Computational Statistics and Data Analysis 119 (2018) 99–117

Contents lists available at ScienceDirect

Computational Statistics and Data Analysis

journal homepage: www.elsevier.com/locate/csda

An algorithm based on semidefinite programming for finding minimax optimal designs[☆]

Belmiro P.M. Duarte[a,b,*], Guillaume Sagnol[c], Weng Kee Wong[d]

[a] Polytechnic Institute of Coimbra, ISEC, Department of Chemical and Biological Engineering, Portugal
[b] CIEPQPF, Department of Chemical Engineering, University of Coimbra, Portugal
[c] Technische Universität Berlin, Institut für Mathematik, Germany
[d] Department of Biostatistics, Fielding School of Public Health, UCLA, USA

HIGHLIGHTS

- An algorithm for finding continuous minimax optimal designs of experiments for semidefinite representable criteria.
- An algorithm based on sequential cutting plane approach.
- Semidefinite programming used to find optimal designs on a previously discretized domain.
- Nonlinear programming used to find the least attainable parameter combination for local designs.
- Algorithm convergence is theoretically demonstrated.

ARTICLE INFO

Article history:
Received 20 April 2017
Received in revised form 27 September 2017
Accepted 28 September 2017
Available online 24 October 2017

Keywords:
Cutting plane algorithm
Design efficiency
Equivalence theorem
Model-based optimal design
Nonlinear programming

ABSTRACT

An algorithm based on a delayed constraint generation method for solving semi-infinite programs for constructing minimax optimal designs for nonlinear models is proposed. The outer optimization level of the minimax optimization problem is solved using a semidefinite programming based approach that requires the design space be discretized. A nonlinear programming solver is then used to solve the inner program to determine the combination of the parameters that yields the worst-case value of the design criterion. The proposed algorithm is applied to find minimax optimal designs for the logistic model, the flexible 4-parameter Hill homoscedastic model and the general nth order consecutive reaction model, and shows that it (i) produces designs that compare well with minimax $D-$optimal designs obtained from semi-infinite programming method in the literature; (ii) can be applied to semidefinite representable optimality criteria, that include the common $A-$, $E-$, $G-$, $I-$ and D-optimality criteria; (iii) can tackle design problems with arbitrary linear constraints on the weights; and (iv) is fast and relatively easy to use.

1. Motivation

We consider the problem of determining model-based optimal designs of experiments (M-bODE) for statistical models. Such a problem has increasing relevance today to control costs with broad applications in areas such as biomedicine,

☆ An example of the code is presented as Supplementary Material.
* Correspondence to: Polytechnic Institute of Coimbra, ISEC, Department of Chemical and Biological Engineering, R. Pedro Nunes, 3030–199 Coimbra, Portugal.
E-mail addresses: bduarte@isec.pt (B.P.M. Duarte), sagnol@math.tu-berlin.de (G. Sagnol), wkwong@ucla.edu (W.K. Wong).

https://doi.org/10.1016/j.csda.2017.09.008
0167-9473/© 2017 Elsevier B.V. All rights reserved.

engineering, and pharmaceutical industry, to name a few (Berger and Wong, 2009; Goos and Jones, 2011; Fedorov and Leonov, 2014). M-bODE are useful because they can provide maximum information with high statistical efficiency at minimum cost. Our setup is that we have a given parametric model defined on a *compact* design space, a given design criterion and a given budget, that typically translates into having a predetermined total number of observations, n, available for the study. The design problem is to determine optimal design points that describe the experimental conditions and whether replicates are required at each of these design points, subject to the requirement that they sum to n. These design issues involve hard combinatorial optimization problems that are known to be NP-hard (Welch, 1982). However, the limit problem where $n \to \infty$, which is the focus of this paper, and in which we search the optimal proportion of the total number of trials to be performed at each individual design point, is an easier, convex continuous optimization problem.

In the field of M-bODE, mathematical programming approaches have been successfully applied to solve design problems. Some examples are Linear Programming (Gaivoronski, 1986; Harman and Jurík, 2008), Second Order Conic Programming (Sagnol, 2011; Sagnol and Harman, 2015), Semidefinite Programming (SDP) (Vandenberghe and Boyd, 1999; Papp, 2012; Duarte and Wong, 2015), Semi-Infinite Programming (SIP) (Duarte and Wong, 2014; Duarte et al., 2015), Nonlinear Programming (NLP) (Chaloner and Larntz, 1989; Molchanov and Zuyev, 2002), NLP combined with stochastic procedures such as genetic algorithms (Heredia-Langner et al., 2004; Zhang, 2006), and global optimization techniques (Boer and Hendrix, 2000; Duarte et al., 2016). Traditional algorithms for finding optimal designs are reviewed, compared and discussed in Cook and Nachtsheim (1982) and Pronzato (2008), among others. Mandal et al. (2015) provides a review of algorithms for generating optimal designs, including nature-inspired meta-heuristic algorithms, which are increasingly used in computer science and engineering to solve high dimensional complex optimization problems.

When the design criterion is not differentiable, finding an optimal design for a general nonlinear model is a difficult computational task. For instance, consider finding a minimax (or maximin) optimal design which has a non-differentiable criterion. The design criterion remains convex and there is an equivalence theorem for checking the optimality of the design. However in practice, there are considerable difficulties in finding and checking whether a design is optimal under the minimax framework (Noubiap and Seidel, 2000). In particular, the problem has two or more levels of optimization and the non-differentiability of the criterion requires mathematical programming based algorithms that can compute sub-gradient and iteratively evaluate the global or local (lower/upper) bounds to solve the optimization problem.

Minimax design problems abound in practice. For example, one may wish to estimate the overall response surface in a dose–response study. A common design criterion is to consider the areas where the largest predictive variance may occur and find a design that minimizes the maximum predictive variance. The outer level program finds a design after the inner problem determines the set of model parameters that results in the largest possible predictive variance. Theoretically, we can use SIP to tackle optimal design problems for any nonlinear model, see for example Duarte and Wong (2014). However, there are three potential issues with the SIP approach: (i) the task to program from scratch the various functionals of the inverse of the Fisher Information Matrix (*e.g.* determinant, trace) for the various design criteria can be complex and so may limit generalization of the algorithm to solve other problems; (ii) the SIP-based approach finds the optimal design in the outer optimization problem (which is a hard problem) using a NLP solver that does not guarantee global optimality of the solution unless global optimization techniques are employed; and (iii) the SIP-based algorithm determines the number of support point for the optimal design iteratively, which can be time consuming. This is in contrast to our proposed SDP–NLP combined approach where (i) we solve the outer optimization problem using SDP which guarantees the global optimality of the design; (ii) we use the NLP solver to only solve the inner optimization program; (iii) find the number of support points for the optimal design simultaneously using SDP; and (iv) our method optimizes the design points from a pre-determined candidate set of points versus having to search for the number and the design points over a continuous domain.

Our main contribution is an algorithm to determining minimax optimal designs using a combination of mathematical programming algorithms that on turn stand on deterministic replicable methods. The algorithm creatively combines SDP and NLP to find minimax $A-$, $E-$ and $D-$optimal designs easily and realize the efficiency gain in computing. In particular, the convex solver is able to identify support points of the design (within the predetermined set of points) by assigning zero-weight to non-support points for a broad class of *semidefinite representable* criteria. Our approach is flexible in that it can incorporate other constraints and strategies when appropriate. For instance, adaptive grid techniques recently proposed by Duarte et al. (2017) may be incorporated into our algorithm to refine the support points and collapse those that are close in proximity using a pre-defined ϵ^{COL}-vicinity tolerance. The key advantages in our approach are that the SDP solver guarantees that it finds the global optimum in polynomial time and the NLP solver only finds the parameter combination at which a locally optimal design is least efficient, which is, in most of the problems, a low dimension program. Another innovative aspect is using the structure of the plausible region of the parameters to judiciously construct the initial approximation of the continuous domain which increases the convergence rate to the solution.

Section 2 presents mathematical background for the SDP and NLP formulation problems. Section 3 provides the specifics for the SDP and NLP formulations for the outer and inner levels of the optimization problem. Section 4 presents three applications of the algorithm to find minimax optimal designs for nonlinear models where the nominal values of the parameters of interest all belong to a user-specified set called the plausibility region. We first consider the logistic model and then the more flexible and widely used 4-parameter homoscedastic Hill model to test if our proposed algorithms can generate minimax optimal designs similar to those reported in the literature. In the third application, we test our algorithm using the nth order consecutive reaction model described by ordinary differential equations. Section 5 concludes with a summary.

2. Background

This section provides the background material and the mathematical formulation for finding optimal experimental designs for nonlinear models. In Section 2.1 we introduce the minimax optimal design problem. Section 2.2 introduces the fundamentals of SDP and Section 2.3 presents the basics of NLP. We use bold face lowercase letters to represent vectors, bold face capital letters for continuous domains, blackboard bold capital letters for discrete domains, and capital letters for matrices. Finite sets containing ι elements are compactly represented by $[\iota] = \{1, \dots, \iota\}$.

Throughout we assume we have a regression model with a univariate response and several regressors $\boldsymbol{x} \in \mathbf{X} \subset \mathbb{R}^{n_x}$ where n_x is the dimension of the design space. The mean response at $\boldsymbol{x}$ is

$$\mathbb{E}[y|\boldsymbol{x}, \boldsymbol{p}] = f(\boldsymbol{x}, \boldsymbol{p}), \tag{1}$$

where $f(\boldsymbol{x}, \boldsymbol{p})$ is a given differentiable function, $\mathbb{E}[\bullet]$ is the expectation operator with respect to the error distribution and the vector of unknown model parameters is $\boldsymbol{p} \in \mathbf{P} \subset \mathbb{R}^{n_p}$. Here, $\mathbf{P}$ is a user-selected n_p-dimensional cartesian box $\mathbf{P} \equiv \times_{j=1}^{n_p} [l_j,\ u_j]$ with each interval $[l_j,\ u_j]$ representing the plausible range of values for the jth parameter. The exact optimal design problem is: given a design criterion and a predetermined sample size, n, select a set of n runs using the best combinations of the levels of the regressors to observe the responses.

Here, we focus on approximate or *continuous designs*, which are large sample designs so that the replicates can be viewed as proportions of the total number of observations to be taken at the design points. A continuous design, ξ, is characterized by the number of support or design points, k, their locations $\boldsymbol{x}_i \in \mathbb{R}^{n_x},\ i \in [k]$, from a user-specified design space $\mathbf{X}$ and the proportion of the total number of observations, w_i, to assign at each design point. Clearly, $w_i \in (0, 1)$, and $w_1 + w_2 + \cdots + w_k = 1$. In practice, continuous designs are implemented by taking roughly $n \times w_i$ replicates at level $\boldsymbol{x}_i,\ i \in [k]$ after rounding $n \times w_i$ to an integer subject to the constraint $n \times w_1 + \cdots + n \times w_k = n$. Advantages of working with continuous designs are many, and there is a unified framework for finding optimal continuous designs for M-bODE problems when the design criterion is a convex function on the set of all approximate designs (Fedorov, 1980). In particular, the optimal design problem can be formulated into a mathematical optimization program with convex properties and equivalence theorems are available to check the optimality of a design in a practical way.

Consider a continuous design ξ with k points, where the weight of the ith design point $\boldsymbol{x}_i^{\mathrm{T}} = (x_{i,1}, \dots, x_{i,n_x})$ is w_i, with $\sum_{i=1}^{k} w_i = 1$. We identify such a design with the discrete probability measure $\xi = \sum_{i=1}^{k} w_i\ \delta_{\boldsymbol{x}_i}$, and simply represent it by a list of k vectors $(\boldsymbol{x}_i^{\mathrm{T}}, w_i),\ i \in [k]$. Under mild regularity assumptions of the probability density function of the observations and design ξ, the variance–covariance matrix of Maximum Likelihood Estimates is well approximated by the inverse of the Fisher Information Matrix (FIM) and attains the lower bound in Cramér–Rao inequality (Rao, 1973). Consequently, one can view an optimal design problem as an allocation scheme of the covariates to minimize in some sense the variance–covariance matrix. Given the relation between the variance–covariance matrix and the FIM, the worth of the design ξ is measured by its FIM, which is the matrix with elements equal to the negative of the expectation of the second order derivatives of the log-likelihood of all observed data with respect to the parameters. When responses are independent, the normalized FIM of a continuous design ξ is

$$\mathcal{M}(\xi, \boldsymbol{p}) = -\mathbb{E}\left[\frac{\partial}{\partial \boldsymbol{p}}\left(\frac{\partial \mathcal{L}(\xi, \boldsymbol{p})}{\partial \boldsymbol{p}^{\mathrm{T}}}\right)\right] = \int_{\mathbf{X}} M(\delta_{\boldsymbol{x}_i}, \boldsymbol{p})\xi(d\boldsymbol{x}) = \sum_{i=1}^{k} w_i\ M(\delta_{\boldsymbol{x}_i}, \boldsymbol{p}). \tag{2}$$

Here, $\mathcal{L}(\xi, \boldsymbol{p})$ is the log-likelihood function of the observed responses using design ξ, $\delta_{\boldsymbol{x}}$ is the degenerate design that puts all its mass at $\boldsymbol{x}$ and $\mathcal{M}(\xi, \boldsymbol{p})$ is the *global* FIM from the design ξ for making inference on the parameter vector $\boldsymbol{p}$. Sometimes, the term *global* FIM is also referred to as the *total* FIM, or simply, FIM and, the FIM from a degenerate design is called *elemental*.

In what is to follow, we show our proposed semidefinite programming based approach is well suited to solve minimax optimal design problems. The methodology requires the design space $\mathbf{X}$ be discretized into many points. Let $\mathbb{X}$ be the discretized version of $\mathbf{X}$ with say q points. A common and simple way to discretize $\mathbf{X}$ is to use a grid set with equally-spaced points $\Delta \boldsymbol{x}$ units apart on each of the design spaces for all the regressor variables. We then search a probability measure χ on $\mathbb{X}$ so that

$$\sum_{\boldsymbol{x} \in \mathbb{X}} M(\delta_{\boldsymbol{x}}, \boldsymbol{p})\ \chi(\boldsymbol{x})$$

approximates (2) as close as possible.

