Optimization and Engineering (2021) 22:247–259
https://doi.org/10.1007/s11081-020-09508-9

RESEARCH ARTICLE

Automatic repair of convex optimization problems

Shane Barratt[1] · **Guillermo Angeris**[1] · **Stephen Boyd**[1]

Received: 4 February 2020 / Revised: 24 April 2020 / Accepted: 25 April 2020 / Published online: 23 May 2020

Abstract
Given an infeasible, unbounded, or pathological convex optimization problem, a natural question to ask is: what is the smallest change we can make to the problem's parameters such that the problem becomes solvable? In this paper, we address this question by posing it as an optimization problem involving the minimization of a convex regularization function of the parameters, subject to the constraint that the parameters result in a solvable problem. We propose a heuristic for approximately solving this problem that is based on the penalty method and leverages recently developed methods that can efficiently evaluate the derivative of the solution of a convex cone program with respect to its parameters. We illustrate our method by applying it to examples in optimal control and economics.

Keywords Convex programming · Infeasibility · Unboundedness · Parametric optimization

1 Introduction

Parametrized convex optimization We consider parametrized convex optimization problems, which have the form

$$\begin{array}{ll} \text{minimize} & f_0(x;\theta) \\ \text{subject to} & f_i(x;\theta) \leq 0, \quad i = 1, \ldots, m, \\ & g_i(x;\theta) = 0, \quad i = 1, \ldots, p, \end{array} \tag{1}$$

where $x \in \mathrm{R}^n$ is the optimization variable, $\theta \in \mathrm{R}^k$ is the parameter, the objective function $f_0 : \mathrm{R}^n \times \mathrm{R}^k \to \mathrm{R}$ is convex in x, the inequality constraints functions

✉ Shane Barratt
stbarratt@gmail.com

Guillermo Angeris
angeris@stanford.edu

Stephen Boyd
boyd@stanford.edu

[1] Department of Electrical Engineering, Stanford University, Stanford, USA

$f_i : \mathrm{R}^n \times \mathrm{R}^k \to \mathrm{R}, i = 1, \ldots, m$, are convex in x, and the equality constraint functions $g_i : \mathrm{R}^n \times \mathrm{R}^k \to \mathrm{R}, i = 1, \ldots, p$, are affine in x.

Solvable problems A point x is said to be feasible if $f_i(x;\theta) \leq 0$, $i = 1, \ldots, m$, and $g_i(x;\theta) = 0, i = 1, \ldots, k$. The optimal value $p^\star$ of the problem (1) is defined as

$$p^\star = \inf\{f_0(x;\theta) \mid f_i(x;\theta) \leq 0,\ i = 1, \ldots, m,\ h_i(x;\theta) = 0,\ i = 1, \ldots, p\}.$$

We allow $p^\star$ to take on the extended values $\pm\infty$. Roughly speaking, we say that (1) is *solvable* if $p^\star$ is finite and attainable. (We will define solvable formally in Sect. 2, when we canonicalize (1) into a cone program.) When the problem is unsolvable, it falls into one of three cases: it is *infeasible* if $p^\star = +\infty$, *unbounded below* if $p^\star = -\infty$, and *pathological* if $p^\star$ is finite, but not attainable by any x, or strong duality does not hold for (1). Unsolvable problems are often undesirable since, in many cases, there does not exist a solution.

Performance metric The goal in this paper is to repair an unsolvable problem by adjusting the parameter θ so that it becomes solvable. We will judge the desirability of a new parameter θ by a (convex) performance metric function $r : \mathrm{R}^k \to \mathrm{R} \cup \{+\infty\}$, which we would like to be small. (Infinite values of r denote constraints on the parameter.) A simple example of r is the Euclidean distance to an initial parameter vector θ_0, or $r(\theta) = \|\theta - \theta_0\|_2$.

Repairing a convex optimization problem In this paper, we consider the problem of repairing a convex optimization problem, as measured by the performance metric, by solving the problem

$$\begin{array}{ll} \text{minimize} & r(\theta) \\ \text{subject to} & \text{problem (1) is solvable,} \end{array} \tag{2}$$

with variable θ.

Pathologies There are various pathologies that can occur in this formulation. For example, the set of θ that lead to solvable problems could be open, meaning there might not exist a solution to (2), or the complement could have (Lebesgue) measure zero, meaning that the problem can be made solvable by essentially any perturbation. Both of these cases can be demonstrated with the following problem:

$$\begin{array}{ll} \text{minimize} & 0 \\ \text{subject to} & \theta x = 1, \end{array} \tag{3}$$

and regularization function $r(\theta) = \theta^2$. The set of solvable θ is $\{\theta \mid \theta \neq 0\}$, which is both open and has complement with measure zero. The optimal value of problem (2) is 0, but is not attainable by any solvable θ. The best we can hope to do in these situations is to produce a minimizing sequence.

