the four algorithms scalability for a variety of parameters that reflect more realistic scenarios. Overall, we see that OptComplete

	n	m	p	k	μ%	OptComplete		CutPlanes		IMC		SIALS		Pavito	
						T	MAPE	T	MAPE	T	MAPE	T	MAPE	T	MAPE
μ	100	100	15	5	20%	1.7s	0.1%	6.0s	0.1%	0.03s	0.01%	0.02s	0.3%	3.6s	0.1%
	100	100	15	5	50%	0.9s	0.02%	4.5s	0.02%	0.07s	0.5%	0.03s	0.9%	2.7s	0.02%
	100	100	15	5	80%	0.6s	0.03%	2.5s	0.03%	0.09s	1.3%	0.06s	5.6%	2.5s	0.03%
	100	100	15	5	95%	0.2s	0.04%	1.2s	0.04%	0.12s	12.1%	0.12s	7.4%	2.0s	0.04%
n	100	100	15	5	50%	0.9s	0.02%	4.5s	0.02%	0.07s	0.5%	0.03s	0.9%	2.7s	0.02%
	10^3	100	15	5	50%	5.1s	0.01%	72.5s	0.01%	0.6s	0.4%	0.1s	0.2%	13.2s	0.01%
	10^4	100	15	5	50%	16.4s	0.004%	957s	0.004%	4.5s	0.3%	6.5s	0.5%	119s	0.004%
	10^5	100	15	5	50%	50.7s	0.003%	10856s	0.003%	32.7s	0.1%	38s	3.0%	1275s	0.003%
m	100	100	15	5	50%	0.9s	0.02%	4.5s	0.02%	0.07s	0.5%	0.03s	0.9%	2.7s	0.02%
	100	10^3	15	5	50%	0.7s	0.01%	18.6s	0.01%	0.8s	0.3%	0.1s	0.5%	6.1s	0.01%
	100	10^4	15	5	50%	0.6s	0.004%	68.5s	0.004%	6.2s	0.2%	0.8s	0.3%	25.9s	0.004%
	100	10^5	15	5	50%	1.3s	0.002%	259s	0.002%	56.2s	0.1%	12.7s	0.8%	173s	0.002%
p	100	100	15	5	50%	0.9s	0.02%	4.5s	0.02%	0.07s	0.5%	0.03s	0.9%	2.7s	0.02%
	100	100	50	5	50%	2.0s	0.02%	18.0s	0.02%	0.3s	0.6%	0.03s	0.9%	4.5s	0.02%
	100	100	200	5	50%	12.1s	0.02%	95.9s	0.02%	1.9s	0.8%	0.03s	0.9%	18.4s	0.02%
	100	100	10^3	5	50%	90.3s	0.02%	680s	0.02%	10.4s	1.0%	0.03s	0.9%	92.4s	0.02%
k	100	100	15	5	50%	0.9s	0.02%	4.5s	0.02%	0.07s	0.5%	0.03s	0.9%	2.7s	0.02%
	100	100	50	10	50%	20.7s	0.06%	130s	0.06%	0.20s	1.2%	0.1s	0.8%	4.0s	0.06%
	100	100	50	20	50%	240s	0.07%	1584s	0.07%	0.35s	2.1%	0.21s	1.0%	6.1s	0.07%
	100	100	50	30	50%	980s	0.09%	8461s	0.09%	0.5s	3.3%	0.43s	2.8%	9.5s	0.09%
	100	100	15	5	95%	0.2s	0.04%	1.2s	0.04%	0.12s	12.1%	0.12s	7.4%	2.0s	0.04%
	10^3	10^3	50	5	95%	1.4s	0.006%	3.5s	0.006%	4.6s	4.7%	2.8s	12.5%	4.3s	0.006%
	10^4	10^3	100	5	95%	5.7s	0.002%	35.2s	0.002%	18s	2.5%	20.7s	12.6%	37.6s	0.002%
	10^5	10^3	200	10	95%	52s	0.001%	1520s	0.001%	295s	1.7%	420s	4.6%	397s	0.001%
	10^5	10^4	200	10	95%	98s	0.001%	5769s	0.001%	1750s	0.5%	4042s	4.1%	2470s	0.001%
	10^6	10^4	200	10	95%	480s	0.001%	N/A	N/A	13750s	0.3%	25094s	2.5%	19580s	0.001%
	10^6	10^5	200	10	95%	680s	0.001%	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	10^6	10^6	200	10	95%	1415s	0.001%	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

Table 3.1: Comparison of OptComplete, IMC and SIALS on synthetic data. N/A means the algorithm did not complete running in 20 hours, corresponding to 72000 seconds.

achieves near-exact retrieval on all datasets evaluated, and successfully recovers the factors in the ground truth. The solutions (and its error) also matches with that of CutPlanes, the standard cutting plane algorithm, and also Pavito, the general convex MINLO solver. The non-zero MAPE is due to the random noise added resulting in slightly perturbed coefficients.

For the realistic and large data sizes in the last panel, we see that OptComplete not only achieves near-exact retrieval, it does so while requiring considerably less time than IMC and SIALS at the same time. For m, n on the scale of $n = 10^6$ and $m = 10^4$, OptComplete is

over 20 times faster than IMC and over 40 times faster than SIALS. It also achieves similar magnitude of performance against Pavito, despite it having access to Ipopt as a state-of-the-art continuous solver. At the scale of $n = 10^6$ and $m = 10^5$, IMC, SIALS, and Pavito, did not finish running within 20 hours, while OptComplete completed in just under 12 minutes. We also see that OptComplete achieves very significant speedups compared to the standard cutting plane algorithm - up to 60x at the scale of $n = 10^5$ and $m = 10^4$.

We analyze the scaling of OptComplete as a function of:

1. μ - The algorithm is able to retrieve the exact factors used even with 95% of missing data. Furthermore, the running time decreased with increasing missing entries, consistent with the fact that is computational complexity scales with $|\Omega|$.

2. n - The algorithm has good scalability in n, reflecting its $O(\sqrt{n}\log(n))$ type complexity with respect to n. This allows the algorithm to support matrices with n in the 10^6 range. Its scaling behavior is superior to IMC, SIALS, and Pavito.

3. m - The algorithm scales exceptionally well in m. We observe that empirically the algorithm runtime seems to grow much slower than the theoretical $O(\sqrt{m}\log(m))$ dependence with respect to m in generation of the cutting planes. A closer examination reveals that as m increases, the number of cutting planes generated by Gurobi is decreasing. Qualitatively, this can be explained by a larger m giving the algorithm more signal to find which k features are the correct ones out of the p ones. We note that such behavior is also exhibited by CutPlanes and Pavito, as it roughly scales proportional to $\sqrt{m}$ rather than the proportional to m as expected. We see that IMC and SIALS scales as proportional to m.

4. p - The algorithm scales relatively well in p, which reflects the performance of the Gurobi solver. We empirically observe that Gurobi is generating roughly cutting planes that is growing slower than proportional to p. Thus, as each cutting plane is $O(p)$ with respect to p, we expect the runtime to scale proportional to $(p - p^2)$, as is observed here. We note that OptComplete achieves similar scaling behavior as IMC in p, while Pavito seems to scale proportional to p. Note here the SoftImpute algorithm does not

utilize feature information and thus a change in p does not affect the algorithm's run speed.

5. k - The algorithm does not scale very well in k. We empirically observe that Gurobi solver is roughly generating proportional to k cutting planes and each cutting plane has cubic dependence on k. It appears that SoftImpute, IMC, and Pavito almost have a linear scaling behavior. However, in most applications, such as recommendation systems or low-rank retrieval, k is usually kept very low ($k \leq 30$), so this is not a particular concern.

3.5 Real-World Experiments

In this section, we report on the performance of OptComplete on the Netflix Prize dataset [15]. This dataset was released in a competition to predict ratings of customers on unseen movies, given over 10 million ratings scattered across $500,000$ people and $16,000$ movies. Thus, when presented in a matrix $\boldsymbol{A}$ where A_{ij} represents the rating of individual i on movie j, the goal is to complete the matrix $\boldsymbol{A}$ under a low-rank assumption.

The OptComplete feature matrix $\boldsymbol{B}$ is constructed using data from the TMDB Database, and covers 59 features that measure geography, popularity, top actors/actresses, box office, runtime, genre and more. The full list of 59 features is contained in Appendix A.3.

For this experiment, we included movies where all 59 features are available, and people who had at least 5 ratings present. This gives a matrix of $471,268$ people and $14,538$ movies. The slight reduction of size from the original data is due to the lack of features for about $2,000$ niche movies. To observe the scalability of OptComplete, we created five data sets:

1. Base - $\boldsymbol{A}_1$ has dimensions $3,923 \times 103$.
2. Small - $\boldsymbol{A}_2$ has dimensions $18,227 \times 323$.
3. Medium - $\boldsymbol{A}_3$ has dimensions $96,601 \times 788$.
4. Large - $\boldsymbol{A}_4$ has dimensions $471,268 \times 1760$.
5. Full - $\boldsymbol{A}$ has dimensions $471,268 \times 14,538$.

These sizes are constructed such that the total number of elements in $\boldsymbol{A}$ in the successive sizes are approximately different by approximately an order of magnitude.

For each individual matrix, we uniformly randomly withhold 20% of the ratings as a test set $\mathcal{S}$, and use the remaining 80% of ratings to impute a complete matrix $\hat{\boldsymbol{A}}$ - we perform cross-validation on the appropriate hyperparameters. Then, we report MAPE.

For comparison, we again use IMC, SIALS, and Pavito. We set the maximum rank of SIALS to be k - the rank optimized for in OptComplete. The results are listed below:

n	m	p	k	μ%	OptComplete		IMC		SIALS		Pavito	
					T	MAPE	T	MAPE	T	MAPE	T	MAPE
3,923	103	59	5	92.6%	6s	29.4%	1s	34.2%	0.3s	31.2%	24s	29.4%
18,227	323	59	5	94.8%	12s	21.8%	5s	29.1%	4.1s	24.1%	137s	21.8%
96,601	788	59	5	94.2%	25s	20.9%	38s	28.7%	30.4s	21.3%	795s	20.9%
471,268	1,760	59	5	93.6%	102s	18.8%	460s	24.6%	430s	19.8%	4012s	18.8%
471,268	14,538	59	5	94.1%	170s	15.7%	3921s	21.5%	5300s	16.7%	8705s	15.7%

Table 3.2: Comparison of methods on Netflix data for $k = 5$.

n	m	p	k	μ%	OptComplete		IMC		SIALS		Pavito	
					T	MAPE	T	MAPE	T	MAPE	T	MAPE
3,923	103	59	10	92.6%	11s	30.4%	1s	36.7%	1s	35.8%	36s	30.4%
18,227	323	59	10	94.8%	20s	24.0%	13s	32.5%	7s	28.9%	213s	24.0%
96,601	788	59	10	94.2%	46s	22.3%	84s	29.6%	51s	22.8%	1077s	22.3%
471,268	1,760	59	10	93.6%	260s	20.7%	1022s	24.8%	870s	20.7%	6238s	20.7%
471,268	14,538	59	10	94.1%	380s	19.6%	8704s	23.1%	10240s	20.0%	10879s	19.6%

Table 3.3: Comparison of methods on Netflix data for $k = 10$.

We can see that OptComplete outperforms both IMC and SIALS in accuracy across the datasets under different k; furthermore in the two largest datasets OptComplete ran 10x to 20x faster than IMC and SIALS. It is especially interesting that OptComplete achieves significantly higher accuracy than IMC strictly dominates OptComplete in its feasible set. We believe this is due to the high inherent noise in the real-world data, and the linear combinations that are fitted to the data do not generalize well to the out-of-sample case.

Here we see some evidence of this as an increase from $k = 5$ to $k = 10$ decreased out-of-sample performance across the board as additional factors are actually not very helpful in predictive customer tastes. The decline for OptComplete and IMC were especially higher due to the fact that the possible factors are fixed and thus an increase in the number of factors caused some non-predictive factors to be included. We further observe that OptComplete is able to converge to the same solution as Pavito does, but in significantly less time. In the largest instances, this resulted in a speedup of roughly $50\times$ for the $k = 5$ case, and $30\times$ for the $k = 10$ case.

For the $k = 5$ case, OptComplete identified the following as the top factors that influences an individual's rating:

- IMDB Rating
- Genre: Drama
- Released within last 10 years
- Number of Top 100 Actors
- Produced in US

These factors provide an intuitive explanation of the individual ratings of each customer in terms of a small number of factors, while exceeding the high predictive accuracy of SIALS. It shows that factors most affecting movie ratings include genre, existence of famous actors, general public opinion, recency, and US production. In contrast, IMC identified dense linear combinations of features. For example, one such identified feature in the $k = 5$ case for the largest matrix setting was (we removed all features that had a coefficient less than 10^{-10}):

$$\begin{aligned}
&5.6 \times \text{IMDB Rating} - 0.34 \times \text{Number of Reviews} \\
&+ 0.00082 \times \text{Total Production Budget} + 9.1 \times \text{Number of Male Actors} \\
&- 1.7 \times \text{Number of Female Actors} + 4.5 \times \text{Genre: Drama} \\
&- 8.6 \times \text{Genre: Sci-fi} + 12.5 \times \text{Total Runtime} \\
&- 2.34 \times \text{Number of Top 250 Actors} - 0.071 \times \text{Number of Top 1000 Actors}
\end{aligned}$$

$$+ 0.0000023 \times \text{Box Office Revenue} - 0.66 \times \text{Genre: Crime}$$

$$+ 0.97 \times \text{Number of Female Actors} - 1.45 \times \text{Released in June}$$

$$+ 8.33 \times \text{Released within 10 Years} - 3.22 \times \text{Released in December}.$$

This dense linear combination of 16 features makes it challenging for decision makers to understand what is important. Furthermore, in total across the 5 identified linear combinations, IMC utilized 35 features with (out of 59 available). This dense set of utilized features inhibits the ability of decision makers at Netflix to utilize the insights from the matrix completion to adjust their product (e.g. what features to show on the movie details screen) to better reflect the key features that customers care about.

3.6 Conclusions

We have presented OptComplete, a scalable algorithm to retrieve a low-rank matrix in the presence of side information. Compared with state of the art algorithms for matrix completion, OptComplete exceeds current benchmarks on both scalability and accuracy and provides insight on the factors that affect the ratings.

Chapter 4

Stochastic Cutting Planes for Data-Driven Optimization

4.1 Introduction

The cutting plane method, or the outer approximation method, has been a popular method to solve Mixed Integer Non-Linear Optimization (MINLO) problems since its original introduction in [57] and [144]. It tackles problems of the form:

$$\begin{aligned} &\min_{\boldsymbol{z}\in\mathcal{Z},\boldsymbol{\theta}\in\Theta} f(\boldsymbol{z},\boldsymbol{\theta}) \\ &\text{subject to } g_i(\boldsymbol{z},\boldsymbol{\theta}) \leq 0, \;\; i \in [m], \end{aligned}$$

where $\boldsymbol{z}$ and $\boldsymbol{\theta}$ are integer and continuous variables constrained to be in sets $\mathcal{Z}$ and Θ, respectively, and f, g_i are convex. The method aims to generate the tangent plane of both the objective and the constraints at an initial feasible solution $(\boldsymbol{z}_0, \boldsymbol{\theta}_0)$, and solve the resultant mixed integer linear optimization (MILO) instead in order to find an optimal solution $(\boldsymbol{z}_1, \boldsymbol{\theta}_1)$. Then, the process is repeated where the tangent plane around $(\boldsymbol{z}_1, \boldsymbol{\theta}_1)$ is added until the solution stops updating (with a tolerance of ϵ). [144] proved that this method does indeed converge to the optimal solution in finite steps, and illustrated its practical effectiveness.

With the rise in availability of data in recent years, the cutting plane method has been applied to many data-driven optimization problems, ranging from risk minimization [71],

matrix completion [22], to multiple kernel learning and inference in graphical models [72].

However, as the data sizes continue to grow, cutting plane approaches face challenges in scaling as generating the full tangent planes usually involves evaluating the objective function and its derivative across all samples. With sample sizes easily reaching scales of 10^6 or higher, this presents a computational difficulty for utilizing cutting-plane methods in large-scale real world applications.

In this paper, we introduce a stochastic variant of the cutting plane algorithm that aims to greatly increase the scalability of the cutting plane problem by multiple orders of magnitude. Specifically, instead of aiming to generate the exact tangent plane in each iteration, we utilize an independent random subset of the data to generate an approximation of the tangent plane. A parallel analogy of this idea in continuous optimization is stochastic gradient descent, where only a subset of the samples is considered to calculate a gradient, resulting in large performance increases that enables the optimization of millions of variables (such as in a convolutional neural network).

Despite the simplicity of the idea, the authors could not find any detailed work exploring the theoretical properties and computational performance of this approach. In this work, we characterize the general convex data-driven optimization problems that stochastic cutting planes can be applied to, and demonstrate that the proposed method is able to converge to a near optimal solution with an exponentially vanishing failure probability. We illustrate that on multiple distinct problems (sparse regression, support vector machines and stochastic knapsack problems), stochastic cutting planes record an order-of-magnitude speedup compared to standard cutting planes and compare competitively with state-of-the art specialized algorithms.

4.1.1 Structure

In Section 4.2, we introduce the general class of convex data-driven optimization problems to which we apply our framework. In Section 4.3, we prove that the stochastic cutting plane algorithm converges to an optimal solution of the original problem with exponentially vanishing failure probability under very weak conditions. In Section 4.4, we present computational results on a variety of problems to show that the algorithm considerably outperforms the

standard cutting plane method, and extends scalability by multiple orders of magnitude.

4.1.2 Literature

[57] first introduced the exact cutting-plane method in the context of mixed integer non-linear optimization (MINLO) problems that are linear in the integer variables. [144] extended the framework to the general class of MINLO problems and provided theoretical guarantees for the case that the MINLO is convex. [140] investigated an approximate cutting-plane method that involved generating a sequence of hyperplanes using non-optimal dual solutions to the inner NLO problem. The approximate cutting planes generated are always valid for the original objective function, but the approximation is only effective "near an optimal solution".

Many other works entitled "stochastic cutting planes" or similar are primarily focused on randomized constraint generation in the presence of a large (possibly infinite) number of constraints, such as [171] or [31] in robust optimization. Concretely, they consider problems of the form:

$$\begin{aligned} &\min_{\boldsymbol{z}\in\mathcal{Z},\boldsymbol{\theta}\in\Theta} f(\boldsymbol{z},\boldsymbol{\theta}) \\ &\text{subject to } g_i(\boldsymbol{z},\boldsymbol{\theta}) \leq 0, \quad i \in [m], \end{aligned}$$

where m is very large or an infinite set. The proposed algorithms generally first solves the problem with only a random finite subset of constraints, and then add the remaining constraints iteratively (and randomly) till convergence. In particular, they assume that the function $f(\boldsymbol{z},\boldsymbol{\theta})$ is relatively simple to evaluate. This work focuses on the case where the objective function takes significant time to calculate and proves theoretical guarantees on optimal solution retrieval using stochastic cutting planes. Furthermore, we demonstrate real-world scalability with data beyond 10^6 samples.

In stochastic optimization, sample average approximation is frequently utilized to provide an approximation to the objective function (which is usually of the form of an expected value). For stochastic MINLO problems, there has been work (see e.g. [174, 103]) adapting the cutting planes methodology for the stochastic problem by solving the sample average

approximation at every iteration using a new sample of size N multiple times. This work complements previous work by exploring how to solve the sample average approximation with sample size N effeiciently, when N is large. In particular in Section 4.4, we demonstrate how stochastic cutting planes generates an 100x speedup to solve the sample average approximation version of the static stochastic knapsack problem (SSKP).

[22] utilized a similar idea to speed up the cutting-plane method utilized to solve a variant of the matrix completion problem. The theoretical guarantees in this paper generalizes the result to a wide class of MINLO problems.

4.2 Problem Formulation and Examples

We consider a general class of mixed-integer data-driven optimization problems with integer variables $\boldsymbol{z} \in \mathbb{Z}^{p_1}$ and continuous variables $\boldsymbol{\theta} \in \mathbb{R}^{p_2}$. In addition, there is data $\boldsymbol{d}_1, \ldots, \boldsymbol{d}_N$ obtained either through historical results or simulations. As common in these settings, the goal is to find a representation of the data under some model driven by the variables $\boldsymbol{z}, \boldsymbol{\theta}$.

Utilizing the shorthand $[N]$ to represent $\{1, \ldots, N\}$, we can write the objective function of such data driven problem as:

$$f(\boldsymbol{z}, \boldsymbol{\theta}; [N]) := f(\boldsymbol{z}, \boldsymbol{\theta}; \boldsymbol{d}_1, \ldots, \boldsymbol{d}_N),$$

where we use the notation $f(\boldsymbol{z}, \boldsymbol{\theta}; [N])$ to denote that the objective function is created with samples in the set $[N]$. One of the most common examples of this type of objective function is data-driven risk minimization, where we have $\boldsymbol{d}_i = (\boldsymbol{x}_i, \boldsymbol{y}_i)$, and the objective function is:

$$f(\boldsymbol{z}, \boldsymbol{\theta}; [N]) = \frac{1}{N} \sum_{i=1}^{N} l(\boldsymbol{y}_i, \boldsymbol{x}_i, \boldsymbol{z}, \boldsymbol{\theta}). \tag{4.1}$$

Here $l(\cdot, \cdot)$ is some pre-defined loss function describing the difference between the target $\boldsymbol{y}_i$ a model fitted by $\boldsymbol{x}_i, \boldsymbol{z}, \boldsymbol{\theta}$. Despite this covers a core class of problems of interest, we here define $f(\boldsymbol{z}, \boldsymbol{\theta}; [N])$ more generally in Assumption 5 introduced below, as some problems amenable to our method do not admit a structure that is additively separable in data.

We consider the data-driven optimization problem:

$$\min_{z\in\mathcal{Z},\theta\in\Theta} f(\boldsymbol{z},\boldsymbol{\theta};[N]) \tag{4.2}$$
$$\text{subject to } g_i(\boldsymbol{z},\boldsymbol{\theta})\leq 0,\quad i\in[m],$$

with the following assumptions:

Assumption 3 (Convexity)**.** *$f(\boldsymbol{z},\boldsymbol{\theta};[N])$ and $g_i(\boldsymbol{z},\boldsymbol{\theta})$ are at least once differentiable and convex in $\boldsymbol{z},\boldsymbol{\theta}$ for all $i\in[m]$.*

Assumption 4 (Compactness)**.** *$\mathcal{Z}$ is a finite set in $\mathbb{Z}^{p_1}$ and Θ is a nonempty compact convex set in $\mathbb{R}^{p_2}$ defined by a system of linear inequality constraints.*

Assumption 5 (Data Concentration)**.** *Let $f(\boldsymbol{z},\boldsymbol{\theta};[N])$ be a data-driven objective function and S^n be a random subset of $[N]$ of size n selected without replacement. Then we have absolute constants M and M' independent of p_1,p_2,n such that for all $n,\epsilon,\boldsymbol{z}\in\mathcal{Z}$ and $\boldsymbol{\theta}\in\Theta$, we have with probability at least $1-\epsilon$:*

$$|f(\boldsymbol{z},\boldsymbol{\theta};S^n)-f(\boldsymbol{z},\boldsymbol{\theta};[N])|\leq\sqrt{M\left(\frac{1}{n}-\frac{1}{N}\right)\log(\frac{1}{\epsilon})},$$
$$\|\nabla f(\boldsymbol{z},\boldsymbol{\theta};S^n)-\nabla f(\boldsymbol{z},\boldsymbol{\theta};[N])\|\leq\sqrt{M'\left(\frac{1}{n}-\frac{1}{N}\right)(p_1+p_2)\log(\frac{1}{\epsilon})},$$

The convexity and compactness assumptions are standard and follows from [144]. In real-world problems, the convexity assumption might prove restrictive as many important problems do not admit convex formulations in both the integer and continuous variables. Nevertheless, the proposed algorithm can also be applied to problems that are non-convex in f and g, but the solution would not carry any theoretical guarantees.

The data concentration assumption is to qualify the general definition of our objective function $f(\boldsymbol{z},\boldsymbol{\theta};[N])$. Specifically, it requires that the objective function (and its derivative) satisfies a Hoeffding-type inequality for sampling without replacement. Note that in particular it requires M and M' to be absolute constants, so the statement is not vacuously true for all possible functions $f(\boldsymbol{z},\boldsymbol{\theta};[N])$. We further observe that for the additively separable objec-

tive functions defined in Equation (4.1), these properties follow immediately from Hoeffding's theorem for the case of sampling without replacement [11]. Note that this assumption in particular does *not* require $f(\boldsymbol{z}; \boldsymbol{\theta}; S^n)$ to be an unbiased estimator of $f(\boldsymbol{z}; \boldsymbol{\theta}; N)$, as long the estimate $f(\boldsymbol{z}; \boldsymbol{\theta}; S^n)$ is sufficiently close. We now introduce a few prominent examples of problems that fall under our setting:

Additively Separable Examples

Examples of such follow the form in Equation (4.1).

- *Machine Learning*: Given data $\boldsymbol{d}_i = (\boldsymbol{x}_i, y_i)$, $i \in [N]$ where $\boldsymbol{x}_i$ are the features and y_i the responses, we aim to learn a particular machine learning model. This includes for example support vector machines, where:

  $$l(\cdot, \cdot, \cdot, \cdot) = \max\{1 - y_i \boldsymbol{\theta}^T \boldsymbol{x}_i, 0\},$$

  least-squares regression:

  $$l(\cdot, \cdot, \cdot, \cdot) = \|y_i - \boldsymbol{\theta}^T \boldsymbol{x}_i\|^2,$$

  and many others.

- *Treatment Effect Estimation*: This includes a wide range of data-driven optimization problems in pricing and policy treatment effect estimation (see e.g. 149, 26, 3) where the loss is constructed from a parametric or semi-parametric loss function and there are further constraints on the parameters (non-negativity, bounds, etc). Many of these problems might involve non-convex loss functions, but they are usually solved by methods that assume convexity (e.g. gradient descent, cutting planes) or create successive convex approximations. For example, given data $\boldsymbol{d}_i = (\boldsymbol{x}_i, y_i)$, and a probit model $\hat{f}(\boldsymbol{x}_i, \boldsymbol{\theta})$ for estimating propensity of purchasing a model, the form of the loss function would appear as:

  $$l(\cdot, \cdot, \cdot, \cdot) = \frac{1}{N} \sum_{i=1}^{N} y_i \log(\hat{f}(\boldsymbol{x}_i, \boldsymbol{\theta})) + (1 - y_i) \log(1 - \hat{f}(\boldsymbol{x}_i, \boldsymbol{\theta})),$$

- *Sample Average Approximation* : Sample Average Approximation (SAA) methods are commonly used in stochastic optimization problems where the master problem:

$$\min_{\boldsymbol{\theta}\in\Theta} \mathbb{E}_{\boldsymbol{W}\sim P}[G(\boldsymbol{\theta}, \boldsymbol{W})]$$

is solved by generating $(\boldsymbol{w}^i)_{i\in[N]}$, a sample of size N (either taken from the distribution or created through historical samples), and solving the sample average approximation:

$$l(\cdot,\cdot,\cdot,\cdot) = \min_{\boldsymbol{\theta}\in\Theta} \frac{1}{N}\sum_{i=1}^{N} G(\boldsymbol{\theta}, \boldsymbol{w}^i).$$

The approximation is often solved many times with different samples to create confidence intervals. It is often assumed that the resulting problem is tractable, but it is not necessarily so when N is large [147]. For a detailed discussion of the sample average approximation for stochastic approximation, please see e.g. [103].

Non-additively Separable Examples

- *Sparse Linear Regression*: W seek at most k out of p factors in the design matrix $\boldsymbol{X} \in \mathbb{R}^{N\times p}$ to explain the response data $\boldsymbol{y} \in \mathbb{R}^N$. The optimization problem can be written as:

$$\min_{\boldsymbol{\beta}:\|\boldsymbol{\beta}\|_0\le k} \quad \frac{1}{N}\left(\|\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta}\|_2^2 + \frac{1}{\gamma}\|\boldsymbol{\beta}\|_2^2\right). \tag{4.3}$$

[24] showed that it can be reformulated as:

$$\min_{\boldsymbol{z}\in\mathcal{Z}} \quad \frac{1}{N}\boldsymbol{y}^T\left(I_N + \gamma\sum_{i=1}^{p} z_i \boldsymbol{X}_i\boldsymbol{X}_i^T\right)^{-1}\boldsymbol{y}, \tag{4.4}$$

where here $\mathcal{Z} = \{z \in \{0,1\}^p \mid \sum_{i=1}^p z_i = k\}$, $\boldsymbol{X}_i$ is the ith column of the design matrix, and γ controls the regularization. Note that this optimization function satisfies Assumptions 3 and 4 as the objective function is convex in $\boldsymbol{z}$, and the set $\mathcal{Z}$ is indeed finite.

Lemma 1. *The function* $f(\boldsymbol{z},\boldsymbol{\theta};[N]) = \frac{1}{N}\boldsymbol{y}^T\left(I_N + \gamma\sum_{i=1}^p z_i\boldsymbol{X}_i\boldsymbol{X}_i^T\right)^{-1}\boldsymbol{y}$ *satisfies As-*

sumption 5.

The proof of the lemma is contained in Appendix C.1.

4.3 Stochastic Cutting Planes and Theoretical Results

In this section, we present a formal description of the cutting plane algorithm, its stochastic cutting plane variant and show the global convergence of the stochastic cutting plane algorithm. The cutting plane algorithm introduced in [57] and generalized in [144] for Problem (4.2) iteratively rotates between solving a nonlinear subproblem and a master mixed-integer linear program (MILP) containing linear approximations of the functions involved. We first define the NLP subproblem as followed:

$$\begin{aligned} \text{NLP}(\boldsymbol{z}, [S_n]) := \min_{\boldsymbol{\theta} \in \Theta} & f(\boldsymbol{z}, \boldsymbol{\theta}; [S_n]) \\ \text{subject to } & g_i(\boldsymbol{z}, \boldsymbol{\theta}) \leq 0, \quad i \in [m] \end{aligned}$$

Then we present the cutting plane algorithm in Algorithm 5. At every iteration, we first check if the current solution $\boldsymbol{z}_t, \boldsymbol{\theta}_t, \eta_t$ is still a feasible solution for the tth iteration, and return $\boldsymbol{z}_t, \boldsymbol{\theta}_t, \eta_t$ (as the optimal solution) if it is feasible. As the cutting plane iteratively adds constraints, if a previous solution $(\boldsymbol{z}_t, \boldsymbol{\theta}_t)$ at the t iteration is also optimal for the $t + a$th iteration for some $a > 0$, then since a cutting plane at $(\boldsymbol{z}_t, \boldsymbol{\theta}_t)$ was added in the $t + 1$th iteration, so we have $\eta_{t+a} \geq \eta_{t+1} \geq f(z_t, \boldsymbol{\theta}_t; [N]) = f(z_{t+a}, \boldsymbol{\theta}_{t+a}; [N])$, and the algorithm terminates. We further note that this guarantees that the algorithm converges in at most $|\mathcal{Z}|$ iterations since if a solution repeats, then the algorithm has to terminate.

In many examples such as risk minimization and sparse regression, the cutting plane algorithm (and its variants) has produced state-of-the-art results. However, one important drawback is its scalability with respect to data. It was noted in [70] that the scalability of the cutting-plane method for Support Vector Machines (SVMs), although significantly better than other exact methods, trails significantly behind stochastic methods such as stochastic gradient descent. This behavior is also observed in matrix completion in [22], where the full cutting plane algorithm scales less favorably than competing algorithms.

Algorithm 5 Cutting-plane algorithm.

procedure CUTPLANES($\{\boldsymbol{d}_i\}_{i=1}^N$)
  $t \leftarrow 1$
  $\boldsymbol{z}_1, \boldsymbol{\theta}_1 \leftarrow$ Initialization ▷ Heuristic Warm Start
  $\eta_1 \leftarrow lb$ ▷ Initialize feasible solution variable with lower bound
  while $\eta_t < f(\boldsymbol{z}_t, \boldsymbol{\theta}_t; [N])$ **or** $\exists j \in [m]\ 0 < g_j(\boldsymbol{z}_t, \boldsymbol{\theta}_t)$ **do**
    ▷ while the current solution is not feasible for the tth iteration

$$\boldsymbol{z}_{t+1}, \eta_{t+1} \leftarrow \underset{\boldsymbol{z}\in\mathcal{Z}, \boldsymbol{\theta}\in\Theta, \eta\geq lb}{\arg\min} \quad \eta$$

$$\text{s.t.} \quad \eta \geq f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; [N]) + \nabla f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; [N])^T \begin{pmatrix} \boldsymbol{z} - \boldsymbol{z}_i \\ \boldsymbol{\theta} - \boldsymbol{\theta}_i \end{pmatrix}, \quad \forall i \in [t]$$

$$0 \geq g_j(\boldsymbol{z}_i, \boldsymbol{\theta}_i) + \nabla g_j(\boldsymbol{z}_i, \boldsymbol{\theta}_i)^T \begin{pmatrix} \boldsymbol{z} - \boldsymbol{z}_i \\ \boldsymbol{\theta} - \boldsymbol{\theta}_i \end{pmatrix}, \quad \forall i \in [t],\ \ \forall j \in [m]$$

    ▷ Solve the master problem
    $\boldsymbol{\theta}_{t+1} \leftarrow \arg_{\boldsymbol{\theta}}\ \text{NLP}(\boldsymbol{z}_{t+1}, [N])$ ▷ Solve the NLP subproblem
    $t \leftarrow t + 1$
  end while
  $\boldsymbol{z}^*, \boldsymbol{\theta}^*, \eta^* \leftarrow \boldsymbol{z}_t, \boldsymbol{\theta}_t, \eta_t$
  return $\eta^*, \boldsymbol{z}^*, \boldsymbol{\theta}^*$ ▷ Return the filled matrix $\boldsymbol{X}$
end procedure

To understand the scalability of cutting plane algorithm, observe that at every iteration we are only solving a MILO problem, which has enjoyed great advances in recent years due to various algorithmic improvements (see [25, 2] for details). As a result, the amount of time spent in the actual solver is usually not excessive. On the contrary, the cutting plane generation for f and g usually accounts for the majority of the running time. In data-driven optimization problems defined in Section 4.2 this is greatly exacerbated by the fact that the function $f(\boldsymbol{z}, \boldsymbol{\theta}; [N])$ and its derivative $\nabla f(\boldsymbol{z}, \boldsymbol{\theta}; [N])$ needs to be evaluated for all samples in $[N]$, which could easily exceed 10^6 samples or higher.

Therefore, with the objective to improve scalability for cutting-plane methods, we propose that instead of evaluating the full function and its derivative, we randomly select n samples without replacement in a set S^n to evaluate the function and its derivative at every iteration of the cutting plane. When $n \ll N$, we obtain a considerable speedup for the generation of the cutting plane. Concretely, we present the stochastic cutting plane algorithm in Algorithm 6.

Algorithm 6 Stochastic Cutting-plane algorithm.

procedure STOCHASTIC CUTTING PLANES($\{\boldsymbol{d}_i\}_{i=1}^N, \epsilon$)
  $t \leftarrow 1$
  $\boldsymbol{z}_1, \boldsymbol{\theta}_1 \leftarrow$ Initialization ▷ Heuristic Warm Start
  $\eta_1 \leftarrow lb$ ▷ Initialize feasible solution variable with lower bound
  $S_1^n \leftarrow$ random n-sized subset of $\{1, \ldots, N\}$ ▷ Initialize random subset of samples
  while $\eta_t < f(\boldsymbol{z}_t, \boldsymbol{\theta}_t; [S_t^n])$ **or** $\exists j \in [m]\ 0 < g_j(\boldsymbol{z}_t, \boldsymbol{\theta}_t)$ **do**
    ▷ while the current solution is not feasible for the tth iteration

$$\boldsymbol{z}_{t+1}, \eta_{t+1} \leftarrow \underset{\boldsymbol{z}\in\mathcal{Z}, \boldsymbol{\theta}\in\Theta, \eta\geq lb}{\arg\min} \quad \eta$$

$$\text{s.t.} \quad \eta \geq f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; S_i^n) + \nabla f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; S_i^n)^T \begin{pmatrix} \boldsymbol{z} - \boldsymbol{z}_i \\ \boldsymbol{\theta} - \boldsymbol{\theta}_i \end{pmatrix}, \quad \forall i \in [t]$$

$$0 \geq g_j(\boldsymbol{z}_i, \boldsymbol{\theta}_i) + \nabla g_j(\boldsymbol{z}_i, \boldsymbol{\theta}_i)^T \begin{pmatrix} \boldsymbol{z} - \boldsymbol{z}_i \\ \boldsymbol{\theta} - \boldsymbol{\theta}_i \end{pmatrix}, \quad \forall i \in [t],\ \ \forall j \in [m]$$

    ▷ Solve the master problem
    $S_{t+1}^n \leftarrow$ random n-sized subset of $\{1, \ldots, N\}$
    $\boldsymbol{\theta}_{t+1} \leftarrow \arg_{\boldsymbol{\theta}}$ NLP($\boldsymbol{z}_{t+1}, [S_{t+1}^n]$) ▷ Solve the NLP subproblem
    $t \leftarrow t + 1$
  end while
  $\boldsymbol{z}^*, \boldsymbol{\theta}^*, \eta^* \leftarrow \boldsymbol{z}_t, \boldsymbol{\theta}_t, \eta_t$
  return $\eta^*, \boldsymbol{z}^*, \boldsymbol{\theta}^*$ ▷ Return the filled matrix $\boldsymbol{X}$
end procedure

4.3.1 Theoretical Guarantees

[144] proved that the cutting plane algorithm is able to converge to an optimal solution in finite steps. In this section, we show that the stochastic cutting plane algorithm, similarly, is able to converge to an ϵ-optimal solution in finite steps with high probability. This is presented in the following theorem:

Theorem 1. *Let $\boldsymbol{d}_1, \ldots, \boldsymbol{d}_n$ be given data, and assume that Problem (4.2) is feasible and has an optimal solution $\boldsymbol{z}^*, \boldsymbol{\theta}^*$. Under Assumptions 3-5, Algorithm 6 converges after a finite number of iterations K with probability one, and $\mathbb{E}[K] \leq (e-1)|\mathcal{Z}| + 1$. When it terminates, it produces a feasible solution $\tilde{\boldsymbol{z}}^*, \tilde{\boldsymbol{\theta}}^*$ that satisfies:*

$$f(\tilde{\boldsymbol{z}}^*, \tilde{\boldsymbol{\theta}}^*; [N]) \leq f(\boldsymbol{z}^*, \boldsymbol{\theta}^*; [N]) + \epsilon \tag{4.5}$$

with probability at least $1 - (2K+1)\exp\left(\frac{-\epsilon^2}{(2+\sqrt{p_1+p_2})^2(1/n-1/N)\max(M,M')}\right)$, where M, M' are

as such defined in Asssumption 5.

The proof of Theorem 1 is contained in Appendix C.2. There are two features of interest in this theorem. The first is that the proof of worst case bound for the expected number of cutting planes utilizes an argument based on the pigeonhole principle and is completely independent from the sampling size n. This means that the number of cutting plane iterations (in the worst case) do not necessarily increase with smaller sample size n. Although the worst-case bound is high ($(e-1)|\mathcal{Z}|+1$), this bound is only a small constant factor away from the worst case for the deterministic cutting plane algorithm ($|\mathcal{Z}|$) as shown in [57]. Furthermore, in practice, the number of cutting planes generated by the algorithm does not exhibit the exponential trend as demonstrated in the worst-case analysis here. we show in Section 4.4 that experimentally the stochastic cutting planes is extremely fast, giving order-of magnitude speedups compared to the standard cutting plane algorithm.

We also see that the probability of the stochastic cutting plane algorithm failing to return an ϵ-optimal solution decreases exponentially as $n \to N$, while it increases exponentially in the constant for the Hoeffding-type inequality in Assumption 5, suggesting that data concentration is a key requirement for the stochastic cutting plane to perform well. Although the number of cutting planes K could be as high as $K \sim \exp(p_1)$, in reality we see much fewer cutting planes generated by the cutting-plane algorithm [144]. Furthermore, in most application settings, we have $p_1 \ll n \ll N$, so even with the worst-case scenario of $K \sim \exp(p_1)$, the failure probability is still on the order of $\exp\left(p_1 - O(n\epsilon^2)\right) \ll 1$ for reasonable ϵ sizes.

4.4 Computational Results

In this section, we present computational results from a variety of examples that demonstrate the versatility, scalability and accuracy of the stochastic cutting plane algorithm 6.

4.4.1 Sparse Regression

In this section, we apply Algorithm 6 to the sparse linear regression problem as introduced in Section 4.2 , where we seek at most k out of p factors in the design matrix $\boldsymbol{X} \in \mathbb{R}^{N\times p}$ to explain the response data $\boldsymbol{y} \in \mathbb{R}^N$. The optimization problem can be written as:

$$\min_{\boldsymbol{z}\in\mathcal{Z}} \quad \frac{1}{N}\boldsymbol{y}^T \left(I_N + \gamma \sum_{i=1}^{p} z_i \boldsymbol{X}_i \boldsymbol{X}_i^T \right)^{-1} \boldsymbol{y}, \tag{4.6}$$

where here $\mathcal{Z} = \{z \in \{0,1\}^p \mid \sum_{i=1}^p z_i = k\}$, $\boldsymbol{X}_i$ is the ith column of the design matrix, and γ controls the regularization. We explore the scalability and accuracy of Algorithm 6.

To study the effectiveness of Algorithm 6 we generate data according to $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$ where the entries of the design matrix $X_{ij} \sim N(0,1)$ independently. A random set S of size k is chosen from $\{1,\ldots,p\}$, and the coefficients are chosen so that $\boldsymbol{\beta}_j = 0, \quad \forall j \notin S$, and $\boldsymbol{\beta}_j \sim N(0,1), \quad \forall j \in S$ independently. This ensures the exact sparsity of rank k for the resulting model. The noise is sampled independently with $\epsilon_i \sim N(0,\sigma)$.

For each parameter combination (N,p,k,σ), we record the time T and the Mean Average Percentage Error (MAPE) of both the standard cutting-plane algorithm (Algorithm 5) and the stochastic cutting plane algorithm (SCP, Algorithm 6). The γ parameter for both algorithms are selected through cross-validation on a validation set of size N using the same setup as above. We set the tolerance in the cutting plane algorithm to $\epsilon = 10^{-4}$. The testing set is of size N sampled using the true β and independent X_{ij} and ϵ_i with the same setup as above. Table 4.1 shows the mean results over 10 random generations of the dataset of the size (N,p,k,σ), separated into 4 blocks where each variable is varied over a range of possibilities. For these experiments, we used $n = \min\{N, 10\sqrt{N}\}$ as the sampling size for stochastic cutting planes. In Section 4.4.4, we detail further experiments on varying the sample size. The testing environment has a six-core i7-5820k CPU with 24GB of RAM. Both algorithms are implemented in pure Julia with Julia v1.4.2.

We see that as N grows, the advantage of stochastic cutting planes in using $n = O(\sqrt{N})$ samples to compute the cutting plane quickly increases. At the largest scale with $N = 10^6$, stochastic cutting planes generates over a 30× speedup compared to the standard cutting

	N	p	k	σ	Cutting Planes		SCP	
					T	MAPE	T	MAPE
N	10^3	10^2	10	0.1	2.4s	4.4%	1.9s	4.4%
	10^4	10^2	10	0.1	5.7s	3.9%	3.7s	3.9%
	10^5	10^2	10	0.1	170s	4.5%	7.8s	4.5%
	10^6	10^2	10	0.1	1356s	3.1%	46s	3.1%
p	10^5	10^2	10	0.1	170s	4.5%	7.8s	4.5%
	10^5	10^3	10	0.1	2073s	4.1%	98.5s	4.1%
	10^5	10^4	10	0.1	29770s	4.7%	945s	4.7%
k	10^5	10^2	10	0.1	170s	4.5%	7.8s	4.5%
	10^5	10^2	20	0.1	1621s	2.9%	41.9s	2.9%
	10^5	10^2	50	0.1	30608s	1.8%	937s	1.8%
σ	10^5	10^2	10	0.1	170s	4.5%	7.8s	4.5%
	10^5	10^2	10	0.2	197s	6.7%	8.9s	6.7%
	10^5	10^2	10	0.3	231s	9.8%	10.7s	9.8%

Table 4.1: Comparison of cutting planes and stochastic cutting planes on synthetic data for sparse regression.

plane algorithm. Furthermore, we see that the stochastic cutting plane converges to the exact same solution as the standard cutting planes, as reflected the identical MAPE figures. This is a reflection of the exponentially vanishing failure probability as indicated in Theorem 1. Such order of magnitude speedup is also preserved when we vary p , sparsity k, or the amount of noise σ in the additional blocks of experiments without any loss of accuracy.

4.4.2 Support Vector Machines

In this section, we apply Algorithm 6 to the problem of support vector machines for classification. In this particular case, the variables are all continuous, and therefore we focus on utilizing the cutting-plane algorithm to solve the NLP subproblem as defined in Algorithm 6 since the set of integer variables $\mathcal{Z} = \emptyset$.

Specifically, we consider a dataset where we have $(\boldsymbol{x}_i, y_i)$, $i \in [N]$ and $y_i \in \{-1, +1\}$, and we wish to create a classifier of the form $f(\boldsymbol{x}) = \text{Sign}(\boldsymbol{\theta}^T \boldsymbol{x})$. Then the SVM problem with

ℓ_2 regularization can be written as:

$$\min_{\boldsymbol{\theta},\epsilon\geq 0} \frac{1}{2}\|\boldsymbol{\theta}\|^2 + CR(\boldsymbol{\theta}), \tag{4.7}$$

where C is a user-defined constant controlling the degree of regularization, and:

$$R(\boldsymbol{\theta}) = \max_{\boldsymbol{c}\in\{0,1\}^N} \left\{ \frac{1}{N}\sum_{i=1}^{N} c_i - \frac{1}{N}\sum_{i=1}^{N} c_i y_i(\boldsymbol{\theta}^T \boldsymbol{x}_i) \right\}. \tag{4.8}$$

In this setting all variables are continuous. We follow the formulation in classical papers of utilizing cutting planes in SVMs (e.g. [70, 97]) by using cutting planes (i.e. linearizing the objective) only on the $R(\boldsymbol{\theta})$ portion of the objective. $R(\boldsymbol{\theta})$ is the maximum of a function that is additively separable in the data, so it satisfies Assumption 5.

For any $\boldsymbol{\theta}$, the following expression is a valid subgradient for $R(\boldsymbol{\theta})$:

$$\partial R(\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N} c_i y_i \boldsymbol{x}_i, \tag{4.9}$$

with

$$c_i = \begin{cases} 1, & y_i(\boldsymbol{\theta}^T \boldsymbol{x}_i) < 1, \\ 0, & \text{otherwise.} \end{cases}.$$

To conduct the experiment, we utilize the forest covertype dataset from the UCI Machine Learning Repository [55] that has 580,012 samples with 54 features, and the objective is to predict the type of forest cover. We binarize the objective variable to predict if the forest cover belongs to Class 2 (the most frequent class at 48.9%) or not, following a similar treatment in [97].

We fix $C = 10^6$ as defined in Equation (4.7), and set the tolerance in the cutting plane to $\epsilon = 10^{-4}$. Then, we randomly sample two disjoint datasets, each of size N, for training and testing from the master dataset. We experiment with different sizes of N to observe the speedup and performance of stochastic cutting planes (SCP) over standard cutting planes. We again set $n = \min\{N, 10\sqrt{N}\}$ as the sampling size for stochastic cutting planes. The results, recording time T and the out-of-sample accuracy on the testing set ACC over 10

random runs are contained in Table 4.2. We further compare to LIBSVM [35], an established reference SVM implementation using Sequential Minimal Optimization (SMO), and stochastic gradient descent (SGD), which is widely regarded as the fastest algorithm for solving SVM problems (at the expense of some accuracy). The testing environment has a six-core i7-5820k CPU with 24GB of RAM. Except for SGD, all algorithms are implemented in pure Julia with Julia v1.4.2 to minimize the difference in conditions. For SGD, we utilize the optimized formulation contained in scikit-learn V0.22.2 and Python V3.7.4.

N	p	**Cutting Planes**		**SCP**		**LIBSVM**		**SGD**	
		T	**ACC**	T	**ACC**	T	**ACC**	T	**ACC**
10^3	54	107.6s	74.9%	7.5s	72.3%	0.75s	74.2%	0.01s	71.8%
10^4	54	145.7s	75.5%	10.7s	74.8%	30.9s	75.0%	0.09s	73.4%
10^5	54	439.2s	75.9%	32.1s	75.3%	960.5s	75.7%	0.45s	74.3%

Table 4.2: Comparison of cutting planes and stochastic cutting planes on the covertype dataset for support vector machines.

We see that stochastic cutting planes achieve a significant order-of-magnitude speedup compared to the standard cutting plane while maintaining a comparable accuracy with minimum degradation. At the largest scales it is over 13× faster than the standard cutting planes formulation and 30× faster than LIBSVM with only a degradation of $\sim 0.5\%$ in out-of-sample accuracy. This reflects the scalability of the stochastic cutting plane method versus other methods in large-scale data. Stochastic gradient descent is still significantly faster, but at the further expense of accuracy of about 1%. We note that however, the running time of SCP appears to grow slower (roughly proportional to $\sqrt{n}$) compared to SGD (roughly proportional to n).

We note that at the largest scale with $N = 10^5$, each stochastic cutting plane uses $n = 10^{3.5}$ samples, and yet it manages to be 3× faster than the full cutting plane algorithm when ran on 10^3 samples while its accuracy is closer to the cutting plane algorithm when ran on 10^4 samples. This shows that the randomness of the data samples selected by each cutting plane is key, and the performance of the stochastic cutting plane using a random

sample of size n cannot be simply replicated by choosing n samples and running the standard cutting plane algorithm.

4.4.3 The Stochastic Knapsack Problem

We further apply stochastic cutting planes to SAA versions of stochastic optimization problems. In particular, we test stochastic cutting planes on the classic *static stochastic knapsack problem* (SSKP). A SSKP is a resource allocation problem where a decision maker has to choose a subset of k known alternative projects to take on. For this purpose, a known quantity q of relatively low-cost resource is available, while any further resource required can be obtained at a known cost of c per unit of resource. The amount W_i of resource required by project i is not known exactly but the decision maker has an estimate of the probability distribution. Each project i has an expected net reward of r_i. Thus, the optimization problem can be formulated as:

$$\max_{\boldsymbol{z}\in\{0,1\}^k} \sum_{i=1}^{k} r_i z_i - c\mathbb{E}\left[\sum_{i=1}^{k} W_i z_i - q\right]^+$$

Where $[\cdot]^+ := \max(\cdot, 0)$ and the expectation is taken over the probability distribution of $\boldsymbol{W} = [W_1, \ldots, W_k]$. This objective function appears in many real-life problems such as airline crew scheduling, shortest path problems, and others, often with further constraints on $\boldsymbol{z}$ to limit the optimization over $\boldsymbol{z} \in \mathcal{Z} \subset \{0,1\}^k$.

We here consider the SAA version of SSKP where N realizations of $\boldsymbol{W}$ are available (either by simulation or historical data) in the form of $[\boldsymbol{W}^1, \ldots, \boldsymbol{W}^N]$. Then the SAA version of SSKP can be written as:

$$\max_{\boldsymbol{z}\in\{0,1\}^k} \sum_{i=1}^{k} r_i z_i - \frac{c}{N}\sum_{j=1}^{N}\left[\sum_{i=1}^{k} W_i^j z_i - q\right]^+ \tag{4.10}$$

This problem is an integer concave optimization problem, and we can utilize cutting planes on the cost part of the objective:

$$C(\boldsymbol{z}) = \frac{c}{N}\sum_{j=1}^{N}\left[\sum_{i=1}^{k} W_i^j z_i - q\right]^+ .$$

The objective function is additively separable in the data, so it satisfies Assumption 5 directly. Its derivative is easily calculated as:

$$(\nabla C(\boldsymbol{z}))_i = \frac{c}{N}\sum_{j=1}^{N} W_i^j \mathbb{1}\left\{\sum_{i=1}^{k} W_i^j z_i - q \geq 0\right\}.$$

To generate the specific instances for this problem, we follow the setup in [103]. The rewards follow $r_i \sim \text{Unif}[10, 20]$, while $W_i \sim N(\mu_i, \sigma_i)$ where $\mu_i \sim \text{Unif}[20, 30]$ and $\sigma_i \in \text{Unif}[5, 15]$. For all instances, we set the per unit penalty $c = 4$, and set the initial available inventory $q = \max(k, 20)$ (the lower bound is so that the optimal solution is not the trivial 0 vector). We investigate different combinations of (N, k) to uncover how the performance of stochastic cutting plane scales with problem size and samples generated.

We compare stochastic cutting planes (with sampling size $n = \min\{N, 10\sqrt{N}\}$) to the standard cutting plane algorithm, and further with the standard linear reformulation of (4.10) using auxiliary variables $\boldsymbol{x} \in \mathbb{R}^n$ as outlined in [103]:

$$\max_{\boldsymbol{z}\in\{0,1\}^k, \boldsymbol{x}\geq 0} \sum_{i=1}^{k} r_i z_i - \frac{c}{N}\sum_{j=1}^{N} z_j \tag{4.11}$$

$$s.t. \quad x_j \geq \sum_{i=1}^{k} W_i^j z_i - q, \;\; \forall j \in [N]. \tag{4.12}$$

The testing environment has a six-core i7-5820k CPU with 24GB of RAM. All algorithms are implemented in pure Julia with Julia v1.4.2 to minimize the difference in conditions, and we utilize Gurobi V9.0.1 as the integer programming solver. We record two statistics: the time taken to solve the formulation (T), and the objective value (Obj). For ease of comparison, we normalized the objective value of the optimal solution to 100%. Each statistic is shown as an average of 20 randomly generated instances:

N	k	Cutting Planes		SCP		Linear Reformulation	
		T	Obj	T	Obj	T	Obj
10^3	10	0.33s	100%	0.24s	99.7%	7.0s	100%
10^4	10	0.89s	100%	0.27s	99.9%	70.3s	100%
10^5	10	3.6s	100%	0.35s	100%	3760s	100%
10^6	10	61.3s	100%	0.97s	100%	N/A	N/A
10^3	20	0.42s	100%	0.25s	99.9%	8.5s	100%
10^4	20	1.1s	100%	0.39s	99.9%	437s	100%
10^5	20	9.3s	100%	0.46s	100%	7328s	100%
10^6	20	124s	100%	1.7s	100%	N/A	N/A
10^3	50	1.3s	100%	1.0s	100%	6.1s	100%
10^4	50	5.3s	100%	1.1s	100%	260s	100%
10^5	50	51.8s	100%	2.5s	100%	4005s	100%
10^6	50	706s	100%	6.9s	100%	N/A	N/A

Table 4.3: Comparison of cutting planes and stochastic cutting planes for SSKP. N/A means that no instance completed running under 10000s.

