Metric Selection in Fast Dual Forward Backward Splitting

Pontus Giselsson[*] and Stephen Boyd[†]

April 12, 2016

Abstract

The performance of fast forward-backward splitting, or equivalently fast proximal gradient methods, depends on the conditioning of the optimization problem data. This conditioning is related to a metric that is defined by the space on which the optimization problem is stated; selecting a space on which the optimization data is better conditioned improves the performance of the algorithm. In this paper, we propose several methods, with different computational complexity, to find a space on which the algorithm performs well. We evaluate the proposed metric selection procedures by comparing the performance to the case when the Euclidean space is used. For the most ill-conditioned problem we consider, the computational complexity is improved by two to three orders of magnitude. We also report comparable to superior performance compared to state-of-the-art optimization software.

1 Introduction

Fast gradient methods have been around since the early 80's when the seminal paper [37] was published. The algorithm in [37] is applicable to unconstrained smooth optimization problems and has since been extended and generalized in various directions. In [38], new acceleration schemes were presented as well as fast gradient methods for constrained optimization. In [39], smoothing techniques for nonsmooth problems are presented. Fast proximal gradient methods, or equivalently fast forward-backward splitting methods, that solve composite convex optimization problems of the form

$$\text{minimize} \quad f(x) + g(x) \tag{1}$$

where f is required to be smooth, are proposed in [40, 2]. In [47], generalizations and unifications of many fast forward-backward splitting methods are presented.

[*]Department of Automatic Control, Lund University. Email: `pontusg@control.lth.se`.
[†]Electrical Engineering Department, Stanford University. Email: `boyd@stanford.edu`.

The smooth part of the composite objective function, f in (1), is in fast forward-backward splitting approximated by the r.h.s. of

$$f(x) \leq f(y) + \langle \nabla f(y), x - y \rangle + \tfrac{\beta}{2}\|x - y\|^2 \tag{2}$$

where the norm and inner-product are given by the space on which the problem is defined. The condition that (2) holds for all x and y is referred to as f being β-smooth. Since the r.h.s. of the smoothness condition (2) is the only information the algorithm has about the smooth function, the smaller the gap in (2) (i.e. the better the r.h.s. of (2) approximates f), the better the performance of the algorithm is likely to be. In this paper, we show how to select a space (or metric, we will use these notions interchangeably since the metric defines the space) on which the fast forward-backward splitting method performs well, when solving the dual of strongly convex composite optimization problems. The spaces we consider are Euclidean spaces with inner product $\langle x, y \rangle = x^T y$ and scaled norm $\|x\|_K = \sqrt{x^T K x}$, where K is a positive definite metric matrix. We show how to select the metric K such that the gap in (2) for the smooth part of the dual problem is minimized. Using this metric in the algorithm often leads to improved performance compared to using the Euclidean metric with $K = I$.

Recently, [43, 42] proposed to use fast dual forward-backward splitting for embedded model predictive control. They apply fast forward-backward splitting with the standard Euclidean metric on two different dual problems. We show how these algorithms can be improved by choosing a metric that reduces the gap in (2). The performance improvement is confirmed by applying the methods to a pitch control problem in an AFTI-16 aircraft. This benchmark has previously been studied in [25, 3] and is a challenging problem for first order methods since it is fairly ill-conditioned. We report computation time improvements by two to three orders of magnitude. Besides this, we also compare the performance to the ADMM-based (see [6] for more on ADMM - the alternating direction method of multipliers) algorithm in [41, 24]. We also compare our algorithms, that are implemented in the MATLAB toolbox QPgen [19], to several other toolboxes and software for embedded optimization, namely: DuQuad, see [34], FiOrdOs, see [48], FORCES, see [12], CVXGEN, see [30], qpOASES, see [14] and the MPT Toolbox, see [22]. Finally, we also compare to the general commercial QP-solver MOSEK, see [31]. QPgen, with the proposed fast dual forward-backward splitting method, performs little to much better than the other methods on this example.