NP-hardness Repairing a convex optimization problem is NP-hard. To show this, we reduce the 0–1 integer programming problem

$$\begin{array}{ll} \text{minimize} & 0 \\ \text{subject to} & Ax = b, \\ & x \in \{0, 1\}^n, \end{array} \tag{4}$$

with variable $x \in \mathrm{R}^n$ and data $A \in \mathrm{R}^{m \times n}$ and $b \in \mathrm{R}^m$ to an instance of problem (2).

Let $r(\theta) = 0$. The convex optimization problem that we would like to be solvable be

$$\begin{array}{ll} \text{minimize} & 0 \\ \text{subject to} & x = \theta, \\ & \theta x = x, \\ & Ax = b, \end{array} \tag{5}$$

with variable x. Problem (2) has the same constraints as (4), since (5) is feasible if and only if $\theta_i(\theta_i - 1) = 0$, $i = 1, \ldots, n$, and $A\theta = b$. Therefore, the problem of finding any feasible parameters for (1) (i.e., with $r(\theta) = 0$), is at least as hard as the 0–1 integer programming problem, which is known to be NP-hard (Karp 1972).

2 Cone program formulation

In practice, most convex optimization problems are solved by reformulating them as equivalent conic programs and passing the numerical data in the reformulated problem to general conic solvers such as SCS (O'Donoghue et al. 2016), MOSEK (2020) or GUROBI (2019). This process of canonicalization is often done automatically by packages like CVXPY (Diamond and Boyd 2016), which generate a conic program from a high-level description of the problem.

Canonicalization For the remainder of the paper, we will consider the canonicalized form of problem (1). The primal (P) and dual (D) form of the canonicalized convex cone program is (see, e.g., Ben-Tal and Nemirovski 2001; Boyd and Vandenberghe 2004)

$$\begin{array}{lll} \text{(P)} & \text{minimize} & c(\theta)^T x \\ & \text{subject to} & A(\theta)x + s = b(\theta), \\ & & s \in \mathcal{K}, \end{array} \qquad \begin{array}{lll} \text{(D)} & \text{minimize} & -b(\theta)^T y \\ & \text{subject to} & A(\theta)^T y + c(\theta) = 0, \\ & & y \in \mathcal{K}^*. \end{array} \tag{6}$$

Here $x \in \mathrm{R}^n$ is the primal variable, $y \in \mathrm{R}^m$ is the dual variable, and $s \in \mathrm{R}^m$ is the slack variable. The functions $A : \mathrm{R}^k \to \mathrm{R}^{m \times n}$, $b : \mathrm{R}^k \to \mathrm{R}^m$, and $c : \mathrm{R}^k \to \mathrm{R}^n$ map the parameter vector θ in problem (1) to the cone program problem data A, b, and c. The set $\mathcal{K} \subseteq \mathrm{R}^m$ is a closed convex cone with associated dual cone $\mathcal{K}^* = \{y \mid y^T x \geq 0 \text{ for all } x \in \mathcal{K}\}$.

Solvability We will adopt a simple and relatively general definition of solvability. We say that an instance of the primal problem (6) is *solvable* whenever the primal (P) is feasible, the dual (D) is feasible, strong duality holds (i.e., the optimal value of problems (P) and (D) are equal), and this value is achievable by a pair of primal and dual feasible points.

These conditions are sufficient to ensure that the problem has a solution that is bounded and that the problem is also easy to solve. There are, of course, many pathological cases where strong duality does not hold or the primal optimal value is not achievable [see, e.g., Ben-Tal and Nemirovski (2001, Sect. 1.4.6), yet the problem

is still 'solvable' in the sense that there exist algorithms that can efficiently solve the problem or give a sequence of feasible points that minimize the objective. We exclude such pathological problems from our definition.

Solution The vector $(x^\star, y^\star, s^\star)$ is a solution to problem (6) if

$$\begin{bmatrix} A(\theta)x^\star + s^\star \\ A(\theta)^T y^\star + c(\theta) \\ c(\theta)^T x^\star + b(\theta)^T y^\star \end{bmatrix} = \begin{bmatrix} b(\theta) \\ 0 \\ 0 \end{bmatrix}, \quad (s^\star, y^\star) \in \mathcal{K} \times \mathcal{K}^*. \tag{7}$$

These conditions merely state that $(x^\star, s^\star)$ is primal feasible, $y^\star$ is dual feasible, and that there is zero duality gap, which implies that $(x^\star, y^\star, s^\star)$ is optimal by weak duality (Boyd and Vandenberghe (2004, Sect. 5.2.2). Problems (1) and (6) are solvable if and only if there exists a point that satisfies (7).

