Noname manuscript No.
(will be inserted by the editor)

Sensitivity analysis for linear changes of the constraint matrix of a linear program

Bardhyl Miftari[1,*] · **Quentin Louveaux**[1] · **Damien Ernst**[1,2] · **Guillaume Derval**[1]

Received: date / Accepted: date

Abstract Understanding the variation of the optimal value with respect to change in the data is an old problem of mathematical optimisation. This paper focuses on the linear problem $f(\lambda) = \min \mathbf{c}^t\mathbf{x}$ such that $(A+\lambda D')\mathbf{x} \leq \mathbf{b}$, where λ is an unknown parameter that varies within an interval and D' is a matrix modifying the coefficients of the constraint matrix A. This problem is used to analyse the impact of multiple affine changes in the constraint matrix on the objective function. The function $f(\lambda)$ can have an erratic behaviour and computing it for a large number of points is computationally heavy while not providing any guarantees in between the computed points. As a new approach to the problem, we derive several bounding methods that provide guarantees on the objective function's behaviour. Those guarantees can be exploited to avoid recomputing the problem for numerous λ. The bounding methods are based on approaches from robust optimisation or Lagrangian relaxations. For each approach, we derive three methods of increasing complexity and precision, one that provides constant bounds, one that provides λ-dependant bounds and one that provides envelope bounds. We assess each bounding method in terms of precision, availability and timing. We show that for a large number of problems, on those metrics, the bound approach is faster than the naive sampling approach considered with 100 points while still providing a good precision and stronger guarantees on a large dataset of problems. We also introduce an iterative algorithm that uses these bounds to compute an approximation of the original function within a given error threshold and discuss its performances.

[1]Department of Electrical Engineering and Computer Science, University of Liège
Liège, Belgium
[2] LTCI, Telecom Paris, Institut Polytechnique de Paris
Paris, France
∗ Corresponding author: bmiftari@uliege.be

1 Introduction

Sensitivity analysis in linear programming is the task of assessing the behaviour of the optimal value of a problem with respect to changes in the data. It is of particular interest for practitioners as it provides multiple insights into the flexibility and stability of a model. In particular, when considering a deterministic problem, it allows for the inclusion of uncertainty in the problem's coefficients, very often with a low cost in terms of computational or modelling effort. For a linear optimisation model, sensitivity analysis for a modification of one right-hand-side coefficient or one cost coefficient is very well-structured and well-understood problem [24,3]. Once we consider the change of several coefficients simultaneously or modification of the constraint coefficients, the structure is lost. Assessing the impact on the objective function of multiple changes in the constraint matrix remains largely an open question. In this paper, we focus on linear modification of the constraint matrix, more specifically on the following problem:

$$\begin{aligned} f(\lambda) = \min \quad & \mathbf{c}^t\mathbf{x} \\ \text{s.t} \quad & A\mathbf{x} + \lambda D'\mathbf{x} \leq \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned} \tag{1}$$

where λ models the uncertainty. From a practitioner's point of view, assessing the impact of the uncertainty λ on $f(\lambda)$ relies on heavy computations, reoptimisations or approximations to compute the objective for a large number of values of λ. In order to alleviate this issue, in this paper, we propose a novel approach consisting of finding upper and lower bounds of the problem depicted in (1) that require fewer computations and give guarantees upon the behaviour of the objective. In the following, we discuss fields and methods related to our approach.

1.1 Related works

Two closely related fields dealing with this problem are Sensitivity Analysis (SA) and Parametric Linear Programming (PLP). They differ based on the assumptions they make upon the modification of λ. Sensitivity analysis considers the effect of the small variation of a parameter around the optimum, whereas linear parametric programming assesses the effect of a parameter on the objective when it varies within a certain range. They both assess the impact of uncertainty on the objective. The methods discussed in this paper are agnostic of the amount of variation and can be applied in both fields.

A complementary field is Robust optimisation (RO). In contrast to both SA and PLP, RO does not assess the impact of the modification on the objective. Indeed, while SA and PLP focus on the evolution of the optimal objective function given the changes in the parameter, RO techniques focus on finding

a solution that is robust to the modification, i.e. a sub-optimal solution that remains feasible for every change in parameter. The three methods, SA, PLP and RO, are complementary as they all try to deal with uncertainty. We note that several methods discussed in this paper use robust optimisation techniques to achieve the sensitivity analysis.

In its most generic form, the linear problem with a single scalar parameter λ, where the objective coefficients, the constraints and the right-hand side can be simultaneously modified, can be written as follows:

$$\begin{aligned} \min \quad & (\mathbf{c} + \lambda \mathbf{c}')^t \mathbf{x} \\ \text{s.t} \quad & (A + \lambda D')\mathbf{x} \leq \mathbf{b} + \lambda \mathbf{b}' \\ & \mathbf{x} \geq \mathbf{0} \end{aligned} \tag{2}$$

where λ is a parameter that varies in a given range $[\underline{\lambda}, \bar{\lambda}]$; $\mathbf{c}'$, D' and $\mathbf{b}'$ are the parameter's impact on the objective, constraint matrix and right-hand-side term, respectively. The end goal is to evaluate the function objective function $f(\lambda)$ over the range $[\underline{\lambda}, \bar{\lambda}]$.

Reoptimising the problem from scratch for a finite subset of value of λ in $[\underline{\lambda}, \bar{\lambda}]$ can be computationally very costly, especially for big problems that take several minutes or hours to solve once (for a single λ) and does not provide any guarantees in between the computed points. Therefore, several approaches have been considered to mitigate this issue. In the following, let us consider $\mathbf{x}^\star$ and $\boldsymbol{\rho}^\star$ the primal and dual optimal solutions of the unmodified problem, i.e. where $\lambda = 0$:

$$\begin{aligned} \mathbf{x}^\star = \text{argmin} \quad & \mathbf{c}^t \mathbf{x} \\ \text{s.t} \quad & A\mathbf{x} \leq \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \end{aligned} \qquad\qquad \begin{aligned} \boldsymbol{\rho}^\star = \text{argmax} \quad & \mathbf{b}^\mathbf{t} \boldsymbol{\rho} \\ \text{s.t} \quad & A^t \boldsymbol{\rho} \geq \mathbf{c} \\ & \boldsymbol{\rho} \leq \mathbf{0}. \end{aligned}$$

The existing literature mainly focuses on one specific modification at a time; in general, either the objective (via $\mathbf{c}'$), or the right-hand side (RHS, via $\mathbf{b}'$), or the left-handside (LHS, via $\mathbf{D}'$) is modified in (2).

Modification of either the objective or the right-hand side. The modifications on the objective and the right-hand side are closely related as by dualising the problem, one can be transformed into the other. Note that when the modification is only performed on the objective, i.e. only $\mathbf{c}' \neq \mathbf{0}$, the previous primal optimal solution $\mathbf{x}^\star$ remains feasible and similarly, the previous optimal dual solution $\boldsymbol{\rho}^\star$ is still a feasible solution. The methods used for these types of modifications can be categorized into three families [12]:

- *ones using optimal partitions.* The set of active (tight) constraints on the primal and on the dual form an optimal partition of the linear problem. From this partition, we can derive validity intervals upon λ. The methods derived from this family use an LP solver as a subroutine to solve the problem and iteratively find these partitions. Adler and Monteiro [1] introduce

the first algorithm from thus family. Berkelaar et al [2] introduce another algorithm based on the additional property that either the primal or the dual optimal set remains unchanged when $\mathbf{c}'$ or $\mathbf{b}'$ is non-zero.
- *ones using optimal values.* The optimal values of the primal, dual and objective are used to build two auxiliary LP problems that give the range upon which solution stays optimal.
- *ones using optimal basis.* The simplex algorithm returns the optimal basis related to a problem. These methods use warm starting and properties related to the basis to perform only a few iterations of the simplex to find another basis when the objective function changes. A two-part paper from Gass and Saaty [20,9] uses a modified simplex method to retrieve the optimal solution for multiple changes in the coefficient of the objective function. Gal and Nedoma [7] present an effective method for finding the regions that keep a basis optimal for multiple changes in the parameters for both types of modification using the simplex tableaux of multiple reoptimisation and graph theory to combine the tableaux.

Various works [11,4,12] make a summary of some techniques and conceptual foundation for post-optimality analysis for modifications on $\mathbf{c}$ and $\mathbf{b}$.

Modifications on the constraint matrix. It should be noted that when considering modifications $\mathbf{c}'$ and $\mathbf{b}'$, there always exists an equivalent problem (of the form (2)) that encompasses these modifications inside its matrix D', and such that its objective and right-hand side do not depend on λ. The converse is not true: the modifications of the constraint matrix are more general. Gal [8] presents a summary of the different existing techniques for modifications on the constraint matrix A, mostly based on two papers [22,21]. Sherman and Morison [22] offer a methodology for adjusting an inverse matrix with respect to change of one of its entries. Sherman and Morison [21] offer a formula for the objective considering the rank-one modification $D' = \mathbf{u^t v}$. Both of these methods lack genericity. Woodbury [23] generalizes the Sherman and Morison formula for any decomposition $D' = UCV$ which can hardly be applied in our case due to having a λ-dependent matrix inversion which what we are trying to avoid.

More recently, Zuidwijk [25] derives explicit local formulae to compute the evolution of the objective function and intervals on which these formulae are valid. They rewrite the basic inverse matrix as a function of λ and recompute it for several values in the range. Their algorithm involves the finding of the optimal basis, computation of the range of λ for which the basis stays optimal, the evaluation of the derived formulae using the said basis and reoptimisation when switching to another basis. Their formulae for computing the objective for a given basis require the computation of the eigenvalues of the matrices A and D' and leads to a polynomial problem with orders equal to the number of constraints. For big LPs, finding the solution of the polynomial problem can only be done using approximation methods. Khalipour et al. [13] also discusses

an algorithm similar to Zuidwijk [25]. However, they claim that their algorithm can be applied when working on larger problems. They use the Flavell and Salkin [6] formula to compute an approximation of the basic inverse matrix. All of the above-mentioned methods for the modification of the constraint matrix rely either on heavy computations, reoptimisations, approximations or are not sufficiently generic. Additionally, we underline that even discretizing $[\underline{\lambda}, \overline{\lambda}]$, with a very fine granularity, at the price of heavy computations, may still lead to a false assessment of the behavior of $f(\lambda)$ over the interval. Indeed, as the matrix D' can contain an arbitrary number of constraints, the resulting problem for a given λ can behave in an unpredictable and sometimes erratic fashion in between two closely located points.

1.2 Our contributions

In this paper, we propose methods to compute upper and lower bounds of problems given in (1), for varying $\lambda \in [\underline{\lambda}, \bar{\lambda}]$. The bounding methods provide guarantees between the computed points. They can capture any erratic behaviour that might be missed by sampling approaches. They also give an indication of the evolution of the objective and can help practitioners to focus on particular values of λ within the interval that have an interesting or unconventional behaviour. The bounding methods are based on two approaches: robust optimisation and Lagrangian relaxations. Out of each of these two approaches, we derive bounding methods of three types: ones that computes constant bounds, ones giving variables bounds depending on λ and ones computing envelopes on the objective function. The envelopes are a combination of multiple variable bounds, proven optimal in the sense that no other variable bound of the same type is tighter, that form an envelope around the objective function. Some methods add new degrees of freedom, and therefore, can lead to multiple variations. We also introduce an iterative algorithm to compute these bounds with relatively small gaps and discuss its efficiency.

Paper outline. In Section 2, we formalise and analyse the problem at hand. In Section 3, we introduce all the bounding methods in the form of theorems and provide the associated proofs. A summary of the bounding methods is given in Table 1. The bounds are derived by applying the bounding methods on the primal or dual. In Section 4, we introduce a new dataset of problems to serve as a benchmark and perform two experiments. First, we benchmark the different bounds on the data problems in terms of availability, precision and timing for providing upper and lower bounds compared to the "naive" solution consisting of sampling 100 points in the interval. In that experiment, we divide the uncertainty interval into into $1, 5$ and 10 uniform subintervals to assess the impact of refining the uncertainty interval on the bounds in terms of the three metrics. In the second experiment, we introduce an iterative algorithm that uses lower and upper bounds to compute the objective function and refines the uncertainty interval to minimize the gap between the bounds.

Approach	Type	Section
Robust optimisation	Constant	3.2
	Linear in λ	3.3
	Envelope	3.4
Lagrangian Relaxations	Constant	3.5
	Linear and quadratic function of λ	3.6
	Envelope	3.7

Table 1: Summary of all the methods presented in this paper.

Notations

We provide a concise reference describing the notation. We denote a scalar with a standard-font symbol a, a vector with a bold lower case symbol $\mathbf{a}$, a matrix with an uppercase letter A, a vector space with a calligraphic uppercase letter $\mathcal{A}$. The i^{th} element of a vector $\mathbf{a}$ is noted a^i. The i^{th} row and j^{th} column of a matrix A are respectively given by $\mathbf{a}^{i,\cdot}$ and $\mathbf{a}^{\cdot,j}$. The element i,j of a matrix A is given by $a^{i,j}$.

2 Problem formalisation

In this section, we provide a full formalisation of the problem we consider.

Let us denote by $\mathcal{P}(\lambda)$ the following optimisation problem:

$$\begin{aligned} \mathcal{P}(\lambda) \equiv \min \quad & \mathbf{c}^t\mathbf{x} \\ \text{s.t} \quad & A\mathbf{x} + \lambda D'\mathbf{x} \leq \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0}. \end{aligned}$$

Thus, with this notation, the function $f(\lambda)$ defined in (2) is the optimal objective function of $\mathcal{P}(\lambda)$.

We can modify this problem without losing any generality: we decompose the constraint matrix $A \in \mathbb{R}^{m\times n}$ in two matrices A_1 and A_2 of respective size $m_1 \times n$ and $m_2 \times n$, with $m_1 + m_2 = m$. The matrix A_1 encompasses all the constraints upon which there are no modifications depending on the external parameter λ, while A_2 contains the constraints affected by the modification $\lambda D'$. Similarly, we divide $\mathbf{b}$ in $\mathbf{b_1}$ and $\mathbf{b_2}$ of respective size $m_1 \times 1$ and $m_2 \times 1$. As we can have $m_1 = 0$, we indeed do not lose any generality. This results in the following equivalent problem:

$$\begin{aligned} \mathcal{P}(\lambda) \equiv \min \quad & \mathbf{c}^t\mathbf{x} \\ \text{s.t} \quad & A_1\mathbf{x} \leq \mathbf{b_1} \\ & A_2\mathbf{x} + \lambda D\mathbf{x} \leq \mathbf{b_2}. \end{aligned} \tag{3}$$

For a fixed value λ_1, an optimal solution of $\mathcal{P}(\lambda_1)$ is denoted $\mathbf{x}^*_{\mathcal{P}(\lambda_1)}$ and an optimal dual solution $\boldsymbol{\rho}^*_{\mathcal{P}(\lambda_1)}$. For the sake of readability, when explicit, we may omit the subscript of $\mathbf{x}^*_{\mathcal{P}(\lambda_1)}$ and $\boldsymbol{\rho}^*_{\mathcal{P}(\lambda_1)}$ and write $\mathbf{x}^*$ and $\boldsymbol{\rho}^*$.

The goal of this paper is to find $ub_{\underline{\lambda},\overline{\lambda}}(\lambda)$ and $lb_{\underline{\lambda},\overline{\lambda}}(\lambda)$, respectively an upper bound and lower bound of the optimal value on the problem written in (3), i.e. $lb_{\underline{\lambda},\overline{\lambda}}(\lambda) \leq f(\lambda) \leq ub_{\underline{\lambda},\overline{\lambda}}(\lambda), \forall \lambda \in [\underline{\lambda}, \overline{\lambda}]$. In general, $f(\lambda)$ has no desirable properties. Depending on $A_1, A_2, D, \mathbf{b_1}$ and $\mathbf{b_2}$, it is generally non-convex, non-concave and non-differentiable. As the problem may be unbounded or infeasible for some λ, it can even be non-continuous. In the following, we illustrate these behaviours with two examples.

Example 1 Let us consider $A_1 = 0$, $\mathbf{b_1} = 0$, $D = -A_2$ and $\lambda \in [0, 2]$, the problem $\mathcal{P}(\lambda)$ becomes,

$$\begin{aligned} \min \quad & \mathbf{c}^t\mathbf{x} \\ \text{s.t } & A_2\mathbf{x} - \lambda A_2\mathbf{x} \leq \mathbf{b_2}. \end{aligned}$$

For $\lambda = 1$, the problem is infeasible if $\mathbf{b_2} < \mathbf{0}$ or unbounded if $\mathbf{b_2} \geq \mathbf{0}$.