Fast dual forward-backward splitting can also be used for distributed optimization when the objective to be minimized is separable. In the context of gradient methods, this has been known since [13, 10, 4]. Recently such approaches have been proposed for distributed model predictive control (DMPC) [36, 11, 21, 16], and resource optimization over networks [15, 1, 33]. Often, centralized coordination is needed when selecting the step-size for the gradient-step. This is relaxed in [1], where the authors noted that the smooth part of the dual problem consists of a sum of local functions. Each of these can compute its own step-size, share with its neighbors and sum, to get a fully distributed step-size selection. This procedure can be augmented by the results of this paper to select local metrics instead of step-sizes. This leads to more efficient algorithms which is confirmed by a numerical example which shows improvements of about one order of magnitude. We also compare the performance to the dual Newton conjugate gradient method in [27], which is outperformed in our numerical example.

This paper unifies and extends the conference publications [17, 18, 20].

2 Notation and Preliminaries

We denote by $\mathbb{R}$, $\mathbb{R}^n$, $\mathbb{R}^{m\times n}$, the sets of real numbers, column vectors, and matrices. We use notation $(x, y, z) := [x^T \; y^T \; z^T]^T$ for stacked real column vectors. We also use notation $\overline{\mathbb{R}} = \mathbb{R} \cup \{\infty\}$ for the extended real line. $\mathbb{S}^n \subseteq \mathbb{R}^{n\times n}$ is the set of symmetric matrices, and $\mathbb{S}^n_{++} \subseteq \mathbb{S}^n$, $[\mathbb{S}^n_{+}] \subseteq \mathbb{S}^n$, are the sets of positive [semi] definite matrices. We use Euclidean spaces with the standard inner product $\langle x, y\rangle = x^T y$ and different norms. When using the induced norm $\|x\| = \sqrt{\langle x, x\rangle}$, we get the standard Euclidean space. We also consider spaces $\mathbb{E}_H$ with Euclidean inner product and scaled norm $\|x\|_H = \sqrt{\langle x, Hx\rangle}$, where $H \in \mathbb{S}^n_{++}$. The dual space to $\mathbb{E}_H$ is denoted by $\mathbb{E}_H^*$. The dual norm to $\|y\|_H$ is $\|y\|_H^* = \max_x \{\langle y, x\rangle_2 \; : \; \|x\|_H = 1\} = \|y\|_{H^{-1}}$, i.e., $\mathbb{E}_H^* = \mathbb{E}_{H^{-1}}$. Further, the class of closed, proper, and convex functions $f \; : \; \mathbb{E}_H \to \overline{\mathbb{R}}$ is denoted by $\Gamma_0(\mathbb{E}_H)$. The conjugate function $f^* \; : \; \mathbb{E}_H^* \to \overline{\mathbb{R}}$ to $f \in \Gamma_0(\mathbb{E}_H)$ is defined as $f^*(y) = \sup_x \{\langle y, x\rangle - f(x)\}$. The adjoint operator to a bounded linear operator $\mathcal{A} \; : \; \mathbb{E}_H \to \mathbb{E}_K$ is denoted by $\mathcal{A}^* \; : \; \mathbb{E}_K^* \to \mathbb{E}_H^*$ and is defined as the unique operator that satisfies $\langle \mathcal{A}x, y\rangle = \langle \mathcal{A}^* y, x\rangle$ for all $x \in \mathbb{E}_H$ and $y \in \mathbb{E}_K^*$. Since the ambient space for $\mathbb{E}_H$ is the standard Euclidean space, we often denote the matrix that corresponds to the operator $\mathcal{A} \; : \; \mathbb{E}_H \to \mathbb{E}_K$ by $A \in \mathbb{R}^{m\times n}$. We use notation $I_{\mathcal{X}}$ for the indicator function for the set $\mathcal{X}$, and $I_{g(x)\leq 0}$ for the indicator function for the set $\mathcal{X} = \{x \mid g(x) \leq 0\}$.

