Distributionally Robust Project Crashing with Partial or No Correlation Information

Selin Damla Ahipasaoglu*  Karthik Natarajan†  Dongjian Shi‡

March 19, 2018

Abstract

Crashing is shortening the project makespan by reducing activity times in a project network by allocating resources to them. Activity durations are often uncertain and exact probability distribution itself might be ambiguous. We study a class of distributionally robust project crashing problems where the objective is to optimize the first two marginal moments (means and standard deviations) of the activity durations to minimize the worst-case expected makespan. Under partial correlation information and no correlation information, the problem is solvable in polynomial time as a semidefinite program and a second order cone program, respectively. However solving semidefinite programs is challenging for large project networks. We exploit the structure of the distributionally robust formulation to reformulate a convex-concave saddle point problem over the first two marginal moment variables and the arc criticality index variables. We then use a projection and contraction algorithm for monotone variational inequalities in conjunction with a gradient method to solve the saddle point problem enabling us to tackle large instances. Numerical results indicate that a manager who is faced with ambiguity in the distribution of activity durations has a greater incentive to invest resources in decreasing the variation rather than the mean of the activity durations.

1 Introduction

Projects are ubiquitous, be it in the construction industry or the software development industry. Formally, a project is defined by a set of activities that must be completed with given precedence constraints. In a project, an activity is a task that must be performed and an event is a milestone marking the start of one or more activities. Before an activity begins, all of its predecessor activities must be completed. Such a project is represented

*Engineering Systems and Design, Singapore University of Technology and Design, 8 Somapah Road, Singapore 487372. Email: ahipasaoglu@sutd.edu.sg
†Engineering Systems and Design, Singapore University of Technology and Design, 8 Somapah Road, Singapore 487372. Email: karthik.natarajan@sutd.edu.sg
‡Engineering Systems and Design, Singapore University of Technology and Design, 8 Somapah Road, Singapore 487372. Email: dongjian.shi@163.com
by an activity-on-arc network $G(V, A)$, where $V = \{1, 2, \ldots, n\}$ is the set of nodes denoting the events, and $A \subseteq \{(i, j) : i, j \in V\}$ is the set of arcs denoting the activities (see (19)). We let $m = |A|$ denote the number of arcs in the network. The network $G(V, A)$ is directed and acyclic where we use node 1 and node $n$ to represent the start and the end of the project respectively. Let $t_{ij}$ denote the duration of activity $(i, j)$. The completion time or the makespan of the project is equal to the length of the critical or the longest path of the project network from node 1 to node $n$ where arc lengths denote activity durations. This problem is formulated as:

$$Z(t) = \max_{x \in \mathcal{X}} \sum_{(i,j) \in A} t_{ij}x_{ij},$$

(1.1)

where

$$\mathcal{X} = \left\{ x : \sum_{j:(i,j) \in A} x_{ij} - \sum_{j:(j,i) \in A} x_{ji} = \begin{cases} 1, & i = 1 \\ 0, & i = 2, 3, \ldots, n - 1 \\ -1, & i = n \end{cases} \right\} \cap \{0, 1\}.$$  \hspace{1cm} (1.2)

Project crashing is a method for shortening the project makespan by reducing the time of one or more of the project activities to less than its normal activity time. To reduce the duration of an activity, the project manager might assign more resources to it which implies additional costs. Such a resource allocation may include using more efficient equipment or hiring more workers. In such a situation it is important to model the tradeoff between the makespan and the crashing cost so as to identify the specific activities to crash and the corresponding amounts by which to crash them. Early work on the deterministic project crashing problem (PCP) dates back to 1960s (see (33, 23)) where parametric network flow methods were developed to solve the problem. One intuitive method is to find the critical path of the project, and then crash one or more activities on the critical path. However this approach is known to be sub-optimal in general, since the original critical path may no longer be critical after a while. The problem of minimizing the project makespan with a given cost budget is formulated as the following optimization problem (see (33, 23)):

$$\min_t Z(t) \text{ s.t. } \sum_{(i,j) \in A} c_{ij}(t_{ij}) \leq B, \sum_{(i,j) \in A} t_{ij}x_{ij} \leq B, \sum_{(i,j) \in A} c_{ij}(t_{ij}) \leq t_{ij}, \forall (i, j) \in A,$$

(1.3)

where $B$ is the cost budget, $\bar{t}_{ij}$ is the original duration of activity $(i, j)$, $\bar{t}_{ij}$ is the minimal value of the duration of activity $(i, j)$ that can be achieved by crashing, and $c_{ij}(t_{ij})$ is the cost of crashing which is a decreasing function of the activity duration $t_{ij}$. Since the longest path problem on a directed acyclic graph is solvable as a linear program by optimizing over the set $\mathcal{X}$ directly, we can use linear programming duality to reformulate the project
crashing problem (1.3) as:
\[
\begin{align*}
\min_{t, y} \quad & y_n - y_1, \\
\text{s.t.} \quad & y_j - y_i \geq t_{ij}, \quad \forall (i, j) \in \mathcal{A}, \\
& \sum_{(i, j) \in \mathcal{A}} c_{ij}(t_{ij}) \leq B, \\
& L_{ij} \leq t_{ij} \leq U_{ij}, \quad \forall (i, j) \in \mathcal{A}.
\end{align*}
\] (1.4)

When the cost functions are linear or piecewise linear convex, the deterministic PCP (1.4) can be solved as a linear program. For nonlinear convex differentiable cost functions, an algorithm based on piecewise linear approximations was proposed by (37) to solve the project crashing problem. When the cost function is nonlinear and concave, (20) proposed a globally convergent branch and bound algorithm to solve the problem of minimizing the total cost and when the cost function is discrete, the project crashing problem has shown to be NP-hard (see (16)). In the next section, we provide a literature review on project crashing problems with uncertain activity durations.

Notations

Throughout the paper, we use bold letters to denote vectors and matrices (such as \( \mathbf{x}, \mathbf{W}, \mathbf{\rho} \)), and standard letters to denote scalars (such as \( x, W, \rho \)). A vector of dimension \( n \) with all entries equal to 1 is denoted by \( \mathbf{1}_n \). The unit simplex is denoted by \( \Delta_{n-1} = \{ \mathbf{x} : \mathbf{1}_n^T \mathbf{x} = 1, \mathbf{x} \geq 0 \} \). We suppress the subscript when the dimension is clear. We use the tilde notation to denote a random variable or random vector (such as \( \tilde{r}, \tilde{r} \)). The number of elements in the set \( \mathcal{A} \) is denoted by \( |\mathcal{A}| \). The set of nonnegative integers is denoted by \( \mathbb{Z}^+ \) and the sets of \( n \) dimensional vectors whose entries are all nonnegative and strictly positive are denoted by \( \mathbb{R}_n^+ \) and \( \mathbb{R}_n^{++} \) respectively. The sets of all symmetric real \( n \times n \) positive semidefinite matrices and positive definite matrices are denoted by \( \mathbb{S}_n^+ \) and \( \mathbb{S}_n^{++} \) respectively. For a positive semidefinite matrix \( \mathbf{X} \), we use \( \mathbf{X}^{1/2} \) to denote the unique positive semidefinite square root of the matrix such that \( \mathbf{X}^{1/2} \mathbf{X}^{1/2} = \mathbf{X} \). For a square matrix \( \mathbf{X} \), \( \mathbf{X}^{\dagger} \) denotes the its unique Moore-Penrose pseudoinverse. We use \( \text{diag}(\mathbf{X}) \) to denote a vector formed by the diagonal elements of a matrix \( \mathbf{X} \) and \( \text{Diag}(\mathbf{x}) \) to denote a diagonal matrix whose diagonal elements are the entries of \( \mathbf{x} \).

2 Literature Review

Our literature review in this section focuses on the project crashing problem with uncertain activity durations. Our goal is to highlight some of the main approaches to solve the project crashing problem before discussing the contributions of this paper.

2.1 Stochastic Project Crashing

In stochastic projects, the activity durations are typically modeled as random variables with a specific probability distribution such as the normal, uniform, exponential or beta distribution. When the probability distribution
of the activity durations is known, a popular performance measure is the expected project makespan which is
defined as follows:

$$E_{\theta(\lambda)}(Z(\tilde{t})).$$  \hspace{1cm} (2.1)

In (2.1), the activity duration vector $\tilde{t}$ is random with a probability distribution denoted by $\theta(\lambda)$ where $\lambda$
is assumed to be a finite dimensional vector that parameterizes the distribution and can be modified by the
project manager. For example in the multivariate normal case, $\lambda$ will incorporate information on the mean and
the covariance matrix of the activity durations. The stochastic PCP that minimizes the expected makespan is
formulated as:

$$\min_{\lambda \in \Lambda} E(Z(\tilde{t})),$$ \hspace{1cm} (2.2)

where $\Lambda$ is the possible set of values from which $\lambda$ can be chosen. For a fixed $\lambda$, computing the expected project
makespan unlike the deterministic makespan is a hard problem. (28) showed that computing the expected project
makespan is NP-hard when the activity durations are independent discrete random variables. Even
with simple distributions such as the multivariate normal distribution, the expected makespan does not have a
simple expression and the standard approach is to use Monte Carlo simulation methods to estimate the expected
makespan (see (53, 10)). Bounds ((24, 43)) and approximations ((39)) have been proposed for the expected
makespan. For example, by replacing the activity times with the mean durations and computing the deterministic
longest path, we obtain a lower bound on the expected makespan due to the convexity of the makespan objective.
Equality holds if and only if there is a path that is the longest with probability 1, but this condition is rarely
satisfied in applications.

To solve the stochastic project crashing problem in (2.2), heuristics and simulation-based optimization methods
have been proposed. (35) developed a heuristic approach to minimize the quantile of the makespan by using a
surrogate deterministic objective function. Other heuristics for the project crashing problem have also been
developed by (42). Stochastic gradient methods for minimizing the expected makespan have been developed in
this context (see (9, 22)). Another popular approach is to use the sample average approximation (SAA) method
to minimize the expected makespan (see (47, 50, 34)).

2.2 Robust Project Crashing

There are two main challenges in the stochastic project crashing model. Firstly, the problem is computationally
challenging and NP-hard in most instances. Secondly, the distribution of the activity durations is assumed to
be given but in reality it is often difficult to estimate the full distributional information. To overcome these
challenges, robust optimization methods (see (3)) can be employed as follows. Let the activity duration vector $t$
lie in a compact uncertainty set denoted by $U(\gamma)$ where $\gamma$ is a finite dimensional vector that parameterizes the
uncertainty set and can be modified by the project manager. The worst-case project makespan is then defined as
follows:

$$\max_{t \in U(\gamma)} Z(t).$$ \hspace{1cm} (2.3)
The robust PCP that minimizes the worst-case makespan is formulated as:

$$\min_{\gamma \in \Gamma} \max_{t \in U(\gamma)} Z(t),$$  \hspace{1cm} (2.4)$$

where $\Gamma$ is the set of possible values from which $\gamma$ can be chosen. (55) showed that for a given $\gamma \in \Gamma$, the problem of computing the worst-case makespan is NP-hard. Examples of uncertainty sets for which computing the worst-case makespan is known to be NP-hard include the ellipsoidal uncertainty set and polyhedral uncertainty sets such as the intersection of a hypercube and a halfspace. The problem is easy only for simple uncertainty sets such as the hypercube, since the worst-case makespan is obtained simply by setting all the activity durations to the maximum possible value. (15) adopted the affinely adjustable robust formulation of (4) to tackle the problem of minimizing the worst-case project makespan under interval and ellipsoidal uncertainty sets. The corresponding problems are formulated as linear and second order cone programs. (32) extended the affinely adjustable robust counterpart to develop Pareto robust optimal solutions for project crashing. However as discussed in (55) while computationally tractable, the linear decision rule is sub-optimal for the robust project crashing problem. (12) proposed new uncertainty sets for the robust project crashing problem to capture asymmetry information in the activity durations. Using linear decision rules they developed second order cone programs for the robust project crashing problem. (13) considered more general decision rules beyond the linear decision rule for the crashing problem. While their results clearly demonstrate that the linear and piecewise linear decision rules are computationally tractable, these approaches only solve a relaxation of the robust problem. (55) proposed alternative methods to solve the robust project crashing problem by developing convergent bounds using path enumeration and path generation methods.