Primal-dual embedding The primal-dual embedding of problem (6) is the cone program

$$\begin{array}{ll} \text{minimize} & t \\ \text{subject to} & \left\| \begin{bmatrix} A(\theta)x + s - b(\theta) \\ A(\theta)^T y + c(\theta) \\ c(\theta)^T x + b(\theta)^T y \end{bmatrix} \right\|_2 \leq t \\ & s \in \mathcal{K}, \quad y \in \mathcal{K}^*, \end{array} \tag{8}$$

with variables t, x, y, and s. This problem is guaranteed to be feasible since, for any $\theta \in \mathrm{R}^k$, setting $x = 0$, $y = 0$, $s = 0$, and $t = \|(b(\theta), c(\theta))\|_2$ yields a feasible point. The problem is also guaranteed to be bounded from below, since the objective is nonnegative. Taken together, this implies that problem (8) always has a solution, assuming it is not pathological.

Optimal value of (8) Let $t^\star : \mathrm{R}^k \to \mathrm{R}$ denote the optimal value of problem (8) as a function of θ. Notably, we have that $t^\star(\theta) = 0$ if and only if problem (1) is solvable, since if $t^\star(\theta) = 0$, the solution to problem (8) satisfies (7) and therefore problem (1) is solvable. On the other hand, if problem (1) is solvable, then there exists a point that satisfies (7) and is feasible for (8), so $t^\star(\theta) = 0$.

Differentiability of $t^\star$ In practice, $t^\star$ is often a differentiable function of θ. This is the case when A, b, and c are differentiable, which we will assume, and the optimal value of problem (8) is differentiable in A, b, and c. Under some technical conditions that are often satisfied in practice, the optimal value of a cone program is a differentiable function of its problem data (Agrawal et al. 2019b). We will assume that $t^\star$ is differentiable, and that we can efficiently compute its gradient $\nabla t^\star(\theta)$ using the methods described in Agrawal et al. (2019b) and the chain rule.

Reformulation In light of these observations, we can reformulate problem (2) as

$$\begin{array}{ll} \text{minimize} & r(\theta) \\ \text{subject to} & t^\star(\theta) = 0, \end{array} \tag{9}$$

with variable θ. Here we have replaced the intractable constraint in problem (2) with an equivalent smooth equality constraint. Since this problem is NP-hard, we must resort to heuristics to find approximate solutions; we give one in Sect. 3.

3 Heuristic solution method

Penalty method One simple heuristic is to use the penalty method to (approximately) solve (9). Starting from $\theta^0 \in \mathrm{R}^k$ and $\lambda^0 > 0$, at iteration ℓ, the penalty method performs the update

$$\theta^{\ell+1} = \operatorname*{argmin}_{\theta} \ \lambda^{\ell} r(\theta) + t^{\star}(\theta), \tag{10}$$

and then decreases λ^{ℓ}, e.g., $\lambda^{\ell+1} = (1/2)\lambda^{\ell}$, until $t^{\star}(\theta^{\ell}) \leq \epsilon_{\text{out}}$ for some given tolerance $\epsilon_{\text{out}} > 0$.

To perform the update (10), we must solve the unconstrained optimization problem

$$\text{minimize} \ \ L(\theta, \lambda^{\ell}) = \lambda^{\ell} r(\theta) + t^{\star}(\theta), \tag{11}$$

with variable θ. The objective is the sum of a differentiable function and a potentially nonsmooth convex function, for which there exist many efficient methods. The simplest (and often most effective) of these methods is the proximal gradient method [which stems from the proximal point method (Martinet 1970); for a modern reference see Nesterov (2013)]. The proximal gradient method consists of the iterations

$$\theta^{\ell+1} = \mathbf{prox}_{\alpha^{\ell}\lambda^{\ell} r}\left(\theta^{\ell} - \alpha^{\ell} \nabla t^{\star}(\theta^{\ell})\right),$$

where the proximal operator is defined as

$$\mathbf{prox}_{\alpha\lambda r}(\tilde{\theta}) = \operatorname*{argmin}_{\theta} \ \alpha\lambda r(\theta) + \frac{1}{2}\|\theta - \tilde{\theta}\|_2^2.$$

Since r is convex, evaluating the proximal operator of $\alpha\lambda r$ requires solving a convex optimization problem. Indeed, for many practical choices of r, its proximal operator has a closed-form expression (Parikh and Boyd 2014, Sect. 6).