A function $f \in \Gamma_0(\mathbb{E}_H)$ is *β-strongly convex* (w.r.t. $\mathbb{E}_H$) if $f - \frac{\beta}{2}\|\cdot\|_H^2$ is convex. A function $f \in \Gamma_0(\mathbb{E}_H)$ is *β-smooth* (w.r.t. $\mathbb{E}_H$) if it is differentiable and $\frac{\beta}{2}\|\cdot\|_H^2 - f$ is convex. An equivalent characterization of β-smoothness w.r.t. $\mathbb{E}_H$ is that

$$f(x) \leq f(y) + \langle \nabla f(y), x - y\rangle + \tfrac{\beta}{2}\|x - y\|_H^2 \tag{3}$$

holds for all $x, y \in \mathbb{E}_H$. As seen in the following proposition, these notions are related through the conjugate function.

Proposition 1 *Suppose that $f \in \Gamma_0(\mathbb{E}_H)$. Then the following are equivalent:*

(i) f is β-strongly convex (w.r.t. $\mathbb{E}_H$).

(ii) f^* is $\frac{1}{\beta}$-smooth (w.r.t. $\mathbb{E}_H^* = \mathbb{E}_{H^{-1}}$).

A proof to this can be found, e.g., in [49, Proposition 3.5.3].

3 Problem formulation

We consider optimization problems of the form

$$\begin{array}{ll} \text{minimize} & f(x)+g(y) \\ \text{subject to} & \mathcal{A}x = y \end{array} \tag{4}$$

and assume that the following assumption holds throughout the paper:

Assumption 1

(a) The extended valued function $f \in \Gamma_0(\mathbb{E}_H)$ is 1-strongly convex (w.r.t. $\mathbb{E}_H$).

(b) The extended valued function $g \in \Gamma_0(\mathbb{E}_K)$.

(c) $\mathcal{A} : \mathbb{E}_H \to \mathbb{E}_K$ is a bounded linear operator.

Remark 1 *A function that satisfies Assumption 1(a) is $f(x) = \frac{1}{2}x^T Hx + \hat{f}$ where $H \in \mathbb{S}^n_{++}$ and $\hat{f} \in \Gamma_0(\mathbb{E}_H)$. Since $\hat{f}$ (and g) are allowed to be extended valued, they can, e.g., be indicator functions for nonempty, closed, and convex constraint sets. Further, the operator $\mathcal{A} : \mathbb{E}_H \to \mathbb{E}_K$ has an associated matrix $A : \mathbb{R}^n \to \mathbb{R}^m$ that satisfies $\mathcal{A}x = Ax$ for all $x \in \mathbb{R}^n$.*

To arrive at the dual problem, we introduce Lagrange multipliers $\mu \in \mathbb{E}_{K^{-1}}$, to get Lagrangian

$$L(x, y, \mu) = f(x) + g(y) + \langle \mathcal{A}x - y, \mu \rangle.$$

By minimizing the Lagrangian over x, and y, we get

$$\begin{aligned} \inf_{x,y} L(x,y,\mu) &= \inf_x \{\langle \mathcal{A}^*\mu, x\rangle + f(x)\} + \inf_y \{\langle -y, \mu\rangle + g(y)\} \\ &= -\sup_x \{\langle -\mathcal{A}^*\mu, x\rangle - f(x)\} - \sup_y \{\langle \mu, y\rangle - g(y)\} \\ &= -f^*(-\mathcal{A}^*\mu) - g^*(\mu). \end{aligned}$$

Negating this, we get the negated dual problem to (4) (see, e.g, [44, §31] for more details):

$$\text{minimize } d(\mu) + g^*(\mu) \tag{5}$$

where

$$d(\mu) := f^*(-\mathcal{A}^*\mu). \tag{6}$$

Note that $d, g^* \in \Gamma_0(\mathbb{E}_{K^{-1}})$. The efficiency of solving this dual problem using fast forward backward splitting is highly dependent on which metric that is used. This paper is about choosing metrics to make the algorithm perform well.

4 Dual problem properties

Here, we will present tight characterizations of the smooth part of the dual problem, i.e. d in (5). These characterizations will later guide us in choosing metric for the dual forward-backward splitting scheme.