We first observe that both the standard and the stochastic cutting plane formulation outperform the linearized reformulation significantly in time needed. This is not surprising as the linear reformulation requires N auxiliary variables to model the $[\cdot]^+$ function, which severely impacts solve time.

We further notice that stochastic cutting planes is able to achieve a near optimal solution across all combinations of N and k tested. In the largest instances (where $N = 10^6$), it was able to almost always return the same solution as the general cutting plane algorithm while spending two orders of magnitude less time. This again demonstrates the significant speedup that stochastic cutting plane provides.

4.4.4 Sample Size Selection

The key parameter in the method of stochastic cutting planes is the sample size n, as it drives both the accuracy in Theorem 1, and the speedup compared to the full cutting plane algorithm. Therefore, in this section, we explore how the sample size n affects the quality of solutions in various problems.

Specifically, we test different combinations of (N, n) on sparse regression with $p = 100$, $k = 10$ using the same setup as in Section 4.4.1 over 10 runs each, and record the percentage of cases that the stochastic cutting plane algorithm returned the same solution as the optimal solution. From Figure 4-1 we can see that for the task of sparse regression, the sample size n

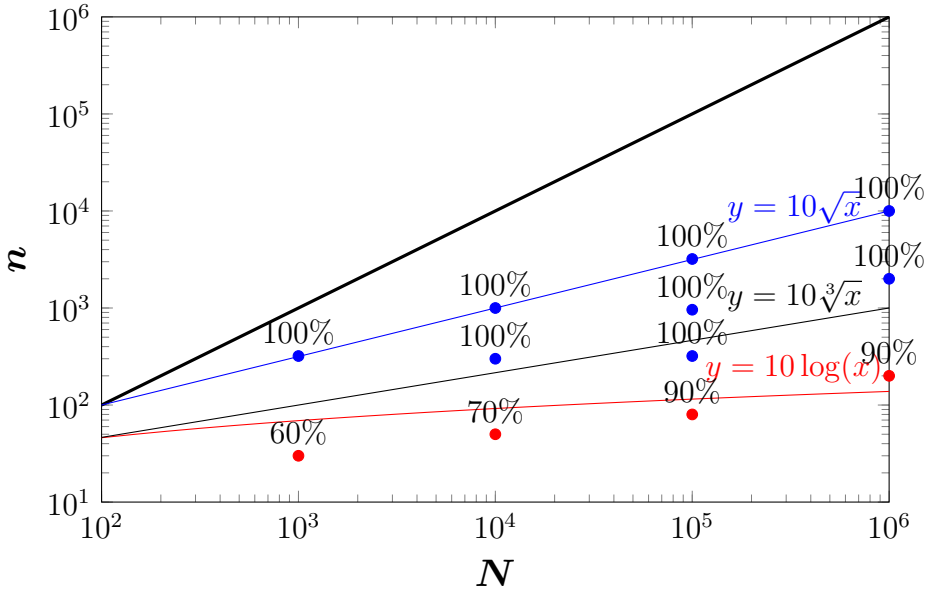


Figure 4-1: Percentage of Runs where the Stochastic Cutting Planes agrees with true solution on different (N, n) combinations for sparse regression with $p = 100$ and $k = 10$.

needed so that the stochastic solution always coincides with the true solution seems to grow slower than $O(\sqrt{N})$ but faster than $O(\log(N))$. Therefore, in cases where accuracy of the solution is important, it would seem to be prudent, at least for the task of sparse regression, to select $O(\sqrt{N})$ samples to ensure that the solution does indeed coincide, as we showed in Table 4.1. However, in scenarios where accuracy is less important than scalability, then the practitioner can explore even smaller n such as $O(\sqrt[3]{N})$ as it is clear from the graph that $O(\sqrt{N})$ is a conservative bound as $N \to \infty$.

We further explore this sample size relationship in the context of the SSKP problem. We follow the same setup as in Section 4.4.3, and set $k = 50$. We record the objective value of SCP as a percentage of the optimal solution, averaged over 20 runs.

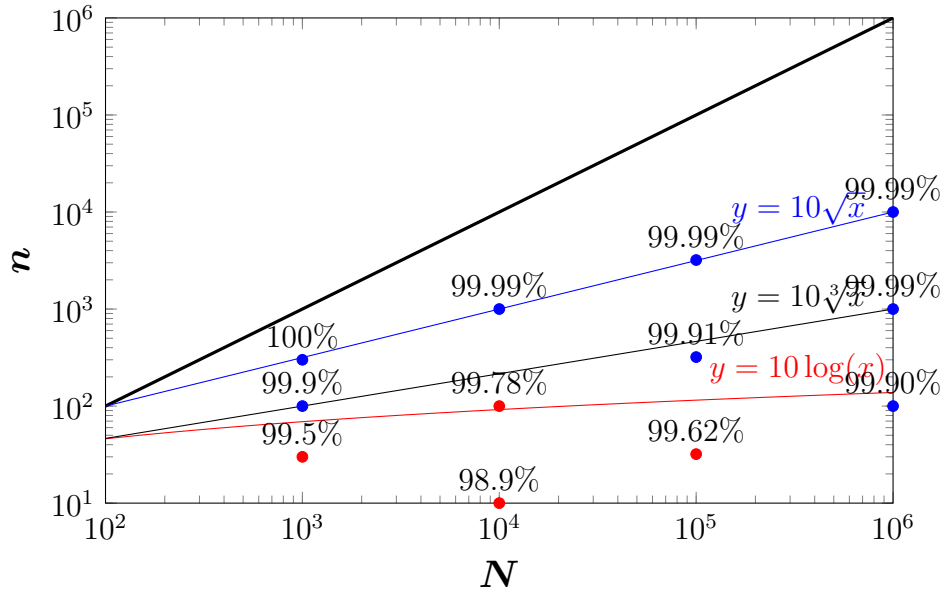


Figure 4-2: Average performance of the solution reached by Stochastic Cutting Planes as a percentage of true optimal objective on different (N, n) combinations for SSKP with $k = 50$. Nodes with $< 99.9\%$ of optimality are colored red.

Similar to sparse regression, we see from Figure 4-2 that the chosen sample size $n = 10\sqrt{N}$ is a quite conservative and the average objective across 20 runs is always within 0.01% of the optimal solution. We further see that even at scales of $n = 10\log(N)$, the objective value of the solution reached by SCP is always within 0.5% of the true solution. Therefore, in problems where absolute optimality is not necessary, further running time gains (compared to what is demonstrated in this work) could be achieved by using a smaller sample size.

4.5 Conclusions

We have presented stochastic cutting planes for data-driven convex MINLPs. Our theoretical results show that the proposed methodology is able to obtain the optimal solution with exponentially vanishing failure probability. Our numerical experiments demonstrate that on a wide variety of problems, the stochastic cutting planes algorithm is able to achieve an

order-of-magnitude speedup compared to the standard cutting planes algorithm without a significant sacrifice in accuracy, and is competitive with state-of-the-art algorithms.

Chapter 5

Forecasting COVID-19 and Analyzing the Effect of Government Interventions

5.1 Introduction

The ongoing COVID-19 pandemic is the deadliest in recent history. As of November 12th, 2021, there were over 250 million confirmed cases of COVID-19 and 5.1 million deaths. In late March 2020, we developed DELPHI, a new epidemiological model that aims to predict the pandemic's evolution. DELPHI extends a classical SEIR model [99] to include additional outcomes, such as deaths, account for underdetection of infections, and estimate the effect of changing government interventions. Since its inception, DELPHI has been one of the top 4 models consistently incorporated into the US Centers for Disease Control and Prevention's (CDC) core ensemble forecast [51] and has been utilized by various health and federal agencies including the Federal Reserve for pandemic planning. DELPHI was used by Janssen Pharmaceuticals to select the locations of its multicenter Phase III trial for its single-dose vaccine candidate Ad26.Cov2.S and by Hartford Healthcare, a major hospital system in the United States, to plan intensive care unit capacity.

A key strength of DELPHI is its explicit, flexible, and parametric modeling of government interventions. During the COVID-19 pandemic, governments around the world enacted wide-ranging non-pharmaceutical interventions (NPIs), including such as social distancing, school closures, and lockdowns, at different stages of their local epidemics. This variation

helps DELPHI to identify the effects of different interventions, which, in turn, allows for predictions of alternative scenarios and can inform policy-making.

We demonstrate two applications of DELPHI that leverage our modeling of government interventions. First, through parameter calibration to time series of observed cases and deaths in various countries, DELPHI can estimate the effect of different NPIs accounting for country-specific baseline infection and case fatality rates. In Section 5.4.1, we estimate that school closings and mass gathering restrictions were among the most effective measures in reducing the rate of infection during the early stages of the pandemic. Although these policies incur a significant social burden, they can be effective in controlling the extreme growth in infections when other preventative measures (e.g. masks) are not widely implementable and while treatment options are being evaluated and developed. Had these restrictions been implemented just one week earlier, most – up to 90%– of the deaths in the early stages of the pandemic could have been avoided.

Another major application of DELPHI is the assessment of alternative scenarios to inform policy making. By utilizing the estimated effect of different NPIs, we can create a scenario analysis toolkit to simulate the pandemic forward under different government policies. In Section 5.4.2, we illustrate Janssen Pharmaceuticals' (a Johnson & Johnson company) use of this scenario analysis toolkit in mid-to-late 2020 to identify countries with predicted high COVID-19 incidence as candidate sites for their multicenter Phase III randomized trial of their leading vaccine candidate Ad26.Cov.S.

DELPHI has been applied to 167 geographic areas (countries/provinces/states) worldwide as of end of April 2020, and more than 215 as of end of September 2020, covering all 6 populated continents. Its results have also been available since early April 2020 on `www.covidanalytics.io`. In this paper, we document the calibration, quantitative results, and insights obtained from the DELPHI model during the pre-vaccination era of the COVID-19 epidemic and illustrate two key applications.

5.1.1 Literature

Many epidemiological models were developed to describe the evolution of the COVID-19 epidemic. Most are mechanistic and represent some variation of the classical Susceptible-

Exposed-Infectious-Recovered (SEIR) compartmental model [99], that partitions a population into mutually exclusive and exhaustive compartments and describes infection dynamics with differential equations. Some of these models are marginal, in that they do not have different compartments for different age, sex, or occupation strata [82] while others account for population substructure [165]. A subset of models parameterize the force of the infection as a function of predictors such as proxies of behavior (e.g., cell phone-derived mobility data, credit card spending data) or governmental policies [178, 43]. Non-mechanistic models use machine learning [150] or statistical time-series modeling to forecast the evolution of outcomes [128]. For a comprehensive review of COVID-19 models, see [51].

DELPHI is a mechanistic compartmental model. For each country or state, it describes government policies as composites of elemental NPIs, including lockdown, school closures, gathering size restrictions, and restrictions on non-essential businesses. DELPHI parameterizes the net impact of government policies on the activity of the epidemic as a non-linear function of the effects of the elemental NPIs, and estimates the latter by fitting them to time series of COVID-19 cases and deaths. DELPHI also accounts for the improvement in the management of COVID-19 patients over time by modeling the mortality of the disease as a function of time, and allows for different dynamics for those who recover versus those who die from the disease. It is one of the models most consistently included in the CDC ensemble forecast (top 4 out of 30 submitted models) [146], and its favorable performance is demonstrated in Section 5.3.2.

We believe that the flexibility of the aforementioned modeling choices largely belies the successful projections of the epidemic's trajectory in over 200 countries on all 6 populated continents. Further, by estimating the effects of elemental NPIs we can inform policy making (Section 5.4.1), estimate the impact of delays in deploying early interventions (Section 5.4.1), and inform the design of clinical trials (Section 5.4.2).

5.2 The DELPHI Model

DELPHI is a closed system, in that it does not include demography (births, non-COVID-19 deaths) or migration. It partitions the population in 11 mutually exclusive and exhaustive

compartments (Figure 5-1a):

- **Susceptible (S)**: People who are susceptible to SARS-CoV-2 infection.
- **Exposed (E)**: Early-infected persons who are not yet contagious and are in the incubation period.
- **Infected (I)**: People currently infected and contagious.
- **Undetected (U_R) & (U_D)**: Infected people who are not detected (and are not counted among the known cases) because they did not get tested. It is assumed that these compartments do not contribute to the infection process because the corresponding persons develop symptoms and are then self-quarantined. Some will die (U_D) with marginal probability of death $\tilde{\mu}(t)$ and the rest will recover.
- **Detected, Hospitalized (DH_R) & (DH_D)**: People who are infected, confirmed, and have severe-enough disease to be hospitalized (and effectively quarantined and not contributing to the infectious process). Some will die (DH_D) with marginal probability of death $\tilde{\mu}(t)$, and the rest will recover (DH_R).
- **Detected, Quarantined (DQ_R) & (DQ_D)**: People who are infected, detected through testing, and home-quarantined so that they do not infect others. Some of these people will die (DQ_D) with marginal probability of death $\tilde{\mu}(t)$, and the rest will recover (DQ_R).
- **Recovered (R)**: People who have recovered from the disease and are assumed to have acquired natural immunity. This immunity does not wane for the time horizon of the model.
- **Deceased (D)**: People who have died from COVID-19.

In a given area (state/country), the full mathematical formulation of the model is as follows:

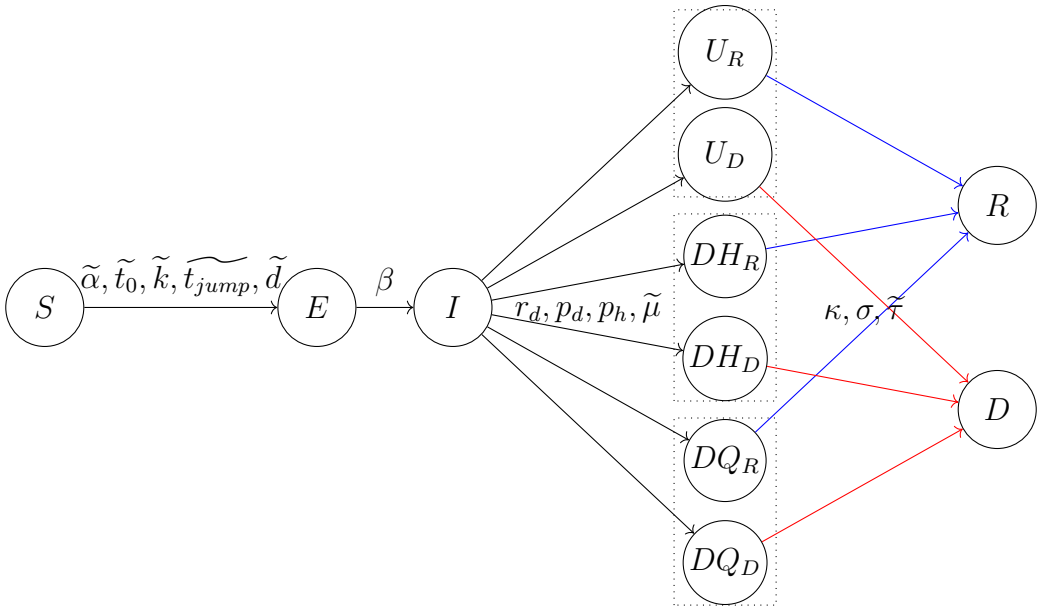
$$\begin{aligned}
\frac{\mathrm{d}S}{\mathrm{d}t} &= -\widetilde{\alpha}\gamma(t)S(t)I(t) \\
\frac{\mathrm{d}E}{\mathrm{d}t} &= \widetilde{\alpha}\gamma(t)S(t)I(t) - \beta E(t) \\
\frac{\mathrm{d}I}{\mathrm{d}t} &= \beta E(t) - r_d I(t)
\end{aligned}$$

$$
\begin{aligned}
\frac{\mathrm{d}U_R}{\mathrm{d}t} &= r_d(1-\widetilde{\mu}(t))(1-p_d)I(t) - \sigma U_R(t) \\
\frac{\mathrm{d}DH_R}{\mathrm{d}t} &= r_d(1-\widetilde{\mu}(t))p_d p_h I(t) - \kappa DH_R(t) \\
\frac{\mathrm{d}DQ_R}{\mathrm{d}t} &= r_d(1-\widetilde{\mu}(t))p_d(1-p_h)I(t) - \sigma DQ_R(t) \\
\frac{\mathrm{d}U_D}{\mathrm{d}t} &= r_d\widetilde{\mu}(t)(1-p_d)I(t) - \widetilde{\tau}U_D(t) \\
\frac{\mathrm{d}DH_D}{\mathrm{d}t} &= r_d\widetilde{\mu}(t)p_d p_h I(t) - \widetilde{\tau}DH_D(t) \\
\frac{\mathrm{d}DQ_D}{\mathrm{d}t} &= r_d\widetilde{\mu}(t)p_d(1-p_h)I(t) - \widetilde{\tau}DQ_D(t) \\
\frac{\mathrm{d}TH}{\mathrm{d}t} &= r_d p_d p_h I(t) \\
\frac{\mathrm{d}DD}{\mathrm{d}t} &= \widetilde{\tau}(DH_D(t) + DQ_D(t)) \\
\frac{\mathrm{d}DT}{\mathrm{d}t} &= r_d p_d I(t) \\
\frac{\mathrm{d}R}{\mathrm{d}t} &= \sigma(U_R(t) + DQ_R(t)) + \kappa DH_R(t) \\
\frac{\mathrm{d}D}{\mathrm{d}t} &= \widetilde{\tau}(U_D(t) + DQ_D(t) + DH_D(t)).
\end{aligned}
$$

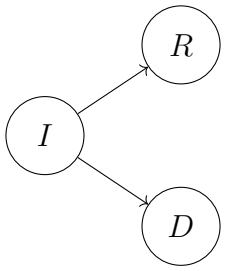
At the beginning of the epidemic, the system above starts at $t = 0$ with the initial condition $S(0) = N - \widetilde{k_1} - \widetilde{k_2}, E(0) = \widetilde{k_1}, I(0) = \widetilde{k_2}$, and all other compartments starting at 0, with $\widetilde{k_1}, \widetilde{k_2}$ country specific fitted parameters. As described later in the current section, the model is restarted after an epidemic wave has been observed. For each wave, we reset $t \to 0$, and use as a starting condition the distribution of the compartments that was reached in the previous period for all compartments except S, E, I. The starting values of the latter are refit using $\widetilde{k_1}$ and $\widetilde{k_2}$:

$$
S(0) = N - \widetilde{k_1} - \widetilde{k_2} - R(0) - \sum_{i\in\{R,D\}} DH_i(0) + DQ_i(0) + U_i(0), \quad E(0) = \widetilde{k_1}, \quad I(0) = \widetilde{k_2}.
$$

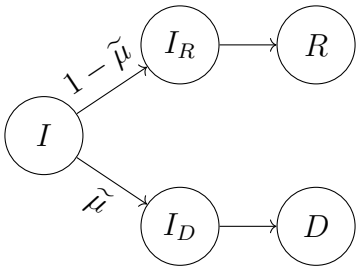
In Figure 5-1a, nodes represent compartments and arrows the allowable transitions. The variables that govern the dynamics are listed in proximity to the associated transitions. Variables with a tilde ($\widetilde{\ }$) are fitted to the time series of known cases and deaths in each area (country/state/province). The remaining are fixed to global values informed from a



(a) Flow Diagram of DELPHI.



(b) Standard Modeling for Recoveries and Deaths.



(c) Updated Modeling for Recoveries and Deaths.

Figure 5-1: The DELPHI Model.

literature review of 174 papers which was current when DELPHI was developed [17]:

- $\widetilde{\alpha}$ is the baseline infection rate.

- $\gamma(t)$ measures the effect of government response and is defined as:

$$\gamma(t) = 1 + \frac{2}{\pi} \arctan\left(\frac{-(t-\widetilde{t_0})}{\widetilde{k}}\right) + \widetilde{c} \exp\left(-\frac{(t-\widetilde{t_{\text{jump}}})^2}{2\widetilde{d}^2}\right), \tag{5.1}$$

  where the parameters $\widetilde{t_0}$ and $\widetilde{k}$ capture, respectively, the timing and the strength of the response. This function is re-fit to data when the model is restarted (e.g., with the emergence of a new wave, as described later). The exponential term intends to reflect a resurgence in infections due to relaxation of governmental policy and societal response;

$\widetilde{c}$ controls the magnitude of the resurgence, $\widetilde{t_{\text{jump}}}$ the time when the resurgence peaks, and $\widetilde{d}$ the duration of the resurgence phase. The effective infection rate in the model is $\widetilde{\alpha}\gamma(t)$, which is time dependent. The exponential resurgence term was added to the model in late June 2020 when we observed the first large resurgence in the pandemic (before July the model assumed $\tilde{c} = 0$). The $\frac{2}{\pi}$ constant normalizes the arctan function. For example, before July 2020, during the first wave, $\tilde{c} = 0$ (no resurgence), $\gamma(t)$ has range $[0, 2]$, and $\gamma(t) = 1$ when $t = t_0$.

- r_d is the detection rate. This equals to $\frac{\log 2}{T_d}$, where T_d is the median time to detection (fixed to be 2 days), see [172].
- β is the rate of infection leaving incubation phase. This equals to $\frac{\log 2}{T_\beta}$, where T_β is the median time to leave incubation (fixed at 5 days), see [110].
- σ is the rate of recovery of non-hospitalized patients. This equals to $\frac{\log 2}{T_\sigma}$, where T_σ is the median time to recovery of non-hospitalized patients (fixed at 10 days), see [87, 104].
- κ is the rate of recovery under hospitalization. This equals to $\frac{\log 2}{T_\kappa}$, where T_κ is the median time to recovery under hospitalization (fixed at 15 days), see [115, 80].
- $\widetilde{\tau}$ is the death rate, the reciprocal of the average time it takes for patients to move from the $U_D, DH_D,$ and DQ_D to the death compartment D. This captures the speed at which patients die.
- $\widetilde{\mu}(t)$ is the time-varying marginal probability of death among all infected patients, also known as the infection fatality rate. This function is re-fit to data when the model is restarted. It is parameterized as a monotone function of time

$$\widetilde{\mu}(t) = (\widetilde{\mu_0} - \mu_{\min}) \left(1 + \frac{2}{\pi} \arctan\left(-\widetilde{r_m} t\right)\right) + \mu_{\min},$$

where $\widetilde{\mu_0}$ is the initial probability of death (at the time the model is [re-]started), $\mu_{\min}$ is its minimum, and $\widetilde{r_m}$ is a daily decay rate for mortality. In most areas, mortality is monotonically decreasing ($\widetilde{r_m} > 0$), reflecting improvements in patient management.

A negative decay rate would correspond to a worsening of the probability of death over time, as was observed in some areas when the capacity of the local health system was exceeded. In the version of the model before June 2020, we assumed $\widetilde{r_m} = 0$, as the mortality rate was relatively constant in the early pandemic when optimal patient management was not universally followed.

- p_d is the probability that a contagious person will be detected. It is fixed at 20% based on various early estimations of the detection probability in countries with earlier outbreaks. [172, 107, 137]
- p_h is the probability that a detected case will be hospitalized, and is set to 15%, see [7, 181].

DELPHI is fit separately in each area (country/state/province, as applicable) and over successive "training windows" that begin when a new wave has lasted for at least a month. For each area and training window we fit 13 parameters from the list above ($\widetilde{k_1}, \widetilde{k_2}$ for the initial condition, and $\widetilde{\alpha}, \widetilde{\mu}, \widetilde{\tau}, \widetilde{t_0}, \widetilde{k}, \widetilde{c}, \widetilde{t_{\text{jump}}}, \widetilde{d}, \widetilde{\mu_0}, \widetilde{r_m}$ from the list above) by minimizing a weighted Mean Squared Error (MSE) loss. Let $DT(t)$ and $DD(t)$ denote the number of reported total detected cases and detected deaths, respectively, on day t. Then, the loss function for a training period of T days is defined as:

$$\sum_{t=1}^{T} \frac{t^2}{T^2} \cdot \left(\widehat{DT}(t) - DT(t)\right)^2 + \lambda^2 \cdot \sum_{t=1}^{T} \frac{t^2}{T^2} \cdot \left(\widehat{DD}(t) - DD(t)\right)^2,$$

where $\widehat{DT}(t)$ and $\widehat{DD}(t)$ are predicted detected cases and deaths, respectively. The factor $\frac{t^2}{T^2}$ gives more prominence to more recent data, as recent errors are more likely to propagate into future errors. The lambda factor $\lambda = \min\left\{\frac{DT(T)}{3 \cdot DD(T)}, 10\right\}$ balances the fitting between detected cases and deaths; this re-scaling coefficient was obtained experimentally through cross-validation.

The training windows are dynamically updated, with the goal that each should cover a period in which the enacted measures are not strengthened. Manually tracking policy changes to trigger a re-training proved impractical, given that DELPHI was applied to a large number of areas. Therefore, re-training the model was triggered once a new wave that has lasted for at least one month was detected by tracking the data. In that case, the start

date of the training period was set as to be the beginning of that wave. The rationale behind this heuristic is that, usually, stricter measures are enacted once a new wave is evident and on the rise. As an example, for the United States, in July 2020 re-training was triggered, with the new training period starting from June 2020; and similarly in November 2020, with a start from October 2020. We specifically exclude historical data starting before an area recorded more than 100 cases, for numerical stability and to exclude sporadic outbreaks before the actual epidemic.

To optimize over the highly non-convex search space, we utilize both a local truncated Newton algorithm (TNC) [138] and a global optimization method of dual annealing (DA) [180]. TNC is utilized to produce forecasts on a daily basis while DA, being more computationally expensive, is performed on a weekly basis to shift and re-adjust the parameters more significantly if the underlying mechanics have changed (e.g., in the case of a new wave of cases). Parameters are fit by using bounds of plus/minus ±20% deviation around the latest value for TNC, and bounds of ±50% deviation for DA. When we first trained the model, we used parameter ranges that were obtained from initial estimates derived in studies in South Korea and China, ±20%.

Parameter Date	**Infection Rate** $\tilde{\alpha}$	**Rate of Death** $\tilde{\tau}$	**Resurgence Magnitude** $\tilde{c}$	**MAPE**
2020-10-22	0.409	0.042	0.924	
2020-10-23 (TNC Bounds)	(0.327, 0.491)	(0.034, 0.050)	(0.739, 1.109)	N/A
2020-10-23 (TNC Result)	0.395	0.036	1.108	1.02%
2020-10-23 (DA Bounds)	(0.205, 0.614)	(0.021, 0.063)	(0.462, 1.386)	N/A
2020-10-23 (DA Result)	0.366	0.021	0.059	1.14%

Table 5.1: Parameter update for Georgia, USA on October 23rd, 2020. MAPE refers to the mean average percentage error on the 7-day holdout validation set.

We would illustrate this with a specific example focused on the region of Georgia, USA. Table 5.1 shows the parameters and their bounds for the update on October 23rd, 2020, when we utilized both TNC and DA. We only show a selection of parameters to improve readability. For example, we see that on October 22nd, 2020, the estimated rate of death in Georgia was 0.042, meaning that the average time till death for COVID-19 fatalities was $\frac{1}{0.042} = 23.8$ days. We train TNC and DA using bounds of 20% and 50% respectively around the latest parameters trained on the previous day. We utilize the rolling training window of

historical data mentioned above, leaving out one week of data as validation. As we observe from the table, the MAPE on the validation set from TNC is lower, and thus, the optimized parameters from TNC are accepted while the DA parameters are discarded. After this update, we would continue to use TNC to update the parameter daily until October 30th, 2020, when both algorithms are utilized again to select the best-performing parameters. This dual-track approach allows efficient optimization and ensures close data fit.

We next discuss three key characteristics of the DELPHI model that allow it to flexibly fit a wide range of time-series of observed data.

Accounting for Under-detection.

Only a subset of the SARS-CoV-2 infections are identified through testing. This is because testing resources were scarce in the beginning of the epidemic; some patients will never develop symptoms that are severe enough to prompt them to seek testing or care; some may attribute any symptoms to another infection, such as the common cold; and some may refrain from getting tested for other reasons (e.g., to avoid losing time from work). Although, in reality, the likelihood that a patient will be identified through testing varies across areas and over time, DELPHI treats the probability of detection, p_d, as a global constant *nuisance parameter*. The probability p_d is not identifiable in each area from the available data, namely detected cases and deaths. Fixing it to 20% (which represents an average estimates in different countries from the early studies) is, in practice, no different to assuming other reasonable estimates that vary by area. The model has enough degrees of freedom to adjust the values of other area-specific partially-identified parameters (e.g., the area-specific reproduction rate).

Allowing both the detection probability and the reproduction rate to vary by area or over time offers no additional advantage for predicting detected cases and deaths and may occasionally lead to overfitting issues (see e.g. [118]). In sensitivity analyses (Section 5.3.3), we show that predictions of detected cases and deaths are robust to moderate deviations from the value of $p_d = 20\%$. We also modeled the detection probability as functions of observable data, such as the number of tests that were administered over time in a country, but this did not improve the out-of-sample empirical performance of the model.

DELPHI's goal is not to infer the number of *all*, detected and undetected, infections. If one wished to point-identify the true detection probability, more and different input data, such as random serology testing, would be required. However, such data was, and still is, very sparse, most often pertains to specific cities and counties (see [53, 158, 156, 14] for examples), and only occasionally to countries (mostly in Europe, [e.g. 62, 175]).

Separation of Recovery and Deaths.

DELPHI's compartmental structure allows for different transition rates for recovery and death processes by using the auxiliary compartments U_R, U_D (for undetected patients), DH_R, DH_D (for the hospitalized), and DQ_R, DQ_D (for detected patients), which were described earlier. Figures 5-1b and 5-1c explain how this is achieved. In Figure 5-1b, the outflow from the I compartment is

$$\frac{\mathrm{d}I^-}{\mathrm{d}t} = -(\nu + \widetilde{\tau})I,$$

where ν is the rate of recovery and τ is the rate of death. This implies a fixed probability of death equal to $\frac{\widetilde{\tau}}{\nu+\widetilde{\tau}}$. By contrast, the structure in Figure 5-1c decouples the probability of death $\widetilde{\mu}$ from the transition rates $\widetilde{\tau}$ and ν.

Modeling Effect of Increasing Government Response.

As shown in Figure 5-2, DELPHI models different phases for the government response during a pandemic by means of the area-specific function $\gamma(t)$ in (5.1), which includes sigmoid (arctan) and exponential terms. The concave-convex nature of the arctan term accounts for the first three phases: The early, concave part models initial limited changes in behavior in response to early information, when most people continue business-as-usual activities. The transition from the concave to the convex part of the curve quantifies the sharp decline in infection rate as policies go into full force and people's behavior changes sharply. The latter convex part of the curve models a flattening out of the response as the government measures reach saturation, representing the diminishing marginal returns in the decline of infection rate. The exponential term models a potential resurgence in cases, for instance due

to premature relaxation of societal restrictions or some change in behavior.

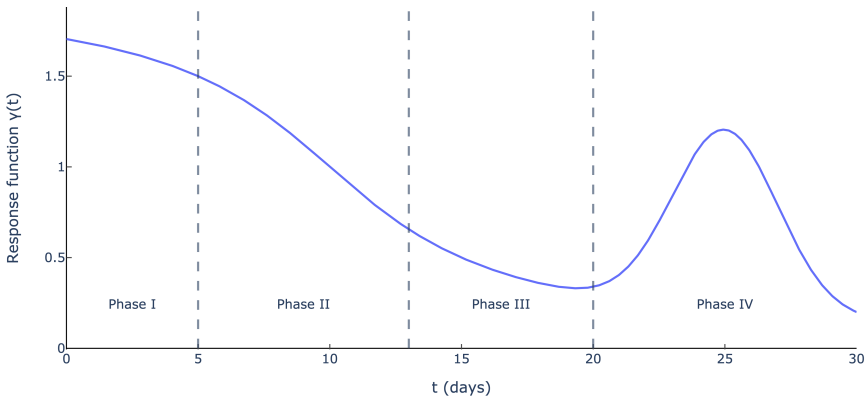


Figure 5-2: Illustration of $\gamma(t) = 1 + \frac{2}{\pi} \arctan\left(-\frac{t-10}{5}\right) + \exp\left(-\frac{(t-25)^2}{8}\right)$ (i.e., $\widetilde{t_0} = 10$, $\widetilde{k} = 5$ $\widetilde{c} = 1$, $\widetilde{t_{\text{jump}}} = 25$, and $\widetilde{d} = 2$).

In (5.1), parameters $\widetilde{t_0}$ and $\widetilde{k}$ control the timing of such measures and the rapidity of their penetration, while the $\widetilde{c}, \widetilde{t_{\text{jump}}}, \widetilde{d}$ control the timing, magnitude, and duration of a resurgence. In Section 5.4.1 we explain that (5.1) forms the basis for modeling a wide variety of policies as the composite of elemental interventions, such as including social distancing, school closings, and stay-at-home orders.

Implementation and availability.

DELPHI was created in early April 2020 and has been continuously updated to reflect new observed data. The codebase is available on GitHub (`https://github.com/COVIDAnalytics/DELPHI`) with the primary model written in Python 3.7 using the SciPy and NumPy libraries. The implementation is also multiprocessing-friendly, which allows it to scale easily (using servers/machines with multiple CPUs/threads) to the high number of areas the model is fitted on every day.

5.3 Results and Performance Analysis

In this section, we present the results of the DELPHI predictive model and its performances in terms of Mean Absolute Percentage Error (MAPE) and Root Mean Squared Error (RMSE) across time and regions, and benchmark it against the state-of-the-art COVID-19 models used by the CDC. We also analyze the sensitivity of DELPHI to perturbations in its parameters.

5.3.1 Forecasting Results

Table 5.3 reports the median MAPE and RMSE for the observed cumulative numbers of cases and of deaths in each area of the world for two periods. The first uses data through April 27th, 2020 and evaluates models up until May 12th, 2020. The second uses data up to September 21st, 2020 and evaluates models through October 6th, 2020. During the second period there was a resurgence of the epidemic, the management of the disease was better understood, and new treatment protocols were in place. For both periods, DELPHI seems to predict the epidemic progression relatively well in most areas with $< 10\%$ MAPE on reported cases, and $< 15\%$ MAPE on reported deaths. The worldwide median MAPE was 5.8% for detected cases and 10.6% for deaths. The areas with the highest MAPE were typically those with the fewest deaths, as shown in the selected examples in Table 5.2. Analogously, the median RMSE for deaths in both periods were < 100 across all regions, which is remarkable given that the RMSE is not scaled by the observed number of deaths and the high number of deaths per region (e.g. by the second period a majority of areas in North America and Europe were reporting over $5,000$ cumulative deaths). The median RMSE on cases are also remarkable at just above $2,000$ cases in the second period given that the daily median number of recorded cases across all areas ($\sim 60,000$).

The worldwide median MAPE for deaths in the second period is smaller than in the first (4.8% vs 10.8%, respectively). This may be partially explained by the better fitting of the probability of death $\widetilde{\mu}(t)$ in the second period, when the cumulative number of deaths was higher and also the higher variability of MAPE when the denominator (observed deaths) is

small (analogously to the observation in Table 5.2).

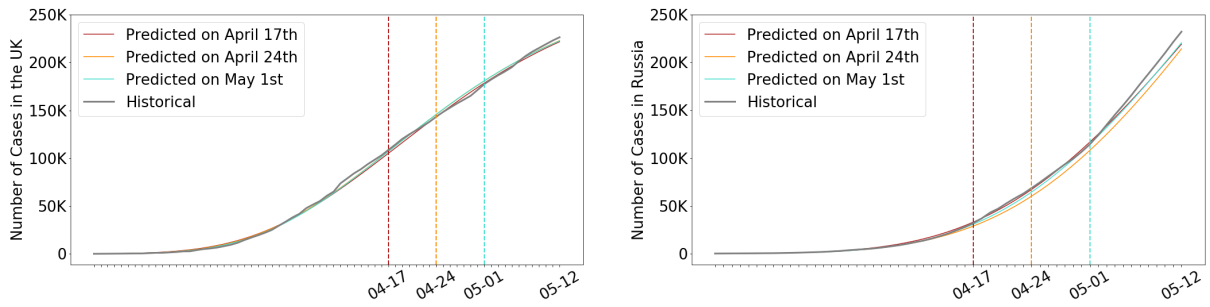
Table 5.2: Breakdown of cumulative number of deaths vs. corresponding prediction MAPE for large errors on the prediction period of April 28th to May12th.

Country/Province	Bahrain	Djibouti	Guinea	Kazakhstan	Sri Lanka	Oman	Qatar	Venezuela
cumulative deaths as of May 11 2020	8	3	11	32	9	17	14	10
MAPE on deaths	89.6%	193.1%	53.3%	62.6%	54.5%	44.0%	106.9%	48.1%

Region	**# Areas**	**Median MAPE Cases (10th, 90th percentile)**	**Median MAPE Deaths (10th, 90th percentile)**	**Median RMSE Cases (10th, 90th percentile)**	**Median RMSE Deaths (10th, 90th percentile)**
April 28th					
Africa	19	14.7% (3.1, 32.0)	23.4% (11.8, 60.3)	138.7 (27.0, 1019.6)	4.8 (1.1, 27.5)
Asia	32	4.8% (2.1, 18.4)	14.4% (2.9, 65.2)	677.3 (51.5, 9778.1)	13.0 (1.0, 151.6)
Europe	42	3.4% (0.8, 12.9)	9.0% (2.3, 24.3)	238.1 (15.4, 3276.4)	14.8 (1.6, 236.2)
North America	10	7.9% (3.9, 28.3)	12.6% (2.8, 23.6)	594.4 (36.1, 1947.5)	16.3 (4.0, 132.3)
Oceania	2	3.2% (2.4, 4.1)	12.0% (11.0, 13.0)	68.6 (49.4, 87.8)	2.0 (1.8, 2.3)
South America	11	14.9% (7.6, 26.7)	6.1% (3.3, 30.1)	683.6 (31.4, 10815.0)	6.8 (0.6, 426.1)
United States	51	8.5% (1.9, 16.7)	7.8% (3.3, 25.1)	1231.6 (73.8, 4861.8)	33.5 (1.7, 210.7)
World	167	5.8% (1.5, 22.6)	10.6% (2.9, 36.6)	412.6 (26.1, 4788.7)	12.0 (1.3, 193.1)
September 22nd					
Africa	53	5.2% (0.6, 30.4)	4.2% (0.0, 42.6)	364.5 (27.5, 2913.2)	8.8 (0.0, 121.7)
Asia	38	6.5% (1.7, 38.4)	8.3% (1.4, 26.8)	6311.3 (120.9, 53046.7)	47.4 (0.7, 719.5)
Europe	44	13.4% (3.8, 38.5)	7.7% (1.1, 22.5)	4452.9 (326.6, 22405.9)	43.6 (1.7, 776.5)
North America	14	7.3% (1.2, 17.1)	4.8% (1.0, 22.0)	2180.4 (187.5, 8656.7)	48.9 (2.0, 269.0)
Oceania	3	2.7% (1.6, 4.2)	1.1% (0.8, 7.1)	51.0 (16.0, 127.5)	0.8 (0.6, 3.8)
South America	13	9.4% (1.2, 15.5)	5.2% (2.1, 22.7)	5683.8 (473.2, 131184.4)	95.6 (3.5, 4438.1)
US	51	5.2% (1.3, 20.6)	3.0% (0.5, 15.0)	5323.5 (661.6, 17571.2)	66.9 (3.1, 275.0)
World	216	6.5% (1.2, 28.2)	4.8% (0.6, 27.6)	2170.0 (69.3, 18549.4)	31.2 (1.1, 505.3)

Table 5.3: Median country-level Mean Absolute Percentage Error (MAPE) and Root Mean Squared Error (RMSE) of the predicted number of cases and deaths in each region. Projections in the first (resp. second) half of the table are made using data up to 04/27 (resp. 09/21) for the period from 04/28 to 05/12 (resp. 09/22 to 10/06).

We now further illustrate the performance of DELPHI with two major countries with very different curves. Figures 5-3a and 5-3b show our projections of the number of cases in Russia and the United Kingdom made on three different dates, and compare them against historical observations. The results for Russia and the UK are consistent with the overall performances across all countries of all regions, as described extensively in the rest of the section. Concretely, the graphs suggest that DELPHI achieves strong predictive performance, as the model has been consistently predicting, with high accuracy, the overall spread of the disease for several weeks across regions with different epidemiological characteristics. Notably, DELPHI was able to anticipate, as early as April 17th, the dynamics of the pandemic



(a) United Kingdom (b) Russia

Figure 5-3: Cumulative number of cases in the UK (a) and Russia (b) according to our projections made at different points in time, against actual observations. Note there predicted curves largely overlap with the actual curve.

in the United Kingdom (resp. Russia) up to May 12th. At a time when 100-110K (resp. 30-35K) cases were reported, the model was predicting 220-230K (resp. 225-235K) cases by May 12th—a prediction that realized a month later. In the case of Russia, DELPHI was able to predict that the country was going to become a global hotspot as well as to accurately estimate the magnitude of the first wave of the outbreak even at an early stage of the pandemic (less than 0.025% of the population infected, vs more than 0.16% a month later which has put the country at the 4^{th} rank worldwide in terms of cumulative number of cases).

5.3.2 Comparison with Other Models

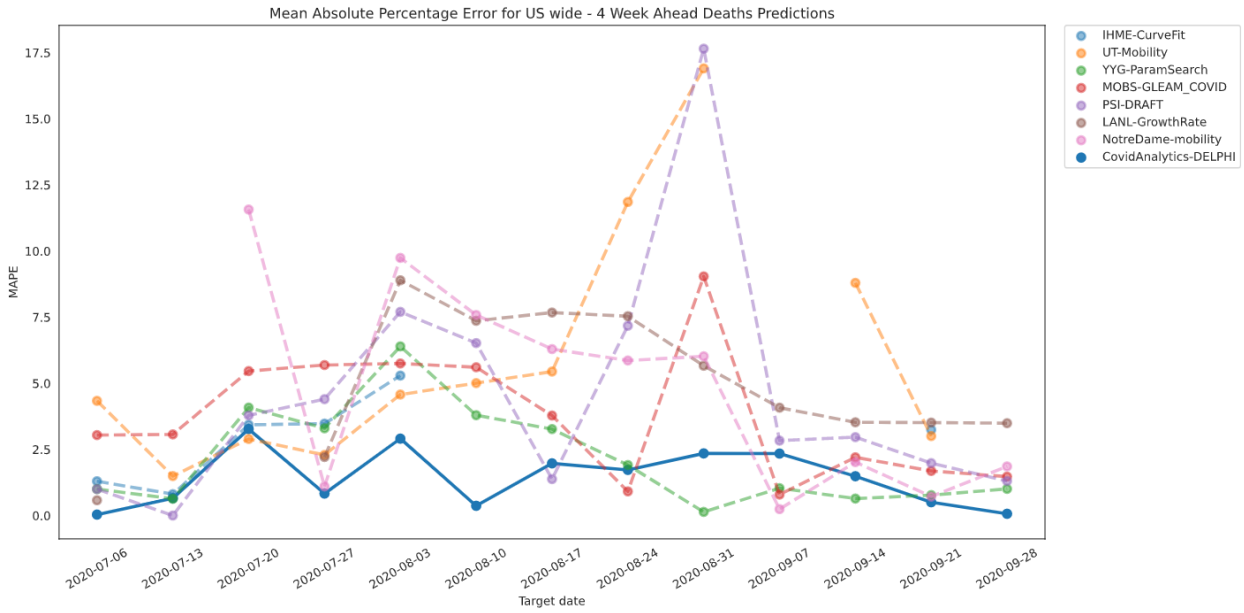
DELPHI compares favorably with other top-performing models submitted to the CDC ensemble forecast in predicting the number of deaths in the United States 4 weeks ahead (the longest time point in the CDC ensemble predictions). As comparator models we selected those that submitted results to the CDC regularly during the epidemic, and were, thus, consistently included in the CDC ensemble forecasts. These were the models by the University of Texas, Austin (UT-Mobility, [178]), the Institute for Health Metrics and Evaluation (IHME-CurveFit, [88]), Youyang Gu (YYG-ParamSearch, [82]), the Northeastern University's Laboratory for the Modeling of Biological and Socio-technical Systems (MOBS-GLEAM_COVID, [43]), Predictive Science Inc. (PSI-DRAFT, [165]), the Los Alamos National Laboratory (LANL-GrowthRate, [139]), and the Notre Dame University (NotreDame-mobility, [142]).

In Figure 5-4a, we compare the out-of-sample MAPE of these models for the number of cumulative deaths 4 weeks in the future above using the actual weekly predictions submitted to the CDC between July and September 2020. This particular period was selected because it encompassed the period of second resurgence in the United States and its decay, making prediction even more difficult. October 2020 was excluded as the CDC ensemble forecast changed its submission and reporting guidelines, prompting submission lapses in many models. We observe that DELPHI consistently achieves low MAPE, and that its predictions are stable with a MAPE never exceeding 3.5% throughout the period. Figure 5-4b further illustrates the performance of DELPHI in comparison to other models by graphing the weekly ranking (with respect to MAPE). We observe that DELPHI consistently outperforms all other models, holds the first rank for 6 out of 13 weeks, and never drops below rank 4 among the 8 models evaluated.

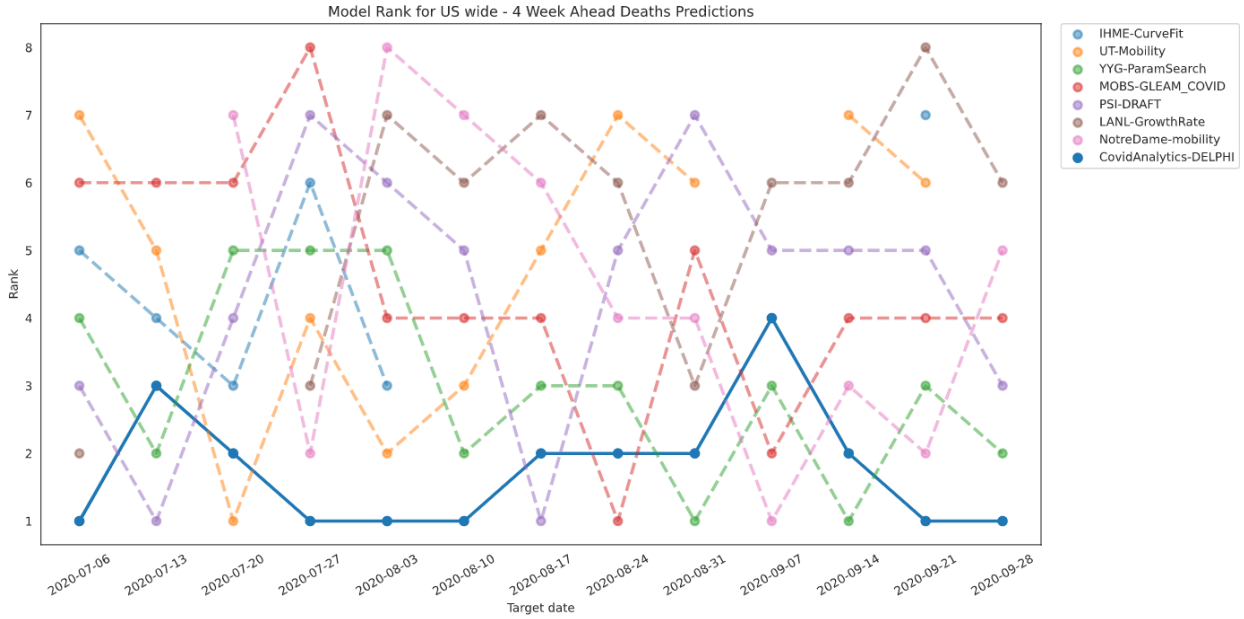
5.3.3 Sensitivity Analysis

We examined the impact on prediction of varying each of the six fixed parameter of DELPHI in univariate sensitivity analyses. For every fixed parameter among β, r_d, σ, κ, p_d, p_h, we randomly perturbed the parameter by a zero-centered normal noise term ε with standard deviation of 20% of the nominal parameter's absolute value, i.e., $\varepsilon \sim \mathcal{N}\left(0, (0.2 \cdot |\text{param}|)^2\right)$. Then we fit the DELPHI model using data up to a certain prediction date using the perturbed fixed parameter, and compared its 30 day out-of-sample MAPE with that of the baseline value.

We conducted these sensitivity analyses for all states in the United States and in six countries around the world with large outbreaks (Italy, Spain, Brazil, South Africa, Japan, and Russia) for three prediction dates. Figure 5-5a and 5-5b record the quantile (box and whisker) plots of the absolute difference between the MAPE of the actual model and the perturbed model for six parameters $(\beta, r_d, \sigma, \kappa, p_d, p_h)$, across the 56 areas (50 US states and six countries) for the three prediction dates. We observe that for all six parameters, across both cases and deaths, the effect of the perturbation on the one-month MAPE is relatively small, with interquartile range mostly falling between $\pm 5\%$ for a perturbation with a standard deviation of 20% of the parameter value. This demonstrates that the results



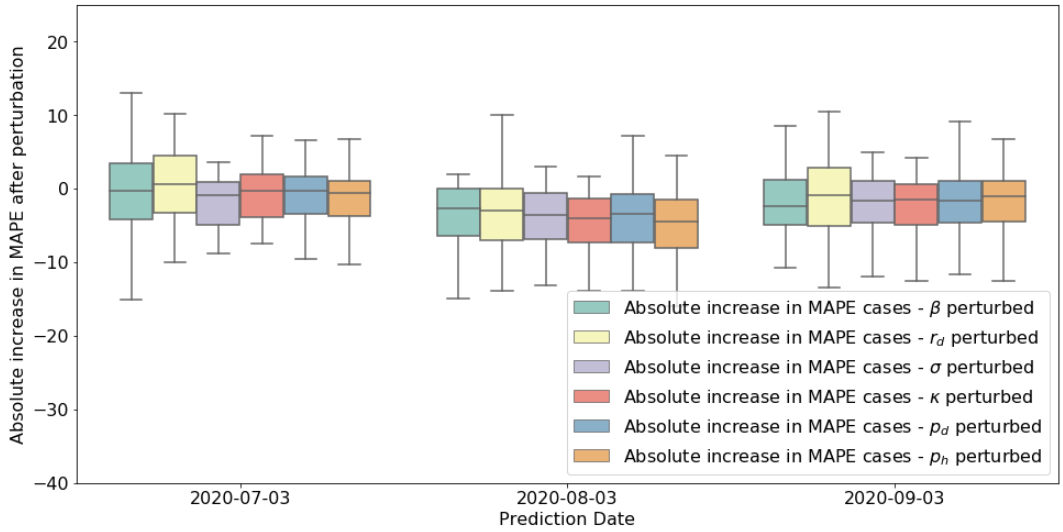
(a) MAPE of DELPHI for US wide 4 week ahead deaths prediction from July to September



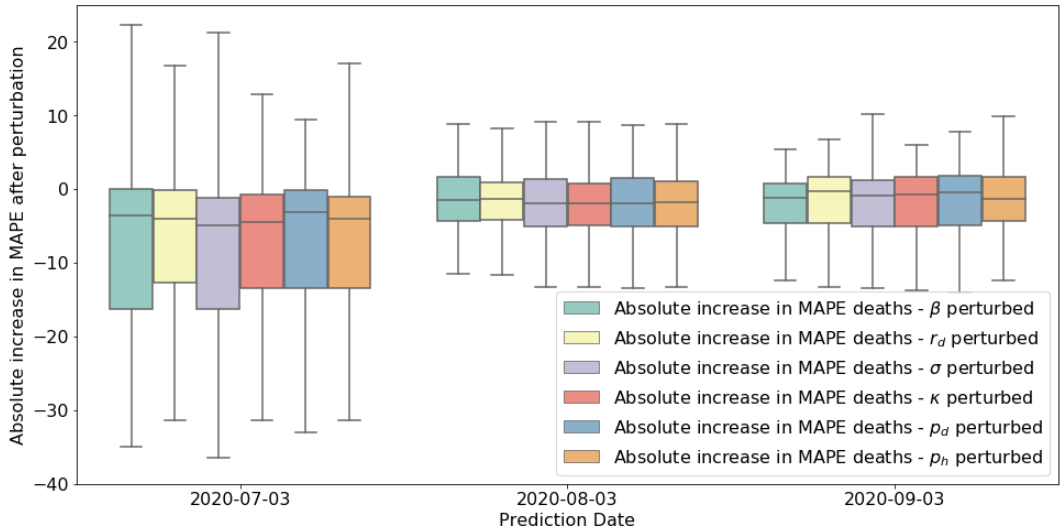
(b) Rank of DELPHI for US wide 4 week ahead deaths prediction from July to September

Figure 5-4: Comparison of 4 week MAPE on deaths prediction in the US between DELPHI and other models used by the CDC

from the DELPHI model are robust to a moderately large perturbation to the underlying parameters.



(a) Sensitivity of predictions on cases based on perturbation of key fixed parameters.



(b) Sensitivity of predictions on deaths based on perturbation of key fixed parameters.