### 2.3 Distributionally Robust Project Crashing

Distributionally robust optimization is a more recent approach that has been used to address the project crashing problem. Under this model, the uncertain activity durations are assumed to be random variables but the probability distribution of the random variables is itself ambiguous and lies in a set of distributions. Assume that $\Theta(\omega)$ is a set of probability distributions where $\omega$ is a finite dimensional parameter vector that parameterizes this set and can be modified by the project manager. The performance measure in this case is the worst-case expected project makespan which is defined as follows:

$$\max_{\theta \in \Theta(\omega)} \mathbb{E}_\theta(Z(\hat{t})).$$  \hspace{1cm} (2.5)$$

The distributionally robust PCP that minimizes the expected makespan is then formulated as:

$$\min_{\omega \in \Omega} \max_{\theta \in \Theta(\omega)} \mathbb{E}_\theta(Z(\hat{t})),\hspace{1cm} (2.6)$$

where $\Omega$ is the possible set of values from which $\omega$ can be chosen. (41) studied the problem of computing the worst-case expected makespan under the assumption that the marginal distributions of the random activity durations are known but the joint distribution of the activity durations is unknown. Under this assumption, they
showed that the worst-case expected makespan can be computed by solving a convex optimization problem. (8) extended this bound to the case where the support for each activity duration is known and up to the first two marginal moments (mean and standard deviation) of the random activity duration is provided. (6, 7) extended this result to the case where general higher order univariate moment information is known and developed a semidefinite program to compute the worst-case expected makespan. (40) applied the dual of the formulation to solve an appointment scheduling problem where the objective is to choose service times to minimize the worst-case expected waiting time and overtime costs as a second order cone program. Under the assumption that the mean, standard deviation and correlation matrix of the activity durations is known, (46) developed a completely positive programming reformulation for the worst-case expected makespan. While this problem is NP-hard, semidefinite relaxations can be used to find weaker upper bounds on the worst-case expected makespan. (36) developed a dual copositive formulation for the appointment scheduling problem where the objective is to choose service times to minimize the worst-case expected waiting time and overtime costs given correlation information. Since this problem is NP-hard, they developed a tractable semidefinite relaxation for this problem. (45) recently showed that the complexity of computing such bounds is closely related to the complexity of characterizing the convex hull of quadratic forms of the underlying feasible region. In a related stream of literature, (25) developed approximations for the distributionally robust project crashing problem using information on the support, mean and correlation matrix of the activity durations. Using linear and piecewise linear decision rules, they developed computationally tractable second order conic programming formulations to find resource allocations to minimize an upper bound on the worst-case expected makespan under both static and adaptive policies. While their numerical results demonstrate the promise of the distributionally robust approach, it is not clear as to how far their solution is from the true optimal solution. Recently, (29) studied a distributionally robust chance constrained version of the project crashing problem and developed a conic program to solve the problem under the assumption of the knowledge of a conic support, the mean and an upper bound on a positive homogeneous dispersion measure of the random activity durations. While their formulation is exact for the distributionally robust chance constrained project crashing problem, the size of the formulation grows in the number of paths in the network. An example where the worst-case expected makespan is computable in polynomial time was developed in (17) who assumed that a discrete distribution is provided for the activity durations for the set of arcs coming out of each node. The dependency structure among the activities for arcs coming out of two different nodes is however unspecified. (38) extended this result to propose a bound on the worst-case expected makespan with information on the mean and covariance matrix.

In this paper, we build on these models to solve a class of distributionally robust project crashing problems in polynomial time. Furthermore, unlike the typical use of semidefinite programming solvers to directly solve the problem, we exploit the structure of the objective function to illustrate that the problem can be reformulated as a convex-concave saddle point problem over the first two marginal moment variables and the arc criticality index variables. This simplification provides the opportunity to make use of the saddle point formulation directly to solve the distributionally robust project crashing problem. To the best of our knowledge, this is one of the few
attempts to solve the distributionally robust project crashing problem using the saddle point formulation directly. We use a projection and contraction algorithm for monotone variational inequalities in conjunction with a gradient method to solve the saddle point problem. As we demonstrate, this helps us solve larger problems than those currently possible with the semidefinite formulations. Lastly, we provide insights into the nature of the crashing solution from distributionally robust models that we believe are useful. Our results show that in comparison to the sample average approximation method for a multivariate normal distribution of activity durations, the distributionally robust models deploy more resources in crashing the standard deviations rather than the means. This implies that the project manager who is facing ambiguity in activity durations has more incentive to invest resources in reducing the variation rather than the mean of the activity durations in comparison to a project manager who does not face any ambiguity.

3 SOCP, SDP and Saddle Point Formulations

In this section, we propose a model of the distributionally robust project crashing problem with moment information and identify instances where the problem is solvable in polynomial time in the size of the network.

3.1 Model

Let $\mu = (\mu_{ij} : (i, j) \in \mathcal{A})$ and $\sigma = (\sigma_{ij} : (i, j) \in \mathcal{A})$ denote the vector of means and standard deviations of the activity durations with $\rho$ denoting additional (possibly limited) information on the correlation matrix that is available. The random duration of the activity $(i, j)$ denoted by $\tilde{t}_{ij}$ is expressed as:

$$\tilde{t}_{ij} = \mu_{ij} + \sigma_{ij} \tilde{\xi}_{ij}, \quad \forall (i, j) \in \mathcal{A},$$

where the random variables $\tilde{\xi}_{ij}$ have zero mean and unit variance for all activities with the correlation information among these random variables captured in $\rho$. We allow for the activity durations to be correlated. This often arises in projects when activities are affected by common factors such as weather in a construction project (see (2)). A typical approach to decrease the average time of an activity duration is by allocating more workers to do the job. Several factors are known to affect the activity duration variability in projects in the construction industry. In particular, (54) identified the top nine causes for activity duration variation from over fifty causes by surveying laborers, foremen and project managers from civilian construction contractors that work with government agencies in the United States. The top nine causes that were identified in this survey which affect the variability in the activity durations were the following: (1) waiting to get answers to questions about a design or drawing, (2) turnaround time from engineers when there are questions about a drawing, (3) completion of previous activities, (4) socializing with fellow workers, (5) weather impacts such as excessive heat or rain, (6) rework that is needed due to the quality of previous work, (7) lack of skills and experience of the worker to perform a task, (8) people arriving late due to illness or personal reasons and (9) needing guidance and instructions from a supervisor. While some of these causes of variation are controllable by allocating resources to standardize the process and workflow,
some of the variability in the activity durations due to factors such as weather are simply unavoidable. In this paper, we assume that it is possible for the project manager to modify the mean and the standard deviation of the individual activity durations \( \tilde{t}_{ij} \) by allocating resources directly to the activity (see Figure 1).

![Density function of original activity duration](image)

**Figure 1:** Reducing the mean and the standard deviation of the activity duration

These resources might jointly affect the mean and the correlation or might affect only one of the two. When the joint distribution of \( \tilde{t} \) is known only to lie in a set of distributions \( \Theta_\rho(\mu, \sigma) \) with the given mean, standard deviation and correlation information, the worst-case expected makespan is defined as:

\[
\max_{\theta \in \Theta_\rho(\mu, \sigma)} \mathbb{E}_\theta \left( \max_{x \in X \cap \{0,1\}^m} \tilde{t}^T x \right),
\]

where the outer maximization is over the set of distributions with the given moment information on the random \( \tilde{t} \) and the inner maximization is over the intersection of the set \( X \) defined in (1.2) and \( \{0,1\}^m \). When the correlation matrix \( \rho \) is completely specified, computing just the worst-case expected makespan for a given \( \mu \) and \( \sigma \) is known to be a hard problem (see (5, 46, 55)). The distributionally robust project crashing problem to minimize the worst-case expected makespan is formulated as:

\[
\min_{(\mu, \sigma) \in \Omega} \max_{\theta \in \Theta_\rho(\mu, \sigma)} \mathbb{E}_\theta \left( \max_{x \in X \cap \{0,1\}^m} \tilde{t}^T x \right),
\]

where \( \Omega \) defines a convex set of feasible allocations for \( \omega = (\mu, \sigma) \). A simple example of the set \( \Omega \) is as follows:

\[
\Omega = \left\{ (\mu, \sigma) : \sum_{(i,j) \in A} c_{ij}^{(1)}(\mu_{ij}) + c_{ij}^{(2)}(\sigma_{ij}) \leq B, \ \mu_{ij} \leq \mu_{ij} \leq \bar{\sigma}_{ij}, \ \bar{\sigma}_{ij} \leq \sigma_{ij} \leq \bar{\sigma}_{ij}, \ \forall (i,j) \in A \right\},
\]

where \( \mu_{ij} \) and \( \sigma_{ij} \) are the mean and standard deviation of the original random duration of activity \((i, j)\), and \( \mu_{ij} \) and \( \sigma_{ij} \) are the minimal mean and standard deviation of the duration of activity \((i, j)\) that can be achieved by allocating resources to it. Further \( B \) is the amount of total cost budget, and \( c_{ij}^{(1)}(\mu_{ij}) \) and \( c_{ij}^{(2)}(\sigma_{ij}) \) are the cost functions which have the following properties: (a) \( c_{ij}^{(1)}(\bar{\sigma}_{ij}) = c_{ij}^{(2)}(\bar{\sigma}_{ij}) = 0 \) which means the extra cost of activity...
(i, j) is 0 under the original mean duration and standard deviation; and (b) $e_{ij}^{(1)}(\mu_{ij})$ is a decreasing function of $\mu_{ij}$ and likewise $e_{ij}^{(2)}(\sigma_{ij})$ is a decreasing function of $\sigma_{ij}$. Note that it is possible in this formulation to just crash the means by forcing the standard deviation to be fixed by setting $\sigma_{ij} = \sigma_{ij}$ in the outer optimization problem.

Another simple example of a set to which our results can be directly applied is:

$$\Omega = \{(\mu, \sigma) : \mu = a + Ay, \sigma = d + Dy, y \in Y\},$$  \hspace{1cm} (3.5)

where the mean vector $\mu$ and the standard deviation vector $\sigma$ are defined through affine transformations of a resource vector $y$ that is assumed to lie in a convex set $Y$. In this case through appropriate specification of the matrices $A$ and $D$ it is possible to allow the resource to jointly affect both the mean and the standard deviation.

### 3.2 No Correlation Information

In this section, we consider the marginal moment model (MMM) where information on the mean and the standard deviation of the activity durations is assumed but no information on the correlations is assumed. The set of probability distributions of the activity durations with the given first moments is defined as:

$$\Theta_{mmm}(\mu, \sigma) = \{\Theta \in M(\mathbb{R}_m) : E_\Theta(\tilde{t}_{ij}) = \mu_{ij}, E_\Theta(\tilde{t}_{ij}^2) = \mu_{ij}^2 + \sigma_{ij}^2, \forall (i, j) \in A\},$$  \hspace{1cm} (3.6)

where $M(\mathbb{R}_m)$ is the set of finite positive Borel measures supported on $\mathbb{R}_m$. In the definition of this set, we allow for the activity durations to be positively correlated, negatively correlated or even possibly uncorrelated. Furthermore, we do not make an explicit assumption on the nonnegativity of activity durations. There are two main reasons for this. Firstly, since we allow for the possibility of any valid correlation matrix with the given means and standard deviations, the most easy to fit multivariate probability distribution to the activity durations is the normal distribution. As a result, this distribution has been used extensively in the literature on project networks (see (14, 2)), particularly when the activity durations are correlated for this very reason. Secondly in practice, such an assumption is reasonable to justify when the mean of the activity duration is comparatively larger than the standard deviation in which case the probability of having a negative realization is small. Assuming no correlation information, the worst-case expected makespan in (3.2) is equivalent to the optimal objective value of the following concave maximization problem over the convex hull of the set $\mathcal{X} \cap \{0, 1\}^m$ which is exactly $\mathcal{X}$ (see Lemma 2 on page 458 in (44)):

$$\max_{x \in \mathcal{X}} f_{mmm}(\mu, \sigma, x),$$  \hspace{1cm} (3.7)

where

$$f_{mmm}(\mu, \sigma, x) = \sum_{(i, j) \in A} \left( \mu_{ij} x_{ij} + \sigma_{ij} \sqrt{x_{ij}(1 - x_{ij})} \right).$$  \hspace{1cm} (3.8)

In the formulation, the optimal $x_{ij}^*$ variables is an estimate of the arc criticality index of activity $(i, j)$ under the worst-case distribution. The worst-case expected makespan in (3.7) is computed using the following second order
cone program (SOCP):
\[
\max_{x, t} \sum_{(i,j) \in A} (\mu_{ij} x_{ij} + \sigma_{ij} t_{ij}) \\
\text{s.t. } x \in \mathcal{X}, \\
\sqrt{t_{ij}^2 + \left( x_{ij} - \frac{1}{2} \right)^2} \leq \frac{1}{2}, \forall (i,j) \in \mathcal{A}.
\]
(3.9)

The distributionally robust PCP (3.3) under the marginal moment model is then formulated as a saddle point over the moment variables and arc criticality index variables as follows:

\[
\min_{(\mu, \sigma) \in \Omega} \max_{x \in \mathcal{X}} f_{\text{mmm}}(\mu, \sigma, x).
\]
(3.10)

One approach to solve the problem is to take the dual of the maximization problem in (3.9) in which case the distributionally robust PCP (3.10) is formulated as the following second order cone program:

\[
\min_{\mu, \sigma, y, \alpha, \beta} y_n - y_1 + \frac{1}{2} \sum_{(i,j) \in \mathcal{A}} (\alpha_{ij} - \beta_{ij}) \\
\text{s.t. } y_j - y_i - \beta_{ij} \geq \mu_{ij}, \forall (i,j) \in \mathcal{A}, \\
\sqrt{\sigma_{ij}^2 + \beta_{ij}^2} \leq \alpha_{ij}, \forall (i,j) \in \mathcal{A} \\
(\mu, \sigma) \in \Omega.
\]
(3.11)

Several points regarding the formulation in (3.11) are important to take note of. Firstly, the formulation is tight, namely it is an exact reformulation of the distributionally robust project crashing problem. Secondly, such a dual formulation has been recently applied by (40) to the appointment scheduling problem where the appointment times are chosen for patients (activities) while the actual service times for the patients are random. Their problem is equivalent to simply crashing the means of the activity durations. From formulation (3.11), we see that it is also possible to crash the standard deviation of the activity durations in a tractable manner using second order cone programming with the marginal moment model. Lastly, though we focus on the project crashing problem, one of the nice features of this model is that it easily extends to all sets \( \mathcal{X} \subseteq \{0, 1\}^n \) with a compact convex hull representation. In the next section, we discuss a partial correlation information structure that makes use of the project network structure in identifying a polynomial time solvable project crashing instance with semidefinite programming.