From, e.g., [44, Theorem 23.5] combined with the chain rule, we know that d is differentiable with

$$\nabla d(\mu) = -\mathcal{A}^* x^\star(\mu) \tag{7}$$

where

$$x^\star(\mu) = \underset{x}{\operatorname{argmin}} \left\{ \langle \mathcal{A}^* \mu, x \rangle + f(x) \right\}. \tag{8}$$

It is also well known, see e.g. [39, Theorem 1], that ∇d is Lipschitz continuous with constant $\|\mathcal{A}^*\|^2 = \|\mathcal{A}\|^2$ (since f is 1-strongly convex (w.r.t. $\mathbb{E}_H$) due to Assumption 1), where the norm is the operator norm. (This also follows directly from Proposition 1, i.e., from [49, Proposition 3.5.3], and the Cauchy-Schwarz inequality.) Since $\mathcal{A} : \mathbb{E}_H \to \mathbb{E}_K$, the norm $\|\mathcal{A}\|$ is defined by

$$\begin{aligned}
\|\mathcal{A}\| &= \max_x \left\{ \|\mathcal{A}x\|_K \ : \ \|x\|_H \leq 1 \right\} \\
&= \left\{ \|K^{1/2} A x\|_2 \ : \ \|H^{1/2} x\|_2 \leq 1 \right\} \\
&= \left\{ \|K^{1/2} A H^{-1/2} v\|_2 \ : \ \|v\|_2 \leq 1 \right\} \\
&= \|K^{1/2} A H^{-1/2}\|_2
\end{aligned}$$

where $\|\cdot\|_2$ denotes the standard induced Euclidean norm and A is the Euclidean matrix representation of $\mathcal{A}$. By defining d on $\mathbb{E}_I$, i.e., by choosing $K = I$, we get that $d \in \Gamma_0(\mathbb{E}_I)$ is Lipschitz continuous with constant $\|AH^{-1/2}\|_2^2 = \|AH^{-1}A^T\|_2$. This is exactly the Lipschitz constant provided in [43] which implies that

$$d(\mu) \leq d(\nu) + \langle \nabla d(\nu), \mu - \mu \rangle_2 + \tfrac{\|AH^{-1}A^T\|_2}{2} \|\mu - \nu\|_2^2 \tag{9}$$

holds for all $\mu, \nu \in \mathbb{R}^m$. This upper bound can be improved by defining d on $\mathbb{E}_{K^{-1}}$ with $K = (AH^{-1}A^T)^{-1}$ (where we have implicitly assumed that A has full row rank). This implies that d is 1-smooth w.r.t. $\mathbb{E}_{AH^{-1}A^T}$, i.e., that

$$d(\mu) \leq d(\nu) + \langle \nabla d(\nu), \mu - \mu \rangle_2 + \tfrac{1}{2} \|\mu - \nu\|_{AH^{-1}A^T}^2 \tag{10}$$

holds for all $\mu, \nu \in \mathbb{E}_{AH^{-1}A^T}$. This is obviously a tighter characterization of d than (9).

Remark 2 *We improve the upper bound on d by defining it on a different space. It is straight-forward to verify that this does not influence the shape of the function d itself, only the bound is improved.*

When selecting $K = (AH^{-1}A^T)^{-1}$, we implicitly assume that A has full row rank. In the following result, we show that (10) also holds when A is not full row rank.

Proposition 2 *Suppose that Assumption 1 holds. Then $d \in \Gamma_0(\mathbb{E}_{K^{-1}})$ as defined in* (6) *satisfies*

$$d(\mu) \le d(\nu) + \langle \nabla d(\nu), \mu - \nu \rangle + \tfrac{1}{2}\|\mu - \nu\|_L^2 \tag{11}$$

for any $L \succeq AH^{-1}A^T$ and for all $\mu, \nu \in \mathbb{E}_{K^{-1}}$, where $A \in \mathbb{R}^{m\times n}$ is the matrix representation of $\mathcal{A}$.