Example 2 In this example, we showcase the erratic behaviour of $f(\lambda)$. Let us consider the following problem,

$$\begin{aligned} \mathcal{P}_{toy}(\lambda) \equiv \min \quad & \begin{pmatrix} 2 & -2 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \\ \text{s.t} \quad & \begin{pmatrix} -2 & 2 \\ -1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \leq \begin{pmatrix} 4 \\ 1 \end{pmatrix} \\ & \begin{pmatrix} 2 & 1 \\ -2 & -3 \\ 2 & 2 \\ -1 & -4 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \lambda \begin{pmatrix} -1 & -4 \\ 0 & 4 \\ -4 & -3 \\ 2 & 4 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \leq \begin{pmatrix} 4 \\ 2 \\ 0 \\ 2 \end{pmatrix} \end{aligned} \tag{4}$$

$\forall \lambda \in [-10, 9]$.

We denote the associated optimal objective function by $f_{\text{toy}}(\lambda)$.

To highlight the difficulty of predicting the behaviour of the objective function of $f_{\text{toy}}(\lambda)$ through sampling and choosing the set size for sampling, we compute $f_{\text{toy}}(\lambda)$ for the same range of λ values whilst using two different levels of granularity. On one hand, we use a coarser discretisation step of 0.01 for all the optimal objectives for $\lambda \in [-10, 9]$ shown as by the blue curve in Figure 1. As we can see, the value of $f(\lambda)$ varies strongly in a neighbourhood located after $\lambda = 0.5$ and is neither concave nor convex. On the other, we use a discretisation step of 1 shown as by the orange curve on the same figure. As we can see, with this discretisation step, we would have missed the erratic behaviour of the function after 0, illustrated by the blue peak. While the difference between the two steps is rather large in the context of this toy example, this problem can also occur in much larger problems at much finer scales.

In Figure 2, we show the evolution of the feasible space for three values of λ in the interval $[0, 1]$ where $f_{\text{toy}}(\lambda)$ shows an erratic behaviour. We see that

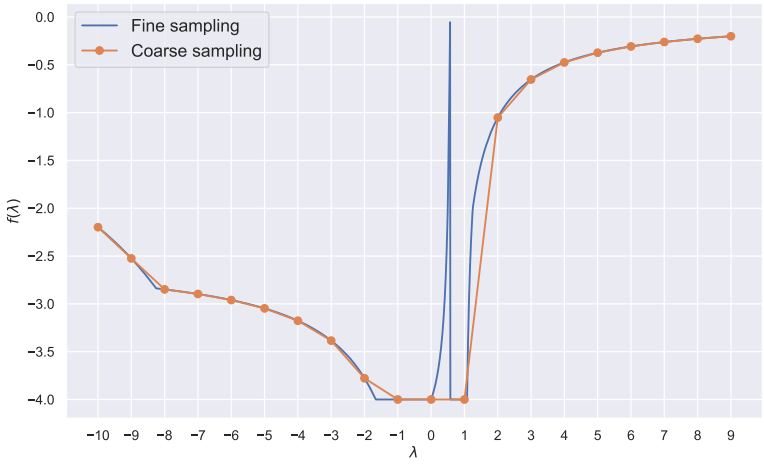

Fig. 1: Plot of $f_{\text{toy}}(\lambda)$ between $[-10, 9]$. The orange line with dots is a coarse sampling of the function, made for each $\lambda \in \{-10, 9\}$, and linearly interpolated between these points.

the feasible space changes constantly due to some constraints switching from tight to non-tight and vice-versa, with small changes in λ.

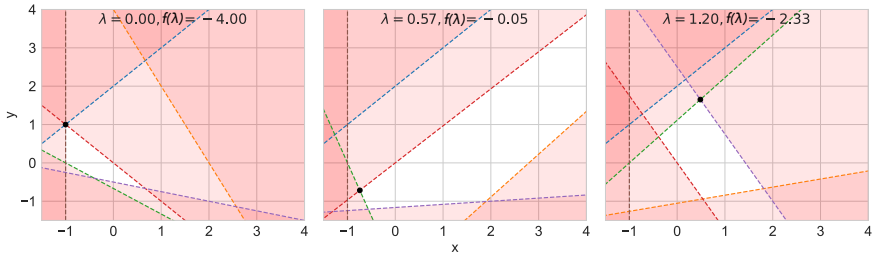

Fig. 2: The feasible space of the problem given in Equation (4) for different values of λ. Each inequality partitions the space into two parts: a feasible space, coloured in white, and a non-feasible space, coloured in red. The optimum is shown as a black dot.

3 Bounding methods formulations

In this section, we show how to compute bounds on the optimal objective function $f(\lambda)$. We first discuss the naive method that consists of reusing an optimal solution as long as it remains feasible and highlight its limitations. Then, we present our two approaches:

- Robust optimisation: we reformulate the inner problem using several robust optimisation bounding methods separated in three types:
  - one consisting of a constant solution for a given range,
  - a second solution linearly dependent on λ,
  - a third one takes a combination of affinely dependent solution, to form an envelope around the objective function.

  These techniques applied on the primal problem provide upper bounds.
- Lagrangian relaxation: we dualise the constraints linked to the modification, $A_2\mathbf{x} + \lambda D\mathbf{x} \leq \mathbf{b_2}$. Depending on the choice of the Lagrangian multiplier, we get three types of this bounding method: constant, polynomially-dependent on λ, and an envelope formed from a set of λ-dependant solutions.

These techniques are discussed in the next section on the primal. Their application to the dual problem leads to the opposite type of bound. In other words, if the technique applied on the primal gives a lower bound $lb(\lambda)$, resp. upper bound $ub(\lambda)$, its application on the dual yields an upper bound $ub(\lambda)$, resp. lower bound $lb(\lambda)$ on $f(\lambda)$.

But first, for the sake of completeness, we discuss briefly the naive method of reusing an optimal solution as long as it remains feasible.

3.1 Optimum reuse

An optimal solution $\mathbf{x}^*_{\mathcal{P}_{\lambda_1}}$ for a given λ_1 may be also a solution of the problem for other values of λ. This is a very simplistic way to obtain a bound quickly. We want to find the interval $[\underline{\lambda_1}, \overline{\lambda_1}]$ upon which $\mathbf{x}^*_{\mathcal{P}_{\lambda_1}}$ is still a feasible solution. By proceeding this way, we limit the number of reoptimisations of $\mathcal{P}(\lambda)$.

Theorem 1 *A solution $\mathbf{x}^*_{\mathcal{P}_{\lambda_1}}$ to the problem $\mathcal{P}(\lambda_1)$ is a solution to $\mathcal{P}(\lambda)$ $\forall \lambda \in [\underline{\lambda_1}, \overline{\lambda_1}]$, with*

$$\underline{\lambda_1} = \max_{i|\mathbf{d}^{i,\cdot}\mathbf{x}^*_{\mathcal{P}_{\lambda_1}} > \mathbf{0}} \left(\frac{b_2^i - \mathbf{a}_2^{i,\cdot}\mathbf{x}^*_{\mathcal{P}_{\lambda_1}}}{\mathbf{d}^{i,\cdot}\mathbf{x}^*_{\mathcal{P}_{\lambda_1}}} \right) \quad \overline{\lambda_1} = \min_{i|\mathbf{d}^{i,\cdot}\mathbf{x}^*_{\mathcal{P}_{\lambda_1}} < \mathbf{0}} \left(\frac{b_2^i - \mathbf{a}_2^{i,\cdot}\mathbf{x}^*_{\mathcal{P}_{\lambda_1}}}{\mathbf{d}^{i,\cdot}\mathbf{x}^*_{\mathcal{P}_{\lambda_1}}} \right). \quad (5)$$

Moreover, $f(\lambda_1)$ is an upper bound for every problem $\mathcal{P}(\lambda)$ with $\lambda \in [\underline{\lambda_1}, \overline{\lambda_1}]$.

Proof We want to identify the interval $[\underline{\lambda_1}, \overline{\lambda_1}]$ so that the solution $\mathbf{x}^\star_{\mathcal{P}_{\lambda_1}}$ remains feasible for all $\lambda \in [\underline{\lambda_1}, \overline{\lambda_1}]$. In order to remain feasible, the solution $\mathbf{x}^\star_{\mathcal{P}_{\lambda_1}}$

needs to respect the following conditions:

$$\begin{aligned} A_1\mathbf{x}^{\star}_{\mathcal{P}_{\lambda_1}} &\leq \mathbf{b_1} \\ A_2\mathbf{x}^{\star}_{\mathcal{P}_{\lambda_1}} + \lambda D\mathbf{x}^{\star}_{\mathcal{P}_{\lambda_1}} &\leq \mathbf{b_2}. \end{aligned} \tag{6}$$

The constraints $A_1\mathbf{x}^{*}_{\mathcal{P}_{\lambda_1}} \leq \mathbf{b_1}$ are always verified as they do not depend on the value of λ. The constraints $A_2\mathbf{x}^{*}_{\mathcal{P}_{\lambda_1}} + \lambda D\mathbf{x}^{*}_{\mathcal{P}_{\lambda_1}} \leq b_2$ depend on λ.

As we consider $\mathbf{x}^{*}_{\mathcal{P}_{\lambda_1}}$ as fixed, it is a system of m_2 inequations with one variable λ. The system $\lambda D\mathbf{x}^{*}_{\mathcal{P}_{\lambda_1}} \leq b_2 - A_2\mathbf{x}^{*}_{\mathcal{P}_{\lambda_1}}$ can be written as

$$\begin{aligned} \lambda(\mathbf{d}^{1,\cdot}\mathbf{x}^{*}_{\mathcal{P}_{\lambda_1}}) &\leq b_2^1 - \mathbf{a}_2^{1,\cdot}\mathbf{x}^{*}_{\mathcal{P}_{\lambda_1}} && \text{Constraint 1} \\ \lambda(\mathbf{d}^{2,\cdot}\mathbf{x}^{*}_{\mathcal{P}_{\lambda_1}}) &\leq b_2^2 - \mathbf{a}_2^{2,\cdot}\mathbf{x}^{*}_{\mathcal{P}_{\lambda_1}} && \text{Constraint 2} \\ &\dots && \dots \\ \lambda(\mathbf{d}^{n_2,\cdot}\mathbf{x}^{*}_{\mathcal{P}_{\lambda_1}}) &\leq b_2^{n_2} - \mathbf{a}_2^{n_2,\cdot}\mathbf{x}^{*}_{\mathcal{P}_{\lambda_1}} && \text{Constraint } n_2. \end{aligned}$$

Based on the sign of term $\mathbf{d}^{i,\cdot}\mathbf{x}^{*}_{\mathcal{P}_{\lambda_1}}$ we have three possibilities for the i^{th} constraint:

- $\mathbf{d}^{i,\cdot}\mathbf{x}^{*}_{\mathcal{P}_{\lambda_1}} = \mathbf{0}$ in which case the constraint is not constraining λ. The term $b_2^i - \mathbf{a}_2^{i,\cdot}\mathbf{x}^{*}_{\mathcal{P}_{\lambda_1}}$ must be positive (since the solution $\mathbf{x}^{*}_{\mathcal{P}_{\lambda_1}}$ is feasible for $\mathcal{P}(\lambda_1)$),
- $\mathbf{d}^{i,\cdot}\mathbf{x}^{*}_{\mathcal{P}_{\lambda_1}} > \mathbf{0}$ in which case λ is constrained to be less than $\frac{b_2^i - \mathbf{a}_2^{i,\cdot}\mathbf{x}^{*}_{\mathcal{P}_{\lambda_1}}}{\mathbf{d}^{i,\cdot}\mathbf{x}^{*}_{\mathcal{P}_{\lambda_1}}}$, and
- $\mathbf{d}^{i,\cdot}\mathbf{x}^{*}_{\mathcal{P}_{\lambda_1}} < \mathbf{0}$ in which case, in which case λ is constrained to be more than $\frac{b_2^i - \mathbf{a}_2^{i,\cdot}\mathbf{x}^{*}_{\mathcal{P}_{\lambda_1}}}{\mathbf{d}^{i,\cdot}\mathbf{x}^{*}_{\mathcal{P}_{\lambda_1}}}$.

By combining all the constraints on λ, we find the maximal-length interval $[\underline{\lambda_1}, \overline{\lambda_1}]$, proving the first claim.

The second claim is trivial: if $\mathbf{x}^{*}_{\mathcal{P}_{\lambda_1}}$ is a solution of both the problems $\mathcal{P}(\lambda_1)$ and $\mathcal{P}(\lambda_2)$, then the objective value for both problems for this solution is $\mathbf{c}^t\mathbf{x}^{*}_{\mathcal{P}_{\lambda_1}}$ (the two problems share the same vector $\mathbf{c}$). The solution is optimal for $f(\lambda_1) = \mathbf{c}^t\mathbf{x}^{*}_{\mathcal{P}_{\lambda_1}}$. As it may be non-optimal for $f(\lambda_2)$, then $f(\lambda_2) \leq \mathbf{c}^t\mathbf{x}^{*}_{\mathcal{P}_{\lambda_1}} = f(\lambda_1)$. □

We can thus derive an interval $[\underline{\lambda_1}, \overline{\lambda_1}]$ for which the solution $\mathbf{x}^{*}_{\mathcal{P}_{\lambda_1}}$ remains feasible and provides an upper bound. In practice, however, the interval can be arbitrarily small, or may even contain only λ_1; thus, we would still need to recompute λ for a large number of points within the range $[\underline{\lambda}, \overline{\lambda}]$ explaining the unpractical and simplistic character of this bound.

3.2 Constant robust solutions

In the previous subsection, we have used an optimal solution of $\mathcal{P}(\lambda_1)$ to derive a bound for a (possibly very small) interval around λ_1. However, nothing

prevents us from sacrificing the guaranteed optimality of the solution on λ_1 for a feasible solution on the whole interval $[\underline{\lambda}, \overline{\lambda}]$. Therefore, we turn to robust optimisation in order to search for a feasible solution that provides the tightest bound on this interval.

An upper bound can be obtained by finding a solution that satisfies $(A_2 + \lambda D)\mathbf{x} \leq \mathbf{b}_2 \ \forall \lambda \in [\underline{\lambda}, \overline{\lambda}]$. The corresponding robust optimisation problem can be written as follows and gives an upper bound on $f(\lambda)$ in the range $[\underline{\lambda}, \overline{\lambda}]$:

$$\begin{aligned} \mathcal{P}^{\mathrm{CR}}_{\underline{\lambda},\overline{\lambda}}(\lambda) \equiv \ & \underset{x}{\text{minimize}} \quad \mathbf{c}^t\mathbf{x} \\ & \text{subject to} \quad A_1\mathbf{x} \leq \mathbf{b_1} \\ & \qquad (A_2 + \lambda D)\mathbf{x} \leq \mathbf{b_2} \ \forall \lambda \in [\underline{\lambda}, \overline{\lambda}] \ . \end{aligned} \tag{7}$$

This new linear reformulation of the initial problem is unusual in the sense that it possesses an infinite number of constraints, one for each $\lambda \in [\underline{\lambda}, \overline{\lambda}]$. It is possible to reformulate the problem in a short finite optimisation problem.

Theorem 2 *The following linear problem is equivalent to problem* (7) *and provides an upper bound for* $f(\lambda) \ \forall \lambda \in [\underline{\lambda}, \overline{\lambda}]$*:*

$$\begin{aligned} \mathcal{P}^{CR}_{\underline{\lambda},\overline{\lambda}}(\lambda) \equiv \ & \underset{\mathbf{x}}{\text{minimize}} \quad \mathbf{c}^t\mathbf{x} \\ & \text{subject to} \quad A_1\mathbf{x} \leq \mathbf{b_1} \\ & \qquad A_2\mathbf{x} + \underline{\lambda} D\mathbf{x} \leq \mathbf{b_2} \\ & \qquad A_2\mathbf{x} + \overline{\lambda} D\mathbf{x} \leq \mathbf{b_2} \ . \end{aligned} \tag{8}$$

Proof Any feasible solution $\mathbf{x}$ to $\mathcal{P}^{\mathrm{CR}}_{\underline{\lambda},\overline{\lambda}}(\lambda)$ must respect the following condition:

$$(A_2 + \lambda D)\mathbf{x} \leq \mathbf{b_2} \ \forall \lambda \in [\underline{\lambda}, \overline{\lambda}] \ . \tag{9}$$

This can be reformulated using as its robust counterpart

$$\max_{\lambda \in [\underline{\lambda}, \overline{\lambda}]} (A_2 + \lambda D)\mathbf{x} \leq \mathbf{b_2} \tag{10}$$

which is equivalent to

$$A_2\mathbf{x} + \max_{\lambda \in [\underline{\lambda}, \overline{\lambda}]} \lambda D\mathbf{x} \leq \mathbf{b_2}, \tag{11}$$

where the max operator is here applied component-wise.