We run the proximal gradient method until the stopping criterion

$$\|(\theta^l - \theta^{l+1})/\alpha^l + (g^{l+1} - g^l)\|_2 \leq \epsilon_{\text{in}},$$

is reached, where $g^l = \nabla t^{\star}(\theta^l)$, for some given tolerance $\epsilon_{\text{in}} > 0$ (Barratt and Boyd 2019). We employ the adaptive step size scheme described in Barratt and Boyd (2019). The full procedure is described in algorithm 3.1 below.

Algorithm 3.1 *Finding the closest solvable convex optimization problem.*

given regularization function r, initial parameter θ^0, penalty λ^0, step size α^0, iterations n_{iter}, outer tolerance ϵ_{out}, inner tolerance ϵ_{in}.

for $l = 1, \ldots, n_{\text{iter}}$

1. *Compute* $t^\star$. Compute $t^\star(\theta^\ell)$.
2. *Compute gradient of* $t^\star$. Let $g^\ell = \nabla t^\star(\theta^\ell)$.
3. *Compute the gradient step.* Let $\theta^{\ell+1/2} = \theta^\ell - \alpha^\ell g^\ell$.
4. *Compute the proximal operator.* Let $\theta^{\text{tent}} = \mathbf{prox}_{\alpha^\ell \lambda^\ell r}(\theta^{\ell+1/2})$.
5. **if** $L(\theta^{\text{tent}}, \lambda^\ell) < L(\theta^\ell, \lambda^\ell)$,
   Increase step size and accept update. $\alpha^{\ell+1} = (1.2)\alpha^\ell, \quad \theta^{\ell+1} = \theta^{\text{tent}}$.
   Inner stopping criterion. if $\|(\theta^\ell - \theta^{\ell+1})/\alpha^\ell + (g^{\ell+1} - g^\ell)\|_2 \leq \epsilon_{\text{in}}$, then set $\lambda^{\ell+1} = (1/2)\lambda^\ell$.
6. **else** *Decrease step size and reject update.* $\alpha^{\ell+1} = (1/2)\alpha^\ell, \quad \theta^{\ell+1} = \theta^\ell$.
7. *Outer stopping criterion.* if $t^\star(\theta^{l+1}) \leq \varepsilon_{\text{out}}$, quit.

end for

Runtime Algorithm 3.1 has two loops: one which updates the parameter in the penalty method, and one which approximately solves the sub-problem using the proximal gradient method. At every iteration of the algorithm, we need to solve an instance of problem (8), and compute the derivative of the optimal value with respect to the parameters. Therefore, algorithm 3.1 involves solving tens to hundreds of convex optimization problems, and as a result can be much slower than simply verifying infeasibility or unboundedness. However, there are several performance enhancements we can employ that make solving each of these problems faster. One enhancement is warm-starting, where we use the solution from the most recently solved problem to initialize the variables in the next problem. Another is factorization caching, where we re-use parts of matrix factorizations from previous problems.

Implementation We have implemented algorithm 3.1 in Python, which is available online at

`https://github.com/cvxgrp/cvxpyrepair`

The interface is the `repair` method, which, given a parametrized CVXPY problem (Diamond and Boyd 2016) and a convex regularization function, uses algorithm 3.1 to find the parameters that approximately minimize that regularization function and result in a solvable CVXPY problem. We use SCS (O'Donoghue et al. 2016) to solve cone programs and diffcp (Agrawal et al. 2019b) to compute the gradient of cone programs. We require the CVXPY problem to be a disciplined parametrized program (DPP), so that the mapping from parameters to (A, b, c) is affine, and hence differentiable (Agrawal et al. 2019a).

Until this point, we have assumed that the optimal value of problem (8) is differentiable in A, b, and c. However, in our implementation, we do not require this to be the case. So long as it is differentiable almost everywhere, it is reasonable to apply the proximal gradient method to (11). At non-differentiable points, we instead compute a heuristic quantity. For example, a source of non-differentiability

is the singularity of a particular matrix; in this case, diffcp computes a least-squares approximation of the gradient (Agrawal et al. 2019b, Sect. 3).