Proof. Since Assumption 1 states that f is 1-strongly convex w.r.t. $\mathbb{E}_H$, Proposition 1, gives that f^* is 1-smooth w.r.t. $\mathbb{E}_{H^{-1}}$. Thus, (3) holds for f^* for any $x, y \in \mathbb{E}_{H^{-1}}$. Further, since f^* is independent of the norm on the space (it only depends on the inner product) (3) holds for f^* also for any $x, y \in \mathbb{R}^n$. Especially, let $x = -A^T\mu$ and $y = -A^T\nu$ to get

$$\begin{aligned} d(\mu) &= f^*(-A^T\mu) \\ &\le d(\nu) + \langle \nabla f^*(-A^T\nu), -A^T(\mu - \nu)\rangle + \tfrac{1}{2}\|A^T(\mu - \nu)\|_{H^{-1}}^2 \\ &= d(\nu) + \langle -A\nabla f^*(-A^T\nu), \mu - \nu\rangle + \tfrac{1}{2}\|\mu - \nu\|_{AH^{-1}A^T}^2 \\ &= d(\nu) + \langle \nabla d(\nu), \mu - \nu\rangle + \tfrac{1}{2}\|\mu - \nu\|_{AH^{-1}A^T}^2. \end{aligned}$$

Since the the inequality holds for $AH^{-1}A^T$, it also holds for any $L \succeq AH^{-1}A^T$. This concludes the proof. $\square$

Next, we show that for many interesting functions f, the bound provided in Proposition 2 on d is indeed tight. Essentially, we show that if the strong convexity bound on the primal is tight, so is the smoothness bound on the dual. A proof is provided in Appendix A.

Proposition 3 *Suppose that Assumption 1 holds and that there exists a full-dimensional ball $\mathcal{B}_r^n(x^\star(\bar{\mu}))$ centered around $x^\star(\bar{\mu})$ for some $\bar{\mu} \in \mathbb{E}_{K^{-1}}$ on which $f - \frac{1}{2}\|\cdot\|_H^2$ is linear. Then no matrix $L \not\succeq AH^{-1}A^T$ exists such that $d : \mathbb{E}_{K^{-1}} \to \mathbb{R}$ as defined in* (6) *satisfies* (11) *for all $\mu, \nu \in \mathbb{E}_{K^{-1}}$.*

The assumptions in Proposition 3 are met, for instance, if A has full column rank and $f(x) = \frac{1}{2}\|x\|_H^2 + h(x)$ where h is the indicator function for a closed and convex constraint set with nonempty interior, the 1-norm, a linear function, or any other function that is linear on a convex subset with nonempty interior. For these examples, Proposition 3 indeed gives the best obtainable quadratic majorizer of d. However, for f being a quadratic plus the indicator functions for an affine subspace $I_{Bx=b}$, the assumptions in Proposition 3 are not met since the interior of $I_{Bx=b}$ is empty (except for in trivial cases). In the following proposition we take care of that specific case. A proof is provided in Appendix B.

Proposition 4 *Assume that $f(x) = \frac{1}{2}x^THx + \xi^Tx + I_{Bx=b}(x)$ with $H \in \mathbb{S}_+^n$, $\xi \in \mathbb{R}^n$, $B \in \mathbb{R}^{p\times n}$, and $b \in \mathbb{R}^p$. Further assume that $x^THx > 0$ whenever $x \neq 0$ and $Bx = 0$, i.e., that H is positive definite on the null-space of B. Then $d : \mathbb{E}_{K^{-1}} \to \mathbb{R}$ satisfies*

$$d(\mu) \leq d(\nu) + \langle \nabla d(\nu), \mu - \nu\rangle + \tfrac{1}{2}\|\mu - \nu\|_L^2 \tag{12}$$

for any $L \succeq AM_{11}A^T$ and all $\mu, \nu \in \mathbb{E}_{K^{-1}}$, where

$$\begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix} = \begin{bmatrix} H & B^T \\ B & 0 \end{bmatrix}^{-1}. \tag{13}$$

Further, no matrix $L \not\succeq AM_{11}A^T$ exists such that (12) *holds for all $\mu, \nu \in \mathbb{E}_{K^{-1}}$.*

Note that the results in Propositions 2, 3, and 4 are independent on which space the function d is defined, i.e., they are independent of K. The results merely relate the curvature of d to $AH^{-1}A^T$.

