Multiple Kernel Learning-Aided Column-and-Constraint Generation Method

Biao Han
Email: hanb16@tsinghua.org.cn

***Abstract*—Two-stage robust optimization (two-stage RO), due to its ability to balance robustness and flexibility, has been widely used in various fields for decision-making under uncertainty. This paper proposes a multiple kernel learning (MKL)-aided column-and-constraint generation (CCG) method to address this issue in the context of data-driven decision optimization, and releases a corresponding registered Julia package, `MKLTwoStageRO.jl`. The proposed method performs efficient multi-process parallel MKL on a large number of directional nullspace projection norm kernels in the uncertainty space, and with the help of one-class support vector machine, constructs a piecewise linear polyhedral intersection uncertainty set enjoying structural sparsity and computational tractability. The significant scenarios identified as boundary support vectors in the MKL phase are used to add valid initial cuts to the subsequent CCG procedure, so that the exact solution of the MKL uncertainty set-induced two-stage RO can be achieved with fewer iterations. Decision-makers are able to adjust the set construction efficiency, model complexity, and risk aversion degree through three hyperparameters in the proposed method. This method implies a connection with two-stage stochastic programming based on the value-at-risk measure, thus providing a quantitative evaluation of the solution quality. Numerical study on the data-driven two-stage robust location-transportation problem demonstrates the effectiveness and practicability of the proposed method and software package. The source code of `MKLTwoStageRO.jl` is available at https://github.com/hanb16/MKLTwoStageRO.jl.**

I. Introduction

Robust optimization (RO) is a methodology for formulating decision-making problems whose solutions are required to have immunization against parameter uncertainties. Its underlying logic differs from probability-based approaches such as stochastic programming that assume perfect knowledge of corresponding probability distributions. In RO, one seeks an optimal solution for the worst-case scenario of uncertain parameters within a prespecified *uncertainty set* [1], [2]. A wide range of RO problems are modeled as static RO or single-stage RO that only contains once-for-all "here-and-now" decisions, which may tend to be overly conservative. A more complicated but flexible RO model is two-stage RO (also known as adjustable or adaptive RO), which involves not only "here-and-now" decisions (first-stage decisions) made before the uncertainty materializes, but also "wait-and-see" decisions (second-stage decisions, or recourse decisions) made after the uncertainty is revealed. Because of the better balance between exploitation and exploration, two-stage RO has recieved substantial applications in various domains including finance [3], energy [4], logistics [5], healthcare [6], etc. In this paper, we focus on the two-stage RO formulation of the following general form

$$\min_{\mathbf{x}\in\mathcal{X}} \mathbf{a}^\top \mathbf{x} + \max_{\mathbf{u}\in\mathcal{U}} \min_{\mathbf{y}\in\mathcal{Y}(\mathbf{x},\mathbf{u})} \mathbf{b}^\top \mathbf{y} \tag{1}$$

with

$$\mathcal{Y}(\mathbf{x},\mathbf{u}) = \left\{\mathbf{y}\in\mathbb{R}^q \,|\mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{y}+\mathbf{C}\mathbf{u}\geq\mathbf{c}, \mathbf{y}\geq\mathbf{0}\right\}, \tag{2}$$

where $\mathbf{a}\in\mathbb{R}^p$, $\mathbf{b}\in\mathbb{R}^q$, $\mathbf{A}\in\mathbb{R}^{t\times p}$, $\mathbf{B}\in\mathbb{R}^{t\times q}$, $\mathbf{C}\in\mathbb{R}^{t\times r}$, $\mathbf{c}\in\mathbb{R}^t$ are the problem's parameters, $\mathbf{x}\in\mathcal{X}\subseteq\mathbb{R}^p$ is the first-stage decision variable which may have mixed integer components, $\mathbf{y}\in\mathbb{R}^q$ is the continuous second-stage decision variable, and $\mathbf{u}\in\mathbb{R}^r$ is the uncertain parameter which lies in a convex and compact uncertainty set $\mathcal{U}$.

The computational intractability of two-stage RO and the over-conservatism of its solution have usually been two major obstacles for scholars trying to overstep. In order to make the problem easier to solve, scholars in many cases restrict the recourse decision $\mathbf{y}$ to be in some simplified function classes such as affine functions suggested by Ben-Tal et al. [7]. And at the same time, people use some structured geometries to model the uncertainty set $\mathcal{U}$, thus giving rise to box uncertainty sets [8], ellipsoidal uncertainty sets [9] and uncertainty sets based on general norms [10], etc. However, these measures often inevitably lead to overly conservative optimization solutions. Intersections and unions of these basic geometries are then developed to capture actual uncertainties to reduce the conservativeness [10]–[12]. But obviously, due to the pre-set elementary structures, this is still unable to break through the dilemma of flexibility and tractability.

In recent years, the big data era has spawned a new paradigm known as data-driven robust decision-making [13]–[16]. Machine learning techniques have been developed to model and solve two-stage RO. In Ning and You [17], a nonparametric kernel density M-estimation method is used to learn the uncertainty set from data for multi-stage adaptive RO, a more general version of the two-stage case. Shang and You [18] construct the uncertainty set with support vector clustering and apply it to multi-stage stochastic model predictive control. In Han et al. [19], a multiple kernel learning (MKL)-based one-class support vector machine (SVM) with tailored basis kernels is proposed to construct the uncertainty set automatically. Wang et al. [20] learn the parameters of the uncertainty set using a stochastic augmented Lagrangian method to meet specific probability level of constraint satisfaction. Although their methods greatly reduced the redundant coverage of the set than traditional methods and obtained tractable robust counterparts, they could actually only achieve suboptimal solutions of (1), because they all adopted affine decision rules when dealing with multi-stage cases. Thus, as a result, the probabilistic guarantees given by them become less convincing for the evaluation of the solution quality. There is also some other related literature addressing the construction of uncertainty sets using machine learning techniques [21]–[24], but they mainly focus on static RO and few of them provide systematic integration with two-stage RO.

In the contex of exactly solving the two-stage RO (1), one of the classic approaches is the Benders-dual cutting plane algorithm, which iteratively solves a master problem and an adversarial sub-problem, while gradually adding cuts to the former according to the dual information of the latter [25]–[27]. Under the master-subproblem framework, another algorithm, termed as column-and-constraint generation (CCG), is proved to have much better scalability and becomes popular amoung scholars [28]. Many variants of CCG have been proposed to further improve its efficiency or broaden its application scope [29]–[32]. It's worth mentioning that the power of machine learning is also harnessed to the CCG algorithm to tackle two-stage RO. Bertsimas and Kim [33] train decision tree classifiers to predict high-quality strategies based on the instance set generated by CCG algorithm, which drastically speeds up the solution process of two-stage RO. Dumouchelle et al. [34] proposes a custom-designed neural network to estimate the value function of the second-stage problem, which expands the applicability of the two-stage framework to large-scale instances with integer decisions in both stages. However, the above methods only consider uncertainty sets in basic forms that are given in advance (for example, the budget uncertainty set), and provide no specific uncertainty set construction approach. As a matter of fact, uncertainty sets are an indispensable element that distinguishes RO from deterministic optimization. The friendliness of a RO algorithm will be compromised if it is separated from a proper uncertainty set construction procedure, and if the set constructed is not exact enough, even an exact solution algorithm will only yield inexact (overly conservative) decisions in practice. Some works have proposed possible general frameworks to handle two-stage RO systematically [35], [36], but they, including many of the aforementioned ones, are usually explicitly or implicitly based on the relatively complete recourse assumption, which can't deal with a wide range of two-stage RO instances where this assumption do not hold but they themselves are feasible.

This paper proposes an MKL-aided CCG method to cope with two-stage RO systematically in a data-driven manner, including the construction of the uncertainty set and the solution of the induced robust counterpart. In the uncertainty set construction phase, an efficient multi-process parallel MKL algorithm is introduced with a novel type of candidate basis kernel termed as directional nullspace projection norm kernel. This algorithm constructs a piecewise linear polyhedral intersection uncertainty set automatically from the historical data of the uncertainties, which has compact structure and concise expression. At the same time, it identifies a small number of boundary support vectors from the massive data to serve as the significant scenarios in the following CCG procedure. The robust counterpart of the two-stage RO can then be solved with a modified CCG algorithm and the significant scenarios identified in the former MKL phase provide useful initial cuts so that they have the potential to reduce the iteration. Our method implies a connection between two-stage RO and the value-at-risk measure-based two-stage stochastic programming, so it can provide a probabilistic quantitative evaluation on the solution quality. The proposed MKL-aided CCG method doesn't require the relatively complete recourse assumption, therefore it can theoretically cope with any feasible two-stage RO (1). The above workflow has been integrated into a Julia package named as `MKLTwoStageRO.jl` for better utilization and improvement. To our best knowledge, it may be the first registered open-source package that deals with data-driven two-stage RO systematically with the help of machine learning techniques in the Julia community.

The remainder of this paper is organized as follows. In Section II, we present the adopted unified MKL-based one-class SVM framework with the novel directional nullspace projection norm kernel. We then propose the efficient parallel HessianMKL algorithm to construct an intersection MKL uncertainty set. In Section III, based on the discussion of the basis of CCG method, we demonstrate the tractability of the intersection MKL uncertainty set and propose an MKL uncertainty set-induced CCG algorithm. Section IV briefly introduces the companion software package, `MKLTwoStageRO.jl`. Section V conducts numerical experiments on a data-driven two-stage robust location-transportation problem, followed by final concluding remarks.

II. Multiple Kernlel Learning-Based Uncertainty Set Construction

A. MKL-Based One-Class SVM

One-class SVM (OC-SVM) is a classic machine learning method for estimating the support of an unknown distribution from its historical observations [37]. It has been widely applied due to its simplicity, efficiency, and good generalization ability. The mechanism of OC-SVM is quite clear, which is to first map the data samples $\mathcal{D} = \{\mathbf{u}_i\}_{i=1}^N$ through a mapping function $\phi(\mathbf{u})$ into a high-dimensional reproducing kernel Hilbert space $\mathbb{H}$, and then in this space separate the data points from the origin as much as possible using a hyperplane $\left\{\phi(\mathbf{u}) \in \mathbb{H} \,\middle|\, \mathbf{w}^\top \phi(\mathbf{u}) - \rho = 0\right\}$. Given a kernel function $K(\mathbf{u}, \mathbf{v})$, the OC-SVM problem can be transformed into a tractable form using the kernel trick $K(\mathbf{u}, \mathbf{v}) = \phi(\mathbf{u})^\top \phi(\mathbf{v})$, even if we do not know what the specific mapping $\phi(\cdot)$ is.