When errors are normally and independently distributed, the volume of the asymptotic confidence region of $\boldsymbol{p}$ is proportional to $\det[\mathcal{M}^{-1/2}(\xi, \boldsymbol{p})]$, and so maximization of the determinant of the FIM leads to the smallest possible volume. Other design criteria maximize the FIM in different ways and are usually formulated as a convex function of the FIM. For example, when $\boldsymbol{p}$ is fixed, the locally $D-$, $A-$ and $E-$optimal designs are each, respectively, defined by

$$\xi_D = \arg\min_{\xi \in \Xi} \left\{\det[\mathcal{M}(\xi, \boldsymbol{p})^{-1}]\right\}^{1/n_p}, \tag{3}$$

$$\xi_A = \arg\min_{\xi \in \Xi} \left\{\mathrm{tr}[\mathcal{M}(\xi, \boldsymbol{p})^{-1}]\right\}, \tag{4}$$

$$\xi_E = \arg\min_{\xi \in \Xi} \left\{1/\lambda_{\min}[\mathcal{M}(\xi, \boldsymbol{p})]\right\}. \tag{5}$$

Here $\lambda_{\min}(\lambda_{\max})$ is the minimum (maximum) eigenvalue of the FIM, and Ξ is the set of all designs on $\mathbf{X}$. We note that $1/\lambda_{\min}[\mathcal{M}(\xi, \boldsymbol{p})] = \lambda_{\max}[\mathcal{M}(\xi, \boldsymbol{p})^{-1}]$, and by continuity, the design criteria (3)–(5) are defined as $+\infty$ for designs with singular information matrices.

More generally, Kiefer (1974) proposed a class of positively homogeneous design criteria defined on the set of symmetric $n_p \times n_p$ positive semidefinite matrices $\mathbb{S}^+_{n_p}$. If $\mathcal{M}(\xi, \boldsymbol{p}) \in \mathbb{S}^+_{n_p}$, this class is

$$\Phi_\delta[\mathcal{M}(\xi, \boldsymbol{p})^{-1}] = \left[\frac{1}{n_p}\ \mathrm{tr}(\mathcal{M}(\xi, \boldsymbol{p})^{-\delta})\right]^{1/\delta}, \tag{6}$$

where $\delta \geq -1$ is a parameter.

We note that Φ_δ is proportional to (i) $[\mathrm{tr}(\mathcal{M}(\xi, \boldsymbol{p})^{-1})]$, which is $A-$optimality when $\delta = 1$; (ii) $1/\lambda_{\min}[\mathcal{M}(\xi, \boldsymbol{p})]$, which is $E-$optimality when $\delta = +\infty$; and (iii) $[\det[\mathcal{M}(\xi, \boldsymbol{p})]^{-1}]^{1/n_p}$, which is $D-$optimality when $\delta \to 0$.

Because these criteria are convex on the space of information matrices, the global optimality of any design ξ in $\mathbf{X}$ can be verified using equivalence theorems, see for example Whittle (1973), Kiefer (1974) and Fedorov (1980). They are derived from directional derivative considerations and have a general form, with each convex criterion having its own specific form. For instance, the equivalence theorems for $D-$ and $A-$optimality are as follows: (i) ξ_D is locally D-optimal if and only if

$$\mathrm{tr}\left\{[\mathcal{M}(\xi_D, \boldsymbol{p})]^{-1}\ \mathcal{M}(\delta_{\boldsymbol{x}_i}, \boldsymbol{p})\right\} - n_p \leq 0, \quad \forall \boldsymbol{x} \in \mathbf{X}; \tag{7}$$

and (ii) ξ_A is locally A-optimal if and only if

$$\mathrm{tr}\left\{[\mathcal{M}(\xi_A, \boldsymbol{p})]^{-2}\ \mathcal{M}(\delta_{\boldsymbol{x}_i}, \boldsymbol{p})\right\} - \mathrm{tr}\left\{[\mathcal{M}(\xi_A, \boldsymbol{p})]^{-1}\right\} \leq 0, \quad \forall \boldsymbol{x} \in \mathbf{X}. \tag{8}$$

We call the functions on the left side of the inequalities in (7) and (8) *dispersion functions*.

2.1. Minimax designs

Nonlinear models are common in many areas with typical applications ranging from engineering to pharmacokinetics. For such models, the FIM depends on the parameters and consequently all design criteria, which are dependent on the FIM, depend on the unknown parameters that we want to estimate. When nominal values are assumed for these parameters, the resulting designs are termed locally optimal. The design strategies commonly used to handle the dependence noticed above include the use of: (i) a sequence of locally optimal designs, each computed using the latest estimate of $\boldsymbol{p}$; (ii) Bayesian designs that optimize the expectation of the optimality criterion value averaged over the prior distribution of model parameters $\boldsymbol{p}$ in $\mathbf{P}$ (Chaloner and Larntz, 1989); (iii) minimax designs that minimize the maximal value of the criterion from the unknown values of the model parameters in $\mathbf{P}$ (Wong, 1992). In either case the prior distribution and the set $\mathbf{P}$ are assumed known. Here we focus on finding minimax optimal designs.

Minimax design criteria have many forms and in this paper, we assume that there is *a priori* minimal knowledge of the true values of the parameters. One scenario is that the user has a plausible range of values for each model parameter. A reasonable goal is to find a design that maximizes the minimal accuracy for the parameter estimates regardless which set of model parameter values in the plausible region is the truth. The maximin, or equivalently, the minimax Φ_δ-optimal design sought is the one that minimizes

$$\max_{\boldsymbol{p}\in\mathbf{P}} \Phi_\delta[\mathcal{M}(\xi, \boldsymbol{p})^{-1}] \tag{9}$$

among all designs in Ξ. The minimax optimal design problems for $A-$, $E-$ and $D-$optimality criteria are constructed from (9) for $\delta = 1,\ +\infty$ and $\delta \to 0$, respectively, see (3)–(5). The mathematical program for problem (9) is

$$\min_{\xi\in\Xi}\ \max_{\boldsymbol{p}\in\mathbf{P}} \Phi_\delta[\mathcal{M}(\xi, \boldsymbol{p})^{-1}] \tag{10a}$$

$$\text{s.t} \sum_{i=1}^{k} w_i = 1. \tag{10b}$$

As an example, application of (10) to find a minimax $E-$optimal design yields:

$$\min_{\xi\in\Xi}\ \max_{\boldsymbol{p}\in\mathbf{P}} (\lambda_{\min}[\mathcal{M}(\xi, \boldsymbol{p})])^{-1} \tag{11a}$$

$$\text{s.t} \sum_{i=1}^{k} w_i = 1 \tag{11b}$$

and similar formulations apply for other criteria in the Φ_δ class.

Our minimax design problems remain convex and so equivalence theorems for verifying optimality of a minimax design can also be constructed. The reader is referred to Berger and Wong (2009) for details on the general equivalence theorems for a minimax type of optimality criteria. This is discussed more generally in Wong (1992), who gave an unified approach to constructing minimax optimal designs. Duarte and Wong (2014), among others, displayed the equivalence theorems for minimax $D-$, $A-$ and $E-$optimality and for space consideration, we do not discuss them in this paper. They are more complicated than the ones for locally optimal designs because sub-gradients are involved in the dispersion functions. These plots represent the dispersion functions and have a general pattern; if the design under investigation is optimal, the dispersion function must be bounded from above by zero with equality at every support point of the design. Fig. 2 in Section 4 display examples of such plots.

We next review programming based tools for solving such minimax problems. Finding the design ξ in (10a) is called the outer problem of the minimax program. The inner problem determines, for a given design, the values of the model parameters $\boldsymbol{p}$ that make the criterion least attainable. The overall optimization problem is complex because it poses several challenges: (i) the objective function of the outer problem is not differentiable, and consequently the subgradient used in the approximation may lead to many maximizers; (ii) the inner problem function can be sensitive and may require long computing time to solve it accurately; and (iii) global maxima of the inner problem are required to solve the minimax problem (Rustem et al., 2008).

2.2. Semidefinite programming

We use semidefinite programming to solve the outer problem by first discretizing the design space before applying a SDP solver to find the optimal design. Details on general use and application of SDP to search for optimal designs for linear models are available in Vandenberghe and Boyd (1996). Additional applications include finding (i) criterion-robust maximin designs for linear models (Filová et al., 2011), (ii) $D-$optimal designs for polynomial models and rational functions (Papp, 2012); and (iii) Bayesian optimal designs for nonlinear models (Duarte and Wong, 2015). This section reviews briefly the basics of this class of mathematical programs.

Let $\mathbb{S}^+_{n_p}$ be the space of $n_p \times n_p$ positive semidefinite matrices. A function $\varphi : \mathbb{R}^{m_1} \mapsto \mathbb{R}$ is called semidefinite representable (SDr) if and only if inequalities of the form $u \le \varphi(\boldsymbol{\zeta})$ can be expressed by *linear matrix inequalities* (LMI), see for example, Papp (2012) and Sagnol (2013). That is, $\varphi(\boldsymbol{\zeta})$ is SDr if and only if there exists some symmetric matrices $M_0, \ldots, M_{m_1}, \ldots, M_{m_1+m_2}$ such that

$$u \le \varphi(\boldsymbol{\zeta}) \iff \exists \boldsymbol{v} \in \mathbb{R}^{m_2} : \; u\, M_0 + \sum_{i=1}^{m_1} \zeta_i\, M_i + \sum_{j=1}^{m_2} v_j\, M_{m_1+j} \succeq 0. \tag{12}$$

Here, $\succeq$ stands for the ordering associated with the semidefinite cone where $A \succeq B$ holds if and only if $A - B \in \mathbb{S}^+_{n_p}$. For a given vector $\boldsymbol{c}$, the optimal values, $\boldsymbol{\zeta} \in \mathbb{R}^{m_1}$, of the SDr functions are found from *semidefinite programs* of the form:

$$\max_{\zeta} \left\{ \boldsymbol{c}^{\mathrm{T}}\, \boldsymbol{\zeta},\, \sum_{i=1}^{m_1} \zeta_i\, M_i - M_0 \succeq 0 \right\}. \tag{13}$$

In the design context, $\boldsymbol{c}$ is a vector of known constants that depends on the design problem and the matrices M_i, $i \in [m_1]$ contain the local FIM's and other matrices produced by the reformulation of the functions φ. The decision variables in the vector $\boldsymbol{\zeta}$ contain the weights w_i, $i \in [q]$ of the optimal design and optimized values for the auxiliary variables introduced, and q is pre-determined by the discretization scheme. The outer problem corresponds to finding a design for a finite set of parameter combinations $\boldsymbol{p}$ found by solving (13) subject to the linear constraints on the weights such that they are non-negative and sum to unity.

Ben-Tal and Nemirovski (2001, Chap. 2–3) provides a list of SDr functions useful in SDP formulations for solving M-bODE problems, see Boyd and Vandenberghe (2004, Sec. 7.3) for the basis. Sagnol (2013) showed that each criterion in the Kiefer's class of optimality criteria in (6) is SDr for all rational values of $\delta \in [-1, +\infty)$ where $\delta = 0$ is taken to mean the limiting case when $\delta \to 0$, i.e., $D-$optimality criterion. We note that when there are a finite number of α SDr functions $\varphi_1, \ldots, \varphi_\alpha$, then $\min_{i=1,\ldots,\alpha}\{\varphi_i\}$ is also SDr, implying that one can formulate the worst-case scenario problem by a semidefinite representable hypograph.

2.3. Nonlinear programming

Nonlinear programming seeks to find the optimum of a mathematical program where some of the constraints or the objective function are nonlinear. A general nonlinear program has the following form:

$$\max_{\boldsymbol{p}\in\mathbf{P}} f(\boldsymbol{p}) \tag{14a}$$

$$\text{s.t}\; \boldsymbol{g}(\boldsymbol{p}) \le 0 \tag{14b}$$

$$\boldsymbol{h}(\boldsymbol{p}) = 0 \tag{14c}$$

where $f(\boldsymbol{p})$ is a linear/nonlinear function, $\boldsymbol{g}(\boldsymbol{p})$ is a set of inequality constraints which may be linear and/or nonlinear and $\boldsymbol{h}(\boldsymbol{p})$ is a set of equality constraints. Here, $\boldsymbol{p}$ are variables to be optimized from a known compact set $\mathbf{P}$. We use nonlinear programming to solve the inner problem of (9) for each fixed design ξ. Our objective function is $\Phi_\delta[\mathcal{M}(\xi, \boldsymbol{p})^{-1}]$ criterion and the solution to the inner problem is the vector of parameters that result in the ξ being least efficient.

Finding the solution of the nonlinear program (14) is the most difficult step in the algorithm (Pázman and Pronzato, 2014) because it requires finding globally optimal solutions in the compact domain $\mathbf{P}$ to guarantee the convergence of the upper bound as we will show in Section 3.

There are several algorithms commonly used to solve NLP problems and they include the following: General Reduced Gradient (GRG) (Drud, 1985, 1994), Sequential Quadratic Programming (SQP) (Gill et al., 2005), Interior-Point (IP) (Byrd et al., 1999), and Trust-Region (Coleman and Li, 1994). Ruszczyński (2006) provides an overview of NLP algorithms. The NLP solvers employed in our proposed algorithm do not guarantee *global optimality*, only *local optimality*. This is in contrast to those based on Lipschitz global optimization techniques that guarantee convergence but are a lot less computationally efficient. There are heuristics, such as resorting to multiple restarts or simulated annealing that can be used to find a "good" local optimum, but in general it would require the use of branch and bound techniques to prove global optimality (Sahinidis, 2014). Most of Lipschitz global optimization solvers are based on branch and bound techniques or stochastic procedures, and may require long CPU times. Consequently, we avoid them for practical reasons, and use a gradient-based local NLP solver. These tools might be unable to find global optima but they guarantee that local optima can be found in mild computational time, and since the problems are of low dimension we believe that in most cases they coincide with the global optima. To increase the efficiency and accuracy of the NLP solver we use an automatic differentiation tool to generate the jacobian analytically. With the same purpose, we developed a global nonlinear programming tool based on a multistart heuristic algorithm that was compared with the NLP solver.