### 3.3 Partial Correlation Information

In the marginal moment model, we do not make any assumptions on the correlation information between the activity durations. Hence it is possible that in the worst-case, the correlations might be unrealistic particularly if some information on the dependence between activity durations is available. In this section, we consider alternative formulations where partial correlation information on the activity durations is known. Since the general version of this problem is hard, we focus on partial correlation information structures where the problem is solvable in polynomial time. Towards this, we first consider a parallel network, before considering a general network.
3.3.1 Parallel Network

Consider a project network with \( k \) parallel paths (see Figure 2). In this network, the total number of nodes \( n = k + 2 \) and the number of arcs \( m = 2k \). The activity durations of arcs \((1,j)\) for \( j = 2, \ldots, k + 1 \) are random while the activity durations of arcs \((j,k+2)\) for \( j = 2, \ldots, k + 1 \) are deterministic with duration 0. We assume that the correlation among the random activity durations is known. In this case, without loss of generality, we can restrict our attention to the set of probability distributions of the activity durations for the arcs \((1,j)\) for \( j = 2, \ldots, k + 1 \) which is defined as:

\[
\Theta_{\text{cmm}, \rho}(\mu, \sigma) = \left\{ \theta \in M(\mathbb{R}_k) : E_{\theta} (\hat{t}) = \mu, E_{\theta}(\hat{t} \hat{t}^T) = \mu \mu^T + \text{Diag}(\sigma) \rho \text{Diag}(\sigma) \right\},
\]

where \( \rho \in S_k^{++} \) denotes the correlation matrix among the \( k \) random activity durations. The covariance matrix is defined as:

\[
\Sigma = \text{Diag}(\sigma) \rho \text{Diag}(\sigma).
\]

We refer to this model as the cross moment model (CMM). The distributionally robust project crashing problem for a parallel network is then formulated as:

\[
\min_{(\mu, \sigma) \in \Omega} \max_{\theta \in \Theta_{\text{cmm}, \rho}(\mu, \sigma)} E_{\theta} \left( \max_{x \in \Delta_{k-1}} (\hat{t} + \text{Diag}(\sigma) \xi)^T x \right),
\]

where the inner maximization is over the simplex since the network is parallel. The inner worst-case expected makespan problem in (3.14) is equivalent to the moment problem over the probability measure of the random vector \( \hat{\xi} = \text{Diag}(\sigma)^{-1}(\hat{t} - \mu) \) denoted by \( \gamma \) as follows:

\[
\max \ E_{\gamma} \left( \max_{x \in \Delta_{k-1}} \left( \mu + \text{Diag}(\sigma) \hat{\xi} \right)^T x \right)
\]

s.t. \( P_{\gamma}(\hat{\xi} \in \mathbb{R}_k) = 1, \)
\( E_{\gamma}(\hat{\xi}) = 0, \)
\( E_{\gamma}(\hat{\xi} \hat{\xi}^T) = \rho. \)
A direct application of standard moment duality for this problem by associating the dual variables \( \lambda_0, \lambda \) and \( \Lambda \) with the constraints and disaggregating the maximum over the extreme points of the simplex implies that problem (3.15) can be solved as a semidefinite program:

\[
\begin{align*}
\min_{\lambda_0, \lambda, \Lambda} & \quad \lambda_0 + \langle \rho, \Lambda \rangle \\
\text{s.t.} & \quad \begin{pmatrix} \lambda_0 - \mu_{1j} & \frac{1}{2}(\lambda - \sigma_{1j}e_j)^T \\
\frac{1}{2}(\lambda - \sigma_{1j}e_j) & \Lambda \end{pmatrix} \succeq 0, \quad \forall j = 2, \ldots, k + 1,
\end{align*}
\] (3.16)

where \( e_j \) is a vector of dimension \( k \) with 1 in the entry corresponding to node \( j \) and 0 otherwise. Strong duality holds in this case under the assumption that the correlation matrix is positive definite. Plugging it back into (3.14), we obtain the semidefinite program for distributionally robust project crashing problem over a parallel network as follows:

\[
\begin{align*}
\min_{\mu, \sigma, \lambda_0, \lambda, \Lambda} & \quad \lambda_0 + \langle \rho, \Lambda \rangle \\
\text{s.t.} & \quad \begin{pmatrix} \lambda_0 - \mu_{1j} & \frac{1}{2}(\lambda - \sigma_{1j}e_j)^T \\
\frac{1}{2}(\lambda - \sigma_{1j}e_j) & \Lambda \end{pmatrix} \succeq 0, \quad \forall j = 2, \ldots, k + 1,
\end{align*}
\] (3.17)

\((\mu, \sigma) \in \Omega.\)

We next provide an alternative reformulation of the project crashing problem in the spirit of (3.10) as a convex-concave saddle point problem where the number of variables in the formulation grow linearly in the number of arcs. Unlike the original minimax formulation in (3.14) where the outer maximization problem is over infinite dimensional probability measures, we transform the outer maximization problem to optimization over finite dimensional variables (specifically the arc criticality indices).

**Proposition 1.** Define \( S(x) = \text{Diag}(x) - xx^T \). Under the cross moment model with a parallel network, the distributionally robust PCP (3.14) is solvable as a convex-concave saddle point problem:

\[
\begin{align*}
\min_{(\mu, \sigma) \in \Omega} & \max_{x \in \Delta_{k-1}} f_{\text{cmm}}(\mu, \sigma, x),
\end{align*}
\] (3.18)

where

\[
f_{\text{cmm}}(\mu, \sigma, x) = \mu^T x + \text{trace} \left( \left( \Sigma^{1/2} S(x) \Sigma^{1/2} \right)^{1/2} \right).
\] (3.19)

The objective function is convex with respect to the moments \( \mu \) and \( \Sigma^{1/2} \) (and hence \( \sigma \) for a fixed \( \rho \)) and strictly concave with respect to the criticality index variables \( x \).

**Proof.** The worst-case expected makespan under the cross moment model for a parallel network was studied in (1) (see Theorem 1) who showed that it is equivalent to the optimal objective value of the following nonlinear concave maximization problem over the unit simplex:

\[
\max_{\theta \in \Theta, \mu, \sigma} \mathbb{E}_\theta \left( \max_{x \in \Delta_{k-1}} t^T x \right) = \max_{x \in \Delta_{k-1}} \left( \mu^T x + \text{trace} \left( \left( \Sigma^{1/2} S(x) \Sigma^{1/2} \right)^{1/2} \right) \right),
\] (3.20)

where the optimal \( x^*_{ij} \) variables is an estimate of the arc criticality index of activity \((i, j)\) under the worst-case distribution. This results in the equivalent saddle point formulation (3.18) for the distributionally robust project...
crashing problem under the cross moment model with a parallel network. The function $f_{\text{cmm}}(\mu, \sigma, x)$ has shown to be strongly concave in the $x$ variable (see Theorem 3 in (1)). The function $f_{\text{cmm}}(\mu, \sigma, x)$ is linear and hence convex in the $\mu$ variable. Furthermore this function is convex with respect to $\Sigma^{1/2} \in S_{k}^{++}$. To see this, we apply Theorem 7.2 in (11) which shows that the function $g(A) = \text{trace}(B^T A B)_{1/2}$ is convex in $A \in S_{k}^{++}$ for a fixed $B \in \mathbb{R}_{k \times k}$. Clearly, the function $\text{trace}((\Sigma^{1/2} S(x) \Sigma^{1/2})_{1/2}) = \text{trace}((S(x))_{1/2}^{1/2} S(x) \Sigma_{1/2}^{1/2})$ since for any square matrix $X$, the eigenvalues of $XX^T$ are the same as $X^T X$, which implies $\text{trace}((XX^T)_{1/2}) = \text{trace}((X^T X)_{1/2})$. Setting $A = \Sigma^{1/2}$ and $B = S(x)_{1/2}$, implies that the objective function is convex with respect to $\Sigma^{1/2} \in S_{k}^{++}$. □

### 3.3.2 General Network

To model the partial correlation information, we assume that for the subset of arcs that leave a node, information on the correlation matrix is available. Let $A_i$ denote the set of arcs originating from node $i$ for $i = 1, \ldots, n - 1$. Note that the sets $A_i$ are non-overlapping. Then, the set of arcs $A = \bigcup_{i=1}^{n-1} A_i$. We let $\tilde{t}_i$ denote the sub-vector of durations $\tilde{t}_{ij}$ for arcs $(i, j) \in A_i$. In the non-overlapping marginal moment model (NMM), we define the set of distributions of the random vector $\tilde{t}$ as follows:

$$
\Theta_{\text{nmm}, \rho_i, \tilde{v}_i}(\mu, \sigma) = \left\{ \theta \in \mathcal{M} : \mathbb{E}_\theta(\tilde{t}_i) = \mu_i, \mathbb{E}_\theta(\tilde{t}_i^T) = \mu_i \mu_i^T + \text{Diag}(\sigma_i) \rho_i \text{Diag}(\sigma_i), \forall i = 1, \ldots, n - 1 \right\},
$$

(3.21)

where $\mu_i$ denotes the mean vector for $\tilde{t}_i$, $\rho_i \in S_{|A_i|}^{++}$ denotes the correlation matrix of $\tilde{t}_i$ and $\Sigma_i = \text{Diag}(\sigma_i) \rho_i \text{Diag}(\sigma_i)$ denotes the covariance matrix of $\tilde{t}_i$. However, note that in the definition of (3.21), we assume that the correlation between activity durations of the arcs that originate from different nodes is unknown. This is often a reasonable assumption in project networks since the local information of activity durations that originate from a node will typically be better understood by the project manager who might subcontract those activities to a group that is responsible for that part of the project while the global dependency information is much more complicated to model. A typical simplifying assumption is to then let the activity durations be independent for arcs leaving different nodes. The expected project completion time is hard to compute in this case and bounds have been proposed under the assumption of independence (see (24)). On the other hand, in the model discussed in this paper, we allow for these activity durations to be arbitrarily dependent for arcs exiting different nodes and thus take a worst-case perspective. Under partial correlation information, the distributionally robust project crashing problem for a general network is formulated as:

$$
\begin{align*}
\min_{(\mu, \sigma) \in \Omega} \quad & \max_{\theta \in \Theta_{\text{nmm}, \rho_i, \tilde{v}_i}(\mu, \sigma)} \mathbb{E}_\theta \left( \max_{x \in \mathcal{X}} \tilde{t}^T x \right).
\end{align*}
$$

(3.22)

Under the nonoverlapping marginal moment model, the worst-case expected makespan in (3.22) is equivalent to the optimal objective value of the following semidefinite maximization problem over the convex hull of the set $\mathcal{X}$
\[
\max_{x_{ij}, w_{ij}, W_{ij}} \sum_{(i,j) \in A} \left( \mu_{ij} x_{ij} + \sigma_{ij} e_{ij}^T w_{ij} \right)
\]
\[
\text{s.t. } x \in X,
\]
\[
\begin{pmatrix}
1 & 0 \\
0 & \rho_i
\end{pmatrix} - \sum_{(i,j) \in A} \begin{pmatrix} x_{ij} & w_{ij}^T \\
w_{ij} & W_{ij} \end{pmatrix} \geq 0, \quad \forall i = 1, \ldots, n - 1,
\]
\[
\begin{pmatrix} x_{ij} & w_{ij}^T \\
w_{ij} & W_{ij} \end{pmatrix} \geq 0, \quad \forall (i,j) \in A,
\]

where \( e_{ij} \) is a vector of dimension \(|A_i|\) with 1 in the element corresponding to node \( j \) and 0 otherwise. The decision variables here correspond to:

\[
\begin{pmatrix} x_{ij} & w_{ij}^T \\
w_{ij} & W_{ij} \end{pmatrix} = P((i,j) \text{ is critical}) \begin{pmatrix} 1 & \mathbb{E}(\tilde{t}_i | (i,j) \text{ is critical}) \\
\mathbb{E}(\tilde{t}_i | (i,j) \text{ is critical}) & \mathbb{E}(\tilde{t}_i^T | (i,j) \text{ is critical}) \end{pmatrix}. \tag{3.23}
\]

By taking the dual of the problem where strong duality holds under the assumption \( \rho_i \in S_i^{++} \) for all \( i \), the distributionally robust PCP (3.22) is solvable as the semidefinite program:

\[
\min_{\mu, \sigma, y, d, \lambda_0, \Lambda_i} \ y_n - y_1 + \sum_{i=1}^{n-1} (\lambda_{0i} + \langle \rho_i, \Lambda_i \rangle)
\]
\[
\text{s.t. } y_j - y_i \geq d_{ij}, \quad \forall (i,j) \in A,
\]
\[
\begin{pmatrix} \lambda_{0i} + d_{ij} - \mu_{ij} & \frac{1}{2}(\lambda_i - \sigma_{ij} e_{ij})^T \\
\frac{1}{2}(\lambda_i - \sigma_{ij} e_{ij}) & \Lambda_i \end{pmatrix} \succeq 0, \quad \forall (i,j) \in A,
\]
\[
\begin{pmatrix} \lambda_{0i} & \frac{1}{2} \lambda_i^T \\
\frac{1}{2} \lambda_i & \Lambda_i \end{pmatrix} \succeq 0, \quad \forall i = 1, \ldots, n - 1,
\]

\( (\mu, \sigma) \in \Omega \).

We next provide an alternative reformulation of the project crashing problem with partial correlation information as a convex-concave saddle point problem using the result from the previous section for parallel networks.