Studies have shown that by introducing more degrees of freedom to the kernel function $K(\mathbf{u}, \mathbf{v})$, e.g., expressing it as a convex combination of multiple basis kernels:

$$K(\mathbf{u}, \mathbf{v}) = \sum_{m=1}^{M} \pi_m K_m(\mathbf{u}, \mathbf{v}) \text{ with } \pi_m \geq 0, \sum_{m=1}^{M} \pi_m = 1, \tag{3}$$

and learning an optimal kernel combination coefficient $\boldsymbol{\pi}$ simultaneously during training, the representation capability and predictive performance of SVM can be enhanced [38]–[40]. Han et al. [19] applied MKL techniques to OC-SVM and proposed the following unified MKL-based OC-SVM (MKL-OC-SVM) framework

$$\begin{aligned}\min_{\{\mathbf{w}_m\},\boldsymbol{\pi},\rho,\boldsymbol{\xi}} \quad & \frac{1}{2}\sum_{m=1}^{M}\frac{1}{\pi_m}\|\mathbf{w}_m\|^2-\rho+\frac{1}{N\nu}\sum_{n=1}^{N}\xi_n \\ \text{s.t.} \quad & \sum_{m=1}^{M}\mathbf{w}_m^\top\phi_m(\mathbf{u}_n)\geq\rho-\xi_n, \quad \forall n\in[N] \\ & \xi_n\geq 0, \quad \forall n\in[N] \\ & \sum_{m=1}^{M}\pi_m=1 \\ & 0\leq\pi_m\leq\tfrac{1}{M\mu}, \quad \forall m\in[M]\end{aligned} \tag{4}$$

with its neatly symmetrical dual form

$$\begin{aligned}\min_{\boldsymbol{\alpha},\gamma,\boldsymbol{\zeta}} \quad & -\gamma+\frac{1}{M\mu}\sum_{m=1}^{M}\zeta_m \\ \text{s.t.} \quad & \frac{1}{2}\sum_{n,n'=1}^{N}\alpha_n\alpha_{n'}K_m(\mathbf{u}_n,\mathbf{u}_{n'})\leq\zeta_m-\gamma, \ \forall m\in[M] \\ & \zeta_m\geq 0, \quad \forall m\in[M] \\ & \sum_{n=1}^{N}\alpha_n=1 \\ & 0\leq\alpha_n\leq\tfrac{1}{N\nu}, \quad \forall n\in[N].\end{aligned} \tag{5}$$

As suggested by Han et al. [19], the MKL-OC-SVM can be used to construct the data-driven uncertainty set for robust optimization. The idea is to train an MKL-OC-SVM based on the uncertainty data samples $\mathcal{D}=\{\mathbf{u}_i\}_{i=1}^{N}$ with plenty of candidate basis kernel functions $\{K_m(\cdot,\cdot)\}_{m=1}^{M}$ provided in advance. After training, a considerable number of data samples and candidate basis kernels will be discarded, leaving only a small amount of necessary support vectors (SVs) and support kernels (SKs) to form a relatively concise data-driven uncertainty set as follows:

$$\mathcal{U}_{\nu,\mu}(\mathcal{D})=\left\{\mathbf{u}\,\middle|\,\sum_{n\in SV}\sum_{m\in SK}\alpha_n^\star\pi_m^\star K_m(\mathbf{u},\mathbf{u}_n)\geq\rho^\star\right\}, \tag{6}$$

where SV and SK denote the index set of the SVs and the SKs respectively, and $\{\alpha^\star\}$, $\{\pi^\star\}$ and $\rho^\star$ are the optimal solutions that solve (4) and (5).

Note that the candidate basis kernels need to be carefully selected rather than arbitrarily taken from the popular classical kernels such as Gaussian kernel, polynomial kernel, etc., so as to ensure the computational tractability of the two-stage RO (1) induced by the MKL-based uncertainty set (6).

B. Directional Nullspace Projection Norm Kernel

Han et al. [19] proposed a *directional projection distance kernel* (DPDK) to serve as the basis kernels in MKL, i.e.,

$$K_m(\mathbf{u},\mathbf{v})=1-\frac{1}{\kappa c_m}\left|\mathbf{q}_m^\top(\mathbf{u}-\mathbf{v})\right|, \tag{7}$$

which enables the uncertainty set constructed through MKL to have a convex polyhedral structure, thereby facilitating the solution of the induced robust optimization. This kernel essentially projects data points in the direction of $\mathbf{q}_m$, and measures a kind of *negative distance* between each pair of projection points.

As a substitute, a novel kernel function termed as *directional nullspace projection norm kernel* (DNPNK) is presented in this paper. Similar to DPDK, it also requires generating unit direction vectors $\{\mathbf{q}_m\}$ in the uncertainty space in advance. The difference lies in that, for a given $\mathbf{q}_m$, DNPNK does not project the data points towards this direction, but towards its orthogonal complement space $\mathbb{Q}_m$, and measures a kind of *negative norm* between each pair of projection points. If we consider $\mathbf{q}_m^\top$ as a map, then $\mathbb{Q}_m$ is also known as the *nullspace* or *kernel* of $\mathbf{q}_m^\top$, i.e., $\mathbb{Q}_m=(\text{span}\{\mathbf{q}_m\})^\perp=\text{null}(\mathbf{q}_m^\top)=\ker(\mathbf{q}_m^\top)$. Now we assume that the column vectors of the matrix $\mathbf{Q}_m\in\mathbb{R}^{r\times(r-1)}$ are composed of a set of standard orthonormal basis $\{\mathbf{q}_m^{(1)},\mathbf{q}_m^{(2)},\cdots,\mathbf{q}_m^{(r-1)}\}$ of $\mathbb{Q}_m$, then the expression of DNPNK is:

$$K_m(\mathbf{u},\mathbf{v})=1-\frac{1}{\kappa c_m}\left\|\mathbf{Q}_m^\top(\mathbf{u}-\mathbf{v})\right\|_p, \tag{8}$$

where $p\geq 1$ so that $\|\cdot\|_p$ is a legitimate ℓ_p-norm, and the kernel parameters c_m and κ can be selected as

$$c_m=\max_{\mathbf{u},\mathbf{v}\in\mathcal{D}}\left\|\mathbf{Q}_m^\top(\mathbf{u}-\mathbf{v})\right\|_p, \quad \kappa>1 \tag{9}$$

according to the following proposition.

Proposition 1 (Positive Defineteness of DNPNK): If the parameters c_m and κ are set according to (9), then every basis kernel matrix $\mathbf{K}_m$ with elements $K_m(\mathbf{u},\mathbf{v}),\forall\mathbf{u},\mathbf{v}\in\mathcal{D},m\in[M]$ induced by DNPNK (8) is positive definite.

Proof: For every pair of $\mathbf{u},\mathbf{v}\in\mathcal{D}\subset\mathbb{R}^r$, we denote by $\hat{\mathbf{u}}=\frac{1}{\kappa c_m}\mathbf{Q}_m^\top\mathbf{u}$ and $\hat{\mathbf{v}}=\frac{1}{\kappa c_m}\mathbf{Q}_m^\top\mathbf{v}$ their corresponding scaled projections on $\mathbb{Q}_m$, and by $\mathbf{c}\in\mathbb{Q}_m$ the midpoint of the line segment connecting $\hat{\mathbf{u}}$ and $\hat{\mathbf{v}}$, i.e., $\mathbf{c}=\frac{1}{2}(\hat{\mathbf{u}}+\hat{\mathbf{v}})$. Now if we make an ℓ_p-norm ball $\mathcal{B}$ with radius $\frac{1}{2}$ centered on $\mathbf{c}$, i.e., $\mathcal{B}=\left\{\mathbf{z}\in\mathbb{Q}_m\,\middle|\,\|\mathbf{z}-\mathbf{c}\|_p\leq\frac{1}{2}\right\}$, it will be easy to prove that $\hat{\mathbf{u}},\hat{\mathbf{v}}\in\text{relint}\mathcal{B},\forall\mathbf{u},\mathbf{v}\in\mathcal{D}$, where $\text{relint}\mathcal{B}$ is the relative interior of $\mathcal{B}$, because $\|\hat{\mathbf{u}}-\mathbf{c}\|_p=\frac{1}{2}\|\hat{\mathbf{u}}-\hat{\mathbf{v}}\|_p=\frac{1}{2\kappa c_m}\left\|\mathbf{Q}_m^\top(\mathbf{u}-\mathbf{v})\right\|_p\leq\frac{1}{2\kappa c_m}\max_{\mathbf{u}',\mathbf{v}'\in\mathcal{D}}\left\|\mathbf{Q}_m^\top(\mathbf{u}'-\mathbf{v}')\right\|_p=\frac{1}{2\kappa}<\frac{1}{2}$ and the same is true for $\hat{\mathbf{v}}$. We further denote by $\{\overline{\mathbf{z}},\underline{\mathbf{z}}\}=\text{aff}\{\hat{\mathbf{u}},\hat{\mathbf{v}}\}\cap\text{relbd}\mathcal{B}$ the intersection set of the affine hull of $\{\hat{\mathbf{u}},\hat{\mathbf{v}}\}$ and the relative boundary of $\mathcal{B}$, then it is not hard to see that DNPNK (8) can be decomposed as $K_m(\mathbf{u},\mathbf{v})=1-\|\hat{\mathbf{u}}-\hat{\mathbf{v}}\|_p=\|\overline{\mathbf{z}}-\underline{\mathbf{z}}\|_p-\|\hat{\mathbf{u}}-\hat{\mathbf{v}}\|_p=\min\{\|\overline{\mathbf{z}}-\hat{\mathbf{u}}\|_p,\|\overline{\mathbf{z}}-\hat{\mathbf{v}}\|_p\}+\min\{\|\underline{\mathbf{z}}-\hat{\mathbf{u}}\|_p,\|\underline{\mathbf{z}}-\hat{\mathbf{v}}\|_p\}=\overline{K}_m(\mathbf{u},\mathbf{v})+\underline{K}_m(\mathbf{u},\mathbf{v})$, where the two minimizations are denoted by $\overline{K}_m(\mathbf{u},\mathbf{v})$ and $\underline{K}_m(\mathbf{u},\mathbf{v})$ respectively. Let $w=\|\overline{\mathbf{z}}-\hat{\mathbf{u}}\|_p>0$ and $w'=\|\overline{\mathbf{z}}-\hat{\mathbf{v}}\|_p>0$, then it follows that $\overline{K}_m(\mathbf{u},\mathbf{v})=\min\{w,w'\}$ is a conventional intersection kernel on one dimensional variables and hence the kernel matrix $\overline{\mathbf{K}}_m$ generated from data set $\mathcal{D}$ satisfies positive defineteness [41]–[43]. The positive definiteness of $\underline{\mathbf{K}}_m$ can be established in a similar fashion. Therefore, $\mathbf{K}_m=\overline{\mathbf{K}}_m+\underline{\mathbf{K}}_m$ is positive definite. ■

In this paper, we take $p = 1$ to achieve better computational tractability of the induced two-stage robust optimization. Therefore, by substituting (8) into (6), we can obtain the MKL uncertainty set based on DNPNKs:

$$\begin{aligned}
&\mathcal{U}_{\nu,\mu}(\mathcal{D}) \\
=&\left\{\mathbf{u}\,\middle|\,\sum_{n\in SV}\sum_{m\in SK}\frac{\alpha_n^\star\pi_m^\star}{\kappa c_m}\left\|\mathbf{Q}_m^\top(\mathbf{u}-\mathbf{u}_n)\right\|_1\le 1-\rho^\star\right\} \\
=&\left\{\mathbf{u}\,\middle|\,\sum_{n\in SV}\sum_{m\in SK}\sum_{k=1}^{r-1}\frac{\alpha_n^\star\pi_m^\star}{\kappa c_m}\left|(\mathbf{q}_m^{(k)})^\top(\mathbf{u}-\mathbf{u}_n)\right|\le 1-\rho^\star\right\}.
\end{aligned} \tag{10}$$

Note that, when the dimension of the uncertainty space $r = 2$, DNPNK and the induced MKL uncertainty set will degenerate into an isomorphism of DPDK and its corresponding set. We also point out that, the proposed DNPNK-based uncertainty set (10) (which has a three-layer summation structure on the absolute value term) not only characterizes the set through SVs and SKs like the DPDK-based one (which has a two-layer summation structure on the absolute value term, i.e., the SV-layer and SK-layer, see [19, (26)]), but also characterizes the set through the dimension of the data space itself like the WGIK-based one proposed by Shang et al. [23] (which only has one dimension-layer of summation structure on the absolute value term, see [19, (25)]). In a nutshell, DNPNK incorporates and generalizes both DPDK and WGIK.