Figure 5-5: Sensitivity analysis of various fixed parameters, comparing perturbed MAPE on cases and deaths to their nominal counterparts without perturbations.

5.4 Applications

DELPHI's predictions of the epidemic's trajectory can inform decisions of policymakers and of research design, as shown in two selected applications. In the first application, we

consider different NPIs to limit social interactions and mixing, and extend the DELPHI model to evaluate their impact on the trajectory of the epidemic. The second application demonstrates a scenario analysis toolkit that can inform the planning of operations, staffing, inventories, or even the development of research designs. This toolkit was used by Janssen Pharmaceuticals to select candidate sites for their Phase III trial of the single-dose COVID-19 vaccine Ad26.Cov-2.S.

5.4.1 Application 1: Evaluating Different Government Intervention Scenarios

In this section, we extend DELPHI to evaluate the impact of government interventions. That allows us to quantify the efficiency of NPIs and predict "what-if" scenarios under different policies, which enable policy-makers to assess their COVID-19 response and decide on their future interventions accordingly. We begin by focusing on the effect of different interventions, and then analyze different "what-if" scenarios.

Effect of Government Interventions

We can use DELPHI to examine the association between five escalating policy categories (1:*no measure*; 2: *restrict travel and work*; 3: *restrict mass gatherings, travel and work*; 4: *restrict mass gatherings, schools, travel and work*; and 5: *stay-at-home*) and the daily infection rate across areas during the first training window up to May 19th, 2020. For each area, we assign each day in the first training window to one of the five policy categories using data from the Oxford Coronavirus Government Response Tracker [85] for countries other than the US, and the Institute for Health Metrics and Evaluation [130] for US states.

We assume that each of the five policy categories has a global fixed effect across all areas. We use a two-step approach: First, we estimate the $\gamma(t)$ function for each area and training window, which captures the net effect of everything that affects infection rates, including government interventions, changes in behavior, workplace policies, etc. In the second step, we treat the $\gamma(t)$ functions as known, and use the presence or absence of the aforementioned five policy categories to explain temporal changes in area-specific infection rates using the

algorithm below.

For each policy category $i = 1, \ldots, 5$ and each area j we extract the average value of $\gamma(t)$, $\bar{\gamma}_{ij}$ across all times for which policy i was in effect. Then we calculate the residual fraction of infection rate under policy i, p_i, compared to the baseline policy of no measure as:

$$p_i := \frac{1}{J} \sum_{j=1}^{J} p_{ij} = \frac{1}{J} \sum_{j=1}^{J} \frac{\bar{\gamma}_{ij}}{\bar{\gamma}_{1j}},$$

where J is the total number of areas we include and p_{ij} is the residual fraction of infection rate under policy i for a specific area j. We normalize the residual fraction of infection rates for different areas because different areas have different background infection rates $\widetilde{\alpha}$.

The estimated "effects" of the five policy categories on infection rates are associations rather than causal estimates, in that they may be confounded by other, umneasured or unobservable, factors. For example, during the first wave, most governments had not yet implemented a mask mandate, and thus we did not include masks in the policy categories. However, some did, and the effects of masking is aliased with the effects of the five policies. On the other hand, the average effect of unaccounted for policies that are active throughout the epidemic, such as contact tracing, is absorbed by the area-specific parameter $\widetilde{\alpha}_j$, which cancels out when estimating each $p_{ij} = \frac{\bar{\gamma}_{ij}}{\bar{\gamma}_{1j}}$. This suggests a small impact of time-invariant unmeasured policies on the estimates p_i.

Restrictions	**Area-Days**	**Residual Infection Rate**
None	2142	100%
Travel and Work	2049	$88.9 \pm 4.5\%$
Mass Gathering, Travel, and Work	340	$59.0 \pm 5.2\%$
Mass Gathering, School, Travel, and Work	1460	$41.7 \pm 4.3\%$
Stay-at-Home Order	6585	$25.6 \pm 3.7\%$

Table 5.4: Implementation Length and Effect of each policy category as implemented across the world.

Table 5.4 shows the number of area-days that each policy was implemented around the world and its average effect over all areas and the standard deviation of the area-specific estimates p_{ij} for each policy i. During the first wave, each policy category was in effect

for hundreds to thousands of area-days worldwide, with the stringent stay-at-home policy category being implemented most extensively. Table 5.4 lists policies in increasing estimated effectiveness. Compared to less restrictive policies, more-stringent policies tend to be implemented later and to be associated with larger decreases in the residual infection rate. This is expected because during the first wave the $\gamma(t)$ function is monotonically decreasing ($\widetilde{c} = 0$), and later-implemented policies will have larger estimated effects. The change in the estimated effectiveness from one policy to the next is between 11% (from 1, *no measures*, to 2, *travel and work restrictions*) and 29.9% (from 2 to 3, *mass gathering, school, travel, and work restrictions*). The most stringent policy category (5, *stay-at-home*) is associated with a reduction of the infection rates to $25.6 \pm 3.7\%$ of the unmitigated value.

The basic reproduction rate, R_0, for the SARS-CoV-2 variant responsible for the first wave of COVID-19 was estimated between 2.5-3.0 ([184, 116]). R_0 measures on average how many new infections one infected patient will generate over the course of their disease in a fully susceptible population (i.e., in the beginning of the epidemic, and approximately, during the first wave). The basic reproduction rate R_0 is proportional to $\gamma(0)$, and the effective reproduction rate R_t, its counterpart for $t > 0$, is proportional to $\gamma(t)$. To control the epidemic R_t should become smaller than 1, or $\frac{\gamma(0)}{\gamma(t)} \lessapprox \frac{1}{2.5}$ to $\frac{1}{3.0}$, if at $t = 0$ the epidemic was unmitigated. This suggests that, to the extent that $\frac{\gamma(0)}{\gamma(t)} \approx \frac{\bar{\gamma}_i}{\bar{\gamma}_1} = p_i$, on average, only policy $i = 5$, stay-at-home-order, appears to be strong enough to fully mitigate the epidemic, albeit at a steep economic and social cost. As we will discuss below, the timing when policies go into effect is critical for minimizing the total number of cases and deaths.

Modeling Alternative Initial Responses

To model what would happen if mitigation were to start m days earlier, we translate the time axis in (5.1) by m days to the left:

$$\gamma'(t) = \frac{2}{\pi} \arctan\left(-\frac{t - (\widetilde{t_0} - m)}{\widetilde{k}}\right) + 1.$$

To illustrate, Figure 5-6 shows the percentage of cases and deaths avoided around the world by May 17th if the government interventions were to be initiated one week earlier for the

50 countries with the highest reduction, which ranges between approximately 30% and 80%. Western European countries such as Switzerland, Spain, and Italy, which had some of the first and steepest outbreaks outside Asia, would have benefited the most. Cumulatively across the world, DELPHI predicts that over 280,000 deaths or 68% of total deaths could have been avoided by May 17th with just a week's earlier start of mitigation efforts.

Another insightful scenario to consider is what would have happened if the epidemic were left unmitigated. This can be modeled as

$$\gamma'(t) = \gamma(0) = \frac{2}{\pi} \arctan\left(\frac{\widetilde{t_0}}{\widetilde{k}}\right) + 1,$$

and would result in over 14.8 million deaths by May 17, 2020.

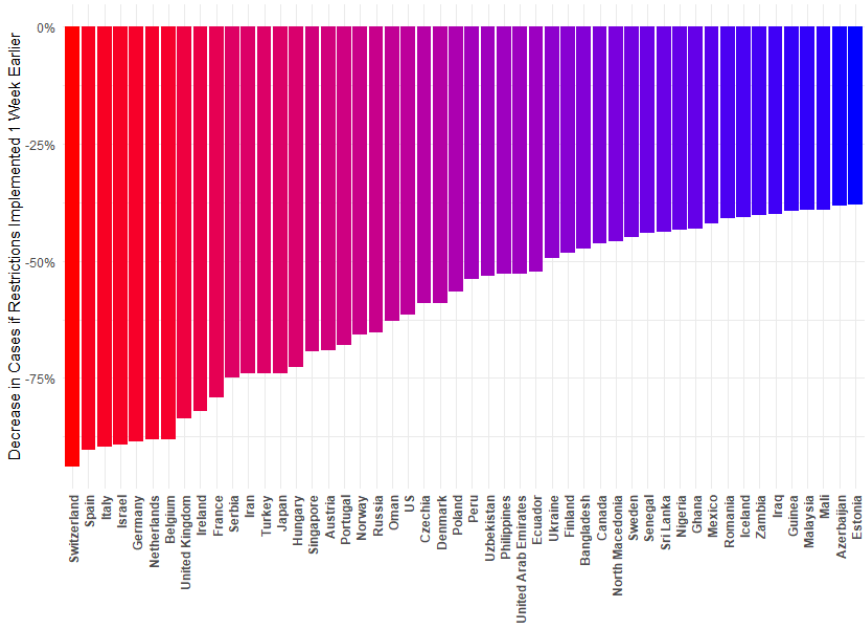
5.4.2 Application 2: Analysis of What-If Scenarios for Long-term Planning

DELPHI was utilized by Janssen Pharmaceuticals in late May 2020 to examine the impact of "what-if" scenarios of relaxing measures in different countries to inform the design of the multicenter Phase III trial of their vaccine candidate Ad26.Cov-2.S. DELPHI's predictions helped identify the best candidate sites (countries with high anticipated incidence and prevalence) to maximize the trial's statistical power.

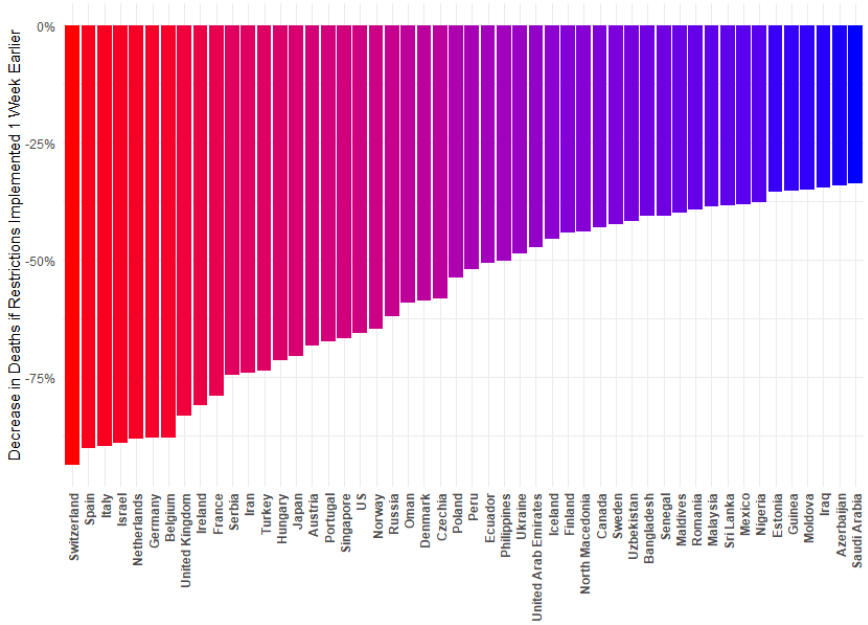
Specifically, suppose that we are considering shifting from policy i to $j < i$ at time t_c in some area that has not yet experienced a resurgence ($\tilde{c} = 0$). Then for all times $t \geq t_c$, we modify (5.1) to

$$\gamma'(t) = \frac{2}{\pi} \arctan\left(-\frac{t - \widetilde{t_0}}{\widetilde{k}}\right) + 1 + \underbrace{(p_j - p_i) \cdot \min\left[\frac{2 - \gamma(t_c)}{1 - p_i}, \frac{\gamma(t_c)}{p_i}\right]}_{\text{Differential in policy effect between policy } i \text{ and } j}, \quad \forall t \geq t_c.$$

The last term is a correction proportional to the difference $p_i - p_j > 0$ in the fractional reductions in the infection rate with policy categories i and j. The multiplicative factor $\min\left[\frac{2-\gamma(t_c)}{1-p_i}, \frac{\gamma(t_c)}{p_i}\right]$ scales the fractional difference so that the resulting $\gamma'(t_c)$ is constrained within the initial range $[0, 2]$. Replacing $\gamma(t)$ with $\gamma'(t)$ forecasts the epidemic under the



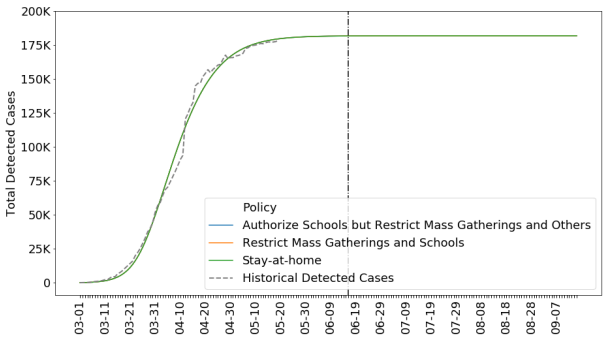
(a) Percentage of cases avoided around the world if policy enacted one week early.



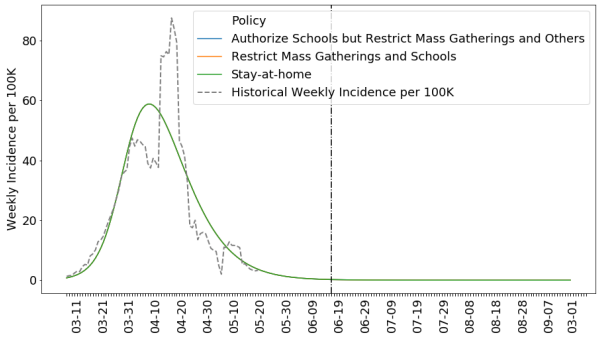
(b) Percentage of deaths avoided around the world if policy enacted one week early.

Figure 5-6: Scenario analysis if restrictions implemented one week earlier.

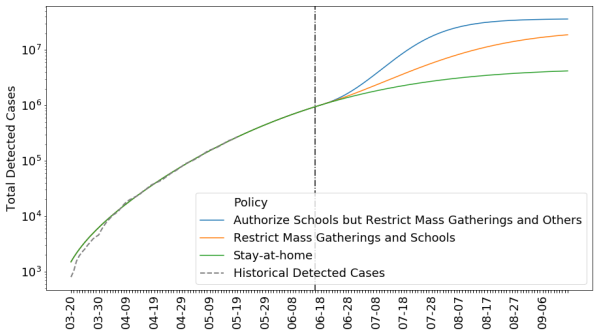
updated policy.



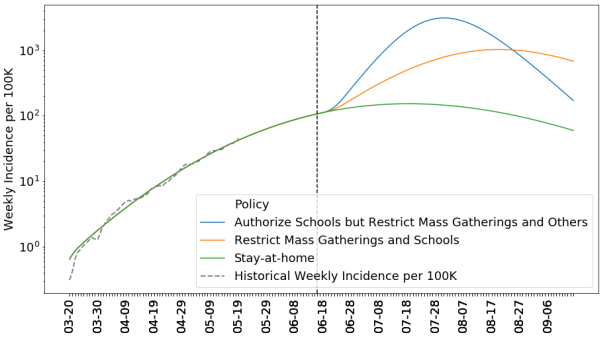
(a) France, total Detected Cases.



(b) France, weekly Incidence per 100K.



(c) Brazil, total detected cases (log scale).

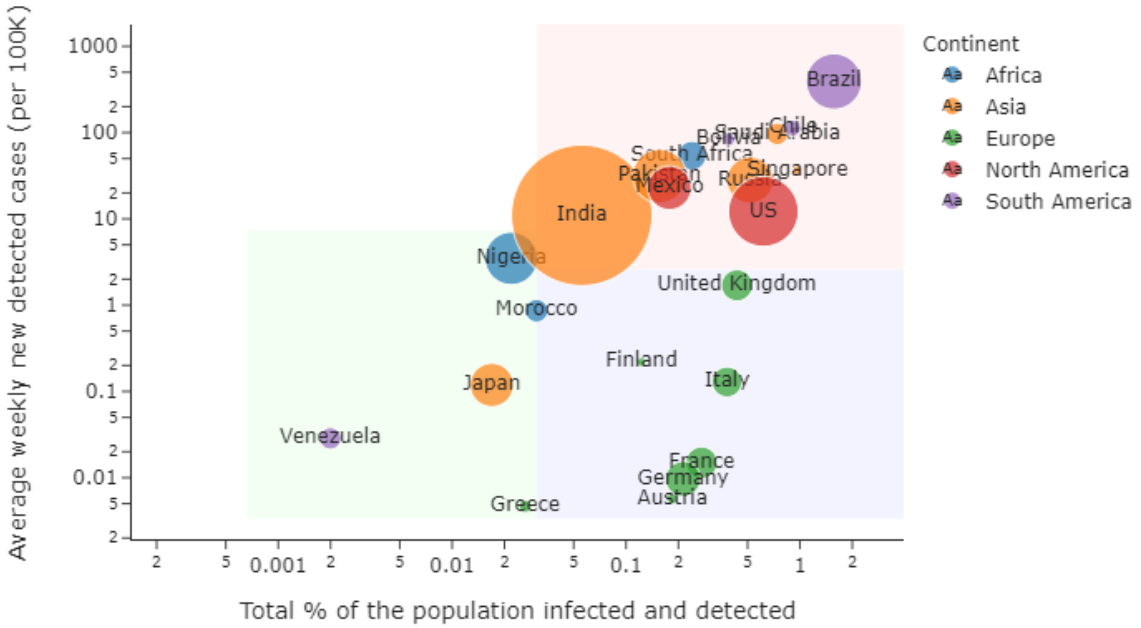


(d) Brazil, weekly incidence per 100K (log scale).

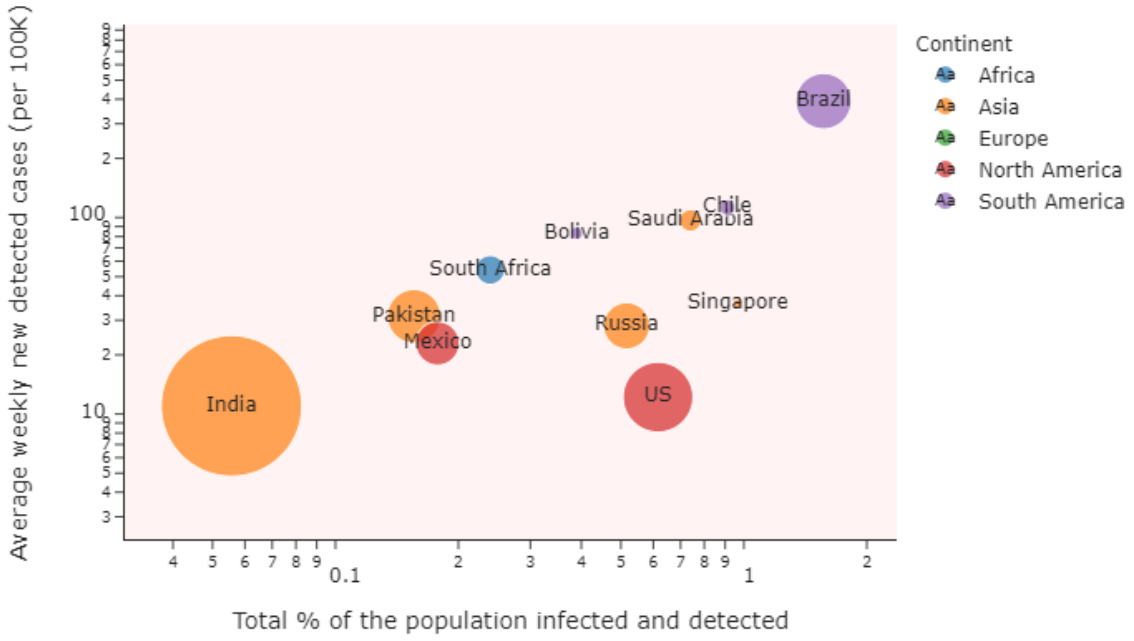
Figure 5-7: Forecasts of total detected cases and weekly incidence per 100K for France and Brazil under various policies. In (a) and (b), the green line completely overlaps with other lines in the asymptotic regime.

The impact of a re-opening strategy varies greatly across areas, as shown in a comparison of Brazil vs France in Figure 5-7. In Brazil, relaxing measures from a stay-at-home order (policy category 5) to restricting mass gathering, travel, and work (category 3) on June 16th would result in a second wave of infections with up to 6.8 million additional cases one month later (Figure 5-7c), as the epidemic was not yet adequately mitigated and the incident cases were still on a steep rise (Figure 5-7d). By contrast, in France, the epidemic had already peaked, was adequately mitigated incident detected cases were declining (Figure 5-7b) and the relaxation of policies would have a much smaller effect (Figure 5-7a). Results for other countries are shown in the Appendix.

Figure 5-8 summarizes analogous scenario analyses across the globe. It shows outcomes one month after a hypothetical relaxing of policy category 5 (stay-at-home) to 3 (mass



(a) Weekly Incidence of Cases (per 100K) in the first half of July against fraction of population infected for multiple countries.



(b) Predictions for total cumulative cases (normalized by the population) vs new cases (per 100K) for countries which are predicted to be highly impacted and still worsening at an alarming rate by July 15th.

Figure 5-8: World Predictions for Early July under Mass Gathering, Travel and Work Restrictions.

gathering, travel, and work restrictions) on June 16, 2020. We observe three clusters of countries:

- Countries with relatively few total cases, with good mitigation of the epidemic (with relatively few incident cases), such as Greece, Japan, Morocco, and Venezuela,
- Countries with comparatively higher total cases, also with an adequate mitigation of the epidemic, mainly in Western and Northern Europe (e.g. the United Kingdom, Italy, France, and Finland), and
- Countries with a large and relatively fast-growing number of cumulative cases, including the United States, India, and Brazil, in which the epidemic is not adequately mitigated. A close-up of these countries is presented in Figure 5-8b. For example, DELPHI predicted that up to 8% of Brazil's population would be confirmed with COVID-19 one month after a relaxation to policy category 3.

Janssen Pharmaceuticals applied this analysis around the world in May-June 2020 to identify candidate sites for their Ad26.Cov2.S vaccine trial, which was planned for September 2020. For all candidate countries, they performed analyses for all policy relaxations, and prioritized for further consideration and feasibility analyses, those with predicted weekly incidence of confirmed cases > 25 per 100,00 people by the anticipated trial start date. For example, from Figure 5-7, Brazil would be a candidate country, but France would not be. Such analyses informed their final selection of 8 countries, namely, Argentina, Brazil, Chile, Columbia, Mexico, Peru, South Africa, and United States. Notably, prior to using DELPHI, Janssen did not consider Brazil and South Africa. In retrospect, adding these two high incidence countries provided valuable information on the effectiveness of the vaccine against emerging SARS-CoV-2 variants (gamma in Brazil, beta in South Africa).

5.5 Limitations

We briefly discuss several limitations of our approach.

First, DELPHI is a deterministic model, with several input parameters fixed to literature-derived point estimates, and other parameters fit to data. As presented here, it is not used

for uncertainty propagation and quantification tasks [47], which are important for contextualizing forecasts and predictions of "what-if" scenarios for policy making. However, insights from deterministic models are still useful in understanding the dynamics of the disease, and for practical uses, as demonstrated in the applications. Changing the input parameters from fixed values (point mass distributions) to parametric or empirical probability models is straightforward, but would be computationally intensive, complicating logistics.

Second, all input parameters that were not fit to data were fixed to global values obtained from the early literature. These include the probability that a case is detected (p_d), and parameters that describe the biology of the disease, such as the mean duration of the incubation period (β^{-1}), the mean time to detection (r_d^{-1}), and the probability that an infection will result in hospitalization (p_h). Some of these parameters, such as the mean time to detection, are fairly consistent in the literature (e.g., [110], [87, 104, 115, 80]) For others, information is sparse. For example, the true (unobserved, latent) detection probability varies by country and over time. However, it is not identifiable in each modeled area on the basis of the available data, namely detected cases and deaths. Fixing it to 20% (which represents an average of early estimates in different countries from the early studies) is, in practice, no different to assuming other reasonable estimates that vary by area. The model has enough degrees of freedom to adjust the values of other area-specific partially-identified parameters (e.g., the area-specific reproduction rate). With DELPHI, the goal is to predict future *detected* cases and deaths. If one wished to also calibrate the model so that it could infer (fit) the actual detection probability, more and different input data would be required, for example random serology testing data. However, such data was (and still is) only very sparsely available among the 200+ areas in which we make predictions.

Third, DELPHI does not explicitly account for population stratification by sex, age, or occupation. There can be substantial variation in transmissions within and across population strata, especially age groups, which in turn can give rise to complicated dynamics [76, 28, 109]. However, marginal modeling of the population allowing some quantities to vary over time (e.g., here $\gamma(t), \mu(t)$) suffices to approximate any dynamics induced by population heterogeneity. This is demonstrated empirically by the fact that DELPHI (and other marginal models) capture the realized dynamics across US states [51] and in many countries.

Fourth, DELPHI uses a marginal probability of death ($\mu(t)$) (infection fatality rate) and a marginal mean time to death $\widetilde{\tau}^{-1}$ for all compartments, irrespective of hospitalization or detection status. However, it has enough degrees of freedom to satisfactorily predict the total detected cases and deaths in diverse settings and under varying mitigation policies.

Fifth, DELPHI assumes that patients only participate in the infection process only for an average duration of $r_d^{-1} \approx 2.9$ days, after the disease has incubated and before patients are self-quarantined (for undetected cases) or quarantined or hospitalized (for detected cases). However, because $R_0 = \widetilde{\alpha} \cdot \gamma(0) \cdot r_d^{-1}$, only the ratio $\widetilde{\alpha}/r_d$ is identified, and the fitting algorithm simply finds a suitable area-specific infection rate $\widetilde{\alpha}$. Further, DELPHI assumes that all patients will eventually develop enough symptoms to prompt their self-isolation, or to seek care and get test-detected. While currently it is understood that not all patients develop symptoms, asymptomatic transmission is possible but perhaps less common than during symptoms [89, 151, 29], and that some patients with symptoms will not self-isolate or seek detection or care (e.g., to not lose working days and income), this was not as clear early on. However, as descussed above, DELPHI has enough degrees of freedom to yield empirically good predictions.

Finally, there are shortcomings in our estimation of the effects of policy interventions in the first application. As discussed in the first application, the estimated "effects" are associations rather than causally-interpretable quantities, and they can be confounded by other, umneasured or unobservable, factors. To facilitate fitting, we assumed that the relative reduction in the infection rate p_i associated with the i-th policy category is homogeneous across areas. These simplifications were motivated by lack of data and pertinent information. We believe that the key qualitative insights on the differential impact of relaxation of policies across areas and the impact of earlier versus later implementation of measures are fundamental and robust to these simplifications.

5.6 Conclusions

DELPHI is a detailed epidemiological model that accurately predicted the spread of COVID-19 in many countries, and aided planning for many organizations worldwide, including gov-

ernmental entities, hospitals, and pharmaceutical companies. By modeling the impact of government interventions, DELPHI provided key insights on the effects of differential timing in the implementation and relaxations of government policies on total detected cases and deaths.

Chapter 6

Where to Locate COVID-19 Mass Vaccination Sites?

6.1 Introduction

The outbreak of the COVID-19 pandemic has started a global race to develop vaccines, fueled by extensive investments, governmental support, and scientific breakthroughs. Thanks to these unprecedented efforts, the scientific community delivered the good news that the whole world was eagerly awaiting. By Summer 2020, several vaccines had been developed. By the end of 2020, several vaccines got approved for emergency use and hundreds more were going under development and testing. Whereas vaccine development used to take years and even decades, these results rank, with no doubt, among the greatest scientific achievements [120, 78].

Unfortunately, discovering and developing a vaccine for COVID-19 was just the beginning: it will now take months to produce, distribute, and deliver vaccines at scale. The world has quickly come to the realization that vaccines cannot be made available immediately to everyone, and policy makers need to make tough decisions to pilot vaccine distribution. A global consensus has naturally emerged to prioritize to healthcare workers, other front line workers, and vulnerable populations such as older people and people with comorbidities [see, e.g., 132]. Within these general principles, each jurisdiction is designing more detailed eligibility guidelines to distribute vaccines effectively and equitably *within a population*, based

on demographic, clinical and geographic factors. However, a question remains open: how to plan vaccine distribution *across populations*, that is, how to allocate a limited vaccine supply across communities, across provinces, and even across countries?

In the United States, this question gained prominence in the midst of a presidential transition. In particular, the new Biden administration relies on higher extents of federal coordination in vaccine distribution, as opposed to a more decentralized approach at the state level. In one of its first major decisions, the administration started opening mass vaccination sites, with many more planned over the next few weeks.* This environment raises the critical question of where to locate these vaccination sites. Obviously, these decisions need to adhere to a number of political and fairness considerations—most notably, there must be at least one site per state. Yet, there remains flexibility to use mass vaccination sites as a strategic lever to effectively combat the pandemic.

This paper addresses this question with a novel data-driven approach, combining epidemiological modeling and prescriptive analytics, to optimize the location of vaccination sites and the subsequent allocation of vaccines. To this end, we leverage a recent compartmental epidemiological model called DELPHI (Differential Equations Lead to Predictions of Hospitalizations and Infections), which extends Susceptible-Exposed-Infected-Recovered (SEIR) models to capture critical drivers of the COVID-19 pandemic: (i) under-detection due to limited testing, (ii) governmental and societal response, and (iii) declining mortality rates [113]. The DELPHI model has been fitted from historical data at the country level, at the state level in the United States, and at the province level in a few other countries. The DELPHI forecasts have been incorporated into the ensemble forecast from [168] and have been utilized in selecting the Phase III trial locations for the Johnson and Johnson COVID-19 vaccine. Historically, the DELPHI model has featured excellent predictive performance, matching the number of detected cases and deaths with high accuracy across the various waves of the pandemic.

In this paper, we integrate the (predictive) DELPHI model into a (prescriptive) optimization model for vaccine allocation. We first propose an extension of DELPHI, referred to

*www.nbcnews.com/politics/white-house/federal-government-opening-first-mass-covid-19-vaccination-sites-california-n1256611

as DELPHI–V, to capture the effects of vaccinations on the dynamics of the pandemic. The DELPHI–V model also disaggregates the dynamics of the pandemic at the subpopulation level to reflect disparities in mortality rates across age groups, which are critical drivers of vaccination strategies. We then formulate an optimization model, referred to as DELPHI–V–OPT, which optimizes the vaccine distribution strategy (that is, the deployment of mass vaccination sites at the strategic level, and the subsequent allocation of vaccines at the tactical level) to minimize the death toll of the pandemic. Our focus on mass vaccination centers does not hinder the role that smaller vaccination sites (e.g., pharmacies) have been playing throughout the country to vaccinate the population. Ideally, our modeling approach would consider these various sites jointly. However, given the lack of publicly available information on the vaccines administered in smaller sites and the lack of coordination between the various vaccination sites, we leave this integration for future research.

From a technical standpoint, the DELPHI–V–OPT model relies on time discretization to embed the system of ordinary differential equations governing the DELPHI–V dynamics into an optimization model. The model is formulated as a bilinear (non-convex) optimization model, due to the SEIR dynamics at the core of DELPHI–V in which the number of new cases is driven by the number of susceptible and infected people. To solve it efficiently in realistic large-scale settings, we propose a coordinate descent algorithm. Starting from a baseline solution, the algorithm iterates, until convergence, between optimizing the vaccine distribution strategy (for given dynamics of the pandemic) and simulating the dynamics of the pandemic (for a given vaccine distribution strategy).

We implement the proposed model and algorithm using real-world data in the United States from [136], [167], and [170]. We leverage the parameter estimates from the DELPHI model in each US state. One challenge, however, is that DELPHI estimates mortality rates in each state in each time period, while [170] reports mortality rates in each age bracket. To develop realistic and consistent estimates for mortality rates in each state, each age group and each time period, we formulate another bilinear optimization model that interpolates these two pieces of information, while ensuring consistency with broader demographic information.

Results suggest that the locations of vaccination sites can have a massive impact on the effectiveness of the vaccination campaign. As compared to several benchmarks based on

demographic information (e.g., city and state population) and epidemiological information (e.g., case counts), our optimization approach increases the number of lives saved by the vaccines by 20%, or 4,000 lives over a three-month period in the United States. These results underscore the necessity to consider both demographics and epidemiological dynamics when determining the locations of vaccination sites and subsequent vaccine allocation, which is achieved by the combination of our DELPHI–V epidemiological model and our optimization framework. In addition, the optimization approach can ensure equity between states and across vaccination sites, thus alleviating the death toll of the pandemic in some states without hurting others. Finally, these benefits are highly robust to misspecifications and fluctuations in the DELPHI parameters. Practically speaking, even though tactical decisions (e.g., vaccine allocation) need to be revised continuously in response to the latest information available throughout the vaccination campaign, strategic decisions (i.e., the location of vaccination sites) are highly robust to noise and uncertainty.

In summary, this paper makes three contributions. From a modeling standpoint, it formulates a novel optimization model for vaccine allocation, DELPHI–V–OPT, that integrates a state-of-the-art epidemiological model into an optimization model that supports vaccine distribution strategies, in order to mitigate the impact of the pandemic. From a computational standpoint, it develops a scalable coordinate descent algorithm, which converges effectively and in short runtimes. From a practical standpoint, it demonstrates that optimizing the locations of mass vaccination sites can curb the death toll of COVID-19 by a sizeable amount, thus highlighting the critical role of vaccine distribution besides vaccine design and vaccine production in combating the pandemic. Obviously, vaccine distribution involves broad political, economic and social considerations, which lie beyond the scope of this paper; yet, this paper can play a critical role to support ongoing mass vaccination efforts in order to mitigate the impact of the pandemic on public health.

6.2 Literature review

Many pharmaceutical companies and academic institutions have explored different technologies toward a SARS-CoV-2 vaccine [153, 69]. These span (i) inactivated or live-attenuated

virus vaccines, which induce an immune response from weakened or killed pathogens (used by the Wuhan Institute of Biological Products, for instance); (ii) viral vector vaccines, which exploit non-replicating adenoviruses to deliver an antigenic element (used by Johnson and Johnson, for instance); (iii) subunit vaccines, which use a minimal structural component of a pathogen such as a protein (used by Clover Biopharmaceuticals, for instance); (iv) nucleic acid vaccines, which deliver DNA or mRNA of viral proteins (used by Pfizer and Moderna, for instance).

From an operational standpoint, a vast literature studies vaccine supply chains [see 56, 112]. A first area involves optimizing vaccine composition [179, 106, 44, 10]. A second area focuses on vaccine production to manage supply-side and demand-side uncertainty and mitigate incentive misalignments between manufacturers and end users [42, 67, 5]. Next, vaccine allocation optimizes the management of a vaccine stockpile [159, 123]. Last, vaccine delivery optimizes inventory, distribution and dispensing operations [90, 1, 49]. Most of this research focuses on predictable and repeatable epidemics, such as seasonal influenza. For less predictable epidemics, such as pandemic influenza, advance planning interventions include stockpiling [91] and anticipatory vaccination [6]. Unfortunately, these approaches are not readily applicable to a new disease such as COVID-19.

Our paper deals with centralized vaccine allocation within a population. Early studies established the importance of partitioning the population into risk classes (e.g., age groups) to reflect the impact of an epidemic [173, 60, 117]. [61] propose a life-cycle model that prioritizes the most valuable subpopulations. Within a region, results suggest prioritizing at-risk populations [141, 45] or active agents who can spread the disease fastest, such as school children [59, 12, 127, 111, 125]. Across regions, results suggest that vaccines should be allocated to the most infected regions and to those affected the latest by the epidemic [4, 98].

Methodologically, most studies integrate SEIR or similar epidemiological models into simple optimization routines based on scenario analysis, enumeration, simulation, or simple heuristics [166, 163]. [162] propose a chance-constrained optimization approach to ensure that the post-vaccination reproduction number is lower than one with high probability. [183] formulate a two-stage stochastic programming model to first plan vaccine allocation and then

distribute additional doses where the epidemic has not been contained. They model the dynamics of disease propagation by means of a stochastic SEIR model, and define scenarios using Monte Carlo simulation. In contrast, this paper directly embeds SEIR dynamics into an optimization model to support vaccine distribution.

Finally, this paper contributes to the fast-growing field of vaccine distribution in the midst of the COVID-19 pandemic. Recent and ongoing research spans vaccine production [101], equity in vaccine distribution [129, 8], and public acceptance [54, 48]. In terms of vaccine distribution, [145] propose a mixed-integer formulation to support influenza vaccine distribution during the COVID-19 pandemic. [124] study which populations to prioritize in a mass vaccination campaign, trading off vaccinating high-risk (older) age-groups vs. high-transmission (younger) age-groups in a given location. In contrast, this paper optimizes the distribution of vaccines across locations. This relates to [79], who study the spatiotemporal distribution of vaccines, using an SEIR model to test various strategies based on demographic and epidemiological factors.

This paper expands this recent body of work in three major ways. First, we optimize vaccine allocation across regions and risk classes (e.g., age groups), based on data-driven estimates of infection and mortality rates. Second, we leverage a recent SEIR-inspired epidemiological model that captures dynamics specific to the COVID-19 pandemic, such as under-detection, governmental response, and declining mortality rates. Third, we propose a formal optimization approach and a coordinate descent algorithm to explicitly optimize vaccine distribution strategies, as opposed to relying on enumeration, simulation or simplified heuristics.

6.3 Model formulation

Our model optimizes vaccine distribution strategy. In the US context, this primarily involves the location of mass vaccination sites. However, optimizing these decisions requires to account for subsequent vaccine allocation across the population, in order to further optimize and evaluate the effects of the vaccination campaign. Therefore, we refer to as *vaccine distribution strategy* the set of three decisions: (i) the location of mass vaccination sites, (ii)

the allocation of vaccines across vaccination sites, and (iii) the allocation of vaccines within each sub-population.

We capture the dynamics of the pandemic by means of an epidemiological model, called DELPHI, which forecasts the number of detected cases, hospitalizations and deaths in each US state [113].† We review it briefly, and augment it to capture the effects of vaccinations—we refer to this model as DELPHI–V. We then embed the DELPHI–V model into a mathematical programming model to optimize vaccine allocation, referred to as DELPHI–V–OPT.

6.3.1 DELPHI: Forecasting the dynamics of the COVID-19 pandemic

DELPHI is a compartmental epidemiological model, which extends the widely used SEIR model to account for specificities of the COVID-19 pandemic. The model is governed by a system of ordinary differential equations (ODEs) across 11 states: susceptible (S), exposed (E), infectious (I), undetected cases who will recover (U^R) or die (U^D), hospitalized cases who will recover (H^R) or die (H^D), quarantined cases who will recover (Q^R) or die (Q^D), recovered (R) and dead (D).

DELPHI differs from most other COVID-19 forecasting models [see, e.g. 102, 142, 150] by capturing three key elements of the pandemic:

- **Under-detection**: Many cases remain undetected due to limited testing, asymptomatic carriers, and detection errors. Ignoring them would underestimate the scale of the pandemic. The DELPHI model captures them through the U^R and U^D states.

- **Governmental and societal response**: Social distancing policies limit the spread of the virus. Ignoring them would overestimate the scale of the pandemic. However, if restrictions are lifted prematurely, a resurgence may occur. We define a governmental and societal response function $\gamma(t)$, which modulates the infection rate and is parameterized as follows:

$$\gamma(t) = 1 + \frac{2}{\pi} \arctan\left(\frac{-(t - t_{\text{int}})}{\omega}\right) + c \exp\left(-\frac{(t - t_{\text{jump}})^2}{2\sigma^2}\right). \tag{6.1}$$

†DELPHI is also applied to each country and to other provinces, but this paper focuses on US states.

This parameterization defines four phases (Figure 6-1). In Phase I, most activities continue normally. In Phase II, the infection rate declines sharply as policies get implemented. The parameters t_{int} and ω can be interpreted as the start time and strength of this response. In Phase III, the decline reaches saturation. The epidemic then experiences a resurgence of magnitude c in Phase IV, due to relaxations in governmental and social restrictions. This is counteracted at time t_{jump}, when restrictions are reimplemented, with σ controlling the duration of this second wave.

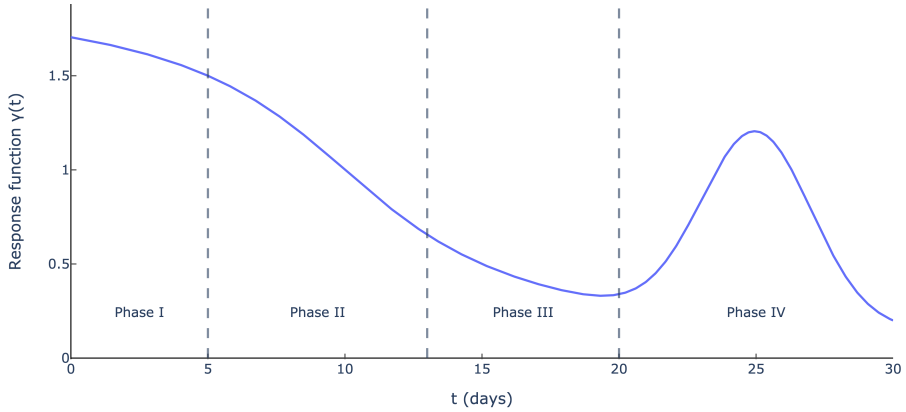


Figure 6-1: Governmental and societal response function $\gamma(t)$ ($\omega = 5$, $t_{\text{int}} = 10$, $c = 1$, $t_{\text{jump}} = 25$ and $\sigma = 2$).

- **Declining mortality rates**: The mortality rate of COVID-19 has been declining through the pandemic, due to a better detection of mild cases, enhanced care for COVID-19 patients, and other factors. We model the mortality rate as a monotonically decreasing function of time:

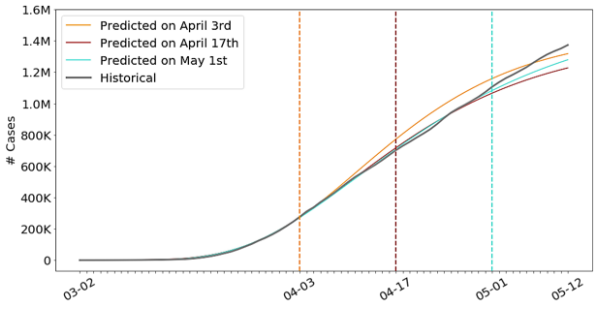
$$m(t) = (m_0 - m_{\min}) \left(1 + \frac{2}{\pi} \arctan\left(-r_m t\right)\right) + m_{\min}, \tag{6.2}$$

  where m_0 is the initial mortality rate, $m_{\min}$ is the minimum mortality rate and r_m is a decay rate.

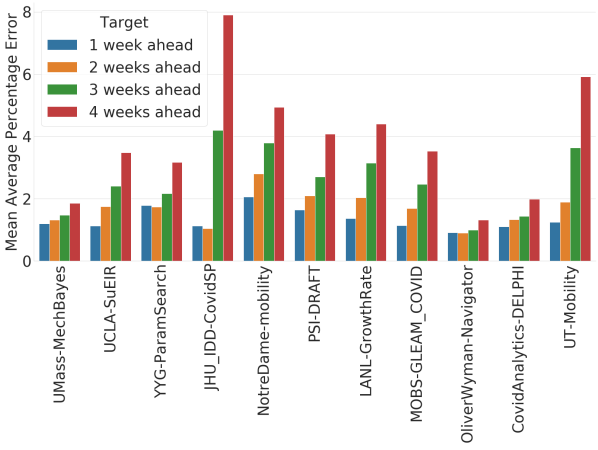
Ultimately, DELPHI involves 16 parameters that define the transition rates between the

11 states. We calibrate seven of them from a database on clinical outcomes [18]. Using non-linear optimization, we estimate the other 9 parameters from historical data on the number of cases and deaths in each region. We refer to [113] for details.

Since its inception in March 2020, DELPHI has been extensively tested and validated against real-world data. Figure 6-2 reports the historical performance of the model in the United States, during the first wave in the Spring of 2020 and the second wave in the Fall of 2020. As the figure shows, the model has been predicting the magnitude of the pandemic with high accuracy up to one month in advance; for instance, as early as April 3, 2020, the model was predicting 1.2–1.4 million cases in the United States by early May, a prediction that became quite accurate a month later (Figure 6-2a). Obviously, subsequent forecasts, by leveraging more up-to-date information, were able to refine these estimates. As a result, the DELPHI model was incorporated into the ensemble forecast from [168]. During the second wave of the pandemic, DELPHI continued to exhibit strong predictive performance, with a mean average percentage error among the lowest of the CDC ensemble forecast (Figure 6-2b).



(a) US First-wave predictions



(b) Second-wave performance

Figure 6-2: Historical performance of the DELPHI predictions in the United States.

6.3.2 Predictive DELPHI–V: Capturing the effects of vaccination

We now augment the DELPHI model to capture two key aspects of vaccinations:

1. *Disparate impacts of the disease across risk classes.* Age is one of the primary drivers of mortality [83, 77, 143]. [169] reports that the mortality rate among Americans aged 70 and over is two orders of magnitude greater than for those aged 30 and under. We partition the population into *risk classes*, defined as homogeneous groups with comparable health characteristics. We consider age-based risk classes in our experiments, but other categorizations could be used (e.g., based on comorbidities). Accordingly, we replicate the 11 model states for each risk class.

2. *Impact of vaccinations on the dynamics of the pandemic.* A fraction of vaccinated people will be immune to the disease (based on the vaccine's effectiveness). Clinical trials suggest that early-approved vaccines prevent mortality but not necessarily infections. Therefore, we assume conservatively that all vaccinated people can still transmit the disease. We relax this assumption later on, to show the robustness of our results when a fraction of vaccinated people become fully immune to the disease. We create four new model states: susceptible and vaccinated (S'), exposed and vaccinated (E'), infected and vaccinated (I'), and immune (M).

Figure 6-3 shows a simplified flow diagram of the DELPHI–V model, with two risk classes (indexed by $k = 1, 2$ and indicated via subscripts). For expositional purposes, we omit dependencies on the region, since the DELPHI–V model is fitted in each region independently. In the remainder of this paper, we also ignore the recovery states, since they do not impact the death-minimization optimization model. Accordingly, we denote the states of undetected, hospitalized and quarantined people who will die from the disease by U, H and Q (as opposed to U^D, H^D and Q^D).

For simplicity, we make three assumptions. First, the effects of vaccines are instantaneous (relaxing this assumption, although straightforward, would merely induce a time lag into the system, without significantly impacting the vaccine distribution strategy). Second, the vaccine has no effect when it fails to immunize the patient (i.e., no partial benefit and no

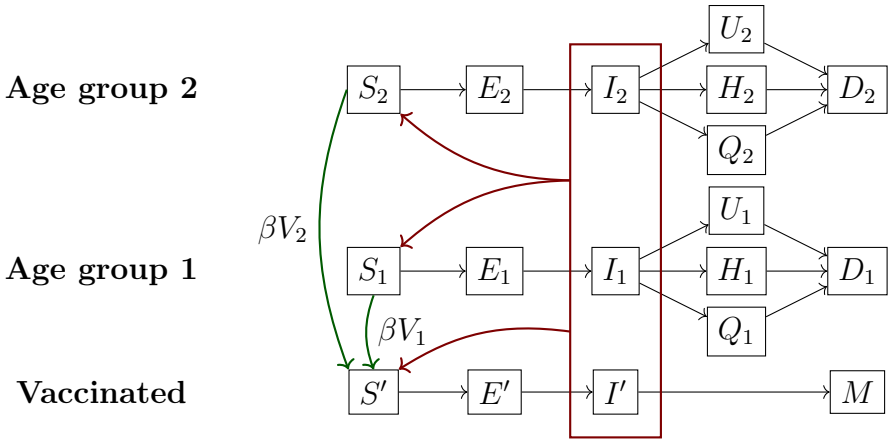


Figure 6-3: Simplified flow diagram of the DELPHI–V model.

side effect). Third, we consider single-dose vaccines. In reality, vaccines can require a single dose or two doses. Double-dose vaccines could be modeled by adding another state of one-dosed patients between S_k and S' (similar to the construction in [122]). This modeling extension would raise new questions surrounding the likelihood of one-dosed patients to contract, transmit and die from the disease—all of which involve significant uncertainties in the absence of relevant data. In addition, given the heterogeneity of vaccines currently available, this extended model extension would end up determining which states get which *type* of vaccines. These decisions, however, are mainly driven by supply chain considerations rather than epidemiological considerations. Therefore, we focus on single-dose vaccines in this paper, and lead the integration of double-dose vaccines for future research.

Given these assumptions, the model captures the effects of vaccinations as follows. Let $V_k(t)$ denote the population mass from risk class $k \in \mathcal{K}$ that gets vaccinated at time $t \in \mathcal{T}$, and let $\beta \in (0, 1]$ denote the vaccine's effectiveness. A mass $\beta V_k(t)$ of people transitions from the susceptible state S_k to the state S', and the remaining mass $(1 - \beta)V_k(t)$ remains in the susceptible state. People in the S' state can become exposed and infected, but then become immune to the disease (as opposed to having a positive probability of dying from it). Note that infections are driven by the total mass of infected people, across all risk classes and vaccinated people. All other transitions shown in Figure 6-3 are consistent with the DELPHI model.

The DELPHI–V model is governed by the following ODE system:

$$\frac{\mathrm{d}S_k}{\mathrm{d}t} = -\beta V_k(t) - \alpha\gamma(t)\left(S_k(t) - \beta V_k(t)\right)\left(\sum_{l=1}^{K} I_l(t) + I'(t)\right) \tag{6.3}$$

$$\frac{\mathrm{d}S'}{\mathrm{d}t} = +\beta\sum_{\kappa=1}^{K} V_\kappa(t) - \alpha\gamma(t)\left(S'(t) + \beta\sum_{\kappa=1}^{K} V_\kappa(t)\right)\left(\sum_{l=1}^{K} I_l(t) + I'(t)\right) \tag{6.4}$$

$$\frac{\mathrm{d}E_k}{\mathrm{d}t} = \alpha\gamma(t)\left(S_k(t) - \beta V_k(t)\right)\left(\sum_{l=1}^{K} I_l(t) + I_V(t)\right) - r^I E_k(t) \tag{6.5}$$

$$\frac{\mathrm{d}E'}{\mathrm{d}t} = \alpha\gamma(t)\left(S'(t) + \beta\sum_{\kappa=1}^{K} V_\kappa(t)\right)\left(\sum_{l=1}^{K} I_l(t) + I'(t)\right) - r^I E'(t) \tag{6.6}$$

$$\frac{\mathrm{d}I_k}{\mathrm{d}t} = r^I E_k(t) - r^d I_k(t) \tag{6.7}$$

$$\frac{\mathrm{d}I'}{\mathrm{d}t} = r^I E'(t) - r^d I'(t) \tag{6.8}$$

$$\frac{\mathrm{d}U_k}{\mathrm{d}t} = r_k^U(t) I_k(t) - r^D U_k(t) \tag{6.9}$$

$$\frac{\mathrm{d}H_k}{\mathrm{d}t} = r_k^H(t) I_k(t) - r^D H_k(t) \tag{6.10}$$

$$\frac{\mathrm{d}Q_k}{\mathrm{d}t} = r_k^Q(t) I_k(t) - r^D Q_k(t) \tag{6.11}$$

$$\frac{\mathrm{d}D_k}{\mathrm{d}t} = r^D\left(U_k(t) + H_k(t) + Q_k(t)\right) \tag{6.12}$$

$$\frac{\mathrm{d}M}{\mathrm{d}t} = r^d I'(t), \tag{6.13}$$

where:

- α is the nominal infection rate;
- $\gamma(t) : \mathbb{R}_+ \to \mathbb{R}_+$ is the governmental and societal response function (Figure 6-1);
- r^I, r^d, r^D, are the progression rate, the detection rate, and the death rate;
- $r_k^U(t)$, $r_k^H(t)$, and $r_k^Q(t)$ capture the detection, hospitalization and death rates, accounting for the probabilities of detection and hospitalization and the mortality rate (Equation (6.2)). Their dependency on t and k reflect disparities over time and across risk classes.

As noted earlier, the dynamics of exposure and infection depend on the total num-

ber of infected people (across risk classes and vaccinated/non-vaccinated people), as opposed to the number of infected people in a given risk class. DELPHI–V captures these interdependencies—indicated by the red rectangle in Figure 6-3 and the terms $\sum_{l=1}^{K} I_l(t) + I'(t)$ in Equations (6.3)–(6.5).

Given initial conditions, the ODE equations uniquely determine the evolution of this system over time—for a given vaccine allocation reflected in the variable $\mathbf{V}$. Next, we optimize the vaccine distribution strategy to minimize the overall impact of the pandemic—estimated by DELPHI–V.

6.3.3 Prescriptive DELPHI–V–OPT: Optimizing the vaccine distribution strategy

The DELPHI–V–OPT model takes as inputs epidemiological information (estimated from the DELPHI–V model), information on the vaccine (including vaccine effectiveness and vaccine budget), and demographic information in the United States (e.g., major cities, distance across counties, population per county). It optimizes the vaccine distribution strategy, including the location of mass vaccination sites and the subsequent allocation of vaccines. It is formulated as a tri-objective model, to minimize (i) the death toll of the pandemic, (ii) the number of exposed people in the termination period, and (iii) the distance between vaccination sites and population centers. The main public health objective is obviously death minimization, so the first objective component is heavily prioritized. However, just considering the number of deaths could result in a waste of vaccines near the end of the planning horizon, as individuals infected in the final periods would not have time to flow to the death state in the epidemiological model. Therefore, the second component of the objective minimizes the number of infections. The last component minimizes geographic disparities. In addition, the model incorporates other equity consideration by means of fairness constraints.

We discretize time to formulate the optimization model and retain tractability. This reduces to solving the system of ODE equations given in Equations (6.3)–(6.13) by a forward difference scheme. We denote by Δt the discretization unit (e.g., 1 day).

Formally, we define the following sets, input parameters, and decision variables.