**Proposition 2.** Let \( x_i \in \mathbb{R}_{|A_i|} \) and define \( S(x_i) = \text{Diag}(x_i) - x_i x_i^T \) for all \( i = 1, \ldots, n - 1 \). Under the nonoverlapping multivariate marginal moment model for a general network, the distributionally robust PCP (3.22) is solvable as a convex-concave saddle point problem:

\[
\min_{(\mu, \sigma) \in \Omega} \max_{x \in X} f_{nmm}(\mu, \sigma, x), \tag{3.24}
\]

where

\[
f_{nmm}(\mu, \sigma, x) = \sum_{i=1}^{n-1} \left( \mu_i^T x_i + \text{trace} \left( \Sigma_i^{1/2} S(x_i) \Sigma_i^{1/2} \right) \right). \tag{3.25}
\]

The objective function is convex with respect to the moment variables \( \mu_i \) and \( \Sigma_i^{1/2} \) (and hence \( \sigma_i \) for a fixed \( \rho_i \)) and strictly concave with respect to the arc criticality index variables \( x \).

**Proof.** See Appendix.
4 Saddle Point Methods for Project Crashing

In this section, we illustrate the possibility of using first order saddle point methods to solve distributionally robust project crashing problems.

4.1 Gradient Characterization and Optimality Condition

We first characterize the gradient of the objective function for the parallel network and the general network before characterizing the optimality condition.

**Proposition 3.** Define \( T(x) = \Sigma^{1/2}S(x)\Sigma^{1/2} \). Under the cross moment model with a parallel network, the gradient of \( f_{\text{cmm}} \) in (3.19) with respect to \( x \) is given as:

\[
\nabla_x f_{\text{cmm}}(\mu, \sigma, x) = \mu + \frac{1}{2} \left( \text{diag}(\Sigma^{1/2}(T^{1/2}(x))^\dagger \Sigma^{1/2}) - 2\Sigma^{1/2}(T^{1/2}(x))^\dagger \Sigma^{1/2}x \right).
\]

The gradient of \( f_{\text{cmm}} \) in (3.19) with respect to \((\mu, \sigma)\) is given as:

\[
\nabla_{\mu, \sigma} f_{\text{cmm}}(\mu, \sigma, x) = \left( x, \text{diag}(\Sigma^{-1/2}T^{1/2}(x)\Sigma^{-1/2}\text{Diag}(\sigma)\rho) \right).
\]

**Proof.** See Appendix. \( \square \)

The optimality condition for (3.18) is then given as:

\[
x = P_\Delta \left( x + \nabla_x f_{\text{cmm}}(\mu, \sigma, x) \right)
\]

\[
(\mu, \sigma) = P_\Omega \left( (\mu, \sigma) - \nabla_{\mu, \sigma} f_{\text{cmm}}(\mu, \sigma, x) \right),
\]

where \( P_S(\cdot) \) denotes the projection onto a set \( S \) and \( \nabla \) denotes the partial derivative.

Similarly for the general network with partial correlations, we can extend the gradient characterization from Proposition 3 to the general network. Define \( T(x_i) = \Sigma_i^{1/2}S(x_i)\Sigma_i^{1/2}, \forall i = 1, \ldots, n-1 \). The gradients of \( f_{\text{cmm}} \) with respect to \( x \) and \((\mu, \sigma)\) are

\[
\nabla_x f_{\text{cmm}}(\mu, \sigma, x) = \begin{pmatrix}
\mu_1 + g_x(\sigma_1, x_1) \\
\mu_2 + g_x(\sigma_2, x_2) \\
\vdots \\
\mu_{n-1} + g_x(\sigma_{n-1}, x_{n-1})
\end{pmatrix},
\nabla_{\mu, \sigma} f_{\text{cmm}}(\mu, \sigma, x) = \begin{pmatrix}
x_1 \\
x_2 \\
\vdots \\
x_{n-1}
\end{pmatrix},
\begin{pmatrix}
g_x(\sigma_1, x_1) \\
g_x(\sigma_2, x_2) \\
\vdots \\
g_x(\sigma_{n-1}, x_{n-1})
\end{pmatrix},
\]

where

\[
g_x(\sigma_i, x_i) = \frac{1}{2} \left( \text{diag}(\Sigma_i^{1/2}(T^{1/2}(x_i))^\dagger \Sigma_i^{1/2}) - 2\Sigma_i^{1/2}(T^{1/2}(x_i))^\dagger \Sigma_i^{1/2}x_i \right), \forall i = 1, \ldots, n-1,
\]

\[
g_\sigma(\sigma_i, x_i) = \text{diag} \left( \Sigma_i^{-1/2}T^{1/2}(x_i)\Sigma_i^{-1/2}\text{Diag}(\sigma_i)\rho_i \right), \forall i = 1, \ldots, n-1.
\]

The optimality condition for (3.24) is then given as:

\[
x = P_X \left( x + \nabla_x f_{\text{cmm}}(\mu, \sigma, x) \right)
\]

\[
(\mu, \sigma) = P_\Omega \left( (\mu, \sigma) - \nabla_{\mu, \sigma} f_{\text{cmm}}(\mu, \sigma, x) \right).
\]

In the next section, we discuss saddle point methods that can be used to solve the problem.
4.2 Algorithm

In this section, we discuss the possibility of the use of saddle point algorithms to solve the distributionally robust project crashing problem. Define the inner maximization problem \( \phi(\mu, \sigma) = \max_{x \in X} f(\mu, \sigma, x) \) which requires solving a maximization problem of a strictly concave function over a polyhedral set \( X \). One possible method is to use a projected gradient method possibly with an Armijo line search method to compute the value of \( \phi(\mu, \sigma) \) and the corresponding optimal \( x^*(\mu, \sigma) \). Such an algorithm is described in Algorithm 1 and has been used in (1) to solve the inner maximization problem in a discrete choice problem setting.

**Algorithm 1: Projected gradient algorithm with Armijo search**

**Input:** \( \mu, \sigma, X \), starting point \( x_0 \), initial step size \( \alpha \), tolerance \( \epsilon \).

**Output:** Optimal solution \( x \).

Initialize stopping criteria: \( \text{criteria} \leftarrow \epsilon + 1 \);

while \( \text{criteria} > \epsilon \) do

\( z \leftarrow P_X(x_0 + \nabla_x f(\mu, \sigma, x_0)) \),

\( \text{criteria} \leftarrow ||z - x_0|| \),

\( x \leftarrow x_0 + \gamma(z - x_0) \), where \( \gamma \) is determined with an Armijo rule, i.e. \( \gamma = \alpha \cdot 2^{-l} \) with

\[ l = \min\{j \in \mathbb{Z}_+ : f(\mu, \sigma, x_0 + \alpha \cdot 2^{-j}(z - x_0)) \geq f(\mu, \sigma, x_0) + \tau \alpha 2^{-j}\langle \nabla_x f(\mu, \sigma, x_0), z - x_0 \rangle \} \]

for some \( \tau \in (0, 1) \).

\( x_0 \leftarrow x \).

end

The optimality condition (4.6) in this case is reduced to:

\[ (\mu, \sigma) = P_\Omega\left((\mu, \sigma) - F(\mu, \sigma)\right), \quad (4.7) \]

where

\[ F(\mu, \sigma) = \nabla_{\mu, \sigma} f(\mu, \sigma, x^*(\mu, \sigma)) \quad (4.8) \]

**Proposition 4.** The operator \( F \) as defined in (4.8) is continuous and monotone.

**Proof.** First, the optimal solution \( x^*(\mu, \sigma) \) to \( \max_{x \in X} f(\mu, \sigma, x) \) is unique because of the strict concavity of \( f(\mu, \sigma, x) \) with respect to \( x \). Moreover, \( x^*(\mu, \sigma) \) is continuous with respect to \( (\mu, \sigma) \) since \( f \) is strictly concave with respect to \( x \) and \( X \) is convex and bounded (see (21)). In addition, the function \( \nabla_{\mu, \sigma} f(\mu, \sigma, x) \) is continuous with respect to \( (\mu, \sigma) \) and \( x \). Therefore, \( F(\mu, \sigma) \) is continuous with respect to \( (\mu, \sigma) \). Notice that \( F(\mu, \sigma) \) is a subgradient of the convex function \( \phi(\mu, \sigma) = \max_{x \in X} f(\mu, \sigma, x) \) (see (48)). Hence \( F \) is monotone:

\[ \langle F(\mu, \sigma) - F(\mu, \sigma), (\mu, \sigma) - (\mu, \sigma) \rangle \geq 0, \quad \forall (\mu, \sigma), (\bar{\mu}, \bar{\sigma}) \in \Omega. \quad (4.9) \]
The optimality condition is then equivalent to the following variational inequality (18):

\[
\text{find } (\mu^*, \sigma^*) \in \Omega : \langle F(\mu^*, \sigma^*), (\mu, \sigma) - (\mu^*, \sigma^*) \rangle \geq 0, \forall (\mu, \sigma) \in \Omega.
\]

(4.10)

Under the condition that the operator \( F \) is continuous monotone, one method to find a solution to such a variational inequality is the projection and contraction method (31). The algorithm is as follows:

Algorithm 2: Projection and contraction algorithm for monotone variational inequalities

**Input**: Parameters for set \( \Omega \), the starting point \((\mu_0, \sigma_0)\), initial step size \(\alpha\), tolerance \(\epsilon\).

**Output**: Optimal solution \((\mu, \sigma)\).

Initialize stopping criteria: \(\text{criteria} \leftarrow \epsilon + 1\), set a value of \(\delta \in (0, 1)\).

while \(\text{criteria} > \epsilon\) do

\(\beta \leftarrow \alpha\)

\(\text{res} \leftarrow (\mu_0, \sigma_0) - P_\Omega((\mu_0, \sigma_0) - \beta F(\mu_0, \sigma_0))\)

\(d \leftarrow \text{res} - \beta[F(\mu_0, \sigma_0) - F(P_\Omega((\mu_0, \sigma_0) - \beta F(\mu_0, \sigma_0)))]\)

\(\text{criteria} \leftarrow \|\text{res}\|\)

while \(\langle \text{res}, d \rangle < \delta\|\text{res}\|^2\) do

\(\beta \leftarrow \beta/2\)

\(\text{res} \leftarrow (\mu_0, \sigma_0) - P_\Omega((\mu_0, \sigma_0) - \beta F(\mu_0, \sigma_0))\)

\(d \leftarrow \text{res} - \beta[F(\mu_0, \sigma_0) - F(P_\Omega((\mu_0, \sigma_0) - \beta F(\mu_0, \sigma_0)))]\)

end

\((\mu, \sigma) \leftarrow (\mu_0, \sigma_0) - \frac{\langle \text{res}, d \rangle}{\langle d, d \rangle} \cdot d, \ (\mu_0, \sigma_0) \leftarrow (\mu, \sigma)\).

end

We have chosen to solve the saddle point formulation of the robust PCP using a projection contraction method because it is relatively easy to implement, uses little storage, and therefore it is an attractive alternative for solving large-scale problems in general. (See for example (56) and the references therein for an extensive discussion on the properties and advantages of projection type algorithms for solving variational inequalities.) Most of the recent papers in robust and distributionally robust optimization solve formulations with min-max objective functions by first taking the dual of the inner problem as we have also discussed in formulations (3.9), (3.17) and (3.24). In this paper instead, we solve the saddle point formulations directly by making use of the projection type algorithms which seems to be reasonable, especially for large scale problems. Our experiments in the next section illustrate this concept and shows that more such algorithms can be used to solve problems with similar structure. (See (30)) for a survey on successful applications of such algorithms to solve variational inequalities arising from a broad range of of applications.)

5 Numerical Experiments

In this section, we report the results from numerical tests in which we solve the project crashing problem under various models. We also demonstrate that for large instances of the problem solving the saddle point reformula-
tions of the robust project crashing problem under CMM and NMM models using Algorithm 2 is more efficient than solving their SDP formulations using a standard solver. For the numerical tests, we assume that the feasible set of \((\mu, \sigma)\) is defined as

\[
\Omega = \left\{ (\mu, \sigma) : \sum_{(i,j) \in A} c^{(1)}_{ij}(\mu_{ij}) + c^{(2)}_{ij}(\sigma_{ij}) \leq B, \mu_{ij} \leq \bar{\mu}_{ij}, \sigma_{ij} \leq \bar{\sigma}_{ij}, \forall (i,j) \in A \right\}. \tag{5.1}
\]

The cost functions are assumed to be convex and quadratic of the form:

\[
c^{(1)}_{ij}(\mu_{ij}) = a^{(1)}_{ij}(\mu_{ij} - \bar{\mu}_{ij}) + b^{(1)}_{ij}(\mu_{ij} - \bar{\mu}_{ij})^2, \quad \forall (i,j) \in A, \tag{5.2}
\]

and

\[
c^{(2)}_{ij}(\sigma_{ij}) = a^{(2)}_{ij}(\sigma_{ij} - \bar{\sigma}_{ij}) + b^{(2)}_{ij}(\sigma_{ij} - \bar{\sigma}_{ij})^2, \quad \forall (i,j) \in A, \tag{5.3}
\]

where \(a^{(1)}_{ij}, a^{(2)}_{ij}, b^{(1)}_{ij}\) and \(b^{(2)}_{ij}\) are given nonnegative real numbers. These cost functions are chosen so that: (a) the costs are 0 under the normal mean and standard deviation denoted by \(\bar{\mu}_{ij}\) and \(\bar{\sigma}_{ij}\); and (b) the costs are convex decreasing functions of \(\mu_{ij}\) and \(\sigma_{ij}\) respectively. In our tests, we compare the distributionally robust project crashing solution with the following models:

1. Deterministic PCP: In this model, we simply use the mean value of the random activity durations as the deterministic activity durations and ignore the variability. The crashing solution in this case is given as:

\[
\min_{(\mu, \sigma) \in \Omega} \max_{x \in X \cap \{0, 1\}^m} \sum_{(i,j) \in A} \mu_{ij} x_{ij}. \tag{5.4}
\]

2. Heuristic PCP: (35) developed a heuristic approach to minimize the quantile of the makespan by using a surrogate deterministic objective function with activity durations defined as:

\[
d_{ij} = \mu_{ij} + \kappa \cdot \sigma_{ij},
\]

with fixed margin coefficients \(\kappa \geq 0\). They then solved the following deterministic model to develop a heuristic solution for project crashing:

\[
\min_{(\mu, \sigma) \in \Omega} \max_{x \in X \cap \{0, 1\}^m} \sum_{(i,j) \in A} (\mu_{ij} + \kappa \cdot \sigma_{ij}) x_{ij}. \tag{5.5}
\]

In the numerical tests, we set \(\kappa = 3\).