3. Algorithms

This section describes our proposed algorithm for finding minimax Φ_δ-optimal designs over Ξ, or more generally, over $\Xi_{A,\mathbf{b}}$ where $\Xi_{A,\mathbf{b}} := \{\{\boldsymbol{x}_i, w_i\}_{i=1,\dots,k} : \mathbf{w} \geq \mathbf{0},\ \sum_i w_i = 1,\ A\mathbf{w} \leq \mathbf{b}\}$, A is a matrix and $\mathbf{b}$ is a vector. In Section 3.1 we formulate and solve the outer optimization problem using SDP, and in Section 3.2 we tackle the inner optimization problem using a NLP solver.

The minimax optimal design problem is handled in program (10) after the entire covariate design space $\mathbf{X}$ is discretized. The optimization problem has finitely many variables to be optimized over a given feasible set described by infinitely many constraints, one for each $\boldsymbol{p} \in \mathbf{P}$ (López and Still, 2007). This falls into the class of semi-infinite programming, where Hettich and Kortanek (1993) and López and Still (2007) each provides a survey of the theory, applications and recent developments. The algorithms employed to solve SIP problems fall into three classes: (i) exchange methods; (ii) discretization based methods; and (iii) local reduction based methods (Hettich et al., 2001). Here, we use an exchange based procedure similar to the one proposed by Blankenship and Falk (1976), and further exploited by Žakovíc and Rustem (2003) and Mitsos and Tsoukalas (2015) among others.

To convert (10) into an equivalent semi-infinite program we note that for a given value of $\tau_{LB} \in \mathbb{R}$:

$$\max_{\boldsymbol{p}\in\mathbf{P}} \Phi_\delta[\mathcal{M}(\xi, \boldsymbol{p})^{-1}] \leq \tau_{LB} \iff \Phi_\delta[\mathcal{M}(\xi, \boldsymbol{p})^{-1}] \leq \tau_{LB},\ \forall \boldsymbol{p} \in \mathbf{P}. \tag{15}$$

Accordingly, the equivalent semi-infinite program obtained using a relaxation procedure is (Shimizu and Aiyoshi, 1980):

$$\min_{\xi\in\Xi,\tau_{LB}\in\mathbf{T}} \tau_{LB} \tag{16a}$$

$$\text{s.t } \Phi_\delta[\mathcal{M}(\xi, \boldsymbol{p})^{-1}] \leq \tau_{LB}, \quad \forall \boldsymbol{p} \in \mathbf{P} \tag{16b}$$

$$\sum_{i=1}^{k} w_i = 1 \tag{16c}$$

where $\mathbf{T} = [\tau^L, \tau^U] \in \mathbb{R}$ is a closed interval with τ^L being a large negative value, and τ^U a large positive one. The semi-infinite program (16) is called the *master problem* when $\mathbf{P}$ is infinite. One advantage of our approach is that additional linear constraints on the design weights, if any, can be handled in a straightforward fashion. Indeed, if a minimax optimal design is sought over $\Xi_{A,\mathbf{b}}$, one must simply add the following constraints into the master problem:

$$A\mathbf{w} \leq \mathbf{b}. \tag{16d}$$

When $\mathbf{P}$ is replaced by a finite subset $\mathbb{P}$ of $\mathbf{P}$, we obtain an approximation of program (16) which is commonly designated a *restricted master problem* (RMP). By construction, the optimal solution of this RMP provides a lower bound of the optimal value of (16), see for example Mutapcic and Boyd (2009), and our strategy is to solve the RMP using semidefinite programming. A *delayed constraint generation method* is used to solve the original problem, which has an infinite number of constraints. We do so by approximating the problem by a finite set of points $\mathbb{P}$ sampled from $\mathbf{P}$. The constraints might be saturated at the optimum (Terry, 2009) and the term "delayed" means that not all constraints are present at the beginning. Initially, the set $\mathbb{P}$ is populated with random points and at each iteration that follows, additional vectors $\boldsymbol{p}$ that make the SDP-generated design least efficient are included. The optimization problem (16) is solved iteratively, alternating between two phases as follows.

The proposed algorithm has similarities with the approach used in Pronzato and Walter (1988) and Walter and Pronzato (1997, Sec. 6.4.4) to find minimax designs. Here, we start the convergence with a finite number of vectors $\mathbb{P}$, not only one as in the latest reference. Similarly to Walter and Pronzato (1997, Sec. 6.4.4), the outer and the inner optimization problems are solved iteratively to convergence. The inner program is formulated as a NLP and serves to find the vector of parameters at which a locally optimal design is less efficient; the outer program, formulated as a SDP in a previously discretized design space, is to determine the optimal design for a finite set $\mathbb{P}$ that replaces $\mathbf{P}$. The vectors of parameters that solve the inner problem for a given locally optimal design are appended to the set of instances $\mathbb{P}$ to increasingly constraining the design. Notice that $\mathbb{P}$ replacing $\mathbf{P}$ changes at each iteration. Suppose that the initial set $\mathbb{P}^{(0)}$ has m_0 elements and at the jth iteration, this set becomes $\mathbb{P}^{(j)}$ and has $m_0 + j$ elements. This assumes that we add one element from $\mathbf{P}$ to the original finite set $\mathbb{P}^{(0)}$ at each iteration, and this element is the worst $\boldsymbol{p}$ for a given local optimum design ξ. The algorithm iterates between the outer program $\mathcal{P}_1$ and the inner problem $\mathcal{P}_2$ until convergence. If $\xi^{(0)}$ is the initial design and $\xi^{(j-1)}$ is the optimal design at the $(j-1)$th iteration, we solve

- the Phase 1 outer problem $\mathcal{P}_1$ (16) by finding a design $\xi^{(j)}$ and a lower bound of the minimax program, $\tau_{LB}^{(j)}$ after $\mathbf{P}$ is replaced by $\mathbb{P}^{(j-1)}$, and
- the Phase 2 inner problem $\mathcal{P}_2$ by finding the set of model-parameter values $\boldsymbol{p}^{(j)}$ that yields the worst (largest) possible value of the criterion $\Phi_\delta[\mathcal{M}(\xi, \boldsymbol{p})^{-1}]$ for $\xi^{(j)}$. The solution provides an upper bound $\tau_{UB}^{(j)}$ for the minimax problem and iterates after adding $\boldsymbol{p}^{(j)}$ to $\mathbb{P}^{(j-1)}$ to obtain

  $$\mathbb{P}^{(j)} = \mathbb{P}^{(j-1)} \cup \boldsymbol{p}^{(j)}, \tag{17}$$

  for the next iteration.

The above algorithm stops when the two bounds converge and reaches an ϵ-optimal solution, where ϵ is a user-specified small positive constant. In practice this means that after each iteration one verifies whether the condition

$$\left| \frac{\tau_{UB}^{(j)} - \tau_{LB}^{(j)}}{\tau_{UB}^{(j)}} \right| \le \epsilon, \tag{18}$$

is satisfied. Blankenship and Falk (1976) showed that the procedure is guaranteed to converge to the global optimum in a finite number of iterations. Moreover, we shall see in Proposition 1 that the stopping criterion (18) ensures a lower bound on the efficiency of the returned design.

Our algorithm has similarities with algorithms developed by Melas (2006, Sec. 8.7.1) using the functional approach for determining maximin and standardized maximin optimal designs. The key idea is that the optimal support points and the associated optimal weights can be treated as functions of the estimated parameters and, as such, can be developed into a Taylor series around the nominal values of these parameters. The author uses standard arguments to prove that a sequence of locally optimal designs obtained for a given discrete set of parameter combinations contains a weakly convergent subsequence with a limit, and the standardized maximin optimality can be checked with a general equivalence theorem, see Dette et al. (2007). The approach is elegant and behaves well numerically but because it exploits problem specificities may be somewhat limited for general applications.

To describe our algorithm in more detail, we need additional notation. Let $\upsilon(\mathbf{P})$ and $E(\mathbf{P})$ be the sets of vertices and edges of $\mathbf{P}$, respectively. Let m_0 be the user-specified sample size of $\mathbb{P}^{(0)}$ and let $\mathcal{U}(\mathbf{Q})$ be the uniform distribution over the general compact domain $\mathbf{Q}$ enclosing a general sub-domain of the parameters plausible region. Let $m_{0,k} \le m_0$ be the number of samples $\boldsymbol{p}$ including the vertices and other points randomly generated by sampling $E(\mathbf{P})$. Further, let $m_{0,r} = m_0 - m_{0,k}$ be the remaining number of samples that are generated by sampling randomly from the overall domain $\mathbf{P}$ and not only from the edges $E(\mathbf{P})$. The sampling procedure uses a uniform random number generator to draw elements from the closed domain $\mathbf{Q}$, see Press et al. (2007, Chap. 7).

We use an empirical relation based on the structure of $\mathbf{P}$ to set the value of $m_{0,k}$. Since $\mathbf{P}$ is formed from the cartesian product of intervals for each of the n_p parameters, the compact domain contains 2^{n_p} vertices and $n_p\, 2^{n_p-1}$ edges. We then set $m_{0,k} = 2^{n_p} + n_p\, 2^{n_p-1}$, and consequently $m_{0,r} = m_0 - 2^{n_p} - n_p\, 2^{n_p-1}$. The collection of data samples in $\mathbb{P}^{(0)}$ is constructed as follows:

$$\mathbb{P}^{(0)} = \left\{\boldsymbol{p}_1, \ldots, \boldsymbol{p}_{m_0}\right\} \tag{19}$$

where $\boldsymbol{p}_1, \ldots, \boldsymbol{p}_{2^{n_p}}$ are the vertices of $\mathbf{P}$, $\boldsymbol{p}_{2^{n_p}+1}, \ldots, \boldsymbol{p}_{m_{0,r}}$ are sampled from the uniform distribution $\mathcal{U}(E(\mathbf{P}))$ with one point from each edge, and $\boldsymbol{p}_{m_{0,r}+1}, \ldots, \boldsymbol{p}_{m_0}$ are drawn from $\mathcal{U}(\mathbf{P})$. The sampling scheme is similar to that used in the H-algorithm proposed by Fackle-Fornius et al. (2015) to first find a prior distribution for constructing Bayesian optimal designs and then use them to check the optimality of minimax standardized designs.

All our examples in Section 4 use $m_0 = 50$ but other values can be used. The above rule is empirical and seems to work well for finding the optimal designs sought here. We reason that the construction of $\mathbb{P}^{(0)}$ relies on the properties of $\Phi_\delta[\mathcal{M}(\xi, \boldsymbol{p})^{-1}]$ and our experience is that while this function is not convex nor concave (Pronzato and Pázman, 2013) over the parameter space, the vectors $\boldsymbol{p}$ for which a design is least efficient are usually the vertices of the plausible region or points located on one of the edges of $\mathbf{P}$. We exploit this feature and so include the vertices and some randomly chosen points of $E(\mathbf{P})$ in the initial set. We find that such a construction can sometimes reduce the computation time, especially when the optimum of the inner problem is one of the vertices.

Algorithm 1 summarizes how we combine SDP with NLP to find minimax optimal designs with accompanying details on the subproblems in Phases 1 and 2. Sections 3.1 and 3.2 provide additional details about the formulations and numerical solvers used in Algorithm 1. Here and throughout, we denote the design obtained after convergence by ξ^{OPT} and one set of worst-case values for the model parameter at convergence by $\boldsymbol{p}^{\text{OPT}}$.

Algorithm 1 Algorithm to find minimax optimal designs

```
procedure MinimaxOptimalDesign($\Delta\boldsymbol{x}, m_{0,r}, m_{0,k}, \tau^L, \tau^U, \epsilon, l_j, u_j, \xi^{OPT}, \boldsymbol{p}^{OPT}$)
    Construct $\mathbb{X}$ using intervals $\Delta\boldsymbol{x}$                ▷ Discretization of the design space
    Construct $\mathbb{P}^{(0)}$                ▷ Sample $m_0$ points from $\mathbf{P}$ according to (19)
    $j \leftarrow 1$                ▷ Initialize iterations counter
    $\tau_{LB}^{(0)} \leftarrow \tau^L,\ \tau_{UB}^{(0)} \leftarrow \tau^U$
    while $|(\tau_{UB}^{(j-1)} - \tau_{LB}^{(j-1)})/\tau_{UB}^{(j-1)}| > \epsilon$ do                ▷ Convergence checking
        Solve $\mathcal{P}_1$ to determine $\tau$ and $\xi$                ▷ SDP problem
        $\tau_{LB}^{(j)} \leftarrow \tau, \quad \xi^{(j)} \leftarrow \xi$                ▷ Update $\tau_{LB}^{(j)}$
        Solve $\mathcal{P}_2$ to determine $\tau$ and $\boldsymbol{p}$                ▷ NLP problem
        $\boldsymbol{p}^{(j)} \leftarrow \boldsymbol{p}$
        if $\tau < \tau_{UB}^{(j-1)}$ then                ▷ Update opt. design and worst parameter
            $\tau_{UB}^{(j)} \leftarrow \tau, \quad \boldsymbol{p}^{OPT} \leftarrow \boldsymbol{p}, \quad \xi^{OPT} \leftarrow \xi$
        else
            $\tau_{UB}^{(j)} \leftarrow \tau_{UB}^{(j-1)}$                ▷ Update $\tau_{UB}^{(j)}$
        end if
        $\mathbb{P}^{(j)} \leftarrow \mathbb{P}^{(j-1)} \cup \boldsymbol{p}^{(j)}$                ▷ Update $\mathbb{P}$
        $j \leftarrow j+1$                ▷ Update the iteration counter
    end while
end procedure
```

The minimax-criteria $\Psi(\xi) := \max_{\boldsymbol{p}\in\mathbf{P}} \Phi_\delta[\mathcal{M}(\xi, \boldsymbol{p})^{-1}]$ is nonnegative (and even positive at the optimum, cf. Pukelsheim (1993)) and positively homogeneous, so it is natural to define the efficiency of a design ξ by

$$\text{Eff}(\xi) = \frac{\Psi(\xi^{\text{OPT}})}{\Psi(\xi)},$$

where ξ^{OPT} is minimax-optimal. Upon convergence, the following proposition bounds the efficiency of the design returned by Algorithm 1:

Proposition 1. *Let ξ^{OPT} be the design returned by Algorithm 1, where we assume that each subproblem $\mathcal{P}_1$ and $\mathcal{P}_2$ is solved exactly (to global optimality). Then,* $\text{Eff}(\xi^{\text{OPT}}) \geq 1-\epsilon$. *More generally, if we assume that the subproblems $\mathcal{P}_1$ and $\mathcal{P}_2$ are solved within relative accuracy ε_L and ε_U, respectively, then the bound on the efficiency becomes*

$$\text{Eff}(\xi^{\text{OPT}}) \geq (1-\epsilon)\frac{1-\varepsilon_L}{1+\varepsilon_U}.$$

Proof. We start with the case where each subproblem is solved exactly. By construction, $\tau_{LB}^{(j)}$ and $\tau_{UB}^{(j)}$ are lower and upper bounds for $\Psi(\xi^{\text{OPT}})$, and at each iteration, the design ξ^{OPT} stored by the algorithm satisfies $\Psi(\xi^{\text{OPT}}) \leq \tau_{UB}^{(j)}$. Indeed, the value τ returned by $\mathcal{P}_2$ is the value of $\Psi(\xi^{(j)})$. It follows that

$$\text{Eff}(\xi) = \frac{\Psi(\xi^{\text{OPT}})}{\Psi(\xi)} \geq \frac{\tau_{LB}^{(j)}}{\tau_{UB}^{(j)}},$$

which is at least $1-\epsilon$ when the convergence criterion is satisfied. When the solutions to the subproblems $\mathcal{P}_1$ and $\mathcal{P}_2$ are approximate and not exact, the proof is similar. This follows by observing that $\Psi(\xi^{\text{OPT}}) \geq \tau_{LB}^{(j)}(1-\varepsilon_L)$ and $\Psi(\xi^{\text{OPT}}) \leq \tau_{UB}^{(j)}(1+\varepsilon_U)$. $\square$

Algorithm 1 has some similarities with that proposed by Duarte and Wong (2014), and the main differences are listed below:

(a) the Phase 1 problem is formulated here as a semidefinite program problem instead of a nonlinear problem that requires global optimization solvers. The use of SDP in Phase 1 allows us to find approximate designs in polynomial time but requires the discretization of the design space, here represented by a finite number of candidate points. In contrast to Duarte and Wong (2014), the algorithm herein does not require an initial estimate of the number of support points.