C. Parallel Learning Algorithm

Instead of solving the dual problem (5) directly, Han et al. [19] extended HessianMKL algorithm [44] to MKL-OC-SVM formulation (4) and proposed to alternatively optimize the following constrained optimization using Newton's method

$$\begin{aligned}
\min_{\boldsymbol{\pi}}\quad & J(\boldsymbol{\pi}) \\
\text{s.t.}\quad & \sum_{m=1}^{M}\pi_m = 1 \\
& 0\le \pi_m\le \tfrac{1}{M\mu},\quad \forall m\in[M]
\end{aligned} \tag{11}$$

and solve the following single kernel OC-SVM problem with given $\hat{\boldsymbol{\pi}}$ (i.e., the kernel is $\hat{\mathbf{K}} = \sum_m \hat{\pi}_m\mathbf{K}_m$) using any existing SVM solver

$$J(\boldsymbol{\pi}) = \begin{cases}
\displaystyle\min_{\{\mathbf{w}_m\},\rho,\boldsymbol{\xi}} & \displaystyle\frac{1}{2}\sum_{m=1}^{M}\frac{1}{\pi_m}\|\mathbf{w}_m\|^2-\rho+\frac{1}{N\nu}\sum_{n=1}^{N}\xi_n \\
\text{s.t.} & \displaystyle\sum_{m=1}^{M}\mathbf{w}_m^\top\phi_m(\mathbf{u}_n)\ge\rho-\xi_n,\ \forall n\in[N] \\
& \xi_n\ge 0,\quad \forall n\in[N].
\end{cases} \tag{12}$$

Although this algorithm also cannot escape the *curse of dimensionality*, it has been proven to be much more efficient than directly solving (5) or using the first-order SimpleMKL algorithm [38], and has been well applied in many practical scenarios such as [45]. In order to further reduce the time cost of MKL and broaden the application range of MKL-OC-SVM, this paper proposes to accelerate the learning process using parallel computing technology.

The idea of parallel learning is simple. If the number M of candidate basis kernels $\{\mathbf{K}_m\}_{m=1}^M$ is large, we can first group all kernels into B batches, and then use the HessianMKL algorithm to learn each batch $b\in[B]$ in parallel, thereby obtaining B MKL sets with shorter time overhead. Finally, we take the intersection of the sets as the final uncertainty set. The basis kernels can be grouped in one of the following ways:

- *Randomly*. Randomly draw l indices from the index set $[M] = \{1, 2, \cdots, M\}$ without replacement to form the index set $I^{(b)}$ of the bth batch of kernels for $b\in[B-1]$, and assign the remaining l' indices to $I^{(B)}$, where $l = \left\lfloor\frac{M}{B}\right\rfloor$ and $l' = M - l(B-1)$.
- *Sequentially*. Let $I^{(b)} = \{(b-1)l+k\,|k\in[l]\}, \forall b\in[B-1]$ and $I^{(B)} = \{(B-1)l+k\,|k\in[l']\}$.
- *Systematacially*. Let $I^{(b)} = \{b+kl\in[M]\,|k\in\mathbb{N}\}, \forall b\in[B]$.

To implement this idea, the original HessianMKL algorithm for MKL-OC-SVM [19] does not need to be modified much. For the sake of self-containedness, here we list several necessary formulas within it. The gradient $\mathbf{g}$ of $J(\boldsymbol{\pi})$ after every time (12) is sovled is given by

$$g_m = \frac{\partial J}{\partial\pi_m} = -\frac{1}{2}(\hat{\boldsymbol{\alpha}}_{SV})^\top\mathbf{K}_m^{(SV,SV)}\hat{\boldsymbol{\alpha}}_{SV},\quad \forall m, \tag{13}$$

where we denote by $\hat{\boldsymbol{\alpha}}_{SV}$ the fragment of $\hat{\boldsymbol{\alpha}}$ (the current optimal dual variable that solves (12)) that is composed of the elements corresponding to SVs, and by $\mathbf{K}_m^{(SV,SV)}$ the submatrix of $\mathbf{K}_m$ that is composed of the rows and columns corresponding to SVs. The Hessian $\mathbf{H}$ is given by

$$\begin{aligned}
H_{mm'} =& \frac{\partial^2 J}{\partial\pi_m\partial\pi_{m'}} \\
=&(\hat{\boldsymbol{\alpha}}_{SV})^\top\mathbf{K}_m^{(SV,BSV)}\mathbf{A}\mathbf{K}_{m'}^{(BSV,SV)}\hat{\boldsymbol{\alpha}}_{SV},\ \forall m,m',
\end{aligned} \tag{14}$$

where $\mathbf{A}$ is the submatrix of the following matrix with the last row and last column removed:

$$\begin{bmatrix}\hat{\mathbf{K}}^{(BSV,BSV)} & \mathbf{1} \\ \mathbf{1}^\top & 0\end{bmatrix}^{-1}.$$

The Newton step $\mathbf{d}$ for updating $\boldsymbol{\pi}$ when optimizing (11) can then be obtained by solving the following quadratic program (QP):

$$\begin{aligned}
\min_{\mathbf{d}}\quad & \tfrac{1}{2}\mathbf{d}^\top\mathbf{H}\mathbf{d}+\mathbf{g}^\top\mathbf{d} \\
\text{s.t.}\quad & \mathbf{1}^\top\mathbf{d} = 0 \\
& \mathbf{0}\le\hat{\boldsymbol{\pi}}+\mathbf{d}\le\tfrac{1}{M\mu}\mathbf{1},
\end{aligned} \tag{15}$$

where $\hat{\boldsymbol{\pi}}$ is the current value of $\boldsymbol{\pi}$. When the globally optimal $\boldsymbol{\alpha}^\star$ and $\boldsymbol{\pi}^\star$ are achieved, a robust $\rho^\star$ can be obtained by averaging according to the following equation:

$$\mathbf{K}^{\star(BSV,\cdot)}\boldsymbol{\alpha}^\star-\rho^\star\mathbf{1} = \mathbf{0}. \tag{16}$$

Now the parallel HessianMKL algorithm for MKL-OC-SVM can be presented as Algorithm 1.

Based on DNPNKs, the output of Algorithm 1 can form B polyhedral uncertainty sets $\{\mathcal{U}_{\nu,\mu}^{(b)}(\mathcal{D})\}_{b=1}^B$, each of which

Algorithm 1 Parallel HessianMKL for MKL-OC-SVM

```
Input: $\{\mathbf{K}_m\}_{m=1}^M$, $\nu$, $\mu$, $B$, $\epsilon > 0$
Output: $\{\boldsymbol{\alpha}^{(b)}, \boldsymbol{\pi}^{(b)}, \rho^{(b)}\}_{b=1}^B$
1: Group the $M$ kernel matrices $\{\mathbf{K}_m\}_{m=1}^M$ into $B$ batches with $M^{(b)}$ the batch size, and renumber the kernel matrices within each batch as $\{\mathbf{K}_m^{(b)}\}_{m=1}^{M^{(b)}}, \forall b \in [B]$.
2: parallel for $b = 1$ to $B$ do
3:     Initialize:
         $\boldsymbol{\alpha}^{(b)} \leftarrow \frac{1}{N}\mathbf{1}$,
         $\boldsymbol{\pi}^{(b)} \leftarrow \frac{1}{M^{(b)}}\mathbf{1}$,
         $\mathbf{K}^{(b)} \leftarrow \sum_{m=1}^{M^{(b)}} \pi_m^{(b)} \mathbf{K}_m^{(b)}$,
         $\Delta J^{(b)} \leftarrow -\infty$,
         $\tilde{J}^{(b)} \leftarrow (\boldsymbol{\alpha}^{(b)})^\top \mathbf{K}^{(b)} \boldsymbol{\alpha}^{(b)}$.
4:     while $\left|\Delta J^{(b)}\right| > \epsilon \cdot \tilde{J}^{(b)}$ do
5:         Update $\boldsymbol{\alpha}^{(b)}$ by solving (12) using single kernel OC-SVM solver with kernel matrix $\mathbf{K}^{(b)}$ and regularization parameter $\nu$.
6:         Compute the gradient $\mathbf{g}^{(b)}$ according to (13) and the Hessian $\mathbf{H}^{(b)}$ according to (14).
7:         Obtain the current Newton step $\mathbf{d}^{(b)}$ by solving (15) using QP solver with batch size $M^{(b)}$ and regularization parameter $\mu$.
8:         Choose a proper step size $\tau^{(b)}$ by backtracking line search.
9:         Update:
             $\boldsymbol{\pi}^{(b)} \leftarrow \boldsymbol{\pi}^{(b)} + \tau^{(b)} \cdot \mathbf{d}^{(b)}$,
             $\mathbf{K}^{(b)} \leftarrow \sum_{m=1}^{M^{(b)}} \pi_m^{(b)} \mathbf{K}_m^{(b)}$,
             $\Delta J^{(b)} \leftarrow (\boldsymbol{\alpha}^{(b)})^\top \mathbf{K}^{(b)} \boldsymbol{\alpha}^{(b)} - \tilde{J}^{(b)}$,
             $\tilde{J}^{(b)} \leftarrow (\boldsymbol{\alpha}^{(b)})^\top \mathbf{K}^{(b)} \boldsymbol{\alpha}^{(b)}$.
10:    end while
11:    Compute $\rho^{(b)}$ according to (16).
12: end parallel for
13: return $\{\boldsymbol{\alpha}^{(b)}, \boldsymbol{\pi}^{(b)}, \rho^{(b)}\}_{b=1}^B$
```

is in the form of (10). They together derive the following intersection MKL uncertainty set

$$\mathcal{U}_{\nu,\mu}^*(\mathcal{D}) = \bigcap_{b=1}^{B} \mathcal{U}_{\nu,\mu}^{(b)}(\mathcal{D}) \tag{17}$$

$$= \left\{ \mathbf{u} \,\middle|\, \begin{array}{l} \sum_{n \in SV^{(b)}} \sum_{m \in SK^{(b)}} \sum_{k=1}^{r-1} \frac{\alpha_n^{(b)\star} \pi_m^{(b)\star}}{\kappa c_m} \left| (\mathbf{q}_m^{(b)(k)})^\top (\mathbf{u} - \mathbf{u}_n) \right| \\ \qquad \leq 1 - \rho^{(b)\star}, \quad \forall b \in [B] \end{array} \right\}. \tag{18}$$

It is quite obvious that if the batch number $B = 1$, Algorithm 1 will be reduced to the original HessianMKL algorithm for MKL-OC-SVM [19] and the intersection MKL uncertainty set (17) will become (6).