4 Examples

4.1 Spacecraft landing

We consider the problem of landing a spacecraft with a gimbaled thruster. The dynamics are

$$m\ddot{x}(t) = f(t) - mge_3,$$

where $m > 0$ is the spacecraft mass, $x(t) \in \mathrm{R}^3$ is the spacecraft position, $f(t) \in \mathrm{R}^3$ is the force applied by the thruster, $g > 0$ is the gravitational acceleration, and $e_3 = (0, 0, 1)$.

Our goal, given some initial position $x^{\mathrm{init}} \in \mathrm{R}^3$ and velocity $v^{\mathrm{init}} \in \mathrm{R}^3$, is to land the spacecraft at zero position and velocity at some touchdown time $T > 0$, i.e., $x(T) = 0$ and $\dot{x}(T) = 0$.

We have a total available fuel M^{fuel} and a thrust limit $F^{\max}$. This results in the constraints

$$\int_0^T \gamma \|f(t)\|_2 \, dt \leq M^{\mathrm{fuel}}, \quad \|f(t)\|_2 \leq F^{\max}, \quad 0 \leq t \leq T,$$

where γ is the fuel consumption coefficient. We also have a gimbal constraint

$$f_3(t) \geq \alpha \|(f_1(t), f_2(t))\|_2,$$

where α is equal to the tangent of the maximum gimbal angle.

We discretize the thrust profile, position, and velocity at intervals of length h, or

$$f_k = f((k-1)h), \quad x_k = x((k-1)h), \quad v_k = \dot{x}((k-1)h), \quad k = 1, \ldots, H,$$

where $H = T/h + 1$.

To find if there exists a thrust profile to land the spacecraft under this discretization, we solve the problem

$$\begin{array}{ll} \text{minimize} & 0 \\ \text{subject to} & x_{k+1} = x_k + (h/2)(v_{k+1} + v_k), \quad k = 1, \ldots, H, \\ & mv_{k+1} = f_k - hmge_3, \quad k = 1, \ldots, H, \\ & \|f_k\|_2 \leq F^{\max}, \quad k = 1, \ldots, H, \\ & \sum_{k=1}^{H} h\gamma \|f_k\|_2 \leq M^{\mathrm{fuel}}, \\ & (f_k)_3 \geq \alpha \|((f_k)_1, (f_k)_2)\|_2, \\ & x_1 = x^{\mathrm{init}}, \quad v_1 = v^{\mathrm{init}}, \\ & x_H = 0, \quad v_H = 0, \end{array} \tag{12}$$

with variables x, v, and f. This problem is a parametrized convex optimization problem, with parameter

$$\theta = (m, M^{\text{fuel}}, F^{\max}, \alpha).$$

Suppose that we are given a parameter vector θ_0 for which it is impossible to find a feasible thrust profile, i.e., problem (12) is infeasible. Suppose, in addition, that we are allowed to change the spacecraft's parameters in a limited way. We seek to find the smallest changes to the mass and constraints on the fuel and thrust limit that guarantees the feasibility of problem (12). We can (approximately) do this with algorithm 3.1.

Numerical example We consider a numerical example with data

$$T = 10, \quad h = 1, \quad g = 9.8, \quad x^{\text{init}} \begin{bmatrix} 10 \\ 10 \\ 50 \end{bmatrix}, \quad v^{\text{init}} = \begin{bmatrix} 10 \\ -10 \\ -10 \end{bmatrix}, \quad \gamma = 1,$$

and initial parameters

$$m_0 = 12, \quad M_0^{\text{fuel}} = 200, \quad F_0^{\max} = 50, \quad \alpha_0 = 0.5.$$

The initial parameters are infeasible, i.e., there is no possible thrust profile which allows the spacecraft to land in time, so we use algorithm 3.1 to modify the design parameters in order to have a feasible landing thrust profile. We use the performance metric

$$r(\theta) = \begin{cases} \frac{|m-m_0|}{m_0} + \frac{|M^{\text{fuel}}-M_0^{\text{fuel}}|}{M_0^{\text{fuel}}} + \frac{|F^{\max}-F_0^{\max}|}{F_0^{\max}} + \frac{|\alpha-\alpha_0|}{\alpha_0} & m \geq 9, \\ +\infty & \text{otherwise}, \end{cases}$$

which constrains the mass to be greater than or equal to 9, and penalizes the percentage change in each of the parameters. The resulting feasible design has the parameters

$$m = 9.03, \quad M^{\text{fuel}} = 271.35, \quad F^{\max} = 67.16, \quad \alpha = 0.5,$$

and $r(\theta) = 0.948$.