(b) the Phase 2 problem is partially similar to that used in Duarte and Wong (2014). Specifically, the calculation of the answering set for the optimal design produced by Phase 1 is also represented as a NLP problem, and solved using a similar method, but does not require iterating the number of support points until the equivalence theorem is validated. This task is highly time consuming since it requires finding the maxima of the dispersion function for a given answering set, which is a rather challenging NLP with multiple optima.

(c) the algorithm proposed herein can easily be extended to find minimax $A-$ and $E-$optimal designs by taking advantage of the semidefinite representability of those criteria; the one proposed in Duarte and Wong (2014) works only for the $D-$optimality criterion. In practice, Algorithm 1 can be extended to every semidefinite representable criterion used for designing a multiple regression model, see Sagnol (2011).

(d) the set $\mathbb{P}^{(0)}$ in our algorithm is formed by m_0 samples judiciously chosen from $\mathbf{P}$, whereas in Duarte and Wong (2014), $\mathbb{P}^{(0)}$ is formed by a singleton element $\boldsymbol{p}$. This approach potentially increases the convergence rate.

Algorithm 1 has common features with that proposed by Pázman and Pronzato (2014) for finding extended $E-$ and $G-$optimal designs for protection against close-to-overlapping situations, and recently extended to find $D-$, $A-$ and E_k-optimal designs for linear models (Burclová and Pázman, 2016). Here, we formulate the Phase 1 problem for finding optimal designs on a discrete design space as a SDP problem and exploit the semidefinite representability of $D-$, $A-$ and $E-$optimality criteria. This is in contrast with the work of Pázman and Pronzato (2014) where the problem is handled by Linear Programming. The SDP formulation is harder to solve than the LP problem because the former requires specific Interior Point algorithms (Boyd and Vandenberghe, 2004), but because it is broader in scope and in particular, the formulation can be generalized to include all problems with a SDr criterion. Further, we solve the Phase 2 problem as a NLP and consequently the convergence of the successive cutting planes that generate the upper bounds is guaranteed if global optimization techniques are used. This desirable property is not found in the method proposed by Pázman and Pronzato (2014), where they used a combination of grid search and local optimization.

3.1. Phase 1 problem - $\mathcal{P}1$

The SDP formulation for solving the problem $\mathcal{P}_1$ and obtaining the successive global lower bounds for the minimax program (10) are as follows. At the jth iteration, the goal is to find the design $\xi^{(j)}$ with weights $\boldsymbol{w}^{(j)} = (w_1^{(j)} \dots, w_q^{(j)})^{\mathrm{T}}$ in the following SDP problem:

$$\min_{\xi^{(j)}\in\Xi,\tau_{LB}^{(j)}\in\mathbf{T}} \quad \tau_{LB}^{(j)} \tag{20a}$$

$$\text{s.t } \Phi_\delta[\mathcal{M}(\xi^{(j)},\boldsymbol{p})^{-1}] \le \tau_{LB}^{(j)}, \quad \forall \boldsymbol{p}\in\mathbb{P}^{(j)} \tag{20b}$$

$$\sum_{i=1}^{q} w_i^{(j)} = 1, \tag{20c}$$

where for our interests, we set $\Phi_\delta[\mathcal{M}(\xi^{(j)},\boldsymbol{p})^{-1}]$ to be either $[\mathrm{tr}(\mathcal{M}(\xi^{(j)},\boldsymbol{p})^{-1})]$ for $A-$optimality, $(\lambda_{\min}[\mathcal{M}(\xi^{(j)},\boldsymbol{p})])^{-1}$ for $E-$optimality, $[\det[\mathcal{M}(\xi^{(j)},\boldsymbol{p})]^{-1}]^{1/n_p}$ for $D-$optimality, and each $w_i^{(j)}$ is the weight of the point $\boldsymbol{x}_i^{\mathrm{T}} \in \mathbb{X}$, $i \in [q]$. As we mentioned for the master problem (16), additional constraints of the form $A\mathbf{w} \le \mathbf{b}$ can be added to the above SDP formulation, for the case of optimization over $\Xi_{A,\mathbf{b}}$. To handle the SDP problems, there are user-friendly interfaces, such as `cvx` (Grant et al., 2012) or Picos (Sagnol, 2012), that automatically transform the constraints (20b) into a series of LMIs before passing them to SDP solvers such as `SeDuMi` (Sturm, 1999) or `Mosek` (Andersen et al., 2009). This is possible if Φ_δ is SDr, which is true for our design criteria of interest. In our work, we solved all SDP problems using the `cvx` environment combined with the solver `Mosek` that uses an efficient interior point algorithm.

3.2. Phase 2 problem - $\mathcal{P}2$

This problem is to find the vector $\boldsymbol{p}$ given the design determined in Phase 1 which results in it having the smallest efficiency. The problem is non-convex and requires nonlinear programming tools. Its solution generates successive global upper bounds for the minimax optimal design problem, with one obtained for each locally optimal design. Let us assume that at the jth iteration the design $\xi^{(j)}$ is already determined after solving (20). The problem to solve in this phase is

$$\tau = \max_{\boldsymbol{p}\in\mathbf{P}} \ \Phi_\delta[\mathcal{M}(\xi^{(j)},\boldsymbol{p})^{-1}], \tag{21}$$

and we use a NLP solver `IPOPT` based on an interior point algorithm (Wächter and Biegler, 2005). To increase the accuracy of the computed gradient, we use an automatic differentiation tool, `ADiMat` (Bischof et al., 2002), where its analytical representation is then passed to the solver to handle the program (21). We notice that `IPOPT` is a local nonlinear programming solver, and the problem (21) may have multiple local optima. To handle this issue we develop a global nonlinear programming tool based on a multistart heuristic algorithm similar to that proposed by Ugray et al. (2005), and in Section 4.1 we compare the local NLP with our algorithm.

In every iteration we assumed that the optimum found by solving $\mathcal{P}_2$ is global and τ is a global upper bound for the minimax problem. The optimal $\boldsymbol{p}$ is used to update the set of data samples in $\mathbb{P}^{(j)}$ employing the rule (17). In many situations, $\epsilon-$optimum is found in the first iteration by the SDP solver, and $\mathcal{P}_2$ is used to compute the upper bound, and hence, to *confirm* the optimality of the design.

4. Results

In this section, we apply our proposed algorithm in Section 3 to find minimax $A-$, $E-$ and $D-$optimal designs for the power logistic model, the 4-parameter logistic model and the general consecutive reaction model that represents the reactants concentration in a continuous stirred tank. The second is also called the Emax model or the Hill model. Such logistic models are commonly used to study the drug effects as the dose varies, see for example (Ting, 2006), and in other areas such as agronomy (Tsoularis and Wallace, 2002; Ludena et al., 2007). The third is described by a system of evolutive ordinary differential equations, and puts additional issues concerning the computation of the FIM. Except for the minimax $D-$optimal designs for the logistic model, the other optimal designs, as far as we know, are not published in the literature. The logistic model is considered a benchmark test for algorithms for finding optimal designs of experiments, and we use it here for comparison. First, we find minimax $D-$optimal designs for the logistic model for different uncertainty parameter regions and compare the results with those of Duarte and Wong (2014). We then apply the algorithm to find minimax $A-$ and $E-$optimal designs for all models. In all cases, we assume that there is a known plausible set of values for each parameter.

All computation in this paper were carried using on an Intel Core i7 machine (Intel Corporation, Santa Clara, CA) running 64 bits Windows 10 operating system with 2.80 GHz. The relative and absolute tolerances used to solve the SDP problems were set to 10^{-5} in all problems. Similarly, the relative and absolute tolerances of the NLP solver were also set to 10^{-5} in all problems. The value of ϵ used in the convergence criterion in Algorithm 1 is 10^{-4}.

4.1. Example 1 - Logistic model

We consider the power logistic model proposed by Prentice (1976) and commonly used in dose–response studies when the outcome is binary:

$$\mathbb{E}[y|x,\boldsymbol{p}] = \frac{1}{\{1+\exp[-\beta\,(x-\mu)]\}^s}, \quad x \in \mathbf{X}, \quad \boldsymbol{p} \in \mathbf{P} \tag{22}$$

where $\mathbf{X}$ is a used-defined bounded interval, $\boldsymbol{p} = [\beta,\ \mu,\ s]^{\mathrm{T}}$ is the vector of model parameters assumed to lie in a known compact set $\mathbf{P}$. The probability of a response at dose $x \in \mathbf{X}$ is $y(x,\boldsymbol{p})$ and the binary outcome is coded as 1 for response and 0 otherwise. When $s = 1$, we have the logistic model, other values of s provide more flexibility in capturing skewness and kurtosis, but for comparison purposes we first assume the simpler case when $s = 1$. Accordingly, let $\boldsymbol{p} = [\beta,\ \mu]^{\mathrm{T}}$ and assume that $\mathbf{P} = [\mu_L,\mu_U] \times [\beta_L,\beta_U]$ which contains all plausible values of the two parameters. Here, μ_L is the lower bound of μ and μ_U is its upper bound. Similarly, β_L is the lower bound of β and β_U is its upper bound. The FIM of the design at the point x_i is $M(x_i,\boldsymbol{p}) = h(x_i,\boldsymbol{p})\,h(x_i,\boldsymbol{p})^{\mathrm{T}}$, where

$$h(x_i,\boldsymbol{p}) = \frac{1}{\sqrt{\mathbb{E}[y|x_i,\boldsymbol{p}]\,(1-\mathbb{E}[y|x_i,\boldsymbol{p}])}}\left(\frac{\partial\mathbb{E}[y|x_i,\boldsymbol{p}]}{\partial\boldsymbol{p}}\right), \quad \frac{\partial\mathbb{E}[y|x_i,\boldsymbol{p}]}{\partial\boldsymbol{p}} = \begin{pmatrix} \dfrac{\partial\mathbb{E}[y|x_i,\boldsymbol{p}]}{\partial\beta} \\ \dfrac{\partial\mathbb{E}[y|x_i,\boldsymbol{p}]}{\partial\mu} \end{pmatrix}.$$

The initial population in $\mathbb{P}^{(0)}$ of the numerical tests presented in this section was constructed using the 2^2 vertices plus 16 points randomly sampled from the edges of $\mathbf{P}$ with one per edge. What we meant is that we sample 16 points uniformly, each on an edge selected at random. More precisely, we use the following procedure to sample a vector $\boldsymbol{p}$ on an edge of the plausible region: we first sample one index j uniformly in $\{1,\ldots,n_p\}$. Then, we sample p_j uniformly at random in the interval $[l_j,u_j]$, and for $i = 1,\ldots,n_p, i \neq j$, we draw p_i from the bounds $\{l_i,\ u_i\}$ using a binary random number generator. In addition, we randomly sample 30 more points from $\mathbf{P}$ using a uniform random number generator, resulting in a total of 50 vectors for $\boldsymbol{p}$.

In our analysis we first study the impact of the size of the plausible parametric region on the optimal design and the computation time. We consider $\mathbf{X} \equiv [-1,5]$ and four different regions for $\boldsymbol{p}$ that result from combining the intervals $[1.0,\ 1.25]$ and $[1.0,\ 3.0]$ for β with the intervals $[0.0,\ 1.0]$ and $[0.0,\ 3.5]$ for μ. Next, we study the impact of the design space range and solve the problem for $\mathbf{X} \equiv [3.0,\ 9.0]$ with $\mu \in [6.0,\ 8.0]$ and $\beta \in [1.0,\ 1.25]$ and $\beta \in [1.0,\ 3.0]$, respectively. In all these tests, the design space was discretized using equally spaced points $\Delta x = 0.02$ unit apart, and so for each of the design spaces $\mathbf{X} \equiv [-1,5]$ and $[3.0,\ 9.0]$, we have 301 candidate design points.

Table 1 presents the $D-$optimal designs for different plausibility parameter regions. Our experience is that the size of $\mathbf{X}$ seems to have a small impact on the speed of our proposed method. The grid obtained after discretizing the design space seems to have impact on the computational time required by the Phase 1 problem. In practice, dense grids result in more nodes, and consequently larger semidefinite programs are produced, thus increasing the difficulty in solving the optimal design problem for a set of samples of parameters. The algorithm performs potentially well with a larger number of points but the SDP solver might be unable to provide a solution in realistic time. Having denser grids allows us to obtain more accurate optimal designs closer to those obtained if $\mathbf{X}$ is not discretized. In practice, one may increase the size of $\mathbf{X}$ by

Table 1
Minimax $D-$optimal designs for the logistic model on $\mathbf{X}$ discretized using $\Delta x = 0.02$, for different plausible regions.