If we have $\mathbf{d}^{i,\cdot}\mathbf{x} \geq 0$ (resp. ≤ 0), then $\lambda = \overline{\lambda}$ (resp. $\lambda = \underline{\lambda}$) is an optimal solution to $\max_{\lambda \in [\underline{\lambda}, \overline{\lambda}]} \lambda \mathbf{d}^{i,\cdot}\mathbf{x}$. Satisfying these two cases is thus equivalent to the original set of constraints:

$$\begin{cases} A_2\mathbf{x} + \underline{\lambda} D\mathbf{x} \leq \mathbf{b_2} \\ A_2\mathbf{x} + \overline{\lambda} D\mathbf{x} \leq \mathbf{b_2} \end{cases} \tag{12}$$

For each component i of the vector $\mathbf{b_2}$, one of these constraints is redundant. Hence, Problem (7) is equivalent to (8). □

This reformulation has n variables and $m_1 + 2m_2$ constraints. It should be noted that for two given values of $\underline{\lambda}$ and $\overline{\lambda}$, the feasible space respecting both $A_2\mathbf{x} + \underline{\lambda}D\mathbf{x} \leq \mathbf{b_2}$ and $A_2\mathbf{x} + \overline{\lambda}D\mathbf{x} \leq \mathbf{b_2}$ can be empty if at least two constraints are conflicting.

Example 3 The following problem is a very simple case where constraints can be conflicting:

$$\begin{aligned} &\min_{x,y} \begin{pmatrix} -2 & -2 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \\ &\text{s.t. } \begin{pmatrix} 3 & 1 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \leq \begin{pmatrix} 3 \\ 3 \end{pmatrix} \\ &\quad \begin{pmatrix} -5 & -2 \\ 1 & 4 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \lambda \begin{pmatrix} -3 & -2 \\ -3 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \leq \begin{pmatrix} 0 \\ -3 \end{pmatrix} \\ &\quad \forall \lambda \in [-2, 2] \ . \end{aligned} \tag{13}$$

Its objective depending on λ is shown in Figure 3.

By applying the Theorem 2, an upper bound is given by,

$$\begin{aligned} &\min_{x,y} \begin{pmatrix} -2 & -2 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \\ &\text{s.t. } \begin{pmatrix} 3 & 1 \\ 0 & -1 \\ 1 & 2 \\ 7 & 4 \\ -11 & -6 \\ -5 & 4 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \leq \begin{pmatrix} 3 \\ 3 \\ 0 \\ 0 \\ -3 \\ -3 \end{pmatrix} \end{aligned} \tag{14}$$

We show by applying Farkas' lemma that the problem (14) is infeasible. Farkas' lemma states that for a given problem $\min c^t\mathbf{x}$ s.t. $A\mathbf{x} \leq \mathbf{b}$ if we find a vector $\mathbf{u} \geq \mathbf{0}$ such that $\mathbf{u}^tA = \mathbf{0}$ and $\mathbf{u}^t\mathbf{b} < \mathbf{0}$ then this problem is infeasible. If we consider $\mathbf{u} = [2, 0, 0, 7, 5, 0]$, we indeed have $\mathbf{u}^tA = \mathbf{0}$ and $\mathbf{u}^t\mathbf{b} = -9$.

3.3 Variable robust solution, affine on λ

A second possibility is to find solutions that vary linearly in λ. We perform a reformulation of all the variables $\mathbf{x}$ by replacing them with $\mathbf{y} + \lambda\mathbf{z}$, a linear function of λ. This transformation allows us to find a solution that adapts to a change in the value of λ.

We aim to find an upper bound $\mathrm{ub}^{\mathrm{rl}}_{\underline{\lambda},\overline{\lambda}}(\lambda)$ on the function $f(\lambda)$ of the form $\mathbf{c}^t(\mathbf{y}+\lambda\mathbf{z})$. The *rl* superscript stands for **r**obust **l**inear. To ensure that $\mathrm{ub}^{\mathrm{rl}}_{\underline{\lambda},\overline{\lambda}}(\lambda)$ is indeed an upper bound, the newly introduced variables $\mathbf{y}$ and $\mathbf{z}$ must satisfy the following conditions:

$$A_1(\mathbf{y} + \lambda\mathbf{z}) \leq \mathbf{b_1} \qquad \forall \lambda \in [\underline{\lambda}, \overline{\lambda}] \tag{15}$$

$$(A_2 + \lambda D)(\mathbf{y} + \lambda\mathbf{z}) \leq \mathbf{b_2} \qquad \forall \lambda \in [\underline{\lambda}, \overline{\lambda}] \tag{16}$$

As in the previous case, we can rewrite this infinite number of constraints to a finite albeit more constrained form and obtain the following Theorem.

Theorem 3 *Any* $\mathbf{y}$ *and* $\mathbf{z}$ *satisfying the constraints*

$$\begin{cases} A_1\mathbf{y} + \underline{\lambda}A_1\mathbf{z} \leq \mathbf{b_1} \\ A_1\mathbf{y} + \overline{\lambda}A_1\mathbf{z} \leq \mathbf{b_1} \\ (A_2 + \underline{\lambda}D)(\mathbf{y} + \underline{\lambda}\mathbf{z}) \leq \mathbf{b_2} \\ (A_2 + \overline{\lambda}D)(\mathbf{y} + \overline{\lambda}\mathbf{z}) \leq \mathbf{b_2} \\ A_2\mathbf{y} + D\mathbf{z}\underline{\lambda}\overline{\lambda} + (D\mathbf{y} + A_2\mathbf{z})\frac{\underline{\lambda}+\overline{\lambda}}{2} \leq \mathbf{b_2} \end{cases} \tag{17}$$

also satisfy both equations (15) *and* (16) *and thus provide an upper bound in the form* $ub^{rl}_{\underline{\lambda},\overline{\lambda}}(\lambda) = \mathbf{c}^t(\mathbf{y} + \lambda\mathbf{z}) \geq f(\lambda)\ \forall\lambda \in [\underline{\lambda}, \overline{\lambda}]$.

Note that the converse is not true: it is only a sufficient condition.

Proof We write the robust counterpart in order to obtain a finite number of constraints. Let us first focus on constraint (15):

$$A_1(\mathbf{y} + \lambda\mathbf{z}) \leq \mathbf{b_1}\ \forall\lambda \in [\underline{\lambda}, \overline{\lambda}] \tag{18}$$

which can be rewritten as

$$\max_{\lambda\in[\underline{\lambda},\overline{\lambda}]} A_1(\mathbf{y} + \lambda\mathbf{z}) \leq \mathbf{b_1} \tag{19}$$

and

$$A_1\mathbf{y} + \max_{\lambda\in[\underline{\lambda},\overline{\lambda}]} \lambda A_1\mathbf{z} \leq \mathbf{b_1} \tag{20}$$

As the max operator is applied component-wise, for every constraint in $\max_{\lambda\in[\underline{\lambda},\overline{\lambda}]} \lambda A_1\mathbf{z}$, the maximum is either achieved for $\lambda = \underline{\lambda}$ or $\lambda = \overline{\lambda}$.

$$A_1\mathbf{y} + \max_{\lambda\in[\underline{\lambda},\overline{\lambda}]} \lambda A_1\mathbf{z} \leq \mathbf{b_1} \equiv \begin{cases} A_1\mathbf{y} + \underline{\lambda}A_1\mathbf{z} \leq \mathbf{b_1} \\ A_1\mathbf{y} + \overline{\lambda}A_1\mathbf{z} \leq \mathbf{b_1} \end{cases}. \tag{21}$$

Hence, any $\mathbf{x}$ satisfying (21) also satisfies (15) and conversely. The case of (16) is more complex as its left-hand side is a quadratic function of λ.

$$(A_2 + \lambda D)(\mathbf{y} + \lambda\mathbf{z}) = A_2\mathbf{y} + \lambda(D\mathbf{y} + A_2\mathbf{z}) + \lambda^2 D\mathbf{z} \leq \mathbf{b_2}\ \forall\lambda \in [\underline{\lambda}, \overline{\lambda}] \tag{22}$$

Again, we rewrite it using the robust counterpart as a maximization

$$\max_{\lambda\in[\underline{\lambda},\overline{\lambda}]} A_2\mathbf{y} + \lambda(D\mathbf{y} + A_2\mathbf{z}) + \lambda^2 D\mathbf{z} \leq \mathbf{b_2} \tag{23}$$

$$:= \max_{\lambda\in[\underline{\lambda},\overline{\lambda}]} g(\lambda) \leq \mathbf{b_2}. \tag{24}$$

The function $g(\lambda)$ is a quadratic function of λ. The maximum of the quadratic equation can be found analytically but is itself non-linear in z. Instead of directly using $g(\lambda)$, we propose using a piecewise-linear upper bound of this function in the above condition; this allows us to relax (16) into a linear programme in $\mathbf{y}$ and $\mathbf{z}$.

For this, we can use the following lemma:

Lemma 1 *Given a quadratic function $q(x) = ax^2+bx+c$. The maximum of $q(x)$ over $x_1 \leq x \leq x_2$ is upper bounded by*

$$\max \begin{cases} ax_1^2 + bx_1 + c \\ ax_2^2 + bx_2 + c \\ ax_1x_2 + b\frac{x_1+x_2}{2} + c. \end{cases}$$

This lemma is proved in Annex A.1. We apply the lemma on $g(\lambda)$ and we obtain that

$$\max_{\lambda \in [\underline{\lambda}, \overline{\lambda}]} g(\lambda) \leq \max \begin{cases} (A_2 + \underline{\lambda} D)(\mathbf{y} + \underline{\lambda}\mathbf{z}) \\ (A_2 + \overline{\lambda} D)(\mathbf{y} + \overline{\lambda}\mathbf{z}) \\ A_2\mathbf{y} + D\mathbf{z}\underline{\lambda}\overline{\lambda} + (D\mathbf{y} + A_2\mathbf{z})\frac{\underline{\lambda}+\overline{\lambda}}{2} \end{cases} \tag{25}$$

Imposing the following is thus sufficient to obtain $\max_{\lambda \in [\underline{\lambda}, \overline{\lambda}]} g(\lambda) \leq \mathbf{b_2}$:

$$\begin{cases} (A_2 + \underline{\lambda} D)(\mathbf{y} + \underline{\lambda}\mathbf{z}) \leq \mathbf{b_2} \\ (A_2 + \overline{\lambda} D)(\mathbf{y} + \overline{\lambda}\mathbf{z}) \leq \mathbf{b_2} \\ A_2\mathbf{y} + D\mathbf{z}\underline{\lambda}\overline{\lambda} + (D\mathbf{y} + A_2\mathbf{z})\frac{\underline{\lambda}+\overline{\lambda}}{2} \leq \mathbf{b_2} \end{cases}. \tag{26}$$

The conditions (21) and (26) are thus sufficient for $\mathbf{y}, \mathbf{z}$ to respect constraints (15) and (16). □

The following problem can be crafted from the conditions of Theorem 3:

$$\begin{aligned} \mathcal{P}^{AR}_{\underline{\lambda},\overline{\lambda}}(\lambda) = \ & \underset{\mathbf{y},\mathbf{z}}{\text{minimize}} \quad \mathbf{c}^t(\mathbf{y} + \lambda\mathbf{z}) \\ & \text{subject to} \quad A_1\mathbf{y} + \underline{\lambda}A_1\mathbf{z} \leq \mathbf{b_1} \\ & \qquad\qquad A_1\mathbf{y} + \overline{\lambda}A_1\mathbf{z} \leq \mathbf{b_1} \\ & \qquad\qquad (A_2 + \underline{\lambda}D)(\mathbf{y} + \underline{\lambda}\mathbf{z}) \leq \mathbf{b_2} \\ & \qquad\qquad (A_2 + \overline{\lambda}D)(\mathbf{y} + \overline{\lambda}\mathbf{z}) \leq \mathbf{b_2} \\ & \qquad\qquad A_2\mathbf{y} + D\mathbf{z}\underline{\lambda}\overline{\lambda} + (D\mathbf{y} + A_2\mathbf{z})\frac{\underline{\lambda}+\overline{\lambda}}{2} \leq \mathbf{b_2}. \end{aligned} \tag{27}$$

where $h^{AR}_{\underline{\lambda},\overline{\lambda}}(\lambda)$ denotes the objective function for a given value λ. This optimisation problem provides, for each value of λ, the robust affine solution that gives the best variable bound on the objective function at the given point λ. By Theorem 3, $h^{AR}_{\underline{\lambda},\overline{\lambda}}(\lambda) \geq f(\lambda)\ \forall \lambda \in [\underline{\lambda}, \overline{\lambda}]$.

Ways to select $\mathbf{y}$ *and* $\mathbf{z}$ There can be a large number of $\mathbf{y}$ and $\mathbf{z}$ that respect the condition in Theorem 3. We added many degrees of freedom in this relaxation. For instance, any constant robust solution $\mathbf{x}$ can be mapped to a solution of the new space from Theorem 3 with $\mathbf{y} = \mathbf{x}$ and $\mathbf{z} = \mathbf{0}$, but many more may exist with $\mathbf{z} \neq 0$ and with different values of $\mathbf{y}$. Most of these solutions will

provide upper bounds that are actually not very tight. It is thus important to select *suitable* $\mathbf{y}$ and $\mathbf{z}$. We propose four ways to select $\mathbf{y}$ and $\mathbf{z}$. Two of these ways are based on the sequential optimisation of two objectives. The third and fourth introduce an additional constraint.

- The first method (called the *left* linear robust solution or *robust line left* in the experiments) requires two optimisations. It first optimises $\mathcal{P}^{AR}_{\underline{\lambda},\overline{\lambda}}(\underline{\lambda})$, computing $h^{AR}_{\underline{\lambda},\overline{\lambda}}(\underline{\lambda})$. It then solves $\mathcal{P}^{AR}_{\underline{\lambda},\overline{\lambda}}(\overline{\lambda})$ with the additional constraint $\mathbf{c}^t(\mathbf{y}+\underline{\lambda}\mathbf{z}) = h^{AR}_{\underline{\lambda},\overline{\lambda}}(\underline{\lambda})$. The bound thus gives the linear robust solution that is the tightest around $\underline{\lambda}$ and among those, the one that has the lowest possible slope.
- Similarly, the second method functions in the same manner but in reverse order, and is called the *right* linear robust solution or *robust line right* in the experiments. It first optimises $\mathcal{P}^{AR}_{\underline{\lambda},\overline{\lambda}}(\overline{\lambda})$, computing $h^{AR}_{\underline{\lambda},\overline{\lambda}}(\overline{\lambda})$, then solves $\mathcal{P}^{AR}_{\underline{\lambda},\overline{\lambda}}(\underline{\lambda})$ with the additional constraint $\mathbf{c}^t(\mathbf{y}+\overline{\lambda}\mathbf{z}) = h^{AR}_{\underline{\lambda},\overline{\lambda}}(\overline{\lambda})$.
- The third method adds the constraint $\mathbf{c}^t\mathbf{z} = \delta$ to the problem $\mathcal{P}^{AR}_{\underline{\lambda},\overline{\lambda}}(\underline{\lambda})$ to force the variable slope to be δ and optimises the objective $\mathbf{c}^t\mathbf{y}$ (as $\mathbf{c}^t\mathbf{z}$ is now constant). In the experiments, we fix the slopes to two values. The first one is $\delta = \frac{f(\overline{\lambda})-f(\underline{\lambda})}{\overline{\lambda}-\underline{\lambda}}$, called robust fixed slope pairwise. The second is $\delta = 0$, called robust yzflat. However, any δ can be chosen. Note that in the case $\delta = 0$, the solution $\mathbf{y}+\lambda\mathbf{z}$ is not necessarily "flat" in the solution space (i.e. $\mathbf{z}$ may not be $\mathbf{0}$) as illustrated in the Example 4.