Sets

$\mathcal{L}$ = set of population centers in the United States (e.g., counties), $\{1, \cdots, L\}$
$\mathcal{M}$ = set of United States regions (e.g., 50 states + District of Columbia), $\{1, \cdots, m\}$
$\mathcal{N}$ = set of candidate vaccination sites, $\{1, \cdots, n\}$
$\mathcal{N}_j$ = subset of candidate vaccination sites located in state $j \in \mathcal{M}$
$\mathcal{N}_l$ = subset of candidate vaccination sites located in the same state as county $l \in \mathcal{L}$
$\mathcal{K}$ = set of risk classes in the population (e.g., age groups), $\{1, \cdots, K\}$
$\mathcal{T}$ = set of discretized time periods (e.g., days), $\{1, \cdots, T\}$

Parameters

Pop_l = total population in center $l \in \mathcal{L}$
Δ_{li} = distance from population center $l \in \mathcal{L}$ to candidate vaccination site $i \in \mathcal{N}$
N = number of vaccination sites to be deployed across the country
B_t = number of vaccines available at time $t \in \mathcal{T}$
β = effectiveness of vaccines
r^I = progression rate of the disease (DELPHI parameter)
r^d = detection rate of the disease (DELPHI parameter)
r^D = death rate of the disease (DELPHI parameter)
r^U_{jkt} = transition rate from I to U, capturing mortality rate
in region $j \in \mathcal{M}$ at time $t \in \mathcal{T}$ for risk class $k \in \mathcal{K}$ (DELPHI parameter)
r^H_{jkt} = transition rate from I to H, capturing mortality rate
in region $j \in \mathcal{M}$ at time $t \in \mathcal{T}$ for risk class $k \in \mathcal{K}$ (DELPHI parameter)
r^Q_{jkt} = transition rate from I to Q, capturing mortality rate
in region $j \in \mathcal{M}$ at time $t \in \mathcal{T}$ for risk class $k \in \mathcal{K}$ (DELPHI parameter)
r^H_{jkt} = transition rate from I to H, capturing mortality rate
α_j = nominal infection rate in region $j \in \mathcal{M}$ (DELPHI parameter)
γ_{jt} = societal response in region $j \in \mathcal{M}$ at time $t \in \mathcal{T}$ (DELPHI parameter)

Note that the parameters r^U, r^H and r^Q are defined for each region, risk class and time period, reflecting underlying variations in mortality rates. In contrast, the parameters r^I, r^d and r^D are treated as uniform characteristics of the disease. In reality, these parameters may vary across risk classes; for instance, the serological estimates from [168] suggest different

prevalence of the disease across age groups. We test this hypothesis in our experiments, to verify the robustness of our results to the uniform infection rate assumption.

We also assume a single vaccine effectiveness value β. In theory, vaccine effectiveness might also vary across risk classes. More importantly, there are now several vaccines available, each with different clinical characteristics. Ideally, we could introduce an additional set to capture vaccine heterogeneity, and let the vaccine effectiveness vary across vaccine types. This approach however, may be somewhat impractical in practice, as it may be difficult to strategically allocate different vaccines to different populations based on vaccine effectiveness. For equity, we therefore assume conservatively that the mix of vaccines remains identical across vaccination sites. Under this restriction, the mix of vaccines can be reduced to a representative vaccine with average effectiveness.

Primary decision variables

$$x_i = \begin{cases} 1 & \text{if vaccination site } i \in \mathcal{N} \text{ is selected} \\ 0 & \text{otherwise} \end{cases}$$

C_{it} : number of vaccines distributed to site $i \in \mathcal{N}$ at time $t \in \mathcal{T}$

$$W_{li} = \begin{cases} 1 & \text{if population center } l \in \mathcal{L} \text{ is assigned to vaccination site } i \in \mathcal{N} \\ 0 & \text{otherwise} \end{cases}$$

$\bar{S}_{jkt}$: number of eligible people to region $j \in \mathcal{M}$ in risk class $k \in \mathcal{K}$ at time $t \in \mathcal{T}$

V_{jkt} : number of vaccines allocated to region $j \in \mathcal{M}$ in risk class $k \in \mathcal{K}$ at time $t \in \mathcal{T}$

To track the impact of vaccine allocation on the resulting dynamics of the pandemic, we create indirect variables, corresponding to all the states in the DELPHI–V model shown in Figure 6-3.

The vaccine distribution problem, referred to as $(\mathcal{P})$, is then formulated in Equations (6.14)–(6.39).

$$\min \quad \sum_{j=1}^{M}\sum_{k=1}^{K}\left(D_{jkT}+H_{jkT}+Q_{jkT}\right)+\lambda_E\sum_{j=1}^{M}\sum_{k=1}^{K}E_{jkT}+\lambda_D\sum_{l=1}^{L}\sum_{i=1}^{n}Pop_l\Delta_{li}W_{li} \tag{6.14}$$

$$\text{s.t.} \quad \sum_{i=1}^{n}x_i=N \tag{6.15}$$

$$\sum_{i\in\mathcal{N}_j}x_i\geq 1, \qquad \forall j\in\mathcal{M} \tag{6.16}$$

$$W_{li}\leq x_i, \qquad \forall l\in\mathcal{L},\ i\in\mathcal{N} \tag{6.17}$$

$$\sum_{i\in\mathcal{N}_l}W_{li}=1, \qquad \forall l\in\mathcal{L} \tag{6.18}$$

$$\sum_{i=1}^{n}C_{it}\leq B_t, \qquad \forall t\in\mathcal{T} \tag{6.19}$$

$$C_{it}\leq B_t x_i, \qquad \forall i\in\mathcal{N},\ t\in\mathcal{T} \tag{6.20}$$

$$\sum_{k=1}^{K}V_{jkt}\leq\sum_{i\in\mathcal{N}_j}C_{it}, \qquad \forall j\in\mathcal{M},\ t\in\mathcal{T} \tag{6.21}$$

$$\bar{S}_{j,k,t+1}\leq\bar{S}_{jkt}-(1-\beta)V_{jkt}-(S_{jkt}-S_{j,k,t+1}), \qquad \forall j\in\mathcal{M},k\in\mathcal{K},t\in\mathcal{T} \tag{6.22}$$

$$V_{jkt}\leq\bar{S}_{jkt}, \qquad \forall j\in\mathcal{M},k\in\mathcal{K},t\in\mathcal{T} \tag{6.23}$$

$$\left|C_{i,t+1}-C_{it}\right|\leq\theta_S C_{it}, \qquad \forall i\in\mathcal{N},t\in\mathcal{T} \tag{6.24}$$

$$\frac{\sum_{l\in\mathcal{L}_j}Pop_l}{\sum_{j'=1}^{m}\sum_{l\in\mathcal{L}_{j'}}Pop_l}N-\Theta_L\leq\sum_{i\in\mathcal{N}_j}x_i\leq\frac{\sum_{l\in\mathcal{L}_j}Pop_l}{\sum_{j'=1}^{m}\sum_{l\in\mathcal{L}_{j'}}Pop_l}N+\Theta_L, \qquad \forall j\in\mathcal{M} \tag{6.25}$$

$$\frac{B_t}{N(1+\theta_V)}x_i\leq C_{it}\leq\frac{B_t}{N}(1+\theta_V)x_i, \qquad \forall i\in\mathcal{N},\ t\in\mathcal{T} \tag{6.26}$$

$$\sum_{i\in\mathcal{N}_j}C_{it}\leq\left(\frac{\sum_{l\in\mathcal{L}_j}Pop_l}{\sum_{j'=1}^{m}\sum_{l\in\mathcal{L}_{j'}}Pop_l}+\theta_P\right)B_t, \qquad \forall j\in\mathcal{M},\ t\in\mathcal{T} \tag{6.27}$$

$$S_{j,k,t+1}\geq S_{jkt}-\beta V_{jkt}-\alpha_j\gamma jt\left(S_{jkt}-\beta V_{jkt}\right)\left(\sum_{\kappa=1}^{K}I_{j\kappa t}+I'_{jt}\right)\Delta t, \qquad \forall j\in\mathcal{M},k\in\mathcal{K},t\in\mathcal{T} \tag{6.28}$$

$$S'_{j,t+1}\geq S'_{jt}+\beta\sum_{\kappa=1}^{K}V_{j\kappa t}-\alpha_j\gamma_{jt}\left(S'_{jt}+\beta\sum_{\kappa=1}^{K}V_{j\kappa t}\right)\left(\sum_{\kappa=1}^{K}I_{j\kappa t}+I'_{jt}\right)\Delta t, \qquad \forall j\in\mathcal{M},t\in\mathcal{T} \tag{6.29}$$

$$E_{j,k,t+1}\geq E_{jkt}+\left(\alpha_j\gamma_{jt}\left(S_{jkt}-\beta V_{jkt}\right)\left(\sum_{\kappa=1}^{K}I_{j\kappa t}+I'_{jt}\right)-r^I E_{jkt}\right)\Delta t, \qquad \forall j\in\mathcal{M},\ k\in\mathcal{K},\ t\in\mathcal{T} \tag{6.30}$$

$$E'_{j,t+1}\geq E'_{jt}+\left(\alpha_j\gamma_{jt}\left(S'_{jt}+\beta\sum_{\kappa=1}^{K}V_{j\kappa t}\right)\left(\sum_{\kappa=1}^{K}I_{j\kappa t}+I'_{jt}\right)-r^I E'_{jt}\right)\Delta t, \qquad \forall j\in\mathcal{M},t\in\mathcal{T} \tag{6.31}$$

$$I_{j,k,t+1}\geq I_{jkt}+\left(r^I E_{jkt}-r^d I_{jkt}\right)\Delta t, \qquad \forall j\in\mathcal{M},\ k\in\mathcal{K},\ t\in\mathcal{T} \tag{6.32}$$

$$I'_{j,t+1}\geq I'_{jt}+\left(r^I E'_{jt}-r^d I'_{jt}\right)\Delta t, \qquad \forall j\in\mathcal{M},\ t\in\mathcal{T} \tag{6.33}$$

$$U_{j,k,t+1}\geq U_{jkt}+\left(r^U_{jkt}I_{jkt}-r^D U_{jkt}\right)\Delta t, \qquad \forall j\in\mathcal{M},\ k\in\mathcal{K},\ t\in\mathcal{T} \tag{6.34}$$

$$H_{j,k,t+1}\geq H_{jkt}+\left(r^H_{jkt}I_{jkt}-r^D H_{jkt}\right)\Delta t, \qquad \forall j\in\mathcal{M},\ k\in\mathcal{K},\ t\in\mathcal{T} \tag{6.35}$$

$$Q_{j,k,t+1}\geq Q_{jkt}+\left(r^Q_{jkt}I_{jkt}-r^D Q_{jkt}\right)\Delta t, \qquad \forall j\in\mathcal{M},\ k\in\mathcal{K},\ t\in\mathcal{T} \tag{6.36}$$

$$D_{j,k,t+1} = D_{jkt} + r^D \left(U_{jkt} + H_{jkt} + Q_{jkt}\right) \Delta t, \qquad \forall j \in \mathcal{M},\ k \in \mathcal{K},\ t \in \mathcal{T} \quad (6.37)$$

$$M_{j,t+1} = M_{jkt} + r^d I'_{jt}, \qquad \forall j \in \mathcal{M},\ t \in \mathcal{T} \quad (6.38)$$

$$\mathbf{x}, \mathbf{W} \text{ binary},\ \mathbf{C}, \mathbf{V}, \mathbf{S}, \mathbf{S}', \bar{\mathbf{S}}, \mathbf{E}, \mathbf{E}', \mathbf{I}, \mathbf{I}', \mathbf{U}, \mathbf{H}, \mathbf{Q}, \mathbf{D}, \mathbf{M} \geq \mathbf{0}. \quad (6.39)$$

Equation (6.14) formalizes the three objectives of the model. The first term corresponds to our primary objective of minimizing the number of deaths over the planning horizon, across all regions and risk classes. This number includes people in the absorbing state D, as well as the transient states H and Q (we ignore undetected deaths). The next terms minimize, as lower-priority objectives, the number of exposed people at the end of the horizon and the distance to the vaccination sites. The hyperparameters λ_E and λ_D are set to small values to prioritize the death-minimization objective.

Next, the constraints capture practical considerations surrounding vaccine distribution:

- **Number of vaccination sites**: We impose a total budget of N vaccination sites (Equation (6.15)). For obvious reasons, there needs to be at least one site in every state (Equation (6.16)). We consider $N = 100$ in our experiments, which leaves flexibility to strategically deploy 49 sites.

- **Assignment**: Equation (6.17) ensures that people get assigned to vaccination sites that have been selected. Equation (6.18) assigns each population center to exactly one site, in the same state. These assignment constraints are used to compute the distance term in the objective function.

- **Inter-regional vaccine capacity**: Due to restrictions in vaccine manufacturing and distribution networks, a limited number of vaccines can be allocated in each time period. Equation (6.19) ensures that the total number of vaccines allocated lies within the available budget in each period.

- **Consistency**: Equation (6.20) ensures that vaccines only get distributed to selected sites. Similarly, Equation (6.21) ensures that the number of people vaccinated in each state (across risk classes) does not exceed the number of vaccines allocated that state. This constraint involves two assumptions. A first, conservative assumption is that

people can only get vaccinated in the state that they live in, which is required in practice for traceability purposes. Another, optimistic assumption is that the vaccine allocation constraint applies to each state, as opposed to each vaccination site. In other words, the model assumes vaccines can be reallocated between vaccination sites within a state, thus maintaining a degree of freedom in intra-state vaccine distribution.

– **Eligibility**: We prevent people from being vaccinated twice: a patient who has been vaccinated but remains susceptible cannot be vaccinated again. Equation (6.22) defines the number eligible people as the previous number of eligible people minus the number of people for whom the vaccine was effective and the number of people who got exposed to the disease. Equation (6.22) then ensures that the number of vaccinated people lies below the number of eligible people.

– **Smoothness**: Large fluctuations in the number of vaccines allocated to each region from day to day would likely cause problems from a supply chain management perspective—both to deliver and to administer the vaccines. Equation (6.24) ensures that such fluctuations remain minimal. The hyperparameter θ_S controls the trade-off between efficiency and smoothness.

– **Fairness**: To be politically and socially viable, vaccine distribution must not neglect any region, even if it is not a virus "hot spot". This also enhances the robustness of the solution, given that inter-regional transmission can occur in practice. Equation (6.25) promotes inter-state fairness at the strategic level, by ensuring that the fraction of vaccination sites in each state does not deviate too much from its population share. Equation (6.26) promotes inter-site fairness, by ensuring that vaccine distribution across sites does not deviate too much from uniform distribution. Finally, Equation (6.27) promotes inter-state fairness at the tactical level, by ensuring that no state receives a fraction of vaccines that exceeds its population share by a wide margin. The hyperparameters Θ_L, θ_V and θ_P control the trade-off between efficiency and fairness. As the results will show, even tight fairness constraints leave critical flexibility when locating vaccination sites and allocating vaccines.

– **DELPHI–V dynamics**: Equations (6.28)–(6.38) capture the dynamics of the DELPHI–V model in a discretized time space (Equations (6.3)–(6.13)).

– **Domain of definition**: Equation (6.39) defines the domain of each variable.

Model Structure

Problem ($\mathcal{P}$) is a non-linear program, due to the bilinear terms in Equations (6.28)–(6.31), which reflect the fact that the number of new infections result from the interactions between susceptible and infected populations—a key characteristic of all SEIR-based compartmental models. These bilinear terms result in non-convex constraints, thus in a highly challenging optimization model.

The latest Gurobi 9.0 release includes a solver for non-convex quadratic problems [84]. Yet, general-purpose technologies are limited to small-scale instances. In our setting, Problem ($\mathcal{P}$) includes $2m(K+1)T$ non-convex constraints each involving $2(K+1)$ bilinear terms, for a total of $4mT(K+1)^2$ bilinear terms. A realistically-sized problem with $M = 51$ (50 US states plus Washington, D.C.), $K = 6$ (6 age groups) and $T = 90$ (a 3 month planning horizon with daily discretization) would result in nearly 900,000 bilinear terms. Problem ($\mathcal{P}$) remains intractable with existing commercial solvers, motivating the development of a tailored algorithm.

6.4 Solution algorithm

We propose an iterative coordinate descent algorithm to solve Problem ($\mathcal{P}$) in short computational times—consistent with practical requirements. We describe the algorithm in this section. We also present three baselines replicating reasonable strategies that could be implemented in the absence of our data-driven optimization model. These baselines are used for two purposes: (i) to provide an initial feasible solution in the coordinate descent algorithm, and (ii) as benchmarks to evaluate the benefits of the data-driven optimization approach proposed in this paper.

6.4.1 Algorithm design

Our algorithm relies on two key observations: 1) aside from Equations (6.28)–(6.31), the objective function and all other constraints in $(\mathcal{P})$ are linear, and 2) given a fixed vaccine distribution strategy, the discretized DELPHI–V model can be solved efficiently. Therefore, we proceed by coordinate descent, alternating between two modules: one that optimizes the vaccination distribution strategy given the infection dynamics, and one that simulates the bilinear dynamics of the pandemic for a given vaccination distribution strategy. The optimization part reduces to a linear program, which can be solved very efficiently. Using the resultant vaccine distribution, the simulation part re-estimates the infected population under bilinear dynamics, using a forward discretization scheme. Specifically, the two modules are defined as follows:

1. **Simulate**: Based on a vaccine allocation solution $\mathbf{V}$, we compute the DELPHI–V dynamics from $t = 0$ to $t = T$ (Section 6.3.2) by solving the ODE system (Equations (6.3)–(6.13)) using a forward difference scheme in a discretized time space. This terminates in $\mathcal{O}(MKT)$ operations. We denote the total infected population (across all risk classes and vaccinated people) in region j at time $t \in \mathcal{T}$ by $\widehat{I}_{jt} = \sum_{k=1}^{K} I_{jkt} + I'_{jt}$. We refer to this procedure as $\texttt{Simulate}\,(\mathbf{x}, \mathbf{C}, \mathbf{W}, \mathbf{V})$.

2. **Optimize**: Given the infectious population estimates $\widehat{\mathbf{I}}$, we can approximate Equation (6.28)–(6.31) by the following linear constraints. The problem can then be efficiently solved as a linear programming model. We refer to this module as $\texttt{Optimize}\left(\widehat{\mathbf{I}}\right)$.

$$
\begin{aligned}
&S_{j,k,t+1} \geq S_{jkt} - \beta V_{jkt} - \alpha_j \gamma_{jt} \left(S_{jkt} - \beta V_{jkt}\right) \widehat{I}_{jt} \Delta t, && \forall j \in \mathcal{M}, k \in \mathcal{K}, t \in \mathcal{T} \\
&S'_{j,t+1} \geq S'_{jt} + \beta \sum_{\kappa=1}^{K} V_{j\kappa t} - \alpha_j \gamma_{jt} \left(S'_{jt} + \beta \sum_{\kappa=1}^{K} V_{j\kappa t} \right) \widehat{I}_{jt} \Delta t, && \forall j \in \mathcal{M}, t \in \mathcal{T} \\
&E_{j,k,t+1} \geq E_{jkt} + \left(\alpha_j \gamma_{jt} \left(S_{jkt} - \beta V_{jkt}\right) \widehat{I}_{jt} - r^I E_{jkt} \right) \Delta t, && \forall j \in \mathcal{M},\ k \in \mathcal{K},\ t \in \mathcal{T} \\
&E'_{j,t+1} \geq E'_{jt} + \left(\alpha_j \gamma_{jt} \left(S'_{jt} + \beta \sum_{\kappa=1}^{K} V_{j\kappa t} \right) \widehat{I}_{jt} - r^I E'_{jt} \right) \Delta t, && \forall j \in \mathcal{M}, t \in \mathcal{T}
\end{aligned}
$$

We iterate between the `Simulate` and `Optimize` modules, until convergence. Specifically,

the algorithm terminates when the variation in the objective function value remains minimal from one iteration to the next. The pseudocode summarizing this approach is presented in Algorithm 7. We turn next to the generation of an initial feasible solution.

Algorithm 7 Coordinate descent algorithm for DELPHI–V–OPT (Problem $(\mathcal{P})$).

Input: Prescriptive DELPHI–V–OPT data; termination tolerance ε

Initialization: $i \leftarrow 0$, $\mathbf{x}^{(i)} \leftarrow \mathbf{0}$, , $\mathbf{C}^{(i)} \leftarrow \mathbf{0}$, , $\mathbf{W}^{(i)} \leftarrow \mathbf{0}$, , $\mathbf{V}^{i} \leftarrow \mathbf{0}$, $\mathbf{I}^{i} \leftarrow \mathbf{0}$, $Z^{i} \leftarrow \infty$

$i \leftarrow i + 1$

$\left(\mathbf{x}^{(i)}, \mathbf{C}^{(i)}, \mathbf{W}^{(i)}, \mathbf{V}^{(i)}, \mathbf{I}^{(i)}, Z^{(i)}\right) \leftarrow$ `GenerateFeasibleSolution`

while $\left|Z^{(i)} - Z^{(i-1)}\right| / Z^{(i-1)} > \varepsilon'$ **do**

$i \leftarrow i + 1$

Run `Simulate` $\left(\mathbf{x}^{(i-1)}, \mathbf{C}^{(i-1)}, \mathbf{W}^{(i-1)}, \mathbf{V}^{(i-1)}\right)$.

Update $\widehat{\mathbf{I}}^{(i)} \leftarrow \widehat{\mathbf{I}}$, where $\widehat{\mathbf{I}}$ is the output of `Simulate` $\left(\mathbf{x}^{(i-1)}, \mathbf{C}^{(i-1)}, \mathbf{W}^{(i-1)}, \mathbf{V}^{(i-1)}\right)$.

Run `Optimize` $\left(\widehat{\mathbf{I}}^{(i)}\right)$.

Update $\left(\mathbf{x}^{(i)}, \mathbf{C}^{(i)}, \mathbf{W}^{(i)}, \mathbf{V}^{(i)}\right) \leftarrow (\mathbf{x}, \mathbf{C}, \mathbf{W}, \mathbf{V})$, where $(\mathbf{x}, \mathbf{C}, \mathbf{W}, \mathbf{V})$ is the output of `Optimize` $\left(\widehat{\mathbf{I}}^{(i)}\right)$.

Update the objective function:

$$Z^{(i)} \leftarrow \sum_{j=1}^{M}\sum_{k=1}^{K}(D_{jkT} + H_{jkT} + Q_{jkT}) + \lambda_E \sum_{j=1}^{M}\sum_{k=1}^{K} E_{jkT} + \lambda_D \sum_{l=1}^{L}\sum_{i=1}^{n} Pop_l \Delta_{li} W_{li}$$

end while

Output: Vaccine distribution strategy $\left(\mathbf{x}^{(i)}, \mathbf{C}^{(i)}, \mathbf{W}^{(i)}, \mathbf{V}^{(i)}\right)$

6.4.2 Baselines

We propose three simple and interpretable baselines for generating a feasible solution to $(\mathcal{P})$. By design, these baselines are heuristics that solely rely on the inputs of the optimization models, as opposed to requiring the full model and algorithm developed in this paper.

Top-cities baseline: This approach prioritizes cities based on population. Specifically, it deploys vaccination sites in the most populous cities, while accounting for the constraint that each state must have at least one center (Equation (6.16)). Subsequently, it allocates an equal fraction of the daily vaccine budget to each vaccination site: we fix the variables $\mathbf{x}$ and $\mathbf{C}$, run the model to optimize vaccine allocation within each state, and estimate the resulting number of deaths. This baseline corresponds to a city-level approach based on

demographic information alone.

Population-based baseline: Under this approach, the number of vaccination sites deployed in each state is based on the state's population share. This is formulated as follows, where y_j is a decision variable denoting the number of vaccination sites in state j.

$$\min \quad \sum_{j=1}^{m} \left| \frac{\sum_{l \in \mathcal{L}_j} Pop_l}{\sum_{j'=1}^{m} \sum_{l \in \mathcal{L}_{j'}} Pop_l} N - y_j \right|, \text{ s.t. } \sum_{j=1}^{m} y_j = 100,\ y_j \geq 1, \forall j \in \mathcal{M},\ \mathbf{y} \text{ integer.}$$

We then solve DELPHI–V–OPT, while fixing the aggregate number of vaccination sites per state, i.e., $\sum_{i \in \mathcal{N}_j} x_i = y_j$, and assuming equal allocation of vaccines across vaccination sites, i.e., $C_{it} = \frac{1}{N} B_t,\ \forall t \in \mathcal{T}$. The model allocates vaccines within each state and estimates the number of deaths. This baseline corresponds to a state-level approach based on demographic information alone.

Case-based baseline: Under this approach, the number of vaccination sites deployed in each state is based on the number of COVID-19 cases at the beginning of the planning horizon. This is formulated as follows, where y_j is a decision variable denoting the number of vaccination sites in state j and $Cases_j$ denotes the case count in state j.

$$\min \quad \sum_{j=1}^{m} \left| \frac{Cases_j}{\sum_{j'=1}^{m} Cases_{j'}} N - y_j \right|, \text{ s.t. } \sum_{j=1}^{m} y_j = 100,\ y_j \geq 1, \forall j \in \mathcal{M},\ \mathbf{y} \text{ integer.}$$

We then proceed as with the population-based baseline, by fixing the number of vaccination sites per state, assuming equal vaccine allocation across sites, and re-solving the model. This baseline corresponds to a state-level approach based on epidemiological information alone.

By design, these baselines satisfy all constraints of Problem ($\mathcal{P}$), and thus provide valid initializations into our coordinate descent algorithm. They also provide sensible and equitable benchmarks based on readily-available demographic information (e.g., census data) and epidemiological information (e.g., case counts), hence easily implementable. Comparisons between our optimized solution and these benchmarks thus estimate the benefits of vaccine distribution optimization.

6.5 Experimental setup

We implement the proposed model and algorithm in the United States. We select $N = 100$ vaccination sites out of the 500 most populous cities in the United States as candidate locations (set $\mathcal{N}$) We define the set $\mathcal{M}$ as 51 "states" (the 50 states plus the District of Columbia) and the set $\mathcal{L}$ as the 3,006 counties. We define six risk classes based on six relatively coarse age groups: 0-9 years, 10-49 years, 50-59 years, 60-69 years, 70-79 years, and 80 years and above. These simplified risk classes facilitate the practical implementation of the solution while capturing broad trends in mortality rates. We define the time horizon $\mathcal{T}$ as the three-month period from February to April 2021, consistently with the ongoing planning horizon of the US federal government.

6.5.1 Data sources

We calibrate the model using multiple data sources (Figure 6-4). First, we estimate the parameters of the DELPHI model (without vaccinations) independently for each state, using historical data on cases and deaths from [136]. We obtain a granular population breakdown by age for each state from [167]. We then run DELPHI (still, without vaccinations) to derive the initial susceptible, exposed and infected populations (on January 30, 2021), which we distribute among the risk classes proportionally to their size.

The next input is an estimate of the mortality rate per region, risk class and time period. We make use of the data from [170], which report the total number of confirmed COVID-19 cases, hospitalizations and deaths by age group in the United States until the end of January 2021. In contrast, the DELPHI model fits a time-dependent mortality rate within each region. To the best of our knowledge, there exists no other data source for nationwide cases and deaths by age group. This leaves a discrepancy between the time-independent estimates at the risk class level from the CDC, and the time-varying estimates at the region-level from DELPHI. To reconcile these data, we employ an optimization procedure that interpolates the mortality rate per region, risk class and time period. We present this approach in the next section.

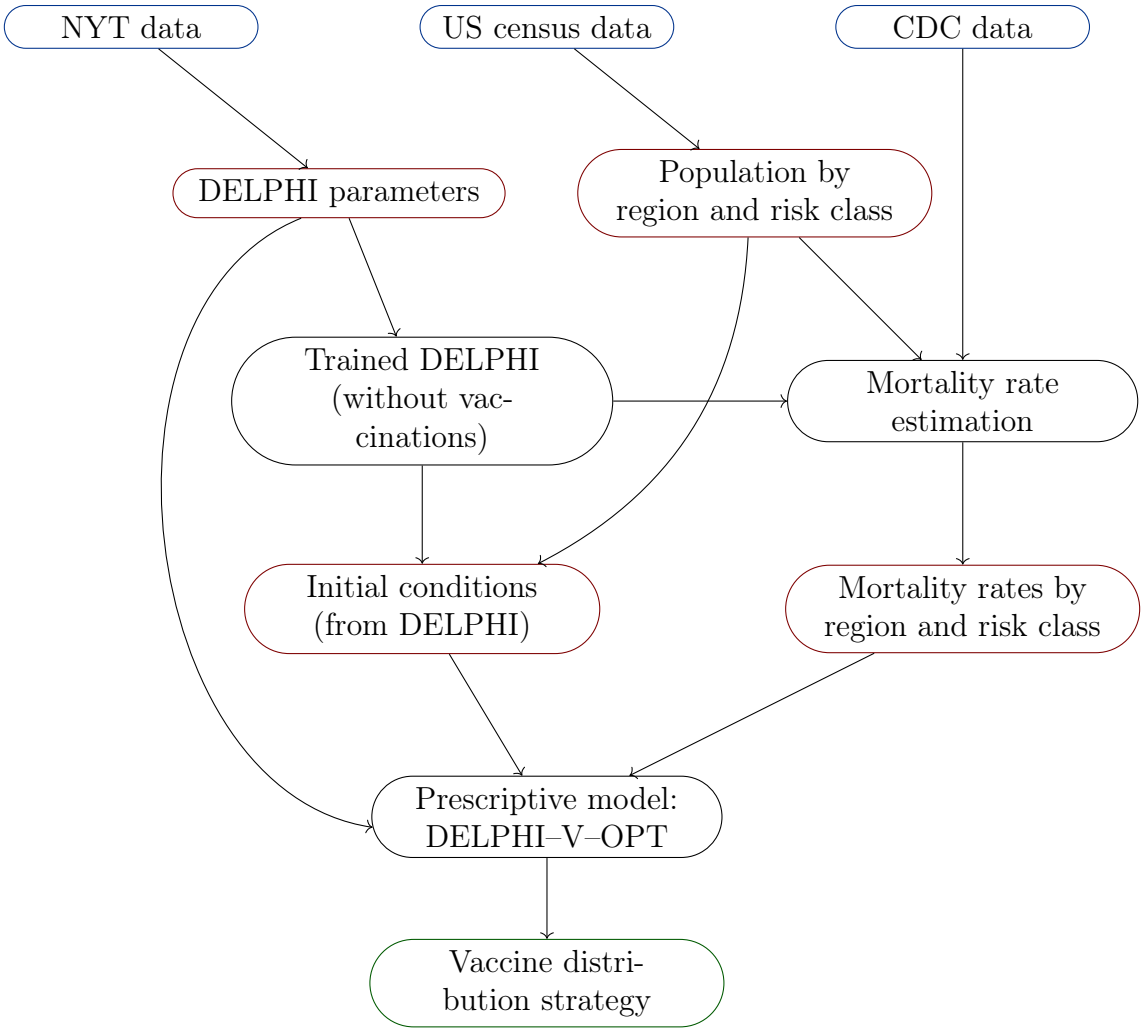


Figure 6-4: Simulation environment: raw data (blue), processed data (red), models (black) and outputs (green).

6.5.2 Mortality rate estimation

Our procedure to estimate mortality rates starts from two sets of inputs:

- **DELPHI predictions:** Let $\widehat{X}_{jt}$ denote the estimated number of new detected cases in region $j \in \mathcal{M}$ and time period $t \in \mathcal{T}$. Let $\widehat{D}^{d}_{j,t+\tau_j}$ denote the number of deaths, where τ_j is the median death time after detection in region j. These quantities are aggregated across risk classes.

- **CDC data:** Let X^{CDC}_{k} and D^{CDC}_{k} denote the number of cases and deaths for risk class $k \in \mathcal{K}$. These quantities are aggregated across regions and time periods.

We define the reference mortality rate $\overline{m}_{jk}$ of risk class $k \in \mathcal{K}$ based in region j as follows:

$$\overline{m}_{jk} = \frac{D_k^{\text{CDC}}}{X_k^{\text{CDC}}} \left(\frac{\sum_{t=1}^{T} \widehat{D}_{j,t+\tau_j}^{d}}{\sum_{t=1}^{T} \widehat{X}_t} \right) \left(\frac{\sum_{l=1}^{K} X_l^{\text{CDC}}}{\sum_{l=1}^{K} D_l^{\text{CDC}}} \right) \tag{6.40}$$

By design, this expression preserves the ratio of mortality rates between different risk classes from the CDC data, while correcting the mean reference mortality rate in each region across the planning horizon to be in line with the DELPHI projections.

We then estimate the mortality rate for each region $j \in \mathcal{M}$, risk class $k \in \mathcal{K}$, and time period $t \in \mathcal{T}$, denoted by m_{jkt}. We also introduce additional decision variables X_{jkt} and $D_{j,k,t+\tau_j}$, reflecting the number of detected cases and the number of future deaths in region j assigned to risk class $k \in \mathcal{K}$ and time period $t \in \mathcal{T}$. Given that the fitting procedure is done separately in each region $j \in \mathcal{M}$, we decouple the problem at the region level—thereby considerably reducing the size of each problem instance. Specifically, we formulate the optimization problem given in Equations (6.41)–(6.48).

$$\text{for all } j \in \mathcal{M}\text{: } \min \sum_{t=1}^{T} \sum_{k=1}^{K} \left[\left(\frac{m_{jkt} - \overline{m}_{jk}}{\overline{m}_{jk}} \right)^2 + \lambda_M \left| \frac{X_{jkt} - p_{jk} \widehat{X}_{jt}}{p_{jk} \widehat{X}_{jt}} \right| \right] \tag{6.41}$$

$$\text{s.t. } \sum_{k=1}^{K} D_{j,k,t+\tau_j} = \widehat{D}_{j,t+\tau_j}, \qquad \forall t \in \mathcal{T} \tag{6.42}$$

$$\sum_{k=1}^{K} X_{jkt} = \widehat{X}_{jt}, \qquad \forall t \in \mathcal{T} \tag{6.43}$$

$$m_{jkt} \cdot X_{jkt} = D_{j,k,t+\tau_j}, \qquad \forall k \in \mathcal{K}, \forall t \in \mathcal{T}, \tag{6.44}$$

$$m_{jkt} \geq m_{j,k,t+1}, \qquad \forall k \in \mathcal{K}, \forall t \in \mathcal{T} \tag{6.45}$$

$$m_{jkt} \geq m_{jlt}, \qquad \forall k, l \in \mathcal{K} : \overline{m}_k \geq \overline{m}_l, \tag{6.46}$$

$$m_{jkt} \leq 1, \qquad \forall k \in \mathcal{K}, \forall t \in \mathcal{T} \tag{6.47}$$

$$\mathbf{m}, \mathbf{C}, \mathbf{D} \geq \mathbf{0}. \tag{6.48}$$

The first term in Equation (6.41) minimizes the squared relative error between the mortality rate estimates and their reference values (Equation (6.40)). The second term is a regularization penalty that minimizes deviations between the proportion of detected cases and the proportion p_{jk} of the population, in each risk class. The parameter λ_M trades off these two objectives (we use $\lambda_M = 0.1$ in our experiments). Equations (6.42)–(6.43) ensure consistency

with the DELPHI predictions. Equation (6.44) defines the mortality rate as the ratio between the number of deaths and cases. Equations (6.45)–(6.46) ensure that mortality rates are decreasing over time and monotonic with risk classes. Finally, Equations (6.47)–(6.48) define the domain of the variables.

Note that the problem is non-linear due to the bilinear term in Equation (6.44). Yet, thanks to the decoupling at the region level, we can solve the problem using the quadratic solver in Gurobi 9.0 [84], which addresses non-convexities using branching and cutting planes algorithms. In practice, a solution within a 1% optimality gap is generally obtained within minutes.

The output of this algorithm is an estimate of the mortality rate at the level of each state and each risk class, in each time period. We report aggregated statistics in Table 6.1.

Month	0-9	10-49	50-59	60-69	70-79	80+
February	0.008%	0.119%	0.882%	2.271%	6.101%	15.027%
March	0.007%	0.116%	0.859%	2.212%	5.942%	14.636%
April	0.007%	0.114%	0.844%	2.171%	5.832%	14.365%

Table 6.1: Calibrated monthly mortality rates, averaged by risk class and over all states.

6.5.3 Implementation details

Our DELPHI–V–OPT model relies on a forward difference scheme to simulate the dynamics of the pandemic, for any vaccine allocation. If the discretization is too coarse, the algorithm will introduce truncation errors. If, however, the discretization is too granular, computational times will be prohibitively long. After extensive experimentations, we found $\Delta t = 1$ day to yield the best compromise. It is also practical choice as it yields a day-by-day plan.

The coordinate descent scheme terminates when the change in the objective function value lies below a pre-determined threshold. We set the termination tolerance to $\varepsilon = 0.1\%$.

Regarding DELPHI-V–OPT, we vary the hyperparameters λ_E, λ_V, θ_S, Θ_L, θ_V and θ_P to balance efficiency with practical and fairness considerations, and perform sensitivity analyses on these parameters. We consider a baseline vaccine effectiveness of 90% (in line with the values from the first vaccine approvals) and a baseline budget of 1 million vaccines per

day (which can be viewed as 1 million single-dose vaccines, 2 million double-dose vaccines or a combination thereof). Given the strong underlying uncertainty, we perform sensitivity analyses to demonstrate the robustness of the benefits of our optimization outputs to vaccine effectiveness and vaccine budget.

All optimization models are implemented in Gurobi 9.0, with 2.3GHz processor and 4 cores. We use a barrier method to solve each linear program, with a barrier convergence tolerance of 10^{-6}.

6.6 Experimental results

We now present results obtained with the modeling and algorithmic framework developed in this paper. Our main focus is on the location of the 100 vaccination sites—decisions that have to be made immediately—as opposed to vaccine allocation—decisions that can be revisited continuously over time as more information becomes available. We first evaluate the benefits of the optimized solution against baseline approaches, and then establish the robustness of these benefits.

6.6.1 Benefits of optimizing the vaccine distribution strategy

Figure 6-5 shows the heatmap of vaccination sites across all 51 states (the exact list of proposed locations is reported in the appendix). Recall that, by design, the optimized solution deploys at least one vaccination site per state (Equation (6.16)). In addition, one of the fairness constraints (Equation (6.25) anchors the number of vaccination sites per state to its population share. Other than that, the vaccination sites are deployed strategically to curb the spread of the pandemic. As a result, the optimized number of vaccination sites varies significantly per state, as a function of underlying demographic and epidemiological factors. For instance, the four largest states by population (California, Texas, Florida and New York) receive the most vaccination sites (6 to 10 each). In contrast, the smaller states receive only one vaccination site. Ultimately, the heatmap suggests that the optimized solution targets large population centers and some of the hot spots of the pandemic, while ensuring equity nationwide.

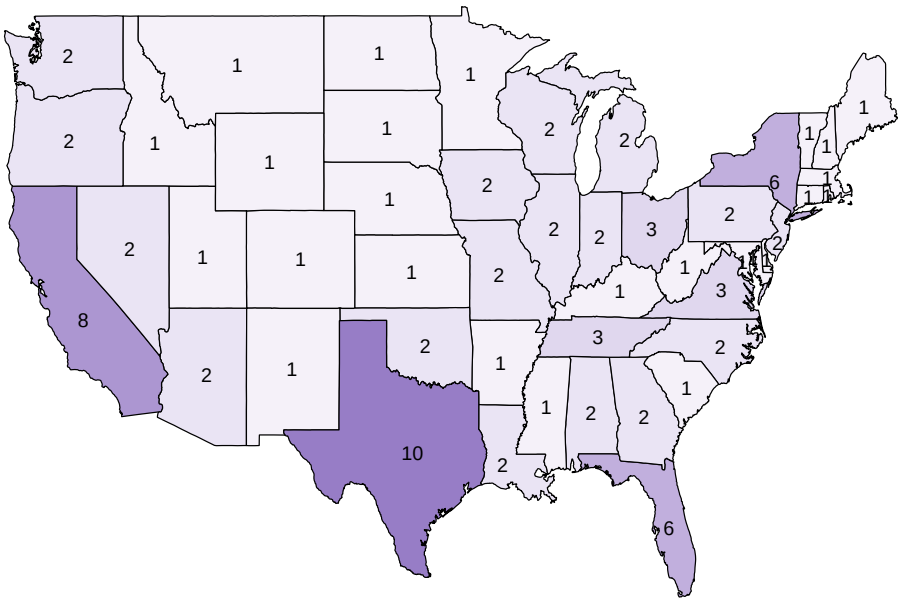


Figure 6-5: Number of centers per state in the proposed solution.

Then, could we have achieved a similar solution with some of our benchmarks (Section 6.4.2), which determine the locations of the vaccination sites based on demographic and epidemiological data but do not make use of our epidemiological and optimization models? To investigate this question, we evaluate the dynamics of the pandemic under each of the three baselines. When it comes to the optimization, we derive three solutions: (i) an "optimized locations" solution, which optimizes the site locations decisions freely (without fairness constraints) but then allocates vaccines uniformly across the 100 selected vaccination sites; (ii) a "fully optimized" solution, which optimizes the site locations decisions and vaccine allocation decisions freely (without fairness constraints); and (iii) a "proposed" solution, which optimizes the site locations decisions and vaccine allocation decisions with fairness constraints. For each solution, Table 6.2 reports the number of lives saved by the vaccination campaign, as compared to a solution without vaccinations.

Let us start with the main observation: the optimized solutions provides significant benefits, as compared to all benchmarks. Comparing the "optimized locations" solution to the top-cities benchmark, we find that optimizing locations alone can save around 4,500 lives over the three-month period under consideration, enhancing the impact of the vaccination

Table 6.2: Comparison of optimized solutions and benchmarks.

Method	Solution	Site locations	Vaccine distribution	Saved lives
Benchmarks	Top-cities	Most populous cities	Uniform across centers	19,045 (base)
	Population	Pro-rata population	Uniform across centers	19,709 (+3.5%)
	Cases	Pro-rata active cases	Uniform across centers	21,037 (+10.5%)
Optimization	Locations	Optimized: minimizes deaths	Uniform across centers	23,622 (+24.0%)
	Full	Optimized: minimizes deaths	Optimized: minimizes deaths	25,615 (+34.5%)
	Proposed	Optimized: minimizes deaths Fair: inter-state equity	Optimized: minimizes deaths Fair: inter-center equity	23,000 (+20.8%)

campaign by 24%. Moving to the "fully optimized" solution, we note that jointly optimizing locations and vaccine allocation achieves further benefits, by saving an extra 2,000 lives and increasing the benefits over the benchmark to 35%. These results underscore the potential of the proposed optimization approach, which leverages available vaccines strategically to target the regions that need them most. Another observation is that determining the locations of vaccination sites does not achieve all potential benefits of the vaccination campaign, underscoring the role of downstream vaccine allocation as an extra lever to combat the pandemic.

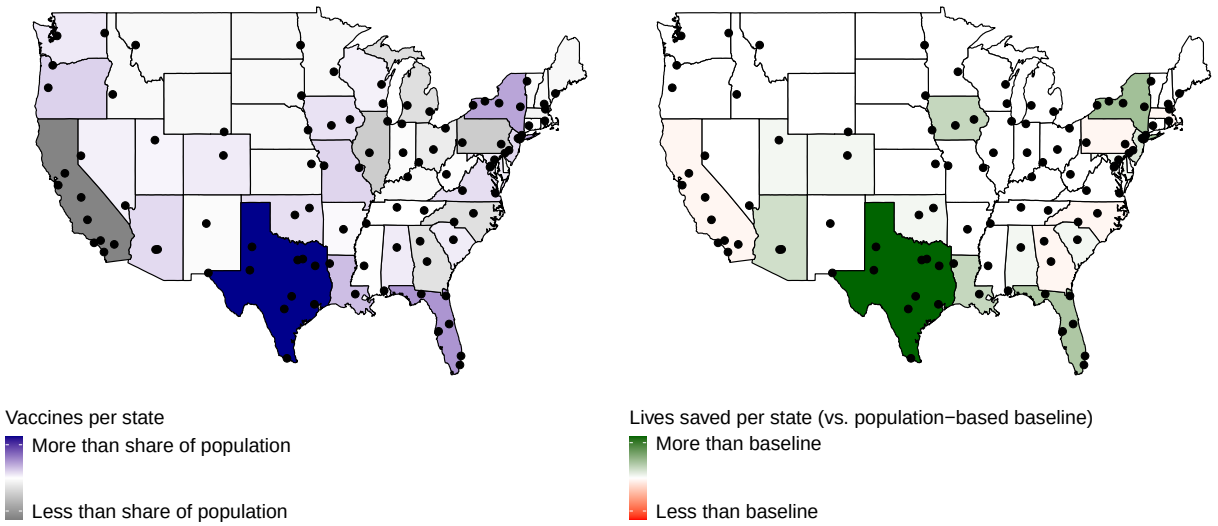
A downside of these two solutions ("locations optimized" and "fully optimized"), however, is that they can result in very inequitable outcomes between states and between vaccination sites. The last ("proposed") solution circumvents this challenge by imposing all fairness constraints (Equations (6.25)–(6.27)), and choosing tight values for the corresponding hyperparameters Θ_L, θ_V and θ_P. As such, the proposed solution ensures fairness across various dimensions (site locations, vaccine allocation across states, and vaccine allocation across sites). Remarkably, even when constraining the optimization as such, the resulting proposed solution still results in 20% benefits, as compared to the top-cities benchmark—saving 4,000 deaths over the three-month horizon.

In comparison, the benchmarks cannot reach the same impact of vaccinations. The top-cities and population-based benchmarks achieve similar outcomes, with 19,000–20,000 lives saved as compared to a no-vaccination baseline. The case-based baseline performs better, by saving 21,000 lives (10% more than the top-cities baseline). Yet, these numbers remain significantly lower than those achieved with our solution. The top-cities and population-based baselines, by relying on demographic information exclusively, fail account for disparities in the severity of the across the country. The case-based baseline captures the status of the pandemic, but not its dynamics, thus treating similarly a state where the pandemic is already waning and a state where it is rising fast—although vaccinations have a stronger impact in the latter state than the former.

These results underscore the main takeaways from this paper: optimization provides significant benefits, as compared to simple benchmarks based on readily-available information at the outset of the vaccination campaign. Instead, the optimized approaches design vaccine distribution strategies based on both demographics and epidemiological dynamics. The edge of optimization can be significant, by saving an extra 20% of lives with the same vaccine capacity. Stated differently, under the proposed optimization approach, each vaccine is effectively "worth" 1.2 vaccines.

A natural question to ask is whether optimization induces strong geographic disparities, by merely shifting the benefits of the vaccines from one state to another. To explore this question, Figure 6-6 plots the number of vaccines distributed to each state, as compared to its population share (Figure 6-6a) and the number of lives saved, as compared to the top-cities benchmark (Figure 6-6b).

Recall that the optimization imposes fairness constraints in the vaccine allocation (Equations
(6.25)–(6.27)); yet, the remaining flexibility can be used strategically to target the populations where vaccines can have the strongest impact. In fact, as Figure 6-6a shows, vaccines do not get distributed proportionally to each state's population. For instance, states like Texas, Florida and New York receive a higher share of vaccines, whereas states like California, Pennsylvania and Illinois receive a lower share. This is expected, given the significantly higher number of lives saved under the optimized ("proposed") solution as compared to the



(a) Vaccines allocation

(b) Lives saved

Figure 6-6: Vaccine allocation (vs. population share) and lives saved (vs. population-based baseline) per state.

population-based benchmark (Table 6.2).

However, Figure 6-6b shows that these disparities in vaccine allocation do not result in sharp disparities in public health outcomes. Specifically, the optimized solution saves hundreds of extra lives (as compared to the population-based benchmark) in seven states with very different epidemiological profiles. Texas benefits the most from optimization (with an additional 1,450 lives saved), followed by New York (440), Florida (380) and Iowa (290). At the same time, the optimized solution does not increase the death toll in any state by more than 100. Pennsylvania is the most negatively impacted state, with an estimated 85 additional deaths—well within the margin of error of DELPHI–V. Overall, the proposed optimization approach can thus distribute vaccine capacities to combat the pandemic in some critical states without hurting others.

The obvious question then is: how does the proposed solution compare to the vaccination plan that was applied in practice? Unfortunately, it is hard to perform a complete apples-to-apples comparison. Indeed, the deployment of vaccination site was conducted in a more ad hoc fashion in practice than modeled in this paper, based on local capacity, heterogeneity across "big" and "small" sites, etc. Moreover, since this paper was written, the Federal government has updated its plan, so that the Federal facilities take a supporting role while

delegating most authority to state-driven vaccination efforts. Therefore, the full plan was not implemented and instead FEMA has opened just 24 facilities in 13 states, as compared to the original 100 planned ones. We provide details in Table E.2 in the Appendix.

Nonetheless, we provide two sources of evidence supporting the results from our analysis. First, we observe that FEMA allocated a higher share to states such as Florida than would be otherwise allocated on a population pro rata basis. Conversely, other states like California where underweighted. This is consistent with the main recommendations outlined in the paper. Second, even though our recommendations were not applied directly to the location of vaccination centers, the subsequent allocation of vaccines to each state was influenced by the model's recommendations. This underscores a second tactical lever in our model-based recommendation—vaccine allocation—beyond its main strategic lever—location of vaccination sites.

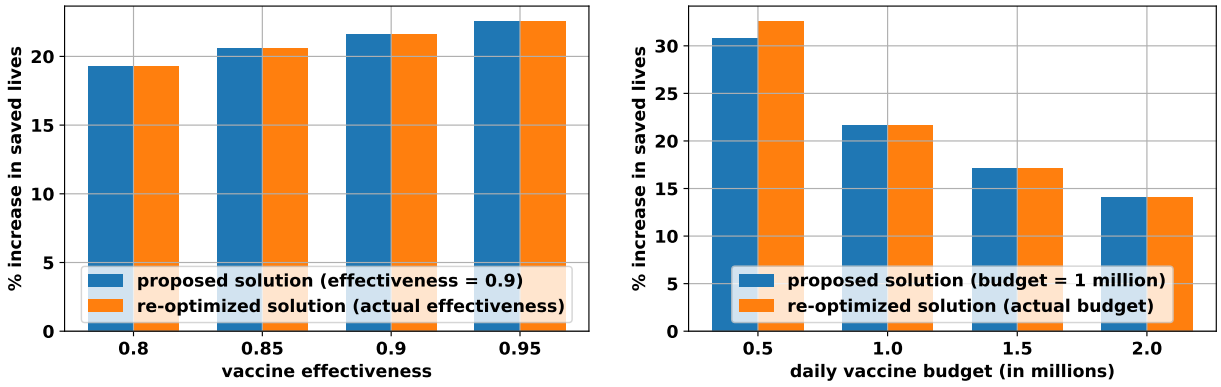
6.6.2 Sensitivity and robustness

A core challenge in vaccine distribution lies in the significant uncertainty regarding the dynamics of the pandemic and the effect of vaccinations. To address this challenge, we assess the sensitivity and the robustness of the optimized solution when the structure of the DELPHI–V model and some of its key parameters are perturbed. For each perturbation, we compare three solutions:

- the top-cities baseline, evaluated with the new perturbed model
- the re-optimized solution, optimized and evaluated with the new perturbed model
- the proposed solution, optimized with the original model and evaluated with the new perturbed model. Specifically, we first run the optimization with the initial inputs. We then introduce perturbations, and re-optimize vaccine allocation decisions (variables $\mathbf{C}$ and $\mathbf{V}$), while fixing the vaccination sites (variable $\mathbf{x}$). Indeed, in practice, vaccination sites are determined once and for all, whereas vaccine allocation can be re-adjusted as information becomes available.

We then compare the number of deaths under these three solutions, estimated with the

perturbed DELPHI–V model. Comparisons between the top-cities benchmark and the re-optimized solution estimate the sensitivity of the benefits of optimization to the perturbations. Comparisons between the top-cities benchmark and the proposed solution estimate the robustness of the solution.



(a) Vaccine effectiveness

(b) Vaccine budget

Figure 6-7: Sensitivity and robustness of results with varying vaccine effectiveness and vaccine budget.

We first vary the two main drivers of the vaccination campaign, that are the vaccine effectiveness (parameter β) and the vaccine budget (parameters B_t). Figure 6-7 reports the percent-wise increase in saved lives, as compared to the top-cities benchmark, for both the proposed and the re-optimized solutions. The main takeaways fall under three categories. First, the benefits of optimization increase with vaccine effectiveness. This is expected, as a higher vaccine effectiveness increases the impact of all strategies (in the extreme example where $\beta = 0$, all strategies have the same null performance). Second, the benefits of optimization decrease with vaccine budget. Indeed, in the extreme scenario with an infinite vaccine budget, any distribution strategy can immediately end the pandemic, leaving essentially no space for optimization. As the budget becomes more scarce, the decisions of *who* receives a vaccine and *when* become increasingly complex, so the optimized strategy has a positive and significant impact on the spread of the disease. It is interesting to note that this monotonic relationship would get inverted in the other regime with a small vaccine budget (again, all solutions perform identically when the budget gets to zero). Yet, with the current vaccine capacities, the benefits of optimization are very strong, saving an extra 15–35% of

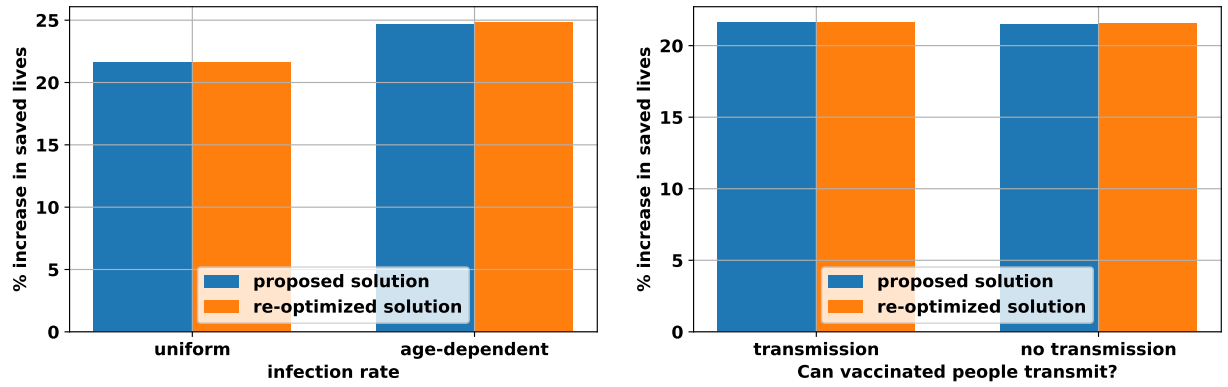
lives.