4.2 Arbitrage

Consider an event (e.g., horse race, sports game, or a financial market over a short time period) with m possible outcomes and n possible wagers on the outcome. The return matrix is $R \in \mathbf{R}^{m\times n}$, where R_{ij} is the return in dollars for the outcome i and wager j per dollar bet. A betting strategy is a vector $w \in \mathbf{R}^n_+$, where w_i is the amount that we bet on the ith wager. If we use a betting strategy w and outcome i occurs, then the return is $(Rw)_i$ dollars.

We say that there is an *arbitrage opportunity* in this event if there exists a betting strategy $w \in \mathbf{R}^n_+$ that is guaranteed to have nonnegative return for each outcome, and positive return in at least one outcome. We can check whether there exists an arbitrage opportunity by solving the convex optimization problem

$$\begin{array}{ll} \text{maximize} & \mathbf{1}^T Rw \\ \text{subject to} & Rw \geq 0, \\ & w \geq 0, \end{array} \tag{13}$$

with variable w. If this problem is unbounded above, then there is an arbitrage opportunity.

Suppose that we are the event organizer (e.g., sports book director, bookie, or financial exchange) and we wish to design the return matrix R such that there are arbitrage opportunities and that some performance metric r is small. We can tackle this problem by finding the nearest solvable convex optimization problem to problem (13) using algorithm 3.1.

Numerical example We consider a horse race with $n = 3$ horses and $m = 5$ outcomes. The initial return matrix is

$$R_0 = \begin{bmatrix} 0.05 & 1.74 & -0.88 \\ 0.08 & 0.45 & -1.02 \\ 0.18 & -0.31 & 1.29 \\ 0.9 & -1.17 & 0.27 \\ -0.93 & 0.17 & 2.39 \end{bmatrix},$$

for which there is an arbitrage opportunity in the direction

$$w = (0.71, 0.62, 0.33).$$

We consider the regularization function $r(R) = \|(R - R_0)/R_0\|_1$, where / is meant elementwise. After running algorithm 3.1, the arbitrage-free return matrix is

$$R_{\text{final}} = \begin{bmatrix} 0.05 & 1.71 & -0.9 \\ 0.08 & 0.42 & -1.09 \\ 0.18 & -0.31 & 1.27 \\ 0.81 & -1.22 & 0.27 \\ -0.97 & 0.17 & 2.37 \end{bmatrix}, \quad R_{\text{final}} - R_0 = \begin{bmatrix} 0 & -0.03 & -0.02 \\ 0 & -0.04 & -0.08 \\ 0 & 0 & -0.02 \\ -0.09 & -0.05 & 0 \\ -0.04 & 0 & -0.03 \end{bmatrix},$$

and $r(R_{\text{final}}) = 0.142$.

5 Related work

Irreducible infeasible subsets An irreducible infeasible subset (IIS) of an optimization problem is a subset of the constraints that are by themselves infeasible, but any proper subset of them is feasible. Carver was the first to mention the concept of an IIS in (1922). Van Loon (1981) and Greenberg (1987) were among the first to recognize IIS in linear programs (LPs), but did not propose practical methods for finding them; Chinneck and Dravnieks (1991) were the first to provide practical methods for finding IIS in LPs. Once IISs have been identified, various methods can be used to remove constraints until feasibility is obtained. Some notable examples of methods to prune constraints include the deletion filter (Chinneck and Dravnieks 1991), the additive method (Tamiz et al. 1995, 1996), the elastic

filter (Chinneck and Dravnieks 1991), the reciprocal filter (Chinneck 1997), and the sensitivity filter (Chinneck and Dravnieks 1991). For an excellent survey of IISs and how to use them to repair infeasible optimization problems, as well as applications, see (Chinneck 2007, Sect. 6) and the many references therein. IISs have also been applied to convex quadratic inequalities in Obuchowska (1998), to convex analytic inequalities in Obuchowska (1999), and to semidefinite systems in Kellner et al. (2019).

Maximum feasible subsystem The maximum feasible subsystem (MAX FS) of a set of constraints is the largest subset of the constraints that is feasible, introduced by Amaldi et al. (1999). MAX FS has some equivalent formulations, including the minimum unsatisfied linear relation problem (Amaldi 1994), and the minimum cardinality IIS set-covering problem (Chinneck 1996). [All of these problems are NP-hard; see, e.g., Sankaran (1993)]. The problem of finding the MAX FS for linear constraints can be formulated as a mixed-integer linear program Greenberg and Murphy (1991). Various heuristics also exist for this problem, e.g., based on IISs Chinneck (1996) and branch-and-cut (Pfetsch 2008). For a survey of MAX FS problems and solution methods, see, e.g., (Chinneck 2007, Sect. 7) and (Pfetsch 2003, Sect. 1).