5 Fast dual forward-backward splitting

We apply fast dual forward-backward splitting to solve the dual problem (5), which is defined on $\mathbb{E}_{K^{-1}}$. In this section, we introduce the notation $L = K^{-1}$, i.e., the dual problem is defined on $\mathbb{E}_{K^{-1}} = \mathbb{E}_L$. The algorithm is

$$\begin{aligned} \nu^k &= \mu^k + \alpha_k(\mu^k - \mu^{k+1}) \\ \mu^{k+1} &= \text{prox}_{g^*}(\nu^k - L^{-1}\nabla d(\nu^k)) \end{aligned}$$

where α_k grows in a specific way, see [38, 2] for details, and the prox (backward) step is defined as

$$\text{prox}_{g^*}(z) := \underset{\mu}{\text{argmin}} \left\{ g^*(\mu) + \tfrac{1}{2}\|\mu - z\|_L^2 \right\}.$$

The μ^{k+1}-update can equivalently be written as (expand the square, remove constant terms and add $d(\nu^k)$ which does not affect the minimizer):

$$\underset{\mu}{\text{argmin}} \left\{ d(\nu^k) + \langle \nabla d(\nu^k), \mu - \nu^k\rangle + \tfrac{1}{2}\|\mu - \nu^k\|_L^2 + g^*(\mu) \right\}.$$

We see that d is approximated by the r.h.s. of the smoothness definition in (3), i.e., by

$$d(\nu^k) + \langle \nabla d(\nu^k), \mu - \nu^k\rangle + \tfrac{1}{2}\|\mu - \nu^k\|_L^2. \tag{14}$$

Indeed, the algorithm converges if d is 1-smooth w.r.t. $\mathbb{E}_L$ where L is used in the algorithm, see [50]. A bound on the convergence rate is (see [50])

$$D(\mu^k) - D(\mu^\star) \leq \frac{2\|\mu^\star - \mu^0\|_L^2}{(k+1)^2} \tag{15}$$

where $D = d + g^*$ and $\mu^\star$ is a solution to (5). Also, convergence rates for the primal iterates as well as the primal infeasibility can be obtained, see [42, 35, 32].

Remark 3 *The constant in the rate bound in* (15) *can be compared to the constant in the rate bound for standard fast dual forward-backward splitting used, e.g., in [43], which is*

$$D(\mu^k) - D(\mu^\star) \leq \frac{2\beta\|\mu^\star - \mu^0\|_2^2}{(k+1)^2} \tag{16}$$

where $\beta = \|AH^{-1}A^T\|_2$ *is a (tight) Lipschitz constant for* d*, see* (9)*. Comparing this to* (15) *where we can choose* $L = AH^{-1}A^T$*, see Proposition 2, we conclude that the constant in* (15) *can be significantly smaller than the constant in* (16)*, especially for ill-conditioned problems.*

Remark 4 *From [20] it follows that forward-backward splitting algorithm applied on* $\mathbb{E}_{K^{-1}}$ *is equivalent to applying forward-backward splitting on the Euclidean space* $\mathbb{R}^m$ *to the dual of the preconditioned problem*

$$\begin{array}{ll} \textit{minimize} & f(x) + g(y) \\ \textit{subject to} & DAx = Dy \end{array}$$

where $K = D^TD$*. We can also define the dual problem on the space with inner product* $\langle x, y\rangle = x^TKy$ *and induced norm* $\|x\| = \sqrt{x^TKx}$*. When applying the forward-backward splitting algorithm on this Hilbert space, we get another equivalent algorithm (note that the metric* K *is inverted compared to* $\mathbb{E}_{K^{-1}}$*). These equivalent approaches are, however, not further discussed here due to space limitations and clarity of exposition.*