$[\beta_L, \beta_U]$	$[\mu_L, \mu_U]$		
	[0.0, 1.0][b]	[0.0, 3.5][b]	[6.0, 8.0][a]
[1.0, 1.25]	(−0.84,0.3810) (−0.82,0.1190) (1.82,0.1190) (1.84,0.3810)	(−0.88,0.2746) (1.74,0.2254) (1.76,0.2254) (4.38,0.2746)	(5.16, 0.0048) (5.18,0.3428) (7.00, 0.3047) (8.82,0.3427) (8.84,0.0049)
$\boldsymbol{p}^{\text{OPT}}$	$[1.25, 1.0]^{\text{T}}$	$[1.25, 3.5]^{\text{T}}$	$[1.25, 8.0]^{\text{T}}$
CPU (s)	10.13	25.00	10.09
Eff	1.0000	1.0000	0.9999
$[\beta_L, \beta_U]$	$[\mu_L, \mu_U]$		
	[0.0, 1.0][b]	[0.0, 3.5][b]	[6.0, 8.0][a]
[1.0, 3.0]	(−0.54,0.2190) (−0.52,0.1421) (0.50,0.1193) (0.52,0.1612) (1.52,0.0514) (1.54,0.3070)	(−0.40,0.0491) (−0.38,0.1516) (0.68,0.2350) (1.52,0.0384) (1.54,0.0305) (2.02,0.0610) (2.82,0.1141) (2.84,0.1251) (3.88,0.0420) (3.90,0.1532)	(5.66, 0.2592) (6.86,0.2408) (7.14, 0.2408) (8.34,0.2592)
$\boldsymbol{p}^{\text{OPT}}$	$[3.0, 0.0]^{\text{T}}$	$[3.0, 3.5]^{\text{T}}$	$[3.0, 8.0]^{\text{T}}$
CPU (s)	10.08	52.13	49.35
Eff	0.9995	1.0000	1.0000

$(x.xx, w.wwww) \equiv$ (design point, weight).
[a] The design space is $\mathbf{X} \equiv [3, 9]$.
[b] The design space is $\mathbf{X} \equiv [-1, 5]$ for plausible regions.

keeping the size of the SDP problem the same by manipulating the discretization grid intervals to minimize the impact on computational time. The Phase 2 problem is independent on the design space because the search is made on the space of the parameters, and so has a small impact on the CPU time.

The resulting designs in Table 1 show good agreement with the designs found with a SIP-based approach which assumes $\mathbf{X}$ is continuous, see Duarte and Wong (2014). In some cases, the SDP-based minimax designs have more support points than the SIP-generated minimax designs. The additional points are usually close to each other and depend on how we discretize the design space. Table 1 shows that larger plausible regions tend to result in designs with more support points, a finding already observed by other authors for Bayesian and minimax designs, see for example, Chaloner and Larntz (1989) and Duarte and Wong (2014). The CPU times required by Algorithm 1 to find minimax $D-$optimal designs is, on average, 8.56 times shorter than that required in the SIP approach in Duarte and Wong (2014). The efficiency of each design in Table 1 is computed as:

$$\text{Eff} = \left(\frac{\det[\mathcal{M}(\xi^{\text{OPT}}, \boldsymbol{p}^{\text{OPT}})]}{\det[\mathcal{M}(\xi^{*}, \boldsymbol{p}^{*})]} \right)^{1/n_p} \tag{23}$$

where ξ^{OPT} is the optimal design obtained from the proposed algorithm, along with the worst-case parameter $\boldsymbol{p}^{\text{OPT}}$, and ξ^* is the design in Duarte and Wong (2014) for the corresponding answering set $\boldsymbol{p}^*$. We note that the efficiencies of the designs obtained from the proposed algorithm relative the SIP-generated designs are all between 0.9990 and 1.0000 and so this suggests our algorithm produces optimal or highly efficient minimax designs but using shorter CPU time. It is possible to find optimal designs with an efficiency below $1 - \epsilon$, because the reference design ξ^{OPT} in Proposition 1 is restricted to have support points on the grid and the designs ξ^* in Duarte and Wong (2014) are constructed assuming a continuous design space and so may perform better. The CPU time of the designs in Table 1 is larger for larger regions $\mathbf{P}$, one reason being the increasing complexity of the NLPs solved to find $\boldsymbol{p}^*$ in each iteration.

To analyze the impact of the grid density on the optimal design and CPU we consider $\mathbf{P} \equiv [1.0, 3.0] \times [0.0, 1.0]$, $\mathbf{X} \equiv [-1, 5]$, and discretize the design space using equally spaced points with $\Delta x = 0.08$ (76 points), $\Delta x = 0.04$ (151 points), $\Delta x = 0.02$ (301 points) and $\Delta x = 0.01$ (601 points), respectively. Table 2 shows that (i) the designs obtained from different grid sets are very close and have a similar efficiency; and (ii) the CPU times are alike, where the larger size semidefinite program is compensated by the higher convergence rate. Although the performance of the method does not deteriorate very much when the size of the grid increases (cf. Table 2), the SDP approach usually fails if it includes between 3000 and 6000 design points, mainly due to memory problems, depending on the hardware. In some multi-factor problems, it is not unusual

Table 2
Minimax $D-$optimal designs for the logistic model on $\mathbf{X} \equiv [-1, 5]$ and $\mathbf{P} \equiv [1.0, 3.0] \times [0.0, 1.0]$ when equally spaced grids of different sizes are used.

	Δx			
	0.08	0.04	0.02	0.01
	(−0.60,0.0542)	(−0.56,0.1265)	(−0.54,0.2190)	(−0.55,0.0760)
	(−0.52,0.3066)	(−0.52,0.2334)	(−0.52,0.1421)	(−0.54,0.2799)
	(0.44,0.0295)	(0.48,0.1552)	(0.50,0.1193)	(0.48,0.2052)
	(0.52,0.2512)	(0.52,0.1242)	(0.52,0.1612)	(0.49,0.0789)
	(1.48,0.1057)	(1.52,0.2422)	(1.52,0.0514)	(1.53,0.1942)
	(1.56,0.2528)	(1.56, 0.1183)	(1.54,0.3070)	(1.54,0.1658)
$\boldsymbol{p}^{\text{OPT}}$	$[3.0, 0.0]^{\text{T}}$	$[3.0, 0.0]^{\text{T}}$	$[3.0, 0.0]^{\text{T}}$	$[3.0, 0.0]^{\text{T}}$
CPU (s)	6.86	7.50	10.08	10.56
Eff	0.9169	0.9659	0.9995	0.9998

$(x.xx, w.wwww) \equiv$ (design point, weight).

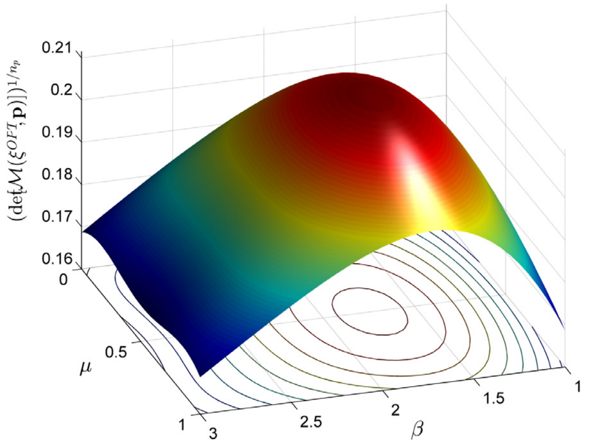

Fig. 1. Display of the objective function $(\det[\mathcal{M}(\xi^{\text{OPT}}, \boldsymbol{p})])^{1/n_p}$ vs. $\boldsymbol{p}$ for the optimal design obtained for $\mathbf{P} \equiv [1.0, 3.0] \times [0.0, 1.0]$, $\mathbf{X} \equiv [-1, 5]$ and the design space discretized using equally spaced points with $\Delta x = 0.02$ (in second line, first column of Table 1).

to have millions of design points and this approach cannot handle large design spaces unless coarser discretization grids are used.

Finally, to study the accuracy of the NLP solver to find global optima in our algorithm, we implemented a multistart heuristic algorithm and compared its performance with the local NLP solver. To briefly describe the algorithm, we set the number of starting points used, n_s, and generate n_s vectors of parameters $\boldsymbol{p}$ employing a uniform random generator mechanism in $\mathbf{P}$. Then, we solve the Phase 2 problem by calling `IPOPT` n_s times using a different starting point in each call. The optimum and the objective function are saved, and subsequently the best is(are) picked. If a singleton is found, $\mathbb{P}$ is updated with a single vector $\boldsymbol{p}$ while when more than one optimum is obtained all of them are included in $\mathbb{P}$, that augments more than a vector $\boldsymbol{p}$ per iteration. For comparison purposes we used $n_s = 20$, and for all the examples in Table 1 we obtained the same optimal designs found with the local NLP solver. Although, the CPU time required increased in average 4.18 times. We also observed that in most of the problems and iterations the optimum is a single point which is in agreement with our result in Table 1 where $\boldsymbol{p}^{\text{OPT}}$ is a singleton.

Now, we analyze the accuracy of our algorithm in determining the vector $\boldsymbol{p}$ where the criterion is least attainable. Fig. 1 displays the objective function, $(\det[\mathcal{M}(\xi^{\text{OPT}}, \boldsymbol{p})])^{1/n_p}$, for the optimal design obtained with the Algorithm 1 for $\mathbf{P} \equiv [1.0, 3.0] \times [0.0, 1.0]$, $\mathbf{X} \equiv [-1, 5]$ and the design space discretized using equally spaced points with $\Delta x = 0.02$, presented in second line, first column of Table 1. The plot shows that for the minimax design ξ^{OPT} the answering set is the singleton $\boldsymbol{p}^{\text{OPT}} = [3.0, 0.0]^{\text{T}}$.

Fig. 2(a) displays the dispersion function of the SDP-generated design under the D-optimality criterion for the logistic model when $\mathbf{X} \equiv [-1, 5]$ and the plausible region is $\mathbf{P} = [1.0, 3.0] \times [0.0, 1.0]$. The dispersion functions of the designs found by our algorithm at termination are constructed after the convergence condition is attained. Briefly, it requires the determination of a probability measure in $\mathbb{P}$ that validates the conditions of the equivalence theorems using a LP formulation (Duarte and Wong, 2014). Practically, the construction of the dispersion function for minimax optimal designs is cumbersome because the need of finding a probability measure on the parameters domain before checking the General Equivalence Theorem on the design. The procedure becomes easier when the answering set is a singleton which is what we have, see Fig. 1. The dispersion function is bounded above by 0 and peaks at the support points of the generated design. It follows that the dispersion function satisfies the conditions required in the equivalence theorem and so confirms the minimax D-optimality of the generated design.

Table 3
Minimax $A-$ and $E-$optimal designs for the logistic model on $\mathbf{X}$ discretized using $\Delta x = 0.02$, for different plausible regions when β is restricted to [1.0, 3.0].

	$A-$optimal design			$E-$optimal design		
	$[\mu_L, \mu_U]$			$[\mu_L, \mu_U]$		
	[0.0, 1.0][b]	[0.0, 3.5][b]	[6.0, 8.0][a]	[0.0, 1.0][b]	[0.0, 3.5][b]	[6.0, 8.0][a]
	(−0.54,0.0510)	(−0.60,0.1240)	(5.54,0.0691)	(−0.56,0.0945)	(−0.58,0.1823)	(5.56,0.0528)
	(−0.52,0.3249)	(−0.58,0.0733)	(5.56,0.1824)	(−0.54,0.2823)	(0.62,0.0809)	(5.58,0.2127)
	(0.52,0.1970)	(0.70,0.1429)	(6.56,0.0467)	(0.50,0.2470)	(0.64,0.0557)	(6.56,0.1650)
	(0.54,0.0608)	(1.60,0.2165)	(6.58,0.1855)	(1.54,0.2798)	(1.46,0.0382)	(6.88,0.0730)
	(1.54,0.1828)	(1.62,0.0071)	(7.32,0.0078)	(1.56,0.0965)	(1.48,0.1620)	(6.90,0.0377)
	(1.56,0.1835)	(2.28,0.0835)	(7.34,0.2448)		(2.16,0.1754)	(7.46,0.2077)
		(2.30,0.0475)	(8.40,0.1123)		(2.18,0.0015)	(8.44,0.0481)
		(2.96,0.1054)	(8.42,0.1514)		(2.92,0.0157)	(8.46,0.2029)
		(4.10,0.1999)			(2.94,0.1064)	
					(4.08,0.1406)	
					(4.10,0.0413)	
$\boldsymbol{p}^{\mathrm{OPT}}$	$[3.0, 1.0]^{\mathrm{T}}$	$[3.0, 0.0]^{\mathrm{T}}$	$[3.0, 8.0]^{\mathrm{T}}$	$[3.0, 1.0]^{\mathrm{T}}$	$[3.0, 3.5]^{\mathrm{T}}$	$[3.0, 6.0]^{\mathrm{T}}$
CPU (s)	8.23	11.31	5.20	7.06	18.25	7.32

$(x.xx, w.wwww)$ ≡(design point, weight).
[a] The design space is $\mathbf{X} \equiv [3, 9]$.
[b] The design space is $\mathbf{X} \equiv [-1, 5]$ for plausible regions.

Similar dispersion functions can be obtained for all designs in Table 1 and for designs found by our algorithm under other criteria. For example, Figs. 2(b) and 2(c) show, respectively, the dispersion functions of the $A-$ and $E-$optimal designs in Table 3 found by our algorithm for the logistic model when $\mathbf{P} = [1.0, 3.0] \times [0.0, 1.0]$, $\Delta x = 0.02$ and $\mathbf{X} \equiv [-1, 5]$.