The third takeaway from Figure 6-7 is the high degree of robustness of the proposed solution. In all but one experiment, the proposed solution remains optimal under the perturbed parameters (indicated by exactly the same benefits obtained with the proposed solution and the re-optimized solution). In the last case (with a daily budget of 500,000 vaccines), the proposed solution is dominated by the re-optimized solution, but remains within 2% of the new optimum. Obviously, the new values of vaccine effectiveness and vaccine budgets impact downstream vaccine allocations and the dynamics of the pandemic. However, the location of vaccination sites is highly robust to variations in vaccine effectiveness and vaccine budgets.

Next, we test the robustness of the proposed solution to the dynamics of the pandemic. One assumption of DELPHI–V is that the infection rate (captured by the parameters α_j and γ_{jt}) is identical across age groups. However, serological evidence suggests potential disparities in infections across sub-populations. To test this, we vary the infection rates with the risk class k, by adjusting the relative variations based on the serological estimates from [168]. Another assumption is that vaccines prevent mortality but not infection and transmission. In practice, vaccines may still reduce the risk of infection and the propensity to transmit the disease. To test this, we perturb the DELPHI–V model by assuming that a fraction β of vaccinated people transition directly to the immune state M (as opposed to the susceptible and vaccinated state S').

Figure 6-8 reports the results from these new perturbations, using the same nomenclature as Figure 6-7. The main takeaway is identical, in that the proposed solution remains near-optimal under the two perturbations. In both tests, the proposed solution results in less than 50 extra deaths than the re-optimized solution—a very small amount in view of the 20,000 lives saved by the optimization. In other words, the proposed solution is not only robust to vaccination characteristics (effectiveness and budget) but also to the dynamics of the COVID-19 pandemic.

Finally, we vary two key parameters in the DELPHI–V dynamics: the infection and mortality rates. Although these parameters are fitted from historical data, there remains considerable uncertainty regarding the dynamics of the pandemic and people's behaviors through a period of mass vaccination. Therefore, we introduce a random perturbation in the



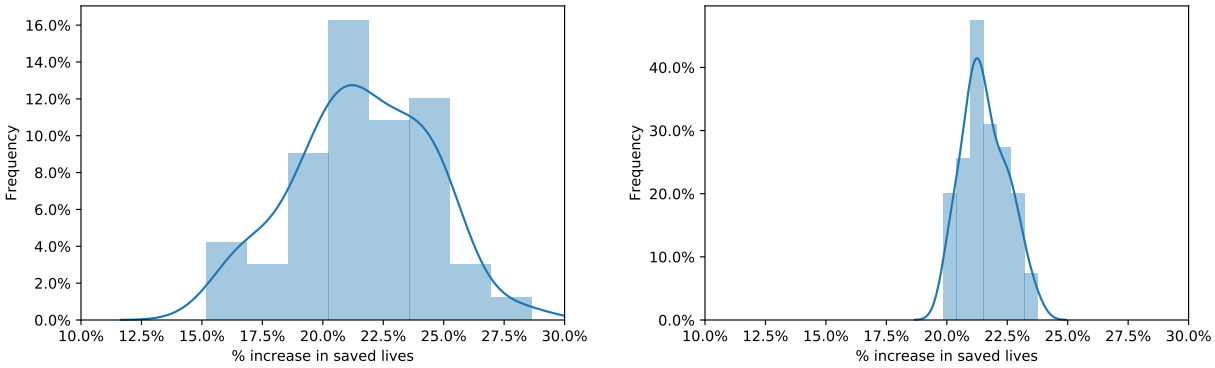
(a) Age-dependent infection rates

(b) No transmission

Figure 6-8: Robustness of results with age-dependent infection rates and no transmission from vaccinated people.

infection or mortality rate, in each state. Specifically, we define 50 perturbation scenarios; in each one, we sample each parameter in each state independently, following a uniform distribution centered around the nominal value and spanning ±20%. Therefore, the full range of infection and mortality rates spans 40% of the nominal value—thus capturing instances where the estimated infection and mortality rates in the DELPHI–V model are subject to very large errors.

Figure 6-9 reports the distribution of the benefits of optimization under these 50 scenarios, with perturbed infection rates (Figure 6-9a) and mortality rates (Figure 6-9b). We compare here the proposed solution (obtained with the nominal values of the infection and mortality rates) and the top-cities benchmark, both evaluated with the perturbed infection and mortality rates. As the results show, the benefits of optimization remains highly significant, and robust to the perturbations. Recall that the benefits of optimization were estimated at 20% under the nominal infection and mortality rates (Table 6.2). After perturbations, they span 12.5%–30% with perturbations in the infection rates and 17.5%–25% with the perturbed mortality rates. In other words, even if the DELPHI–V model makes large errors when estimating the key dynamics of the pandemic, the proposed optimized vaccination sites still increase the impact of the vaccination campaign by over 10%. In fact, the variations can go either way: in over half the simulations, the relative benefits of the optimized solution are even higher under the perturbed parameters than the nominal ones.



(a) Infection rates

(b) Mortality rates

Figure 6-9: Robustness of the benefits of optimization with infection and mortality rates.

Note, finally, that the infection rate has a more significant impact on the relative benefits of optimized vaccine allocation than the mortality rate. This suggests that the impact of the pandemic depends mainly on how vaccines can curb infections at the upstream level—as opposed to mortality at the downstream level. This, again, illustrates the effects of the non-linear SEIR dynamics on the spread of the disease, and how we can leverage an epidemiological model such as DELPHI–V to allocate resources strategically in order to combat the pandemic most effectively.

From a practical standpoint, the results from this section are highly significant. Indeed, the DELPHI–V model, like any epidemiological model, only provides a rough approximation of the dynamics of the pandemic and the effects of vaccinations. Yet, the robustness of the solution provides guarantees that the locations of mass vaccination sites, even optimized against this approximate model, remain highly robust when evaluated against alternative dynamics. This is obviously not to say that we can commit to a full-scale vaccine distribution strategy that spans the entire vaccination campaign. However, the "here-and-now" decisions (i.e., the location of vaccination sites) are likely to remain near-optimal (or even optimal) in the next phases of the pandemic, ultimately enabling the vaccination campaign to save an extra 15–35% of lives.

6.7 Conclusion

This paper has presented a new prescriptive approach to optimize vaccine distribution strategies in response to the COVID-19 pandemic. The approach starts with a state-of-the-art epidemiological model called DELPHI, which augments SEIR models by capturing dynamics specific to COVID-19 (under-detection, governmental and societal response, and declining mortality rates). This paper has first proposed an extension, named DELPHI–V, which captures the effects of vaccinations and reflects the disaggregated impact of COVID-19 on mortality across risk classes (e.g., age groups). Then, this paper has embedded the predictive DELPHI–V model into an optimization model, termed DELPHI–V–OPT, to support vaccine distribution. DELPHI–V–OPT is formulated as a bilinear optimization model, and solved using a tailored algorithm based on coordinate descent.

We applied the model and algorithm to one of the priorities of the new Biden administration in the United States: determining the locations of mass vaccination sites across the country. We formalize the problem by selecting locations strategically to minimize the death toll of the pandemic, subject to practical and fairness constraints. Experimental results using real-world data suggest that the proposed optimization approach can yield significant benefits, as compared to benchmark solutions that locate vaccination sites based on readily-available demographic and epidemiological information. Specifically, the model can increase the effectiveness of the vaccination campaign by 20%, saving an extra 4,000 lives over a three-month period. Remarkably, the proposed solution achieves critical fairness objectives—by significantly reducing the death toll of the pandemic in several states without hurting others—and is highly robust to uncertainties and forecast errors—by achieving similar benefits under a vast range of perturbations.

Obviously, the optimization approach developed in this paper is not without limitations. For instance, our experiments have only partitioned the population according to age groups, thus ignoring other objectives such as prioritizing allocations to healthcare workers, other essential workers, or patients with comorbidities. Moreover, our epidemiological model does not capture heterogeneity in population mixing across subpopulations (e.g., interactions may be more intense in urban areas and between young people than in rural areas and between

older populations). Similarly, our model only captures the first-order effects of vaccinations, ignoring for instance heterogeneity across multiple vaccine types and double-dose vaccines. Finally, our methodology relies on a time discretization approximation and a coordinate descent approach, which do not yield theoretical guarantees on solution quality.

Whereas these limitations undoubtedly motivate further research, this paper lays one of the first data-driven bricks on the optimal distribution of COVID-19 vaccines at a macroscopic level. At a time where vaccine development and vaccine production are going full speed, the results from this paper highlight the critical role of vaccine distribution strategies to combat the pandemic. Obviously, it is essential to make every effort possible to develop new vaccines, enhance vaccine effectiveness, and produce as many vaccines as possible. But this paper highlights another lever that can be pulled to curb the effect of the pandemic: strategically managing vaccine stockpiles to prevent the spread of the pandemic and mitigate its impact. As such, this paper can provide critical decision-making support to governmental agencies as they are currently planning mass vaccinations around the globe.

Acknowledgments

The authors gratefully acknowledge T. Greg McKelvey Jr. for feedback and discussions that have greatly improved the quality of the paper.

Chapter 7

Conclusion

In this thesis, we developed scalable algorithms for data-driven analytics and also illustrated how data-driven analytics could create real-world impact and save lives in the COVID-19 pandemic.

Chapter 2 started with the classical matrix completion problem. We explored a novel reformulation of the classical problem and showed that the formulation exhibited a favorable structure that permitted first-order methods to converge to global optimality. We then designed a projected stochastic gradient descent algorithm, fastImpute, to solve matrix completion efficiently with guarantees. Our synthetic and real-world data experiments demonstrate that fastImpute is able to significantly outperform state-of-the-art methods on accuracy and on scalbility by up to 50x.

Chapter 3 introduced the problem of Interpretable Matrix Completion that extend matrix completion to provide insights toward the latent features of the low-rank decomposition using side information. We showed how this problem can be reconstructed as a binary convex optimization problem, and developed an algorithm based on the novel concept of stochastic cutting planes to efficiently solve matrices up to the scale of $10^6 \times 10^6$. In 4, we then further extended the algorithm to a wide range of data-driven mixed-integer non-linear optimization problems, including sparse regression, support vector machines, and the stochastic knapsack problem. We illustrate how the generic stochastic cutting plane algorithm is competitive with the state-of-the-art domain-specific algorithms in each of these applications.

In Chapter 5, we turn our focus to the application of data-driven analytics during the

COVID-19 pandemic. We developed a novel policy-driven epidemiological model, DELPHI, and demonstrated that it can predict major outbreaks around the world in advance and compares favorably with other top epidemiological models. Furthermore, we demonstrate how we can extract the effect of different government responses during the pandemic using DELPHI's explicit policy modeling. We also illustrate how the extracted policy effects can then be utilized to create a scenario analysis tool for understanding possible future pandemic trajectories. The latter tool was utilized by Janssen in selecting their trial sites for its Ad26.Cov2.S COVID-19 vaccine.

Finally, Chapter 6 demonstrated how epidemiological models can be utilized in downstream applications, in particular vaccine allocations. Working with FEMA, we developed an optimization model to determine the allocation of the limited COVID-19 vaccine supply in mass vaccination centers in order to maximize the number of lives saved by the vaccines, while fully incorporating the non-linear dynamics of DELPHI. We introduced a new coordinate descent algorithm to solve the optimization problem efficiently and showed that optimized vaccine allocation can save 20% more lives. Our results directly impacted FEMA's vaccine allocation and increased their focus in states including Florida and Texas.

Together, these chapters illustrate the both the methodology and the application side of data-driven analytics for impact. We hope the application of these scalable algorithms would improve the ability to extract data-driven insights across different industries, while our successful applications could inspire future work on using data-driven analytics to drive impact in healthcare and epidemiology.

Bibliography

[1] Kay Aaby, Jeffrey W Herrmann, Carol S Jordan, Mark Treadwell, and Kathy Wood. Montgomery county's public health service uses operations research to plan emergency mass dispensing and vaccination clinics. *Interfaces*, 36(6):569–579, 2006.

[2] Tobias Achterberg and Roland Wunderling. Mixed integer programming: Analyzing 12 years of progress. In *Facets of combinatorial optimization*, pages 449–481. Springer, 2013.

[3] Michael Alley, Max Biggs, Rim Hariss, Charles Herrmann, Michael Li, and Georgia Perakis. Pricing for heterogeneous products: Analytics for ticket reselling. *Available at SSRN 3360622*, 2019.

[4] Ozgur M Araz, Alison Galvani, and Lauren A Meyers. Geographic prioritization of distributing pandemic influenza vaccines. *Health Care Management Science*, 15(3):175–187, 2012.

[5] Kenan Arifoğlu, Sarang Deo, and Seyed MR Iravani. Consumption externality and yield uncertainty in the influenza vaccine supply chain: Interventions in demand and supply sides. *Management Science*, 58(6):1072–1091, 2012.

[6] Nimalan Arinaminpathy, Oliver Ratmann, Katia Koelle, Suzanne L Epstein, Graeme E Price, Cecile Viboud, Mark A Miller, and Bryan T Grenfell. Impact of cross-protective vaccines on epidemiological and evolutionary dynamics of influenza. *Proceedings of the National Academy of Sciences*, 109(8):3173–3177, 2012.

[7] Melissa M Arons, Kelly M Hatfield, Sujan C Reddy, Anne Kimball, Allison James, Jesica R Jacobs, Joanne Taylor, Kevin Spicer, Ana C Bardossy, Lisa P Oakley, et al. Presymptomatic sars-cov-2 infections and transmission in a skilled nursing facility. *New England Journal of Medicine*, 2020.

[8] Joseph Bae, Darshan Gandhi, Jil Kothari, Sheshank Shankar, Jonah Bae, Parth Patwa, Rohan Sukumaran, Krutika Mishra, Srinidhi Murali, Aishwariya Saxena, et al. Challenges of equitable vaccine distribution in the covid-19 pandemic. *arXiv preprint arXiv:2012.12263*, 2020.

[9] Laura Balzano, Robert Nowak, and Benjamin Recht. Online identification and tracking of subspaces from highly incomplete information. In *2010 48th Annual allerton conference on communication, control, and computing (Allerton)*, pages 704–711. IEEE, 2010.

[10] Hari Bandi and Dimitris Bertsimas. Optimizing influenza vaccine composition: From predictions to prescriptions. In *Machine Learning for Healthcare Conference*, pages 121–142. PMLR, 2020.

[11] Rémi Bardenet, Odalric-Ambrym Maillard, et al. Concentration inequalities for sampling without replacement. *Bernoulli*, 21(3):1361–1385, 2015.

[12] Nicole E Basta, Dennis L Chao, M Elizabeth Halloran, Laura Matrajt, and Ira M Longini Jr. Strategies for pandemic and seasonal influenza vaccination of schoolchildren in the United States. *American journal of epidemiology*, 170(6):679–686, 2009.

[13] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.

[14] Eran Bendavid, Bianca Mulaney, Neeraj Sood, Soleil Shah, Emilia Ling, Rebecca Bromley-Dulfano, Cara Lai, Zoe Weissberg, Rodrigo Saavedra, James Tedrow, et al. Covid-19 antibody seroprevalence in santa clara county, california. *MedRxiv*, 2020.

[15] James Bennett, Stan Lanning, et al. The netflix prize. In *KDD Cup and Workshop 2007*. Citeseer, 2007.

[16] Dimitri P Bertsekas. Nonlinear programming. *Journal of the Operational Research Society*, 48(3):334–334, 1997.

[17] Dimitris Bertsimas, Hari Bandi, Leonard Boussioux, Ryan Cory-Wright, Arthur Delarue, Vasileios Digalakis, Samuel Gilmour, Justin Graham, Adam Kim, Driss Lahlou Kitane, Zhen Lin, Galit Lukin, Michael Li, Luca Mingardi, Liangyuan Na, Agni Orfanoudaki, Theodore Papalexopoulos, Ivan Paskov, Jean Pauphilet, Omar Skali Lami, Matthew Sobiesk, Bartolomeo Stellato, Kimberly Carballo, Yuchen Wang, Holly Wiberg, and Cynthia Zeng. An aggregated dataset of clinical outcomes for covid-19 patients, 2020.

[18] Dimitris Bertsimas, Leonard Boussioux, Ryan Cory Wright, Arthur Delarue, Vassilis Digalakis Jr, Alexandre Jacquillat, Driss Lahlou Kitane, Galit Lukin, Michael Lingzhi Li, Luca Mingardi, et al. From predictions to prescriptions: A data-driven response to COVID-19. *arXiv preprint arXiv:2006.16509*, 2020.

[19] Dimitris Bertsimas and Martin S Copenhaver. Characterization of the equivalence of robustification and regularization in linear and matrix regression. *European Journal of Operational Research*, 270(3):931–942, 2018.

[20] Dimitris Bertsimas and Martin S Copenhaver. Characterization of the equivalence of robustification and regularization in linear, median, and matrix regression. *European Journal of Operations Research*, 270:931–942, 2018.

[21] Dimitris Bertsimas, Vassilis Digalakis Jr, Alexander Jacquillat, Michael Lingzhi Li, and Alessandro Previero. Where to locate covid-19 mass vaccination facilities? *Naval Research Logistics (NRL)*, 2021.

[22] Dimitris Bertsimas and Michael Lingzhi Li. Interpretable matrix completion: A discrete optimization approach. *arXiv preprint arXiv:1812.06647*, 2018.

[23] Dimitris Bertsimas and Bart Van Parys. Sparse high-dimensional regression: Exact scalable algorithms and phase transitions. *arXiv preprint arXiv:1709.10029*, 2017.

[24] Dimitris Bertsimas and Bart Van Parys. Sparse high-dimensional regression: Exact scalable algorithms and phase transitions. *The Annals of Statistics*, 48(1):300–323, 2020.

[25] Robert Bixby and Edward Rothberg. Progress in computational mixed integer programming—a look back from the other side of the tipping point. *Annals of Operations Research*, 149(1):37–41, 2007.

[26] Richard W Blundell and James L Powell. Endogeneity in semiparametric binary response models. *The Review of Economic Studies*, 71(3):655–679, 2004.

[27] Stephen Boyd, Laurent El Ghaoui, Eric Feron, and Venkataramanan Balakrishnan. *Linear matrix inequalities in system and control theory*, volume 15. Siam, 1994.

[28] Tom Britton, Frank Ball, and Pieter Trapman. A mathematical model reveals the influence of population heterogeneity on herd immunity to sars-cov-2. *Science*, 369(6505):846–849, 2020.

[29] Oyungerel Byambasuren, Magnolia Cardona, Katy Bell, Justin Clark, Mary-Louise McLaws, and Paul Glasziou. Estimating the extent of true asymptomatic covid-19 and its potential for community transmission: systematic review and meta-analysis. *Available at SSRN 3586675*, 2020.

[30] Richard H Byrd, Jorge Nocedal, and Richard A Waltz. Knitro: An integrated package for nonlinear optimization. In *Large-scale nonlinear optimization*, pages 35–59. Springer, 2006.

[31] Giuseppe Calafiore and Marco C Campi. Uncertain convex programs: randomized solutions and confidence levels. *Mathematical Programming*, 102(1):25–46, 2005.

[32] Emmanuel J Candes and Yaniv Plan. Matrix completion with noise. *Proceedings of the IEEE*, 98(6):925–936, 2010.

[33] Emmanuel J Candès and Benjamin Recht. Exact matrix completion via convex optimization. *Foundations of Computational mathematics*, 9(6):717, 2009.

[34] Emmanuel J Candès and Terence Tao. The power of convex relaxation: Near-optimal matrix completion. *IEEE Transactions on Information Theory*, 56(5):2053–2080, 2010.

[35] Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):1–27, 2011.

[36] Yudong Chen, Srinadh Bhojanapalli, Sujay Sanghavi, and Rachel Ward. Coherent matrix completion. In *International Conference on Machine Learning*, pages 674–682, 2014.

[37] Yudong Chen, Ali Jalali, Sujay Sanghavi, and Huan Xu. Clustering partially observed graphs via convex optimization. *The Journal of Machine Learning Research*, 15(1):2213–2238, 2014.

[38] Yudong Chen and Martin J Wainwright. Fast low-rank estimation by projected gradient descent: General statistical and algorithmic guarantees. *arXiv preprint arXiv:1509.03025*, 2015.

[39] Herman Chernoff. A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *The Annals of Mathematical Statistics*, 23(4):493–507, 1952.

[40] Kai-Yang Chiang, Cho-Jui Hsieh, and Inderjit S Dhillon. Matrix completion with noisy side information. In *Advances in Neural Information Processing Systems*, pages 3447–3455, 2015.

[41] Kai-Yang Chiang, Cho-Jui Hsieh, Nagarajan Natarajan, Inderjit S Dhillon, and Ambuj Tewari. Prediction and clustering in signed networks: a local to global perspective. *The Journal of Machine Learning Research*, 15(1):1177–1213, 2014.

[42] Stephen E Chick, Hamed Mamani, and David Simchi-Levi. Supply chain coordination and influenza vaccination. *Operations Research*, 56(6):1493–1506, 2008.

[43] Matteo Chinazzi, Jessica T Davis, Marco Ajelli, Corrado Gioannini, Maria Litvinova, Stefano Merler, Ana Pastore y Piontti, Kunpeng Mu, Luca Rossi, Kaiyuan Sun, et al. The effect of travel restrictions on the spread of the 2019 novel coronavirus (covid-19) outbreak. *Science*, 368(6489):395–400, 2020.

[44] Soo-Haeng Cho. The optimal composition of influenza vaccines subject to random production yields. *Manufacturing & Service Operations Management*, 12(2):256–277, 2010.

[45] Gerardo Chowell, Cécile Viboud, Xiaohong Wang, Stefano M Bertozzi, and Mark A Miller. Adaptive vaccination strategies to mitigate pandemic influenza: Mexico as a case study. *PLoS One*, 4(12):e8164, 2009.

[46] Chris Coey, Miles Lubin, and Juan Pablo Vielma. Outer approximation with conic certificates for mixed-integer convex problems. *Mathematical Programming Computation*, pages 1–45, 2020.

[47] National Research Council et al. *Assessing the reliability of complex models: mathematical and statistical foundations of verification, validation, and uncertainty quantification*. National Academies Press, 2012.

[48] Alberto Coustasse, Craig Kimble, and Kenneth Maxik. Covid-19 and vaccine hesitancy: A challenge the united states must overcome. *The Journal of Ambulatory Care Management*, 44(1):71–75, 2021.

[49] Tinglong Dai, Soo-Haeng Cho, and Fuqiang Zhang. Contracting for on-time delivery in the us influenza vaccine supply chain. *Manufacturing & Service Operations Management*, 18(3):332–346, 2016.

[50] Hadi Daneshmand, Jonas Kohler, Aurelien Lucchi, and Thomas Hofmann. Escaping saddles with stochastic gradients. *arXiv preprint arXiv:1803.05999*, 2018.

[51] Natalie E Dean, Ana Pastore y Piontti, Zachary J Madewell, Derek AT Cummings, Matthew DT Hitchings, Keya Joshi, Rebecca Kahn, Alessandro Vespignani, M Elizabeth Halloran, and Ira M Longini Jr. Ensemble forecast modeling for the design of covid-19 vaccine efficacy trials. *Vaccine*, 38(46):7213–7216, 2020.

[52] Paramveer Dhillon, Yichao Lu, Dean P Foster, and Lyle Ungar. New subsampling algorithms for fast least squares regression. In *Advances in neural information processing systems*, pages 360–368, 2013.

[53] Asako Doi, Kentaro Iwata, Hirokazu Kuroda, Toshikazu Hasuike, Seiko Nasu, Aya Kanda, Tomomi Nagao, Hiroaki Nishioka, Keisuke Tomii, Takeshi Morimoto, et al. Estimation of seroprevalence of novel coronavirus disease (covid-19) using preserved serum at an outpatient setting in kobe, japan: A cross-sectional study. *medRxiv*, 2020.

[54] Amiel A Dror, Netanel Eisenbach, Shahar Taiber, Nicole G Morozov, Matti Mizrachi, Asaf Zigron, Samer Srouji, and Eyal Sela. Vaccine hesitancy: the next challenge in the fight against covid-19. *European journal of epidemiology*, 35(8):775–779, 2020.

[55] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[56] Lotty Evertje Duijzer, Willem van Jaarsveld, and Rommert Dekker. Literature review: The vaccine supply chain. *European Journal of Operational Research*, 268(1):174–192, 2018.

[57] Marco A Duran and Ignacio E Grossmann. An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Mathematical programming*, 36(3):307–339, 1986.

[58] Marco A Duran and Ignacio E Grossmann. An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Mathematical programming*, 36(3):307–339, 1986.

[59] Jonathan Dushoff, Joshua B Plotkin, Cecile Viboud, Lone Simonsen, Mark Miller, Mark Loeb, and JD David. Vaccinating to protect a vulnerable subpopulation. *PLoS Med*, 4(5):e174, 2007.

[60] Lila R Elveback, John P Fox, Eugene Ackerman, Alice Langworthy, Mary Boyd, and Lael Gatewood. An influenza simulation model for immunization studies. *American Journal of Epidemiology*, 103(2):152–165, 1976.

[61] Ezekiel J Emanuel and Alan Wertheimer. Who should get influenza vaccine when not all can? *Science*, 312(5775):854–855, 2006.

[62] Christian Erikstrup, Christoffer Egeberg Hother, Ole Birger Vestager Pedersen, Kaare Molbak, Robert Leo Skov, Dorte Kinggaard Holm, Susanne Saekmose, Anna Christine Nilsson, Patrick Terrence Brooks, Jens Kjaergaard Boldsen, et al. Estimation of sars-cov-2 infection fatality rate by real-time antibody screening of blood donors. *medRxiv*, 2020.

[63] Cong Fang, Zhouchen Lin, and Tong Zhang. Sharp analysis for nonconvex sgd escaping from saddle points. *arXiv preprint arXiv:1902.00247*, 2019.

[64] Maha R Farhat, B Jesse Shapiro, Karen J Kieser, Razvan Sultana, Karen R Jacobson, Thomas C Victor, Robin M Warren, Elizabeth M Streicher, Alistair Calver, Alex Sloutsky, et al. Genomic analysis identifies targets of convergent positive selection in drug-resistant mycobacterium tuberculosis. *Nature genetics*, 45(10):1183, 2013.

[65] Federal Emergency Management Agency. FEMA COVID-19 Vaccination Update. `https://www.fema.gov/press-release/20210402/fema-covid-19-vaccination-update`, April 2021. Accessed: 2021-04-09.

[66] Federal Emergency Management Agency. FEMA Supporting Vaccination Centers Nationwide. `https://www.fema.gov/press-release/20210226/fema-supporting-vaccination-centers-nationwide`, February 2021. Accessed: 2021-04-09.

[67] Awi Federgruen and Nan Yang. Competition under generalized attraction models: Applications to quality competition under yield uncertainty. *Management science*, 55(12):2028–2043, 2009.

[68] Willliam Feller. *An introduction to probability theory and its applications, vol 2*. John Wiley & Sons, 2008.

[69] Helena F Florindo, Ron Kleiner, Daniella Vaskovich-Koubi, Rita C Acúrcio, Barbara Carreira, Eilam Yeini, Galia Tiram, Yulia Liubomirski, and Ronit Satchi-Fainaro. Immune-mediated approaches against COVID-19. *Nature nanotechnology*, pages 1–16, 2020.

[70] Vojtěch Franc and Soeren Sonnenburg. Optimized cutting plane algorithm for support vector machines. In *Proceedings of the 25th international conference on Machine learning*, pages 320–327, 2008.

[71] Vojtěch Franc and Sören Sonnenburg. Optimized cutting plane algorithm for large-scale risk minimization. *Journal of Machine Learning Research*, 10(10), 2009.

[72] Vojtech Franc, Sören Sonnenburg, and Tomáš Werner. Cutting plane methods in machine learning. *Optimization for Machine Learning*, pages 185–218, 2011.

[73] Rong Ge, Furong Huang, Chi Jin, and Yang Yuan. Escaping from saddle points—online stochastic gradient for tensor decomposition. In *Conference on Learning Theory*, pages 797–842, 2015.

[74] Rong Ge, Chi Jin, and Yi Zheng. No spurious local minima in nonconvex low rank problems: A unified geometric analysis. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1233–1242. JMLR. org, 2017.

[75] Rong Ge, Jason D Lee, and Tengyu Ma. Matrix completion has no spurious local minimum. In *Advances in Neural Information Processing Systems*, pages 2973–2981, 2016.

[76] M Gabriela M Gomes, Rodrigo M Corder, Jessica G King, Kate E Langwig, Caetano Souto-Maior, Jorge Carneiro, Guilherme Gonçalves, Carlos Penha-Gonçalves, Marcelo U Ferreira, and Ricardo Aguas. Individual variation in susceptibility or exposure to sars-cov-2 lowers the herd immunity threshold. *MedRxiv*, 2020.

[77] Parag Goyal, Justin J Choi, Laura C Pinheiro, Edward J Schenck, Ruijun Chen, Assem Jabri, Michael J Satlin, Thomas R Campion Jr, Musarrat Nahid, Joanna B Ringel, et al. Clinical characteristics of Covid-19 in New York City. *New England Journal of Medicine*, 2020.

[78] Barney S Graham. Rapid COVID-19 vaccine development. *Science*, 368(6494):945–946, 2020.

[79] Jens Grauer, Hartmut Löwen, and Benno Liebchen. Strategic spatiotemporal vaccine distribution increases the survival rate in an infectious disease like covid-19. *Scientific reports*, 10(1):1–10, 2020.

[80] Jonathan Grein, Norio Ohmagari, Daniel Shin, George Diaz, Erika Asperges, Antonella Castagna, Torsten Feldt, Gary Green, Margaret L Green, François-Xavier Lescure, et al. Compassionate use of remdesivir for patients with severe covid-19. *New England Journal of Medicine*, 2020.

[81] David Gross and Vincent Nesme. Note on sampling without replacing from a finite collection of matrices. *arXiv preprint arXiv:1001.2738*, 2010.

[82] Youyang Gu. Covid-19 projections using machine learning, October 2020.

[83] Wei-jie Guan, Zheng-yi Ni, Yu Hu, Wen-hua Liang, Chun-quan Ou, Jian-xing He, Lei Liu, Hong Shan, Chun-liang Lei, David SC Hui, et al. Clinical characteristics of coronavirus disease 2019 in China. *New England journal of medicine*, 382(18):1708–1720, 2020.

[84] Gurobi Optimization. Gurobi optimizer reference manual, 2020.

[85] Thomas Hale, Sam Webster, Anna Petherick, Toby Phillips, and Beatriz Kira. Oxford covid-19 government response tracker, blavatnik school of government. *Oxford University. Creative Commons Attribution CC BY standard. Available at: https://www. bsg. ox. ac. uk/covidtracker. Accessed on: April*, 14:2020, 2020.

[86] Trevor Hastie, Rahul Mazumder, Jason D Lee, and Reza Zadeh. Matrix completion and low-rank svd via fast alternating least squares. *The Journal of Machine Learning Research*, 16(1):3367–3402, 2015.

[87] Zhiliang Hu, Ci Song, Chuanjun Xu, Guangfu Jin, Yaling Chen, Xin Xu, Hongxia Ma, Wei Chen, Yuan Lin, Yishan Zheng, et al. Clinical characteristics of 24 asymptomatic infections with covid-19 screened among close contacts in nanjing, china. *Science China Life Sciences*, pages 1–6, 2020.

[88] IHME. health service utilization forecasting team, murray cjl. *Forecasting COVID-19 impact on hospital bed-days, ICU-days, ventilator-days and deaths by US state in the next*, 4, 19.

[89] Alvin J Ing, Christine Cocks, and Jeffery Peter Green. Covid-19: in the footsteps of ernest shackleton. *Thorax*, 2020.

[90] Sheldon H Jacobson, Edward C Sewell, and Ruben A Proano. An analysis of the pediatric vaccine supply shortage problem. *Health Care Management Science*, 9(4):371–389, 2006.

[91] Sheldon H Jacobson, Edward C Sewell, Ruben A Proano, and Janet A Jokela. Stockpile levels for pediatric vaccines: How much is enough? *Vaccine*, 24(17):3530–3537, 2006.

[92] Prateek Jain and Inderjit S Dhillon. Provable inductive matrix completion. *arXiv preprint arXiv:1306.0626*, 2013.

[93] Prateek Jain, Raghu Meka, and Inderjit S Dhillon. Guaranteed rank minimization via singular value projection. In *Advances in Neural Information Processing Systems*, pages 937–945, 2010.

[94] Prateek Jain and Praneeth Netrapalli. Fast exact matrix completion with finite samples. In *Conference on Learning Theory*, pages 1007–1034, 2015.

[95] Hui Ji, Chaoqiang Liu, Zuowei Shen, and Yuhong Xu. Robust video denoising using low rank matrix completion. In *Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 2010.

[96] Chi Jin, Sham M Kakade, and Praneeth Netrapalli. Provable efficient online matrix completion via non-convex stochastic gradient descent. In *Advances in Neural Information Processing Systems*, pages 4520–4528, 2016.

[97] Thorsten Joachims. Training linear svms in linear time. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 217–226, 2006.

[98] Matt J Keeling and Andrew Shattock. Optimal but unequitable prophylactic distribution of vaccine. *Epidemics*, 4(2):78–85, 2012.

[99] William Ogilvy Kermack and Anderson G McKendrick. A contribution to the mathematical theory of epidemics. *Proceedings of the royal society of london. Series A, Containing papers of a mathematical and physical character*, 115(772):700–721, 1927.

[100] Raghunandan H Keshavan, Sewoong Oh, and Andrea Montanari. Matrix completion from a few entries. In *Information Theory, 2009. ISIT 2009. IEEE International Symposium on*, pages 324–328. IEEE, 2009.

[101] Roxanne Khamsi. If a coronavirus vaccine arrives, can the world make enough. *Nature*, 580(7805):578–580, 2020.

[102] Stephen M Kissler, Christine Tedijanto, Edward Goldstein, Yonatan H Grad, and Marc Lipsitch. Projecting the transmission dynamics of SARS-CoV-2 through the postpandemic period. *Science*, 2020.

[103] Anton J Kleywegt, Alexander Shapiro, and Tito Homem-de Mello. The sample average approximation method for stochastic discrete optimization. *SIAM Journal on Optimization*, 12(2):479–502, 2002.

[104] Marjolein Kluytmans, Anton Buiting, Suzan Pas, Robbert Bentvelsen, Wouter van den Bijllaardt, Anne van Oudheusden, Miranda van Rijen, Jaco Verweij, Marion Koopmans, and Jan Kluytmans. Sars-cov-2 infection in 86 healthcare workers in two dutch hospitals in march 2020. *medRxiv*, 2020.

[105] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, pages 30–37, 2009.

[106] Laura J Kornish and Ralph L Keeney. Repeated commit-or-defer decisions with a deadline: The influenza vaccine composition. *Operations Research*, 56(3):527–541, 2008.

[107] Steven G Krantz and Arni SR Srinivasa Rao. Level of under-reporting including underdiagnosis before the first peak of covid-19 in various countries: Preliminary retrospective results based on wavelets and deterministic modeling. *Infection Control & Hospital Epidemiology*, pages 1–8, 2020.

[108] Jan Kronqvist, David E Bernal, Andreas Lundell, and Ignacio E Grossmann. A review and comparison of solvers for convex minlp. *Optimization and Engineering*, 20(2):397–455, 2019.

[109] Richard C Larson. Simple models of influenza progression within a heterogeneous population. *Operations research*, 55(3):399–412, 2007.

[110] Stephen A Lauer, Kyra H Grantz, Qifang Bi, Forrest K Jones, Qulu Zheng, Hannah R Meredith, Andrew S Azman, Nicholas G Reich, and Justin Lessler. The incubation period of coronavirus disease 2019 (covid-19) from publicly reported confirmed cases: estimation and application. *Annals of internal medicine*, 172(9):577–582, 2020.

[111] Sunmi Lee, Michael Golinski, and Gerardo Chowell. Modeling optimal age-specific vaccination strategies against pandemic influenza. *Bulletin of mathematical biology*, 74(4):958–980, 2012.

[112] Stef Lemmens, Catherine Decouttere, Nico Vandaele, and Mauro Bernuzzi. A review of integrated supply chain network design models: Key issues for vaccine supply chains. *Chemical Engineering Research and Design*, 109:366–384, 2016.

[113] Michael L Li, Hamza Tazi Bouardi, Omar Skali Lami, Thomas A Trikalinos, Nikolaos K Trichakis, and Dimitris Bertsimas. Forecasting COVID-19 and analyzing the effect of government interventions. *medRxiv*, 2020.

[114] Michael Lingzhi Li, Hamza Tazi Bouardi, Omar Skali Lami, Thomas A Trikalinos, Nikolaos K Trichakis, and Dimitris Bertsimas. Forecasting covid-19 and analyzing the effect of government interventions. *MedRxiv*, pages 2020–06, 2021.

[115] Yanli Liu, Wenwu Sun, Liangkai Chen, Yujun Wang, Lijuan Zhang, and Li Yu. Clinical characteristics and progression of 2019 novel coronavirus-infected patients concurrent acute respiratory distress syndrome. *medRxiv*, 2020.

[116] Ying Liu, Albert A Gayle, Annelies Wilder-Smith, and Joacim Rocklöv. The reproductive number of covid-19 is higher compared to sars coronavirus. *Journal of travel medicine*, 2020.

[117] Ira M Longini Jr, Eugene Ackerman, and Lila R Elveback. An optimization model for influenza a epidemics. *Mathematical Biosciences*, 38(1-2):141–157, 1978.

[118] José Lourenço, Robert Paton, Mahan Ghafari, Moritz Kraemer, Craig Thompson, Peter Simmonds, Paul Klenerman, and Sunetra Gupta. Fundamental principles of epidemic spread highlight the immediate need for large-scale serological surveys to assess the stage of the sars-cov-2 epidemic. *medRxiv*, 2020.

[119] Jin Lu, Guannan Liang, Jiangwen Sun, and Jinbo Bi. A sparse interactive model for matrix completion with side information. In *Advances in neural information processing systems*, pages 4071–4079, 2016.

[120] Nicole Lurie, Melanie Saville, Richard Hatchett, and Jane Halton. Developing covid-19 vaccines at pandemic speed. *New England Journal of Medicine*, 382(21):1969–1973, 2020.

[121] Cong Ma, Kaizheng Wang, Yuejie Chi, and Yuxin Chen. Implicit regularization in nonconvex statistical estimation: Gradient descent converges linearly for phase retrieval, matrix completion, and blind deconvolution. *Foundations of Computational Mathematics*, pages 1–182, 2019.

[122] Ho-Yin Mak, Tinglong Dai, and Christopher S Tang. Managing two-dose covid-19 vaccine rollouts with limited supply. *Available at SSRN 3790836*, 2021.

[123] Hamed Mamani, Stephen E Chick, and David Simchi-Levi. A game-theoretic model of international influenza vaccination coordination. *Management Science*, 59(7):1650–1670, 2013.

[124] Laura Matrajt, Julia Eaton, Tiffany Leung, and Elizabeth R Brown. Vaccine optimization for covid-19: who to vaccinate first? *medRxiv*, 2020.

[125] Laura Matrajt, M Elizabeth Halloran, and Ira M Longini Jr. Optimal vaccine allocation for the early mitigation of pandemic influenza. *PLoS Comput Biol*, 9(3):e1002964, 2013.

[126] Rahul Mazumder, Trevor Hastie, and Robert Tibshirani. Spectral regularization algorithms for learning large incomplete matrices. *Journal of machine learning research*, 11(Aug):2287–2322, 2010.

[127] Jan Medlock and Alison P Galvani. Optimizing influenza vaccine distribution. *Science*, 325(5948):1705–1708, 2009.

[128] Prakhar Mehrotra and Ji Ivan. Prophet logistic forecasting, October 2020.

[129] Jean-Jacques Muriel and Howard Bauchner. Vaccine distribution—equity left behind? *JAMA*, 2021.

[130] CJ Murray et al. Forecasting the impact of the first wave of the covid-19 pandemic on hospital demand and deaths for the usa and european economic area countries. *medRxiv*, 2020.

[131] Nagarajan Natarajan and Inderjit S Dhillon. Inductive matrix completion for predicting gene–disease associations. *Bioinformatics*, 30(12):i60–i68, 2014.

[132] National Academies of Sciences, Engineering, and Medicine. *Framework for equitable allocation of COVID-19 vaccine*. National Academies Press, 2020.

[133] Ivan Nazarov, Boris Shirokikh, Maria Burkina, Gennady Fedonin, and Maxim Panov. Sparse group inductive matrix completion. *arXiv preprint arXiv:1804.10653*, 2018.

[134] Sahand Negahban and Martin J Wainwright. Restricted strong convexity and weighted matrix completion: Optimal bounds with noise. *Journal of Machine Learning Research*, 13(May):1665–1697, 2012.

[135] Yurii E Nesterov. A method for solving the convex programming problem with convergence rate o (1/kˆ 2). In *Dokl. akad. nauk Sssr*, volume 269, pages 543–547, 1983.

[136] New York Times. New York Times, Coronavirus in the U.S.: Latest Map and Case Count. https://www.nytimes.com/interactive/2020/us/coronavirus-us-cases.html, 2020.

[137] Rene Niehus, Pablo Martinez de Salazar Munoz, Aimee Taylor, and Marc Lipsitch. Quantifying bias of covid-19 prevalence and severity estimates in wuhan, china that depend on reported cases in international travelers. *medRxiv*, 2020.

[138] Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[139] Dave Osthus, Sara Del Valle, Carrie Manore, Brian Weaver, Lauren Castro, Courtney Shelley, Manhong Smith, Julie Spencer, Geoffrey Fairchild, Travis Pitts, Dax Gerts, Lori Dauelsberg, Ashlynn Daughton, Morgan Gorris, Beth Hornbein, Daniel Israel, Nidhi Parikh, Deborah Shutt, and Amanda Ziemann. Lanl covid-19 cases and deaths forecasts, October 2020.

[140] SC Parikh. Approximate cutting planes in nonlinear programming. *Mathematical Programming*, 11(1):194–198, 1976.

[141] Rajan Patel, Ira M Longini Jr, and M Elizabeth Halloran. Finding optimal vaccination strategies for pandemic influenza using genetic algorithms. *Journal of theoretical biology*, 234(2):201–212, 2005.

[142] Alex Perkins and Guido Espana. Notredame-fred covid-19 forecasts, October 2020.

[143] Christopher M Petrilli, Simon A Jones, Jie Yang, Harish Rajagopalan, Luke F O'Donnell, Yelena Chernyak, Katie Tobin, Robert J Cerfolio, Fritz Francois, and Leora I Horwitz. Factors associated with hospitalization and critical illness among 4,103 patients with COVID-19 disease in New York City. *medRxiv*, 2020.

[144] Fletcher R and Leyffer S. Solving mixed integer nonlinear programs by outer approximation. *Mathematical programming*, 66(3):327–349, 1994.

[145] Mehdi Rastegar, Madjid Tavana, Afshin Meraj, and Hassan Mina. An inventory-location optimization model for equitable influenza vaccine distribution in developing countries during the covid-19 pandemic. *Vaccine*, 39(3):495–504, 2021.

[146] Evan L Ray, Nutcha Wattanachit, Jarad Niemi, Abdul Hannan Kanji, Katie House, Estee Y Cramer, Johannes Bracher, Andrew Zheng, Teresa K Yamana, Xinyue Xiong, et al. Ensemble forecasts of coronavirus disease 2019 (covid-19) in the us. *MedRXiv*, 2020.

[147] Meisam Razaviyayn. *Successive convex approximation: Analysis and applications*. PhD thesis, University of Minnesota, 2014.

[148] Benjamin Recht and Christopher Ré. Parallel stochastic gradient algorithms for large-scale matrix completion. *Mathematical Programming Computation*, 5(2):201–226, 2013.

[149] Douglas Rivers and Quang H Vuong. Limited information estimators and exogeneity tests for simultaneous probit models. *Journal of econometrics*, 39(3):347–366, 1988.

[150] Alexander Rodriguez, Anika Tabassum, Jiaming Cui, Jiajia Xie, Javen Ho, Pulak Agarwal, Bijaya Adhikari, and B Aditya Prakash. Deepcovid: An operational deep learning-driven framework for explainable real-time covid-19 forecasting. *medRxiv*, 2020.

[151] Andrew A Sayampanathan, Cheryl S Heng, Phua Hwee Pin, Junxiong Pang, Teoh Yee Leong, and Vernon J Lee. Infectivity of asymptomatic versus symptomatic covid-19. *The Lancet*, 397(10269):93–94, 2021.

[152] Vatsal Shah, Nikhil Rao, and Weicong Ding. Matrix factorization with side and higher order information. *stat*, 1050:4, 2017.

[153] Matthew D Shin, Sourabh Shukla, Young Hun Chung, Veronique Beiss, Soo Khim Chan, Oscar A Ortega-Rivera, David M Wirth, Angela Chen, Markus Sack, Jonathan K Pokorski, et al. COVID-19 vaccine development and a potential nanomaterial path forward. *Nature Nanotechnology*, pages 1–10, 2020.

[154] Si Si, Kai-Yang Chiang, Cho-Jui Hsieh, Nikhil Rao, and Inderjit S Dhillon. Goal-directed inductive matrix completion. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1165–1174. ACM, 2016.

[155] Akshay Soni, Troy Chevalier, and Swayambhoo Jain. Noisy inductive matrix completion under sparse factor models. *arXiv preprint arXiv:1609.03958*, 2016.

[156] Neeraj Sood, Paul Simon, Peggy Ebner, Daniel Eichner, Jeffrey Reynolds, Eran Bendavid, and Jay Bhattacharya. Seroprevalence of sars-cov-2–specific antibodies among adults in los angeles county, california, on april 10-11, 2020. *JAMA*, 2020.

[157] Gilbert W Stewart. *Matrix perturbation theory*. Citeseer, 1990.

[158] Hendrik Streeck, Bianca Schulte, Beate Kuemmerer, Enrico Richter, Tobias Höller, Christine Fuhrmann, Eva Bartok, Ramona Dolscheid, Moritz Berger, Lukas Wessendorf, et al. Infection fatality rate of sars-cov-2 infection in a german community with a super-spreading event. *medRxiv*, 2020.

[159] Peng Sun, Liu Yang, and Francis De Véricourt. Selfish drug allocation for containing an international influenza pandemic at the onset. *Operations Research*, 57(6):1320–1332, 2009.

[160] Ruoyu Sun and Zhi-Quan Luo. Guaranteed matrix completion via non-convex factorization. *IEEE Transactions on Information Theory*, 62(11):6535–6579, 2016.

[161] Jared Tanner and Ke Wei. Normalized iterative hard thresholding for matrix completion. *SIAM Journal on Scientific Computing*, 35(5):S104–S125, 2013.

[162] Matthew W Tanner, Lisa Sattenspiel, and Lewis Ntaimo. Finding optimal vaccination strategies under parameter uncertainty using stochastic programming. *Mathematical biosciences*, 215(2):144–151, 2008.

[163] Anna Teytelman and Richard C Larson. Multiregional dynamic vaccine allocation during an influenza epidemic. *Service Science*, 5(3):197–215, 2013.

[164] Joel A Tropp. User-friendly tail bounds for sums of random matrices. *Foundations of computational mathematics*, 12(4):389–434, 2012.

[165] James Turtle, Michal Ben-Nun, and Pete Riley. Psi-draft, October 2020.

[166] Andrés Uribe-Sánchez, Alex Savachkin, Alfredo Santana, Diana Prieto-Santa, and Tapas K Das. A predictive decision-aid methodology for dynamic mitigation of influenza pandemics. *OR spectrum*, 33(3):751–786, 2011.

[167] US Census Bureau. County Population by Characteristics: 2010-2019. https://www.census.gov/data/tables/time-series/demo/popest/2010s-counties-detail.html, June 2020.

[168] US Center for Disease Control. COVID-19 Forecasts. https://www.cdc.gov/coronavirus/2019-ncov/covid-data/forecasting-us.html, 2020.

[169] US Center for Disease Control. COVID-19 Hospitalization and Death by Age. https://www.cdc.gov/coronavirus/2019-ncov/covid-data/investigations-discovery/hospitalization-death-by-age.html, August 2020.

[170] US Center for Disease Control. Provisional Death Counts for Coronavirus Disease 2019 (COVID-19). https://www.cdc.gov/nchs/nvss/vsrr/covid_weekly, February 2021.

[171] YV Volkov and SK Zavriev. A general stochastic outer approximations method. *SIAM journal on control and optimization*, 35(4):1387–1421, 1997.

[172] Chaolong Wang, Li Liu, Xingjie Hao, Huan Guo, Qi Wang, Jiao Huang, Na He, Hongjie Yu, Xihong Lin, An Pan, et al. Evolving epidemiology and impact of non-pharmaceutical interventions on the outbreak of coronavirus disease 2019 in wuhan, china. *medRxiv*, 2020.

[173] Raymond Keith Watson. On an epidemic in a stratified population. *Journal of Applied Probability*, 9(3):659–666, 1972.

[174] Jing Wei and Matthew J Realff. Sample average approximation methods for stochastic minlps. *Computers & Chemical Engineering*, 28(3):333–346, 2004.

[175] Jacqui Wise. Covid-19: Surveys indicate low infection level in community, 2020.

[176] Max A Woodbury. The stability of out-input matrices. *Chicago, IL*, 93, 1949.

[177] Max A Woodbury. Inverting modified matrices. *Memorandum report*, 42(106):336, 1950.

[178] Spencer Woody, Mauricio Garcia Tec, Maytal Dahan, Kelly Gaither, Michael Lachmann, Spencer Fox, Lauren Ancel Meyers, and James G Scott. Projections for first-wave covid-19 deaths across the us using social-distancing measures derived from mobile phones. *medRxiv*, 2020.

[179] Joseph T Wu, Lawrence M Wein, and Alan S Perelson. Optimization of influenza vaccine selection. *Operations Research*, 53(3):456–476, 2005.

[180] Y Xiang, DY Sun, W Fan, and XG Gong. Generalized simulated annealing algorithm and its application to the thomson model. *Physics Letters A*, 233(3):216–220, 1997.

[181] Hui Xu, Sufang Huang, Shangkun Liu, Juan Deng, Bo Jiao, Ling Ai, Yaru Xiao, Li Yan, and Shusheng Li. Evaluation of the clinical characteristics of suspected or confirmed cases of covid-19 during home care with isolation: A new retrospective analysis based on o2o. *Available at SSRN 3548746*, 2020.

[182] Miao Xu, Rong Jin, and Zhi-Hua Zhou. Speedup matrix completion with side information: Application to multi-label learning. In *Advances in neural information processing systems*, pages 2301–2309, 2013.

[183] Hamed Yarmand, Julie S Ivy, Brian Denton, and Alun L Lloyd. Optimal two-phase vaccine allocation to geographically different regions under uncertainty. *European Journal of Operational Research*, 233(1):208–219, 2014.

[184] Sheng Zhang, MengYuan Diao, Wenbo Yu, Lei Pei, Zhaofen Lin, and Dechang Chen. Estimation of the reproductive number of novel coronavirus (covid-19) and the probable outbreak size on the diamond princess cruise ship: A data-driven analysis. *International Journal of Infectious Diseases*, 93:201–204, 2020.

[185] Qinqing Zheng and John Lafferty. Convergence analysis for rectangular matrix completion using burer-monteiro factorization and gradient descent. *arXiv preprint arXiv:1605.07051*, 2016.

Appendix A

Appendix to Chapter 2

A.1 Proof of Theorem 3

We want to prove that our objective function $c(\boldsymbol{S})$ is a (θ, ζ, η)-strict saddle. Furthermore, we want to prove that there are no spurious local minima.

To do so, we claim that proving the following statement would imply both results.

"For all $\boldsymbol{S}$ such that $\|\nabla c(\boldsymbol{S})\| \leq \epsilon$ and $\boldsymbol{S}$ more than η-away from a *global* minimum, we have that $\lambda_{\min}(\nabla^2 f(\boldsymbol{x})) \leq -\zeta$ for some suitable ζ."

By construction, this statement shows that for $c(\boldsymbol{S})$, if Condition 1 and 3 of the strict saddle definition is not true, then 2 is always true, which implies $c(\boldsymbol{S})$ is (θ, ζ, η)-strict saddle.

Furthermore, this statement implies that every approximate stationary point (which satisfies $\|\nabla c(\boldsymbol{S})\| \leq \epsilon$) that is not a global minimum has a negative eigenvalue in its Hessian. Thus, every stationary point that is not a global minimum is a saddle point, since local minima have no negative eigenvalues in the Hessian. Therefore, there are no spurious local minima - every local minimum is a global minimum.

Thus, proving this statement would give us the two required results we need. Therefore, we now set out to prove the statement above.