We can extend the concepts of *boundary support vectors* (BSVs) and *outliers* (OLs) in [19] to the intersection MKL uncertainty set (18) by naturally defining

$$\mathcal{BSV} = \mathcal{D} \cap \text{relbd}\,\mathcal{U}_{\nu,\mu}^*(\mathcal{D}) = \bigcap_{b=1}^{B} \left\{ \mathbf{u}_n \in \mathcal{D} \,\middle|\, n \in BSV^{(b)} \right\} \tag{19}$$

as the *point set* of BSVs and

$$\mathcal{OL} = \mathcal{D} \setminus \mathcal{U}_{\nu,\mu}^*(\mathcal{D}) = \bigcup_{b=1}^{B} \left\{ \mathbf{u}_n \in \mathcal{D} \,\middle|\, n \in OL^{(b)} \right\} \tag{20}$$

as the *point set* of OLs, where $BSV^{(b)}$ and $OL^{(b)}$ stand for the *index sets* of the BSVs and the OLs generated from the batch b MKL, and their definitions strictly follow [19]. We have the following proposition about the outliers of the intersection MKL set.

Proposition 2 (Relationship Between ν, B and the Proportion of Outliers): If an intersection MKL set (17) is derived by Algorithm 1 with given $\nu \in (0, 1]$, $B \geq 1$ then $B\nu$ is an upper bound on the proportion of outliers defined by (20).

Proof: Firstly, the latter equation of (20) is obtained by substituting (17) and performing simple set operations. Then, based on the inclusion-exclusion principle, the following inequality about the cardinality of finite point set $\mathcal{OL}$ can be established: $\text{card}\mathcal{OL} = \text{card}\bigcup_{b=1}^{B}\left\{\mathbf{u}_n \in \mathcal{D} \,\middle|\, n \in OL^{(b)}\right\} \leq \sum_{b=1}^{B} \text{card}\left\{\mathbf{u}_n \in \mathcal{D} \,\middle|\, n \in OL^{(b)}\right\} = \sum_{b=1}^{B} \text{card}OL^{(b)}$. According to [19, Proposition 2], $\text{card}OL^{(b)}$ has an upper bound $N\nu$, hence we have $\text{card}\mathcal{OL} \leq \sum_{b=1}^{B} N\nu = NB\nu$. Therefore, $\text{card}\mathcal{OL}/N \leq B\nu$, and the proposition is proven. ■

Proposition 2 provides a theoretical upper bound for the proportion of outliers prior to learning the uncertainty set. This upper bound equals to the regularization parameter ν when $B = 1$ while it may be very loose as B becomes large. Nevertheless, numerical case study shows that the actual proportion of outliers posterior to the construction of the uncertaity set does not increase linearly with respect to B as $B\nu$ does, and it does not overstep ν too dramatically within reasonable ranges of B and ν, if we choose the directions $\{\mathbf{q}_m\}$ of all basis kernels evenly amoung the uncertaity space and group them systemeticially into batches (cf. Section V-C). Therefore, in practice when we use Algorithm 1 it still makes sense to take the value of ν as an important *a priori* reference for the proportion of outliers. The useful meaning of the proportion of outliers under the proposed MKL-aided CCG framework will be discussed later.

III. MKL Uncertainty Set-Induced Column-and-Constraint Generation Method

A. CCG Method Description

The CCG method proposed by Zeng and Zhao [28] is a general iterative framework to solve two-stage RO (1) exactly, which is easy to implement and has good scalability. It alternates between solving the following master problem (MP) with variables and constraints generated by a gradually enriched finite significant scenario set $\mathcal{S}$ to find a lower bound LB on the objective value and the corresponding first-stage

decision $\mathbf{x}$:
[MP]:

$$P(\mathcal{S}) = \begin{cases} \min\limits_{\mathbf{x},\eta,\{\mathbf{y}^{(\mathbf{s})}\}_{\mathbf{s}\in\mathcal{S}}} & \mathbf{a}^\top\mathbf{x} + \eta \\ \quad \text{s.t.} & \mathbf{x} \in \mathcal{X} \\ & \eta \geq \mathbf{b}^\top\mathbf{y}^{(\mathbf{s})}, \quad \forall \mathbf{s} \in \mathcal{S} \\ & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y}^{(\mathbf{s})} + \mathbf{C}\mathbf{s} \geq \mathbf{c}, \quad \forall \mathbf{s} \in \mathcal{S} \\ & \mathbf{y}^{(\mathbf{s})} \geq \mathbf{0}, \quad \forall \mathbf{s} \in \mathcal{S}, \end{cases} \tag{21}$$

and solving the following sub-problem (SP) with given $\mathbf{x}$ using proper oracle to derive an upper bound UB for the overall objective and identify some new significant scenario for $\mathbf{u}$ that can then be included in the set $\mathcal{S}$:
[SP]:

$$Q(\mathbf{x}) = \begin{cases} \max\limits_{\mathbf{u}\in\mathcal{U}} \min\limits_{\mathbf{y}} & \mathbf{b}^\top\mathbf{y} \\ \quad \text{s.t.} & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} + \mathbf{C}\mathbf{u} \geq \mathbf{c} \\ & \mathbf{y} \geq \mathbf{0}. \end{cases} \tag{22}$$

By making use of the Karush-Kuhn-Tucker (KKT) conditions, Zeng and Zhao [28] provided a general *optimality oracle* for SP (22), which can be converted into a 0-1 mixed integer program with the Big-M method. However, this oracle is rooted in the *relatively complete recourse assumption* (RCRA) that $\mathcal{Y}(\mathbf{x}, \mathbf{u}) \neq \varnothing$ for any $\mathbf{x} \in \mathcal{X}$ and any $\mathbf{u} \in \mathcal{U}$, which is so restrictive that if the oracle is applied to cases where the RCRA does not hold, the CCG procedure might not converge in a finite number of steps or might converge to an unrobust solution with the corresponding objective value more aggressive (less) than the true optimal one, even though the two-stage RO problem being solved is itself feasible. Bertsimas and Shtern [30] rigorously clarified this issue and extended the CCG method to a more general *feasibility assumption* (FA), i.e., there exists at least one $\mathbf{x} \in \mathcal{X}$ such that for any $\mathbf{u} \in \mathcal{U}$ the feasible region of the recourse problem $\mathcal{Y}(\mathbf{x}, \mathbf{u}) \neq \varnothing$. They proposed to determine the feasibility of the given $\mathbf{x}$ before solving (22) at each iteration by checking if the following optimization problem has a strictly positive objective value:

$$F(\mathbf{x}) = \begin{cases} \max\limits_{\mathbf{u}\in\mathcal{U}} \min\limits_{\mathbf{y},\delta} & \delta \\ \quad \text{s.t.} & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} + \mathbf{C}\mathbf{u} + \delta\mathbf{1} \geq \mathbf{c} \\ & \mathbf{y} \geq \mathbf{0}, \quad \delta \geq 0. \end{cases} \tag{23}$$

For a given $\mathbf{x}$, if the optimal value $F(\mathbf{x}) > 0$ (i.e., the given $\mathbf{x}$ is infeasible), the upper bound UB of (1) will be set to $+\infty$ by convention and the corresponding $\mathbf{u}$ will be incorperated into the significant scenario set $\mathcal{S}$, otherwise the optimality oracle will be called as before. Since (23) naturally satisfies the RCRA and has a structure similar to (22), it can also be equivalently transformed into a mixed integer program using the KKT conditions and the Big-M method, which is referred to as the *feasibility oracle*. Wang and Zeng [31] also adopted a similar feasibility oracle when handling two-stage RO with mixed integer recourse.

In this paper, we propose to embed the MKL-based uncertianty set into the extended CCG framework discussed above, thus providing an integrated and practical workflow for modeling and solving data-driven two-stage RO problems of the form (1) that satisfy the FA (but not necessarily the RCRA).

B. Tractability of the Intersection MKL Uncertainty Set

Similar to the DPDK-based MKL uncertainty set [19], the proposed DNPNK-based MKL uncertainty set (10) and the intersection version (18) also have benign computational tractability. By introducing auxiliary variables $\theta_{nmk}, \forall n \in SV, m \in SK, k \in [r-1]$, (10) can be linearized as

$$\tilde{\mathcal{U}}_{\nu,\mu}(\mathcal{D}) = \left\{ \mathbf{u} \;\middle|\; \begin{array}{l} \displaystyle\sum_{n\in SV}\sum_{m\in SK}\sum_{k=1}^{r-1} \frac{\alpha_n^\star \pi_m^\star}{\kappa c_m}\theta_{nmk} \leq 1 - \rho^\star, \\ -\,\theta_{nmk} \leq (\mathbf{q}_m^{(k)})^\top(\mathbf{u} - \mathbf{u}_n) \leq \theta_{nmk}, \\ \qquad \forall n \in SV, m \in SK, k \in [r-1] \end{array} \right\}, \tag{24}$$

and consequently (18) becomes

$$\begin{aligned} &\tilde{\mathcal{U}}^*_{\nu,\mu}(\mathcal{D}) = \bigcap_{b=1}^{B} \tilde{\mathcal{U}}^{(b)}_{\nu,\mu}(\mathcal{D}) \\ &= \left\{ \mathbf{u} \;\middle|\; \begin{array}{l} \displaystyle\sum_{n\in SV^{(b)}}\sum_{m\in SK^{(b)}}\sum_{k=1}^{r-1} \frac{\alpha_n^{(b)\star} \pi_m^{(b)\star}}{\kappa c_m}\theta^{(b)}_{nmk} \leq 1 - \rho^{(b)\star}, \\ -\,\theta^{(b)}_{nmk} \leq (\mathbf{q}_m^{(b)(k)})^\top(\mathbf{u} - \mathbf{u}_n) \leq \theta^{(b)}_{nmk}, \\ \qquad \forall n \in SV^{(b)}, m \in SK^{(b)}, k \in [r-1], b \in [B] \end{array} \right\}. \end{aligned} \tag{25}$$