The important question how to choose $L = K^{-1}$ remains. As mentioned, the smooth part of the dual problem, d in (6), is approximated by a quadratic majorizor, namely the r.h.s. of the smoothness definition (3). The tighter this quadratic majorizer is, the more accurate function approximation is used in the algorithm, and consequently the faster the convergence of the algorithm is likely to be. For example, if the majorizer is tight (i.e., (3) holds with equality for all x, y), the algorithm converges in one iteration. This can be seen from the μ^{k+1}-update

$$\begin{aligned} \mu^{k+1} &= \underset{\mu}{\operatorname{argmin}} \left\{d(\nu^k) + \langle \nabla d(\nu^k), \mu - \nu^k\rangle + \tfrac{1}{2}\|\mu - \nu^k\|_L^2 + g^*(\mu)\right\} \\ &= \underset{\mu}{\operatorname{argmin}} \left\{d(\mu) + g^*(\mu)\right\} \end{aligned}$$

which reduces to solving the original problem (5). Proposition 2 suggests that $L = AH^{-1}A^T$ is a good metric for the algorithm, since the r.h.s. of (3) would (in many cases) be the best quadratic majorizer of d, see Proposition 3. In most cases, however, it is not advisable to let $L = AH^{-1}A^T$ since the prox operation of g^* could become computationally too expensive. For instance, if g^* is separable, using a non-diagonal L would typically increase the computational cost in each iteration more than what is saved by the reduced number of iterations. Therefore, we should choose $L \approx AH^{-1}A^T$ with a structure of L

to keep the computational complexity of evaluating the prox operator low. We also need $L \succeq AH^{-1}A^T$ to guarantee convergence of the algorithm. By letting an invertible matrix E satisfy $L = (E^TE)^{-1}$, these objectives can be formulated as choosing E such that $I \approx EAH^{-1}A^TE^T$ and $I \succeq EAH^{-1}A^TE^T$. A natural choice is then to minimize $\kappa(EAH^{-1}A^TE^T)$ subject to $I \succeq EAH^{-1}A^TE^T$, where κ denotes the condition number. However, if A does not have full row rank, or if the objective is to satisfy $L \approx AM_{11}A^T$ (when the assumptions in Proposition 4 hold), the matrix APA^T (with $P = H^{-1}$ or $P = M_{11}$) is rank deficient and does not have a well defined condition number. Then, the ratio between the largest and smallest non-zero eigenvalues of $EAPA^TE^T$ could be minimized instead. This is reasonable, since the zero eigenvalues of APA^T cannot be affected or made positive by pre- and post-multiplying APA^T with E and E^T respectively.

6 Computing the metric matrix

In this section, we show how to solve

$$\text{minimize } \frac{\lambda_1(EQE^T)}{\lambda_r(EQE^T)} \tag{17}$$

where λ_1 denotes the largest (non-zero) eigenvalue, λ_r denotes the smallest non-zero eigenvalue, and $Q \in \mathbb{S}^n_+$ is $Q = AH^{-1}A^T$ or $AM_{11}A^T$. We restrict the matrix E to be full, block-diagonal, or diagonal, since then also $L = (E^TE)^{-1}$ has the same structure. We denote by $\mathcal{E}$ any of these structural constraints. Besides showing how to solve (17) exactly, we also present heuristic methods to (hopefully) reduce the (pseudo) condition number in (17).

6.1 Exact condition number minimization

First, we show how to solve (17) exactly. We consider two different cases; Q positive definite, and Q positive semi-definite. In [5], it has been shown that minimizing the ratio between the largest and smallest eigenvalues of $(FRE^T)^T(FRE^T)$ by selecting E and F can be posed as a quasi convex optimization problem. Here, we are interested in the case where either $E = I$ or $F = I$. For these cases, (17) can be solved by convex optimization. Before we show this, we state the following lemma which follows directly from the definition of singular values.