To ease the notation burden, we define the following :

$$\boldsymbol{S}_i = \boldsymbol{W}_i \boldsymbol{S},$$
$$\boldsymbol{P}_{\boldsymbol{S}} = \boldsymbol{I}_m - \boldsymbol{S}(\boldsymbol{S}^T \boldsymbol{S})^{-1} \boldsymbol{S}^T,$$

$$\boldsymbol{P}_{\boldsymbol{S}_i} = \boldsymbol{W}_i - \boldsymbol{S}_i(\boldsymbol{S}_i^T\boldsymbol{S}_i)^{-1}\boldsymbol{S}_i^T.$$

Note that the projection matrices are setup so that $\boldsymbol{P}_{\boldsymbol{S}_i}\boldsymbol{S}_i = \boldsymbol{0}$ and $\boldsymbol{P}_{\boldsymbol{S}}\boldsymbol{S} = \boldsymbol{0}$. Furthermore, let us denote $\boldsymbol{U}^* \in \mathbb{R}^{n\times k}, \boldsymbol{S}^* \in \mathbb{R}^{m\times k}$ as the true solution to the matrix completion problem (i.e., $\boldsymbol{A} = \boldsymbol{U}^*\boldsymbol{S}^{*T}$). Then we can write

$$c(\boldsymbol{S}) = \frac{1}{nm}\sum_{i=1}^{n} \boldsymbol{a}_i\boldsymbol{P}_{\boldsymbol{S}_i}\overline{\boldsymbol{a}}_i^T. \tag{A.1}$$

We calculate its gradient as follows.

Lemma 4.

$$\nabla c(\boldsymbol{S}) = \frac{1}{nm}\sum_{i=1}^{n} \boldsymbol{P}_{\boldsymbol{S}_i}\boldsymbol{a}_i^T\boldsymbol{a}_i\boldsymbol{S}_i(\boldsymbol{S}_i^T\boldsymbol{S}_i)^{-1}.$$

Proof. This follows from direct calculation of the derivative. In particular, we note that $\boldsymbol{S}^T\boldsymbol{P}_{\boldsymbol{S}_i} = \boldsymbol{0}$ for all i by definition, and thus *every term* of the derivative in the sum is orthogonal to $\boldsymbol{S}$. □

To calculate the Hessian, note that we expressed $\boldsymbol{S}$ as a $m \times k$ matrix, so the Hessian has dimensions $(m \times k) \times (m \times k)$. Therefore, to calculate the second derivative in a specific direction $\boldsymbol{M}$, we use the following notation to represent the quadratic form:

$$\boldsymbol{M} : \nabla^2 c(\boldsymbol{S}) : \boldsymbol{M} = \sum_{i,j,k,l} M_{ij}\nabla^2 c(\boldsymbol{S})_{ijkl}M_{kl}.$$

Where $\nabla^2 c(\boldsymbol{S})_{ijkl} = \frac{\partial^2 c(\boldsymbol{S})}{\partial \boldsymbol{S}_{ij}\partial \boldsymbol{S}_{kl}}$.

Lemma 5. *For any matrix $\boldsymbol{M} \in \mathbb{R}^{m\times k}$, the Hessian in the direction of $\boldsymbol{M}$ is*

$$\begin{aligned}\boldsymbol{M} : \nabla^2 c(\boldsymbol{S}) : \boldsymbol{M} = \frac{2}{nm}\sum_{i=1}^{n}\Big(&\boldsymbol{a}_i\boldsymbol{S}_i(\boldsymbol{S}_i^T\boldsymbol{S}_i)^{-1}\boldsymbol{M}^T\boldsymbol{P}_{\boldsymbol{S}_i}\boldsymbol{M}(\boldsymbol{S}_i^T\boldsymbol{S}_i)^{-1}\boldsymbol{a}_i^T \\ &+ 2\boldsymbol{a}_i\boldsymbol{S}_i(\boldsymbol{S}_i^T\boldsymbol{S}_i)^{-1}\boldsymbol{M}^T\boldsymbol{S}_i(\boldsymbol{S}_i^T\boldsymbol{S}_i)^{-1}\boldsymbol{M}^T\boldsymbol{P}_{\boldsymbol{S}_i}\boldsymbol{a}_i^T \\ &- \overline{\boldsymbol{a}}_i\boldsymbol{M}(\boldsymbol{S}_i^T\boldsymbol{S}_i)^{-1}\boldsymbol{M}^T\boldsymbol{P}_{\boldsymbol{S}_i}\boldsymbol{a}_i^T\end{aligned}$$

$$+ \boldsymbol{a}_i \boldsymbol{S}_i (\boldsymbol{S}_i^T \boldsymbol{S}_i)^{-1} \boldsymbol{S}_i^T \boldsymbol{M} (\boldsymbol{S}_i^T \boldsymbol{S}_i)^{-1} \boldsymbol{M}^T \boldsymbol{P}_{\boldsymbol{S}_i} \boldsymbol{a}_i^T \Bigg).$$

Proof. The result follows through by direct calculation. □

As a reminder, our goal is to prove that, given $\|\nabla c(\boldsymbol{S})\| \leq \epsilon$ and $\|\boldsymbol{S} - \boldsymbol{S}^*\| \geq \eta$ for any global minima $\boldsymbol{S}^*$, there exists a negative eigenvalue for the Hessian $\nabla^2 c(\boldsymbol{S})$.

Note the existence of a negative eigenvalue for the Hessian $\nabla^2 c(\boldsymbol{S})$ is equivalent to existence of a matrix $\boldsymbol{M}$ such that $\boldsymbol{M} : \nabla^2 c(\boldsymbol{S}) : \boldsymbol{M} < 0$. Now let us look at the expression for the second derivative.

$$\begin{aligned}
\boldsymbol{M} : \nabla^2 c(\boldsymbol{S}) : \boldsymbol{M} = \frac{2}{nm} \sum_{i=1}^{n} \Bigg(& \underbrace{\boldsymbol{a}_i \boldsymbol{S}_i (\boldsymbol{S}_i^T \boldsymbol{S}_i)^{-1} \boldsymbol{M}^T \boldsymbol{P}_{\boldsymbol{S}_i} \boldsymbol{M} (\boldsymbol{S}_i^T \boldsymbol{S}_i)^{-1} \boldsymbol{S}_i^T \boldsymbol{a}_i^T}_{\text{(T1)}} \\
& + \underbrace{2 \boldsymbol{a}_i \boldsymbol{S}_i (\boldsymbol{S}_i^T \boldsymbol{S}_i)^{-1} \boldsymbol{M}^T \boldsymbol{S}_i (\boldsymbol{S}_i^T \boldsymbol{S}_i)^{-1} \boldsymbol{M}^T \boldsymbol{P}_{\boldsymbol{S}_i} \boldsymbol{a}_i^T}_{\text{(T2)}} \\
& - \underbrace{\overline{\boldsymbol{a}}_i \boldsymbol{M} (\boldsymbol{S}_i^T \boldsymbol{S}_i)^{-1} \boldsymbol{M}^T \boldsymbol{P}_{\boldsymbol{S}_i} \boldsymbol{a}_i^T}_{\text{(T3)}} \\
& + \underbrace{\boldsymbol{a}_i \boldsymbol{S}_i (\boldsymbol{S}_i^T \boldsymbol{S}_i)^{-1} \boldsymbol{S}_i^T \boldsymbol{M} (\boldsymbol{S}_i^T \boldsymbol{S}_i)^{-1} \boldsymbol{M}^T \boldsymbol{P}_{\boldsymbol{S}_i} \boldsymbol{a}_i^T}_{\text{(T4)}} \Bigg).
\end{aligned}$$

We first prove that for *any* $\boldsymbol{M}$, the terms (T2) and (T4) are small. Then, we exhibit a specific $\boldsymbol{M}$, such that (T1) is close to 0, (T3) is large and positive, and so that the resulting quadratic form $\boldsymbol{M} : \nabla^2 c(\boldsymbol{S}) : \boldsymbol{M}$ is negative.

Now let us prove that terms (T2) and (T4) are small (on the order of $O(\epsilon)$) for any $\boldsymbol{M}$.

Lemma 6. *Let $\boldsymbol{S}$ satisfy Assumptions 1-2, and be such that $\|\nabla c(\boldsymbol{S})\| \leq \epsilon$. Then we have that, for $r > \frac{Ck}{m}$ for sufficiently large C, with probability at least $1 - O(\frac{1}{n})$:*

$$\begin{aligned}
\left\| \frac{1}{nm} \sum_{i=1}^{n} \boldsymbol{a}_i \boldsymbol{S}_i (\boldsymbol{S}_i^T \boldsymbol{S}_i)^{-1} \boldsymbol{M}^T \boldsymbol{S}_i (\boldsymbol{S}_i^T \boldsymbol{S}_i)^{-1} \boldsymbol{M}^T \boldsymbol{P}_{\boldsymbol{S}_i} \boldsymbol{a}_i^T \right\| &\leq Kr\epsilon \|\boldsymbol{M}\|^2, \\
\left\| \frac{1}{nm} \sum_{i=1}^{n} \boldsymbol{a}_i \boldsymbol{S}_i (\boldsymbol{S}_i^T \boldsymbol{S}_i)^{-1} \boldsymbol{S}_i^T \boldsymbol{M} (\boldsymbol{S}_i^T \boldsymbol{S}_i)^{-1} \boldsymbol{M}^T \boldsymbol{P}_{\boldsymbol{S}_i} \boldsymbol{a}_i^T \right\| &\leq Lr\epsilon \|\boldsymbol{M}\|^2,
\end{aligned}$$

for some absolute constants K, L.

Proof. Note that we are trying to bound the second derivative when the first derivative is approximately 0. Using Lemma 4, we can rewrite $\|\nabla c(\boldsymbol{S})\| \leq \epsilon$ as

$$\left\| \frac{1}{nm} \sum_{i=1}^{n} \boldsymbol{a}_i \boldsymbol{S}_i (\boldsymbol{S}_i^T \boldsymbol{S}_i)^{-1} \boldsymbol{M}^T \boldsymbol{P}_{\boldsymbol{S}_i} \boldsymbol{a}_i^T \right\| \leq \epsilon \|\boldsymbol{M}\|. \tag{A.2}$$

We only prove for term (T2) as the proof for term (T4) follows the same steps. First, we bound the difference between the first derivative in (A.2) and its "non-stochastic" version:

$$\left\| \frac{1}{nm} \sum_{i=1}^{n} \boldsymbol{a}_i \boldsymbol{S}_i (\boldsymbol{S}_i^T \boldsymbol{S}_i)^{-1} \boldsymbol{M}^T \boldsymbol{P}_{\boldsymbol{S}_i} \boldsymbol{a}_i^T - \frac{r}{nm} \sum_{i=1}^{n} \boldsymbol{a}_i \boldsymbol{S} (\boldsymbol{S}^T \boldsymbol{S})^{-1} \boldsymbol{M}^T \boldsymbol{P}_{\boldsymbol{S}} \boldsymbol{a}_i^T \right\|. \tag{A.3}$$

Then we focus on each individual term inside the sum (taking $\frac{1}{m}$ inside). We have

$$\begin{aligned}
&\left\| \frac{\boldsymbol{a}_i \boldsymbol{S}_i (\boldsymbol{S}_i^T \boldsymbol{S}_i)^{-1} \boldsymbol{M}^T \boldsymbol{P}_{\boldsymbol{S}_i} \boldsymbol{a}_i^T}{m} - \frac{l}{m} \boldsymbol{a}_i \boldsymbol{S} (\boldsymbol{S}^T \boldsymbol{S})^{-1} \boldsymbol{M}^T \boldsymbol{P}_{\boldsymbol{S}} \boldsymbol{a}_i^T \right\| \\
&\leq r \left\| \left(\frac{\boldsymbol{a}_i \boldsymbol{S}_i}{l} - \frac{\boldsymbol{a}_i \boldsymbol{S}}{m} \right) \left(\frac{\boldsymbol{S}_i^T \boldsymbol{S}_i}{l} \right)^{-1} \frac{\boldsymbol{M}^T \boldsymbol{P}_{\boldsymbol{S}_i} \boldsymbol{a}_i^T}{l} \right\| \\
&+ r \left\| \frac{\boldsymbol{a}_i \boldsymbol{S}}{m} \left(\left(\frac{\boldsymbol{S}_i^T \boldsymbol{S}_i}{l} \right)^{-1} - \left(\frac{\boldsymbol{S}^T \boldsymbol{S}}{m} \right)^{-1} \right) \frac{\boldsymbol{M}^T \boldsymbol{P}_{\boldsymbol{S}_i} \boldsymbol{a}_i^T}{l} \right\| \\
&+ r \left\| \frac{\boldsymbol{a}_i \boldsymbol{S}}{m} \left(\frac{\boldsymbol{S}^T \boldsymbol{S}}{m} \right)^{-1} \left(\frac{\boldsymbol{M}^T \boldsymbol{P}_{\boldsymbol{S}_i} \boldsymbol{a}_i^T}{l} - \frac{\boldsymbol{M}^T \boldsymbol{P}_{\boldsymbol{S}} \boldsymbol{a}_i^T}{m} \right) \right\|.
\end{aligned}$$

We utilize Hoeffding's theorem for the first and third term, along with Lemma 11 for the second term to obtain that with probability $1 - \delta$ the sum of the three terms is less than or equal to $r\sqrt{\frac{Dk \log(\frac{1}{\delta})}{l}} \|\boldsymbol{M}\|$, where D is an absolute constant.

Then, we have, by Lemma 10, with probability $1 - \delta$

$$\begin{aligned}
&\left\| \frac{1}{nm} \sum_{i=1}^{n} \boldsymbol{a}_i \boldsymbol{S}_i (\boldsymbol{S}_i^T \boldsymbol{S}_i)^{-1} \boldsymbol{M}^T \boldsymbol{P}_{\boldsymbol{S}_i} \boldsymbol{a}_i^T - \frac{l}{nm} \sum_{i=1}^{n} \boldsymbol{a}_i \boldsymbol{S} (\boldsymbol{S}^T \boldsymbol{S})^{-1} \boldsymbol{M}^T \boldsymbol{P}_{\boldsymbol{S}} \boldsymbol{a}_i^T \right\| \\
&\leq r \sqrt{\frac{D' k \log(\frac{1}{\delta})}{ln}} \|\boldsymbol{M}\|,
\end{aligned}$$

where D' is an absolute constant. We let $\delta = \frac{1}{n}$, and $l \geq Ck$ for some constant C. Then, for

n sufficiently large, we have

$$r\sqrt{\frac{D'k\log(\frac{1}{\delta})}{ln}}\|\boldsymbol{M}\| \leq r\epsilon\|\boldsymbol{M}\|.$$

Therefore, we know that, with probability at least $1-\frac{1}{n}$, we have

$$\left\|\frac{l}{nm}\sum_{i=1}^{n}\boldsymbol{a}_i\boldsymbol{S}(\boldsymbol{S}^T\boldsymbol{S})^{-1}\boldsymbol{M}^T\boldsymbol{P}_{\boldsymbol{S}}\boldsymbol{a}_i^T\right\| \leq 2r\epsilon\|\boldsymbol{M}\|, \tag{A.4}$$

for any $\boldsymbol{M}$. Now we do the same for term (T2) by bounding the difference between term (T2) and its non-stochastic variant. The details are omitted as they follow the same logic as above. The final result is that with probability at least $1-\frac{1}{n}$, we have

$$\begin{aligned}\|\frac{1}{nm}\sum_{i=1}^{n}&\boldsymbol{a}_i\boldsymbol{S}_i(\boldsymbol{S}_i^T\boldsymbol{S}_i)^{-1}\boldsymbol{M}^T\boldsymbol{S}_i(\boldsymbol{S}_i^T\boldsymbol{S}_i)^{-1}\boldsymbol{M}^T\boldsymbol{P}_{\boldsymbol{S}_i}\boldsymbol{a}_i^T\\ &\underbrace{-\frac{l}{nm}\sum_{i=1}^{n}\boldsymbol{a}_i\boldsymbol{S}(\boldsymbol{S}^T\boldsymbol{S})^{-1}\boldsymbol{M}^T\boldsymbol{S}(\boldsymbol{S}^T\boldsymbol{S})^{-1}\boldsymbol{M}^T\boldsymbol{P}_{\boldsymbol{S}}\boldsymbol{a}_i^T}_{\text{(S1)}}\| \leq r\epsilon\|\boldsymbol{M}\|^2.\end{aligned} \tag{A.5}$$

Now let us focus on the term (S1). Such term is exactly (A.4) with the only difference being substituting $\boldsymbol{M}$ with $\boldsymbol{M}^T\boldsymbol{S}(\boldsymbol{S}^T\boldsymbol{S})^{-1}\boldsymbol{M}^T$. Therefore, we have

$$\left\|\frac{l}{nm}\sum_{i=1}^{n}\boldsymbol{a}_i\boldsymbol{S}(\boldsymbol{S}^T\boldsymbol{S})^{-1}\boldsymbol{M}^T\boldsymbol{S}(\boldsymbol{S}^T\boldsymbol{S})^{-1}\boldsymbol{M}^T\boldsymbol{P}_{\boldsymbol{S}}\boldsymbol{a}_i^T\right\| \leq Kr\epsilon\|\boldsymbol{M}\|^2, \tag{A.6}$$

for some constant K. Then combining (A.5) and (A.6) gives the required result. □

We next utilize the bounds on (T2) and (T4) to explicitly construct a $\boldsymbol{M}$ to bound (T1) and (T3). Specifically, we define $\boldsymbol{M} = \boldsymbol{z}^T\boldsymbol{x}$, $\|\boldsymbol{M}\| = 1$, where $\boldsymbol{z}^T \in \mathbb{R}^{m\times 1}$ is a non-zero (rescaled) column of $\boldsymbol{P}_{\boldsymbol{S}}\boldsymbol{S}^*$ (it exists because $\boldsymbol{S}$ is not optimal, so $\boldsymbol{P}_{\boldsymbol{S}}\boldsymbol{S}^* \neq \boldsymbol{0}$) and $\boldsymbol{x}^T \in \mathbb{R}^{k\times 1}$ is an eigenvector of $\boldsymbol{S}^{*T}\boldsymbol{S}(\boldsymbol{S}^T\boldsymbol{S})^{-1}$ that has an eigenvalue of size $O(\epsilon)$. We utilize the row vector convention to be consistent with our other notations. We prove the existence of the eigenvector $\boldsymbol{x}^T$ in Lemma 7 below.

Lemma 7. *Let $\boldsymbol{S}$ satisfy Assumptions 1-2, and be such that $\|\nabla c(\boldsymbol{S})\| \leq \epsilon$ for some ϵ*

sufficiently small. Assume that $\min_{\boldsymbol{R}^T\boldsymbol{R}=\boldsymbol{I}} \|\boldsymbol{S} - \boldsymbol{S}^*\boldsymbol{R}\| \geq \eta$. *Then* $\boldsymbol{S}^{*T}\boldsymbol{S}(\boldsymbol{S}^T\boldsymbol{S})^{-1}$ *has an eigenvector with an eigenvalue of size* $O(\epsilon)$.

Proof. First it is easy to see that

$$\min_{\boldsymbol{R}^T\boldsymbol{R}=\boldsymbol{I}} \|\boldsymbol{S}^* - \boldsymbol{S}\boldsymbol{R}\| = \|\boldsymbol{P}_{\boldsymbol{S}}\boldsymbol{S}^*\| \geq \eta. \tag{A.7}$$

Now, let us consider the first derivative bound from (A.4):

$$\left\| \frac{1}{nm} \sum_{i=1}^{n} \boldsymbol{a}_i \boldsymbol{S}(\boldsymbol{S}^T\boldsymbol{S})^{-1}\boldsymbol{M}^T\boldsymbol{P}_{\boldsymbol{S}}\boldsymbol{a}_i^T \right\| \leq 2r\epsilon\|\boldsymbol{M}\|.$$

We prove that given this, there exists an eigenvector of $\boldsymbol{S}^{*T}\boldsymbol{S}(\boldsymbol{S}^T\boldsymbol{S})^{-1}$ that has an eigenvalue of size $O(\epsilon)$. Assume that there is not. By definition $\boldsymbol{a}_i = \boldsymbol{u}_i^*\boldsymbol{S}^{*T}$, so we can write the bound as:

$$\left\| \frac{1}{nm} \sum_{i=1}^{n} \boldsymbol{u}_i^*\boldsymbol{S}^{*T}\boldsymbol{S}(\boldsymbol{S}^T\boldsymbol{S})^{-1}\boldsymbol{M}^T\boldsymbol{P}_{\boldsymbol{S}}\boldsymbol{S}^*\boldsymbol{u}_i^{*T} \right\| \leq 2r\epsilon\|\boldsymbol{M}\|. \tag{A.8}$$

Then since there is not an eigenvalue of magnitude $O(\epsilon)$, let the smallest eigenvalue of $\boldsymbol{S}^{*T}\boldsymbol{S} \times (\boldsymbol{S}^T\boldsymbol{S})^{-1}$ have a magnitude of $O(\gamma) \gg O(\epsilon)$. Let $\boldsymbol{x}^T \in \mathbb{R}^{k\times 1}$ be the eigenvector associated with such eigenvalue. Then $\boldsymbol{S}^{*T}\boldsymbol{S}(\boldsymbol{S}^T\boldsymbol{S})^{-1}\boldsymbol{x}^T = O(\gamma)\boldsymbol{x}^T$. From Equation (A.7) we know that $\|\boldsymbol{P}_{\boldsymbol{S}}\boldsymbol{S}^*\| \geq \eta$, so let $\boldsymbol{y} \in \mathbb{R}^{1\times m}$ be such that $\boldsymbol{y}\boldsymbol{P}_{\boldsymbol{S}}\boldsymbol{S}^* = O(\eta)\boldsymbol{v}^T$ where $\boldsymbol{v} \in \mathbb{R}^{1\times k}$, and rescaled such that for $\boldsymbol{M} = \boldsymbol{y}^T\boldsymbol{x}$, we have $\|\boldsymbol{M}\| = O(1)$. Then we have

$$\left\| \frac{1}{nm} \sum_{i=1}^{n} \boldsymbol{u}_i^*\boldsymbol{S}^{*T}\boldsymbol{S}(\boldsymbol{S}^T\boldsymbol{S})^{-1}\boldsymbol{M}^T\boldsymbol{P}_{\boldsymbol{S}}\boldsymbol{S}^*\boldsymbol{u}_i^{*T} \right\| = \left\| \frac{O(\eta\gamma)}{nm} \sum_{i=1}^{n} \boldsymbol{u}_i^*\boldsymbol{x}\boldsymbol{v}^T\boldsymbol{u}_i^{*T} \right\| = O\left(\frac{\eta\gamma}{m}\right) \leq O(\epsilon),$$

where the last equality follows from the submatrix full-rank property of $\boldsymbol{U}^*$ (note that $\boldsymbol{u}_i^* \in \mathbb{R}^{1\times k}$ are row vectors here). Since η is independent from ϵ (η only depends on how far $\boldsymbol{S}$ is from $\boldsymbol{S}^*$) and $\gamma \gg \epsilon$ by setup, we obtain a contradiction for sufficiently small ϵ.

Therefore, there exists ϵ sufficiently small such that $\boldsymbol{S}^{*T}\boldsymbol{S}(\boldsymbol{S}^T\boldsymbol{S})^{-1}$ has an eigenvector $\boldsymbol{x}^T$, with an eigenvalue of size $O(\epsilon)$. □

Having proved that the desired matrix $\boldsymbol{M}$ does indeed exist, we show that indeed with such $\boldsymbol{M}$, (T1) is small, and (T3) is large and positive. We first bound (T1).

Lemma 8. *Let $\boldsymbol{S}$ satisfy Assumptions 1-2, and be such that $\|\nabla c(\boldsymbol{S})\| \leq \epsilon$ for some ϵ sufficiently small. Assume that $\min_{\boldsymbol{R}^T\boldsymbol{R}=\boldsymbol{I}} \|\boldsymbol{S} - \boldsymbol{S}^*\boldsymbol{R}\| \geq \eta$. Then for $\boldsymbol{M} = \boldsymbol{z}^T\boldsymbol{x}$ where $\boldsymbol{z}$, $\boldsymbol{x}$ defined above, with probability at least $1 - O(\frac{1}{n})$:*

$$\left\| \frac{1}{nm} \sum_{i=1}^{n} \boldsymbol{a}_i \boldsymbol{S}_i (\boldsymbol{S}_i^T \boldsymbol{S}_i)^{-1} \boldsymbol{M}^T \boldsymbol{P}_{\boldsymbol{S}_i} \boldsymbol{M} (\boldsymbol{S}_i^T \boldsymbol{S}_i)^{-1} \boldsymbol{S}_i^T \boldsymbol{a}_i^T \right\| \leq O(r\epsilon),$$

Proof. From Lemma 7, we know that there is an eigenvector of $\boldsymbol{S}^{*T}\boldsymbol{S}(\boldsymbol{S}^T\boldsymbol{S})^{-1}$, $\boldsymbol{x}^T$, that has an eigenvalue of size $O(\epsilon)$. Similar to the proof for Lemma 6, we can bound term (T1) with the non-stochastic version with at least probability $1 - \frac{1}{n}$:

$$\left\| \frac{1}{nm} \sum_{i=1}^{n} \boldsymbol{a}_i \boldsymbol{S}_i (\boldsymbol{S}_i^T \boldsymbol{S}_i)^{-1} \boldsymbol{M}^T \boldsymbol{P}_{\boldsymbol{S}_i} \boldsymbol{M} (\boldsymbol{S}_i^T \boldsymbol{S}_i)^{-1} \boldsymbol{S}_i^T \boldsymbol{a}_i^T \right.$$
$$\left. - \underbrace{\frac{r}{n} \sum_{i=1}^{n} \boldsymbol{a}_i \boldsymbol{S} (\boldsymbol{S}^T \boldsymbol{S})^{-1} \boldsymbol{M}^T \boldsymbol{P}_{\boldsymbol{S}} \boldsymbol{M} (\boldsymbol{S}^T \boldsymbol{S})^{-1} \boldsymbol{S}^T \boldsymbol{a}_i^T}_{\text{(S2)}} \right\| \leq r\epsilon \|\boldsymbol{M}\|^2. \tag{A.9}$$

Let us look at each term in the sum (S2). Taking $\boldsymbol{M} = \boldsymbol{z}^T\boldsymbol{x}$. We have

$$\begin{aligned}
&\|\boldsymbol{a}_i \boldsymbol{S} (\boldsymbol{S}^T \boldsymbol{S})^{-1} \boldsymbol{M}^T \boldsymbol{P}_{\boldsymbol{S}} \boldsymbol{M} (\boldsymbol{S}^T \boldsymbol{S})^{-1} \boldsymbol{S}^T \boldsymbol{a}_i^T\| \\
&= \|\boldsymbol{P}_{\boldsymbol{S}} \boldsymbol{M} (\boldsymbol{S}^T \boldsymbol{S})^{-1} \boldsymbol{S}^T \boldsymbol{a}_i^T\|^2 \\
&= \|\boldsymbol{P}_{\boldsymbol{S}} \boldsymbol{M} (\boldsymbol{S}^T \boldsymbol{S})^{-1} \boldsymbol{S}^T \boldsymbol{S}^* \boldsymbol{u}_i^{*T}\|^2 \\
&= \|\boldsymbol{P}_{\boldsymbol{S}} \boldsymbol{z}^T \boldsymbol{x} (\boldsymbol{S}^T \boldsymbol{S})^{-1} \boldsymbol{S}^T \boldsymbol{S}^* \boldsymbol{u}_i^{*T}\|^2 \\
&= O(\epsilon^2) \|\boldsymbol{z}^T \boldsymbol{x} \boldsymbol{u}_i^{*T}\|^2 \\
&= O(\epsilon^2),
\end{aligned}$$

where in the last equality we used the identity that $\boldsymbol{P}_{\boldsymbol{S}} \boldsymbol{z}^T = \boldsymbol{z}^T$. Then, we can bound the sum (S2) as

$$\left\| \frac{r}{n} \sum_{i=1}^{n} \boldsymbol{a}_i \boldsymbol{S} (\boldsymbol{S}^T \boldsymbol{S})^{-1} \boldsymbol{M}^T \boldsymbol{P}_{\boldsymbol{S}} \boldsymbol{M} (\boldsymbol{S}^T \boldsymbol{S})^{-1} \boldsymbol{S}^T \boldsymbol{a}_i^T \right\|$$

$$\leq \frac{r}{n}\sum_{i=1}^{n}\|\boldsymbol{a}_i\boldsymbol{S}(\boldsymbol{S}^T\boldsymbol{S})^{-1}\boldsymbol{M}^T\boldsymbol{P}_{\boldsymbol{S}}\boldsymbol{M}(\boldsymbol{S}^T\boldsymbol{S})^{-1}\boldsymbol{S}^T\boldsymbol{a}_i^T\|$$
$$= O(r\epsilon^2).$$

Substituting such bound on equation (S2) into (A.9), we have that

$$\left\|\frac{1}{nm}\sum_{i=1}^{n}\boldsymbol{a}_i\boldsymbol{S}_i(\boldsymbol{S}_i^T\boldsymbol{S}_i)^{-1}\boldsymbol{M}^T\boldsymbol{P}_{\boldsymbol{S}_i}\boldsymbol{M}(\boldsymbol{S}_i^T\boldsymbol{S}_i)^{-1}\boldsymbol{S}_i^T\boldsymbol{a}_i^T\right\| \leq O(r\epsilon).$$

As here we have $\|\boldsymbol{M}\| = O(1)$. □

Now, finally, we prove that (T3) is large under such $\boldsymbol{M}$.

Lemma 9. *Let $\boldsymbol{S}$ satisfy Assumptions 1-2. Further assume that $\min_{\boldsymbol{R}^T\boldsymbol{R}=\boldsymbol{I}}\|\boldsymbol{S}^* - \boldsymbol{S}\boldsymbol{R}\| \geq \eta$ and $\|\nabla c(\boldsymbol{S})\| \leq \epsilon$ for some ϵ sufficiently small. Then for $\boldsymbol{M} = \boldsymbol{z}^T\boldsymbol{x}$ where $\boldsymbol{z}$, $\boldsymbol{x}$ defined above, with probability at least $1 - O(\frac{1}{n})$:*

$$\frac{1}{nm}\sum_{i=1}^{n}\boldsymbol{a}_i\boldsymbol{M}(\boldsymbol{S}_i^T\boldsymbol{S}_i)^{-1}\boldsymbol{M}^T\boldsymbol{P}_{\boldsymbol{S}_i}\boldsymbol{a}_i^T \geq O\left(\frac{r\eta^2}{k}\right),$$

Proof. First it is easy to see that

$$\min_{\boldsymbol{R}^T\boldsymbol{R}=\boldsymbol{I}}\|\boldsymbol{S}^* - \boldsymbol{S}\boldsymbol{R}\| = \|\boldsymbol{P}_{\boldsymbol{S}}\boldsymbol{S}^*\| \geq \eta. \tag{A.10}$$

We consider the term $\frac{1}{nm}\sum_{i=1}^{n}\boldsymbol{a}_i\boldsymbol{M}(\boldsymbol{S}^T\boldsymbol{S})^{-1}\boldsymbol{M}^T\boldsymbol{P}_{\boldsymbol{S}}\boldsymbol{a}_i^T$. We have that

$$\frac{l}{nm}\sum_{i=1}^{n}\boldsymbol{a}_i\boldsymbol{M}(\boldsymbol{S}^T\boldsymbol{S})^{-1}\boldsymbol{M}^T\boldsymbol{P}_{\boldsymbol{S}}\boldsymbol{a}_i^T$$
$$= \frac{l}{nm}\sum_{i=1}^{n}\boldsymbol{a}_i\boldsymbol{z}^T\boldsymbol{x}(\boldsymbol{S}^T\boldsymbol{S})^{-1}\boldsymbol{x}^T\boldsymbol{z}\boldsymbol{a}_i^T,$$

where we utilized $\boldsymbol{z}\boldsymbol{P}_{\boldsymbol{S}} = \boldsymbol{z}$. By definition $\boldsymbol{a}_i^T = \boldsymbol{S}^*\boldsymbol{u}_i^{*T}$, so we have

$$\frac{l}{nm}\sum_{i=1}^{n}\boldsymbol{a}_i\boldsymbol{z}^T\boldsymbol{x}(\boldsymbol{S}^T\boldsymbol{S})^{-1}\boldsymbol{x}^T\boldsymbol{z}\boldsymbol{a}_i^T = \frac{r}{n}\sum_{i=1}^{n}\boldsymbol{u}_i^*\boldsymbol{S}^{*T}\boldsymbol{z}^T\boldsymbol{x}(\boldsymbol{S}^T\boldsymbol{S})^{-1}\boldsymbol{x}^T\boldsymbol{z}\boldsymbol{S}^*\boldsymbol{u}_i^{*T}.$$

Since $(\boldsymbol{S}^T\boldsymbol{S})^{-1}$ is positive definite (PD), we have a sum of quadratic forms over a PD matrix, which is positive as long as $\boldsymbol{x}^T\boldsymbol{z}\boldsymbol{S}^*\boldsymbol{u}_i^{*T}$ is non-zero. Since we have that $\boldsymbol{z}$ is a non-zero column vector of $\boldsymbol{P}_{\boldsymbol{S}}\boldsymbol{S}^*$, and we know that $\|(\boldsymbol{P}_{\boldsymbol{S}}\boldsymbol{S}^*)^T\boldsymbol{S}^*\| = \|\boldsymbol{S}^{*T}\boldsymbol{P}_{\boldsymbol{S}}\boldsymbol{S}^*\|^2 \geq O(\eta^2)$, so in particular, we have that $\|\boldsymbol{x}^T\boldsymbol{z}\boldsymbol{S}^*\| \geq O(\eta^2)$. Thus, we only need $\boldsymbol{u}_i^{*T}$ to not always be in the null space of $\boldsymbol{x}^T\boldsymbol{z}\boldsymbol{S}^*$. Now by the sub-matrix full rank condition of $\boldsymbol{A}$, we know that $\boldsymbol{u}_i^*$ is not in the null space of $\boldsymbol{x}^T\boldsymbol{z}\boldsymbol{S}^*$ for $O(n)$ indices. Therefore, we have that $\|\boldsymbol{x}^T\boldsymbol{z}\boldsymbol{S}^*\boldsymbol{u}_i^{*T}\| = O(\eta^2)$. Then, we have

$$\frac{r}{n}\sum_{i=1}^{n}\boldsymbol{u}_i^*\boldsymbol{S}^{*T}\boldsymbol{z}^T\boldsymbol{x}(\boldsymbol{S}^T\boldsymbol{S})^{-1}\boldsymbol{x}^T\boldsymbol{z}\boldsymbol{S}^*\boldsymbol{u}_i^{*T} \geq \frac{r}{\sigma_{max}(\boldsymbol{S})}C\eta^2,$$

for some constant C and where $\sigma_{max}(\boldsymbol{S})$ is the largest singular value of $\boldsymbol{S}$. By the nuclear-Frobenius norm inequality, we have

$$\|\boldsymbol{S}^T\boldsymbol{S}\|_* \leq \text{Rank}(\boldsymbol{S}^T\boldsymbol{S})\|\boldsymbol{S}\|^2 = k.$$

Therefore, the maximum singular value of $\boldsymbol{S}$ is at most k, resulting in the bound

$$\frac{r}{\sigma_{max}(\boldsymbol{S})}C\eta^2 \geq \frac{Cr\eta^2}{k}. \tag{A.11}$$

By a similar proof to what is done in Lemma 6, we can show that with probability greater than $1 - O(\frac{1}{n})$ we have

$$\left\|\frac{1}{nm}\sum_{i=1}^{n}\boldsymbol{a}_i\boldsymbol{P}_{\boldsymbol{S}}\boldsymbol{S}^*(\boldsymbol{S}^T\boldsymbol{S})^{-1}\boldsymbol{S}^{*T}\boldsymbol{P}_{\boldsymbol{S}}\boldsymbol{a}_i^T - \frac{l}{nm}\sum_{i=1}^{n}\overline{\boldsymbol{a}}_i\boldsymbol{P}_{\boldsymbol{S}}\boldsymbol{S}^*(\boldsymbol{S}^T\boldsymbol{S})^{-1}\boldsymbol{S}^{*T}\boldsymbol{P}_{\boldsymbol{S}}\boldsymbol{P}_{\boldsymbol{S}_i}\boldsymbol{a}_i^T\right\| \leq \frac{Cr\eta^2}{2k}. \tag{A.12}$$

Combining (A.12) with (A.11) proves the lemma. □

Finally, we combine the results of Lemmas 6, 8 and 9 to prove that for $\boldsymbol{M} = \boldsymbol{z}^T\boldsymbol{x}$, $\|\boldsymbol{M}\| = 1$ defined previously, we have

$$\begin{aligned}\boldsymbol{M} : \nabla^2 c(\boldsymbol{S}) : \boldsymbol{M} = \frac{2}{nm}\sum_{i=1}^{n}\Bigg(&\boldsymbol{a}_i\boldsymbol{S}_i(\boldsymbol{S}_i^T\boldsymbol{S}_i)^{-1}\boldsymbol{M}^T\boldsymbol{P}_{\boldsymbol{S}_i}\boldsymbol{M}(\boldsymbol{S}_i^T\boldsymbol{S}_i)^{-1}\boldsymbol{a}_i^T \\ &+ 2\boldsymbol{a}_i\boldsymbol{S}_i(\boldsymbol{S}_i^T\boldsymbol{S}_i)^{-1}\boldsymbol{M}^T\boldsymbol{S}_i(\boldsymbol{S}_i^T\boldsymbol{S}_i)^{-1}\boldsymbol{M}^T\boldsymbol{P}_{\boldsymbol{S}_i}\boldsymbol{a}_i^T \\ &- \overline{\boldsymbol{a}}_i\boldsymbol{M}(\boldsymbol{S}_i^T\boldsymbol{S}_i)^{-1}\boldsymbol{M}^T\boldsymbol{P}_{\boldsymbol{S}_i}\boldsymbol{a}_i^T\end{aligned}$$

$$
\begin{aligned}
&+ \boldsymbol{a}_i \boldsymbol{S}_i (\boldsymbol{S}_i^T \boldsymbol{S}_i)^{-1} \boldsymbol{S}_i^T \boldsymbol{M} (\boldsymbol{S}_i^T \boldsymbol{S}_i)^{-1} \boldsymbol{M}^T \boldsymbol{P}_{\boldsymbol{S}_i} \boldsymbol{a}_i^T \Big) \\
&\leq O(r\epsilon) + O(r\epsilon) - O\left(\frac{r\eta^2}{k}\right) + O(r\epsilon) \\
&= -O\left(\frac{r\eta^2}{k}\right),
\end{aligned}
$$

as long as $\epsilon \leq O(\frac{\eta^2}{k})$. Therefore, for $\|\nabla c(\boldsymbol{S})\| \leq \epsilon$ and $\boldsymbol{S}$ at least $\eta = O(\sqrt{\epsilon})$ away from a global minimum, we have $\lambda_{min}(\nabla^2 c(\boldsymbol{S})) \leq -O(\frac{r\epsilon}{k})$. Thus, the desired statement

"For all $\boldsymbol{S}$ such that $\|\nabla c(\boldsymbol{S})\| \leq \epsilon$ and $\boldsymbol{S}$ more than η-away from a *global* minimum, we have that $\lambda_{\min}(\nabla^2 f(\boldsymbol{x})) \leq -\zeta$ for some suitable ζ."

is true for $\eta = O(\sqrt{\epsilon})$ and $\zeta = O(\frac{r\epsilon}{k})$, with probability $1 - O(\frac{1}{n})$.

Thus, all local minima are global minima, and in particular the function $c(\boldsymbol{S})$ satisfies a $(\epsilon, O(\frac{r\epsilon}{k}), O(\sqrt{\epsilon}))$-strict saddle for sufficiently small ϵ.

A.2 Proof of Technical Lemmas

Lemma 10. *Let $X_1, \cdots, X_n$ be independent (but not necessarily identically distributed) random variables which satisfy the following:*

$$\mathbb{P}\left(|X_i - a_i| \geq \sigma_i \sqrt{\log(\frac{1}{\epsilon})}\right) \leq \epsilon.$$

Then we have

$$\mathbb{P}\left(\left|\frac{\sum_{i=1}^n X_i - a_i}{n}\right| \geq \frac{2\sqrt{\sum_{i=1}^n \sigma_i^2}}{n}\sqrt{\log(\frac{1}{4\epsilon})}\right) \leq \epsilon. \tag{A.13}$$

Proof. See [22]. □

Lemma 11. *Assume that $\boldsymbol{S}$ satisfies Assumption 2. Then we have*

$$\mathbb{P}\left(\left\|\left(\frac{\boldsymbol{S}^T\boldsymbol{S}}{m}\right)^{-1} - \left(\frac{\boldsymbol{S}_i^T\boldsymbol{S}_i}{l}\right)^{-1}\right\| \leq \sqrt{\frac{D\log(\frac{1}{\epsilon})}{l}}\right) \geq 1 - \epsilon. \tag{A.14}$$

Proof. This follows immediately from Lemma 13 and the following lemma from matrix perturbation theory (for proof, see e.g. [157]):

Lemma 12. *Let $\boldsymbol{A}, \boldsymbol{B}$ be invertible matrices and let $\boldsymbol{B} = \boldsymbol{A} + \boldsymbol{\Delta}$. Then, we have the following bound:*

$$\|\boldsymbol{A}^{-1} - \boldsymbol{B}^{-1}\| \leq \|\boldsymbol{A}^{-1}\|\|\boldsymbol{B}^{-1}\|\|\boldsymbol{\Delta}\|. \tag{A.15}$$

□

Lemma 13.

$$\mathbb{P}\left(\left\|\frac{\boldsymbol{S}^T\boldsymbol{S}}{m} - \frac{\boldsymbol{S}_i^T\boldsymbol{S}_i}{l}\right\| \leq \sqrt{\frac{C\log(\frac{1}{\epsilon})}{l}}\right) \geq 1 - \epsilon. \tag{A.16}$$

To prove this, we first introduce a matrix analog of the well-known Chernoff bound, the proof of which can be found in [164]:

Lemma 14. *Let $\mathcal{X} \in \mathbb{R}^{k\times k}$ be a finite set of positive-semidefinite matrices, and suppose that*

$$\max_{\boldsymbol{X}\in\mathcal{X}} \lambda_{\max}(\boldsymbol{X}) \leq D,$$

where $\lambda_{\min}/\lambda_{\max}$ is the minimum/maximum eigenvalue function. Sample $\{\boldsymbol{X}_1, \cdots, \boldsymbol{X}_\ell\}$ uniformly at random without replacement. Compute:

$$\mu_{\min} := \ell \cdot \lambda_{\min}(\mathbb{E}\boldsymbol{X}_1) \qquad \mu_{\max} := \ell \cdot \lambda_{\max}(\mathbb{E}\boldsymbol{X}_1).$$

Then:

$$\mathbb{P}\left\{\lambda_{\min}\left(\sum_j \boldsymbol{X}_j\right) \leq (1-\delta)\mu_{\min}\right\} \leq k \cdot \exp\left(\frac{-\delta^2\mu_{\min}}{4D}\right) \quad \textit{for } \delta \in [0,1),$$
$$\mathbb{P}\left\{\lambda_{\max}\left(\sum_j \boldsymbol{X}_j\right) \leq (1+\delta)\mu_{\max}\right\} \leq k \cdot \exp\left(\frac{-\delta^2\mu_{\max}}{4D}\right) \quad \textit{for } \delta \geq 0.$$

Now we proceed with the proof.

Proof. (Lemma 13) First, let us write L as the set of indices j such that $(\boldsymbol{W}_i)_{jj} = 1$. Then, we decompose $\boldsymbol{S} = \boldsymbol{Q}\boldsymbol{R}$ in a reduced $\boldsymbol{Q}\boldsymbol{R}$ factorization, where $\boldsymbol{Q}^T\boldsymbol{Q} = m\boldsymbol{I}$ (so that $\|\boldsymbol{R}\| = O(1)$). Then define $\boldsymbol{Q}_L$ as the $|L| \times k$ submatrix of $\boldsymbol{Q}$ formed with the rows in L. Then we can see that

$$\boldsymbol{S}^T\boldsymbol{S} = \boldsymbol{R}^T\boldsymbol{Q}^T\boldsymbol{Q}\boldsymbol{R}, \qquad \boldsymbol{S}_i^T\boldsymbol{S}_i = \boldsymbol{R}^T\boldsymbol{Q}_L^T\boldsymbol{Q}_L\boldsymbol{R}.$$

Now let us decompose the inner parts, which can be written using the rows of $\boldsymbol{Q}$, $\boldsymbol{q}_i \in \mathbb{R}^{1\times k}$:

$$\boldsymbol{Q}^T\boldsymbol{Q} = \sum_{i=1}^{m} \boldsymbol{q}_i^T\boldsymbol{q}_i,$$
$$\boldsymbol{Q}_L^T\boldsymbol{Q}_L = \sum_{i\in L} \boldsymbol{q}_i^T\boldsymbol{q}_i,$$

where $\boldsymbol{q}_i^T\boldsymbol{q}_i \in \mathbb{R}^{k\times k}$ rank-one positive semi-definite matrices. Therefore, we can take $\boldsymbol{Q}_L^T\boldsymbol{Q}_L$ as a random sample of size l from the set $\mathcal{X} = \{\boldsymbol{q}_i^T\boldsymbol{q}_i\}_{i=1,\cdots,m}$, which satisfies the conditions in Lemma 14 with $D = O(1)$. Furthermore, with $\mathcal{X}$, we observe that we have $\mathbb{E}\boldsymbol{X}_1 = \frac{\boldsymbol{Q}^T\boldsymbol{Q}}{m} = \boldsymbol{I}_k$, so we have

$$\lambda_{\min}(\mathbb{E}\boldsymbol{X}_1) = \lambda_{\max}(\mathbb{E}\boldsymbol{X}_1) = 1.$$

Therefore, we apply Lemma 14 to $\boldsymbol{Q}_L^T\boldsymbol{Q}_L$ and have that

$$\mathbb{P}\left\{\lambda_{\min}\left(\boldsymbol{Q}_L^T\boldsymbol{Q}_L\right) \leq (1-\delta)l\right\} \leq \exp\left(\frac{-\delta^2 l}{D'}\right),$$
$$\mathbb{P}\left\{\lambda_{\max}\left(\boldsymbol{Q}_L^T\boldsymbol{Q}_L\right) \geq (1+\delta)l\right\} \leq \exp\left(\frac{-\delta^2 l}{D'}\right),$$

where we set $D = \frac{D'}{4}$ with $D' = O(1)$. Some rearrangement gives:

$$\mathbb{P}\left\{\lambda_{\min}\left(\frac{\boldsymbol{Q}_L^T\boldsymbol{Q}_L}{l}\right) \geq 1 - \sqrt{\frac{D'\log\left(\frac{2}{\epsilon}\right)}{l}} \text{ and } \lambda_{\max}\left(\frac{\boldsymbol{Q}_L^T\boldsymbol{Q}_L}{l}\right) \leq 1 + \sqrt{\frac{D'\log\left(\frac{2}{\epsilon}\right)}{l}}\right\} \geq 1-\epsilon, \tag{A.17}$$

Now since $\frac{\boldsymbol{Q}^T\boldsymbol{Q}}{m} = \boldsymbol{I}_k$, we have

$$\lambda_{\min}\left(\frac{\boldsymbol{Q}^T\boldsymbol{Q}}{m}\right) = \lambda_{\max}\left(\frac{\boldsymbol{Q}^T\boldsymbol{Q}}{m}\right) = 1. \tag{A.18}$$

Combining equation (A.18) and (A.17) gives:

$$\mathbb{P}\left\{\left\|\frac{\boldsymbol{Q}_L^T\boldsymbol{Q}_L}{l} - \frac{\boldsymbol{Q}^T\boldsymbol{Q}}{m}\right\| \leq \sqrt{\frac{D'\log\left(\frac{2}{\epsilon}\right)}{l}}\right\} \geq 1-\epsilon. \tag{A.19}$$

Then, we have

$$\mathbb{P}\left\{\left\|\frac{\boldsymbol{R}^T\boldsymbol{Q}_L^T\boldsymbol{Q}_L\boldsymbol{R}}{l} - \frac{\boldsymbol{R}^T\boldsymbol{Q}^T\boldsymbol{Q}\boldsymbol{R}}{m}\right\| \leq \|\boldsymbol{R}\|^2\sqrt{\frac{D'\log\left(\frac{2}{\epsilon}\right)}{l}}\right\} \geq 1-\epsilon. \tag{A.20}$$

Taking $C = D'\|\boldsymbol{R}\|^4\log(2)$ gives the required result. □

A.3 List of Features for Netflix Data

- 24 Indicator Variables for Genres: Action, Adventure, Animation, Biography, Comedy, Crime, Documentary, Drama, Family, Fantasy, Film Noir, History, Horror, Music, Musical, Mystery, Romance, Sci-Fi, Short, Sport, Superhero, Thriller, War, Western
- 5 Indicator Variables for Release Date: Within last 10 years, Between 10-20 years,

Between 20-30 years, Between 30-40 years, Between 40-50 Years

- 6 Indicator Variables for Top Actors/Actresses defined by their Influence Score at time of release: Top 100 Actors, Top 100 Actresses, Top 250 Actors, Top 250 Actresses, Top 1000 Actors, Top 1000 Actresses
- IMDB Rating
- Number of Reviews
- Total Production Budget
- Total Runtime
- Total Box Office Revenue
- Indicator Variable for whether it is US produced
- 11 Indicator Variables for Month of Year Released (January removed to prevent multicollinearity)
- Number of Original Music Score
- Number of Male Actors
- Number of Female Factors
- 3 Indicator Variables for Film Language: English, French, Japanese
- Constant

Appendix B

Appendix to Chapter 3

B.1 Proof of Theorem 5

We first note that since S_k^p is a finite set, we only need to prove the result for a particular $\boldsymbol{s} \in S_k^p$, and it would apply for all $\boldsymbol{s}$. Therefore, we assume $\boldsymbol{s}$ is fixed below. For simplicity, we only demonstrate the proof for $\alpha_i(\boldsymbol{s}) = \overline{\boldsymbol{a}}_i \boldsymbol{\tau}_i(s)$, as the one for $(\boldsymbol{B}\boldsymbol{W}_i\boldsymbol{\tau}_i(s))^2$ follows in the same exact fashion. Furthermore, since we are only focusing on one particular $i \in \{1, \ldots, n\}$, we drop all i subscripts below for ease of notation. The quantities of interest are therefore:

$$\alpha(\boldsymbol{s}) = \frac{1}{m}\left(\overline{\boldsymbol{a}}\overline{\boldsymbol{a}}^T - \overline{\boldsymbol{a}}\boldsymbol{V}\left(\frac{\boldsymbol{I}_k}{\gamma} + \boldsymbol{V}^T\boldsymbol{W}\boldsymbol{V}\right)^{-1}\boldsymbol{V}^T\overline{\boldsymbol{a}}^T\right)$$
$$\alpha^F(\boldsymbol{s}) = \frac{1}{f}\left(\overline{\boldsymbol{a}}_F\overline{\boldsymbol{a}}_F^T - \overline{\boldsymbol{a}}_F\boldsymbol{V}_F\left(\frac{\boldsymbol{I}_k}{\gamma} + \boldsymbol{V}_F^T\boldsymbol{W}\boldsymbol{V}_F\right)^{-1}\boldsymbol{V}_F^T\overline{\boldsymbol{a}}_F^T\right).$$

First, let $\overline{\boldsymbol{V}} = \boldsymbol{W}\boldsymbol{V}$, and let us consider a reduced QR factorization of $\overline{\boldsymbol{V}} = \boldsymbol{Q}\boldsymbol{R}$ where $\boldsymbol{Q} \in \mathbb{R}^{m\times k}$ has orthogonal columns such that $\boldsymbol{Q}^T\boldsymbol{Q} = m \cdot \boldsymbol{I}_k$, and $\boldsymbol{R}_i \in \mathbb{R}^{k\times k}$. Note such definition implies $\|\boldsymbol{R}\| = O(1)$. Then, we rewrite the terms as follows:

$$\alpha(\boldsymbol{s}) = \frac{\overline{\boldsymbol{a}}\overline{\boldsymbol{a}}^T}{m} - \frac{\overline{\boldsymbol{a}}\boldsymbol{V}}{m}\left(\frac{\boldsymbol{I}_k}{m\gamma} + \frac{\boldsymbol{R}^T\boldsymbol{Q}^T\boldsymbol{Q}\boldsymbol{R}}{m}\right)^{-1}\frac{\boldsymbol{V}^T\overline{\boldsymbol{a}}^T}{m},$$
$$\alpha^F(\boldsymbol{s}) = \frac{\overline{\boldsymbol{a}}_F\overline{\boldsymbol{a}}_F^T}{f} - \frac{\overline{\boldsymbol{a}}_F\boldsymbol{V}_F}{f}\left(\frac{\boldsymbol{I}_k}{f\gamma} + \frac{\boldsymbol{R}^T\boldsymbol{Q}_F^T\boldsymbol{Q}_F\boldsymbol{R}}{f}\right)^{-1}\frac{\boldsymbol{V}_F^T\overline{\boldsymbol{a}}_F^T}{f}.$$

We note that

$$\overline{\boldsymbol{a}}\overline{\boldsymbol{a}}^T = \sum_{i=1}^{m} \overline{a}_i^2, \qquad \overline{\boldsymbol{a}}\boldsymbol{V} = \sum_{i=1}^{m} \overline{a}_i \boldsymbol{v}_i,$$
$$\overline{\boldsymbol{a}}_F\overline{\boldsymbol{a}}_F^T = \sum_{i\in F} \overline{a}_i^2, \qquad \overline{\boldsymbol{a}}_F\boldsymbol{V}_F = \sum_{i\in F} \overline{a}_i \boldsymbol{v}_i.$$

Therefore, if we treat $\overline{a}_1^2, \ldots \overline{a}_m^2$ as a finite population, then $\overline{\boldsymbol{a}}_F\overline{\boldsymbol{a}}_F^T$ is a random sample of f points drawn without replacement from that set, and similarly for $\overline{\boldsymbol{a}}_F\boldsymbol{V}_F$. Therefore, we can then utilize Hoeffding's inequality to bound the deviation of these terms, as reproduced below:

Proposition 3 (Hoeffding's Inequality)**.** *Let $\mathcal{X} = (x_1, \ldots, x_n)$ be a finite population of N points and $X_1, \ldots, X_n$ be a random sample drawn without replacement from $\boldsymbol{X}$. Let*

$$a = \min_{1\le i\le n} x_i \qquad \text{and} \qquad b = \max_{1\le i\le n} x_i.$$

Then, for all $\epsilon > 0$, we have

$$\mathbb{P}\left(\left|\frac{\sum_{i=1}^n X_i}{n} - \mu\right| \ge \epsilon\right) \le 2\exp\left(-\frac{2n\epsilon^2}{(b-a)^2}\right). \tag{B.1}$$

For a proof, see for example [11]. Then, applying Proposition 3 to $\overline{\boldsymbol{a}}_f\overline{\boldsymbol{a}}_f^T$, $\overline{\boldsymbol{a}}_f\boldsymbol{V}_f$, and inverting the inequality, we have

$$\mathbb{P}\left(\left|\frac{\overline{\boldsymbol{a}}_F\overline{\boldsymbol{a}}_F^T}{f} - \frac{\overline{\boldsymbol{a}}\overline{\boldsymbol{a}}^T}{m}\right| \le \sqrt{\frac{A\log(\frac{1}{\epsilon})}{f}}\right) \ge 1-\epsilon, \tag{B.2}$$

$$\mathbb{P}\left(\left\|\frac{\overline{\boldsymbol{a}}_F\boldsymbol{V}_F}{f} - \frac{\overline{\boldsymbol{a}}\boldsymbol{V}}{m}\right\| \le \sqrt{\frac{Bk\log(\frac{k}{\epsilon})}{f}}\right) \ge 1-\epsilon, \tag{B.3}$$

where A, B are constants independent of k, f, m, ϵ.