Based on the linearized intersection MKL uncertainty set (25), by deriving the robust counterparts of (22) and (23) using the KKT conditions and the Big-M method, the optimality and feasibility oracles mentioned in the previous subsection are presented as
[Optimality Oracle]:

$$\tilde{Q}(\mathbf{x}) = \begin{cases} \max\limits_{\substack{\mathbf{u}\in\mathbb{R}^r, \mathbf{y}\in\mathbb{R}^q_+, \boldsymbol{\lambda}\in\mathbb{R}^t_+, \\ \mathbf{v}\in\{0,1\}^t, \mathbf{w}\in\{0,1\}^q}} & \mathbf{b}^\top\mathbf{y} \\ \quad \text{s.t.} & \mathbf{u} \in \tilde{\mathcal{U}}^*_{\nu,\mu}(\mathcal{D}) \\ & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} + \mathbf{C}\mathbf{u} \geq \mathbf{c} \\ & \mathbf{B}^\top\boldsymbol{\lambda} \leq \mathbf{b} \\ & \boldsymbol{\lambda} \leq \bar{M}\mathbf{v} \\ & \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{y} + \mathbf{C}\mathbf{u} - \mathbf{c} \leq \bar{M}(\mathbf{1} - \mathbf{v}) \\ & \mathbf{y} \leq \bar{M}\mathbf{w} \\ & \mathbf{b} - \mathbf{B}^\top\boldsymbol{\lambda} \leq \bar{M}(\mathbf{1} - \mathbf{w}) \end{cases} \tag{26}$$

and

[Feasibility Oracle]:

$$\tilde{F}(\mathbf{x}) = \begin{cases} \max\limits_{\substack{\mathbf{u}\in\mathbb{R}^r,(\mathbf{y},\delta)\in\mathbb{R}_+^{q+1},\boldsymbol{\lambda}\in\mathbb{R}_+^t,\\ \mathbf{v}\in\{0,1\}^t,\mathbf{w}\in\{0,1\}^{q+1}}} \quad \delta \\ \text{s.t.} \quad \mathbf{u}\in\tilde{\mathcal{U}}^*_{\nu,\mu}(\mathcal{D}) \\ \mathbf{Ax}+\mathbf{By}+\mathbf{Cu}+\delta\mathbf{1}\geq\mathbf{c} \\ [\mathbf{B}\quad\mathbf{1}]^\top\boldsymbol{\lambda}\leq\mathbf{e}_{q+1} \\ \boldsymbol{\lambda}\leq\bar{M}\mathbf{v} \\ \mathbf{Ax}+\mathbf{By}+\mathbf{Cu}+\delta\mathbf{1}-\mathbf{c}\leq\bar{M}(\mathbf{1}-\mathbf{v}) \\ (\mathbf{y},\delta)\leq\bar{M}\mathbf{w} \\ \mathbf{e}_{q+1}-[\mathbf{B}\quad\mathbf{1}]^\top\boldsymbol{\lambda}\leq\bar{M}(\mathbf{1}-\mathbf{w}), \end{cases} \tag{27}$$

where the constraint $\mathbf{u}\in\tilde{\mathcal{U}}^*_{\nu,\mu}$ in both oracles can be equivalently replaced by the expressions in (25) with some additional auxiliary variables.

Remark 1: It is obvious that the use of the intersection MKL uncertainty set $\tilde{\mathcal{U}}^*_{\nu,\mu}$ does not structurally increase the solution difficulty of the oracles, keeping them as mixed integer linear programs that can be efficiently solved by existing solvers, but only introduces $(r-1)\sum_{b=1}^{B}\mathrm{card}SV^{(b)}\cdot\mathrm{card}SK^{(b)}$ additional scalar variables and $\left(2(r-1)\sum_{b=1}^{B}\mathrm{card}SV^{(b)}\cdot\mathrm{card}SK^{(b)}+B\right)$ linear scalar constraints in total. Since in the MKL-OC-SVM (4) ν and μ are lower bounds on the proportions of SVs and SKs respectively [19], an optimistic estimation for the number of additional scalar variables brought by $\tilde{\mathcal{U}}^*_{\nu,\mu}$ should be $(r-1)N\nu M\mu$ and that for the linear scalar constraints is $(2(r-1)N\nu M\mu+B)$.

C. MKL Uncertainty Set-Induced CCG Algorithm

The CCG algorithm needs at least one scenario $\mathbf{u}^0\in\mathcal{U}$ to be incorperated into the significant scenario set $\mathcal{S}$ to start from solving MP (21). Although it is valid to initialize $\mathcal{S}$ with any partial enumeration of $\mathcal{U}$, a conceivable fact is that a good initialization will yield a tighter lower bound LB in the first iteration and thereby speed up the convergence of the algorithm. Scholars have devised various strategies to obtain a better initialization $\mathcal{S}^0$ for $\mathcal{S}$. For example, Zhao and Zeng [29] suggest to randomly select a feasible point $\mathbf{u}^0$ from the uncertainty set $\mathcal{U}$ to establish a singleton $\mathcal{S}^0=\{\mathbf{u}^0\}$. Specifically, they take an extreme point of $\mathcal{U}$ as $\mathbf{u}^0$ by arbitrarily solving the problem $\max_{\mathbf{u}}\{\mathbf{1}^\top\mathbf{u}|\mathbf{u}\in\mathcal{U}\}$ in advance, or take the worst-case scenario $\mathbf{u}^0$ obtained from the previous solution within their algorithm procedure. Tsang et al. [32] propose an inexact CCG method, whose early iterations can essentially be regarded as repeatedly pre-solving the MP and the SP in a fast but inexact manner and checking and collecting scenarios one by one until obtaining an $\mathcal{S}^0$ that can give an ideal LB. In Bertsimas and Kim [33], to reduce the number of different target classes the machine learning algorithm needs to be trained on, the authers provide a data-driven heuristic warm start to the CCG algorithm, which plays an important role in their proposed method.

Under the data-driven two-stage RO setup in this paper, although we originally have a large number of discrete uncertainty scenarios in $\mathcal{D}$ available for initializing $\mathcal{S}$, it is clearly unrealistic and unnecessary to include all of them in $\mathcal{S}^0$. Fortunately, the construction process of uncertainty set based on MKL-OC-SVM naturally identifies some potentially significant scenarios, namely the BSVs mentioned earlier. Thanks to the sparsity of MKL-OC-SVM, these relatively few BSVs are very likely to be the extreme points of $\mathcal{U}^*_{\nu,\mu}(\mathcal{D})$, and even if they are not, they have the potential to induce MP (21) to give a more valuable lower bound. With this intuition in mind, this paper suggests to incorporate the BSVs discovered by MKL-OC-SVM into the $\mathcal{S}^0$ obtained by the aforementioned initialization methods to initialize $\mathcal{S}$, in the hope of further reducing the iterations of the CCG algorithm and improving its efficiency.

Algorithm 2 MKL Uncertainty Set-Induced CCG

```
Input: $\mathcal{X}$, $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$, $\tilde{\mathcal{U}}^*_{\nu,\mu}(\mathcal{D})$, $\mathcal{BSV}$, $\mathcal{S}^0$, $\epsilon>0$
Output: $\mathbf{x}$
 1: Initialize: $LB\leftarrow-\infty$, $UB\leftarrow+\infty$, $\mathcal{S}\leftarrow\mathcal{S}^0\cup\mathcal{BSV}$.
 2: loop
 3:   Compute $\mathbf{x}$ and $\eta$ by solving MP $P(\mathcal{S})$, i.e., (21).
 4:   Update the lower bound $LB\leftarrow\mathbf{a}^\top\mathbf{x}+\eta$.
 5:   if $UB-LB\leq\epsilon$ then
 6:     return $\mathbf{x}$
 7:   end if
 8:   Obtain $\delta$ and $\mathbf{u}$ by calling Feasibility Oracle $\tilde{F}(\mathbf{x})$, i.e., (27).
 9:   if $\delta>\epsilon$ then {Note: $\mathbf{x}$ is infeasible.}
10:     Update the upper bound $UB\leftarrow\min\{UB,+\infty\}$.
11:   else {Note: $\mathbf{x}$ is feasible.}
12:     Obtain $\mathbf{y}$ and $\mathbf{u}$ by calling Optimality Oracle $\tilde{Q}(\mathbf{x})$, i.e., (26).
13:     Update the unpper bound $UB\leftarrow\min\{UB,\mathbf{a}^\top\mathbf{x}+\mathbf{b}^\top\mathbf{y}\}$.
14:   end if
15:   if $UB-LB\leq\epsilon$ then
16:     return $\mathbf{x}$
17:   end if
18:   Update the significant scenario set $\mathcal{S}\leftarrow\mathcal{S}\cup\{\mathbf{u}\}$.
19: end loop
```

As an integration of all discussions in this section, the MKL uncertainty set-induced CCG (MKLCCG) algorithm is summarized in Algorithm 2.

Remark 2: The proposed MKL-aided CCG method yields a numerically exact optimal solution to the two-stage RO (1) for any given MKL-based uncertainty set (18), which thus establishes a connection with two-stage stochastic programming under *sample average approximation* [46]–[48] with the commonly used risk measure *value-at-risk* ($\mathbb{V}@\mathbb{R}$) [49], [50]. Specifically, if the probability distribution of the uncertainty is estimated empirically based on the finite discrete scenarios $\mathcal{D}$, i.e., $\hat{\mathbb{P}}\{\mathbf{u}=\mathbf{u}_i\}=\frac{1}{N},\forall i$, the optimal objective value $z^\star$

of (1) can then be interpreted as

$$z^\star = \mathbb{V}@\mathbb{R}_\chi[z] = \min\left\{\tilde{z}\,\middle|\,\hat{\mathbb{P}}\left\{z \le \tilde{z}\right\} \ge \chi\right\}, \tag{28}$$

where χ is given by the probability that an uncertainty realization falls into the uncertaity set $\mathcal{U}^*_{\nu,\mu}(\mathcal{D})$. In other words, we have at least an confidence level of $\chi \times 100\%$ to believe that the actual loss z does not exceed $z^\star$ if we follow the decision suggested by the optimal solution of (1). A conservative *a priori* lower bound on χ is provided by $\underline{\chi} = 1 - B\nu$ in the light of Proposition 2, with a reference value $\tilde{\chi} = 1 - \nu$, which can be used as a rough guide to determine the values of hyperparameters when learning the uncertainty set. Once the uncertainty set is constructed, a calibrated empirical value of χ based on the current training will be revealed, i.e., $\hat{\chi} = 1 - \text{card}\mathcal{OL}/N$. This indicates that the MKL-aided CCG method proposed in this paper can not only provide an optimal solution to (1), but also offer a quantitative assessment of the risk level associated with that solution. On the other hand, decision-makers are also able to select appropriate hyperparameters through a trial-and-error process thereby constructing an uncertainty set that meets a specified degree of risk aversion and then inducing the MKLCCG algorithm to give a satisfying solution. In addition, a more mathematically rigorous *generalization* lower bound on χ can be obtained according to [51] under some restrictive conditions.