In the next section, we discuss another method, more sophisticated, to find good values of $\mathbf{y}$ and $\mathbf{z}$.

Example 4 The problem presented in eq. (13) has no constant robust solution over the range $[-2, 2]$ (see Example 3). It has, however, variable robust solutions (even a flat one), as shown in Figure 3.

3.4 Concave envelope of robust linear solutions

In the previous section, we proved that we can find a robust variable solution with several possible slopes, e.g. the bound robust fixed slope pairwise. In this section, we introduce a bound tightness criterion to find the interval $[\underline{\lambda_1}, \overline{\lambda_1}]$ upon which a given optimal (for a certain value of λ_1) robust variable solution $\mathbf{y}+\lambda\mathbf{z}$ remains optimal over the interval, i.e. on that particular interval, we cannot find another robust variable solution whose objective function improves. From that criterion, coupled with warm starting, we describe a corresponding bound and discuss its strengths and weaknesses.

Looking at the problem $\mathcal{P}^{AR}_{\underline{\lambda},\overline{\lambda}}(\lambda)$ defined in (27), an interesting observation is that λ is now only present in the objective function; we have reduced a generic problem where modifications are applied to the constraints' coefficients to a simpler problem where the modifications are only located on the objective

Fig. 3: Bounds obtained with the different robust variable solution for the problem (13).

function coefficients. In principle, we would have to solve this optimisation problem for every value of λ. In the following, we show how to do it without reoptimising for an infinite number of values of λ.

However, techniques based on reduced costs can be used to find the interval $[\underline{\lambda_1}, \overline{\lambda_1}]$ where a given robust linear solution optimised for λ_1 is the tightest variable bound that can be found. In order to do this, and to simplify part of the proofs below, we rewrite the problem $\mathcal{P}^{AR}_{\underline{\lambda},\overline{\lambda}}(\lambda)$ by creating explicit slack variables and putting all the constraints into a single matrix M and the right-hand side in a vector $\mathbf{E}$:

$$\begin{aligned} \mathcal{P}^{AR}_{\underline{\lambda},\overline{\lambda}}(\lambda) \equiv \underset{\mathbf{y},\mathbf{z}}{\text{minimize}} \quad & \begin{pmatrix} \mathbf{c}^t & \lambda\mathbf{c}^t & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{y} \\ \mathbf{z} \\ \mathbf{s} \end{pmatrix} \\ \text{subject to} \quad & M \begin{pmatrix} \mathbf{y} \\ \mathbf{z} \\ \mathbf{s} \end{pmatrix} = \mathbf{E}, \\ & \mathbf{s} \geq \mathbf{0}. \end{aligned} \tag{28}$$

We use this formulation to derive the associated theorem.

Theorem 4 (Robust bound tightness criterion) *Let $\langle \mathbf{y}^*, \mathbf{z}^*, \mathbf{s}^* \rangle$ be an optimal basic solution of the problem $\mathcal{P}^{AR}_{\underline{\lambda},\overline{\lambda}}(\lambda)$ and let B and N be the basic and non-basic sets of variables associated to the solution. Additionally, let $\mathbf{r}$ be the vector of reduced costs.*

Then, $h^{AR}_{\underline{\lambda},\overline{\lambda}}(\lambda)$ equals to $\mathbf{c}^t(\mathbf{y}^* + \lambda\mathbf{z}^*)$ for any λ respecting the following conditions simultaneously:

$$(\lambda - \lambda_1)\left(\begin{pmatrix}\mathbf{0}\ \mathbf{c}_{Nz}^T\ \mathbf{0}\end{pmatrix} - \begin{pmatrix}\mathbf{0}\ \mathbf{c}_{Bz}^T\ \mathbf{0}\end{pmatrix} M_B^{-1} M_N\right) \geq -\mathbf{r} \tag{29}$$

where $\mathbf{c}_{B_z}^T$ *(resp.* $\mathbf{c}_{N_z}^T$*) is the vector* $\mathbf{c}^T$ *restricted to the variables in* B_z *(resp.* N_z*), itself the subset of* $\mathbf{z}$ *variables in* B *(resp.* N*).*

Proof The reduced costs for the basis related to $\langle \mathbf{y}^*, \mathbf{z}^*, \mathbf{s}^* \rangle$ on the problem $\mathcal{P}^{AR}_{\underline{\lambda},\overline{\lambda}}(\lambda_1)$ are

$$\mathbf{r} := \begin{pmatrix}\mathbf{c}_{N_y}^T\ \lambda_1 \mathbf{c}_{N_z}^T\ \mathbf{0}\end{pmatrix} - \begin{pmatrix}\mathbf{c}_{B_y}^T\ \lambda_1 \mathbf{c}_{B_z}^T\ \mathbf{0}\end{pmatrix} M_B^{-1} M_N \tag{30}$$

with $\mathbf{r} \geq \mathbf{0}$, as the solution is optimal. If we now consider the same basis on the sightly modified problem $\mathcal{P}^{AR}_{\underline{\lambda},\overline{\lambda}}(\lambda_1 + \delta)$, we obtain that the new reduced costs are:

$$\begin{pmatrix}\mathbf{c}_{Ny}^T\ (\lambda_1 + \delta)\mathbf{c}_{Nz}^T\ \mathbf{0}\end{pmatrix} - \begin{pmatrix}\mathbf{c}_{By}^T\ (\lambda_1 + \delta)\mathbf{c}_{Bz}^T\ \mathbf{0}\end{pmatrix} M_B^{-1} M_N \tag{31}$$

$$= \mathbf{r} + \begin{pmatrix}\mathbf{0}\ \delta\mathbf{c}_{Nz}^T\ \mathbf{0}\end{pmatrix} - \begin{pmatrix}\mathbf{0}\ \delta\mathbf{c}_{Bz}^T\ \mathbf{0}\end{pmatrix} M_B^{-1} M_N \tag{32}$$

For the basis to stay optimal for the given δ, these reduced costs must be non-negative. Therefore, we have that

$$\begin{pmatrix}\mathbf{0}\ \delta\mathbf{c}_{Nz}^T\ \mathbf{0}\end{pmatrix} - \begin{pmatrix}\mathbf{0}\ \delta\mathbf{c}_{Bz}^T\ \mathbf{0}\end{pmatrix} M_B^{-1} M_N \geq -\mathbf{r}$$

If we pose $\lambda = \lambda_1 + \delta$, we obtain the conditions stated initially. □

Derived bounding method

Theorem 4 allows us to find the interval $[\underline{\lambda_1}, \overline{\lambda_1}]$ for which a given robust linear (basic) solution remains optimal, i.e. where another affine solution $\mathbf{c}^t(\mathbf{y} + \lambda\mathbf{z})$ whose value is lower than the one represented by the basic solution does not exist. Once the basis solution is known, computing the range of λs for which it stays optimal can be done in $\mathcal{O}(m^2 + m(n-m))$ where n is the number of variables in the (full) problem and m the number of constraints. Most of the operations are indeed vector-matrix multiplications and rely on standard warm starting techniques.

This procedure paves the way to compute a so-called "envelope" of robust linear solutions, i.e. an envelope containing the best combination of lower and upper linear robust bounds, in a fast way using simplex warm starting. For minimization problems, the upper bounds give together a concave envelope, while the lower bounds give a convex one. Their combination form a convex space.

Given an interval $[\underline{\lambda}, \overline{\lambda}]$, the steps to compute this envelope are given as follows:

- First, we solve $\mathcal{P}^{AR}_{\underline{\lambda},\overline{\lambda}}(\underline{\lambda})$ and retrieve an optimal basic solution $\mathbf{x}^*_{\mathcal{P}^{AR}_{\underline{\lambda},\overline{\lambda}}(\underline{\lambda})}$ at $\lambda = \underline{\lambda}$ with its associated basis.

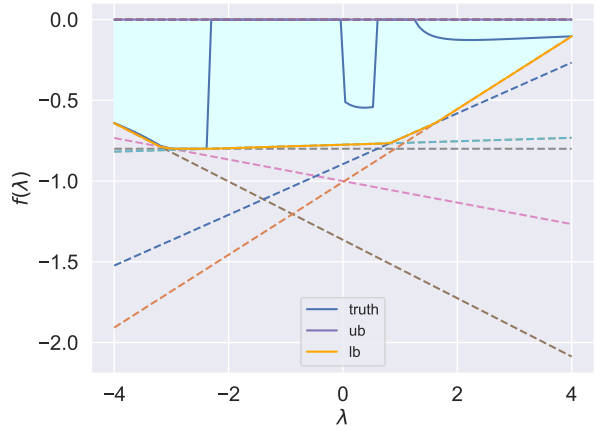

Fig. 4: Illustration of the robust envelope algorithm on problem $\mathcal{P}_{\text{toy 1}}$ (see Annex A.2)

- Second, we compute the biggest value of δ for which the current solution is still optimal, i.e. by using Theorem 4 we take the biggest δ such that the reduced costs of $\mathcal{P}_{h_{\underline{\lambda},\overline{\lambda}}}(\underline{\lambda}+\delta)$ are non-negative.
- Third, we use warm starting methods to reoptimise the problem $\mathcal{P}^{AR}_{\underline{\lambda},\overline{\lambda}}(\underline{\lambda}+\delta+\epsilon)$. We get a new optimal basis and new optimal solution.
- Fourth, we repeat the three previous steps until $\overline{\lambda}$ is reached.
- The algorithm finishes and produces a finite number of solutions (as the polytope of the problem has a finite number of vertices), which all taken together form a concave upper bound for the range.

In Figure 4, we illustrate the algorithm on both upper and lower bounds. For the lower bound, the algorithm first finds the linear bound in purple that minimizes the problem for $\lambda = -4$ and its equivalent basis. Then iteratively, it updates the next basis, found by using the optimality criterion, and reoptimises for the several values of λ obtaining the bound: pink, grey, yellow and finally light blue in that order. For each reoptimisation, the previous basis is used, upon it, a few simplex iterations are performed to find the new optimal objective and basis.

The main drawback of the method resides in the need for an optimal basis whereas the other methods only need an optimal solution. Similarly to the robust variable solution, this method reformulates the problem using $2n$ variables and $2m_1 + 3m_2$ constraints. Finding an optimal basis would require the use of either the simplex or the crossover. This operation may take a lot of time. However, if an optimal basis is known, the method is fast and accurate. In the experiments this bound is named the robust envelope.

3.5 Constant lower bounds using Lagrangian relaxations

In this section and next section, we propose two lower bounds based on Lagrangian relaxations of (3). Let us define the problem $\mathcal{P}^L(\boldsymbol{\rho}, \lambda)$ as the Lagrangian relaxation of $\mathcal{P}(\lambda)$, obtained by relaxing the second set of constraints:

$$\begin{aligned} \mathcal{P}^L(\boldsymbol{\rho}, \lambda) \equiv \underset{x}{\text{minimize}} \quad & \mathbf{c}^t\mathbf{x} - \boldsymbol{\rho}((A_2 + \lambda D)\mathbf{x} - \mathbf{b_2}) \\ \text{subject to} \quad & A_1\mathbf{x} \leq \mathbf{b_1}. \end{aligned} \tag{33}$$

The optimal objective of $\mathcal{P}^L(\boldsymbol{\rho}, \lambda)$ is denoted $h^L(\boldsymbol{\rho}, \lambda)$. Then, the following result holds:

Theorem 5 *A lower bound to* (3) *is given by,*

$$\max\left(\min\left(f(\underline{\lambda}), h^L(\boldsymbol{\rho}^*_{\underline{\boldsymbol{\lambda}}}, \overline{\lambda})\right), \min\left(f(\overline{\lambda}), h^L(\boldsymbol{\rho}^*_{\overline{\boldsymbol{\lambda}}}, \underline{\lambda})\right)\right) \leq \min_{\lambda \in [\underline{\lambda}, \overline{\lambda}]} f(\lambda). \tag{34}$$

where $\boldsymbol{\rho}^*_{\underline{\boldsymbol{\lambda}}}$ *and* $\boldsymbol{\rho}^*_{\overline{\boldsymbol{\lambda}}}$ *are the optimal dual variables related to the* λ *constraints of* $\mathcal{P}(\underline{\lambda})$ *and* $\mathcal{P}(\overline{\lambda})$ *respectively.*

Proof If $\boldsymbol{\rho}^*_{\underline{\boldsymbol{\lambda}}}$ and $\boldsymbol{\rho}^*_{\overline{\boldsymbol{\lambda}}}$ are the optimal dual variables of the second set of constraints for $\mathcal{P}(\underline{\lambda})$ and $\mathcal{P}(\overline{\lambda})$ respectively, then

$$h^L(\boldsymbol{\rho}^*_{\underline{\boldsymbol{\lambda}}}, \underline{\lambda}) = f(\underline{\lambda}) \tag{35}$$

$$h^L(\boldsymbol{\rho}^*_{\overline{\boldsymbol{\lambda}}}, \overline{\lambda}) = f(\overline{\lambda}) \tag{36}$$

Over the interval $[\underline{\lambda}, \overline{\lambda}]$, $\min_{\lambda \in [\underline{\lambda}, \overline{\lambda}]} h^L(\boldsymbol{\rho}, \lambda)$ provides a valid lower bound. If we extend the equations:

$$\begin{aligned} \min_{\lambda} h^L(\boldsymbol{\rho}, \lambda) = \underset{\lambda, \mathbf{x}}{\text{minimize}} \quad & \mathbf{c}^t\mathbf{x} - \boldsymbol{\rho}((A_2 + \lambda D)\mathbf{x} - \mathbf{b_2}) \\ \text{subject to} \quad & A_1\mathbf{x} \leq \mathbf{b_1} \end{aligned} \tag{37}$$

the objective function can be rewritten as $\mathbf{c}^t\mathbf{x} - \boldsymbol{\rho}A_2\mathbf{x} - \lambda\boldsymbol{\rho}D\mathbf{x} + \boldsymbol{\rho}\mathbf{b_2}$. As λ evolves linearly in a given range, the minimum is either $\lambda = \underline{\lambda}$ or $\lambda = \overline{\lambda}$, i.e.:

$$\min_{\lambda} h^L(\boldsymbol{\rho}, \lambda) = \min(h^L(\boldsymbol{\rho}, \underline{\lambda}), h^L(\boldsymbol{\rho}, \overline{\lambda})) \tag{38}$$

and this for any $\boldsymbol{\rho}$. We chose to use the dual variables $\boldsymbol{\rho}^*_{\underline{\boldsymbol{\lambda}}}$ and $\boldsymbol{\rho}^*_{\overline{\boldsymbol{\lambda}}}$ related to the λ dependent constraints of $\mathcal{P}(\underline{\lambda})$ and $\mathcal{P}(\overline{\lambda})$, which are good candidates as it is expected that the optimal dual variables do not change too much around small modifications, we obtain:

$$\min_{\lambda} h^L(\boldsymbol{\rho}^*_{\underline{\boldsymbol{\lambda}}}, \lambda) = \min\left(h^L(\boldsymbol{\rho}^*_{\underline{\boldsymbol{\lambda}}}, \underline{\lambda}), h^L(\boldsymbol{\rho}^*_{\underline{\boldsymbol{\lambda}}}, \overline{\lambda})\right) = \min\left(f(\underline{\lambda}), h^L(\boldsymbol{\rho}^*_{\underline{\boldsymbol{\lambda}}}, \overline{\lambda})\right) \tag{39}$$

and

$$\min_{\lambda} h^L(\boldsymbol{\rho}^*_{\overline{\boldsymbol{\lambda}}}, \lambda) = \min(f(\overline{\lambda}), h^L(\boldsymbol{\rho}^*_{\overline{\boldsymbol{\lambda}}}, \underline{\lambda})). \tag{40}$$

Given two known points $f(\underline{\lambda})$ and $f(\overline{\lambda})$, we can thus compute two lower bounds at the expense of running two linear programs with a smaller number of constraints ($\mathcal{P}^L(\boldsymbol{\rho}^*_{\overline{\boldsymbol{\lambda}}}, \underline{\lambda})$ and $\mathcal{P}^L(\boldsymbol{\rho}^*_{\underline{\boldsymbol{\lambda}}}, \overline{\lambda})$). Both are lower bounds for the whole range $\lambda \in [\underline{\lambda}, \overline{\lambda}]$. Thus their maximum is also a lower bound:

$$\max\left(\min\left(f(\underline{\lambda}), h^L(\boldsymbol{\rho}^*_{\underline{\boldsymbol{\lambda}}}, \overline{\lambda})\right), \min\left(f(\overline{\lambda}), h^L(\boldsymbol{\rho}^*_{\overline{\boldsymbol{\lambda}}}, \underline{\lambda})\right)\right) \leq \min_{\lambda \in [\underline{\lambda}, \overline{\lambda}]} f(\lambda) \tag{41}$$

□

It should be noted that as $h^L(\boldsymbol{\rho}, \lambda)$ is a lower bound for $f(\lambda)$: $h^L(\boldsymbol{\rho}, \lambda) \leq f(\lambda)\ \forall \boldsymbol{\rho}, \lambda$. A solution of the underlying problem of $h^L(\boldsymbol{\rho}, \lambda)$ is not necessarily a feasible solution of the original problem (in $f(\lambda)$) and can in some cases lead to an unbounded problem. This bound reduces the size of the problem to m_1 constraints and does not affect the number of variables. In the experiments, this bound is referred to as Lagrangian flat.