In contrast to minimax $D-$optimal designs, minimax $A-$ and $E-$optimal designs have not been reported in the literature for the logistic model. Our proposed algorithm can be directly used to find such minimax optimal designs using the same discretization scheme. Table 3 displays selected minimax $A-$ and $E-$optimal designs obtained when $[\beta_L, \beta_U] = [1.0, 3.0]$ and there are different plausible regions for μ. We observe from Tables 1 and 3 that the minimax $A-$ and $E-$optimal designs have more support points than $D-$optimal designs. Further, the computational times required by all the designs are all quite similar, and the differences are mainly due to the number of iterations required to reach convergence, as defined by the user-specified tolerance level ϵ, see Eq. (18). Overall, the proposed algorithm shows good flexibility and determines the various optimal designs without requiring a prohibitive amount of computational resources.

The number of initial sampling points tends to have low impact on the convergence rate. In each iteration we determine the hypograph of the design criterion by considering all elements in $\mathbb{P}$ and generate the optimal cutting plan corresponding to the lower bound. It follows that when $\mathbb{P}^{(0)}$, by chance, contains $\boldsymbol{p}^{\mathrm{OPT}}$ the convergence is potentially faster since the lower bound limit is attained at the end of the first iteration but is independent on the number of sampling points. In contrast, the location of the sampling points might have significant impact on the convergence rate. We exploit this feature by judiciously choosing the points in $\mathbb{P}^{(0)}$, which include the vertices of the parameter domain. When $\boldsymbol{p}^{\mathrm{OPT}}$ coincides with one of the vertices, which is a feature we observed empirically for many problems, the final lower bound is obtained at the first iteration and the subsequent iterations serve only to let the upper bound converge. General problems where $\boldsymbol{p}^{\mathrm{OPT}}$ does not coincide with one of the vertices require a few iterations since successive cutting plans generating lower and upper bounds are constructed until the condition (18) is attained. Consequently, the number of points in the initial sample does not explain the differences observed in CPU time which are mainly due to the algorithm.

4.2. Example 2 - Four-parameter homoscedastic Hill model

Consider the 4-parameter homoscedastic Hill model, commonly used for curve-fitting analysis in bioassays or immunoassays such as ELISAs or dose–response curves (Hill, 1910). The model is widely used in various types of dose–response studies and in other disciplines because it is flexible and can model mean response with different shapes. Let y be the outcome at level x of the independent variable defined in the design space $\mathbf{X}$, which we assumed is a close and bounded interval for the dose or log dose. The outcome depends on x, and its mean response at x is

$$\mathbb{E}[y|x, \boldsymbol{p}] = E_0 + \frac{(E_\infty - E_0)\, x^m}{x^m + k_d^m}, \quad x \in \mathbf{X}, \quad \boldsymbol{p} \in \mathbf{P}. \tag{24}$$

Here, $\boldsymbol{p} = [E_0, E_\infty, k_d^m, m]^{\mathrm{T}}$ is the vector of parameters in the Hill model and its plausible values are known to lie in a compact set $\mathbf{P}$. The interpretations of the Hill parameters are: E_0 is the control effect at zero dose concentration, E_∞ is the effect of a very large dose of the drug, k_d is the dose that induces 50% of the maximal effect, and m is the power that controls the slope. Khinkis et al. (2003) used (24) for studying the effect of an inhibitory drug on tumor growth and found locally $D-$optimal designs using seven sets of nominal values for $\boldsymbol{p}$ and assuming no uncertainty in parameters. These designs were found to be sensitive to mis-specification of the values of $\boldsymbol{p}$, meaning that a design can lose its efficiency greatly if the

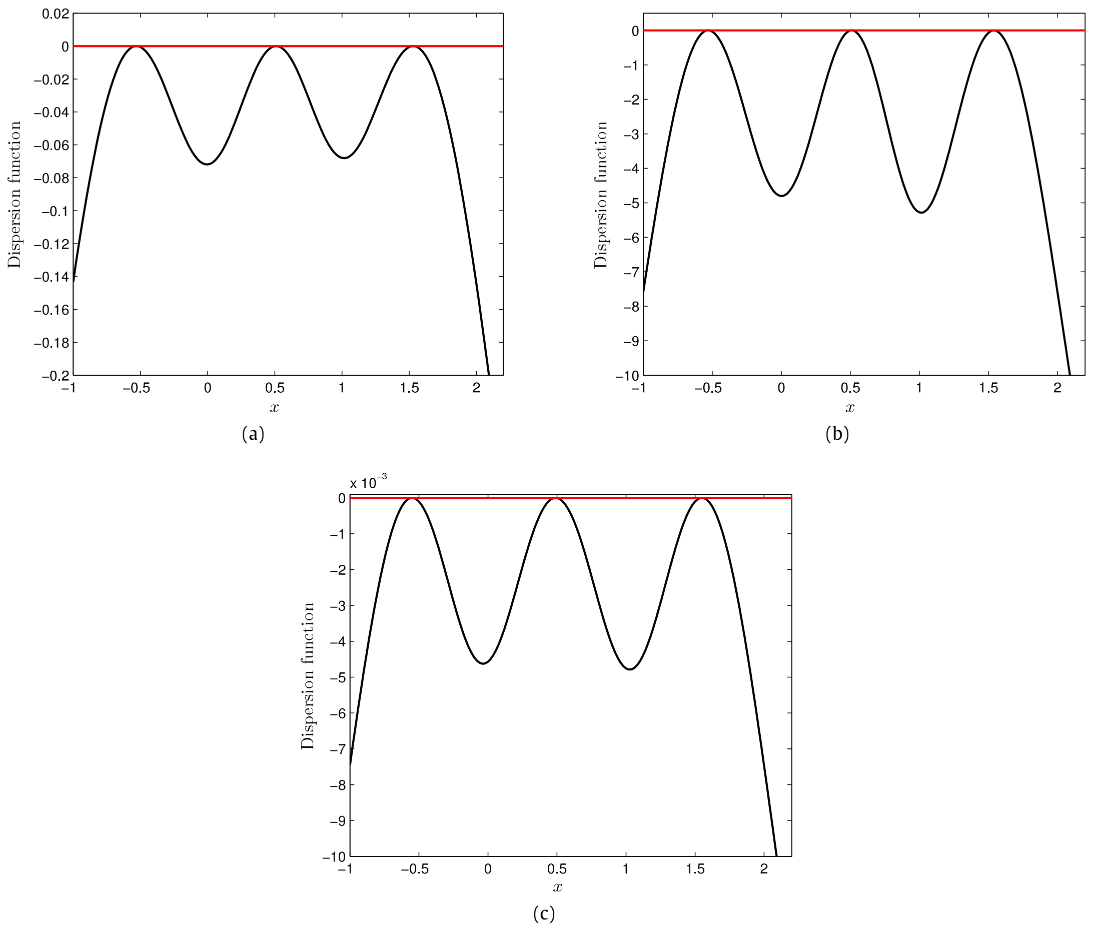

Fig. 2. Dispersion functions of the generated designs for the logistic model under (a) $D-$optimality criterion, (b) $A-$optimality criterion, (c) $E-$optimality criterion, when $\mathbf{X} \equiv [-1, 5]$ is discretized by $\Delta x = 0.02$ and the plausible region is $\mathbf{P} = [1.0, 3.0] \times [0.0, 1.0]$.

nominal values are wrong. One design strategy that may possibly overcome this problem is to adopt a minimax strategy for designing the study. Our goal is to find optimal designs for parameter uncertainty scenarios employing a minimax framework formalized in Section 3.

Assume that $\mathbf{P} = [E_0^L, E_0^U] \times [E_\infty^L, E_\infty^U] \times [(k_d^m)^L, (k_d^m)^U] \times [m^L, m^U]$ is the known set that contains all plausible values of all the parameters in the model. The superscripts L and U designate the lower and upper bounds for values of the model parameters. The FIM at the point x_i is $M(x_i, \boldsymbol{p}) = h(x_i, \boldsymbol{p})\, h(x_i, \boldsymbol{p})^{\mathrm{T}}$, where

$$h(x_i, \boldsymbol{p}) = \begin{pmatrix} \dfrac{\partial \mathbb{E}[y|x_i, \boldsymbol{p}]}{\partial E_\infty} \\ \dfrac{\partial \mathbb{E}[y|x_i, \boldsymbol{p}]}{\partial E_0} \\ \dfrac{\partial \mathbb{E}[y|x_i, \boldsymbol{p}]}{\partial (k_d^m)} \\ \dfrac{\partial \mathbb{E}[y|x_i, \boldsymbol{p}]}{\partial m} \end{pmatrix} = \begin{pmatrix} \dfrac{x_i^m}{k_d^m + x_i^m} \\ \dfrac{k_d^m}{k_d^m + x_i^m} \\ -\dfrac{(E_\infty - E_0)\, x_i^m}{(k_d^m + x_i^m)^2} \\ \dfrac{(E_\infty - E_0)\, k_d^m\, x_i^m\, \log(x_i)}{(k_d^m + x_i^m)^2} \end{pmatrix}.$$

Tables 4–5 presents minimax $D-$, $A-$ and $E-$ optimal designs for a plausibility region $\mathbf{P} = [1.0, 2.0] \times [0.1, 0.5] \times [0.5, 1.0] \times [m^L, m^U]$, and the design space is $\mathbf{X} \equiv [10^{-5}, 10]$. We consider two intervals for possible values of m to assess the impact of the slope (positive and negative values) on the algorithm and the optimal designs. In all cases, the design interval is discretized using intervals of length $\Delta x = 0.05$, yielding 201 candidate design points. The initial population in $\mathbb{P}^{(0)}$ is formed by the 2^4 vertices of $\mathbf{P}$ plus 32 points random uniformly sampled from the edges, one per edge, and 2 additional points sampled from inside the domain, in a total of 50 vectors $\boldsymbol{p}$.

Table 4
Minimax $D-$, $A-$ and $E-$ optimal designs found by our algorithm for the 4-parameter homoscedastic Hill model on $\mathbf{X} \equiv [10^{-5}, 10]$ using a grid of equally spaced points $\Delta x = 0.05$ unit apart and $\mathbf{P} \equiv [1.0, 2.0] \times [0.1, 0.5] \times [0.5, 1.0] \times [0.5, 1]$.

	$D-$optimal design	$A-$optimal design	$E-$optimal design
	$(10^{-5}$,0.2453)	$(10^{-5}$,0.0697)	$(10^{-5}$,0.0368)
	(0.05,0.2218)	(0.05,0.2126)	(0.05,0.2058)
	(0.30,0.0547)	(0.45,0.0013)	(0.50,0.0227)
	(1.35,0.2292)	(0.50,0.0066)	(1.40,0.0313)
	(10.00,0.2490)	(1.35,0.0022)	(1.45,0.4178)
		(10.00,0.2711)	(10.00,0.2856)
$\boldsymbol{p}^{\text{OPT}}$	$[1.0, 0.5, 1.0, 0.5]^{\text{T}}$	$[1.0, 0.5, 1.0, 1.0]^{\text{T}}$	$[1.0, 0.5, 1.0, 1.0]^{\text{T}}$
CPU (s)	44.67	56.38	38.19

$(xx.xx,\ w.wwww) \equiv$ (design point, weight).

Table 5
Minimax $D-$, $A-$ and $E-$ optimal designs found by our algorithm for the 4-parameter homoscedastic Hill model on $\mathbf{X} \equiv [10^{-5}, 10]$ using a grid of equally spaced points $\Delta x = 0.05$ unit apart and $\mathbf{P} \equiv [1.0, 2.0] \times [0.1, 0.5] \times [0.5, 1.0] \times [-2, -0.5]$.

	$D-$optimal design	$A-$optimal design	$E-$optimal design
	$(10^{-5}$,0.2422)	$(10^{-5}$,0.0969)	$(10^{-5}$,0.0816)
	(0.05,0.2190)	(0.05,0.2328)	(0.05,0.2289)
	(0.10,0.0121)	(0.85,0.0901)	(0.80,0.0076)
	(0.65,0.0662)	(1.75,0.0010)	(0.85,0.0818)
	(1.80,0.0239)	(1.80,0.3048)	(1.80,0.0011)
	(1.85,0.1876)	(1.85,0.0234)	(1.85,0.3425)
	(10.00,0.2490)	(1.90,0.0015)	(10.00,0.2565)
	(10.00,0.2490)	(10.00,0.2495)	
$\boldsymbol{p}^{\text{OPT}}$	$[1.0, 0.5, 1.0, -2.0]^{\text{T}}$	$[1.0, 0.5, 1.0, -1.1092]^{\text{T}}$	$[1.0, 0.5, 0.5, -1.0763]^{\text{T}}$
CPU (s)	41.94	47.95	38.92

$(xx.xx,\ w.wwww) \equiv$ (design point, weight).

All the designs have support points at the extremes of $\mathbf{X}$. When the model has a positive slope, i.e. $m > 0$, the minimax optimal designs include three additional points, one at $x \in [0.3, 0.5]$, another at $x \in [1.35, 1.45]$ and one at $x = 0.05$. When the model has a negative slope, the optimal designs have a support point at $x \in [0.05, 0.1]$, another at $x \in [0.65, 0.85]$ and the last one at $x \in [1.80, 1.85]$. Our designs have 5 points, one point more than the locally $D-$optimal designs with four support points found by Khinkis et al. (2003) and Qiu (2014). This is not surprising because minimax optimal designs often require more support points than locally optimal designs based on a single best guess for the nominal values of the parameters. We notice that $\boldsymbol{p}^{\text{OPT}}$ in four cases is located on one of the vertex of the plausible region, and in the remaining two cases is on one of the edges. In those particular cases, the algorithm takes several (more than 2) iterations to converge to $\boldsymbol{p}^{\text{OPT}}$.

The optimality of the generated minimax designs for the four-parameter homoscedastic Hill model in Tables 4–5 was also confirmed using the equivalence theorem.