Optimal repair IIS and MAX FS, the two aforementioned methods, both try to make the optimization problem feasible by removing constraints. In many cases, however, the goal is not to remove constraints, but to adjust parameters in the problem so that it becomes solvable (Chinneck 2007, Sect. 8). Also, removing or adjusting the constraints to obtain feasibility might inadvertently make the problem unbounded, so we have to assure that the problem becomes both feasible and bounded.

Problem (2) is often tractable when, in its conic representation (6), A is constant, and b and c are affine functions of θ. (This is not the case in any of our examples.) In this case, problem (2) can be expressed as a single convex problem, as noted by Roodman (1979). In the case where the cones are products of the nonnegative reals, the resulting problem is immediately convex, while the more general case requires some care (see "Appendix"). Similar ideas have been applied to the more difficult problem of adjusting the coefficient matrix in LPs Amaral et al. (2008), and to repairing polynomial optimization problems (Gambella et al. 2019) (for more examples see (Chinneck 2007, Sect. 8) and the references therein). To the best of our knowledge, our paper is the first to consider automatic repair of convex optimization problems, allowing any parameter to be adjusted and guaranteeing both feasibility and boundedness.

Software Various software packages have been developed for IIS, MAX FS, and optimal repair, and are part of most numerical solvers. One of the earliest is ANALYZE by (Greenberg 1987), which is an interactive system for analyzing and modifying linear programming models; more recent software packages include PERUSE (Kurator and O'Neill 1980) and MProbe (Chinneck 2001). Implementations of simple optimal repair methods for the righthand side of constraints in LPs can be found in the solvers MOSEK (2020, Sect. 14.2), GUROBI (2019), and CPLEX (2016, Sect. 34).

Acknowledgements S. Boyd is an Engineering Subject Editor for the Optimization and Engineering journal. S. Barratt is supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-1656518.

Compliance with ethical standards

Conflicts of interest The authors declare that they have no conflict of interest.

Appendix: Convex formulation

In the case that A is a constant while b and c are affine functions of θ, we can write (9) as an equivalent convex optimization problem. In the linear case (i.e., when $\mathcal{K} = \mathbf{R}^n_+$), we can simply drop the strong duality requirement (which always holds in this case) and express (9) as

$$\begin{array}{ll} \text{minimize} & r(\theta) \\ \text{subject to} & Ax + s = b(\theta) \\ & A^T y + c(\theta) = 0 \\ & s \in \mathcal{K}, \quad y \in \mathcal{K}^*. \end{array}$$

For more general cones $\mathcal{K}$ (such as, e.g., the second order cone), a sufficient condition for strong duality is that there exist a feasible point in the interior of the cone. We can write this as, for example,

$$\begin{array}{ll} \text{minimize} & r(\theta) \\ \text{subject to} & Ax + s = b(\theta) \\ & A^T y + c(\theta) = 0 \\ & s \in \mathbf{int}\, \mathcal{K}, \quad y \in \mathcal{K}^*. \end{array} \tag{14}$$

(We could similarly constrain $y \in \mathbf{int}\, \mathcal{K}^*$ and $s \in \mathcal{K}$.)

In general, optimizing over open constraint sets is challenging and these problems may not even have an optimal point, but, in practice (and for well-enough behaved r, e.g., r continuous) we can approximate the true optimal value of (6) by approximating the open set $\mathbf{int}\, \mathcal{K}$ as a sequence of closed sets $\mathcal{K}_\varepsilon \subseteq \mathbf{int}\, \mathcal{K}$ such that $\mathcal{K}_\varepsilon \to \mathbf{int}\, \mathcal{K}$ as $\varepsilon \downarrow 0$.

References

Agrawal A, Amos B, Barratt S, Boyd S, Diamond S, Kolter J.Z (2019a) Differentiable convex optimization layers. In: Advances in neural information processing systems, pp 9558–9570

Agrawal A, Barratt S, Boyd S, Busseti E, Moursi W (2019b) Differentiating through a cone program. J Appl Numer Optim 1(2):107–115

Amaldi E (1994) From finding maximum feasible subsystems of linear systems to feedforward neural network design. Ph.D. thesis, Citeseer

Amaldi E, Pfetsch M, Trotter L (1999) Some structural and algorithmic properties of the maximum feasible subsystem problem. In: International conference on integer programming and combinatorial optimization. Springer, pp 45–59