3. Sample Average Approximation (SAA): Consider the case where the activity duration vector \(\hat{t}\) is a multivariate normal random vector \(N(\mu, \Sigma)\), where \(\mu\) and \(\Sigma\) are the mean and the covariance matrix of the random vector \(\hat{t}\), respectively. Then \(\hat{t} = \mu + \text{Diag}(\sigma)\xi\), where \(\sigma\) is a vector of standard deviations of \(\hat{t}_{ij}, (i,j) \in A\) and \(\xi \sim N(0, \rho)\) is a normally distributed random vector where \(\rho\) is the correlation matrix of \(\xi\) and hence \(\hat{t}\). Let \(\xi^{(k)}, k = 1, 2, \ldots, N\), denote a set of independent and identical samples of the random vector \(\xi\). The SAA formulation is then given as:

\[
\min_{(\mu, \sigma) \in \Omega} \frac{1}{N} \sum_{k=1}^{N} \left( \max_{x \in X} (\mu + \text{Diag}(\sigma)\xi^{(k)})^T x \right), \tag{5.6}
\]
which is equivalent to the optimization problem:

\[
\begin{align*}
\min_{\mu, \sigma, y^{(k)}} & \quad \frac{1}{N} \sum_{k=1}^{N} (y^{(k)}_n - y^{(k)}_1) \\
\text{s.t.} & \quad y^{(k)}_j - y^{(k)}_i \geq \mu_{ij} + \sigma_{ij} \xi^{(k)}_{ij}, \quad \forall k = 1, \ldots, N, \\
& \quad (\mu, \sigma) \in \Omega.
\end{align*}
\] (5.7)

In our experiments, we use a sample size of \(N = 5000\).

Since the feasible region \(\Omega\) in (5.1) is quadratic and convex, all the three formulations are solvable as convex quadratic programming problems.

**Example 1 (Small Network)**

In the first example, we consider the small project network in Figure 3 to illustrate the difference in the crashing solutions from the different models. The mean and standard deviation for the original activity durations are set to:

\[
\begin{align*}
\mu_1 &= \begin{pmatrix} \bar{\mu}_{12} \\ \bar{\mu}_{13} \end{pmatrix} = \begin{pmatrix} 2 \\ 2.5 \end{pmatrix}, & \sigma_1 &= \begin{pmatrix} \bar{\sigma}_{12} \\ \bar{\sigma}_{13} \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, & \rho_1 &= \begin{pmatrix} 1 & 0.5 & 0 \end{pmatrix} \\
\mu_2 &= \begin{pmatrix} \bar{\mu}_{23} \\ \bar{\mu}_{24} \end{pmatrix} = \begin{pmatrix} 4 \\ 3 \end{pmatrix}, & \sigma_2 &= \begin{pmatrix} \bar{\sigma}_{23} \\ \bar{\sigma}_{24} \end{pmatrix} = \begin{pmatrix} 1.5 \\ 2 \end{pmatrix}, & \rho_2 &= \begin{pmatrix} 1 & -0.2 \end{pmatrix} \\
\mu_3 &= \bar{\mu}_{34} = 4, & \sigma_3 &= \bar{\sigma}_{34} = 3.
\end{align*}
\]

For the SAA approach, the correlations between the activity durations originating from different nodes is set to 0. To apply the CMM approach for this small network, we convert the network to a parallel network by path enumeration. The makespan in this case is equal to \(\max(\tilde{t}_{12} + \tilde{t}_{24}, \tilde{t}_{14}, \tilde{t}_{12} + \tilde{t}_{23} + \tilde{t}_{34}, \tilde{t}_{13} + \tilde{t}_{34})\). We assume that
the minimum value of the mean and standard deviation that can be achieved by crashing is half of the mean and standard deviation of the original activity durations. As the cost budget $B$ varied, the time cost trade-off of the six models: Deterministic PCP (5.4), Heuristic PCP (5.5), SAA (5.7), and the Distributionally Robust PCP under MMM, CMM and NMM is shown in Figure 4 for both the linear (setting $b_{ij}^{(1)} = b_{ij}^{(2)} = 0$) and quadratic cost function case.

<table>
<thead>
<tr>
<th>Model</th>
<th>Linear Cost Function</th>
<th>Quadratic Cost Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deterministic</td>
<td>Optimal value</td>
<td>Optimal value</td>
</tr>
<tr>
<td>Heuristic</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SAA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MMM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NMM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CMM</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 4: Optimal objective value as cost budget increases

From Figure 4, we see that all the makespans decrease as the cost budget increases as should be expected. The deterministic PCP always has the smallest objective, since its objective is the lower bound for the expected makespan by Jensen’s inequality. The objective value of distributionally robust PCP are tight upper bounds for the expected makespan under MMM, CMM and NMM. We also find the objective values of the distributionally robust PCP is fairly close to the expected makespan under normal distribution, and by using more correlation information the objective value is closer to the expected makespan under SAA. However, the objective value of the heuristic PCP is much larger than the objective value of other models implying it is a poor approximation of the expected makespan. We also see that the objective value of deterministic PCP and heuristic PCP does not change when the cost budget exceeds a certain value (close to half of the budget upper bound). This implies that under these models the critical paths of the two models do not change beyond this budget and the mean and standard deviation of the activity durations on the critical paths have been crashed to minimum. This highlights the limitations of such heuristic methods. In this case, reducing the mean and variance of other activity durations will not change the objective value of the deterministic PCP and heuristic PCP. In comparison, under uncertainty the objective values of SAA and the distributionally robust PCP always decreases as the cost increases.

Next, we compare the expected makespan under the assumption of a normal distribution using the crashed solution of these models. Using the optimal $(\mu, \sigma)$ obtained by the tested models, and assuming that the activity duration vector follows a multivariate normal distribution with mean $\mu$ and covariance matrix $\Sigma = \text{Diag}(\sigma) \rho \text{Diag}(\sigma)$, we compute the expected makespan by a Monte Carlo simulation with 10000 samples. The results are shown
in Figure 5. As expected, using the optimal solution obtained by SAA, we get the smallest expected makespan.

The expected makespan with linear cost function

The expected makespan with quadratic cost function

Figure 5: The expected makespan under the multivariate normal distribution

When the cost budget is small, the deterministic PCP also provides a reasonable decision with small expected makespan, but this model is not robust. When the budget is large, we observe that the expected makespan of the deterministic model is much larger than the expected makespans of other models. The heuristic PCP has better performance that the deterministic PCP, but still the gap of the expected makespan between this model and SAA is large. The solutions obtained by the distributionally robust models are very close to the expected makespan of SAA.

We now provide a comparison of the optimal solutions from the various models. Assume the cost function is linear and the cost budget $B = 5.4375$. In this case the expected makespans are 5.43 (MMM), 5.43 (NMM), 5.40 (CMM) and 5.39 (SAA), and the standard deviation of the makespans are 1.10 (MMM), 1.10 (NMM), 1.14 (CMM) and 1.18 (SAA). The optimal solutions of the four models are shown in Figure 6. The optimal solutions

Figure 6: Optimal solutions obtained by MMM, NMM, CMM and SAA

from the four models are fairly close. For activities (1,2) and (1,3), SAA tends to crash the mean more while the
robust models tend to crash the standard deviation more. We validate this observation with a larger numerical example later.

**Example 2 (Parallel Network)**

In this example, we consider a project with \( k \) parallel activities. The data is randomly generated as follows:

1. For every activity, the mean and the standard deviation of the original activity duration are generated by uniform distributions \( \mu_{ij} \sim U(10, 20), \sigma_{ij} \sim U(6, 10) \), and the minimal values of mean and standard deviation that can be obtained by crashing are \( \mu_{ij} \sim U(5, 10), \sigma_{ij} \sim U(2, 6) \).

2. The coefficients in the cost function (5.2) are chosen as follows \( a^{(1)}_{ij} \sim U(1, 2), b^{(1)}_{ij} \sim U(0, 1), a^{(1)}_{ij} \sim U(1, 2) \) and \( b^{(2)}_{ij} \sim U(0, 1) \). The amount of the cost budget is chosen as \( \frac{1}{4} \sum_{(i,j)} (c^{(1)}_{ij}(\mu_{ij}) + c^{(2)}_{ij}(\sigma_{ij})) \).

We first consider a simple case with two parallel activities. In Figure 7, we plot the optimal values of \( f_{cmm} \) as the correlation between the two activity durations increases from \(-1\) to \(1\), and compare these values with the optimal value of \( f_{mmm} \) for one such random instance. The worst-case expected project makespan \( f_{cmm} \) is a decreasing function of the correlation \( \rho \), and when \( \rho = -1 \) (perfectly negatively correlated), the worst-case expected makespan under CMM and MMM are the same. Clearly if the activity durations are positively correlated, then the bound from capturing correlation information is much tighter.

Next, we consider a parallel network with 10 activities. We compute the optimal values of the crashed moments \((\mu, \sigma)\) under the MMM, CMM and SAA models and compute the expected makespans under these moments by assuming that the activity durations follows a multivariate normal distribution \( N(\mu, \sigma) \). We consider two types of instances - one with uncorrelated activity durations and the other with highly correlated activity durations. The distribution of the makespan and the statistics are provided in Figure 8 and Table 1. When the activities are
uncorrelated, with the optimal solution obtained from MMM and CMM, the distribution of the makespan is very close. However when the activities are highly correlated, the distribution is farther apart. As should be expected, SAA provides the smallest expected makespan under the normal distribution. However, the maximum value and standard deviation of the makespan obtained by SAA are the largest in comparison to the distributionally robust models indicating that the robust models provide a reduction in the variability of the makespan.

![Probability density function of the makespan](image)

(a) Uncorrelated activity durations

(b) Highly correlated activity durations

Figure 8: Probability density function of the makespan

<table>
<thead>
<tr>
<th>Makespan Statistics</th>
<th>Uncorrelated activities</th>
<th>Highly correlated activities</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MMM</td>
<td>CMM</td>
</tr>
<tr>
<td>Max</td>
<td>34.9514</td>
<td>35.0389</td>
</tr>
<tr>
<td>Mean</td>
<td>20.5953</td>
<td>20.5539</td>
</tr>
<tr>
<td>Median</td>
<td>20.312</td>
<td>20.2567</td>
</tr>
</tbody>
</table>

Table 1: Statistics of the makespan

Finally, we compare the CPU time of solving the SDP reformulation (3.17) and the saddle point reformulation (3.18) for the distributionally robust PCP under CMM. To solve the SDP (3.17), we used CVX, a package for specifying and solving convex programs ((27, 26)) with the solver SDPT3 ((51, 52)). We set the accuracy of the SDP solver with “cvx_precision low”. To solve the saddle point problem (3.18), we use Algorithm 2 with tolerance $\epsilon = 10^{-3}$. In our implementation, we used the mean and standard deviation obtained from MMM by solving a SOCP as a warm start for Algorithm 2. The average value of the CPU time and the objective function for 10 randomly generated instances is provided in Table 2. For large instances, it is clear that using the saddle point algorithm to solve the problem to reasonable accuracy for all practical purposes is much faster than solving the semidefinite program formulation of the problem with interior point method solvers.
Table 2: CPU time of the SDP solver and Algorithm 2

<table>
<thead>
<tr>
<th>k</th>
<th>CPU time in seconds</th>
<th>Objective value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SDP formulation</td>
<td>Saddle point formulation</td>
</tr>
<tr>
<td></td>
<td>SDP formulation</td>
<td>Saddle point formulation</td>
</tr>
<tr>
<td>20</td>
<td>0.81</td>
<td>4.47</td>
</tr>
<tr>
<td></td>
<td>31.642</td>
<td>31.643</td>
</tr>
<tr>
<td>40</td>
<td>3.36</td>
<td>9.06</td>
</tr>
<tr>
<td></td>
<td>39.273</td>
<td>39.274</td>
</tr>
<tr>
<td>60</td>
<td>22.73</td>
<td>37.57</td>
</tr>
<tr>
<td></td>
<td>45.909</td>
<td>45.910</td>
</tr>
<tr>
<td>80</td>
<td>77.53</td>
<td>71.42</td>
</tr>
<tr>
<td></td>
<td>50.400</td>
<td>50.401</td>
</tr>
<tr>
<td>100</td>
<td>255.54</td>
<td>169.65</td>
</tr>
<tr>
<td></td>
<td>54.261</td>
<td>54.261</td>
</tr>
<tr>
<td>120</td>
<td>685.56</td>
<td>297.57</td>
</tr>
<tr>
<td></td>
<td>58.541</td>
<td>58.542</td>
</tr>
<tr>
<td>140</td>
<td>1749.25</td>
<td>458.60</td>
</tr>
<tr>
<td></td>
<td>62.577</td>
<td>62.579</td>
</tr>
<tr>
<td>160</td>
<td>**</td>
<td>568.91</td>
</tr>
<tr>
<td></td>
<td>**</td>
<td>65.025</td>
</tr>
<tr>
<td>180</td>
<td>**</td>
<td>810.99</td>
</tr>
<tr>
<td></td>
<td>**</td>
<td>68.581</td>
</tr>
<tr>
<td>200</td>
<td>**</td>
<td>1255.38</td>
</tr>
<tr>
<td></td>
<td>**</td>
<td>70.919</td>
</tr>
</tbody>
</table>

** means the instances cannot be solved in 2 hours by the SDP solver.

Example 3 (Grid Network)

In this third example, we consider a grid network (see Figure 9). The size of the problem is determined by its width and height. Let width = w and height = h in which case there are n = (w + 1)(h + 1) nodes, m = w(h + 1) + h(w + 1) activities and (w+h-1/2) possible critical paths in the project. For example, in Figure 9, w = 6 and h = 4, then there are 35 nodes, 58 activities and 210 possible critical paths.