Now we show that $\frac{\boldsymbol{R}^T\boldsymbol{Q}^T\boldsymbol{Q}\boldsymbol{R}}{m}$ is close to $\frac{\boldsymbol{R}^T\boldsymbol{Q}_F^T\boldsymbol{Q}_F\boldsymbol{R}}{f}$:

Lemma 15.

$$\mathbb{P}\left(\left\|\frac{\boldsymbol{R}^T\boldsymbol{Q}^T\boldsymbol{Q}\boldsymbol{R}}{m}-\frac{\boldsymbol{R}^T\boldsymbol{Q}_F^T\boldsymbol{Q}_F\boldsymbol{R}}{f}\right\|\leq\sqrt{\frac{Ck\log(\frac{k}{\epsilon})}{f}}\right)\geq 1-\epsilon. \tag{B.4}$$

To prove this, we first introduce a matrix analog of the well-known Chernoff bound, the proof of which can be found in [164] (We note that [164] proves the following lemma for the sampling *with* replacement case, but the result can be transferred directly to the *without* replacement case using Lemma 1.1 in [11]):

Lemma 16. *Let $\mathcal{X}\in\mathbb{R}^{k\times k}$ be a finite set of positive-semidefinite matrices, and suppose that*

$$\max_{\boldsymbol{X}\in\mathcal{X}}\lambda_{\max}(\boldsymbol{X})\leq D,$$

where $\lambda_{\min}/\lambda_{\max}$is the minimum/maximum eigenvalue function. Sample $\{\boldsymbol{X}_1,\ldots,\boldsymbol{X}_\ell\}$ uniformly at random without replacement. Compute:

$$\mu_{\min}:=\ell\cdot\lambda_{\min}(\mathbb{E}\boldsymbol{X}_1)\qquad\mu_{\max}:=\ell\cdot\lambda_{\max}(\mathbb{E}\boldsymbol{X}_1).$$

Then,

$$\mathbb{P}\left\{\lambda_{\min}\left(\sum_j\boldsymbol{X}_j\right)\leq(1-\delta)\mu_{\min}\right\}\leq k\cdot\exp\left(\frac{-\delta^2\mu_{\min}}{4D}\right),\quad \textit{for } \delta\in[0,1),$$
$$\mathbb{P}\left\{\lambda_{\max}\left(\sum_j\boldsymbol{X}_j\right)\leq(1+\delta)\mu_{\max}\right\}\leq k\cdot\exp\left(\frac{-\delta^2\mu_{\max}}{4D}\right),\quad \textit{for } \delta\geq 0.$$

Using this lemma, we proceed with the proof of Lemma 15.

Proof. Proof of Lemma 15: First, we note that

$$\boldsymbol{Q}^T\boldsymbol{Q}=\sum_{i=1}^m\boldsymbol{q}_i^T\boldsymbol{q}_i,$$
$$\boldsymbol{Q}_F^T\boldsymbol{Q}_F=\sum_{i\in F}\boldsymbol{q}_i^T\boldsymbol{q}_i,$$

where $\boldsymbol{q}_i^T \boldsymbol{q}_i \in \mathbb{R}^{k\times k}$ rank-one positive semi-definite matrices. Therefore, we can take $\boldsymbol{Q}_F^T \boldsymbol{Q}_F$ as a random sample of size f from the set $\mathcal{X} = \{\boldsymbol{q}_i^T \boldsymbol{q}_i\}_{i=1,\ldots,m}$, which satisfies the conditions in Lemma 16 with $D = O(k)$. Furthermore, with $\mathcal{X}$, we observe that we have $\mathbb{E}\boldsymbol{X}_1 = \frac{\boldsymbol{Q}^T\boldsymbol{Q}}{m} = \boldsymbol{I}_k$, so we have

$$\lambda_{\min}(\mathbb{E}\boldsymbol{X}_1) = \lambda_{\max}(\mathbb{E}\boldsymbol{X}_1) = 1.$$

Therefore, we apply Lemma 16 to $\boldsymbol{Q}_F^T \boldsymbol{Q}_F$ and obtain

$$\mathbb{P}\left\{\lambda_{\min}\left(\boldsymbol{Q}_F^T\boldsymbol{Q}_F\right) \le (1-\delta)f\right\} \le k \cdot \exp\left(\frac{-\delta^2 f}{kD'}\right),$$
$$\mathbb{P}\left\{\lambda_{\max}\left(\boldsymbol{Q}_F^T\boldsymbol{Q}_F\right) \ge (1+\delta)f\right\} \le k \cdot \exp\left(\frac{-\delta^2 f}{kD'}\right),$$

where we set $D = \frac{kD'}{4}$ with $D' = O(1)$. Some rearrangement gives:

$$\mathbb{P}\left\{\lambda_{\min}\left(\frac{\boldsymbol{Q}_F^T\boldsymbol{Q}_F}{f}\right) \ge 1 - \sqrt{\frac{kD'\log\left(\frac{2k}{\epsilon}\right)}{f}} \text{ and } \lambda_{\max}\left(\frac{\boldsymbol{Q}_F^T\boldsymbol{Q}_F}{f}\right) \le 1 + \sqrt{\frac{kD'\log\left(\frac{2k}{\epsilon}\right)}{f}}\right\}$$
$$\ge 1 - \epsilon \tag{B.5}$$

Now since $\frac{\boldsymbol{Q}^T\boldsymbol{Q}}{m} = \boldsymbol{I}_k$, we have

$$\lambda_{\min}\left(\frac{\boldsymbol{Q}^T\boldsymbol{Q}}{m}\right) = \lambda_{\max}\left(\frac{\boldsymbol{Q}^T\boldsymbol{Q}}{m}\right) = 1 \tag{B.6}$$

Combining equation (B.6) and (B.5) gives

$$\mathbb{P}\left\{\left\|\frac{\boldsymbol{Q}_F^T\boldsymbol{Q}_F}{f} - \frac{\boldsymbol{Q}^T\boldsymbol{Q}}{m}\right\| \le \sqrt{\frac{kD'\log\left(\frac{2k}{\epsilon}\right)}{f}}\right\} \ge 1-\epsilon. \tag{B.7}$$

Then, we have

$$\mathbb{P}\left\{\left\|\frac{\boldsymbol{R}^T\boldsymbol{Q}_F^T\boldsymbol{Q}_F\boldsymbol{R}}{f} - \frac{\boldsymbol{R}^T\boldsymbol{Q}^T\boldsymbol{Q}\boldsymbol{R}}{m}\right\| \le \|\boldsymbol{R}\|^2\sqrt{\frac{kD'\log\left(\frac{2k}{\epsilon}\right)}{f}}\right\} \ge 1-\epsilon. \tag{B.8}$$

Taking $C = D'\|\boldsymbol{R}\|^4\log(2)$ gives the required result. (Note $\|\boldsymbol{R}\| = O(1)$ as we setup the QR

decomposition to have $\boldsymbol{Q}^T\boldsymbol{Q} = O(m)$). □ □

With equations (B.2), (B.3), and (B.4), we are now ready to bound $\alpha(\boldsymbol{s})$ and $\alpha^F(\boldsymbol{s})$. We first introduce another lemma from matrix perturbation theory (for proof, see e.g. [157]).

Lemma 17. *Let $\boldsymbol{A}, \boldsymbol{B}$ be invertible matrices and let $\boldsymbol{B} = \boldsymbol{A} + \boldsymbol{\Delta}$. Then, we have*

$$\|\boldsymbol{A}^{-1} - \boldsymbol{B}^{-1}\| \leq \|\boldsymbol{A}^{-1}\|\|\boldsymbol{B}^{-1}\|\|\boldsymbol{\Delta}\|. \tag{B.9}$$

Then, we have

$$\begin{aligned}\|\alpha(\boldsymbol{s}) - \alpha^F(\boldsymbol{s})\| \leq & \left|\frac{\overline{\boldsymbol{a}}\overline{\boldsymbol{a}}^T}{m} - \frac{\overline{\boldsymbol{a}}_F\overline{\boldsymbol{a}}_F^T}{f}\right| \\ & + \left|\frac{\overline{\boldsymbol{a}}\boldsymbol{V}}{m}\left(\frac{\boldsymbol{I}_k}{m\gamma} + \frac{\boldsymbol{R}^T\boldsymbol{Q}^T\boldsymbol{Q}\boldsymbol{R}}{m}\right)^{-1}\frac{\boldsymbol{V}^T\overline{\boldsymbol{a}}^T}{m} - \right. \\ & \left.\frac{\overline{\boldsymbol{a}}_F\boldsymbol{V}_F}{f}\left(\frac{\boldsymbol{I}_k}{f\gamma} + \frac{\boldsymbol{R}^T\boldsymbol{Q}_F^T\boldsymbol{Q}_F\boldsymbol{R}}{f}\right)^{-1}\frac{\boldsymbol{V}_F^T\overline{\boldsymbol{a}}_F^T}{f}\right|.\end{aligned}$$

Using (B.2) and triangle inequality, we have

$$\begin{aligned}& \leq \sqrt{\frac{A\log(\frac{1}{\epsilon})}{f}} + \left|\left(\frac{\overline{\boldsymbol{a}}\boldsymbol{V}}{m} - \frac{\overline{\boldsymbol{a}}_F\boldsymbol{V}_F}{f}\right)\left(\frac{\boldsymbol{I}_k}{m\gamma} + \frac{\boldsymbol{R}^T\boldsymbol{Q}^T\boldsymbol{Q}\boldsymbol{R}}{m}\right)^{-1}\frac{\boldsymbol{V}^T\overline{\boldsymbol{a}}^T}{m}\right| \\ & + \left|\frac{\overline{\boldsymbol{a}}_F\boldsymbol{V}_F}{f}\left(\left(\frac{\boldsymbol{I}_k}{m\gamma} + \frac{\boldsymbol{R}^T\boldsymbol{Q}^T\boldsymbol{Q}\boldsymbol{R}}{m}\right)^{-1} - \left(\frac{\boldsymbol{I}_k}{f\gamma} + \frac{\boldsymbol{R}^T\boldsymbol{Q}_F^T\boldsymbol{Q}_F\boldsymbol{R}}{f}\right)^{-1}\right)\frac{\boldsymbol{V}^T\overline{\boldsymbol{a}}^T}{m}\right| \\ & + \left|\frac{\overline{\boldsymbol{a}}_F\boldsymbol{V}_F}{f}\left(\frac{\boldsymbol{I}_k}{f\gamma} + \frac{\boldsymbol{R}^T\boldsymbol{Q}_F^T\boldsymbol{Q}_F\boldsymbol{R}}{f}\right)^{-1}\left(\frac{\boldsymbol{V}^T\overline{\boldsymbol{a}}^T}{m} - \frac{\boldsymbol{V}_F^T\overline{\boldsymbol{a}}_F^T}{f}\right)\right|.\end{aligned}$$

Using (B.3) and Lemma 17, we have

$$\begin{aligned}& \leq \sqrt{\frac{A\log(\frac{1}{\epsilon})}{f}} + \sqrt{\frac{B'k\log(\frac{k}{\epsilon})}{f}} \\ & + D'\left|\frac{\overline{\boldsymbol{a}}_F\boldsymbol{V}_F}{f}\left(\frac{\boldsymbol{I}_k}{m\gamma} - \frac{\boldsymbol{I}_k}{f\gamma} + \frac{\boldsymbol{R}^T\boldsymbol{Q}^T\boldsymbol{Q}\boldsymbol{R}}{m} - \frac{\boldsymbol{R}^T\boldsymbol{Q}_F^T\boldsymbol{Q}_F\boldsymbol{R}}{f}\right)\frac{\boldsymbol{V}^T\overline{\boldsymbol{a}}^T}{m}\right| \\ & + \sqrt{\frac{B''k\log(\frac{k}{\epsilon})}{f}},\end{aligned}$$

for some constant D'. Note that $\|\frac{\boldsymbol{I}_k}{m\gamma} - \frac{\boldsymbol{I}_k}{f\gamma}\| \leq \frac{M}{f}$, so using (B.4), we obtain

$$\mathbb{P}\left(\left\|\frac{\boldsymbol{I}_k}{m\gamma} - \frac{\boldsymbol{I}_k}{f\gamma} + \frac{\boldsymbol{R}^T\boldsymbol{Q}^T\boldsymbol{Q}\boldsymbol{R}}{m} - \frac{\boldsymbol{R}^T\boldsymbol{Q}_F^T\boldsymbol{Q}_F\boldsymbol{R}}{f}\right\| \leq \sqrt{\frac{C'k\log(\frac{k}{\epsilon})}{f}}\right) \geq 1-\epsilon, \tag{B.10}$$

for some new constant C'. Then, using (B.10), we have

$$\leq \sqrt{\frac{A\log(\frac{1}{\epsilon})}{f}} + \sqrt{\frac{B'k\log(\frac{k}{\epsilon})}{f}} + \sqrt{\frac{C''k\log(\frac{k}{\epsilon})}{f}} + \sqrt{\frac{B''k\log(\frac{k}{\epsilon})}{f}}.$$

Since $k \geq 1$, we have

$$\leq \sqrt{\frac{Mk\log(\frac{k}{\epsilon})}{f}},$$

For some constant M with probability at least $1-4\epsilon$. Therefore, taking $\epsilon' = \frac{\epsilon}{4}$ and absorbing the extra constant into M gives the requires result.

B.2 Proof of Theorem 6

We first prove a Hoeffding-type bound as follows

Proposition 4. *Let $X_1, \ldots, X_n$ be independent (but not necessarily identically distributed) random variables which satisfy*

$$\mathbb{P}\left(|X_i - a_i| \geq t\right) \leq \exp\left(\frac{-t^2}{\sigma_i^2}\right).$$

Then, we have

$$\mathbb{P}\left(\left|\frac{\sum_{i=1}^n X_i - a_i}{n}\right| \geq t\right) \leq 4\exp\left(-\frac{n^2t^2}{4\sum_{i=1}^n \sigma_i^2}\right). \tag{B.11}$$

We note that such statement differs from Hoeffding's inequality as we do not require $\mathbb{E}[X_i] = a_i$. The proof is as follows.

Proof. Proof of Proposition 4: We first introduce a lemma known as Chernoff's bounding method (proven in [39]):

Lemma 18. *Let Z be a random variable on $\mathbb{R}$. Then for all $t > 0$, we have*

$$\mathbb{P}(Z \geq t) \leq \inf_{s>0}[e^{-st}M_Z(s)],$$

where $M_Z(s)$ is the moment generating function of Z.

Let $Y_i = X_i - a_i$. Then, we have

$$\mathbb{P}\left(\sum_{i=1}^n Y_i \geq t\right) \leq \min_{s>0} e^{-st}\prod_{i=1}^n \mathbb{E}\left[e^{sY_i}\right]. \tag{B.12}$$

We aim to bound $\mathbb{E}\left[e^{sY_i}\right]$, subject to $\mathbb{P}(|Y_i| \geq t) \leq \exp\left(\frac{-t^2}{\sigma_i^2}\right)$. Since e^{st} is an increasing function of t, $\mathbb{E}\left[e^{sY_i}\right]$ is maximized when $\mathbb{P}(Y_i \geq t) = \exp\left(\frac{-t^2}{\sigma_i^2}\right)$, which results in a probability distribution function of $f_{Y_i}(y) = 2\frac{t}{\sigma_i^2}\exp\left(\frac{-t^2}{\sigma_i^2}\right)$ for Y_i. Then, we have

$$\mathbb{E}\left[e^{sY_i}\right] \leq \int_0^\infty 2\frac{t}{\sigma_i^2}\exp\left(st - \frac{t^2}{\sigma_i^2}\right)\mathrm{d}t$$

$$
\begin{aligned}
&= \int_0^\infty 2\frac{t}{\sigma_i^2} \exp\left(-\left(\frac{t}{\sigma_i} - \frac{s\sigma_i}{2}\right)^2 + \frac{s^2\sigma_i^2}{4}\right) \mathrm{d}t \\
&= 2\exp\left(\frac{s^2\sigma_i^2}{4}\right) \int_0^\infty t \exp\left(-\left(t - \frac{s\sigma_i}{2}\right)^2\right) \mathrm{d}t \\
&= \exp\left(\frac{s^2\sigma_i^2}{4}\right) \left(\exp\left(-\frac{s^2\sigma_i^2}{4}\right) + \sqrt{\pi}\frac{s\sigma_i}{2}\left(\mathrm{erf}\left(\frac{s\sigma_i}{2}\right) + 1\right)\right),
\end{aligned}
$$

where erf is the standard error function. As the error function is upper bounded by 1, the last expression is less than or equal to:

$$
\begin{aligned}
&\leq 1 + \sqrt{\pi} s\sigma_i \exp\left(\frac{s^2\sigma_i^2}{4}\right) \\
&\leq 2\exp\left(s^2\sigma_i^2\right).
\end{aligned}
$$

We then substitute this result into (B.12) and obtain

$$
\mathbb{P}\left(\sum_{i=1}^n Y_i \geq t\right) \leq \min_{s>0} 2\exp\left(-st + s^2\sum_{i=1}^n \sigma_i^2\right)
$$

Note that this is minimized at $s = \frac{t}{2\sum_{i=1}^n \sigma_i^2}$, so we have

$$
\leq 2\exp\left(-\frac{t^2}{4\sum_{i=1}^n \sigma_i^2}\right).
$$

Therefore,

$$
\mathbb{P}\left(\frac{\sum_{i=1}^n Y_i}{n} \geq t\right) \leq 2\exp\left(-\frac{n^2t^2}{4\sum_{i=1}^n \sigma_i^2}\right).
$$

By applying the previous derivation to $-Y_1, \ldots, -Y_n$, we obtain

$$
\mathbb{P}\left(\frac{\sum_{i=1}^n Y_i}{n} \leq -t\right) \leq 2\exp\left(-\frac{n^2t^2}{4\sum_{i=1}^n \sigma_i^2}\right).
$$

Combining the two results completes the proof. □

Using Proposition 4, we bound the difference between $\tilde{c}_G^F(\boldsymbol{s}_t)$ and $c(\boldsymbol{s}_t)$. We have

$$
|\tilde{c}_G^F(\boldsymbol{s}_t) - c(\boldsymbol{s}_t)| = \left|\frac{1}{n}\sum_{i=1}^n \alpha_i(\boldsymbol{s}) - \frac{1}{g}\sum_{i\in G} \alpha_i^F(\boldsymbol{s})\right|
$$

$$\leq \left| \frac{1}{n} \sum_{i=1}^{n} \alpha_i(\boldsymbol{s}) - \frac{1}{g} \sum_{i \in G} \alpha_i(\boldsymbol{s}) \right| + \left| \frac{1}{g} \sum_{i \in G} \alpha_i(\boldsymbol{s}) - \frac{1}{g} \sum_{i \in G} \alpha_i^F(\boldsymbol{s}) \right|.$$

The first term can be seen as the tail bound for a random sample of size g chosen without replacement from the finite set $\{\alpha_i(\boldsymbol{s})\}_{i \in [n]}$. Thus, we can apply Hoeffding's theorem in Proposition 3, and obtain that with probability at least $1 - \epsilon$

$$\left| \frac{1}{n} \sum_{i=1}^{n} \alpha_i(\boldsymbol{s}) - \frac{1}{g} \sum_{i \in G} \alpha_i(\boldsymbol{s}) \right| \leq \sqrt{\frac{M \log\left(\frac{1}{\epsilon}\right)}{g}}. \tag{B.13}$$

Now for *any* fixed set G, for $i, j \in G$ with $i \neq j$, we have that $\alpha_i^F(\boldsymbol{s})$ and $\alpha_j^F(\boldsymbol{s})$ are independent (as we construct F separately and independently for every i). Therefore, Theorem 5 can be inverted to read that, for any i, we have:

$$\mathbb{P}\left(|\alpha_i(\boldsymbol{s}) - \alpha_i^F(\boldsymbol{s})| \geq t\right) \leq \exp\left(-\frac{ft^2}{Mk\log(k)}\right).$$

Therefore $\alpha_i^F(\boldsymbol{s})$ satisfies the conditions of Proposition 4 with $X_i = \alpha_i^F(\boldsymbol{s})$ and parameters $\sigma_i^2 = \frac{Mk\log(k)}{f}$, $a_i = \alpha_i(\boldsymbol{s})$. Then, applying Proposition 4 to the second term, we have for *any* fixed G:

$$\mathbb{P}\left(\left| \frac{1}{g} \sum_{i \in G} \alpha_i(\boldsymbol{s}) - \frac{1}{g} \sum_{i \in G} \alpha_i^F(\boldsymbol{s}) \right| \leq \epsilon \,\middle|\, G \right) \leq \sqrt{\frac{M' k \log(k/\epsilon)}{fg}} \tag{B.14}$$

Now using Frechet inequalities and Equations (B.13) and (B.14), we have:

$$\begin{aligned}
&\mathbb{P}\left(|\tilde{c}_G^F(\boldsymbol{s}_t) - c(\boldsymbol{s}_t)| \leq \epsilon\right) \\
=&\mathbb{P}\left(\left| \frac{1}{n} \sum_{i=1}^{n} \alpha_i(\boldsymbol{s}) - \frac{1}{g} \sum_{i \in G} \alpha_i(\boldsymbol{s}) \right| + \left| \frac{1}{g} \sum_{i \in G} \alpha_i(\boldsymbol{s}) - \frac{1}{g} \sum_{i \in G} \alpha_i^F(\boldsymbol{s}) \right| \leq \epsilon \right) \\
\geq&\mathbb{P}\left(\left| \frac{1}{n} \sum_{i=1}^{n} \alpha_i(\boldsymbol{s}) - \frac{1}{g} \sum_{i \in G} \alpha_i(\boldsymbol{s}) \right| \leq \frac{\epsilon}{2} \right) \\
+&\mathbb{P}\left(\left| \frac{1}{g} \sum_{i \in G} \alpha_i(\boldsymbol{s}) - \frac{1}{g} \sum_{i \in G} \alpha_i^F(\boldsymbol{s}) \right| \leq \frac{\epsilon}{2} \,\middle|\, G : \left| \frac{1}{n} \sum_{i=1}^{n} \alpha_i(\boldsymbol{s}) - \frac{1}{g} \sum_{i \in G} \alpha_i(\boldsymbol{s}) \right| \leq \frac{\epsilon}{2} \right) \\
\leq&\sqrt{\frac{M \log\left(\frac{1}{\epsilon}\right)}{g}} + \sqrt{\frac{M' k \log\left(\frac{k}{\epsilon}\right)}{fg}},
\end{aligned}$$

with probability $1 - 2\epsilon$. As $k, f \geq 1$, a loose bound is therefore

$$\leq \sqrt{\frac{M'' k \log\left(\frac{k}{\epsilon}\right)}{g}},$$

with probability $1 - 2\epsilon$. Therefore, taking $\epsilon' = 2\epsilon$ and $A = M'' \log(2)$ we have

$$\mathbb{P}\left(|\tilde{c}_G^F(\boldsymbol{s}_t) - c(\boldsymbol{s}_t)| \leq \sqrt{\frac{Ak \log\left(\frac{k}{\epsilon'}\right)}{g}}\right) \geq 1 - \epsilon'. \tag{B.15}$$

Through a similar derivation, we have

$$\mathbb{P}\left(\|\nabla \tilde{c}_G^F(\boldsymbol{s}_t) - \nabla c(\boldsymbol{s}_t)\|_2 \leq \sqrt{\frac{B(p+k) \log\left(\frac{k}{\epsilon'}\right)}{g}}\right) \geq 1 - \epsilon'. \tag{B.16}$$

B.3 Proof of Theorem 7

In this section, we provide the proof of Theorem 7.

B.3.1 Proof of Finiteness

We first prove that the outer approximation algorithm indeed converges in finite number of steps C with probability 1. To do so, we note that the set of feasible solutions S_p^k is finite, and we prove that the series of feasible integer solutions $\boldsymbol{s}_1, \boldsymbol{s}_2, \cdots$ created during Algorithm 1 never repeats indefinitely, with probability 1.

First, we show that if we have $\boldsymbol{s}_a = \boldsymbol{s}_b$ for some $a+1 \leq b$ where we have not terminated at iteration b, then we must have $\tilde{c}_G^F(\boldsymbol{s}_b) > \tilde{c}_G^F(\boldsymbol{s}_a)$. At iteration a, the following cutting plane was added:

$$\eta \geq \tilde{c}_G^F(\boldsymbol{s}_a) + \nabla \tilde{c}_G^F(\boldsymbol{s}_a)^T(\boldsymbol{s} - \boldsymbol{s}_a). \tag{B.17}$$

Therefore, using (B.17) at iteration $b-1 \geq a$ (note $\boldsymbol{s}_b, \eta_b$ is the optimal solution in the $b-1$th iteration, so it must satisfy the inequality in (B.17)) gives:

$$\eta_b \geq \tilde{c}_G^F(\boldsymbol{s}_a) + \nabla \tilde{c}_G^F(\boldsymbol{s}_b)^T(\boldsymbol{s}_b - \boldsymbol{s}_a) \geq \tilde{c}_G^F(\boldsymbol{s}_a). \tag{B.18}$$

Now consider the cutting plane problem at iteration b, where the following cutting plane was added:

$$\eta \geq \tilde{c}_G^F(\boldsymbol{s}_b) + \nabla \tilde{c}_G^F(\boldsymbol{s}_b)^T(\boldsymbol{s} - \boldsymbol{s}_b). \tag{B.19}$$

Assume that we have $\tilde{c}_G^F(\boldsymbol{s}_b) \leq \tilde{c}_G^F(\boldsymbol{s}_a)$. Then $\eta = \eta_b$, $\boldsymbol{s} = \boldsymbol{s}_b$ is a feasible solution for the cutting plane problem at iteration b since it satisfies the newly added constraint: $\eta_b \geq \tilde{c}_G^F(\boldsymbol{s}_a) \geq \tilde{c}_G^F(\boldsymbol{s}_b)) + \nabla \tilde{c}_G^F(\boldsymbol{s}_b)^T(\boldsymbol{s}_b - \boldsymbol{s}_b) = \tilde{c}_G^F(\boldsymbol{s}_b))$. By construction of the algorithm, we therefore satisfy the termination condition, so the algorithm terminates, a contradiction. Therefore, we must have $\tilde{c}_G^F(\boldsymbol{s}_b) > \tilde{c}_G^F(\boldsymbol{s}_a)$.

Then consider Algorithm 1 after $Q = \binom{p}{k}(q-1)+1$ iterations where $q > 0$ is a positive integer, with the corresponding series of feasible integer solutions being $\boldsymbol{s}_1, \cdots, \boldsymbol{s}_Q$. Then since there are only $\binom{p}{k}$ feasible solutions, by the pigeonhole principle, there must be $\boldsymbol{s} \in S_k^p$

that has appeared at least q times in the series, and denote these as $\boldsymbol{s}_{l_1}, \cdots, \boldsymbol{s}_{l_q}$. By the observation above, we know that if Algorithm 1 has not terminated, we must have:

$$\tilde{c}_G^F(\boldsymbol{s}_{l_1}) < \tilde{c}_G^F(\boldsymbol{s}_{l_2}) < \cdots < \tilde{c}_G^F(\boldsymbol{s}_{l_q})$$

Note that since $\boldsymbol{s}_{l_1} = \boldsymbol{s}_{l_2} = \cdots = \boldsymbol{s}_{l_q}$, we have that $\tilde{c}_G^F(\boldsymbol{s}_{l_1}), \cdots, \tilde{c}_G^F(\boldsymbol{s}_{l_q})$ are q independent realizations of an identically distributed estimate of $c(\boldsymbol{s}_{l_1})$, and thus by standard probability results (see e.g. [68]) we have that:

$$\mathbb{P}(\tilde{c}_G^F(\boldsymbol{s}_{l_1}) < \tilde{c}_G^F(\boldsymbol{s}_{l_2}) < \cdots < \tilde{c}_G^F(\boldsymbol{s}_{l_q})) \leq \frac{1}{q!}$$

The result follows by noting that $\tilde{c}_G^F(\boldsymbol{s}_{l_1}), \cdots, \tilde{c}_G^F(\boldsymbol{s}_{l_q})$ are exchangable random variables so any ordering is equally likely, and at most 1 out of all $q!$ permutations of $\tilde{c}_G^F(\boldsymbol{s}_{l_1}), \cdots, \tilde{c}_G^F(\boldsymbol{s}_{l_q})$ satisfy the monotone relationship. Then we have that:

$$\begin{aligned}
&\mathbb{P}(\text{Algorithm has not terminated after } Q \text{ iterations} \mid \boldsymbol{s}_1, \cdots \boldsymbol{s}_Q) \\
&\leq \mathbb{P}(\tilde{c}_G^F(\boldsymbol{s}_{l_1}) < \tilde{c}_G^F(\boldsymbol{s}_{l_2}) < \cdots < \tilde{c}_G^F(\boldsymbol{s}_{l_q}) \mid \boldsymbol{s}_1, \cdots \boldsymbol{s}_Q) \\
&\leq \frac{1}{q!}
\end{aligned}$$

Therefore, by taking expectations on both sides, we have:

$$\mathbb{P}(\text{Algorithm has not terminated after } Q \text{ iterations}) \leq \frac{1}{q!},$$

which converges to 0 as $Q \to \infty$.

We have thus proved that the algorithm terminates in finite number of iterations with probability 1.

B.3.2 Proof of Optimality Convergence

We next proceed to prove our main convergence result. Consider the cutting plane problem at termination with C cutting planes:

$$\min_{\boldsymbol{s}\in S_p^k, \eta>0} \quad \eta \tag{B.20}$$

$$\text{s.t.} \quad \eta \geq \tilde{c}_G^F(\boldsymbol{s}_i) + \nabla \tilde{c}_G^F(\boldsymbol{s}_i)^T(\boldsymbol{s} - \boldsymbol{s}_i), \quad \forall i \in [C]. \tag{B.21}$$

Denote $\boldsymbol{s}^*$ as the true optimal solution to the problem in (2.1). Let $\tilde{\boldsymbol{s}}^*$ the optimal solution to the problem in (B.20)-(B.21). For notational simplicity, let us denote $\eta^*(\boldsymbol{s})$ as the optimal value for η under $\boldsymbol{s}$ in the problem (B.20)-(B.21). In particular, we have that:

$$\eta^*(\boldsymbol{s}) = \max_{i\in[C]} \tilde{c}_G^F(\boldsymbol{s}_i) + \nabla \tilde{c}_G^F(\boldsymbol{s}_i)^T(\boldsymbol{s} - \boldsymbol{s}_i). \tag{B.22}$$

We introduce a few useful lemmas:

Lemma 19. *For any $\boldsymbol{s} \in S_k^p$ with $c(\boldsymbol{s}) \neq c(\boldsymbol{s}^*)$, we have, for some $b > 0$:*

$$c(\boldsymbol{s}) \geq c(\boldsymbol{s}^*) + b.$$

Proof. Proof: Note that any $\boldsymbol{s}$ with $c(\boldsymbol{s}) \neq c(\boldsymbol{s}^*)$ is non-optimal. Define the set of non-optimal solutions $S_{no} = \{\boldsymbol{s} \in S_p^k \mid c(\boldsymbol{s}) > c(\boldsymbol{s}^*)\} \subset S_p^k$. Then for any $s \in S_{no}$, the conclusion immediately follows as S_p^k is a finite set:

$$c(\boldsymbol{s}) - c(\boldsymbol{s}^*) \geq \min_{\boldsymbol{s}\in S_{no}} c(\boldsymbol{s}) - c(\boldsymbol{s}^*) = b > 0$$

□ □

Lemma 20.

$$\tilde{c}_G^F(\tilde{\boldsymbol{s}}^*) \leq \eta^*(\boldsymbol{s}^*).$$

Proof. Proof: By definition of the cutting-plane algorithm (Algorithm 1), there must exist

a cutting plane at the optimal point $\tilde{\boldsymbol{s}}^*$ of the form:

$$\eta \geq \tilde{c}_G^F(\tilde{\boldsymbol{s}}^*) + \nabla \tilde{c}_G^F(\boldsymbol{s}_i)^T(\boldsymbol{s} - \tilde{\boldsymbol{s}}^*), \tag{B.23}$$

since we only terminate when $\boldsymbol{s}_t = \boldsymbol{s}_{t-1}$. Then, by feasibility of $\eta^*(\tilde{\boldsymbol{s}}^*)$, using (B.23) we have:

$$\eta^*(\tilde{\boldsymbol{s}}^*) \geq \tilde{c}_G^F(\tilde{\boldsymbol{s}}^*) + \nabla \tilde{c}_G^F(\boldsymbol{s}_i)^T(\tilde{\boldsymbol{s}}^* - \tilde{\boldsymbol{s}}^*) = \tilde{c}_G^F(\tilde{\boldsymbol{s}}^*). \tag{B.24}$$

By the optimality of $\tilde{\boldsymbol{s}}^*$, we also have:

$$\eta^*(\boldsymbol{s}^*) \geq \eta^*(\tilde{\boldsymbol{s}}^*). \tag{B.25}$$

Combining (B.25) and (B.24) gives us the assertion required. □ □

Lemma 21.

$$\mathbb{P}\left(\eta^*(\boldsymbol{s}^*) - c(\boldsymbol{s}^*) < \epsilon\right) \geq 1 - Ck\left(\exp\left(\frac{-g\epsilon^2}{Ak}\right) + \exp\left(\frac{-ga^4}{A'(p+k)}\right)\right),$$

where A, A' are constants.

Proof. Proof: By definition of η^*, we have:

$$\begin{aligned}
\mathbb{P}\left(\eta^*(\boldsymbol{s}^*) - c(\boldsymbol{s}^*) < \epsilon\right) &= \mathbb{P}\left(\max_{i\in[C]} \tilde{c}_G^F(\boldsymbol{s}_i) + \nabla \tilde{c}_G^F(\boldsymbol{s}_i)^T(\boldsymbol{s}^* - \boldsymbol{s}_i) - c(\boldsymbol{s}^*) < \epsilon\right) \\
&= \mathbb{P}\left(\tilde{c}_G^F(\boldsymbol{s}_i) + \nabla \tilde{c}_G^F(\boldsymbol{s}_i)^T(\boldsymbol{s}^* - \boldsymbol{s}_i) - c(\boldsymbol{s}^*) < \epsilon \ \ \forall i\right) \\
&\geq \sum_{i=1}^{C} \mathbb{P}\left(\tilde{c}_G^F(\boldsymbol{s}_i) + \nabla \tilde{c}_G^F(\boldsymbol{s}_i)^T(\boldsymbol{s}^* - \boldsymbol{s}_i) - c(\boldsymbol{s}^*) < \epsilon\right) \qquad \text{(B.26)} \\
&\quad -(C-1), \qquad \text{(B.27)}
\end{aligned}$$

where the last inequality follows by Frechet's inequalities:

$$\mathbb{P}(A_1 \cap A_2 \cap \ldots \cap A_n) \geq \mathbb{P}(A_1) + \mathbb{P}(A_2) + \ldots + \mathbb{P}(A_n) - (n-1).$$

We focus on bounding each term $\mathbb{P}\left(\tilde{c}_G^F(\boldsymbol{s}_i) + \nabla \tilde{c}_G^F(\boldsymbol{s}_i)^T(\boldsymbol{s}^* - \boldsymbol{s}_i) - c(\boldsymbol{s}^*) < \epsilon\right)$ in the sum.

Note that by the definition of the convexity parameter (3.16), we have that:

$$c(\boldsymbol{s}^*) \geq c(\boldsymbol{s}_i) + \nabla c(\boldsymbol{s}_i)^T(\boldsymbol{s}^* - \boldsymbol{s}_i) + \frac{\|\boldsymbol{s}^* - \boldsymbol{s}_i\|^2 a^2}{2}, \quad \forall i. \tag{B.28}$$

Using (B.28), we have:

$$\begin{aligned}
&\mathbb{P}\left(\tilde{c}_G^F(\boldsymbol{s}_i) + \nabla \tilde{c}_G^F(\boldsymbol{s}_i)^T(\boldsymbol{s}^* - \boldsymbol{s}_i) - c(\boldsymbol{s}^*) < \epsilon\right) \\
&\geq \mathbb{P}\left((\tilde{c}_G^F(\boldsymbol{s}_i) - c(\boldsymbol{s}_i)) + \left(\nabla \tilde{c}_G^F(\boldsymbol{s}_i) - \nabla c(\boldsymbol{s}_i)\right)^T(\boldsymbol{s}^* - \boldsymbol{s}_i) < \epsilon + \frac{\|\boldsymbol{s}^* - \boldsymbol{s}_i\|^2 a^2}{2}\right).
\end{aligned} \tag{B.29}$$

If $\boldsymbol{s}_i = \boldsymbol{s}^*$, then (B.29) simplifies to $\mathbb{P}\left((\tilde{c}_G^F(\boldsymbol{s}_i) - c(\boldsymbol{s}_i)) < \epsilon\right)$. If $\boldsymbol{s}_i \neq \boldsymbol{s}^*$, then by Frechet's inequalities, we have the bound:

$$(B.29) \geq \mathbb{P}\left((\tilde{c}_G^F(\boldsymbol{s}_i) - c(\boldsymbol{s}_i)) < \epsilon\right) + \mathbb{P}\left(\left(\nabla \tilde{c}_G^F(\boldsymbol{s}_i) - \nabla c(\boldsymbol{s}_i)\right)^T(\boldsymbol{s}^* - \boldsymbol{s}_i) < \frac{\|\boldsymbol{s}^* - \boldsymbol{s}_i\|^2 a^2}{2}\right) - 1,$$

which is always less than or equal to $\mathbb{P}\left((\tilde{c}_G^F(\boldsymbol{s}_i) - c(\boldsymbol{s}_i)) < \epsilon\right)$. Therefore, we can assume without loss of generality $\boldsymbol{s}^* \neq \boldsymbol{s}_i$, and utilize the bound:

$$\begin{aligned}
(B.29) \quad &\geq \quad \mathbb{P}\left((\tilde{c}_G^F(\boldsymbol{s}_i) - c(\boldsymbol{s}_i)) < \epsilon\right) + \mathbb{P}\left(\|\nabla \tilde{c}_G^F(\boldsymbol{s}_i) - \nabla c(\boldsymbol{s}_i)\| < \frac{\|\boldsymbol{s}^* - \boldsymbol{s}_i\| a^2}{2}\right) - 1. \\
&\geq \quad \mathbb{P}\left((\tilde{c}_G^F(\boldsymbol{s}_i) - c(\boldsymbol{s}_i)) < \epsilon\right) + \mathbb{P}\left(\|\nabla \tilde{c}_G^F(\boldsymbol{s}_i) - \nabla c(\boldsymbol{s}_i)\| < \frac{a^2}{2}\right) - 1,
\end{aligned} \tag{B.30}$$

where the last inequality follows from $\|\boldsymbol{s}^* - \boldsymbol{s}_i\| \geq 1$ ($\boldsymbol{s}^* \neq \boldsymbol{s}_i$). Substituting the results from Theorem 6 gives:

$$\begin{aligned}
(\text{B.30}) \quad &\geq \quad 1 - k\exp\left(\frac{-g\epsilon^2}{Ak}\right) + 1 - k\exp\left(\frac{-ga^4}{4B(p+k)}\right) - 1 \\
&\geq \quad 1 - k\left(\exp\left(\frac{-g\epsilon^2}{Ak}\right) + \exp\left(\frac{-ga^4}{A'(p+k)}\right)\right).
\end{aligned} \tag{B.31}$$

Therefore, using (B.31), we have that, for (B.27):

$$\begin{aligned}
&(B.27) \\
&\geq C - Ck\left(\exp\left(\frac{-g\epsilon^2}{Ak}\right) + \exp\left(\frac{-ga^4}{A'(p+k)}\right)\right) - (C-1)
\end{aligned}$$

$$= 1 - Ck\left(\exp\left(\frac{-g\epsilon^2}{Ak}\right) + \exp\left(\frac{-ga^4}{A'(p+k)}\right)\right),$$

as required. □ □

Now we are ready to prove the main assertion regarding the probability that the solution returned by Algorithm 1 is indeed the true optimal solution. We have

$$\begin{aligned}
& \mathbb{P}(c(\tilde{\boldsymbol{s}}^*) = c(\boldsymbol{s}^*)) \\
= \quad & \mathbb{P}(c(\tilde{\boldsymbol{s}}^*) - c(\boldsymbol{s}^*) < b) \quad \text{(Lemma 19)} \\
\geq \quad & \mathbb{P}\left(c(\tilde{\boldsymbol{s}}^*) - \tilde{c}_G^F(\tilde{\boldsymbol{s}}^*) < \tfrac{b}{2}\right) + \mathbb{P}\left(\tilde{c}_G^F(\tilde{\boldsymbol{s}}^*) - c(\boldsymbol{s}^*) < \tfrac{b}{2}\right) - 1 \quad \text{(By Frechet's inequalities)} \\
\geq \quad & \mathbb{P}\left(c(\tilde{\boldsymbol{s}}^*) - \tilde{c}_G^F(\tilde{\boldsymbol{s}}^*) < \tfrac{b}{2}\right) + \mathbb{P}\left(\eta^*(\boldsymbol{s}^*) - c(\boldsymbol{s}^*) < \tfrac{b}{2}\right) - 1 \quad \text{(Lemma 20)} \\
\geq \quad & 1 - k\exp\left(\tfrac{-gb^2}{4Ak}\right) - Ck\left(\exp\left(\tfrac{-gb^2}{4Ak}\right) + \exp\left(\tfrac{-ga^4}{A'(p+k)}\right)\right) \quad \text{(Theorem 6, Lemma 21)} \\
= \quad & 1 - Ck\exp\left(\tfrac{-Dga^4}{p+k}\right) - (C+1)k\exp\left(\tfrac{-Eg}{k}\right),
\end{aligned}$$

for some constants D, E as required. □

B.4 List of Features Used in the Netflix Problem

- 24 Indicator Variables for Genres: Action, Adventure, Animation, Biography, Comedy, Crime, Documentary, Drama, Family, Fantasy, Film Noir, History, Horror, Music, Musical, Mystery, Romance, Sci-Fi, Short, Sport, Superhero, Thriller, War, Western
- 5 Indicator Variables for Release Date: Within last 10 years, Between 10-20 years, Between 20-30 years, Between 30-40 years, Between 40-50 Years
- 6 Indicator Variables for Top Actors/Actresses defined by their Influence Score at time of release: Top 100 Actors, Top 100 Actresses, Top 250 Actors, Top 250 Actresses, Top 1000 Actors, Top 1000 Actresses
- IMDB Rating
- Number of Reviews
- Total Production Budget
- Total Runtime
- Total Box Office Revenue
- Indicator Variable for whether it is US produced
- 11 Indicator Variables for Month of Year Released (January removed to prevent multicollinearity)
- Number of Original Music Score
- Number of Male Actors
- Number of Female Factors
- 3 Indicator Variables for Film Language: English, French, Japanese
- Constant

Appendix C

Appendix to Chapter 4

C.1 Proof of Lemma 1

We only prove the lemma for the function $f(\boldsymbol{z}, \boldsymbol{\theta}; [N])$ as the derivative ∇f follows the exact same procedure just with additional notation.

First, by the matrix inversion lemma [176], for any feasible solution $\boldsymbol{z} \in \mathcal{Z}$, we can rewrite $f(\boldsymbol{z}, \boldsymbol{\theta}; [N])$ as:

$$\begin{aligned} f(\boldsymbol{z}, \boldsymbol{\theta}; [N]) &= \frac{1}{N}\boldsymbol{y}^T \left(I_N - \boldsymbol{X}_{\boldsymbol{z}} \left(\frac{I_k}{\gamma} + \boldsymbol{X}_{\boldsymbol{z}}^T \boldsymbol{X}_{\boldsymbol{z}} \right)^{-1} \boldsymbol{X}_{\boldsymbol{z}} \right) \boldsymbol{y} \\ &= \frac{1}{N}\boldsymbol{y}^T\boldsymbol{y} - \frac{1}{N}\boldsymbol{y}^T \boldsymbol{X}_{\boldsymbol{z}} \left(\frac{I_k}{\gamma} + \boldsymbol{X}_{\boldsymbol{z}}^T \boldsymbol{X}_{\boldsymbol{z}} \right)^{-1} \boldsymbol{X}_{\boldsymbol{z}}\boldsymbol{y} \\ &= \frac{1}{N}\boldsymbol{y}^T\boldsymbol{y} - \frac{1}{N}\boldsymbol{y}^T \boldsymbol{V} \left(\frac{I_k}{\gamma} + \boldsymbol{V}^T \boldsymbol{V} \right)^{-1} \boldsymbol{V}\boldsymbol{y}, \end{aligned} \tag{C.1}$$

where $\boldsymbol{V} = \boldsymbol{X}_{\boldsymbol{z}} \in \mathbb{R}^{N \times k}$ is the subset of the design matrix $\boldsymbol{X}$ with only the columns $\{i \mid z_i = 1\}$. For simplicity of notation, we use $\boldsymbol{V}$ to suppress the dependence on $\boldsymbol{z}$.

We let S be a subset of $[N]$ of size n (we suppress the dependence of S on n to reduce notation complexity). Then, we have, similar to Equation (C.1):

$$f(\boldsymbol{z}, \boldsymbol{\theta}; S) = \frac{1}{n}\boldsymbol{y}_S^T\boldsymbol{y}_S - \frac{1}{n}\boldsymbol{y}_S^T \boldsymbol{V}_S \left(\frac{I_k}{\gamma} + \boldsymbol{V}_S^T \boldsymbol{V}_S \right)^{-1} \boldsymbol{V}_S\boldsymbol{y}_S, \tag{C.2}$$

where $\boldsymbol{y}_S$ is the subvector of $\boldsymbol{y}$ with only the entries in set S, and similar for $\boldsymbol{V}$. Therefore,

what we need to prove is that:

$$\mathbb{P}\left(\left|\frac{1}{N}\boldsymbol{y}^T\boldsymbol{y} - \frac{1}{N}\boldsymbol{y}^T\boldsymbol{V}\left(\frac{I_k}{\gamma} + \boldsymbol{V}^T\boldsymbol{V}\right)^{-1}\boldsymbol{V}\boldsymbol{y} - \frac{1}{n}\boldsymbol{y}_S^T\boldsymbol{y}_S + \frac{1}{n}\boldsymbol{y}_S^T\boldsymbol{V}_S\left(\frac{I_k}{\gamma} + \boldsymbol{V}_S^T\boldsymbol{V}_S\right)^{-1}\boldsymbol{V}_S\boldsymbol{y}_S\right|\right.$$
$$\left.\leq M\sqrt{\frac{\log(\frac{1}{\epsilon})}{n}}\right)$$
$$\geq 1-\epsilon. \tag{C.3}$$

We now introduce and prove lemmas that allow us to prove Equation (C.3):

Lemma 22. *Let S be a random subset of $[N]$ of size n. Then, we have*

$$\mathbb{P}\left(\left|\frac{\boldsymbol{y}_S^T\boldsymbol{y}_S}{n} - \frac{\boldsymbol{y}^T\boldsymbol{y}}{N}\right| \leq \sqrt{A\left(\frac{1}{n}-\frac{1}{N}\right)\log(\frac{1}{\epsilon})}\right) \geq 1-\epsilon, \tag{C.4}$$

$$\mathbb{P}\left(\left\|\frac{\boldsymbol{y}_S^T\boldsymbol{V}_S}{n} - \frac{\boldsymbol{y}^T\boldsymbol{V}}{N}\right\| \leq \sqrt{B\left(\frac{1}{n}-\frac{1}{N}\right)\log(\frac{1}{\epsilon})}\right) \geq 1-\epsilon. \tag{C.5}$$

Proof. Proof: Notice that we have:

$$\boldsymbol{y}^T\boldsymbol{y} = \sum_{i=1}^N y_i^2, \qquad \boldsymbol{y}^T\boldsymbol{V} = \sum_{i=1}^N y_i\boldsymbol{v}_i,$$
$$\boldsymbol{y}_S^T\boldsymbol{y}_S = \sum_{i\in S} y_i^2, \qquad \boldsymbol{y}_S^T\boldsymbol{V}_S = \sum_{i\in S} y_i\boldsymbol{v}_i.$$

Therefore, if we treat $y_1^2, \ldots y_N^2$ as a finite population, then $\boldsymbol{y}_S^T\boldsymbol{y}_S$ is a random sample of n points drawn without replacement from that set, and similarly for $\boldsymbol{y}_S^T\boldsymbol{V}_S$. Then, the required inequality follows directly from Hoeffding's Inequality in the case without replacement, [11] Then, we have the following lemma:

Lemma 23.

$$\mathbb{P}\left(\left\|\left(\frac{I_k}{n\gamma} + \frac{\boldsymbol{V}_S^T\boldsymbol{V}_S}{n}\right)^{-1} - \left(\frac{I_k}{N\gamma} + \frac{\boldsymbol{V}^T\boldsymbol{V}}{N}\right)^{-1}\right\| \leq \sqrt{\frac{H\log(\frac{1}{\epsilon})}{n}}\right) \geq 1-\epsilon. \tag{C.6}$$

To prove this, we first introduce a matrix analog of the well-known Chernoff bound, the

proof of which can be found in [164] (adapted for the sampling without replacement case):

Lemma 24. *Let $\mathcal{X} \in \mathbb{R}^{k\times k}$ be a set of N positive-semidefinite matrices, and suppose that:*

$$\max_{\boldsymbol{X}\in\mathcal{X}} \lambda_{\max}(\boldsymbol{X}) \leq D,$$

where $\lambda_{\min}/\lambda_{\max}$is the minimum/maximum eigenvalue function. Sample $\{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n\}$ uniformly at random without replacement. Then

$$\mathbb{P}\left\{\lambda_{\min}\left(\frac{\sum_j \boldsymbol{X}_j}{n}\right) \leq (1-\delta)\lambda_{\min}(\mathbb{E}\boldsymbol{X}_1)\right\} \leq k \cdot \exp\left(\frac{-\delta^2\lambda_{\min}(\mathbb{E}\boldsymbol{X}_1)}{4D(1/n - 1/N)}\right), \quad \textit{for } \delta \in [0,1),$$
$$\mathbb{P}\left\{\lambda_{\max}\left(\frac{\sum_j \boldsymbol{X}_j}{n}\right) \leq (1+\delta)\lambda_{\max}(\mathbb{E}\boldsymbol{X}_1)\right\} \leq k \cdot \exp\left(\frac{-\delta^2\lambda_{\max}(\mathbb{E}\boldsymbol{X}_1)}{4D(1/n - 1/N)}\right), \quad \textit{for } \delta \geq 0.$$

Now we proceed with the proof.