Amaral P, Júdice J, Sherali H (2008) A reformulation–linearization–convexification algorithm for optimal correction of an inconsistent system of linear constraints. Comput Oper Res 35(5):1494–1509
Barratt S, Boyd S (2019) Least squares auto-tuning. arXiv preprint arXiv:1904.05460
Ben-Tal A, Nemirovski A (2001) Lectures on modern convex optimization: analysis, algorithms, and engineering applications, vol 2. SIAM
Boyd S, Vandenberghe L (2004) Convex optimization. Cambridge University Press, Cambridge
Carver W (1922) Systems of linear inequalities. Ann Math 212–220
Chinneck J (1996) An effective polynomial-time heuristic for the minimum-cardinality IIS set-covering problem. Ann Math Artif Intell 17(1):127–144
Chinneck J (1997) Finding a useful subset of constraints for analysis in an infeasible linear program. INFORMS J Comput 9(2):164–174
Chinneck J (2001) Analyzing mathematical programs using MProbe. Ann Oper Res 104(1–4):33–48
Chinneck J (2007) Feasibility and infeasibility in optimization: algorithms and computational methods, vol 118. Springer, Berlin
Chinneck J, Dravnieks E (1991) Locating minimal infeasible constraint sets in linear programs. ORSA J Comput 3(2):157–168
Diamond S, Boyd S (2016) CVXPY: a Python-embedded modeling language for convex optimization. J Mach Learn Res 17(83):1–5
Gambella C, Marecek J, Mevissen M (2019) Projections onto the set of feasible inputs and the set of feasible solutions. In: Allerton conference on communication, control, and computing. IEEE, pp 937–943
Greenberg H (1987) ANALYZE: a computer-assisted analysis system for linear programming models. Oper Res Lett 6(5):249–255
Greenberg H (1987) Computer-assisted analysis for diagnosing infeasible or unbounded linear programs. In: Computation mathematical programming. Springer, pp 79–97
Greenberg H, Murphy F (1991) Approaches to diagnosing infeasible linear programs. ORSA J Comput 3(3):253–261
GUROBI Optimization (2019) Gurobi optimizer reference manual
IBM (2016) IBM ILOG CPLEX optimization studio CPLEX user's manual
Karp R (1972) Reducibility among combinatorial problems. In: Complexity of computer computations, pp 85–103
Kellner K, Pfetsch M, Theobald T (2019) Irreducible infeasible subsystems of semidefinite systems. J Optim Theory Appl 181(3):727–742
Kurator W, O'Neill R (1980) PERUSE: an interactive system for mathematical programs. ACM Trans Math Softw 6(4):489–509
Martinet B (1970) Brève communication. régularisation d'inéquations variationnelles par approximations successives. ESAIM: Mathematical Modelling and Numerical Analysis-Modélisation Mathématique et Analyse Numérique 4(R3):154–158
MOSEK Aps (2020) MOSEK optimizer API for Python. https://docs.mosek.com
Nesterov Y (2013) Gradient methods for minimizing composite functions. Math Program 140(1):125–161
Obuchowska W (1998) Infeasibility analysis for systems of quadratic convex inequalities. Eur J Oper Res 107(3):633–643
Obuchowska W (1999) On infeasibility of systems of convex analytic inequalities. J Math Anal Appl 234(1):223–245
O'Donoghue B, Chu E, Parikh N, Boyd S (2016) Conic optimization via operator splitting and homogeneous self-dual embedding. J Optim Theory Appl 169(3):1042–1068
Parikh N, Boyd S (2014) Proximal algorithms. Found Trends® Optim 1(3):127–239. https://doi.org/10.1561/2400000003
Pfetsch M (2003) The maximum feasible subsystem problem and vertex-facet incidences of polyhedra. Ph.D. thesis
Pfetsch M (2008) Branch-and-cut for the maximum feasible subsystem problem. SIAM J Optim 19(1):21–38
Roodman G (1979) Note–post-infeasibility analysis in linear programming. Manage Sci 25(9):916–922
Sankaran J (1993) A note on resolving infeasibility in linear programs by constraint relaxation. Oper Res Lett 13(1):19–20

Tamiz M, Mardle S, Jones D (1995) Resolving inconsistency in infeasible linear programmes. Technical report, School of Mathematical Studies, University of Portsmouth, UK
Tamiz M, Mardle S, Jones D (1996) Detecting IIS in infeasible linear programmes using techniques from goal programming. Comput Oper Res 23(2):113–119
Van Loon JNM (1981) Irreducibly inconsistent systems of linear inequalities. Eur J Oper Res 8(3):283–288