![Figure 9: Grid project network with width = 6, height=4](image)

We test the distributionally robust project crashing models with randomly generated data. The data is chosen as follows:

1. For every activity (i, j) ∈ A, the mean and the standard deviation of the original activity duration are generated by uniform distributions π_{ij} ∼ U(5, 10), σ_{ij} ∼ U(4, 8), and the minimal values of mean and standard deviation that can be obtained by crashing are chosen as μ_{ij} ∼ U(2, π_{ij}), σ_{ij} ∼ U(1, σ_{ij}).

2. For the coefficients in the cost function (5.2), we choose a_{1j}^{(1)} ∼ U(2, 4), b_{1j}^{(1)} ∼ U(0, 1), a_{1j}^{(2)} ∼ U(1, 2) and b_{1j}^{(2)} ∼ U(0, 1) for all (i, j) ∈ A.
3. The amount of the cost budget is chosen as $\sum_{(i,j)} a_{ij}^{(1)} (\mu_{ij} - \mu_{ij}) + a_{ij}^{(2)} (\mu_{ij} - \mu_{ij})^2$. In the deterministic model (5.4), an optimal strategy is to reduce the mean of every activity duration to its lower bound without the change of variance.

4. For simulations, the activities are assumed to be independent which implies that the correlation matrix for the activity durations is an identity matrix.

Both the deterministic PCP (5.4) and the heuristic PCP (5.5) can be formulated as convex quadratic programs which can be quickly solved. Solving the distributionally robust PCP and SAA are more computationally expensive. We compare the expected makespan with the optimal solutions obtained by the Deterministic PCP (5.4), Heuristic PCP (5.5), SAA and the distributionally robust PCP under MMM and NMM. Let $E(T_0)$ denote the expected makespan without crashing, and $E(T_1)$ denote the expected makespan with crashing by the deterministic model (5.4). We define the “reduction” as the percentage of the extra expected makespan achieved by the other models, that is

$$100 \cdot \left( \frac{E(T_0) - E(T_{new})}{E(T_0) - E(T_1)} - 1 \right),$$

where $E(T_{new})$ is the expected makespan with crashed activity durations obtained from a project crashing model (Heuristic, SAA, MMM or NMM).

The numerical results presented in Table 3 are the average of 10 randomly generated instances. With the crashed activity durations, we compare the expected makespan under four different distributions including normal, uniform, gamma and the worst-case distribution in NMM. The expected makespans achieved by the distributionally robust PCP are always smaller than the deterministic and heuristic PCP models. The improvement of the distributionally robust PCP is much larger than the improvement of the heuristic PCP under all the distributions. In comparison with SAA, we see that the expected makespans obtained by MMM and NMM are bigger than the expected makespans obtained by SAA under normal, uniform and gamma distributions. However, the gaps are quite small and under the worst-case distribution the expected makespan of NMM and MMM are always smaller than the expected makespan of SAA. Moreover, we find that the computational time of solving MMM and NMM is smaller than solving SAA. Between MMM and NMM, the additional information in this graph is the correlation between each pair of activities originated from a node. Due to the grid network structure, we find that the optimal solutions between MMM and NMM are much closer in comparison to the parallel graph in Example 2. We also compare the optimal crashing decisions obtained by SAA and the distributionally robust PCP. The results for all 10 instances for the $8 \times 8$ grid network are plotted in Figure 10. From the figure, we see that the SAA model tends to crash the means more while NMM tends to crash the standard deviations more.

6 Conclusions

In this paper, we proposed a class of distributionally robust project crashing problems that is solvable in polynomial time where the objective is to choose the first two marginal moments to minimize the worst-case expected
<table>
<thead>
<tr>
<th>size</th>
<th>models</th>
<th>objective</th>
<th>time</th>
<th>expected makespan</th>
<th>reduction (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 x 1 grid</td>
<td>Deterministic</td>
<td>14.30</td>
<td>0.44</td>
<td>20.12 22.27 19.65</td>
<td>-</td>
</tr>
<tr>
<td>7 arcs</td>
<td>Heuristic</td>
<td>45.92</td>
<td>0.52</td>
<td>21.41 21.46 21.41</td>
<td>-1.44 -13.89</td>
</tr>
<tr>
<td>2 x 2 grid</td>
<td>MMM</td>
<td>26.04</td>
<td>0.72</td>
<td>19.47 19.55 19.36</td>
<td>8.28 8.55 3.01</td>
</tr>
<tr>
<td>12 arcs</td>
<td>NMM</td>
<td>25.98</td>
<td>0.82</td>
<td>19.40 19.48 19.27</td>
<td>9.10 9.35 3.84</td>
</tr>
<tr>
<td>3 x 3 grid</td>
<td>SAA</td>
<td>19.00</td>
<td>42.34</td>
<td>19.00 19.08 18.80</td>
<td>13.75 14.04 8.93</td>
</tr>
<tr>
<td>24 arcs</td>
<td>Deterministic</td>
<td>18.37</td>
<td>0.45</td>
<td>28.90 29.03 28.46</td>
<td>-</td>
</tr>
<tr>
<td>4 x 3 grid</td>
<td>Heuristic</td>
<td>59.95</td>
<td>0.61</td>
<td>29.06 29.12 29.15</td>
<td>-1.21 -0.50 -5.56</td>
</tr>
<tr>
<td>31 arcs</td>
<td>MMM</td>
<td>37.71</td>
<td>0.97</td>
<td>27.32 27.39 27.32</td>
<td>14.86 15.45 10.51</td>
</tr>
<tr>
<td>6 x 4 grid</td>
<td>NMM</td>
<td>36.71</td>
<td>1.14</td>
<td>27.28 27.35 27.28</td>
<td>15.14 15.74 10.74</td>
</tr>
<tr>
<td>6 x 6 grid</td>
<td>Deterministic</td>
<td>33.00</td>
<td>0.62</td>
<td>54.83 54.89 55.67</td>
<td>-</td>
</tr>
<tr>
<td>84 arcs</td>
<td>Heuristic</td>
<td>106.59</td>
<td>0.96</td>
<td>52.67 52.67 53.29</td>
<td>11.37 11.66 12.51</td>
</tr>
<tr>
<td>31 arcs</td>
<td>MMM</td>
<td>74.66</td>
<td>1.68</td>
<td>50.30 50.34 50.77</td>
<td>25.31 25.38 27.18</td>
</tr>
<tr>
<td>8 x 4 grid</td>
<td>NMM</td>
<td>61.74</td>
<td>1.41</td>
<td>42.52 42.51 42.76</td>
<td>23.06 22.98 20.92</td>
</tr>
<tr>
<td>58 arcs</td>
<td>SAA</td>
<td>60.73</td>
<td>2.50</td>
<td>42.52 42.51 42.74</td>
<td>23.07 22.99 21.02</td>
</tr>
<tr>
<td>8 x 6 grid</td>
<td>Deterministic</td>
<td>41.39</td>
<td>128.06</td>
<td>41.39 41.39 41.90</td>
<td>29.46 29.37 25.64</td>
</tr>
<tr>
<td>110 arcs</td>
<td>Heuristic</td>
<td>33.00</td>
<td>0.62</td>
<td>54.83 54.89 55.67</td>
<td>-</td>
</tr>
<tr>
<td>8 x 6 grid</td>
<td>MMM</td>
<td>116.44</td>
<td>2.83</td>
<td>73.60 73.52 74.66</td>
<td>31.11 30.80 35.74</td>
</tr>
<tr>
<td>144 arcs</td>
<td>NMM</td>
<td>73.64</td>
<td>2.98</td>
<td>50.31 50.35 50.79</td>
<td>25.25 25.33 27.10</td>
</tr>
<tr>
<td>10 x 10 grid</td>
<td>SAA</td>
<td>48.89</td>
<td>163.02</td>
<td>48.89 48.91 49.77</td>
<td>32.44 32.59 32.15</td>
</tr>
<tr>
<td>220 arcs</td>
<td>Deterministic</td>
<td>47.63</td>
<td>0.85</td>
<td>81.31 81.17 83.64</td>
<td>-</td>
</tr>
<tr>
<td>8 x 8 grid</td>
<td>Heuristic</td>
<td>153.50</td>
<td>1.49</td>
<td>76.91 76.82 78.37</td>
<td>16.87 16.57 19.93</td>
</tr>
<tr>
<td>144 arcs</td>
<td>MMM</td>
<td>116.44</td>
<td>2.83</td>
<td>73.60 73.52 74.66</td>
<td>31.11 30.80 35.74</td>
</tr>
<tr>
<td>10 x 10 grid</td>
<td>NMM</td>
<td>115.41</td>
<td>4.58</td>
<td>73.62 73.55 74.65</td>
<td>31.04 30.72 35.80</td>
</tr>
<tr>
<td>220 arcs</td>
<td>SAA</td>
<td>71.45</td>
<td>388.74</td>
<td>71.45 71.26 73.14</td>
<td>39.16 39.27 41.29</td>
</tr>
<tr>
<td>10 x 10 grid</td>
<td>Deterministic</td>
<td>55.77</td>
<td>1.07</td>
<td>99.36 99.19 103.30</td>
<td>-</td>
</tr>
<tr>
<td>220 arcs</td>
<td>Heuristic</td>
<td>181.39</td>
<td>1.97</td>
<td>92.47 92.37 94.55</td>
<td>21.42 21.17 27.97</td>
</tr>
<tr>
<td>8 x 8 grid</td>
<td>MMM</td>
<td>147.81</td>
<td>3.90</td>
<td>89.12 89.04 90.86</td>
<td>33.88 33.67 41.66</td>
</tr>
<tr>
<td>144 arcs</td>
<td>NMM</td>
<td>146.77</td>
<td>6.28</td>
<td>89.15 89.07 90.84</td>
<td>33.80 33.58 41.78</td>
</tr>
<tr>
<td>10 x 10 grid</td>
<td>SAA</td>
<td>86.21</td>
<td>594.79</td>
<td>86.21 86.05 88.81</td>
<td>42.87 42.89 48.08</td>
</tr>
<tr>
<td>220 arcs</td>
<td>Deterministic</td>
<td>66.33</td>
<td>1.31</td>
<td>117.60 117.56 123.52</td>
<td>-</td>
</tr>
<tr>
<td>8 x 6 grid</td>
<td>Heuristic</td>
<td>214.57</td>
<td>2.67</td>
<td>109.63 109.51 112.37</td>
<td>22.04 22.32 31.54</td>
</tr>
<tr>
<td>110 arcs</td>
<td>MMM</td>
<td>180.23</td>
<td>5.13</td>
<td>105.34 105.30 107.57</td>
<td>35.77 35.88 47.10</td>
</tr>
<tr>
<td>8 x 6 grid</td>
<td>NMM</td>
<td>179.19</td>
<td>8.08</td>
<td>105.37 105.33 107.58</td>
<td>35.69 35.80 47.09</td>
</tr>
<tr>
<td>144 arcs</td>
<td>SAA</td>
<td>101.82</td>
<td>880.33</td>
<td>101.82 101.65 105.34</td>
<td>45.51 46.01 53.39</td>
</tr>
<tr>
<td>10 x 10 grid</td>
<td>Deterministic</td>
<td>74.47</td>
<td>1.62</td>
<td>135.59 135.47 143.48</td>
<td>-</td>
</tr>
<tr>
<td>220 arcs</td>
<td>Heuristic</td>
<td>242.32</td>
<td>3.50</td>
<td>125.20 125.03 128.78</td>
<td>24.65 24.79 36.44</td>
</tr>
<tr>
<td>8 x 8 grid</td>
<td>MMM</td>
<td>214.44</td>
<td>6.77</td>
<td>120.64 120.50 123.74</td>
<td>37.73 37.89 51.21</td>
</tr>
<tr>
<td>144 arcs</td>
<td>NMM</td>
<td>213.40</td>
<td>10.43</td>
<td>120.67 120.54 123.78</td>
<td>37.66 37.81 51.16</td>
</tr>
<tr>
<td>10 x 10 grid</td>
<td>SAA</td>
<td>116.41</td>
<td>1385.43</td>
<td>116.41 116.11 120.96</td>
<td>47.87 48.44 58.12</td>
</tr>
<tr>
<td>220 arcs</td>
<td>Deterministic</td>
<td>93.18</td>
<td>2.50</td>
<td>172.33 171.67 183.83</td>
<td>-</td>
</tr>
<tr>
<td>10 x 10 grid</td>
<td>Heuristic</td>
<td>303.30</td>
<td>5.95</td>
<td>158.44 157.90 163.43</td>
<td>26.64 26.39 41.51</td>
</tr>
<tr>
<td>220 arcs</td>
<td>MMM</td>
<td>287.21</td>
<td>10.82</td>
<td>152.56 152.06 156.98</td>
<td>40.43 40.17 57.00</td>
</tr>
<tr>
<td>10 x 10 grid</td>
<td>NMM</td>
<td>286.17</td>
<td>16.88</td>
<td>152.59 152.09 157.06</td>
<td>40.38 40.11 56.87</td>
</tr>
<tr>
<td>220 arcs</td>
<td>SAA</td>
<td>146.78</td>
<td>2657.14</td>
<td>146.78 146.14 153.16</td>
<td>51.79 51.83 64.90</td>
</tr>
</tbody>
</table>
Comparison of the optimal mean obtained by SAA and NMM

Comparison of the optimal standard deviation obtained by SAA and NMM

Figure 10: Optimal solution comparison of SAA and NMM

project makespan. While semidefinite programming is the typical approach to tackle such problems, we provide an alternative saddle point reformulation over the moment and arc criticality index variables which helps us to use alternative methods to solve the problem. Numerical experiments show that this can help us solve larger instances of such problems. Furthermore, in terms of insights the robust models tend to crash the standard deviations more in comparison with the sample average approximation for standard distributions such as the multivariate normal distribution.