3.6 Non-constant lower bounds using Lagrangian relaxations

We can extend the previous technique to any function of $\boldsymbol{\rho}$ depending on λ:

$$\begin{aligned} \mathcal{P}^L(\boldsymbol{\rho}(\lambda), \lambda) \equiv \ & \underset{\mathbf{x}}{\text{minimize}} \quad \mathbf{c}^t\mathbf{x} - \boldsymbol{\rho}(\lambda)\left((A_2 + \lambda D)\,\mathbf{x} - \mathbf{b_2}\right) \\ & \text{subject to} \quad A_1\mathbf{x} \leq \mathbf{b_1} \end{aligned} \tag{42}$$

the optimal value of $\mathcal{P}^L(\boldsymbol{\rho}(\lambda), \lambda)$, denoted $h^L(\boldsymbol{\rho}(\lambda), \lambda)$, always provides valid lower bound, whatever the choice of $\boldsymbol{\rho}(\lambda)$. For most non-trivial choices of $\boldsymbol{\rho}(\lambda)$, we however lose the linearity of the objective w.r.t. $\mathbf{x}$ when computing the constant bound.

Example 5 By choosing $\boldsymbol{\rho}(\lambda)$ as a linear function of λ, i.e. $\boldsymbol{\rho}(\lambda) = \boldsymbol{\rho}_a\lambda + \boldsymbol{\rho}_b$, we obtain a quadratic objective

$$\begin{aligned} \mathbf{c}^t\mathbf{x} - (\boldsymbol{\rho}_a\lambda + \boldsymbol{\rho}_b)((A_2 + \lambda D)\mathbf{x} - \mathbf{b_2}) = \quad & -\lambda^2(\boldsymbol{\rho}_a D\mathbf{x}) \\ & -\lambda(\boldsymbol{\rho}_a(A_2\mathbf{x} - \mathbf{b_2}) + \boldsymbol{\rho}_b D\mathbf{x}) \\ & +\mathbf{c}^t\mathbf{x} - \boldsymbol{\rho}_b A_2\mathbf{x} + \boldsymbol{\rho}_b\mathbf{b_2}. \end{aligned}$$

For a given x, its minimum is either at $\lambda = \overline{\lambda}, \lambda = \underline{\lambda}$ or $\lambda = \frac{\boldsymbol{\rho}_a(A_2\mathbf{x} - \mathbf{b_2}) + \boldsymbol{\rho}_b D\mathbf{x}}{2\boldsymbol{\rho}_a D\mathbf{x}}$, the last being non-linear in $\mathbf{x}$.

The non-linearities make the computation of more complex constant bounds difficult. It is however possible to further relax the problem, which we demonstrate by focusing on the case where $\boldsymbol{\rho}(\lambda)$ is a polynomial function of λ.

Theorem 6 *Let* $\boldsymbol{\rho}(\lambda) = \boldsymbol{\rho}_n\lambda^n + \boldsymbol{\rho}_{n-1}\lambda^{n-1} + \ldots + \boldsymbol{\rho}_0$. *Then,*

$$\begin{aligned}
&\lambda^{n+1}\left(\min_{\mathbf{x}|A_1\mathbf{x}\leq\mathbf{b_1}} -\boldsymbol{\rho}_n D\mathbf{x}\right)\\
&+\sum_{i=1}^{n}\lambda^i\left(\min_{\mathbf{x}|A_1\mathbf{x}\leq\mathbf{b_1}} -\boldsymbol{\rho}_{i-1}D\mathbf{x} - \boldsymbol{\rho}_i(A_2\mathbf{x}-\mathbf{b_2})\right)\\
&+\left(\min_{\mathbf{x}|A_1\mathbf{x}\leq\mathbf{b_1}} -\boldsymbol{\rho}_0(A_2\mathbf{x}-\mathbf{b_2}) + \mathbf{c}^t\mathbf{x}\right)
\end{aligned} \tag{43}$$

is a lower bound of $f(\lambda)$ $\forall\lambda \geq 0$.

Proof With this $\boldsymbol{\rho}(\lambda)$, $h^L(\boldsymbol{\rho}(\lambda),\lambda)$ becomes

$$\begin{aligned}
\underset{x}{\text{minimize}} \quad & -\lambda^{n+1}\boldsymbol{\rho}_n D\mathbf{x} - \lambda^n(\boldsymbol{\rho}_{n-1}D\mathbf{x} + \boldsymbol{\rho}_n(A_2\mathbf{x}-\mathbf{b_2}))\\
& -\lambda^{n-1}(\boldsymbol{\rho}_{n-2}D\mathbf{x} + \boldsymbol{\rho}_{n-1}(A_2\mathbf{x}-\mathbf{b_2}))\\
& -\ldots\\
& -\lambda(\boldsymbol{\rho}_0 D\mathbf{x} + \boldsymbol{\rho}_1(A_2\mathbf{x}-\mathbf{b_2}))\\
& -\boldsymbol{\rho}_0(A_2\mathbf{x}-\mathbf{b_2}) + \mathbf{c}^t\mathbf{x}\\
\text{subject to} \quad & A_1\mathbf{x} \leq b_1.
\end{aligned} \tag{44}$$

Using the fact that $\min_\lambda f_1(\lambda)+f_2(\lambda) \geq \min_\lambda f_1(\lambda)+\min_\lambda f_2(\lambda)$, we obtain that

$$\begin{aligned}
h^L(\boldsymbol{\rho}(\lambda),\lambda) \geq & \min_{\mathbf{x}|A_1\mathbf{x}\leq b_1} -\lambda^{n+1}\boldsymbol{\rho}_n D\mathbf{x}\\
&+\sum_{i=1}^{n}\left(\min_{\mathbf{x}|A_1\mathbf{x}\leq b_1} -\lambda^i(\boldsymbol{\rho}_{i-1}D\mathbf{x} + \boldsymbol{\rho}_i(A_2\mathbf{x}-\mathbf{b_2}))\right)\\
&+\min_{\mathbf{x}|A_1\mathbf{x}\leq b_1} -\boldsymbol{\rho}_0(A_2\mathbf{x}-\mathbf{b_2}) + \mathbf{c}^t\mathbf{x}.
\end{aligned} \tag{45}$$

As λ is unaffected by the minimization (by hypothesis, $\lambda \geq 0$), it can be moved out of the objective; by doing this we obtain the equation shown in the theorem.

The resulting function is a lower bound as $h^L(\boldsymbol{\rho}(\lambda),\lambda)$ is itself a lower bound on $f(\lambda)$. □

Obviously, another version of the theorem exists for $\lambda \leq 0$, but special care need to be taken for cases where $\lambda^i \leq 0$, as $\min_x \lambda^i g(x) = \lambda^i \max_x g(x)$, for any function $g(x)$ and $\lambda^i \leq 0$. Thus, for a negative λ, all minimizations related to odd exponents of λ need to be transformed to maximizations.

There are four interesting observations to make here. The first is that the bound is not constant and varies in λ, in a polynomial fashion. The second is that the bound is valid for any λ (at least, as soon as all the linear problems have bounded solutions): they are not constrained to a particular range unlike previously discussed bounds. The third is that all these problems are only constrained by $A_1\mathbf{x} \leq \mathbf{b_1}$, which may lead to unbounded solutions. Finally,

since these problems all share the same constraints and variables and differ only by their objective function, warm-start methods can be easily used.

We propose to select two instances of this bound (here shown for $\lambda > 0$):

- A linear instance of the bound where $\boldsymbol{\rho}(\lambda) = \boldsymbol{\rho}^*$, where $\boldsymbol{\rho}^*$ is the dual variable associated to the second set of constraints of $f(\lambda^*)$, for a given λ^*. This gives the following bound, which we call the *linear Lagrangian bound* or Lagrangian line:

$$\lambda \left(\min_{\mathbf{x}|A_1\mathbf{x}\leq b_1} -\boldsymbol{\rho}^* D\mathbf{x} \right) + \left(\min_{\mathbf{x}|A_1\mathbf{x}\leq b_1} -\boldsymbol{\rho}^*(A_2\mathbf{x} - \mathbf{b_2}) + \mathbf{c}^t\mathbf{x} \right). \tag{46}$$

- A second idea is to use the linear interpolation between the known dual variables of $f(\underline{\lambda})$ and $f(\overline{\lambda})$ related to the A_2 matrix, $\underline{\boldsymbol{\rho}}$ and $\overline{\boldsymbol{\rho}}$, i.e. with $\boldsymbol{\rho} = \frac{\overline{\boldsymbol{\rho}} - \underline{\boldsymbol{\rho}}}{\overline{\lambda} - \underline{\lambda}}(\lambda - \underline{\lambda}) + \underline{\boldsymbol{\rho}}$. This produces a quadratic lower bound (that we call the *quadratic Lagrangian bound* $lb^{ql}_{\underline{\lambda},\overline{\lambda}}(\lambda)$ or Lagrangian quadratic):

$$\begin{aligned} &\lambda^2 \left(\min_{\mathbf{x}|A_1\mathbf{x}\leq b_1} \frac{\overline{\boldsymbol{\rho}} - \underline{\boldsymbol{\rho}}}{\overline{\lambda} - \underline{\lambda}} D\mathbf{x} \right) \\ &+ \lambda \left(\min_{\mathbf{x}|A_1\mathbf{x}\leq b_1} \frac{\underline{\boldsymbol{\rho}}\overline{\lambda} - \overline{\boldsymbol{\rho}}\underline{\lambda}}{\overline{\lambda} - \underline{\lambda}} D\mathbf{x} + \frac{\overline{\boldsymbol{\rho}} - \underline{\boldsymbol{\rho}}}{\overline{\lambda} - \underline{\lambda}}(A_2\mathbf{x} - \mathbf{b_2}) \right) \\ &+ \left(\min_{\mathbf{x}|A_1\mathbf{x}\leq b_1} \frac{\underline{\boldsymbol{\rho}}\overline{\lambda} - \overline{\boldsymbol{\rho}}\underline{\lambda}}{\overline{\lambda} - \underline{\lambda}}(A_2\mathbf{x} - \mathbf{b_2}) + \mathbf{c}^t\mathbf{x} \right) \end{aligned} \tag{47}$$

These two bounds respectively require two and three additional minimization operations per bound, on the same number of variables than the original problem but with only n_1 constraints and produces bounds that are valid for any λ.

3.7 Lagrangian envelope

By using the principle of Lagrangian relaxations, we move the constraints depending on λ to the objective function. Therefore, we can apply a similar methodology to the one developed in Section 3.4 and build an equivalent Lagrangian convex envelope. First, we derive a bound tightness criterion, and then describe the corresponding bound.

Bound tightness criterion

We recall the problem $\mathcal{P}^L(\boldsymbol{\rho}, \lambda)$,

$$\begin{aligned} \mathcal{P}^L(\boldsymbol{\rho}, \lambda) \equiv \ & \underset{x}{\text{minimize}} \quad \mathbf{c}^t\mathbf{x} - \boldsymbol{\rho}((A_2 + \lambda D)\mathbf{x} - \mathbf{b_2}) \\ & \text{subject to} \quad A_1\mathbf{x} \leq \mathbf{b_1}. \end{aligned} \tag{48}$$

For any $\boldsymbol{\rho} \geq \mathbf{0}$, $\mathcal{P}^L(\boldsymbol{\rho}, \lambda)$ is a relaxed version of $\mathcal{P}(\lambda)$ and its optimal objective, denoted $h^L(\boldsymbol{\rho}, \lambda)$, is a lower bound to $f(\lambda)$.

For a given positive vector $\boldsymbol{\rho}$, we add the slack variables and proceed similarly to Section 3.4. We rewrite the original problems as,

$$\begin{aligned} h^L(\boldsymbol{\rho}, \lambda) = \underset{\mathbf{x}}{\text{minimize}} \quad & \left(\mathbf{c}^t - \boldsymbol{\rho}(A_2 + \lambda D)\right) \begin{pmatrix} \mathbf{x} \\ \mathbf{s} \end{pmatrix} \\ \text{subject to} \quad & M \begin{pmatrix} \mathbf{x} \\ \mathbf{s} \end{pmatrix} = \mathbf{b_1}, \\ & \mathbf{s} \geq \mathbf{0}. \end{aligned} \tag{49}$$

where $M = \begin{pmatrix} A_1 & \mathbb{I} \end{pmatrix}$ and $\mathbb{I}$ is the identity matrix. We use this formulation to derive the associated theorem

Theorem 7 (Lagrangian bound tightness criterion) *Let $\mathbf{x}^*$ be an optimal basic solution to the problem $\mathcal{P}^L(\boldsymbol{\rho}, \lambda)$ where $\boldsymbol{\rho}$ is positive and B and N the basic and non-basic sets of variables. Additionally, let $\mathbf{r}$ be the optimal reduced costs. Then, $h^L(\boldsymbol{\rho}, \lambda)$ equals to $\mathbf{c}^t - \boldsymbol{\rho}(A_2 + \lambda D)\mathbf{x}^*$ for any λ respecting the following conditions simultaneously:*

$$(\lambda - \lambda_1) \left(-\boldsymbol{\rho} D\right)_N + \left(\boldsymbol{\rho} D\right)_B M_B^{-1} M_N \geq -\mathbf{r}$$

where $\left(\boldsymbol{\rho} D\right)_B$ (resp. $\left(\boldsymbol{\rho} D\right)_N$) is the vector $\boldsymbol{\rho} D$ restricted to the variables in B (resp. N).

Proof The optimal reduced costs of (49) are given by,

$$\mathbf{r} = \left(\mathbf{c}^t - \boldsymbol{\rho}(A_2 + \lambda D)\right)_N - \left(\mathbf{c}^t - \boldsymbol{\rho}(A_2 + \lambda D)\right)_B M_B^{-1} M_N$$

with $\mathbf{r} \geq 0$. If we consider the same basis on the slightly modified problem $\lambda + \delta$, we obtain the new reduced costs :

$$\begin{aligned} & \left(\mathbf{c}^t - \boldsymbol{\rho}(A_2 + \lambda D + \delta D)\right)_N - \left(\mathbf{c}^t - \boldsymbol{\rho}(A_2 + \lambda D + \delta D)\right)_B M_B^{-1} M_N \\ = \; & \mathbf{r} + \left(-\boldsymbol{\rho}\delta D\right)_N - \left(-\boldsymbol{\rho}\delta D\right)_B M_B^{-1} M_N \end{aligned}$$

For the basis to stay optimal for the given δ, the reduced costs must be non-negative. Therefore, we have that,

$$\left(-\boldsymbol{\rho}\delta D\right)_N + \left(\boldsymbol{\rho}\delta D\right)_B M_B^{-1} M_N \geq -\mathbf{r}$$

We pose $\lambda = \lambda_1 + \delta$, we obtain the conditions stated. □

Derived bound

The bound derived is similar to the one derived in Section 3.4. The main complexity lies in the choice of the vector $\boldsymbol{\rho}$. Selecting an appropriate value for $\boldsymbol{\rho}$ is complex. Any positive value of $\boldsymbol{\rho}$ can be used to compute the bound. We choose to optimise the problem $\mathcal{P}^L(\underline{\lambda})$ and retrieve its optimal dual values $\boldsymbol{\rho}^*$, a vector of size $m_1 + m_2$, where m_1 is the number of constraints in A_1 and m_2 number of constraints in A_2. The vector $\boldsymbol{\rho}^*$ can therefore be split as $\left(\boldsymbol{\rho}_1^* \; \boldsymbol{\rho}_2^*\right)$ where $\boldsymbol{\rho}_1^*$ are the optimal dual variables of the constraints A_1 and $\boldsymbol{\rho}_1^*$ the optimal dual variables of the constraints A_2. We know that

$$f(\underline{\lambda}) = h^L(\boldsymbol{\rho}_2^*, \underline{\lambda}).$$

Therefore we know that for $\underline{\lambda}$ the bound is equal to the optimal value, making it a good choice for $\boldsymbol{\rho}$. A similar argument can be done using $\overline{\lambda}$ instead.