IV. MKLTwoStageRO.jl: An Open-Source Software Package

To facilitate users in practical evaluation and application, the MKL-aided CCG method proposed in this paper has been incorporated into an open-source software package `MKLTwoStageRO.jl`. With its assistance, users can construct MKL uncertainty sets based on DNPNKs parallelly, visualize the uncertainty sets, solve the two-stage RO using the MKLCCG algorithm, and retrieve the adaptive recourse decisions according to revealed uncertainties at the second-stage, with a small amount of code. This package is written in the Julia language [52], and a prototype version has been registered in the General registry of Julia.

This package also supports constructing MKL-based uncertainty sets using DPDKs [19] or other existing or custom kernels (some additional acceleration techniques are also adopted in this package when computing the plenty of kernel matrices so that the efficiency of the entire MKL process can be further improved), while also allowing for the construction of non-MKL-based polyhedral uncertainty sets for two-stage RO (e.g., those based on general algebraic expressions or polyhedral geometric operations). Users can also choose other classic two-stage RO algorithms besides the proposed MKLCCG, such as extended column-and-constraint generation (ECCG) [30], original (nested) column-and-constraint generation (CCG) [28], [29], and Benders-dual cutting plane method (BDCP) [25], [26], [28].

Readers are directed to the link https://github.com/hanb16/MKLTwoStageRO.jl for the source code, examples and documentation of the package `MKLTwoStageRO.jl`.

V. Case Study: Data-Driven Robust Location-Transportation Problem

A. Problem Statement

Consider the location-transportation problem where m potential facilities supply a commodity to n customers. For each potential facility i, the fixed cost for building it at the corresponding site is f_i, its unit capacity cost is a_i, and its maximal production capacity is c_i. The unit transportation cost between facility i and customer j is given by b_{ij}. Furthermore, the demand of each customer j is d_j, which in practice is unknown before any facility is opened and its capacity is determined. However, in many cases, we can obtain a certain amount of historical or experimental demand data $\mathcal{D}$ through market survey or other approaches. The question now is how to make robust yet flexible decisions about location, production and transportation based on these data.

We denote by $x_i \in \{0,1\}$ whether potential facility i is built, by $z_i \in \mathbb{R}_+$ how much the facility is to produce, and by $y_{ij} \in \mathbb{R}_+$ the transportation amount from facility i to customer j. Then this problem can be modeled by the following two-stage RO:

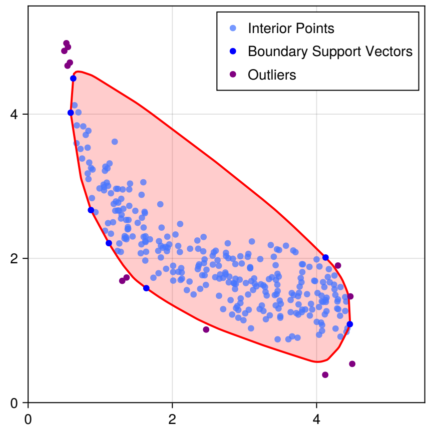
$$\begin{aligned}
\min_{(\mathbf{x},\mathbf{z})\in\mathcal{X}} & \sum_i (f_i x_i + a_i z_i) + \max_{\mathbf{d}\in\mathcal{U}(\mathcal{D})} \min_{\mathbf{y}\in\mathcal{Y}(\mathbf{x},\mathbf{z},\mathbf{d})} \sum_{i,j} b_{ij} y_{ij} \\
\text{s.t.} \quad & \mathcal{X} = \left\{(\mathbf{x},\mathbf{z}) \in \{0,1\}^m \times \mathbb{R}^m_+ \,|\, z_i \le c_i x_i,\ \forall i\right\} \\
& \mathcal{Y}(\mathbf{x},\mathbf{z},\mathbf{d}) = \Big\{\mathbf{y} \in \mathbb{R}^{m\times n}_+ \,\Big|\, \textstyle\sum_j y_{ij} \le z_i,\ \forall i, \\
& \qquad \textstyle\sum_i y_{ij} \ge d_j,\ \forall j\Big\},
\end{aligned} \tag{29}$$

where $\mathcal{U}(\mathcal{D})$ is the data-driven uncertainty set that we are going to construct with the proposed MKL-based method.

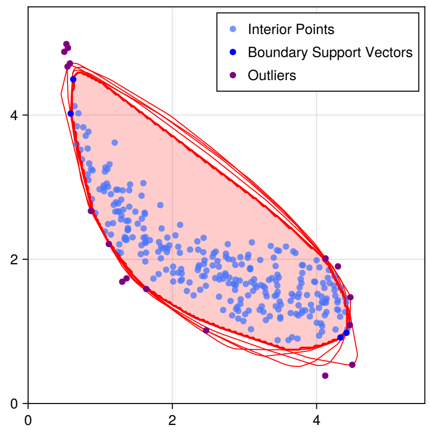
In the following subsections, the parameters of the instances are chosen from the following ranges: $f_i \in [300, 500]$, $a_i \in [15, 25]$ and $b_{ij} \in [10, 40]$. In addition, for a given demand data set $\mathcal{D}$, we chose a proper scaling factor $\sigma > 1$ to have $\sum_i c_i = \sigma \cdot \max_{\mathbf{d}}\{\sum_j d_j | \mathbf{d} \in \mathcal{D}\}$ so that the feasibility of (29) can be ensured for any possible realization in the data-driven uncertainty set $\mathcal{U}(\mathcal{D})$ (i.e., the FA holds).

B. An Illustrative Toy Instance

We first study a toy instance where there are $m = 4$ potential facilitate and $n = 2$ customers. The demand data $\mathcal{D}$ of the two customers is given by the $N = 300$ scatter points in Fig. 1, which presents significant negative correlation between the dimensions. We generated $M = 50$ basis DNPNKs for this problem with their corresponding unit direction vectors $\{\mathbf{q}_m\}_{m=1}^{50}$ drawn evenly from the 2-dimensional uncertainty space [19]. These kernels were treated as a single bundle to learn using the conventional HessianMKL for MKL-OC-SVM algorithm [19] (equivalently $B = 1$), and were systemeticially grouped into $B = 5$ batches to resort to the proposed parallel HessianMKL for MKL-OC-SVM Algorithm 1, respectively.



(a) The conventional MKL uncertainty set ($B = 1$)



(b) The intersection MKL uncertainty set ($B = 5$)

Fig. 1. Conventional and intersection MKL uncertainty sets constructed by Algorithm 1 ($\nu = 0.05$, $\mu = 0.05$).

In both cases, we chose the kernel regularization parameter as $\mu = 0.05$ in order to ensure the total number of SKs won't be less than $\lceil M\mu \rceil = 3$, and the data regularization parameter was also set to be $\nu = 0.05$ with the hope that the number of SVs in each batch would be no less than $N\nu = 15$ and the number of outliers won't exceed this.

The above two cases were both implemented in Julia v1.10.3 with the help of the package `MKLTwoStageRO.jl` on a general Windows personal computer. Due to the small scale, their MKL processes were quickly completed within 2 seconds, and the efficiency advantage of the proposed parallel algorithm was conceivably not significant for this instance. The conventional and intersection MKL uncertainty sets constructed in both cases are shown in Fig. 1, where the pink areas with thick red boundaries are the sets learned and the areas surrounded by the thin red lines in Fig. 1b represent the sets obtained by each parallel batch. It can be seen by comparing the two subfigures in Fig. 1 that the intersection MKL uncertainty set tends to reduce redundant coverage, while the conventional OC-SVM does not seek to minimize the volume of the set [51]. As a result, the conventional MKL uncertainty set covers 96.0% of uncertainty scenarios with 12 outliers, and the intersection MKL uncertainty set covers 94.3% of uncertainty scenarios with 17 outliers.

TABLE I
ALGORITHM PERFORMANCE COMPARISON

Iter.	ECCG			MKLCCG		
	LB	UB	Gap	LB	UB	Gap
0	$-\infty$	$+\infty$	$+\infty$	$-\infty$	$+\infty$	$+\infty$
1	1480.55	$+\infty$	$+\infty$	1505.53	1510.38	4.85
2	1480.55	1513.99	33.44	1510.38	–	0.00
3	1510.38	1510.38	0.00			

The intersection MKL uncertainty set was then adopted for the subsequent CCG procedure. In addition to the proposed MKLCCG Algorithm 2, as a comparison, the same set is also used to solve with the algorithm denoted as ECCG [30]. The main difference between them is that ECCG only treats the intersection MKL uncertainty set as a general polyhedron and does not take full advantage of the BSV significant scenarios identified in the MKL phase. For both algorithms, the initial significant scenario sets $\mathcal{S}^0$ are given by a same singleton $\{\mathbf{d}^0\}$ where $\mathbf{d}^0$ is the scenario that solves $\max_{\mathbf{d}}\{\mathbf{1}^\top\mathbf{d}|\mathbf{d} \in \mathcal{U}(\mathcal{D})\}$ [29].

The dynamic behaviors of the two algorithms are presented in Table I. It distinctly shows that the ECCG algorithm terminated after 3 entire iterations, while the MKLCCG algorithm met the termination criterion earlier after updating the LB at the middle of the 2nd iteration. We further looked into the iteration processes of both algorithms and found that, in the 1st iteration of the ECCG algorithm it solved the MP $P(\mathcal{S}^0)$ and yielded the lower bound 1480.55 with the *active* scenario $\mathbf{d}^0 = (4.04, 2.07)$, whereas for the MKLCCG algorithm the *active* scenario in the 1st iteration was one of the BSVs $(0.63, 4.50)$ so that it raised the first lower bound to 1505.53 through the MP $P(\mathcal{S}^0 \cup \mathcal{BSV})$, which accelerated the convergence. Furthermore, due to the fact that the RCRA doesn't hold for this instance, the ECCG algorithm unluckily produced an infeasible $\mathbf{x}$ at the first solution of the MP, thus delaying its UB update. Finally, both algorithms reached the worst-case scenario $(0.79, 4.58)$ and achieved the same optimal objective value 1510.38 that we might believe the actual overall cost of the location-transportation problem won't exceed with an empirical confidence level more than 94.3%. Multiple repeated solutions showed that the MKLCCG algorithm could converge within 1 second, while ECCG took more than 2 seconds to complete. This seems negligible in this particular instance, but we point out that since the MKLCCG algorithm has the potential to reduce the number of iterations, its advantage may be significant when dealing with some tough cases where calling the Feasibility Oracle $\tilde{F}(\mathbf{x})$ may be very time-consuming even if the corresponding Optimality Oracle $\tilde{Q}(\mathbf{x})$ is efficient, as found by Bertsimas and Shtern [30] in their numerical experiments.