We believe there are several ways to build on this work. Given several developments that have occurred in first order methods for saddle point formulations in the recent years, we believe more can be done to apply these methods to solve distributionally robust optimization problems. To the best of knowledge, little has been done in this area thus far. Another research direction is to identify new instances where the distributionally robust project crashing problem is solvable in polynomial time. Lastly it would be interesting if these results can be used to find approximation guarantees for the general distributionally robust project crashing problem with arbitrary correlations.

Appendix

Proof of Proposition 2

We consider the inner maximization problem of (3.22), which is to compute the worst-case expected duration of the project with given mean, standard deviation and partial correlation information of the activity durations under the nonoverlapping structure. We denote it by

\[
\phi_{\text{nnmm}}(\mu, \sigma) = \max_{\theta \in \Theta_{\text{nnmm}, \rho_i, \forall i}(\mu_i, \sigma_i, \forall i)} \mathbb{E}_{\theta} \left( \max_{\bar{x} \in \bar{X}} \sum_{i=1}^{n-1} \bar{y}_i^T \bar{x}_i \right).
\]  

(6.1)
Applying Theorem 15 on page 467 in (38), the worst-case expected makespan in (6.1) is formulated as the following SDP:

$$
\phi_{nm\mu}(\mu, \sigma) = \max_{x_{ij}, w_{ij}, W_{ij}} \sum_{(i,j) \in A} e_{ij}^T w_{ij}
$$

s.t.  
\begin{align*}
\begin{pmatrix}
1 & \mu_i^T \\
\mu_i & \Sigma_i + \mu_i \mu_i^T
\end{pmatrix} - \sum_{(i,j) \in A} \begin{pmatrix}
x_{ij} & w_{ij}^T \\
w_{ij} & W_{ij}
\end{pmatrix} & \succeq 0, \quad \forall i = 1, \ldots, n - 1, \\
x_{ij} & \succeq 0, & \forall (i,j) \in A.
\end{align*}

To show the result of Proposition 2, we need the following lemma:

**Lemma 1.** The SDP problem (6.2) can be simplified as

$$
\max_{x, Y_i} \sum_{i=1}^{n-1} \text{trace}(Y_i)
$$

s.t.  
\begin{align*}
\begin{pmatrix}
\Sigma_i + \mu_i \mu_i^T & Y_i^T \\
Y_i & \text{Diag}(x_i)
\end{pmatrix} & \succeq 0, \quad \forall i = 1, \ldots, n - 1.
\end{align*}

**Proof.** First, we show the optimal value of (6.2) \leq the optimal value of (6.3). Consider an optimal solution to the SDP (6.2) denoted by \((x_{ij}^*, w_{ij}^*, W_{ij}^*)\) for \((i,j) \in A\). Let \(x = x^*\) and \(Y_i^T e_{ij} = w_{ij}^*\) for all \((i,j) \in A\). Then

$$
\text{trace}(Y_i) = \sum_{(i,j) \in A} e_{ij}^T w_{ij}^*,
$$

which implies

$$
\sum_{i=1}^{n-1} \text{trace}(Y_i) = \sum_{(i,j) \in A} e_{ij}^T w_{ij}^*.
$$

Next we verify that \(x_{ij}, Y_i, i = 1, \ldots, n - 1\) is feasible for (6.3). For an \(i = 1, \ldots, n - 1\), we consider the case with all the \(x_{ij}\) values being strictly positive first. In this case

\begin{align*}
& \begin{pmatrix}
\Sigma_i + \mu_i \mu_i^T & \mu_i \\
\mu_i^T & 1
\end{pmatrix} - \begin{pmatrix}
Y_i^T \\
x_i^T
\end{pmatrix} \text{Diag}(x_i)^{-1} \begin{pmatrix}
Y_i \\
x_i
\end{pmatrix} \\
= & \begin{pmatrix}
\Sigma_i + \mu_i \mu_i^T - Y_i^T \text{Diag}(x_i)^{-1} Y_i & \mu_i - Y_i^T 1 \\
\mu_i^T - 1^T Y_i & 1 - 1^T x_i
\end{pmatrix} \\
= & \begin{pmatrix}
\Sigma_i + \mu_i \mu_i^T - \sum_{(i,j) \in A} \frac{w_{ij} w_{ij}^T}{x_{ij}} & \mu_i - \sum_{(i,j) \in A} w_{ij} \\
\mu_i - \sum_{(i,j) \in A} w_{ij}^T & 1 - 1^T x_i
\end{pmatrix} \\
\succeq & \begin{pmatrix}
\Sigma_i + \mu_i \mu_i^T - W_{ij}^* & \mu_i - \sum_{(i,j) \in A} w_{ij} \\
\mu_i - \sum_{(i,j) \in A} w_{ij}^T & 1 - \sum_{(i,j) \in A} x_{ij}
\end{pmatrix} \\
\succeq & 0.
\end{align*}
The last two matrix inequalities come from the feasibility condition of (6.2). The case with some of the variables \(x_{ij} = 0\) is handled similarly by dropping the rows and columns corresponding to the zero entries. Thus the solution \((Y_i, x_i), i = 1, \ldots, n - 1\) is feasible to the semidefinite program (6.3) by the Schur complement condition for positive semidefiniteness. Therefore, the optimal value of (6.2) is less than or equal to the optimal value of (6.3).

Next, we show the optimal value of (6.2) ≥ the optimal value of (6.3). Consider an optimal solution to (6.3) denoted by \((Y_i^*, x_i^*), i = 1, \ldots, n - 1\). For an \(i = 1, \ldots, n - 1\) we consider the case \(x_{ij}^*\) are all positive for \((i, j) \in A_i\). From Schur’s complement, the positive semidefiniteness constraint in (6.3) is equivalent to:

\[
\begin{pmatrix}
\Sigma_i + \mu_i \mu_i^T - Y_i^{-1} \text{Diag}(x_i^*)^{-1} Y_i^* & \mu_i - Y_i^{-1} 1 \\
\mu_i^T - 1^T Y_i^* & 1 - 1^T x_i^*
\end{pmatrix} \succeq 0,
\]

Define:

\[
\begin{pmatrix}
W_{ij} & w_{ij} \\
w_{ij}^T & x_{ij}
\end{pmatrix} = \begin{pmatrix}
Y_i^* e_{ij} e_{ij}^T Y_i^* / x_{ij}^* & Y_i^* e_{ij} \\
e_{ij}^T Y_i^* & x_{ij}^*
\end{pmatrix}, \quad (i, j) \in A_i.
\]

Then \((W_{ij}, w_{ij}, x_{ij}), (i, j) \in A\) is a feasible solution to the SDP (6.2), the objective function has the same value as the optimal objective function value of (6.3). As before, the case with some of the \(x_{ij}^* = 0\) can be handled by dropping the rows and columns corresponding to the zeros. Therefore, the optimal value of (6.2) is greater than or equal to the optimal value of (6.3).

Given the formulation (6.2) and using Theorem 2 from (1) for each node \(i\), it is easy to verify that the SDP (6.3) is equivalent to:

\[
\max_{x \in X} \sum_{i=1}^{n-1} \left( \mu_i^T x_i + \text{trace} \left( \left( \Sigma_i^{1/2} S(x_i) \Sigma_i^{1/2} \right)^{1/2} \right) \right).
\]

Therefore, the project crashing problem is equivalent to

\[
\min_{(\mu, \sigma) \in \Omega} \max_{x \in X} \sum_{i=1}^{n} \left( \mu_i^T x_i + \text{trace} \left( \left( \Sigma_i^{1/2} S(x_i) \Sigma_i^{1/2} \right)^{1/2} \right) \right),
\]

where \(\Sigma_i = \text{Diag}(\sigma_i) \rho_i \text{Diag}(\sigma_i)\) is a matrix function of \(\sigma_i\), \(S(x_i) = \text{Diag}(x_i) - x_i x_i^T\). The convexity of the objective function with respect to \(\mu_i\) and \(\Sigma_i^{1/2}\) and concavity with respect to the \(x_i\) variables follows naturally from Proposition 1.

**Proof of Proposition 3**

The gradient of the function with respect to \(x\) is derived in Theorem 4 in (1). The gradient with respect to \(\mu\) is straightforward. We derive the expression for the gradient of \(f_{cm}\) with respect to \(\sigma\) next. Towards this, we first characterize the gradient of the trace function \(f(A) = \text{trace}((ASA)^{1/2})\) with \(A\) defined on the set of positive definite matrices.

**Proposition 5.** Function \(f : S^n_{++} \rightarrow \mathbb{R}\) is defined as \(f(A) = \text{trace}((ASA)^{1/2})\) where \(S \in S^n_{++}\). When the matrix \(S\) is positive definite, then the gradient of \(f\) at the point \(A\) is

\[
g(A) = \frac{1}{2} [A^{-1}(ASA)^{1/2} + (ASA)^{1/2} A^{-1}].
\]
Proof. Let $F(A) = (ASA)^{1/2}$, then $f(A) = \text{trace}(F(A))$. For a given symmetric matrix $D$,

$$F(A + tD) - F(A) = (ASA + E_D(t, A))^{1/2} - (ASA)^{1/2},$$

where $E_D(t, A) = t(DSA + ASD) + t^2 DSD$. Since both $A$ and $S$ are positive definite, the matrix $ASA$ is positive definite. Let $L_{1/2}(ASA, E_D(t, A))$ (or $L_{1/2}$ in short format) denote the Fréchet derivative for the matrix square root which is the unique solution to the Sylvester equation:

$$(ASA)^{1/2}L_{1/2} + L_{1/2}(ASA)^{1/2} = E_D(t, A). \quad (6.7)$$

By the definition of Fréchet derivative, we have

$$\|F(A + tD) - F(A) - L_{1/2}(ASA, E_D(t, A))\| = o(\|E_D(t, A)\|) = o(t).$$

Then

$$f(A + tD) - f(A) = \text{trace}(F(A + tD) - F(A))$$

$$= \text{trace}((ASA)^{-1/2}(ASA)^{1/2}[F(A + tD) - F(A)])$$

$$= \text{trace}((ASA)^{-1/2}(ASA)^{1/2}L_{1/2}) + o(t)$$

$$= \frac{1}{2}\text{trace}((ASA)^{-1/2}E_D(t, A)) + o(t)$$

$$= \frac{1}{2}\text{trace}((ASA)^{-1/2}[t(DSA + ASD) + t^2 DSD]) + o(t)$$

$$= \frac{1}{2}t \cdot \text{trace}(SA(ASA)^{-1/2} + (ASA)^{-1/2}AS) + o(t).$$

Hence the directional derivative of $f$ in the direction $D \in S^n$ is

$$\nabla_D f(A) = \lim_{t \to 0} \frac{1}{t}(f(A + tD) - f(A))$$

$$= \frac{1}{2}[SA(ASA)^{-1/2} + (ASA)^{-1/2}AS], D).$$

Therefore, the gradient of $f$ at point $A$ is

$$g(A) = \frac{1}{2}[SA(ASA)^{-1/2} + (ASA)^{-1/2}AS]$$

$$= \frac{1}{2}[A^{-1}(ASA)^{1/2} + (ASA)^{1/2}A^{-1}].$$

□

We next extend the result of Proposition 5 to a more general case in which the matrix $S$ is positive semidefinite but not necessarily positive definite.

**Proposition 6.** Function $f : S^n_+ \to \mathbb{R}$ is defined as $f(A) = \text{trace}((ASA)^{1/2})$ where $S \in S^n_+$. Then the gradient of $f$ at the point $A$ is

$$g(A) = \frac{1}{2}[A^{-1}(ASA)^{1/2} + (ASA)^{1/2}A^{-1}]. \quad (6.8)$$
Proof. Let \( f(\epsilon, A) = \text{trace}(A(S + \epsilon I)A^{1/2}) \), \( \epsilon \in (0, 1) \), then \( f(A) = \lim_{\epsilon \to 0} f(\epsilon, A) \). From Proposition 5 we know that the gradient of \( f(\epsilon, A) \) is

\[
g(\epsilon, A) = \frac{1}{2} [A^{-1}(A(S + \epsilon I)A^{1/2} + (A(S + \epsilon I)A^{1/2}A^{-1}].
\]

For a given symmetric matrix \( D \), there exists \( \delta > 0 \) such that \( A + tD \to 0 \) when \( t \in [-\delta, \delta] \). The directional derivative of \( f \) on the direction \( D \) is

\[
\lim_{t \to 0} \frac{1}{t} [f(A + tD) - f(A)] = \lim_{t \to 0} \lim_{\epsilon \to 0} \frac{1}{t} [f(\epsilon, A + tD) - f(\epsilon, A)]
\]

\[
= \lim_{\epsilon \to 0} \lim_{t \to 0} \frac{1}{t} [f(\epsilon, A + tD) - f(\epsilon, A)]
\]

\[
= \lim_{\epsilon \to 0} (g(\epsilon, A), D)
\]

\[
= \langle g(A), D \rangle.
\]

In the second equality, we change limits which we justify next. For given matrices \( A \) and \( D \), we define

\[
G(\epsilon, t) = \begin{cases} \frac{1}{t} [f(\epsilon, A + tD) - f(\epsilon, A)] & \text{if } t \neq 0, \\ \langle g(\epsilon, A), D \rangle & \text{if } t = 0 \end{cases}
\]

as a function of \( \epsilon \in (0, 1] \) and \( t \in [-\delta, \delta] \). To show that

\[
\lim_{t \to 0} \lim_{\epsilon \to 0} G(\epsilon, t) = \lim_{\epsilon \to 0} \lim_{t \to 0} G(\epsilon, t),
\]

a sufficient condition is (see Theorem 7.11 in (49)):

(a) For every \( \epsilon \in (0, 1] \) the finite limit \( \lim_{t \to 0} G(\epsilon, t) \) exists.