4 Experiments

In this section, we discuss the different bounding methods proposed and provide several empirical studies related to them. The section is divided in three parts:

- We discuss the dataset.
- We highlight the performance of each bounding method. We uniformly split the uncertainty interval in 1, 5 and 10 subintervals. We apply each bounding method on every problem in the dataset for these ranges and report the computation time, relative precision and overall availability.
- We propose an iterative algorithm that uses both upper and lower bounds to minimize the gap between the two and benchmark its performance in term of time and accuracy.

A summary of the ten different bounds method used in the experiments is given in Table 2. The optimum reuse scheme is not used in the experiments due to its simplistic nature.

Approach	Type	Section	Name	Comment
Robust	Constant	3.2	Robust flat	
Optimization	Linear in λ	3.3	Robust line left	First optimize for $\underline{\lambda}$ then for $\overline{\lambda}$.
			Robust line right	First optimize for $\overline{\lambda}$ then for $\underline{\lambda}$.
			Robust yzflat	Add constraint $\mathbf{c^t z} = 0$.
			Robust fixed slope pairwise	$\mathbf{c^t z} = \delta$ where δ is the slope between the optimum for $\underline{\lambda}$ and $\overline{\lambda}$.
	Envelope	3.4	Robust envelope	
Lagrangian	Constant	3.5	Lagrangian flat	
Relaxations	Linear in λ	3.6	Lagrangian line	No variation of the dual
	Quadratic in λ	3.6	Lagrangian quadratic	Linear variation of the dual
	Envelope	3.7	Lagrangian envelope	

Table 2: Summary of all the bounding methods used in the experiments

4.1 Dataset

To the best of our knowledge, there does not exist a dataset for our category of problems, i.e. where there is a modification on the constraint matrix coefficients, hence, the need to build a dataset. Our proposed dataset of problems is made of three categories of problems:

- **Cat. 1 - Illustrative problems:** Four small problems with two variables and less than 15 constraints.
- **Cat. 2 - Energy problems:** Three real-life problems from the energy field, two of which are large-scale.
- **Cat. 3 - Netlib[10] problems:** 191 problems derived from 81 original Netlib library problems. For each problem in Netlib, we randomly chose 100 constraints. For each constraint, we randomly select three coefficients that we copy in the Matrix D. We multiply each coefficient in D by a number between 0.1 and 0.9 selected uniformly and change its sign uniformly. We consider $\lambda \in [-1, 1]$. We only keep the problems that are feasible for $\lambda = -1, 0, 1$ and that have a non-linear behaviour. Two other variants are made where we select only equality or inequality constraints.

The third category of problems makes sure that the bounds are computed on a large number of different problems. A full description of the illustrative and real-life problems is given in Appendix A.2 as well as the list of all the Netlib problems considered. All problems in the dataset are minimization problems and their size is shown in Figure 5. The dataset is available on Zenodo [17] and the readers are encouraged to use it and extend the dataset with more problems.

These problems have very different ranges of variation for their objective function, i.e. some objectives are very small and other very big. For all the problems in the dataset, we compute the optimal objective for 100 uniformly separated values of λ selected from $[\underline{\lambda}, \overline{\lambda}]$. We linearly normalise the objectives in such a way that the objective value for these 100 points varies between $[1, 2]$.

4.2 Bound assessment

In this experiment, we apply each bounding method to each problem presented in Section 4.1. We compare the generated bounds in terms of availability (percentage of points for which a bounding method returns a finite value), error and time taken to compute each bound in order to generate upper and lower bounds. We benchmark the performance of each bound against the *naive* solution of computing $f(\lambda)$ for 100 values of λ in the interval $[\underline{\lambda}, \overline{\lambda}]$ and against each other.

Let us first define the set of the uniformly separated points selected from the interval $[\underline{\lambda}, \overline{\lambda}]$ and its subset of points for which a bound returns a finite

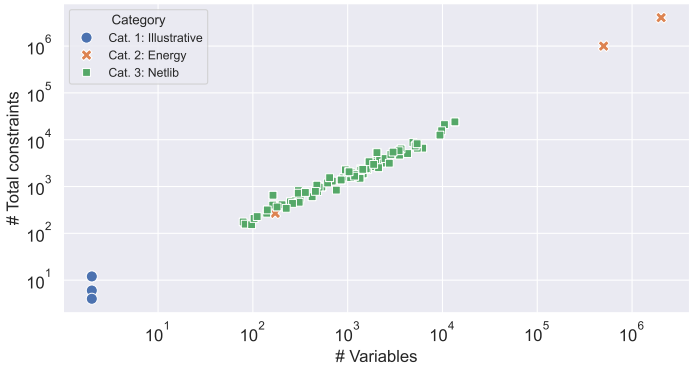

Fig. 5: Scatterplot of the number of constraints with respect of the number of variables for all the problems in the dataset by category.

value:

$$\mathcal{S} = \{\underline{\lambda} + \frac{i}{99}(\overline{\lambda} - \underline{\lambda}) \mid i \in \{0, ..., 99\}\} \tag{50}$$

$$\mathcal{A}_b = \{\lambda \in \mathcal{S} \mid \text{bound } b \text{ returns a finite value at } \lambda\} \ . \tag{51}$$

The *availability* is defined as the percentage of points in $\mathcal{S}$ for which the bound is *available*, i.e. returns a finite value.

$$\text{avail}_b = \frac{|\mathcal{A}_b|}{|\mathcal{S}|} * 100 \ . \tag{52}$$

This metric is inherently biased against methods (such as ours) that do not sample the solution space but rather provide stronger guarantees. To illustrate, consider the hypothetic case where the image of $f(\lambda)$ is only defined at the 100 values of λ that were sampled and unbounded everywhere else: bounding methods would fail to provide strong guarantees around the points and report no bounds, and the availability would be 0.

The root mean square error (RMSE) is defined as

$$\text{RMSE}_b = \sqrt{\frac{\sum_{\lambda \in \mathcal{A}_b}(b(\lambda) - f(\lambda))^2}{|\mathcal{A}_b|}} + 1 \ . \tag{53}$$

We add one to avoid computational problems related to an error of 0, linked to the use of this error in a denominator later in the experiments. We chose to only compute the error for points where the bound exists. If the bound does not exist, the error is considered to be infinite. The relative timing is defined as the time required to compute the bound divided by the time required to compute $f(\lambda)$ for 100 points in the range. We put a timeout of 90 minutes (5400 seconds). If a bound has not been found in that time, we consider it

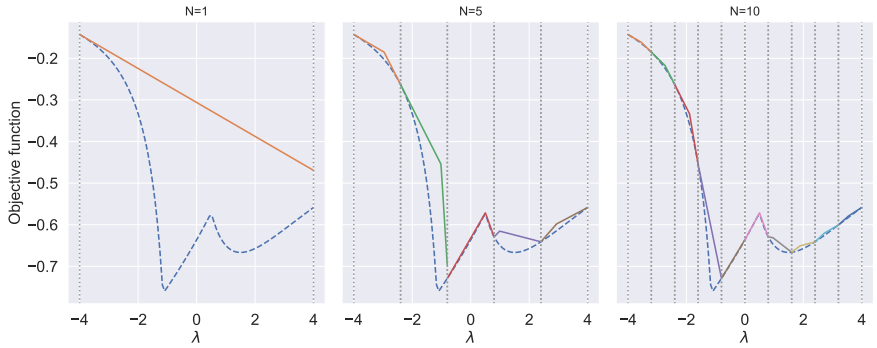

Fig. 6: Illustration of robust envelope to provide an upper bound for the problem "Toy 2" given in (65) in Annex A.2.

unavailable.

We also assess the impact of dividing the interval $[\underline{\lambda}, \overline{\lambda}]$ on the availability, the RMS error and relative timing. We split the interval uniformly in $N = 1$, 5 and 10 subintervals. In other words, for N, we split the interval $[\underline{\lambda}, \overline{\lambda}]$ in N sub-intervals of same size,

$$[\underline{\lambda} + \frac{i}{N}(\overline{\lambda} - \underline{\lambda}), \underline{\lambda} + \frac{i+1}{N}(\overline{\lambda} - \underline{\lambda})] \qquad \forall i \in \{0, ..., N-1\} \ . \tag{54}$$

We perform this for the bounding methods. We illustrate the impact of cutting the interval for $N = 1$, $N = 5$ and $N = 10$ in Figure 6 for the robust envelope bound on one particular problem.

The following analysis is divided in three subparts:

- On the macro level to answer questions related to the efficiency of this approach and highlight the overall tendencies.
- Assessing the impact of increasing the number of subintervals.
- Comparing the bounds between themselves on the three criteria.

Overall results

In terms of availability, the methods have a higher availability when providing lower bounds compared to upper bounds for every bounding method except the robust flat and Lagrangian envelope, as shown in Table 3. For the Lagrangian envelope, the difference in availability between providing an upper and lower bound is not large enough to be able to draw any conclusion. The robust methods, particularly for high number of subintervals, have a high availability on average with the robust concave envelope and robust yzflat having the highest availability with 81% for $N = 10$ to provide lower bounds. The bound with the smallest availability is the Lagrangian quadratic bound which, on average, is unavailable to provide upper bounds.

	Lower bounds			Upper bounds		
N	1	5	10	1	5	10
Lagrangian envelope	2%	2%	2%	3%	4%	4%
Lagrangian flat	11%	24%	29%	7%	15%	18%
Lagrangian line	5%	8%	8%	1%	2%	2%
Lagrangian quadratic	4%	6%	6%	0%	0%	0%
Robust concave envelope	42%	73%	81%	29%	57%	64%
Robust fixed slope pairwise	41%	73%	80%	28%	56%	63%
Robust flat	19%	32%	35%	24%	42%	46%
Robust line left	40%	67%	75%	28%	53%	62%
Robust line right	40%	69%	76%	27%	54%	61%
Robust yzflat	42%	73%	81%	29%	57%	65%

Table 3: Average availability in percentage over all the dataset for all the bounds to provide lower and upper bounds for N=1,5 and 10.

In terms of precision and timing, the methods perform better when used to provide upper bounds rather than lower bounds as shown in Table 4 and Table 5. As mentioned previously, we hypothesise that all the problems in the dataset are minimization problems and the fact that the original problem is dualised to get the lower bound for robust problems significantly increases the problem size and leads to this difference in performance. Indeed, in the dualised version of (3), all the constraints are dependent on λ. In the context of robust bounds, the size of the problem is significantly increase. For the Lagrangian bounding methods, we get an optimisation problem that has no constraints as they are moved to the objective.

In terms of relative timing, all the methods except the robust envelope (for 5 and 10 subintervals) and the robust fixed slope pairwise (when providing lower bounds with 10 subintervals) have a smaller median time than the one required to compute 100 points. In terms of relative error, for $N = 5$ and $N = 10$, all methods provide a relative error smaller than 2 for both upper and lower bounds, except the Lagrangian line (for both lower and upper bounds), and the Lagrangian quadratic and robust flat for lower bounds only. Figure 7 shows the evolution of the percentage of problems that are within an error range for a growing error for all the bounds and all the subintervals $N = 1$, 5 and 10.

Splitting the uncertainty interval

Splitting the uncertainty interval in uniform subinterval leads to an improvement in terms of availability and error. In terms of availability, increasing the number of splits increases the chances of bounds to be found for at least one of the subinterval of the range. In terms of error, there are a few exceptions for which the low initial availability induces a bias. For instance, the Lagrangian envelope has an error of 1.00 and an availability of 3% to provide upper bounds for $N = 1$. By increasing the number of subintervals to $N = 5$,

	Lower bounds			Upper bounds		
N	1	5	10	1	5	10
Lagrangian envelope	1.00	1.22	1.27	1.00	1.20	1.20
Lagrangian flat	4.79	1.67	1.28	1.78	1.15	1.07
Lagrangian line	3.99	3.12	3.74	2.40	2.25	2.41
Lagrangian quadratic	5.95	5.79	6.50	1.93	1.54	1.54
Robust concave envelope	1.89	1.16	1.07	1.36	1.02	1.01
Robust fixed slope pairwise	2.20	1.21	1.10	1.54	1.04	1.01
Robust flat	2.94	2.88	2.88	1.75	1.16	1.08
Robust line left	5.47	1.58	1.26	1.64	1.07	1.02
Robust line right	4.35	1.53	1.24	1.71	1.06	1.02
Robust yzflat	2.04	1.25	1.12	1.62	1.12	1.06

Table 4: Median error over all the dataset for all the bounds to provide lower and upper bounds for N=1,5 and 10.

	Lower bounds			Upper bounds		
N	1	5	10	1	5	10
Lagrangian envelope	0.10	0.27	0.43	0.11	0.30	0.51
Lagrangian flat	0.08	0.20	0.33	0.11	0.26	0.44
Lagrangian line	0.12	0.22	0.31	0.16	0.29	0.43
Lagrangian quadratic	0.14	0.25	0.35	0.18	0.32	0.45
Robust concave envelope	0.89	3.89	7.30	0.31	1.56	2.61
Robust fixed slope pairwise	0.17	0.58	1.06	0.15	0.51	0.93
Robust flat	0.02	0.07	0.12	0.02	0.06	0.11
Robust line left	0.10	0.46	0.91	0.10	0.49	0.91
Robust line right	0.11	0.48	0.92	0.10	0.51	0.94
Robust yzflat	0.08	0.36	0.69	0.09	0.37	0.69

Table 5: Median timing over all the dataset for all the bounds to provide lower and upper bounds for N=1,5 and 10.

the bound becomes more available (4%) which leads to an increased relative error of 1.2.

In terms of timing, increasing the number of subintervals does not lead to an N-fold increase in the timing for all the bounds. In other words, increasing the number of subintervals from 1 to 5 or to 10 does not lead to an increased relative timing of 5 or 10. For the Lagrangian bounds, the median time required to compute 5 subintervals and 10 subintervals is respectively 2.18-times and 3.41-times the one required when $N = 1$ on average. For the robust bounds, on average, the median time required to compute 5 subintervals and 10 subintervals is respectively 4.19-times and 7.73-times the one required when $N = 1$.

Comparing the bounding methods

In terms of availability, the bound with the highest availability for lower bounds and upper bounds respectively is robust yzflat with robust fixed slope

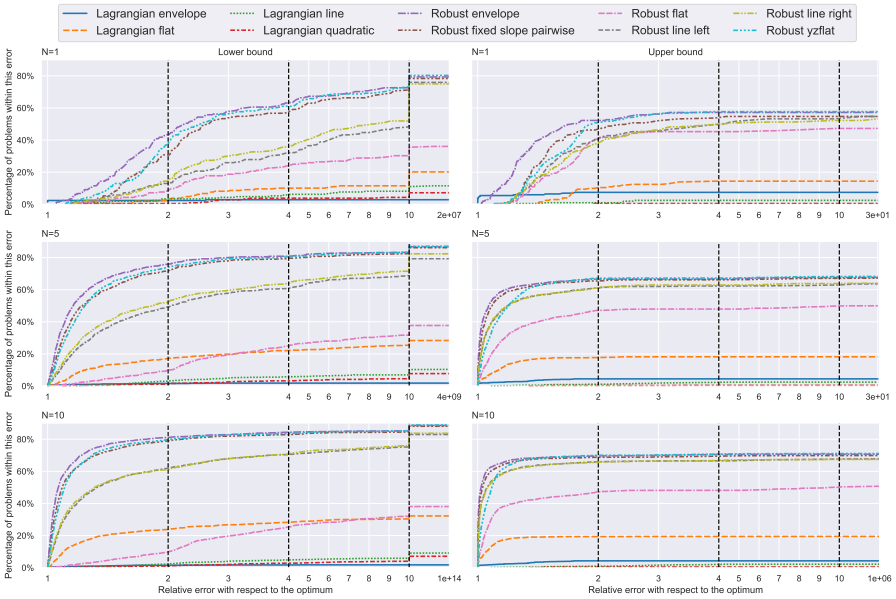

Fig. 7: Percentage of problems that are within an error range for an increasing error for every bound and value of $N = 1$, 5 and 10.

pairwise and robust concave envelope being close seconds. The robust concave envelope suffers from the timeout as it requires more computation.