Proof. Proof of Lemma 23: First, we do a QR decomposition of the matrix $\boldsymbol{V} = \boldsymbol{Q}\boldsymbol{R}$ where $\boldsymbol{Q} \in \mathbb{R}^{N\times k}$ has orthogonal columns and $\boldsymbol{R} \in \mathbb{R}^{k\times k}$ is upper triangular. We further normalize the decomposition so we have that $\frac{\boldsymbol{Q}^T\boldsymbol{Q}}{N} = \boldsymbol{I}_k$. Then, we have that $\boldsymbol{V}_S = \boldsymbol{Q}_S\boldsymbol{R}$, and we note that:

$$\boldsymbol{Q}^T\boldsymbol{Q} = \sum_{i=1}^{N} \boldsymbol{q}_i\boldsymbol{q}_i^T,$$
$$\boldsymbol{Q}_S^T\boldsymbol{Q}_S = \sum_{i\in S} \boldsymbol{q}_i\boldsymbol{q}_i^T,$$

where $\boldsymbol{q}_i\boldsymbol{q}_i^T \in \mathbb{R}^{k\times k}$ rank-one positive semi-definite matrices. Therefore, we can take $\boldsymbol{Q}_S^T\boldsymbol{Q}_S$ as a random sample of size n from the set $\mathcal{X} = \{\boldsymbol{q}_i\boldsymbol{q}_i^T\}_{i\in[N]}$, which satisfies the conditions in Lemma 24 with $D = O(k)$. Therefore, we use Lemma 24 to bound $\boldsymbol{Q}_S^T\boldsymbol{Q}_S$ that:

$$\mathbb{P}\left\{\lambda_{\min}\left(\frac{\boldsymbol{Q}_S^T\boldsymbol{Q}_S}{n}\right) \leq (1-\delta)\right\} \leq k \cdot \exp\left(\frac{-\delta^2}{kD(1/n - 1/N)}\right),$$
$$\mathbb{P}\left\{\lambda_{\max}\left(\frac{\boldsymbol{Q}_S^T\boldsymbol{Q}_S}{n}\right) \geq (1+\delta)\right\} \leq k \cdot \exp\left(\frac{-\delta^2}{kD(1/n - 1/N)}\right),$$

for some absolute constant D. Some rearrangement gives:

$$\mathbb{P}\left\{\lambda_{\min}\left(\frac{\boldsymbol{Q}_S^T\boldsymbol{Q}_S}{n}\right) \geq 1 - \sqrt{\left(\frac{1}{n}-\frac{1}{N}\right)kD\log\left(\frac{2k}{\epsilon}\right)} \right. \tag{C.7}$$

$$\left. \text{and } \lambda_{\max}\left(\frac{\boldsymbol{Q}_S^T\boldsymbol{Q}_S}{n}\right) \leq 1 + \sqrt{\left(\frac{1}{n}-\frac{1}{N}\right)kD\log\left(\frac{2k}{\epsilon}\right)}\right\}$$

$$\geq 1-\epsilon. \tag{C.8}$$

Now since $\frac{\boldsymbol{Q}^T\boldsymbol{Q}}{N} = \boldsymbol{I}_k$, we have:

$$\lambda_{\min}\left(\frac{\boldsymbol{Q}^T\boldsymbol{Q}}{N}\right) = \lambda_{\max}\left(\frac{\boldsymbol{Q}^T\boldsymbol{Q}}{N}\right) = 1. \tag{C.9}$$

Combining equation (C.9) and (C.8) we obtain

$$\mathbb{P}\left\{\left\|\frac{\boldsymbol{Q}_S^T\boldsymbol{Q}_S}{n} - \frac{\boldsymbol{Q}^T\boldsymbol{Q}}{N}\right\| \leq \sqrt{\left(\frac{1}{n}-\frac{1}{N}\right)kD\log\left(\frac{2k}{\epsilon}\right)}\right\} \geq 1-\epsilon. \tag{C.10}$$

Thus, we have

$$\mathbb{P}\left\{\left\|\frac{\boldsymbol{V}_S^T\boldsymbol{V}_S}{n} - \frac{\boldsymbol{V}^T\boldsymbol{V}}{N}\right\| \leq \|\boldsymbol{R}\|^2\sqrt{\left(\frac{1}{n}-\frac{1}{N}\right)kD\log\left(\frac{2k}{\epsilon}\right)}\right\}$$

$$\geq \mathbb{P}\left\{\left\|\frac{\boldsymbol{Q}_S^T\boldsymbol{Q}_S}{n} - \frac{\boldsymbol{Q}^T\boldsymbol{Q}}{N}\right\| \leq \sqrt{\left(\frac{1}{n}-\frac{1}{N}\right)kD\log\left(\frac{2k}{\epsilon}\right)}\right\} \geq 1-\epsilon.$$

We let $D' = \log(2)\|\boldsymbol{R}\|^4 D$ and obtain

$$\mathbb{P}\left\{\left\|\frac{\boldsymbol{V}_S^T\boldsymbol{V}_S}{n} - \frac{\boldsymbol{V}^T\boldsymbol{V}}{N}\right\| \leq \sqrt{\left(\frac{1}{n}-\frac{1}{N}\right)kD'\log\left(\frac{k}{\epsilon}\right)}\right\} \geq 1-\epsilon. \tag{C.11}$$

From this, we can easily see that there exists a constant G such that:

$$\mathbb{P}\left\{\left\|\left(\frac{\boldsymbol{I}_k}{n\gamma} + \frac{\boldsymbol{V}_S^T\boldsymbol{V}_S}{n}\right) - \left(\frac{\boldsymbol{I}_k}{N\gamma} + \frac{\boldsymbol{V}^T\boldsymbol{V}}{N}\right)\right\| \leq \sqrt{\left(\frac{1}{n}-\frac{1}{N}\right)kG\log\left(\frac{k}{\epsilon}\right)}\right\} \geq 1-\epsilon, \tag{C.12}$$

as $\frac{\boldsymbol{I}_k}{n\gamma}$ and $\frac{\boldsymbol{I}_k}{N\gamma}$ are of order of magnitude $O(\frac{1}{n})$. Now let us introduce another lemma from

matrix perturbation theory (for proof, see e.g. [157]):

Lemma 25. *Let $\boldsymbol{A}, \boldsymbol{B}$ be invertible matrices. Then, we have the following bound:*

$$\|\boldsymbol{A}^{-1} - \boldsymbol{B}^{-1}\| \leq \|\boldsymbol{A}^{-1}\|\|\boldsymbol{B}^{-1}\|\|\boldsymbol{A} - \boldsymbol{B}\|. \tag{C.13}$$

Thus, now let $H = G\left(\max_{S,1\leq n\leq N}\left\|\left(\frac{\boldsymbol{I}_k}{n\gamma} + \frac{\boldsymbol{V}_S^T\boldsymbol{V}_S}{n}\right)^{-1}\right\|^2\right)\left\|\left(\frac{\boldsymbol{I}_k}{N\gamma} + \frac{\boldsymbol{V}^T\boldsymbol{V}}{N}\right)^{-1}\right\|^2$, be a constant (note that the maximum is finite by finiteness of the the set). Then we have, using Lemma 25 and Equation (C.12):

$$\begin{aligned}
&\mathbb{P}\left\{\left\|\left(\frac{\boldsymbol{I}_k}{n\gamma} + \frac{\boldsymbol{V}_S^T\boldsymbol{V}_S}{n}\right)^{-1} - \left(\frac{\boldsymbol{I}_k}{N\gamma} + \frac{\boldsymbol{V}^T\boldsymbol{V}}{N}\right)^{-1}\right\| \leq \sqrt{\left(\frac{1}{n} - \frac{1}{N}\right)kH\log\left(\frac{k}{\epsilon}\right)}\right\} \\
&\geq \mathbb{P}\left\{\left\|\left(\frac{\boldsymbol{I}_k}{n\gamma} + \frac{\boldsymbol{V}_S^T\boldsymbol{V}_S}{n}\right) - \left(\frac{\boldsymbol{I}_k}{N\gamma} + \frac{\boldsymbol{V}^T\boldsymbol{V}}{N}\right)\right\|\left\|\left(\frac{\boldsymbol{I}_k}{n\gamma} + \frac{\boldsymbol{V}_S^T\boldsymbol{V}_S}{n}\right)^{-1}\right\|\left\|\left(\frac{\boldsymbol{I}_k}{N\gamma} + \frac{\boldsymbol{V}^T\boldsymbol{V}}{N}\right)^{-1}\right\|\right. \\
&\left.\leq \sqrt{\left(\frac{1}{n} - \frac{1}{N}\right)kH\log\left(\frac{k}{\epsilon}\right)}\right\} \\
&\geq \mathbb{P}\left\{\left\|\left(\frac{\boldsymbol{I}_k}{n\gamma} + \frac{\boldsymbol{V}_S^T\boldsymbol{V}_S}{n}\right) - \left(\frac{\boldsymbol{I}_k}{N\gamma} + \frac{\boldsymbol{V}^T\boldsymbol{V}}{N}\right)\right\| \leq \sqrt{\left(\frac{1}{n} - \frac{1}{N}\right)kG\log\left(\frac{k}{\epsilon}\right)}\right\} \\
&\geq 1 - \epsilon.
\end{aligned}$$

As required. We then absorb $H' = k\log(k)H$ as k is a dimension independent from the variable dimension p and the sample dimension n. □ □

With Lemma 22 and 23, we are now ready to prove the main result in Equation (C.3). First, let M be a constant such that:

$$\begin{aligned}
\sqrt{M} \geq \max_{S,n}\max\Bigg\{&\sqrt{4A}, 36\left\|\left(\frac{I_k}{N\gamma} + \frac{\boldsymbol{V}^T\boldsymbol{V}}{N}\right)^{-1}\frac{\boldsymbol{V}\boldsymbol{y}}{N}\right\|^2 B, \\
&36\left\|\frac{\boldsymbol{y}_S^T\boldsymbol{V}_S}{n}\right\|\left\|\frac{\boldsymbol{V}\boldsymbol{y}}{N}\right\|^2 H, 36\left\|\left(\frac{I_k}{n\gamma} + \frac{\boldsymbol{V}_S^T\boldsymbol{V}_S}{n}\right)^{-1}\frac{\boldsymbol{V}_S\boldsymbol{y}_S}{n}\right\|^2 B\Bigg\}.
\end{aligned}$$

We introduce the Frechet inequalities, which state that:

$$\mathbb{P}(A_1 \cap A_2 \cap \ldots \cap A_n) \geq \mathbb{P}(A_1) + \mathbb{P}(A_2) + \ldots + \mathbb{P}(A_n) - (n-1).$$

Then we have, by Frechet's inequalities:

$$\begin{aligned}
&\mathbb{P}\left(\left|\frac{1}{N}\boldsymbol{y}^T\boldsymbol{y} - \frac{1}{N}\boldsymbol{y}^T\boldsymbol{V}\left(\frac{I_k}{\gamma} + \boldsymbol{V}^T\boldsymbol{V}\right)^{-1}\boldsymbol{V}\boldsymbol{y} - \frac{1}{n}\boldsymbol{y}_S^T\boldsymbol{y}_S + \frac{1}{n}\boldsymbol{y}_S^T\boldsymbol{V}_S\left(\frac{I_k}{\gamma} + \boldsymbol{V}_S^T\boldsymbol{V}_S\right)^{-1}\boldsymbol{V}_S\boldsymbol{y}_S\right|\right.\\
&\left.\leq \sqrt{M\left(\frac{1}{n} - \frac{1}{N}\right)\log(\frac{1}{\epsilon})}\right)\\
&\geq \mathbb{P}\left(\left|\frac{1}{N}\boldsymbol{y}^T\boldsymbol{y} - \frac{1}{n}\boldsymbol{y}_S^T\boldsymbol{y}_S\right| \leq \frac{1}{2}\sqrt{M\left(\frac{1}{n} - \frac{1}{N}\right)\log(\frac{1}{\epsilon})}\right) +\\
&+ \mathbb{P}\left(\left|\frac{1}{N}\boldsymbol{y}^T\boldsymbol{V}\left(\frac{I_k}{\gamma} + \boldsymbol{V}^T\boldsymbol{V}\right)^{-1}\boldsymbol{V}\boldsymbol{y} - \frac{1}{n}\boldsymbol{y}_S^T\boldsymbol{V}_S\left(\frac{I_k}{\gamma} + \boldsymbol{V}_S^T\boldsymbol{V}_S\right)^{-1}\boldsymbol{V}_S\boldsymbol{y}_S\right|\right.\\
&\left.\leq \frac{1}{2}\sqrt{M\left(\frac{1}{n} - \frac{1}{N}\right)\log(\frac{1}{\epsilon})}\right) - 1.
\end{aligned}$$

Again, by Frechet inequalities, we can further breakdown the expression above to:

$$\begin{aligned}
&\geq \mathbb{P}\left(\left|\frac{1}{N}\boldsymbol{y}^T\boldsymbol{y} - \frac{1}{n}\boldsymbol{y}_S^T\boldsymbol{y}_S\right| \leq \frac{1}{2}\sqrt{M\left(\frac{1}{n} - \frac{1}{N}\right)\log(\frac{1}{\epsilon})}\right)\\
&+ \mathbb{P}\left(\left|\left(\frac{1}{N}\boldsymbol{y}^T\boldsymbol{V} - \frac{1}{n}\boldsymbol{y}_S^T\boldsymbol{V}_S\right)\left(\frac{I_k}{N\gamma} + \frac{\boldsymbol{V}^T\boldsymbol{V}}{N}\right)^{-1}\frac{\boldsymbol{V}\boldsymbol{y}}{N}\right| \leq \frac{1}{6}\sqrt{M\left(\frac{1}{n} - \frac{1}{N}\right)\log(\frac{1}{\epsilon})}\right)\\
&+ \mathbb{P}\left(\left|\frac{1}{n}\boldsymbol{y}_S^T\boldsymbol{V}_S\left(\left(\frac{I_k}{N\gamma} + \frac{\boldsymbol{V}^T\boldsymbol{V}}{N}\right)^{-1} - \left(\frac{I_k}{n\gamma} + \frac{\boldsymbol{V}_S^T\boldsymbol{V}_S}{n}\right)^{-1}\right)\frac{\boldsymbol{V}\boldsymbol{y}}{N}\right|\right.\\
&\left.\leq \frac{1}{6}\sqrt{M\left(\frac{1}{n} - \frac{1}{N}\right)\log(\frac{1}{\epsilon})}\right)\\
&+ \mathbb{P}\left(\left|\frac{1}{n}\boldsymbol{y}_S^T\boldsymbol{V}_S\left(\frac{I_k}{n\gamma} + \frac{\boldsymbol{V}_S^T\boldsymbol{V}_S}{n}\right)^{-1}\left(\frac{\boldsymbol{V}\boldsymbol{y}}{n} - \frac{\boldsymbol{V}_S\boldsymbol{y}_S}{n}\right)\right| \leq \frac{1}{6}\sqrt{M\left(\frac{1}{n} - \frac{1}{N}\right)\log(\frac{1}{\epsilon})}\right) - 3
\end{aligned}$$

By the definition of M, we can simplify the expressions above to:

$$\geq \mathbb{P}\left(\left|\frac{1}{N}\boldsymbol{y}^T\boldsymbol{y} - \frac{1}{n}\boldsymbol{y}_S^T\boldsymbol{y}_S\right| \leq \sqrt{\left(\frac{1}{n} - \frac{1}{N}\right)A\log\left(\frac{1}{\epsilon}\right)}\right)$$

$$+\mathbb{P}\left(\left\|\frac{1}{N}\boldsymbol{y}^T\boldsymbol{V}-\frac{1}{n}\boldsymbol{y}_S^T\boldsymbol{V}_S\right\|\leq\sqrt{\left(\frac{1}{n}-\frac{1}{N}\right)B\log\left(\frac{1}{\epsilon}\right)}\right)$$
$$+\mathbb{P}\left(\left\|\left(\frac{I_k}{N\gamma}+\frac{\boldsymbol{V}^T\boldsymbol{V}}{N}\right)^{-1}-\left(\frac{I_k}{n\gamma}+\frac{\boldsymbol{V}_S^T\boldsymbol{V}_S}{n}\right)^{-1}\right\|\leq\sqrt{\left(\frac{1}{n}-\frac{1}{N}\right)H\log\left(\frac{1}{\epsilon}\right)}\right)$$
$$+\mathbb{P}\left(\left\|\frac{\boldsymbol{V}\boldsymbol{y}}{n}-\frac{\boldsymbol{V}_S\boldsymbol{y}_S}{n}\right\|\leq\sqrt{\left(\frac{1}{n}-\frac{1}{N}\right)B\log\left(\frac{1}{\epsilon}\right)}\right)-3.$$

Now we apply Lemma 22 and 23 to evaluate these expressions:

$$\geq 4-4\epsilon-3$$
$$=1-4\epsilon.$$

Thus, taking $\epsilon'=4\epsilon$, and $M'=\frac{M}{\log(2)}$ gives the result we want.

C.2 Proof of Theorem 1

C.2.1 Proof of Finiteness

We first prove that the outer approximation algorithm indeed converges in finite number of steps C with probability 1. To do so, we note that the set of feasible solutions for the integer variables $\mathcal{Z}$ is finite, and we prove that the series of feasible integer solutions $\boldsymbol{z}_1, \boldsymbol{z}_2, \cdots$ created during Algorithm 6 never repeats indefinitely, with probability 1.

First, we show that if we have $\boldsymbol{z}_a = \boldsymbol{z}_b$ for some $a+1 \leq b$ where we have not terminated at iteration b, then we must have $f(\boldsymbol{z}_b, \boldsymbol{\theta}_b; [S_b]) > f(\boldsymbol{z}_a, \boldsymbol{\theta}_a; S_a])$. At iteration a, the following cutting plane was added:

$$\eta \geq f(\boldsymbol{z}_a, \boldsymbol{\theta}_a; [S_a]) + \nabla f(\boldsymbol{z}_a, \boldsymbol{\theta}_a; [S_a])^T \begin{pmatrix} \boldsymbol{z} - \boldsymbol{z}_a \\ \boldsymbol{\theta} - \boldsymbol{\theta}_a \end{pmatrix}. \tag{C.14}$$

Therefore, using (C.14) at iteration $b-1 \geq a$ (note $\boldsymbol{z}_b, \boldsymbol{\theta}_b, \eta_b$ is the optimal solution in the $b-1$th iteration, so it must satisfy the inequality in (C.14)) gives:

$$\eta_b \geq f(\boldsymbol{z}_a, \boldsymbol{\theta}_a; [S_a]) + \nabla f(\boldsymbol{z}_a, \boldsymbol{\theta}_a; [S_a])^T \begin{pmatrix} \boldsymbol{z}_b - \boldsymbol{z}_a \\ \boldsymbol{\theta}_b - \boldsymbol{\theta}_a \end{pmatrix} \tag{C.15}$$

$$= f(\boldsymbol{z}_a, \boldsymbol{\theta}_a; [S_a]) + \nabla_{\boldsymbol{\theta}} f(\boldsymbol{z}_a, \boldsymbol{\theta}_a; [S_a])^T (\boldsymbol{\theta}_b - \boldsymbol{\theta}_a) \tag{C.16}$$

$$\geq f(\boldsymbol{z}_a, \boldsymbol{\theta}_a; [S_a]). \tag{C.17}$$

The last inequality follows because $\boldsymbol{\theta}_a$ is the optimal solution to NLP$(\boldsymbol{\theta}_a; [S_a])$, so no feasible descent direction can exist at $\boldsymbol{\theta}_a$. Now consider the cutting plane problem at iteration b, where the following cutting plane was added:

$$\eta \geq f(\boldsymbol{z}_b, \boldsymbol{\theta}_b; [S_b]) + \nabla f(\boldsymbol{z}_b, \boldsymbol{\theta}_b; [S_b])^T \begin{pmatrix} \boldsymbol{z} - \boldsymbol{z}_b \\ \boldsymbol{\theta} - \boldsymbol{\theta}_b \end{pmatrix}. \tag{C.18}$$

Assume that we have $f(\boldsymbol{z}_b, \boldsymbol{\theta}_b; [S_b]) \leq f(\boldsymbol{z}_a, \boldsymbol{\theta}_a; [S_a])$. Then $\eta = \eta_b$, $\boldsymbol{z} = \boldsymbol{z}_b$ is a feasible solution for the cutting plane problem at iteration b since it satisfies the newly added constraint: $\eta_b \geq f(\boldsymbol{z}_a, \boldsymbol{\theta}_a; [S_a]) \geq f(\boldsymbol{z}_b, \boldsymbol{\theta}_b, [S_b]) + \nabla f(\boldsymbol{z}_b, \boldsymbol{\theta}_b; [S_b])^T \binom{\boldsymbol{z}_b - \boldsymbol{z}_b}{\boldsymbol{\theta}_b - \boldsymbol{\theta}_b} = f(\boldsymbol{z}_b, \boldsymbol{\theta}_b; [S_b])$. By construction of the algorithm, we therefore satisfy the termination condition, so the algo-

rithm terminates, a contradiction. Therefore, we must have $f(\boldsymbol{z}_b, \boldsymbol{\theta}_b; [S_b]) > f(\boldsymbol{z}_a, \boldsymbol{\theta}_a; [S_a])$.

Then consider Algorithm 6 after $Q = |\mathcal{Z}|(q-1)+1$ iterations where $q > 0$ is a positive integer, with the corresponding series of feasible integer solutions being $\boldsymbol{z}_1, \cdots, \boldsymbol{z}_Q$. Then since there are only $|\mathcal{Z}|$ feasible solutions, by the pigeonhole principle, there must be $\boldsymbol{z} \in \mathcal{Z}$ that has appeared at least q times in the series, and denote these as $\boldsymbol{z}_{l_1}, \cdots, \boldsymbol{z}_{l_q}$. By the observation above, we know that if Algorithm 6 has not terminated, we must have:

$$f(\boldsymbol{z}_{l_1}, \boldsymbol{\theta}_{l_1}; [S_{l_1}]) < f(\boldsymbol{z}_{l_2}, \boldsymbol{\theta}_{l_2}; [S_{l_2}]) < \cdots < f(\boldsymbol{z}_{l_q}, \boldsymbol{\theta}_{l_q}; [S_{l_q}])$$

Now note that $\boldsymbol{z}_{l_1} = \boldsymbol{z}_{l_2} = \cdots = \boldsymbol{z}_{l_q}$. Given $\boldsymbol{z}_t$, $f(\boldsymbol{z}_t, \boldsymbol{\theta}_t; [S_t])$ is a random variable that only depends on S_t (by construction of the algorithm, $\boldsymbol{\theta}_t$ is the optimal solution to NLP$(\boldsymbol{z}_t, [S_t])$. Therefore, we have that $f(\boldsymbol{z}_{l_1}, \boldsymbol{\theta}_{l_1}; [S_{l_1}]), \cdots, f(\boldsymbol{z}_{l_q}, \boldsymbol{\theta}_{l_q}; [S_{l_q}])$ are q independent realizations (since $S_{l_1}, S_{l_2}, \cdots$ are independent) of an identically distributed random variable, and thus by standard probability results (see e.g. [68]) we have that:

$$\mathbb{P}(f(\boldsymbol{z}_{l_1}, \boldsymbol{\theta}_{l_1}; [S_{l_1}]) < f(\boldsymbol{z}_{l_2}, \boldsymbol{\theta}_{l_2}; [S_{l_2}]) < \cdots < f(\boldsymbol{z}_{l_q}, \boldsymbol{\theta}_{l_q}; [S_{l_q}])) \leq \frac{1}{q!}$$

The result follows by noting that $f(\boldsymbol{z}_{l_1}, \boldsymbol{\theta}_{l_1}; [S_{l_1}]), \cdots, f(\boldsymbol{z}_{l_q}, \boldsymbol{\theta}_{l_q}; [S_{l_q}])$ are exchangable random variables so any ordering is equally likely, and only at most 1 out of all $q!$ permutations of $f(\boldsymbol{z}_{l_1}, \boldsymbol{\theta}_{l_1}; [S_{l_1}]), \cdots, f(\boldsymbol{z}_{l_q}, \boldsymbol{\theta}_{l_q}; [S_{l_q}])$ satisfy the monotone relationship. Then we have that:

$$\begin{aligned}
&\mathbb{P}(\text{Algorithm has not terminated after } Q \text{ iterations} \mid \boldsymbol{z}_1, \cdots \boldsymbol{z}_Q) \\
&\leq \mathbb{P}(f(\boldsymbol{z}_{l_1}, \boldsymbol{\theta}_{l_1}; [S_{l_1}]) < f(\boldsymbol{z}_{l_2}, \boldsymbol{\theta}_{l_2}; [S_{l_2}]) < \cdots < f(\boldsymbol{z}_{l_q}, \boldsymbol{\theta}_{l_q}; [S_{l_q}])) \\
&\leq \frac{1}{q!}
\end{aligned}$$

Therefore, by taking expectations on both sides, we have:

$$\mathbb{P}(\text{Algorithm has not terminated after } Q \text{ iterations}) \leq \frac{1}{q!},$$

which converges to 0 as $Q \to \infty$.

Now we can bound the number of iterations K when the algorithm terminates. Then by

we have:

$$\begin{aligned}\mathbb{E}[K] &= \sum_{i=0}^{\infty} \mathbb{P}(K > i) \\ &\leq \sum_{q=1}^{\infty} |\mathcal{Z}| \mathbb{P}(K > |Z|(q-1)+1) + 1 \\ &= \sum_{q=1}^{\infty} |\mathcal{Z}| \frac{1}{q!} + 1 \\ &= (e-1)|\mathcal{Z}| + 1\end{aligned}$$

We have thus proved that the algorithm terminates in finite number of iterations with probability 1, and furthermore that its expected value is bounded by $(e-1)|\mathcal{Z}| + 1$.

C.2.2 Proof of Feasibility and Optimality

[144] has shown that when the cutting plane algorithm terminates, it returns infeasibility or the optimal (feasible) solution. In particular, the cutting plane algorithm returns infeasibility if and only if the original problem is infeasible. Now note that we have assumed the problem is indeed feasible and the formulation of constraints in the stochastic cutting plane algorithm is unchanged from the standard cutting plane algorithm. Therefore, Algorithm 6 returns a feasible solution when it terminates.

Therefore, we focus on proving that the solution is ϵ-optimal with high probability.

As in the theorem, let $(\boldsymbol{z}^*, \boldsymbol{\theta}^*)$ be an optimal solution for the original problem and $(\tilde{\boldsymbol{z}}^*, \tilde{\boldsymbol{\theta}}^*)$ be the solution returned by Algorithm 6. Now let us consider the MILO at the iteration when the algorithm terminated at the Kth iteration:

$$\min_{\boldsymbol{z} \in \mathcal{Z}, \boldsymbol{\theta} \in \Theta, \eta \geq lb} \quad \eta \tag{C.19}$$

$$\text{s.t.} \quad \eta \geq f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; S_i^n) + \nabla f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; S_i^n)^T \begin{pmatrix} \boldsymbol{z} - \boldsymbol{z}_i \\ \boldsymbol{\theta} - \boldsymbol{\theta}_i \end{pmatrix}, \quad \forall i \in [K], \tag{C.20}$$

$$0 \geq g_j(\boldsymbol{z}_i, \boldsymbol{\theta}_i) + \nabla g_j(\boldsymbol{z}_i, \boldsymbol{\theta}_i)^T \begin{pmatrix} \boldsymbol{z} - \boldsymbol{z}_i \\ \boldsymbol{\theta} - \boldsymbol{\theta}_i \end{pmatrix}, \quad \forall i \in [K], \ \ \forall j \in [m]. \tag{C.21}$$

For notational simplicity, we define $\eta^*(\boldsymbol{z}, \boldsymbol{\theta})$ to be the optimal value for η under the solution

$(\boldsymbol{z}, \boldsymbol{\theta})$ for the problem above. In particular, if $(\boldsymbol{z}, \boldsymbol{\theta})$ is feasible, then we have:

$$\eta^*(\boldsymbol{z}, \boldsymbol{\theta}) = \max_{i \in [K]} f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; S_i^n) + \nabla f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; S_i^n)^T \begin{pmatrix} \boldsymbol{z} - \boldsymbol{z}_i \\ \boldsymbol{\theta} - \boldsymbol{\theta}_i \end{pmatrix}. \tag{C.22}$$

Now we introduce a few lemmas:

Lemma 26. $\eta^*(\boldsymbol{z}^*, \boldsymbol{\theta}^*) \geq f(\tilde{\boldsymbol{z}}^*, \tilde{\boldsymbol{\theta}}^*; S_K^n)$.

Proof. Proof: Since the algorithm terminated at $(\tilde{\boldsymbol{z}}^*, \tilde{\boldsymbol{\theta}}^*)$, the termination condition for Algorithm 6 directly implies:

$$\eta^*(\tilde{\boldsymbol{z}}^*, \tilde{\boldsymbol{\theta}}^*) \geq f(\tilde{\boldsymbol{z}}^*, \tilde{\boldsymbol{\theta}}^*; S_K^n). \tag{C.23}$$

Now by optimality of $(\tilde{\boldsymbol{z}}^*, \tilde{\boldsymbol{\theta}}^*)$, we must have:

$$\eta^*(\boldsymbol{z}^*, \boldsymbol{\theta}^*) \geq \eta^*(\tilde{\boldsymbol{z}}^*, \tilde{\boldsymbol{\theta}}^*). \tag{C.24}$$

Combining equation (C.23) and (B.25) gives the required statement. □ □

Lemma 27. *Under the randomization scheme in Algorithm 6, we have, for some absolute constant D:*

$$\mathbb{P}\left(\eta^*(\boldsymbol{z}^*, \boldsymbol{\theta}^*) - f(\boldsymbol{z}^*, \boldsymbol{\theta}^*; [N]) \leq \epsilon\right) \geq 1 - 2K \exp\left(-\frac{n\epsilon^2}{(1 + \sqrt{p_1 + p_2})^2 D}\right). \tag{C.25}$$

Proof. Proof: Note that we can write $\eta^*(\boldsymbol{z}^*, \boldsymbol{\theta}^*)$ as:

$$\eta^*(\boldsymbol{z}^*, \boldsymbol{\theta}^*) = \max_{i \in [K]} f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; S_i^n) + \nabla f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; S_i^n)^T \begin{pmatrix} \boldsymbol{z}^* - \boldsymbol{z}_i \\ \boldsymbol{\theta}^* - \boldsymbol{\theta}_i \end{pmatrix}.$$

Now by Frechet's inequalities:

$$\begin{aligned}
&\mathbb{P}\left(\eta^*(\boldsymbol{z}^*, \boldsymbol{\theta}^*) - f(\boldsymbol{z}^*, \boldsymbol{\theta}^*; [N]) \leq \epsilon\right) \\
&= \mathbb{P}\left(\max_{i \in [K]} f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; S_i^n) + \nabla f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; S_i^n)^T \begin{pmatrix} \boldsymbol{z}^* - \boldsymbol{z}_i \\ \boldsymbol{\theta}^* - \boldsymbol{\theta}_i \end{pmatrix} - f(\boldsymbol{z}^*, \boldsymbol{\theta}^*; [N]) \leq \epsilon\right) \\
&= \mathbb{P}\left(f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; S_i^n) + \nabla f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; S_i^n)^T \begin{pmatrix} \boldsymbol{z}^* - \boldsymbol{z}_i \\ \boldsymbol{\theta}^* - \boldsymbol{\theta}_i \end{pmatrix} - f(\boldsymbol{z}^*, \boldsymbol{\theta}^*; [N]) \leq \epsilon \ \ \forall i \in [K]\right)
\end{aligned}$$

$$\geq \sum_{i=1}^{K} \mathbb{P}\left(f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; S_i^n) + \nabla f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; S_i^n)^T \begin{pmatrix} \boldsymbol{z}^* - \boldsymbol{z}_i \\ \boldsymbol{\theta}^* - \boldsymbol{\theta}_i \end{pmatrix} - f(\boldsymbol{z}^*, \boldsymbol{\theta}^*; [N]) \leq \epsilon \right) - (K-1). \tag{C.26}$$

We focus on each term:

$$\mathbb{P}\left(f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; S_i^n) + \nabla f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; S_i^n)^T \begin{pmatrix} \boldsymbol{z}^* - \boldsymbol{z}_i \\ \boldsymbol{\theta}^* - \boldsymbol{\theta}_i \end{pmatrix} - f(\boldsymbol{z}^*, \boldsymbol{\theta}^*; [N]) \leq \epsilon \right)$$

By the convexity assumption on f (Assumption 3), we have that:

$$f(\boldsymbol{z}^*, \boldsymbol{\theta}^*; [N]) \geq f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; [N]) + \nabla f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; [N])^T \begin{pmatrix} \boldsymbol{z}^* - \boldsymbol{z}_i \\ \boldsymbol{\theta}^* - \boldsymbol{\theta}_i \end{pmatrix}, \quad \forall i \in [K]. \tag{C.27}$$

Therefore, by substituting Equation (C.27), we have, for all $i \in [K]$:

$$\mathbb{P}\left(f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; S_i^n) + \nabla f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; S_i^n)^T \begin{pmatrix} \boldsymbol{z}^* - \boldsymbol{z}_i \\ \boldsymbol{\theta}^* - \boldsymbol{\theta}_i \end{pmatrix} - f(\boldsymbol{z}^*, \boldsymbol{\theta}^*; [N]) \leq \epsilon \right) \tag{C.28}$$

$$= \mathbb{P}\left((f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; S_i^n) - f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; [N])) + \left(\nabla f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; S_i^n)^T - \nabla f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; [N])^T\right) \begin{pmatrix} \boldsymbol{z}^* - \boldsymbol{z}_i \\ \boldsymbol{\theta}^* - \boldsymbol{\theta}_i \end{pmatrix} \leq \epsilon \right)$$

$$\geq \mathbb{P}\left((f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; S_i^n) - f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; [N])) \leq \frac{\epsilon}{1 + \sqrt{p_1 + p_2}} \right.$$

$$\left. \cap \ \left(\nabla f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; S_i^n)^T - \nabla f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; [N])^T\right) \begin{pmatrix} \boldsymbol{z}^* - \boldsymbol{z}_i \\ \boldsymbol{\theta}^* - \boldsymbol{\theta}_i \end{pmatrix} \leq \frac{(\sqrt{p_1 + p_2})\epsilon}{1 + \sqrt{p_1 + p_2}} \right). \tag{C.29}$$

Using the Frechet inequalities, we can further simplify the expression above:

$$(C.29) \geq \mathbb{P}\left((f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; S_i^n) - f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; [N])) \leq \frac{\epsilon}{1 + \sqrt{p_1 + p_2}} \right) \tag{C.30}$$

$$+ \mathbb{P}\left(\left(\nabla f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; S_i^n)^T - \nabla f(\boldsymbol{z}_i, \boldsymbol{\theta}_i; [N])^T\right) \begin{pmatrix} \boldsymbol{z}^* - \boldsymbol{z}_i \\ \boldsymbol{\theta}^* - \boldsymbol{\theta}_i \end{pmatrix} \leq \frac{(\sqrt{p_1 + p_2})\epsilon}{1 + \sqrt{p_1 + p_2}} \right) - 1.$$

Now by Assumption 5, these expressions lead to :

$$\geq 1 - \exp\left(-\frac{\epsilon^2}{(1 + \sqrt{p_1 + p_2})^2 (1/n - 1/N) M} \right) \tag{C.31}$$

$$+ 1 - \exp\left(-\frac{\epsilon^2}{(1 + \sqrt{p_1 + p_2})^2 (1/n - 1/N) M'} \right) - 1 \tag{C.32}$$

$$\geq 1 - 2\exp\left(-\frac{\epsilon^2}{(1 + \sqrt{p_1 + p_2})^2 (1/n - 1/N) \max(M, M')} \right), \tag{C.33}$$

where D is an absolute constant. Now we substitute Equation (C.33) into Equation (C.26) to get that:

$$\begin{aligned}
&\mathbb{P}\left(\eta^*(\boldsymbol{z}^*, \boldsymbol{\theta}^*) - f(\boldsymbol{z}^*, \boldsymbol{\theta}^*; [N]) \leq \epsilon\right) \\
&\geq K - 2K \exp\left(-\frac{\epsilon^2}{(1+\sqrt{p_1+p_2})^2(1/n - 1/N)\max(M, M')}\right) - (K-1) \\
&= 1 - 2K \exp\left(-\frac{\epsilon^2}{(1+\sqrt{p_1+p_2})^2(1/n - 1/N)\max(M, M')}\right).
\end{aligned}$$

as needed. □ □

We now combine the lemmas to derive the final inequality:

$$\begin{aligned}
&\mathbb{P}\left(f(\tilde{\boldsymbol{z}}^*, \tilde{\boldsymbol{\theta}}^*; [N]) - f(\boldsymbol{z}^*, \boldsymbol{\theta}^*; [N]) \leq \epsilon\right) \\
&\geq \mathbb{P}\left(f(\tilde{\boldsymbol{z}}^*, \tilde{\boldsymbol{\theta}}^*; [N]) - f(\tilde{\boldsymbol{z}}^*, \tilde{\boldsymbol{\theta}}^*; S_K^n) \leq \frac{\epsilon}{2+\sqrt{p_1+p_2}}\right. \\
&\left.\cap\, f(\tilde{\boldsymbol{z}}^*, \tilde{\boldsymbol{\theta}}^*; S_K^n) - f(\boldsymbol{z}^*, \boldsymbol{\theta}^*; N) \leq \frac{(1+\sqrt{p_1+p_2})\epsilon}{2+\sqrt{p_1+p_2}}\right).
\end{aligned}$$

By the Frechet inequalities, we can simplify the expression above to:

$$\begin{aligned}
&\geq \mathbb{P}\left(f(\tilde{\boldsymbol{z}}^*, \tilde{\boldsymbol{\theta}}^*; [N]) - f(\tilde{\boldsymbol{z}}^*, \tilde{\boldsymbol{\theta}}^*; S_K^n) \leq \frac{\epsilon}{2+\sqrt{p_1+p_2}}\right) \\
&+ \mathbb{P}\left(f(\tilde{\boldsymbol{z}}^*, \tilde{\boldsymbol{\theta}}^*; S_K^n) - f(\boldsymbol{z}^*, \boldsymbol{\theta}^*; N) \leq \frac{(1+\sqrt{p_1+p_2})\epsilon}{2+\sqrt{p_1+p_2}}\right) - 1.
\end{aligned}$$

Using Lemma 26, we can bound $f(\tilde{\boldsymbol{z}}^*, \tilde{\boldsymbol{\theta}}^*; S_K^n)$:

$$\begin{aligned}
&\geq \mathbb{P}\left(f(\tilde{\boldsymbol{z}}^*, \tilde{\boldsymbol{\theta}}^*; [N]) - f(\tilde{\boldsymbol{z}}^*, \tilde{\boldsymbol{\theta}}^*; S_K^n) \leq \frac{\epsilon}{2+\sqrt{p_1+p_2}}\right) \\
&+ \mathbb{P}\left(\eta^*(\boldsymbol{z}^*, \boldsymbol{\theta}^*) - f(\boldsymbol{z}^*, \boldsymbol{\theta}^*; N) \leq \frac{(1+\sqrt{p_1+p_2})\epsilon}{2+\sqrt{p_1+p_2}}\right) - 1.
\end{aligned}$$

Using Assumption 5, we can evaluate the first expression:

$$\begin{aligned}
&\geq 1 - \exp\left(\frac{-n\epsilon^2}{(2+\sqrt{p_1+p_2})^2 M^2}\right) \\
&+ \mathbb{P}\left(\eta^*(\boldsymbol{z}^*, \boldsymbol{\theta}^*) - f(\boldsymbol{z}^*, \boldsymbol{\theta}^*; N) \leq \frac{(1+\sqrt{p_1+p_2})\epsilon}{2+\sqrt{p_1+p_2}}\right) - 1.
\end{aligned}$$

Substituting Lemma 27 gives us the formula for the second expression:

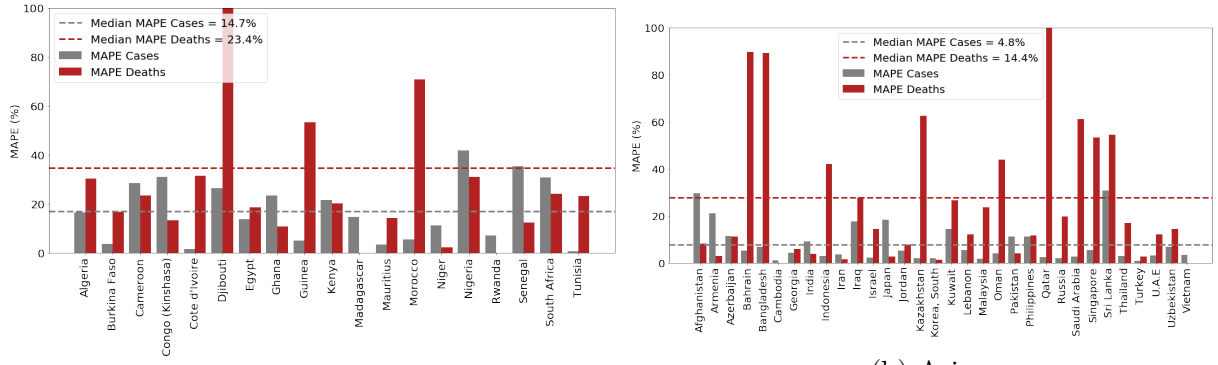
$$\begin{aligned}
&\geq 1 - \exp\left(\frac{-\epsilon^2}{(2+\sqrt{p_1+p_2})^2(1/n-1/N)M}\right) \\
&+ 1 - 2K\exp\left(\frac{-\epsilon^2}{(2+\sqrt{p_1+p_2})^2(1/n-1/N)\max(M,M')}\right) - 1 \\
&\geq 1 - (2K+1)\exp\left(\frac{-\epsilon^2}{(2+\sqrt{p_1+p_2})^2(1/n-1/N)\max(M,M')}\right),
\end{aligned}$$

where J is an absolute constant, as required. □

Appendix D

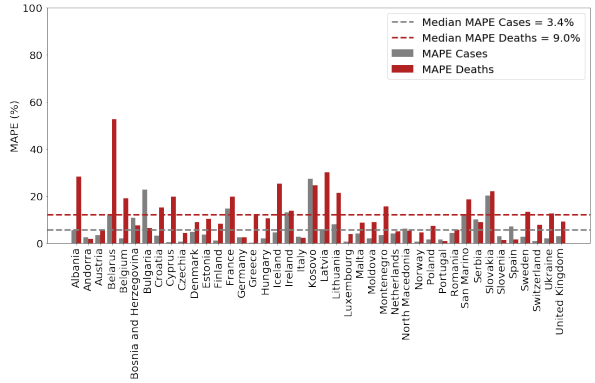
Appendix to Chapter 5

D.1 Detailed Forecasting Results for the World

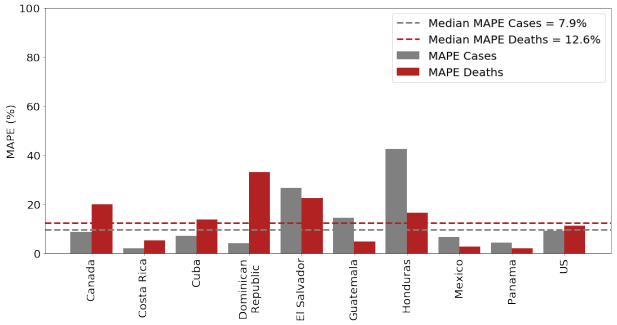


(a) Africa

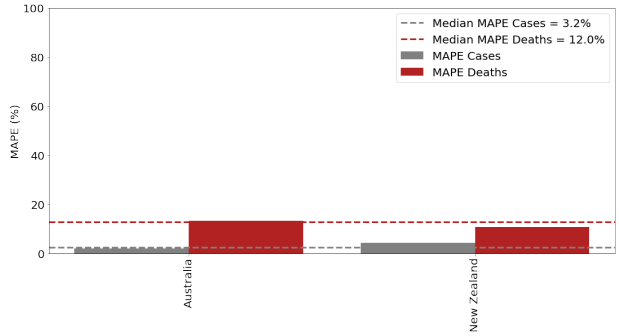
(b) Asia



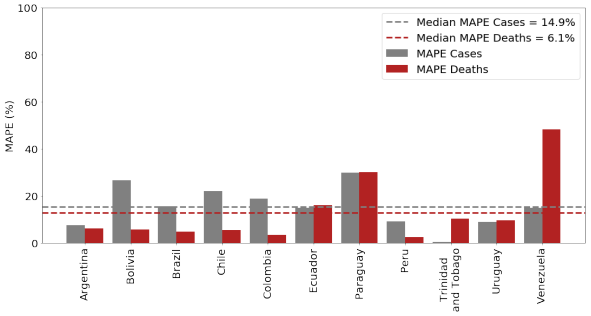
(c) Europe



(d) North America



(e) Oceania



(f) South America

Figure D-1: Mean Absolute Percentage Error (MAPE) of the predicted number of cases and deceases in each country (projections made using data up to 04/27 for the period from 04/28 to 05/12).

D.2 Correspondence between Policy Data and Categorized Policies

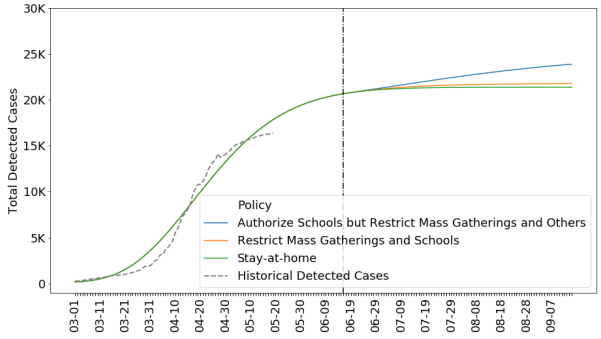
Oxford Policy	Categorized Policy
C3: Cancel Public Events C4: Restrictions on gatherings C5: Close public transport	Restrict Mass Gatherings
C2: Workplace closing C7: Restrictions on internal movement C8: International travel controls	Travel and Work Restrictions
C1: School closing	School Restrictions
C6: Stay at home requirements	Stay-at-Home Order

Table D.1: Correspondence between Oxford Policy Data and Categorized Policies.

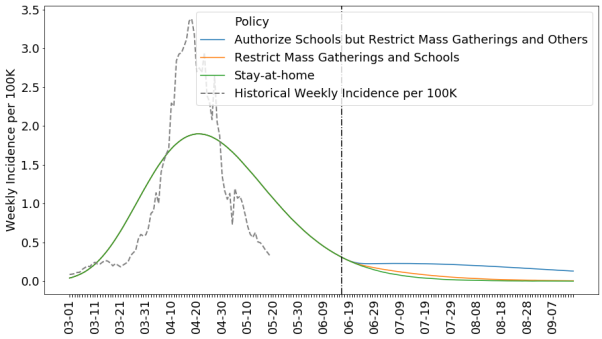
IHME Policy	Categorized Policy
Mass Gathering Restrictions	Restrict Mass Gatherings
Business Closure Non-essential Businesses Closed Travel Severely Limited	Travel and Work Restrictions
Educational Facilities Closed	School Restrictions
Stay-at-Home Order	Stay-at-Home Order

Table D.2: Correspondence between IHME Policy Data and Categorized Policies.

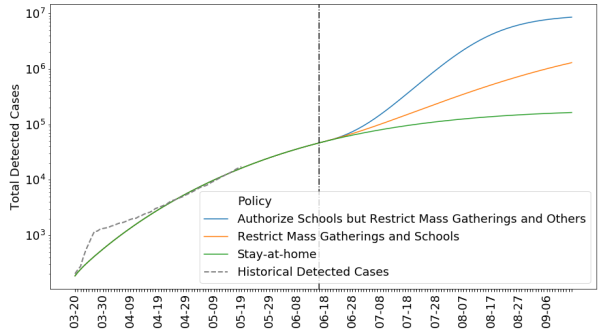
D.3 Additional Results for Reopening Strategies



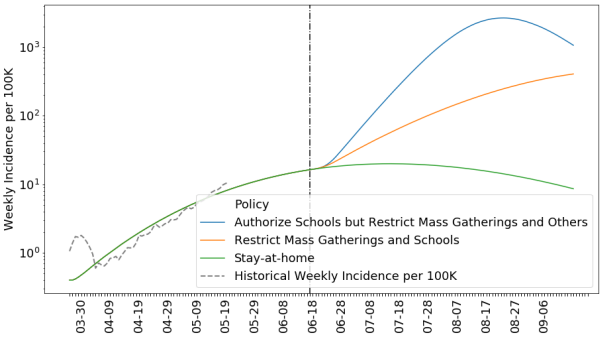
(a) Japan, total detected cases.



(b) Japan, weekly incidence per 100K.



(c) South Africa, total detected cases (log scale).



(d) South Africa, weekly incidence per 100K (log scale).

Figure D-2: Forecasts of total detected cases and weekly incidence per 100K for Japan and South Africa under various policies.

Appendix E

Appendix to Chapter 6

Table E.1 presents the proposed center allocation; for each state, we show the number of centers allocated, the number of vaccines per day, and the cities where the proposed centers are located.

State	**# centers**	**# vaccines/day**	**Selected cities**
Alabama	2	14458	Birmingham, Mobile
Alaska	1	6592	Anchorage
Arizona	2	29663	Mesa, Scottsdale
Arkansas	1	14094	Little Rock
California	8	52734	Bakersfield, Chula Vista, Citrus Heights, Fremont, Fresno Glendale, Indio, Irvine
Colorado	2	28289	Centennial, Colorado Springs
Connecticut	1	8699	Waterbury
Delaware	1	6592	Wilmington
District of Columbia	1	6592	Washington
Florida	6	88989	Boca Raton, Jacksonville, Miami, Orlando, Tallahassee, Tampa
Georgia	2	13357	Sandy Springs, Warner Robins
Hawaii	1	6592	Honolulu

Idaho	1	6592	Meridian
Illinois	2	13184	Cicero, Springfield
Indiana	2	20807	Indianapolis, South Bend
Iowa	2	16356	Cedar Rapids, Des Moines
Kansas	1	6592	Topeka
Kentucky	1	6592	Louisville
Louisiana	2	29386	Baton Rouge, Shreveport
Maine	1	6592	Portland
Maryland	1	12167	Baltimore
Massachusetts	1	11756	Lowell
Michigan	2	13184	Farmington Hills, Wyoming
Minnesota	1	6592	Brooklyn Park
Mississippi	1	8991	Jackson
Missouri	2	19139	Lee's Summit, O'Fallon
Montana	1	6592	Missoula
Nebraska	1	6592	Omaha
Nevada	2	13184	Henderson, Sparks
New Hampshire	1	6592	Nashua
New Jersey	2	29663	Camden, Newark
New Mexico	1	11670	Rio Rancho
New York	5	70791	Buffalo, Rochester, Schenectady, Syracuse, Yonkers
North Carolina	2	13184	Cary, Charlotte
North Dakota	1	6592	Fargo
Ohio	3	19775	Cincinnati, Cleveland, Columbus
Oklahoma	2	20835	Broken Arrow, Edmond
Oregon	1	13395	Portland
Pennsylvania	2	13184	Pittsburgh, Reading
Rhode Island	1	6592	Pawtucket

South Carolina	1	10979	Columbia
South Dakota	1	6592	Sioux Falls
Tennessee	3	19775	Knoxville, Memphis, Nashville
Texas	11	163146	Austin, Corpus Christi, Dallas, El Paso, Fort Worth, Houston, Lubbock, McAllen, Midland, San Antonio, Tyler
Utah	1	10818	West Valley City
Vermont	1	6804	Burlington
Virginia	3	22841	Alexandria, Newport News, Roanoke
Washington	2	26447	Kent, Spokane
West Virginia	1	8370	Charleston
Wisconsin	2	13184	Appleton, Waukesha
Wyoming	1	6592	Cheyenne

Table E.1: Proposed center allocation

In Table E.2, we outline the realized center allocations which has so far been implemented by the Federal Emergency Management Agency (FEMA) [66, 65]. In Table E.3, we compare the realized center allocation (R) with the proposed center allocation (Pr). For the latter, we calculate percentages by taking into account only the 48 centers that were allocated to the 13 states under consideration. We also show the population-based center allocation (Pop), for which we calculate percentages by taking into account the 54 centers that were allocated to the 13 states under consideration. The last column in Table E.3 indicates whether (R) and (Pr) agree, that is, whether the fraction of centers allocated by both to any given state is greater/equal/less than the population share of the state.

Similar to Table E.3, Table E.4 compares the number of vaccines per day (v/d, in thousands) allocated to each state in the realized solution (R) with that in the proposed solution (Pr) and the population-based baseline solution (Pop). For (Pop), we take the (Pop) center allocation shown in Table E.3 and assume that the vaccines are distributed uniformly

across centers. More specifically, the number of vaccines that each center gets is equal to the average number of vaccines allocated to each center in (R), namely, in the realized solution.

Both tables show that, despite obvious differences given the rollout plan and a number of practical considerations, the model-based recommendations are generally in line with the center and vaccine allocation decisions that were made in practice. Specifically, the realized and proposed solutions agree in the majority of cases, both in terms of centers and in terms of vaccines. These results underscore the role of our model as a strategic tool to support the deployment of vaccination sites as well as a tactical tool to support vaccine distribution.

State	**# centers**	**# vaccines/day**	**Selected cities**
California	2	12000	Oakland, Los Angeles
Florida	4	12000	Miami, Tampa, Orlando, Jacksonville
Illinois	1	6000	Chicago
Indiana	1	3000	Gary
Massachusetts	1	5000	Boston
Missouri	1	3000	St. Louis
New Jersey	1	5000	Newark
New York	6	12000	Brooklyn, Queens, Yonkers, Rochester, Buffalo, Albany
North Carolina	1	3000	Greensboro
Pennsylvania	1	6000	Philadelphia,
Texas	3	12000	Houston, Dallas, Arlington
Virginia	1	5000	Norfolk
Washington	1	1000	Yakima

Table E.2: Realized center allocation

	Realized		Proposed		Population		
State	#	%	#	%	#	%	**(R) $\iff$ (Pr)**
California	2	8.0%	8	16.6%	12	22.2%	1
Florida	4	17.0	6	12.5	6	11.1	1
Illinois	1	4.2	2	4.2	4	7.4	1
Indiana	1	4.2	2	4.2	2	3.7	1
Massachusetts	1	4.2	1	2.1	2	3.7	0
Missouri	1	4.2	2	4.2	1	1.9	1
New Jersey	1	4.2	2	4.2	2	3.7	1
New York	6	25.0	5	10.4	6	11.1	0
North Carolina	1	4.2	2	4.2	3	5.6	1
Pennsylvania	1	4.2	2	4.2	4	7.4	1
Texas	3	12.5	11	22.9	8	14.8	0
Virginia	1	4.2	3	6.2	2	3.7	0
Washington	4.2	4.2	2	4.2	2	3.7	1
Total	24	100.0%	48	100.0%	54	100.0%	9

Table E.3: Comparison between realized and proposed center allocations

	Realized		Proposed		Population		
State	#	%	#	%	#	%	**(R) $\iff$ (Pr)**
California	12	0.14	53	0.1	42.5	0.22	1
Florida	12	0.14	89	0.16	21.25	0.11	1
Illinois	6	0.07	13	0.02	14.167	0.07	0
Indiana	3	0.04	21	0.04	7.083	0.04	1
Massachusetts	5	0.06	12	0.02	7.083	0.04	0
Missouri	3	0.04	19	0.03	3.542	0.02	1
New Jersey	5	0.06	30	0.05	7.083	0.04	1
New York	12	0.14	71	0.13	21.25	0.11	1
North Carolina	3	0.04	13	0.02	10.625	0.06	1
Pennsylvania	6	0.07	13	0.02	14.167	0.07	0
Texas	12	0.14	163	0.3	28.333	0.15	0
Virginia	5	0.06	23	0.04	7.083	0.04	0
Washington	1	0.01	26	0.05	7.083	0.04	0
Total	85	1.01	546	0.98	191.249	1.01	7

Table E.4: Comparison between realized and proposed vaccine allocations