(b) For every \( t \in [-\delta, \delta] \), the finite limit \( \lim_{\epsilon \to 0} G(\epsilon, t) \) exists.

(c) As \( t \to 0 \), \( G(\epsilon, t) \) uniformly converges to a limit function for \( \epsilon \in (0, 1] \).

It is obvious that conditions (a) and (b) are true. A sufficient and necessary condition for (c) is (see Theorem 7.9 in (49)):

\[
\lim_{t \to 0} \sup_{\epsilon \in (0, 1]} |G(\epsilon, t) - G(\epsilon, 0)| = 0.
\]

(6.9)

Next, we prove the result of (6.9). By the mean value theorem and the proof of Proposition 5, there exists a \( t_1 \) between 0 and \( t \), such that \( G(\epsilon, t) = \langle (g(\epsilon, A + t_1D), D) \rangle \). Then

\[
G(\epsilon, t) - G(\epsilon, 0) = \langle (g(\epsilon, A + t_1D), D) \rangle - \langle (g(\epsilon, A), D) \rangle.
\]

(6.10)

For \( t \in (-\delta, \delta) \), define \( h(\epsilon, t) = \langle (g(\epsilon, A + tD), D) \rangle \). Let \( A_t = (A + tD), S_\epsilon = S + \epsilon I \), then

\[
\frac{\partial h(\epsilon, t)}{\partial t} = \lim_{t \to 0} \frac{1}{\Delta t} \text{trace} \left( D(A_t + \Delta t D)^{-1}[A_t + \Delta t D]S_\epsilon(A_t + \Delta t D)]^{1/2} - DA_t^{-1}[A_t S_\epsilon A_t]^{1/2} \right)
\]

\[
= \lim_{\Delta t \to 0} \frac{1}{\Delta t} \text{trace} \left( DA_t^{-1}(I - \Delta t DA_t^{-1} + o(\Delta t))[(A_t S_\epsilon A_t)^{1/2} + L_{1/2}(A_t S_\epsilon A_t, \Delta t(A_t S_\epsilon D + DS A_t)) + o(t)] \right)
\]

31
\[ -DA_t^{-1}(A_t S_t A_t)^{1/2}. \]

where \( L_{1/2}(A_t S_t A_t, t(A_t S_t D + DS_t A_t)) \) is the Fréchet derivative of the matrix square root. For simplicity we denote it by \( L_{1/2} \), and it satisfies

\[ (A_t S_t A_t)^{1/2} L_{1/2} + L_{1/2} (A_t S_t A_t)^{1/2} = \Delta t (A_t S_t D + DS_t A_t). \]

Let \( L = L_{1/2}/\Delta t \), then

\[ (A_t S_t A_t)^{1/2} L + L (A_t S_t A_t)^{1/2} = (A_t S_t D + DS_t A_t) \] (6.11)

and

\[
\frac{\partial h(\epsilon, t)}{\partial t} = \lim_{\Delta t \to 0} \frac{1}{\Delta t} \text{trace} \left( DA_t^{-1}(\Delta t L - \Delta t DA_t^{-1}(A_t S_t A_t)^{1/2}) \right) + o(\Delta t) \\
= \text{trace} (DA_t^{-1} L) - \text{trace} \left( (DA_t^{-1})^2 (A_t S_t A_t)^{1/2} \right). \] (6.12)

\( L \) is the solution of the Sylvester equation (6.11) which is unique, hence the partial derivative of \( h(\epsilon, t) \) with respect \( t \) exists. Next, we show that \( \frac{\partial h(\epsilon, t)}{\partial t} \) is bounded for \((\epsilon, t) \in (0, 1] \times (-\delta, \delta)\). We can find that the second item of (6.12) is well defined and continuous on a compact set \([0, 1] \times [-\delta, \delta]\), hence it is bounded on \((0, 1] \times (-\delta, \delta)\).

For the first item of (6.12), we know that

\[ |\text{trace} (DA_t^{-1} L)| \leq \frac{1}{2} \| DA_t^{-1} \|_F^2 + \frac{1}{2} \| L \|_F^2, \]

where \( \| \cdot \|_F \) denotes the Frobenius norm of a matrix. \( \| DA_t^{-1} \|_F^2 \) is continuous in \( t \) on the set \([-\delta, \delta]\), hence it is bounded. We only need to show that \( \| L \|_F \) is bounded. Actually, we can obtain the closed form of \( L \) by solving the Sylvester equation (6.11).

Let \( P^T A P \) be the eigenvalue decomposition of \( A_t S_t A_t \). Then (6.11) can be written as

\[ P^T \Lambda^{1/2} P L + LP^T \Lambda^{1/2} P = (A_t S_t A_t) A_t^{-1} D + DA_t^{-1} (A_t S_t A_t) = P^T \Lambda P A_t^{-1} D + DA_t^{-1} P^T A \]

Let \( \bar{L} = PLP^T \), we have

\[ \Lambda^{1/2} \bar{L} + \bar{L} \Lambda^{1/2} = \Lambda P A_t^{-1} D P^T + PDA_t^{-1} P^T \Lambda = \Lambda E + E^T \Lambda, \] (6.13)

where \( E = PA_t^{-1} D P^T \). The solution for equation (6.13) is

\[ \bar{L}_{ij} = \frac{\lambda_i E_{ij} + \lambda_j E_{ji}}{\lambda_i^{1/2} + \lambda_j^{1/2}}. \]

Therefore \( |\bar{L}_{ij}| \leq (\lambda_i^{1/2} + \lambda_j^{1/2}) \text{max}_{i,j} |E_{ij}| \). Notice that \( \| E \| \leq \| P \| \cdot \| A_t^{-1} D \| \cdot \| P^T \| \) is bounded on \((0, 1] \times [-\delta, \delta]\).

The eigenvalues of \( A_t (S + \epsilon I) A_t \) are no larger than the eigenvalues of \( A_t (S + I) A_t \) when \( \epsilon \in (0, 1] \). Since

32
\[ \| A_t(S + I)A_t \|_F \] is continuous on \([-\delta, \delta]\), it is bounded. Hence the eigenvalues of \( A_t(S + \epsilon I)A_t \) are bounded on \((0, \epsilon] \times [-\delta, \delta]\). Therefore \( \| \hat{L} \|_F \) is bound which implies that \( \| L \|_F \) is also bounded.

By the above discussion, we know that for all \( \epsilon \in (0, 1] \), and \( t \in (-\delta, \delta) \), there exists a constant \( M \) such that \( |\frac{\partial h(\epsilon, t)}{\partial \epsilon}| \leq M \). Therefore, for all \( \epsilon \in (0, 1] \), and \( t \in (-\delta, \delta) \), \( |h(\epsilon, t) - h(\epsilon, 0)| \leq M|t| \). Then by (6.10) and the definition of \( h(\epsilon, t) \),

\[
\begin{align*}
\lim_{t \to 0} \sup_{\epsilon \in (0, 1]} |G(\epsilon, t) - G(\epsilon, 0)| &= \lim_{t \to 0} \sup_{\epsilon \in (0, 1]} |h(\epsilon, t_1) - h(\epsilon, 0)| \\
&\leq \lim_{t \to 0} M|t| \\
&= 0.
\end{align*}
\]

We now provide the proof of the gradient of the function which helps complete the proof.

**Proposition 7.** Function \( V : \mathbb{R}_n^{++} \to \mathbb{R} \) is defined as

\[
V(\sigma) = \text{trace} \left( \left[ A\text{Diag}(\sigma)C\text{Diag}(\sigma)A^T \right]^{1/2} S \left[ A\text{Diag}(\sigma)C\text{Diag}(\sigma)A^T \right]^{1/2} \right),
\]

(6.14)

where \( A \in \mathbb{R}^{m \times n} \) is a matrix with full row rank, \( C \in \mathbb{S}_n^{++} \) is a given positive definite matrix, and \( S \in \mathbb{S}_n^+ \) is a given positive semidefinite matrix. Then the gradient of \( V \) is

\[
\text{grad}(\sigma) = \text{diag} \left( A^T h(\sigma)^{-1} (h(\sigma)Sh(\sigma))^{1/2} h(\sigma)^{-1} A\text{Diag}(\sigma)C \right),
\]

(6.15)

where \( h(\sigma) = (A\text{Diag}(\sigma)C\text{Diag}(\sigma)A^T)^{1/2} \).

**Proof.** Let \( L(\sigma, \cdot) \) be the Fréchet derivative of \( h \). Then for all unit vector \( v \in \mathbb{R}_n \) and \( t \in \mathbb{R} \),

\[
\| h(\sigma + tv) - h(\sigma) - L(\sigma, tv) \| = o(t).
\]

(6.16)

By simple calculation, we have

\[
h(\sigma + tv) - h(\sigma) = (h(\sigma) + E_v(t, \sigma))^{1/2} - h(\sigma),
\]

where \( E_v(t, \sigma) = tA[\text{Diag}(\sigma)C\text{Diag}(v) + \text{Diag}(v)C\text{Diag}(\sigma)]A^T + t^2 A\text{Diag}(v)C\text{Diag}(v)A^T \). Let \( L_{1/2} \) denote the Fréchet derivative of the matrix square root, then

\[
\| h(\sigma + tv) - h(\sigma) - L_{1/2}(h(\sigma)^2, E_v(t, \sigma)) \| = o(\| E_v(t, \sigma) \|) = o(t).
\]

(6.18)

By (6.16) and (6.18), we have

\[
\| L(\sigma, tv) - L_{1/2}(h(\sigma)^2, E_v(t, \sigma)) \| = o(t).
\]

(6.19)

From the Sylvester equation for the Fréchet derivative of matrix square root, we obtain

\[
h(\sigma)L(\sigma, tv) + L(\sigma, tv)h(\sigma) = tA[\text{Diag}(\sigma)C\text{Diag}(v) + \text{Diag}(v)C\text{Diag}(\sigma)]A^T + o(t).
\]

(6.20)
By the above equation we have \( \|L(\sigma, tv)\| = O(t) \).

Let \( f \) be the trace function \( f(A) = \text{trace}(ABA^{1/2}) \), and \( g \) be its gradient as in Proposition 6. Then \( V(\sigma) = f(h(\sigma)) \). By the mean value theorem, there exist an \( \alpha \in [0, 1] \) such that

\[
V(\sigma + tv) - V(\sigma) = f(h(\sigma + tv) - h(\sigma))
\]

\[
= (g(\alpha h(\sigma + tv) + (1 - \alpha)h(\sigma)), h(\sigma + tv) - h(\sigma))
\]

\[
= (g(\alpha h(\sigma + tv) + (1 - \alpha)h(\sigma)), L(\sigma, tv)) + o(t)
\]

Then the directional derivative of \( V \) at the direction \( v \) is

\[
\lim_{t \to 0} \frac{1}{t}(V(\sigma + tv) - V(\sigma)) = \lim_{t \to 0} \langle g(h(\sigma)), L(\sigma, tv) \rangle
\]

\[
= \lim_{t \to 0} \text{trace} \left( \frac{1}{2}[h(\sigma)^{-1}(h(\sigma)Sh(\sigma))^{1/2} + (h(\sigma)Sh(\sigma))^{1/2}h(\sigma)^{-1}]L(\sigma, tv) \right)
\]

\[
= \lim_{t \to 0} \text{trace} \left( h(\sigma)^{-1}(h(\sigma)Sh(\sigma))^{1/2}L(\sigma, tv) \right)
\]

\[
= \lim_{t \to 0} \text{trace} \left( B(\sigma)h(\sigma)L(\sigma, tv) \right),
\]

where \( B(\sigma) = h(\sigma)^{-1}(h(\sigma)Sh(\sigma))^{1/2}h(\sigma)^{-1} \) is a symmetric matrix. By (6.20), we know that

\[
\text{trace} \left( B(\sigma)h(\sigma)L(\sigma, tv) \right) = \frac{t}{2}\text{trace}(B(\sigma)A[\text{Diag}(\sigma)C\text{Diag}(v) + \text{Diag}(v)\text{Diag}(\sigma)]A^T) + o(t)
\]

\[
= t \cdot \text{trace}(B(\sigma)A\text{Diag}(\sigma)C\text{Diag}(v)A^T) + o(t).
\]

Then

\[
\lim_{t \to 0} \frac{1}{t}(V(\sigma + tv) - V(\sigma)) = \text{trace}(B(\sigma)A\text{Diag}(\sigma)C\text{Diag}(v)A^T)
\]

\[
= \langle \text{diag}(A^TB(\sigma)A\text{Diag}(\sigma)C), v \rangle.
\]

Hence, the gradient of \( V \) at \( \sigma \) is \( \text{diag}(A^TB(\sigma)A\text{Diag}(\sigma)C) \) and the result is proved. \( \square \)

Using the result of Proposition 7, the closed form (4.2) for the gradient of \( f_{cmm} \) with respect to \( \sigma \) can be easily obtained.

**References**