The robust bounds, with the exception robust flat, have approximately the same availability and the maximum difference between the bound with the highest availability and the one with the lowest is 6%. However, robust flat always has a much smaller availability, approximately half of the others for lower bounds and between 0.71 and 0.81 the others when providing upper bounds. As mentioned previously, the bound with the lowest availability for upper bounds is a Lagrangian quadratic which is almost never available. When providing lower bounds, the Lagrangian envelope is the bound that has the smallest availability. The Lagrangian bounds with the exception of Lagrangian flat have very low availability $< 1\%$ even for $N = 10$. The Lagrangian flat is the Lagrangian bound with the highest availability with 0.29 when providing lower bounds for $N = 10$ and 1.18 when providing upper bounds for $N = 10$. For a given number of subintervals N, the robust bound with the smallest availability, robust flat, has still an availability higher than the Lagrangian bound with the highest availability, i.e. Lagrangian flat, for both upper and lower bounds.

In terms of error, Figure 8 displays the percentage of problems within a relative error range compared to the best found bound for a given problem for every bound and value of $N = 1$, 5 and 10. In that Figure, we see some clear tendencies in terms of error:

- **Three bounding methods dominate all the charts regardless of N:** Robust concave envelope, robust yzflat and robust fixed slope pairwise are the bounds with the smallest relative error which is also confirmed in Figure 7. For $N = 5$ and $N = 10$, when providing upper bounds, the robust line left and robust line right show significant improvements and are added to the bounds with the highest percentage of problems within a small error. Robust concave envelope displays a little growth throughout the charts meaning that it gives the best found bound for most problems.
- **Lagrangian flat dominates the other Lagrangian methods:** the Lagrangian flat always has a higher percentage of problems within a relative error range compared to other Lagrangian methods.
- **For upper bounds, a small number of problems have a relative error bigger than 2:** For upper bounds, most methods reach their maximum percentage of problems within an error range of $[1, 2]$, especially when $N = 5$ and $N = 10$.
- **For most bounds, there is no significant improvement between $\mathbf{N = 5}$ and $\mathbf{N = 10}$:** The difference between the curves of $N = 5$ and $N = 10$ for both lower and upper bounds is not significant, even though there is a small improvement.

The median relative timing is displayed in Table 5. When providing upper and lower bounds alike and for all N, robust flat has the smallest relative timing and robust concave envelope has the highest relative timing. All the other Lagrangian methods vary in the similar ranges as do all the other robust methods. The relative timing of the other Lagrangian and robust methods, even with $N = 10$, have a median of less than 0.51 and 1.06 respectively.

In summary, the method with the highest availability and smallest error is robust concave envelope but it suffers from its large relative timing (particularly for large N). The robust methods yzflat and fixed slope pairwise have small errors and have an availability close to robust concave envelope with a timing significantly smaller and much closer to 1 in relative time. Robust flat is the fastest method to compute but has larger errors and has a smaller availability. Out of all the Lagrangian bounds, the Lagrangian flat method is the most promising on all metrics.

4.3 Combining upper and lower bounds

In this section, we propose an iterative algorithm that tries to minimize the gap between upper and lower bounds. We use this iterative algorithm to compare the performance of different combinations of methods to get these bounds in terms of precision, availability and computation time.

4.3.1 Iterative algorithm

The idea behind the iterative algorithm is simple: we have two sets of methods, one that generates an upper bound and one a lower bound. The resulting

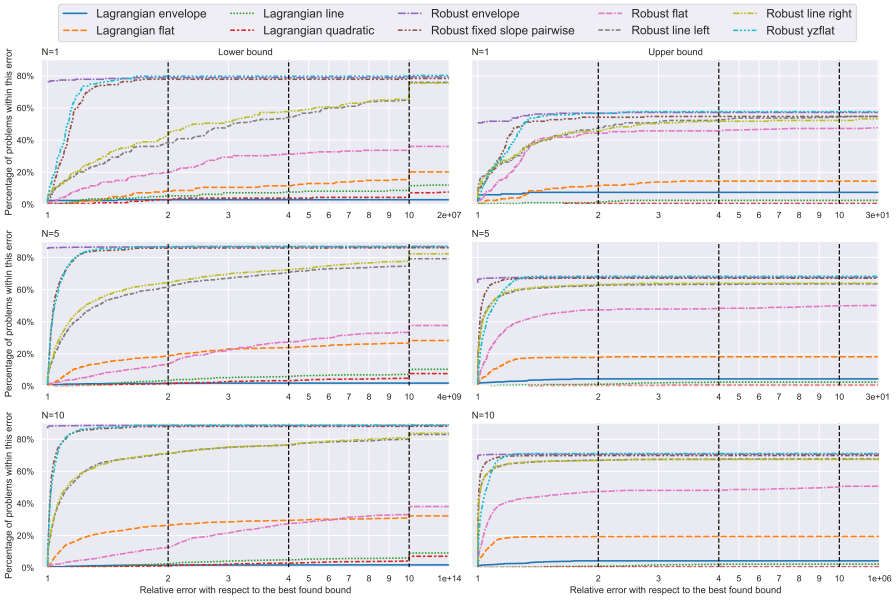

Fig. 8: Percentage of problems that are within a relative error range compared to the best bound for a given problem for every bound and value of $N = 1$, 5 and 10.

bounds can be refined by reducing the range $[\underline{\lambda}, \overline{\lambda}]$, i.e. dividing the range and reapplying the methods to find new upper and lower bounds for this new smaller range.

Algorithm 1 works using this principle. It maintains a priority queue of ranges to explore, starting with $[\underline{\lambda}, \overline{\lambda}]$. It iteratively selects the range with the largest error (defined by the maximum distance between the bounds computed in each range) and refines it, by dividing it into two equal parts and computing the bounds on these two parts, then reinserting them into the priority queue.

Degenerate cases exist where the upper/lower bounds cannot be found for a given range (for example when the Lagrangian relaxation is unbounded); in such cases, we write that the lower/upper bounds are respectively $-\infty$ and ∞ and that the distance between the bounds is infinite. This can lead the algorithm to exclusively focus on these ranges. In order to avoid this behaviour, we add a stopping condition on the algorithm: when the range length being refined is smaller than a user-defined ϵ_λ, the algorithm computes the ground truth $f(\lambda)$ at the middle of the range and stops refining this particular range (i.e. it considers that the error in this range is 0).

This is an anytime algorithm: at any point in time, the algorithm maintains the sets of bounds that have been computed.

Figure 9 shows four different steps of the iterative algorithm using robust yzflat for both upper and lower bounds., nominally the 1^{st}, 26^{th}, 76^{th} and last

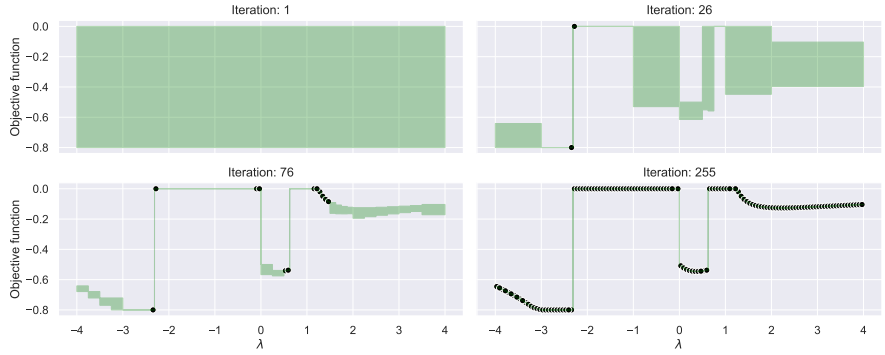

Fig. 9: Four steps of the iterative algorithm using robust yzflat for both upper and lower bounds on Problem (64) (Annex A.2). At any iteration, the algorithms ensures that $f(\lambda)$ lies in the green zone or is equal to the black dots.

(255^{th}) iteration for Problem (64) (Toy 1 from Annex A.2). The black dots are the truth $f(\lambda)$ points computed.

4.3.2 Results

In these experiments, we selected four upper bounds and five lower bounds based on the results from the previous section. We only selected those that had a high availability and a relatively low error and time requirements for computing 5 and 10 subintervals. For both bounds types of bounds, we selected robust fixed slope pairwise, robust yzflat, robust line right and robust flat. For the lower bounds, we add one additional method, Lagrangian flat. We pair each method that generates upper bounds with one method that generates lower bounds to obtain our sets of method to run the iterative algorithm. When talking about a particular combination of these methods, we specify the method used as an upper or lower bound by writing (ub) or (lb) respectively next to the method name. For each problem, we compute the iterative algorithm introduced in the previous section using each pair of lower and upper bounds from our selection. We define the timeout at three minutes. For each pair of bounds, we report the errors in the form of a boxplot when we stop at a relative timing of 1, i.e. the time required to compute the 100 points from $\mathcal{S}$. The errors are computed as the gap between the lower and upper bound. The results are shown in Figure 10. In the Figure, each pair is sorted by median error. The boxplot enables us to see different quartiles, each showing the percentage of problems for which the error is bigger than the value given by the line, e.g. the first line shows the quartile 10, for that value, we know that 10% of available problems have an error smaller and 90% an error greater than the value given by the line. In Figure 11, the median relative error of each bound is associated to its availability.

Algorithm 1 Iterative algorithm

function IT_ALG($\mathcal{P}$, $\underline{f}$, $\overline{f}$, $[\underline{\lambda}, \overline{\lambda}], \epsilon_\lambda, T$)
Input: $\mathcal{P}$ the problem
$\underline{f} : \mathbb{R}^2 \mapsto (\mathbb{R} \mapsto \mathbb{R} \cup \{-\infty\})$ which provides lower bounds
$\overline{f} : \mathbb{R}^2 \mapsto (\mathbb{R} \mapsto \mathbb{R} \cup \{\infty\})$ which provides upper bounds
$[\underline{\lambda}, \overline{\lambda}]$ the range to consider
ϵ_λ the size of smallest interval to consider before computing a point
T the time limit.
Output: $\mathcal{L}, \mathcal{U}, \mathcal{D}$ the sets of lower bounds, upper bounds and points (resp.) that have been computed.

$\mathcal{T} \leftarrow$ empty max-Priority Queue ▷ Todo-list of ranges
$\mathcal{L} \leftarrow \emptyset$ ▷ Set of computed lower bounds
$\mathcal{U} \leftarrow \emptyset$ ▷ Set of computed upper bounds
$\mathcal{D} \leftarrow \emptyset$ ▷ Set of computed points

$lb_i, ub_i \leftarrow \underline{f}(\underline{\lambda}, \overline{\lambda}), \overline{f}(\underline{\lambda}, \overline{\lambda})$
Insert $[\underline{\lambda}, \overline{\lambda}]$ with priority $\max_{\lambda \in [\underline{\lambda}, \overline{\lambda}]} ub_i(\lambda) - lb_i(\lambda)$ in $\mathcal{T}$
$\mathcal{L} \leftarrow \mathcal{L} \cup \{lb_i\}$
$\mathcal{U} \leftarrow \mathcal{U} \cup \{ub_i\}$

while $\mathcal{T} \neq \emptyset$ and time limit is not reached **do**
$[\underline{\lambda_i}, \overline{\lambda_i}] \leftarrow$ pop range from $\mathcal{T}$ with highest priority
mid $\leftarrow \frac{\overline{\lambda_i} + \underline{\lambda_i}}{2}$
if $\overline{\lambda_i} - \underline{\lambda_i} > \epsilon_l ambda$ **then**
$lb_1, ub_1 \leftarrow \underline{f}(\underline{\lambda_i}, \text{mid}), \overline{f}(\underline{\lambda_i}, \text{mid})$
$lb_2, ub_2 \leftarrow \underline{f}(\text{mid}, \overline{\lambda_i}), \overline{f}(\text{mid}, \overline{\lambda_i})$
Insert $[\underline{\lambda_i}, \text{mid}]$ with priority $\max_{\lambda \in [\underline{\lambda_i}, \text{mid}]} ub_1(\lambda) - lb_1(\lambda)$ in $\mathcal{T}$
Insert $[\text{mid}, \overline{\lambda}]$ with priority $\max_{\lambda \in [\text{mid}, \overline{\lambda}]} ub_2(\lambda) - lb_2(\lambda)$ in $\mathcal{T}$
$\mathcal{L} \leftarrow \mathcal{L} \cup \{lb_1, lb_2\}$
$\mathcal{U} \leftarrow \mathcal{U} \cup \{ub_1, ub_2\}$
else ▷ stop condition
$\mathcal{D} \leftarrow \mathcal{D} \cup \{(\text{mid}, f_{\mathcal{P}}(\text{mid}))\}$
end if
end while
return $\mathcal{L}, \mathcal{U}, \mathcal{D}$
end function

Overall, we see that all the combinations have an availability lower than 54% and all of them have a median error lower than 1. The methods with the highest value for the availability are robust yzflat (lb) with robust line right (ub) 53.1%, robust yzflat (lb and ub) 52.5% and robust line right (lb) with robust yzflat (ub) 52.1%. The lowest availability is Lagrangian flat (lb) with robust flat (ub) 6.1%. Figure 11 confirms that using robust yzflat (ub) and robust fixed slope pairwise (ub) have the best trade-off between availability and median relative error. The same conclusion can be taken when considering robust yzflat (lb), robust line right (lb) and robust fixed slope pairwise (lb) for the lower bound. We can see that the lower bound choice directly influences the availability of the bound whereas the upper bound influences more the median error for most methods with the exception of robust flat (ub). The

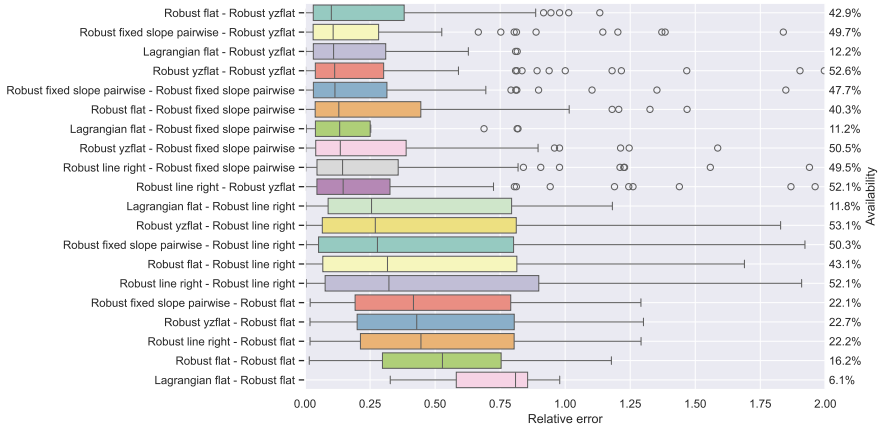

Fig. 10: Relative error for each pair of lower and upper bounding methods on the available problems for a relative timing of 1.

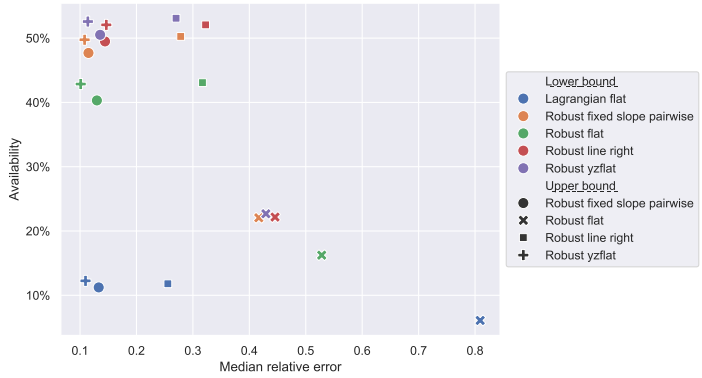

Fig. 11: Scatter plot of the bounding methods' availability with respect to the median relative error for a relative timing of 1.

Lagrangian flat method gives the worst trade-off in every case.