4.3. Example 3 - nth order successive kinetic rate model

To further test the algorithm in Section 3, we next use a model defined by ordinary differential equations (ODEs) and discussed in Atkinson et al. (2007, page 270). The model describes the dynamics of two consecutive reactions $A \xrightarrow{r_1} B \xrightarrow{r_2} C$ occurring in a constant volume continuous stirred tank reactor (CSTR), where the concentration C_B of product B is measured. Let C_A be the concentration of reactant A, C_C the concentration of the product C, and r_1 and r_2 the kinetic rates of $A \longrightarrow B$ and $B \longrightarrow C$, respectively. Both kinetic rates are temperature independent and follow the Arrhenius law, i.e. $r_i = \pi_i\, C_{r,i}^{\alpha_i},\ i \in \{1, 2\}$, where π_i and α_i are, respectively, the pre-exponential factor and the order of ith kinetic law; $C_{r,i}$ is the concentration of the reactant. The model equations are

$$\mathbb{E}(C_B|t, \boldsymbol{p}) = C_B(t), \quad t \in \mathbf{T}, \quad \boldsymbol{p} \in \mathbf{P} \tag{25a}$$

$$\frac{\mathrm{d}C_A}{\mathrm{d}t} = -\pi_1\, C_A^{\alpha_1} \tag{25b}$$

$$\frac{\mathrm{d}C_B}{\mathrm{d}t} = \pi_1\, C_A^{\alpha_1} - \pi_2\, C_B^{\alpha_2} \tag{25c}$$

$$\frac{\mathrm{d}C_C}{\mathrm{d}t} = \pi_2\, C_B^{\alpha_2} \tag{25d}$$

$$C_A(0) = 1.0 \quad C_B(0) = 0.0 \quad C_C(0) = 0.0. \tag{25e}$$

Here, $\mathbf{T}$ is a user-selected, bounded time interval, $\boldsymbol{p} = [\pi_1,\ \pi_2,\ \alpha_1,\ \alpha_2]^{\text{T}}$ is the vector of model parameters belonging to a known compact set $\mathbf{P}$. Eqs. (25b)–(25d) describe the dynamics of the concentrations of the species in the CSTR and (25e) is

Table 6
Minimax optimal designs for the nth order consecutive reaction model with $\Delta t = 0.2$, $\mathbf{T} \equiv [0, 20]$ and $\mathbf{P} \equiv [0.5, 1.0] \times [0.1, 0.5] \times [1.0, 2.0] \times [1.0, 2.0]$.

	$D-$optimal design	$A-$optimal design	$E-$optimal design
	(0.4,0.2493)	(0.2,0.3881)	(0.2,0.3889)
	(1.6,0.1517)	(1.6,0.2266)	(1.6,0.2278)
	(1.8,0.0940)	(4.6,0.1595)	(4.6,0.1565)
	(4.4,0.1436)	(10.8,0.1392)	(10.8,0.1341)
	(4.6,0.1057)	(20.0,0.0862)	(20.0,0.0941)
	(10.8,0.0330)		
	(11.0,0.2158)		
$\boldsymbol{p}^{\text{OPT}}$	$[1.0, 0.1, 2.0, 2.0]^{\text{T}}$	$[1.0, 0.5, 2.0, 2.0]^{\text{T}}$	$[1.0, 0.5, 2.0, 2.0]^{\text{T}}$
CPU (s)	121.31	77.94	109.38

$(tt.t, w.wwww) \equiv$ (design point, weight).

the set of initial conditions. The design interval is $\mathbf{T} = [0, 20]$, and we discretize it to become $\mathbb{T}$ with a uniform grid Δt units apart. Each time instant $t_i = t_{i-1} + \Delta t$ is a candidate point from $\mathbb{T}$ for inclusion in the support of the optimal design.

Our goal is to prescribe an optimal design of experiments to estimate $\boldsymbol{p}$ by choosing appropriate time points, t_i, to sample from the CSTR and measure C_B in the CSTR. The plausibility region of the parameters is known and equal to $\mathbf{P} \equiv [0.5, 1.0] \times [0.1, 0.5] \times [1.0, 2.0] \times [1.0, 2.0]$. The sampling mechanism to generate the first m_0 parameter combinations populating $\mathbb{P}^{(0)}$ is similar to previous examples, that is, we select the $2^4 = 16$ corners of $\mathbf{P}$, plus 32 vectors drawn from a uniform distribution over the edges of $\mathbf{P}$ and 2 additional vectors drawn from $\mathbf{P}$.

To construct the FIM for model (25b)–(25e), we adopt the process systems terminology and designate the variables representing the process dynamics as *states*. The states characterize the evolution of the reactional mass in time, and the vector of states is $\mathcal{Z}(t) = [C_A(t),\ C_B(t),\ C_C(t)]^{\text{T}}$. The variables used for monitoring the process along the time is a subset or a linear combination of the states called *measurements*, and generically represented by $\mathcal{Y}(t)$. Here, a single *measurement* is employed for monitoring the process where $\mathcal{Y}(t) = [C_B(t)]^{\text{T}}$. The vector of functions $\boldsymbol{f}(\mathcal{Z}(t), \boldsymbol{p}) = [-\pi_1\, C_A^{\alpha_1}(t),\ \pi_1\, C_A^{\alpha_1}(t) - \pi_2\, C_B^{\alpha_2}(t),\ \pi_2\, C_B^{\alpha_2}(t)]^{\text{T}}$ contains the right hand side of the Differential Equations (25b)–(25d) and the vector $\boldsymbol{g}(\mathcal{Z}(t), \boldsymbol{p}) = [C_B(t)]^{\text{T}}$ the right side of the measurements Eq. (25a). Finally, the sensitivity of the ith state denoted by $z_i \in \mathcal{Z}(t)$ with respect to parameter p_j at reference point $\boldsymbol{p}^{\text{ref}}$ is designated by $\sigma_{i,j}$, yielding:

$$\frac{\mathrm{d}\sigma_{i,j}}{\mathrm{d}t} = \sum_{k=1}^{3} \frac{\partial f_i(\mathcal{Z}(t), \boldsymbol{p}^{\text{ref}})}{\partial z_k}\, \sigma_{k,j} + \frac{\partial f_i(\mathcal{Z}(t), \boldsymbol{p}^{\text{ref}})}{\partial p_j}, \quad i \in \{1, 2, 3\},\ j \in \{1, \dots, n_p\}, \tag{26a}$$

$$\eta_{p_j}^{C_B}(t, \boldsymbol{p}^{\text{ref}}) = \frac{\partial g(\mathcal{Z}(t), \boldsymbol{p}^{\text{ref}})}{\partial z_2}\, \sigma_{2,j}(t, \boldsymbol{p}^{\text{ref}}), \qquad j \in \{1, \dots, n_p\}. \tag{26b}$$

$$\sigma_{i,j}(0) = 0, \qquad i \in \{1, 2, 3\},\ j \in \{1, \dots, n_p\} \tag{26c}$$

where $\eta_{p_j}^{C_B}(t, \boldsymbol{p}^{\text{ref}})$ is the sensitivity of the measure C_B with respect to parameter p_j at time instant t for $\boldsymbol{p}^{\text{ref}}$. The vector of sensitivities used to compute the FIM at each candidate time instant $t_i \in \mathbb{T}$ is

$$\boldsymbol{\eta}^{C_B}(t_i, \boldsymbol{p}^{\text{ref}}) = [\eta_{\pi_1}^{C_B}(t_i, \boldsymbol{p}^{\text{ref}}),\ \eta_{\pi_2}^{C_B}(t_i, \boldsymbol{p}^{\text{ref}}),\ \eta_{\alpha_1}^{C_B}(t_i, \boldsymbol{p}^{\text{ref}}),\ \eta_{\alpha_2}^{C_B}(t_i, \boldsymbol{p}^{\text{ref}})]^{\text{T}},$$

and

$$M(t_i, \boldsymbol{p}) = \boldsymbol{\eta}^{C_B}(t_i, \boldsymbol{p}^{\text{ref}}) \left[\boldsymbol{\eta}^{C_B}(t_i, \boldsymbol{p}^{\text{ref}})\right]^{\text{T}}.$$

The sensitivity of $C_B(t)$ with respect to the parameters at $\boldsymbol{p}_i \in \mathbb{P}$ is determined by solving the Eq. (26) simultaneously with the Model (25b)–(25d) employing an ODEs solver. Here, the ODEs system is solved for each $\boldsymbol{p}_i$; in each iteration $\boldsymbol{p}^{\text{ref}}$ assumes a value $\boldsymbol{p}_i$ of $\mathbb{P}$. A variable order, variable step stiff implicit integrator was used to solve the ODEs system (25b)–(26c) for each vector $\boldsymbol{p} \in \mathbb{P}$. The absolute and relative tolerances of the integrator were set to 10^{-5}.

Table 6 presents the optimal designs for $\Delta t = 0.2$, and we observe that the D-optimal design is in good agreement with the locally $D-$optimal designs in Atkinson et al. (2007, page 270). The locally $D-$optimal designs were determined for a singleton $\boldsymbol{p}$ employing an exchange algorithm which does not require the discretization of the time domain, and are consistently based on 4 support points. Our design has 7 support points, which is consistent with the trend observed by several authors, that minimax and Bayesian optimal designs tend to have more support points than locally optimal designs. The sampling instances are similar for all criteria except that the $D-$optimality criterion produces a design that does not include $t = 20$, which also has low weight in the $A-$ and $E-$optimal designs. We note that a considerable fraction of the CPU time (about 80%) is devoted to computing the FIM that requires solving the ODE's for every vector $\boldsymbol{p} \in \mathbb{P}$. We also observe that for all the designs in Table 6, $\boldsymbol{p}^{\text{OPT}}$ is on a vertex, which explains why the algorithm required only 2 iterations to reach the $\epsilon-$optimality where the second is confirmatory.

4.4. *Alternative algorithms*

In previous sections we compared the proposed algorithm with another deterministic based procedure based on SIP. Here, we briefly discuss other mathematical programming based tools and other kinds of algorithms, such as metaheuristic optimization techniques that were recently used to find minimax optimal designs of experiments.

Our work suggests that the SDP based algorithm is generally more flexible, faster and easier to use than SIP for obtaining minimax optimal designs. Exchange and multiplicative algorithms, along with many of their variants, are also systematic ways of finding various optimal designs but they are not applicable when the optimality criteria are non-differentiable, which is the case when we have a minimax or maximin type of criterion. Another algorithmic approach to find maximin and maximin standardized optimal designs is to use functional approach and methods developed by Melas (2006, Sec. 8.7.1). We feel that the method offers potential but is not well tested for different applications.

Recently, nature-inspired metaheuristic algorithms are increasingly used to solve large and difficult optimization problems (Yang, 2010; Whitacre, 2011a, b) in computer science and engineering. These are general optimization tools and have gained much attention in recent years because of their flexibility, ease of implementation and their many reported successes in solving or nearly solving different types of high dimensional and complex optimization problems in practice. Metaheuristic algorithms are generally assumptions free, relatively powerful and can solve an optimization problem regardless of the model, optimality criteria and the types of constraints imposed on the problem. However, global optimality of the solutions cannot be guaranteed. Qiu (2014) and Chen et al. (2015) applied a well known member of this class of algorithms, called Particle Swarm Optimization (PSO) to generate several types of optimal designs. Masoudi et al. (2017) adopted another metaheuristic procedure called Imperialist Competitive Algorithm (ICA) for the same purpose.

We did not compare performance of such algorithms with our proposed approach because we feel that comparing performance of algorithms should always be done meaningfully and fairly and we do not feel it is feasible to do so here. At the onset, we focus on a systematic approach for finding optimal designs and an approach that uses metaheuristic algorithms is not. The latter algorithms have very different motivations and work very differently from our proposed procedure. For example, metaheuristic algorithms are stochastic in nature, meaning that repeated runs may produce different or slightly different results, there is generally no firm rationale behind the construction of these algorithms and they also frequently depend on a host of tuning parameters that can affect the performance of the algorithm dramatically. Exacerbating the comparison issues is that each metaheuristic algorithm has different number of tuning parameters with different interpretations and there are no firm guidelines for choosing these parameters. Consequently, if say PSO fails to find the optimum, it could well be that the tuning parameters were not properly chosen and if PSO did find the optimum, there is no guarantee that a rerun of the algorithm under the same settings will produce the same result. In addition, the maximum iteration number and the flock size in PSO are somewhat arbitrary but their choices can affect the convergence of the algorithm. Ultimately, it is a matter of user's preference which type of algorithms to use and the training of the user.

5. Summary

We propose a systematic approach based on mathematical programming to find minimax optimal designs of experiments for nonlinear models. Our algorithm uses NLP to solve the inner level problem after using SDP to solve the outer problem. Our approach requires the design space to be discretized before employing a delayed constraint generation method to solve the minimax program iteratively until convergence is attained. The SDP problem is formulated as a restricted problem where a finitely constrained set of parameters replaces the plausible region of the model parameters. We use information from the parameters domain to generate the initial sample of instances that replace the compact domain $\mathbf{P}$.

We apply the proposed algorithm to find minimax $D-$optimal designs for the logistic model, the 4-parameter Hill model and the model for the dynamics of two temperature independent nth order consecutive reactions in a CSTR. The results obtained are similar to those obtained with an algorithm that assumes a continuous design space. Our designs typically contain more support points than that obtained from a continuous domain, where the extra points may be freely collapsed into a few obvious points. With a discrete design space, a true optimal design point may be clustered among a few grid points and has its weights spread among the nearby points. Our experience is that there are small differences in the efficiencies of the resulting designs if any one of the clustered points is selected as the support point.

We also applied our algorithm to generate minimax $A-$ and $E-$optimal designs for the previous models. Such optimal designs have not been reported before in the literature. This suggests that our algorithm is flexible and can generate optimal designs for semidefinite representable criteria and requires mild computation times. We believe that this is the first algorithm for finding minimax optimal designs that uses SDP formulations within a cutting plane approach. The global optima of the consecutive outer problems are solved iteratively to convergence. In the majority of practical cases that we have tested, our algorithm requires a single iteration to converge, and occasionally 2 or 3 iterations are required. This means that the NLP solver is used only to "confirm" the optimality of the design and frequently after the SDP solver finds a highly efficient design.

Minimax or maximin types of optimal design problems are generally hard to solve because the design criterion is not differentiable and involve two or more layers of optimization. Many current algorithms assume the criterion is differentiable and so they are not applicable. Our focus was on developing an effective algorithm for finding minimax optimal designs under a broad setup. Our work includes several examples of varying complexities that demonstrate our approach works, including cases when the model is more complex, such as when there are additional linear constraints on the weights and we wish to optimize among designs ξ in $\Xi_{A,\mathbf{b}}$. Such situations occur when each weight may have its lower or upper bounds, cf. e.g. Uciński (2015) or when the sought optimal design is marginally constrained (Martín-Martín et al., 2007). It can be shown such a problem falls under our framework and so our approach applies. The equivalence theorem for confirming the optimality of a design in $\Xi_{A,\mathbf{b}}$ is more complicated than Eq. (7) or (8) because the Lagrange multipliers for the constraints $A\mathbf{w} \leq \mathbf{b}$ are involved, see Cook and Fedorov (1995). For space consideration, we omit further discussion and examples.