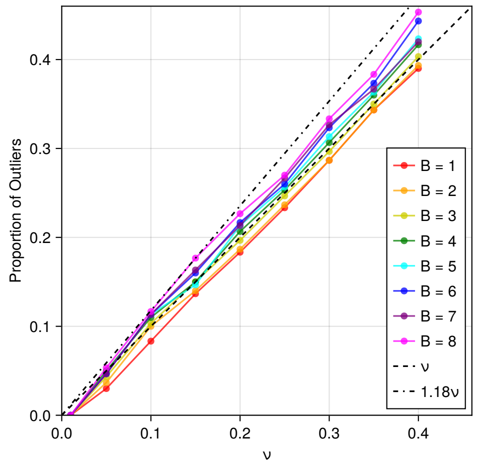


Fig. 2. The variation trend of the proportion of outliers with respect to ν under different B.

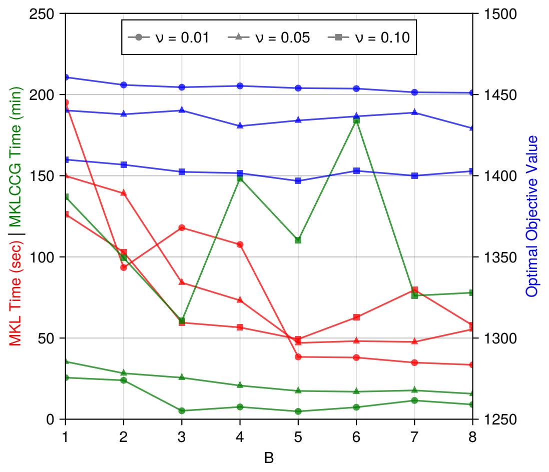


Fig. 3. The variation trends of the learning (MKL) and solution (MKLCCG) time and the optimal objective value with respect to B under different ν.

C. A Higher Dimensional Instance

We then study a higher dimensional instance to find out how the hyper-parameters ν and B influence the result and the performance of the proposed method. There are $m = 4$ potential facilities and $n = 4$ customers in this instance. The demand data was sampled with size $N = 300$ from a 4-dimensional Gaussian distribution whose mean vector was randomly selected from $[3, 6]^4$ and covariance matrix was a randomly generated symmetric positive definite matrix with diagonal elements taken from $[1, 2]$ and non-diagonal elements taken from $[-1, 0]$. We constructed the uncertainty set with Algorithm 1 and solved the induced two-stage RO with Algorithm 2 for this instance under varying ν and B on the High-Performance Computing Platform of Peking University using Julia v1.10.3 and `MKLTwoStageRO.jl`.

When constructing the uncertainty set under each combination of ν and B, we uniformly provided $M = 11^3 = 1331$ candidate basis kernels for learning, with the corresponding unit direction vectors $\{\mathbf{q}_m\}_{m=1}^{1331}$ evenly selected within the 4-dimensional uncertainty space. The kernel regularization parameter μ was set to 0.05 for all cases. For cases where $B > 1$, we systematically grouped the basis kernels for parallel learning. Fig. 2 shows the trend of the proportion of outliers identified by the learned sets under different values of B as ν varies. We can see from the figure that, within the illustrated range of ν and B, the proportion of outliers generally increases approximately linearly with ν, and the growth rate roughly increases as B gets larger. When $B = 1$, ν is a strict upper bound on the proportion of outliers [19], [51], while when $B > 1$, this bound is crossed as ν increases. Proposition 2 presents an upper bound $B\nu$ for $B \geq 1$, but it is conservative. As a matter of fact, over all the values of ν and B shown in Fig. 2, the proportion of outliers does not actually exceed 1.18ν at most. Therefore, for the proposed intersection MKL uncertainty set, ν can effectively regulate the number of outliers that the set fails to cover, and it is still meaningful to regard it as an *a prior* reference value for the proportion of outliers within a reasonable range of ν and B in practice.

Fig. 3 depicts the variations with respect to B of the times taken to construct uncertainty set through MKL and to solve the induced two-stage RO using the MKLCCG algorithm, together with the corresponding optimal objective value obtained, under different ν. Note that the MKL time (in red) in the figure includes not only the execution time of Algorithm 1, but also the preceding kernel generation and kernel matrix computation time. As can be seen from the figure, after taking the measure of multi-process parallel learning, the MKL time is significantly reduced, with a maximum *speedup ratio* (the wall clock time using a single process divided by the wall clock time using multiple processes) of about 2–6. This speedup effect seems to be more significant for smaller values of ν. The MKLCCG time (in green) also shows a decreasing trend as B increases, although there is a large fluctuation at $\nu = 0.10$ (This may be due to the fact that the algorithm process occasionally encounters the aforementioned time-consuming situation of calling the Feasibility Oracle). The optimal objective value (in blue) of the induced two-stage RO also decreases with increasing B, which is a direct consequence of the tendency of intersection MKL uncertainty set to reduce set redundant coverage. From the perspective of ν, it can be found that an increase in ν significantly increases the time consumption of MKLCCG and significantly reduces the optimal objective value. This is because a larger ν corresponds to a set containing fewer potential uncertainty scenarios, thus leading to a robust optimization solution with higher risk tolerance, and at the same time, this brings more scalar variables and constraints to the robust counterpart.

VI. Concluding Remarks

In this paper, we presented a data-driven integrated workflow for modeling and solving two-stage RO problems, namely, the MKL-aided CCG method. This method covers the two main phases of dealing with general two-stage RO, i.e., the construction of uncertainty set and the solution of the induced robust counterpart, which are supported by two algorithms respectively. We showed that by using this method, a compact piecewise linear polyhedral uncertainty set that is the intersection of different MKL-based uncertainty sets can be constructed efficiently from massive historical uncertainty data and plenty of basis DNPNKs. This intersection uncertainty set consists of sparse SVs and SKs so that the tractability of the robust counterpart can be ensured, and its identified BSVs can provide effective initial cuts for the subsequent CCG procedure, thereby assisting its convergence. The proposed method seeks to provide the decision-makers with exact robust decision solutions that can resist over-conservatism when only the feasibility assumption holds, and at the same time, provide probability-based quality evaluation for them. And as depicted in the figures, the risk aversion degree of the solution can be adjusted quantificationally through hyperparameters. We also released an easy-to-use companion software package to the Julia community, the effectiveness and practicability of which have been verified through numerical experiments together with the proposed method. Future work may be directed towards further developing the theoretical framework of the method and improving the software package to make it more user-friendly.

Acknowledgment

This work is supported by the High-Performance Computing Platform of Peking University.

The author would like to thank Dr. Ying Guan, research fellow at Harvard University, for providing substantial support.

By default, the package `MKLTwoStageRO.jl` implicitly uses open-source packages like `KernelFunctions.jl`[1], optimization solvers `HiGHS.jl`[2] and `Ipopt.jl`[3], single kernel SVM solver `LIBSVM.jl`[4], etc. Thanks for these quality packages, although the users are also allowed to replace them with other alternatives.

References

[1] A. Ben-Tal, A. Nemirovski, and L. El Ghaoui, *Robust Optimization*, ser. Princeton Series in Applied Mathematics Ser. Princeton: Princeton University Press, 2009.

[2] D. Bertsimas, D. B. Brown, and C. Caramanis, "Theory and applications of robust optimization," *SIAM Review*, vol. 53, no. 3, pp. 464–501, Jan. 2011. [Online]. Available: https://doi.org/10.1137/080734510

[3] D. Luan, C. Wang, Z. Wu, and Z. Xia, "Two-stage robust optimization model for uncertainty investment portfolio problems," *Journal of Mathematics*, vol. 2021, no. 1, p. 3087066, 2021. [Online]. Available: https://doi.org/10.1155/2021/3087066

[4] H. Qiu, W. Gu, P. Liu, Q. Sun, Z. Wu, and X. Lu, "Application of two-stage robust optimization theory in power system scheduling under uncertainties: A review and perspective," *Energy*, vol. 251, p. 123942, 2022. [Online]. Available: https://doi.org/10.1016/j.energy.2022.123942

[5] C. Cheng, M. Qi, Y. Zhang, and L.-M. Rousseau, "A two-stage robust approach for the reliable logistics network design problem," *Transportation Research Part B: Methodological*, vol. 111, pp. 185–202, 2018. [Online]. Available: https://doi.org/10.1016/j.trb.2018.03.015

[1] https://github.com/JuliaGaussianProcesses/KernelFunctions.jl
[2] https://github.com/jump-dev/HiGHS.jl
[3] https://github.com/jump-dev/Ipopt.jl
[4] https://github.com/JuliaML/LIBSVM.jl

[6] S. Neyshabouri and B. P. Berg, "Two-stage robust optimization approach to elective surgery and downstream capacity planning," *European Journal of Operational Research*, vol. 260, no. 1, pp. 21–40, Jul. 2017. [Online]. Available: https://doi.org/10.1016/j.ejor.2016.11.043

[7] A. Ben-Tal, A. Goryashko, E. Guslitzer, and A. Nemirovski, "Adjustable robust solutions of uncertain linear programs," *Mathematical Programming*, vol. 99, no. 2, pp. 351–376, Mar. 2004. [Online]. Available: https://doi.org/10.1007/s10107-003-0454-y

[8] A. L. Soyster, "Technical note—convex programming with set-inclusive constraints and applications to inexact linear programming," *Operations Research*, vol. 21, no. 5, pp. 1154–1157, Oct. 1973. [Online]. Available: https://doi.org/10.1287/opre.21.5.1154

[9] A. Ben-Tal and A. Nemirovski, "Robust convex optimization," *Mathematics of Operations Research*, vol. 23, no. 4, pp. 769–805, Nov. 1998. [Online]. Available: https://doi.org/10.1287/moor.23.4.769

[10] D. Bertsimas, D. Pachamanova, and M. Sim, "Robust linear optimization under general norms," *Operations Research Letters*, vol. 32, no. 6, pp. 510–516, Nov. 2004. [Online]. Available: https://doi.org/10.1016/j.orl.2003.12.007

[11] A. Ben-Tal and A. Nemirovski, "Robust solutions of linear programming problems contaminated with uncertain data," *Mathematical Programming*, vol. 88, no. 3, pp. 411–424, Sep. 2000. [Online]. Available: https://doi.org/10.1007/PL00011380