In terms of error, robust yzflat (ub) and robust fixed slope pairwise (ub) have a small relative error difference between the 25 and 75 quartile and keep a median lower than 0.25 making them particularly interesting. Robust flat and robust line right have bigger median errors and are more spreed between the quartiles. The variants of robust yzflat (ub) and robust fixed slope pairwise (ub) when combined with robust yzflat (lb), robust fixed slope pairwise (lb) and robust line right (lb) offer a high availability $> 49\%$ with a small median relative error < 0.25.

5 Conclusion

In this paper, we propose a novel approach for assessing the behaviour of the optimal objective function of a problem whose constraint coefficients can linearly vary within a given interval $[\underline{\lambda}, \overline{\lambda}]$. This approach consists of building bounds around the optimal objective function. Building bounds provide strong guarantees on the objective function and helps identify problematic behaviours in the objective that can otherwise be missed.

We present bounding methods based on robust optimization and Lagrangian relaxations derived in three groups of methods, one that computes constant bounds, the other variables bounds depending on λ and finally, envelope bounds. We benchmark these methods against a sampling protocol and highlight the efficiency of the bounding approach in terms of precision, availability and timing. For most problems, the bounding methods provide a high precision, a timing lower than the sampling approach and up to 80% availability. In particular, the bounding methods, robust envelope, robust yzflat, robust fixed slope pairwise and robust variable left and right outperform the other methods on all three metrics.

In the benchmark, we also prove that splitting the interval in equal subintervals can increase the accuracy and availability of a given bound and used this result to propose an iterative algorithm. The iterative algorithm uses a lower bound and an upper bound and iteratively refines the gap between the two by dividing the interval and reapplying the methods. We benchmark this algorithm against the same sampling approach on the three metrics. We show that the best combination of bounds can achieve a relative error of less than 0.25 in the same time as the one required by the sampling approach. However, the iterative approach has an availability lower than 55% in that amount of time. This can be improved by using more bounds concurrently.

As future work, we want to further assess our methods on real-life problems. First, we want to study if tuning the methods for particular problems, instead of going with a fully generic approach, can improve the performances of these methods. For instance, we expect "problem-specific" Lagrangian bounds to outperform their generic counterpart presented in this paper. Second, we want to increase the number of instances in our database for the generic study. To achieve that, we plan on integrating and automating the deployment of these bounding methods with other sensitivity analysis approaches in modelling tools such as JuMP[14], Pyomo[5] or The Graph-Based Optimization Modelling Language[16]. From a practitioners point of view, such integration would enable users to perform these analysis in a seamless manner.

Disclosure statement

The authors report there are no competing interests to declare.

Funding

The authors gratefully acknowledge the support Public Service of Wallonia through the funding of the NKL project in the framework of the Recovery and Resilience Plan (PNRR), initiated and financed by the European Union.

Availability of data and materials

The dataset and codes to run the experiments are available on Zenodo[17]. The code archive contains a Snakemake[19] build file allowing to reproduce all experiments. The raw outputs are also available[18].

References

1. Ilan Adler and Renato D. C. Monteiro. A geometric view of parametric linear programming. *Algorithmica*, 8:161–176, 1992.
2. Arjan B. Berkelaar, Kees Roos, and Tamás Terlaky. *The Optimal Set and Optimal Partition Approach to Linear and Quadratic Programming*, pages 159–202. Springer US, Boston, MA, 1997.
3. Dimitris Bertsimas and John Tsitsiklis. *Introduction to Linear Optimization*. Athena Scientific, 1st edition, 1997.
4. Dimitris Bertsimas and John N. Tsitsiklis. *Introduction to linear optimization*. Athena Scientific, 1997.
5. Michael L. Bynum, Gabriel A. Hackebeil, William E. Hart, Carl D. Laird, Bethany L. Nicholson, John D. Siirola, Jean-Paul Watson, and David L. Woodruff. *Pyomo–optimization modeling in python*, volume 67. Springer Science & Business Media, third edition, 2021.
6. Richard Flavell and Gerald R. Salkin. An approach to sensitivity analysis. *Journal of the Operational Research Society*, 26:857–866, 1975.
7. Thomas Gal and Josef Nedoma. Multiparametric linear programming. *Management Science*, 7(18):406–422, 1972.
8. Tomas Gal. *Postoptimal Analyses, Parametric Programming, and Related Topics, Degeneracy, Multicriteria Decision Making, Redundancy*. De Gruyter, New York, 1994.
9. Saul I. Gass and Thomas L. Saaty. Parametric objective function (part 2)- generalization. *Journal of the Operations Research Society of America*, 3(4):395–401, 1955.
10. David M. Gay. Electronic mail distribution of linear programming test problems. *Mathematical Programming Society COAL Newsletter*, 13:10–12, 1985.
11. Arthur M. Geoffrion and Ruth Nauss. Parametric and postoptimality analysis in integer linear programming. *Management Science*, 23(5):453–466, 1977.
12. Benjamin Jansen, Johan J. de Jong, Cees Roos, and Tamas Terlaky. Sensitivity analysis in linear programming: just be careful! *European Journal of Operational Research*, 101(1):15–28, 1997.
13. Rajab Khalilpour and Iftekar A. Karimin. Parametric optimization with uncertainty on the left hand side of linear programs. *Computers & Chemical Engineering*, 60:31–40, 2014.
14. Miles Lubin and Iain Dunning. Computing in operations research using Julia. *INFORMS Journal on Computing*, 27(2):238–248, 2015.
15. Jocelyn Mbenoun, Amina Benzerga, Bardhyl Miftari, Ghislain Detienne, Thierry Deschuyteneer, Juan Vazquez, Guillaume Derval, and Damien Ernst. Integration of offshore energy into national energy system: a case study on belgium. 2024.
16. Bardhyl Miftari, Mathias Berger, Guillaume Derval, Quentin Louveaux, and Damien Ernst. Gboml: a structure-exploiting optimization modelling language in python. *Optimization Methods and Software*, 0(0):1–30, 2023.

17. Bardhyl Miftari and Guillaume Derval. MiftariB/lhs-bounding-sensitivity: Paper (october 2024 version), October 2024.
18. Bardhyl Miftari and Guillaume Derval. Paper: "Sensitivity analysis for linear changes of the constraint matrix of a linear program" output, October 2024.
19. Félix Möder, Kim Philipp Jablonski, Brice Letcher, Michael B. Hall, Christopher H. Tomkins-Tinch, Vanessa Sochat, Jan Forster, Soohyun Lee, Sven O. Twardziok, Alexander Kanitz, Andreas Wilm, Manuel Holtgrewe, Sven Rahmann, Sven Nahnsen, and Johannes Köter. Sustainable data analysis with snakemake [version 1; peer review: 1 approved, 1 approved with reservations]. *F1000Research*, 10(33), 2021.
20. Thomas Saaty and Saul Gass. Parametric objective function (part 1). *Journal of the Operations Research Society of America*, 2(3):316–319, 1954.
21. Jack Sherman and Winifred J. Morrison. Adjustment of an inverse matrix corresponding to changes in the elements of a given column or a given row of the original matrix. *The Annals of Mathematical Statistics*, 20:620–624, 1949.
22. Jack Sherman and Winifred J. Morrison. Adjustment of an inverse matrix corresponding to a change in one element of a given matrix. *The Annals of Mathematical Statistics*, 21(1):124–127, 1950.
23. Max A. Woodbury. *Inverting Modified Matrices*. Memorandum Report / Statistical Research Group, Princeton. Department of Statistics, Princeton University, 1950.
24. Laurence Wosley. Integer programming. *IIE transactions.*, 32(273-285), 2000.
25. Rob A. Zuidwijk. Linear parametric sensitivity analysis of the constraint coefficient matrix in linear programs. *ERIM report series research in management*, ERS-2005-055-LIS, 2005.

A APPENDIX

A.1 Lemma quadratic function

This Lemma is used in Section 3.3.

Lemma 2 *Given a quadratic function $q(x) = ax^2 + bx + c$. The maximum of $q(x)$ over $x_1 \leq x \leq x_2$ is upper bounded by*

$$\max \begin{cases} ax_1^2 + bx_1 + c \\ ax_2^2 + bx_2 + c \\ ax_1x_2 + b\frac{x_1+x_2}{2} + c \end{cases} \tag{55}$$

Proof We analyse separately the concave and the convex case:

- If the function $q(x)$ is convex, that is if $a \geq 0$, then the maximum of the function in the polyhedron $[x_1, x_2]$. The equation (2) is known to be attained at one of its vertices, either x_1 or x_2.
- For the concave case, let us consider the two tangents at x_1 and x_2:

$$t_{x_1}(x) = ax_1^2 + bx_1 + c + (x - x_1)(2ax_1 + b) = c + 2ax_1x - ax_1^2 + bx \tag{56}$$

$$t_{x_2}(x) = ax_2^2 + bx_2 + c + (x - x_2)(2ax_2 + b) = c + 2ax_2x - ax_2^2 + bx. \tag{57}$$

If $q(x)$ is concave ($a \leq 0$) then any tangent is an upper bound for the function for any x. Taking two tangent at different points and taking the minimum of the two functions is also an upper bound. That is, for any $x_1 < x_2$:

$$\min(t_{x_1}(x), t_{x_2}(x)) \geq q(x) \quad \forall x. \tag{58}$$

$\min(t_{x_1}(x), t_{x_2}(x))$ forms a piecewise linear function. The tangents t_{x_1} and t_{x_2} intersect at the middle point of $[x_1, x_2]$:

$$t_{x_1}(x) = t_{x_2}(x) \tag{59}$$

$$2ax_1x - ax_1^2 = 2ax_2x - ax_2^2 \tag{60}$$

$$2x_1x - x_1^2 = 2x_2x - x_2^2 \tag{61}$$

$$x = \frac{x_2^2 - x_1^2}{2(x_2 - x_1)} = \frac{x_1 + x_2}{2} \tag{62}$$

with the value $t_1(\frac{x_1+x_2}{2}) = ax_1x_2 + b\frac{x_1+x_2}{2} + c$.
Now, the maximum of the function $\min(t_{x_1}(x), t_{x_2}(x))$ over $[x_1, x_2]$ is achieved either on $t_{x_1}(x_1)$ or $t_{x_2}(x_2)$ or in the middle of the range, at $t_1(\frac{x_1+x_2}{2})$:

$$\max_{x \in [x_1, x_2]} \min(t_{x_1}(x), t_{x_2}(x)) = \max \begin{cases} ax_1^2 + bx_1 + c \\ ax_2^2 + bx_2 + c \\ ax_1x_2 + b\frac{x_1+x_2}{2} + c \end{cases} \tag{63}$$

A.2 Dataset

In this Appendix, we provide a full overview the problems in the proposed dataset.

A.2.1 Illustrative problems

This category contains three small two-variable problems that are used for the sake of illustration rather than being of particular interest.

1. **Toy 1:**

$$\begin{aligned}
\mathcal{P}_{\text{toy }1}(\lambda) \equiv \min \quad & \begin{pmatrix} -1 & 2 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \\
\text{s.t} \quad & \begin{pmatrix} -2 & -1 \\ 1 & 2 \\ -1 & 1 \\ 1 & -1 \\ 1 & -3 \\ 2 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \leq \begin{pmatrix} 3 \\ 0 \\ 2 \\ 1 \\ 1 \\ 3 \end{pmatrix} \\
& \begin{pmatrix} 3 & -1 \\ -3 & 1 \\ -2 & 1 \\ 2 & -1 \\ -1 & -2 \\ -1 & -3 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \lambda \begin{pmatrix} 3 & 3 \\ 3 & -2 \\ 2 & 3 \\ 2 & -1 \\ -2 & 1 \\ 0 & -2 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \leq \begin{pmatrix} 0 \\ 0 \\ 0 \\ 3 \\ 3 \\ 1 \end{pmatrix} \\
& \forall \lambda \in [-4, 4]
\end{aligned} \tag{64}$$

2. **Toy 2:**

$$\begin{aligned}
\mathcal{P}_{\text{toy }2}(\lambda) \equiv \min \quad & \begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \\
\text{s.t} \quad & \begin{pmatrix} -2 & 0 \\ 2 & 2 \\ -1 & 0 \\ -1 & -1 \\ 0 & 1 \\ -3 & -2 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \leq \begin{pmatrix} 1 \\ 2 \\ 2 \\ 3 \\ 3 \\ 1 \end{pmatrix} \\
& \begin{pmatrix} 2 & 3 \\ 0 & 1 \\ 1 & -3 \\ 0 & -3 \\ 3 & 0 \\ -2 & 3 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \lambda \begin{pmatrix} 3 & 3 \\ 0 & 2 \\ -2 & -1 \\ 2 & 1 \\ 2 & 0 \\ 2 & 2 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \leq \begin{pmatrix} 3 \\ 1 \\ 2 \\ 3 \\ 3 \\ 1 \end{pmatrix} \\
& \forall \lambda \in [-4, 4]
\end{aligned} \tag{65}$$

3. **Toy 3:** This problem is the one written in (4). We rewrite it here for the sake of completeness.

$$\begin{aligned}
\mathcal{P}_{\text{toy }3}(\lambda) \equiv \min \quad & \begin{pmatrix} 2 & -2 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \\
\text{s.t} \quad & \begin{pmatrix} -2 & 2 \\ -1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \leq \begin{pmatrix} 4 \\ 1 \end{pmatrix} \\
& \begin{pmatrix} 2 & 1 \\ -2 & -3 \\ 2 & 2 \\ -1 & -4 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \lambda \begin{pmatrix} -1 & -4 \\ 0 & 4 \\ -4 & -3 \\ 2 & 4 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \leq \begin{pmatrix} 4 \\ 2 \\ 0 \\ 2 \end{pmatrix} \\
& \forall \lambda \in [-10, 9]
\end{aligned}$$

4. **Toy 4:** This problem is the one written in (13). We rewrite it for the sake of completeness.

$$\begin{aligned} \mathcal{P}_{\text{toy 4}}(\lambda) \equiv \min_{x,y} \quad & \begin{pmatrix} -2 & -2 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \\ \text{s.t} \quad & \begin{pmatrix} 3 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \leq \begin{pmatrix} 3 \end{pmatrix} \\ & \begin{pmatrix} -5 & -2 \\ 1 & 4 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \lambda \begin{pmatrix} -3 & -2 \\ -3 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \leq \begin{pmatrix} 0 \\ -3 \end{pmatrix} \\ & \forall \lambda \in [-2, 2] \end{aligned} \tag{66}$$

A.2.2 Real-life problems

This category contains two real-life problems from the energy field.

1. A **microgrid** sizing problem: determining the optimal number of PV panels and batteries to install in order to minimize the overall energy bill given electricity prices and consumption. The uncertainty is applied on the capacity factor, i.e. a conversion factor that is directly dependant on the solar irradiance, PV inclination and much more, that determines the amount of electricity produced. The problem has 172 variables and 342 total constraints with 24 constraints concerned by the modification.
2. A **multi-energy nation-wide system**: The energy system is the one presented in Mbenoun et al. [15]. The problem is a multi-million-variables model of the Belgian energy system, focusing on the impact of potential offshore hydrogen production and transportation. The uncertainty is applied on the electrolysis efficiency that varies between 0.01% and 100% efficiency. The problem has 501229 variables and 992339 constraints with 6480 constraints concerned by the modification.

A.2.3 Netlib derived problems

Here is a list of all the problems that we have taken from Netlib[10]:

- 25fv47
- 80bau3b
- adlittle
- agg
- agg2
- agg3
- bandm
- beaconfd
- blend
- bnl1
- bnl2
- boeing1
- boeing2
- bore3d
- brandy
- capri
- cycle
- czprob
- d2q06c
- d6cube
- degen2
- degen3
- e226
- etamacro
- ffff800
- finnis
- fit1d
- fit1p
- fit2d
- fit2p
- forplan
- ganges
- gfrd-pnc
- greenbea
- greenbeb
- grow15
- grow22
- grow7
- israel
- lotfi
- maros
- maros-r7
- modszk1
- nesm
- perold
- pilot4
- pilot87
- pilot
- pilotnov
- recipe
- sc105
- sc205
- scagr25
- scagr7
- scfxm1
- scfxm2
- scfxm3
- scorpion
- scrs8
- scsd1
- scsd6
- scsd8
- sctap1
- sctap2
- sctap3
- seba
- share1b
- share2b
- shell
- ship04l
- ship04s
- ship08l
- ship08s
- ship12l
- ship12s
- sierra
- standata
- standgub
- standmps
- stocfor1
- stocfor2