Acknowledgments

The research of Professor Wong reported in this paper was partially supported by the National Institute of General Medical Sciences of the National Institutes of Health under the Grant Award Number R01GM107639. The contents in this paper are solely the responsibility of the authors and do not necessarily represent the official views of the National Institutes of Health.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.csda.2017.09.008.

References

Andersen, E., Jensen, B., Jensen, J., Sandvik, R., Worsøe, U., 2009. MOSEK version 6, Technical Report TR–2009–3, MOSEK.
Atkinson, A.C., Donev, A.N., Tobias, R.D., 2007. Optimum Experimental Designs, with SAS. Oxford University Press, Oxford.
Ben-Tal, A., Nemirovski, A.S., 2001. Lectures on Modern Convex Optimization: Analysis, Algorithms, and Engineering Applications. Society for Industrial and Applied Mathematics, Philadelphia.
Berger, M.P.F., Wong, W.K., 2009. An Introduction to Optimal Designs for Social and Biomedical Research. John Wiley & Sons, Chichester.
Bischof, C.H., Bücker, H.M., Lang, B., Rasch, A., Vehreschild, A., 2002. Combining source transformation and operator overloading techniques to compute derivatives for matlab programs. In: Proceedings of the Second IEEE International Workshop on Source Code Analysis and Manipulation. (SCAM 2002), IEEE Computer Society, Los Alamitos, CA, USA, pp. 65–72.
Blankenship, J.W., Falk, J.E., 1976. Infinitely constrained optimization problems. J. Optim. Theory Appl. 19 (2), 261–281.
Boer, E.P.J., Hendrix, E.M.T., 2000. Global optimization problems in optimal design of experiments in regression models. J. Global Optim. 18, 385–398.
Boyd, S., Vandenberghe, L., 2004. Convex Optimization. University Press, Cambridge.
Burclová, K., Pázman, A., 2016. Optimal design of experiments via linear programming. Statist. Papers 57 (4), 893–910.
Byrd, R.H., Hribar, M.E., Nocedal, J., 1999. An interior point algorithm for large-scale nonlinear programming. SIAM J. Optim. 9 (4), 877–900.
Chaloner, K., Larntz, K., 1989. Optimal Bayesian design applied to logistic regression experiments. J. Statist. Plann. Inference 59, 191–208.
Chen, R.-B., Chang, S.-P., Wang, W., Tung, H.-C., Wong, W.K., 2015. Minimax optimal designs via particle Swarm optimization methods. Stat. Comput. 25 (5), 975–988.
Coleman, T.F., Li, Y., 1994. On the convergence of reflective Newton methods for large-scale nonlinear minimization subject to bounds. Math. Program. 67 (2), 189–224.
Cook, D., Fedorov, V., 1995. Invited discussion paper constrained optimization of experimental design. Statistics 26 (2), 129–148.
Cook, R.D., Nachtsheim, C.J., 1982. Model robust, linear-optimal designs. Technometrics 24, 49–54.
Dette, H., Haines, L.M., Imhof, L.A., 2007. Maximin and Bayesian optimal designs for regression models. Statist. Sinica 17, 463–480.
Drud, A., 1985. CONOPT: A GRG code for large sparse dynamic nonlinear optimization problems. Math. Program. 31, 153–191.
Drud, A., 1994. CONOPT - A large–scale GRG code. ORSA J. Comput. 6 (2), 207–216.
Duarte, B.P.M., Wong, W.K., 2014. A semi-infinite programming based algorithm for finding minimax optimal designs for nonlinear models. Stat. Comput. 24 (6), 1063–1080.
Duarte, B.P.M., Wong, W.K., 2015. Finding Bayesian optimal designs for nonlinear models: a semidefinite programming-based approach. Internat. Statist. Rev. 83 (2), 239–262.
Duarte, B.P.M., Wong, W.K., Atkinson, A.C., 2015. A semi-infinite programming based algorithm for determining $T-$optimum designs for model discrimination. J. Multivariate Anal. 135, 11–24.
Duarte, B.P.M., Wong, W.K., Dette, H., 2017. Adaptive grid semidefinite programming for finding optimal designs. Stat. Comput. http://dx.doi.org/10.1007/s11222-017-9741-y.
Duarte, B.P.M., Wong, W.K., Oliveira, N. M.C., 2016. Model-based optimal design of experiments - Semidefinite and nonlinear programming formulations. Chemometr. Intell. Lab. Syst. 151, 153–163.
Fackle-Fornius, E., Miller, F., Nyquist, H., 2015. Implementation of maximin efficient designs in dose-finding studies. Pharm. Stat. 14 (1), 63–73.
Fedorov, V.V., 1980. Convex design theory. Math. Oper.forsch. Stat. Ser. Stat. 11, 403–413.
Fedorov, V.V., Leonov, S.L., 2014. Optimal Design for Nonlinear Response Models. Chapman and Hall/CRC Press, Boca Raton.
Filová, L., Trnovská, M., Harman, R., 2011. Computing maximin efficient experimental designs using the methods of semidefinite programming. Metrika 64 (1), 109–119.
Gaivoronski, A., 1986. Linearization methods for optimization of functionals which depend on probability measures. In: Prékopa, A., Wets, R. J.-B. (Eds.), Stochastic Programming 84 Part II. In: Mathematical Programming Studies, vol. 28, Springer Berlin Heidelberg, pp. 157–181.
Gill, P.E., Murray, W., Saunders, M.A., 2005. SNOPT: An SQP algorithm for large-scale constrained optimization. SIAM Rev. 47 (1), 99–131.
Goos, P., Jones, B., 2011. Optimal Design of Experiments: A Case Study Approach. Wiley, New York.
Grant, M., Boyd, S., Ye, Y., 2012. cvx Users Guide for cvx version 1.22. CVX Research, Inc., 1104 Claire Ave., Austin, TX 78703-2502.
Harman, R., Jurík, T., 2008. Computing $c-$optimal experimental designs using the Simplex method of linear programming. Comput. Statist. Data Anal. 53 (2), 247–254.
Heredia-Langner, A., Montgomery, D.C., Carlyle, W.M., Borror, C.M., 2004. Model-robust optimal designs: A genetic algorithm approach. J. Qual. Technol. 36, 263–279.
Hettich, R., Kaplan, A., Tichatschke, R., 2001. Semi-infinite programming: Numerical methods. In: Encyclopedia of Optimization, Vol. 5. Kluwer, Amsterdam, pp. 112–117.
Hettich, R., Kortanek, K.O., 1993. Semi-infinite programming: Theory, methods and applications. SIAM Rev. 35, 380–429.
Hill, A., 1910. The possible effects of the aggregation of the molecules of haemoglobin on its dissociation curves. J. Physiol. 40 (Suppl.), iv–vii.

Khinkis, L.A., Levasseur, L., Faessel, H., Greco, W.R., 2003. Optimal design for estimating parameters of the 4-parameter Hill model. Nonlinearity Biol. Toxicol. Med. 1, 363–377.
Kiefer, J., 1974. General equivalence theory for optimum design (approximate theory). Ann. Statist. 2, 849–879.
López, M., Still, G., 2007. Semi-infinite programming. European J. Oper. Res. 180, 491–518.
Ludena, C.E., Hertel, T.W., Preckel, P.V., Foster, K., Nin, A., 2007. Productivity growth and convergence in crop, ruminant, and nonruminant production: measurement and forecasts. Agricult. Econ. 37 (1), 1–17.
Mandal, A., Wong, W.K., Yu, Y., 2015. Algorithmic searches for optimal designs. In: Handbook of Design and Analysis of Experiments. CRC Press, Boca Ratton, pp. 755–786.
Martín-Martín, R., Torsney, B., López-Fidalgo, J., 2007. Construction of marginally and conditionally restricted designs using multiplicative algorithms. Comput. Statist. Data Anal. 51 (12), 5547–5561.
Masoudi, E., Holling, H., Wong, W.K., 2017. Application of imperialist competitive algorithm to find minimax and standardized maximin optimal designs. Comput. Statist. Data Anal. 113, 330–345.
Melas, V., 2006. Functional approach to optimal experimental design. In: Lecture Notes in Statistics, Springer.
Mitsos, A., Tsoukalas, A., 2015. Global optimization of generalized semi-infinite programs via restriction of the right hand side. J. Global Optim. 61 (1), 1–17.
Molchanov, I., Zuyev, S., 2002. Steepest descent algorithm in a space of measures. Stat. Comput. 12, 115–123.
Mutapcic, A., Boyd, S.P., 2009. Cutting-set methods for robust convex optimization with pessimizing oracles. Optim. Methods Softw. 24 (3), 381–406.
Noubiap, R.F., Seidel, W., 2000. A minimax algorithm for constructing optimal symmetrical balanced designs for a logistic regression model. J. Statist. Plann. Inference 91 (1), 151–168.
Papp, D., 2012. Optimal designs for rational function regression. J. Amer. Statist. Assoc. 107, 400–411.
Pázman, A., Pronzato, L., 2014. Optimum design accounting for the global nonlinear behavior of the model.. Ann. Statist. 42 (4), 1426–1451.
Prentice, R., 1976. A generalization of the probit and logit methods for dose response curves. Biometrics 32, 761–768.
Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P., 2007. Numerical recipes 3rd edition: The art of scientific computing, third ed.. Cambridge University Press, New York, NY, USA.
Pronzato, L., 2008. Optimal experimental design and some related control problems. Automatica 44, 303–325.
Pronzato, L., Pázman, A., 2013. Design of Experiments in Nonlinear Models. Springer.
Pronzato, L., Walter, É., 1988. Robust experiment design via maximin optimization. Math. Biosci. 89 (2), 161–176.
Pukelsheim, F., 1993. Optimal Design of Experiments. SIAM, Philadelphia.
Qiu, J., 2014. Finding Optimal Experimental Designs for Models in Biomedical Studies via Particle Swarm Optimization (Ph.D. thesis), University of California, Los Angeles.
Rao, C.R., 1973. Linear Statistical Inference and its Applications, second ed.. Wiley.
Rustem, B.,Žakovíc, S., Parpas, P., 2008. Convergence of an interior point algorithm for continuous minimax problems. J. Optim. Theory Appl. 136 (1), 87–103.
Ruszczyński, A.P., 2006. Nonlinear Optimization. In: Nonlinear optimization, vol. 13, Princeton University Press.
Sagnol, G., 2011. Computing optimal designs of multiresponse experiments reduces to second-order cone programming. J. Statist. Plann. Inference 141 (5), 1684–1708.
Sagnol, G., 2012. PICOS, a Python interface to conic optimization solvers, Tech. Rep., 12-48, ZIB.
Sagnol, G., 2013. On the semidefinite representation of real functions applied to symmetric matrices. Linear Algebra Appl. 439 (10), 2829–2843.
Sagnol, G., Harman, R., 2015. Computing exact $D-$optimal designs by mixed integer second order cone programming. Ann. Statist. 43 (5), 2198–2224.
Sahinidis, N.V., 2014. BARON 14.3.1: Global Optimization of Mixed-Integer Nonlinear Programs,User's Manual.
Shimizu, K., Aiyoshi, E., 1980. Necessary conditions for min-max problems and algorithms by a relaxation procedure. IEEE Trans. Automat. Control 25, 62–66.
Sturm, J., 1999. Using SeDuMi 1.02, a Matlab toolbox for optimization oversymmetric cones. Optim. Methods Softw. 11, 625–653.
Terry, T.L., 2009. Robust Linear Optimization with Recourse: Solution Methods and Other Properties (Ph.D. thesis), University of Michigan.
Ting, N., 2006. Dose Finding in Drug Development. In: Statistics for Biology and Health, Springer, New York.
Tsoularis, A., Wallace, J., 2002. Analysis of logistic growth models. Math. Biosci. 179 (1), 21–55.
Uciński, D., 2015. An algorithm for construction of constrained $d-$ optimum designs. In: Stochastic Models, Statistics and their Applications. Springer International Publishing, pp. 461–468.
Ugray, Z., Lasdon, L., Plummer, J., Glover, F., Kelly, J., Martí, R., 2005. A multistart scatter search heuristic for smooth NLP and MINLP problems. In: Metaheuristic Optimization Via Memory and Evolution. Springer, pp. 25–51.
Vandenberghe, L., Boyd, S., 1996. Semidefinite programming. SIAM Rev. 8, 49–95.
Vandenberghe, L., Boyd, S., 1999. Applications of semidefinite programming. Appl. Numer. Math. 29, 283–299.
Wächter, A., Biegler, T.L., 2005. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Math. Program. 106 (1), 25–57.
Walter, E., Pronzato, L., 1997. Identification of Parametric Models from Experimental Data. Springer Verlag.
Welch, W.J., 1982. Algorithmic complexity: Three NP-hard problems in computational statistics. J. Statist. Comput. Simul. 15 (1), 17–25.
Whitacre, J.M., 2011a. Recent trends indicate rapid growth of nature-inspired optimization in academia and industry. Computing 93 (2), 121–133.
Whitacre, J.M., 2011b. Survival of the flexible: explaining the recent popularity of nature-inspired optimization within a rapidly evolving world. Computing 93 (2), 135–146.
Whittle, P., 1973. Some general points in the theory of optimal experimental design. J. Roy. Statist. Soc. Ser. B 35, 123–130.
Wong, W., 1992. A unified approach to the construction of minimax designs. Biometrika 79, 611–620.
Yang, X.-S., 2010. Nature-Inspired Metaheuristic Algorithms, second ed.. Luniver Press.
Žakovíc, S., Rustem, B., 2003. Semi-infinite programming and applications to minimax problems. Ann. Oper. Res. 124 (1–4), 81–110.
Zhang, Y., 2006. Bayesian $D-$Optimal Design for Generalized Linear Models (Ph.D. thesis), Virginia Polytechnic Institute and State University.