[12] Z. Li, R. Ding, and C. A. Floudas, "A comparative theoretical and computational study on robust counterpart optimization: I. robust linear optimization and robust mixed integer linear optimization," *Industrial & Engineering Chemistry Research*, vol. 50, no. 18, pp. 10 567–10 603, Aug. 2011. [Online]. Available: https://doi.org/10.1021/ie200150p

[13] D. Bertsimas and A. Thiele, *Robust and data-driven optimization: modern decision making under uncertainty*. INFORMS, Sep. 2006, pp. 95–122. [Online]. Available: https://doi.org/10.1287/educ.1063.0022

[14] C. Ning and F. You, "Optimization under uncertainty in the era of big data and deep learning: When machine learning meets mathematical programming," *Computers & Chemical Engineering*, vol. 125, pp. 434–448, Jun. 2019. [Online]. Available: https://doi.org/10.1016/j.compchemeng.2019.03.034

[15] T. Chen, X. Chen, W. Chen, H. Heaton, J. Liu, Z. Wang, and W. Yin, "Learning to optimize: A primer and a benchmark," *Journal of Machine Learning Research*, vol. 23, no. 189, pp. 1–59, 2022. [Online]. Available: http://jmlr.org/papers/v23/21-0308.html

[16] L. J. Hong, Z. Huang, and H. Lam, "Learning-based robust optimization: Procedures and statistical guarantees," *Management Science*, vol. 67, no. 6, pp. 3447–3467, Jun. 2021. [Online]. Available: https://doi.org/10.1287/mnsc.2020.3640

[17] C. Ning and F. You, "A data-driven multistage adaptive robust optimization framework for planning and scheduling under uncertainty," *AIChE Journal*, vol. 63, no. 10, pp. 4343–4369, May 2017. [Online]. Available: https://doi.org/10.1002/aic.15792

[18] C. Shang and F. You, "A data-driven robust optimization approach to scenario-based stochastic model predictive control," *Journal of Process Control*, vol. 75, pp. 24–39, 2019. [Online]. Available: https://doi.org/10.1016/j.jprocont.2018.12.013

[19] B. Han, C. Shang, and D. Huang, "Multiple kernel learning-aided robust optimization: Learning algorithm, computational tractability, and usage in multi-stage decision-making," *European Journal of Operational Research*, vol. 292, no. 3, pp. 1004–1018, Aug. 2021. [Online]. Available: https://doi.org/10.1016/j.ejor.2020.11.027

[20] I. Wang, C. Becker, B. Van Parys, and B. Stellato, "Learning decision-focused uncertainty sets in robust optimization," *arXiv preprint arXiv:2305.19225v4*, 2024. [Online]. Available: https://arxiv.org/abs/2305.19225v4

[21] T. Tulabandhula and C. Rudin, "Robust optimization using machine learning for uncertainty sets," *arXiv preprint arXiv:1407.1097v1*, 2014. [Online]. Available: https://arxiv.org/abs/1407.1097v1

[22] T. Campbell and J. P. How, "Bayesian nonparametric set construction for robust optimization," in *2015 American Control Conference (ACC)*, IEEE. IEEE, Jul. 2015, pp. 4216–4221. [Online]. Available: https://doi.org/10.1109/acc.2015.7171991

[23] C. Shang, X. Huang, and F. You, "Data-driven robust optimization based on kernel learning," *Computers & Chemical Engineering*, vol. 106, pp. 464–479, Nov. 2017. [Online]. Available: https://doi.org/10.1016/j.compchemeng.2017.07.004

[24] M. Goerigk and J. Kurtz, "Data-driven robust optimization using deep neural networks," *Computers & Operations Research*, vol. 151, p.

106087, Mar. 2023. [Online]. Available: https://doi.org/10.1016/j.cor.2022.106087

[25] J. F. Benders, "Partitioning procedures for solving mixed-variables programming problems," *Numerische Mathematik*, vol. 4, no. 1, pp. 238–252, Dec. 1962. [Online]. Available: https://doi.org/10.1007/bf01386316

[26] A. M. Geoffrion, "Generalized Benders decomposition," *Journal of Optimization Theory and Applications*, vol. 10, no. 4, pp. 237–260, Oct. 1972. [Online]. Available: https://doi.org/10.1007/bf00934810

[27] L. Zhao and B. Zeng, "Robust unit commitment problem with demand response and wind energy," in *2012 IEEE Power and Energy Society General Meeting*, IEEE. IEEE, Jul. 2012, pp. 1–8. [Online]. Available: https://doi.org/10.1109/pesgm.2012.6344860

[28] B. Zeng and L. Zhao, "Solving two-stage robust optimization problems using a column-and-constraint generation method," *Operations Research Letters*, vol. 41, no. 5, pp. 457–461, Sep. 2013. [Online]. Available: https://doi.org/10.1016/j.orl.2013.05.003

[29] L. Zhao and B. Zeng, "An exact algorithm for two-stage robust optimization with mixed integer recourse problems," University South Florida, Tech. Rep., 2012. [Online]. Available: https://optimization-online.org/?p=11876

[30] D. Bertsimas and S. Shtern, "A scalable algorithm for two-stage adaptive linear optimization," *arXiv preprint arXiv:1807.02812v1*, 2018. [Online]. Available: https://arxiv.org/abs/1807.02812v1

[31] W. Wang and B. Zeng, "Computing two-stage robust optimization with mixed integer structures," *arXiv preprint arXiv:2312.13607v1*, 2023. [Online]. Available: https://arxiv.org/abs/2312.13607v1

[32] M. Y. Tsang, K. S. Shehadeh, and F. E. Curtis, "An inexact column-and-constraint generation method to solve two-stage robust optimization problems," *Operations Research Letters*, vol. 51, no. 1, pp. 92–98, Jan. 2023. [Online]. Available: https://doi.org/10.1016/j.orl.2022.12.002

[33] D. Bertsimas and C. W. Kim, "A machine learning approach to two-stage adaptive robust optimization," *European Journal of Operational Research*, vol. 319, no. 1, pp. 16–30, Nov. 2024. [Online]. Available: https://doi.org/10.1016/j.ejor.2024.06.012

[34] J. Dumouchelle, E. Julien, J. Kurtz, and E. B. Khalil, "Deep learning for two-stage robust integer optimization," *arXiv preprint arXiv.2310.04345*, 2024. [Online]. Available: https://arxiv.org/abs/2310.04345v3

[35] C. Ning and F. You, "Data-driven adaptive nested robust optimization: General modeling framework and efficient computational algorithm for decision making under uncertainty," *AIChE Journal*, vol. 63, no. 9, pp. 3790–3817, Apr. 2017. [Online]. Available: https://doi.org/10.1002/aic.15717

[36] ——, "Data-driven decision making under uncertainty integrating robust optimization with principal component analysis and kernel smoothing methods," *Computers & Chemical Engineering*, vol. 112, pp. 190–210, Apr. 2018. [Online]. Available: https://doi.org/10.1016/j.compchemeng.2018.02.007

[37] B. Schölkopf and A. J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[38] A. Rakotomamonjy, F. Bach, S. Canu, and Y. Grandvalet, "Simplemkl," *Journal of Machine Learning Research*, vol. 9, no. 83, pp. 2491–2521, 2008. [Online]. Available: https://www.jmlr.org/papers/v9/rakotomamonjy08a.html

[39] G. R. Lanckriet, N. Cristianini, P. Bartlett, L. E. Ghaoui, and M. I. Jordan, "Learning the kernel matrix with semidefinite programming," *Journal of Machine learning research*, vol. 5, no. Jan, pp. 27–72, 2004. [Online]. Available: https://www.jmlr.org/papers/v5/lanckriet04a.html

[40] X. Xu, I. W. Tsang, and D. Xu, "Soft margin multiple kernel learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, no. 5, pp. 749–761, May 2013. [Online]. Available: https://doi.org/10.1109/tnnls.2012.2237183

[41] F. Odone, A. Barla, and A. Verri, "Building kernels from binary strings for image matching," *IEEE Transactions on Image Processing*, vol. 14, no. 2, pp. 169–180, Feb. 2005. [Online]. Available: https://doi.org/10.1109/tip.2004.840701

[42] S. Maji, A. C. Berg, and J. Malik, "Classification using intersection kernel support vector machines is efficient," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE. IEEE, Jun. 2008, pp. 1–8. [Online]. Available: https://doi.org/10.1109/cvpr.2008.4587630

[43] ——, "Efficient classification for additive kernel svms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 66–77, Jan. 2012. [Online]. Available: https://doi.org/10.1109/tpami.2012.62

[44] O. Chapelle and A. Rakotomamonjy, "Second order optimization of kernel parameters," in *Proc. of the NIPS Workshop on Kernel Learning: Automatic Selection of Optimal Kernels*, vol. 19. Citeseer, 2008, p. 87. [Online]. Available: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=a294a7cfe9093d12f0fd29e35ad44cf5dbbc4412

[45] Y. Liu, W. He, and L. Zhao, "Data-driven robust optimization with multiple kernel learning for refinery planning under price uncertainty," in *2022 Australian & New Zealand Control Conference (ANZCC)*, IEEE. IEEE, Nov. 2022, pp. 81–86. [Online]. Available: https://doi.org/10.1109/anzcc56036.2022.9966966

[46] A. J. King and R. T. Rockafellar, "Asymptotic theory for solutions in statistical estimation and stochastic programming," *Mathematics of Operations Research*, vol. 18, no. 1, pp. 148–162, Feb. 1993. [Online]. Available: https://doi.org/10.1287/moor.18.1.148

[47] A. Shapiro, "Asymptotic behavior of optimal solutions in stochastic programming," *Mathematics of Operations Research*, vol. 18, no. 4, pp. 829–845, Nov. 1993. [Online]. Available: https://doi.org/10.1287/moor.18.4.829

[48] T. Homem-de Mello and G. Bayraksan, "Monte carlo sampling-based methods for stochastic optimization," *Surveys in Operations Research and Management Science*, vol. 19, no. 1, pp. 56–85, Jan. 2014. [Online]. Available: https://doi.org/10.1016/j.sorms.2014.05.001

[49] R. Rockafellar and S. Uryasev, "Conditional value-at-risk for general loss distributions," *Journal of Banking & Finance*, vol. 26, no. 7, pp. 1443–1471, Jul. 2002. [Online]. Available: https://doi.org/10.1016/s0378-4266(02)00271-6

[50] S. Sarykalin, G. Serraino, and S. Uryasev, *Value-at-risk vs. conditional value-at-risk in risk management and optimization*. INFORMS, Sep. 2008, pp. 270–294. [Online]. Available: https://doi.org/10.1287/educ.1080.0052

[51] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Computation*, vol. 13, no. 7, pp. 1443–1471, Jul. 2001. [Online]. Available: https://doi.org/10.1162/089976601750264965

[52] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, "Julia: A fresh approach to numerical computing," *SIAM Review*, vol. 59, no. 1, pp. 65–98, Jan. 2017. [Online]. Available: https://doi.org/10.1137/141000671