Lemma 1 *For any matrix $\Phi \in \mathbb{R}^{m\times n}$, the non-zero eigenvalues of $\Phi^T\Phi$ are the same as the non-zero eigenvalues of $\Phi\Phi^T$.*

6.1.1 The positive definite case

Here, we assume that $Q \in \mathbb{S}^n_{++}$, which occurs, for instance, if $Q = AH^{-1}A^T$, where $H \in \mathbb{S}^n_{++}$ and $A \in \mathbb{R}^{m\times n}$ has full row rank.

Proposition 5 *Assume that $Q \in \mathbb{S}^m_{++}$. Then a matrix $E \in \mathcal{E}$ that minimizes the ratio* (17) *can be computed by solving the convex semi-definite program*

$$\begin{array}{ll} \textit{minimize} & t \\ \textit{subject to} & tQ \succeq L \\ & Q \preceq L \\ & L \in \mathcal{E} \end{array} \tag{18}$$

where $L = (E^T E)^{-1}$. Further, $L \succeq Q$.

Proof. Since Q has full rank, (17) is the condition number. Thus, according to [5, Section 3.1], (18) can be solved in order to minimize (17). Since the cost and constraints in (18) are all convex, this is a convex optimization problem. Further, the second constraint implies that $L \succeq Q$. □

The condition number minimization problem (17) is also investigated in [29], where they search for E directly using a convex relaxation of the nonconvex constraint $EQE^T \succeq \frac{1}{t}I$. It is shown that the convex relaxation is tight if E is diagonal. Therefore the approach in [29] is slightly more restrictive than our setting since we allow also for block-diagonal structures.

6.1.2 The positive semi-definite case

Here, we assume that Q is positive semi-definite. This situation occurs, e.g., if $Q = AH^{-1}A^T$ and $A \in \mathbb{R}^{m\times n}$ with $m > n$, or if $Q = AM_{11}A^T$.

Proposition 6 *Assume that $Q \in \mathbb{S}^m_{+}$ is factorized as $Q = R^T R$, where $R \in \mathbb{R}^{q\times n}$ has rank q. Then a matrix $E \in \mathcal{E}$ that minimizes the ratio* (17) *can be computed by solving the convex semi-definite program*

$$\begin{array}{ll} \textit{minimize} & -t \\ \textit{subject to} & RMR^T \preceq I \\ & RMR^T \succeq tI \\ & M \in \mathcal{E} \end{array} \tag{19}$$

where $M = (E^T E)$. Further $L = M^{-1} = (E^T E)^{-1} \succeq Q$.

Proof. Since RMR^T has full rank, we get from Lemma 1 and equalities $M = E^T E$ and $Q = R^T R$ that minimizing the condition number of RMR^T is equivalent to minimizing the ratio between the largest and smallest non-zero eigenvalues of EQE^T, i.e. equivalent to solving (17). From [5, Section 3.1], we get that (19) minimizes the condition number of RMR^T, i.e., it minimizes (17). Since the cost and constraints in (19) are all convex, this is a convex optimization problem. Further, the first inequality in (19) implies through Lemma 1 that $EQE^T \preceq I$, which is equivalent to that $L = (E^T E)^{-1} \succeq Q$. This concludes the proof. □

6.2 Heuristic 1 – Trace minimization

We also propose to use a trace minimization heuristic to reduce the (pseudo) condition number of EQE^T. Let $L = (E^TE)^{-1}$ to get

$$\begin{array}{ll} \text{minimize} & \operatorname{trace} L \\ \text{subject to} & Q \preceq L \\ & L \in \mathcal{E} \end{array}$$

Also this is a semi-definite program and therefore restricted to small-scale problems.

6.3 Heuristic 2 – Equilibration

In symmetric equilibration, given a matrix $Q \in \mathbb{R}^{m\times n}$, the objective is to find a positive and diagonal matrix $E \in \mathbb{R}^{n\times n}$ such that all rows and columns of EQE^T have the same norm. This is a heuristic to reduce the condition number of EQE^T compared to Q, see [8] for an overview of (symmetric) equilibration and further references. There are no guarantees that the condition number is reduced, but in practice this is most often the case. Below, we present different methods to achieve symmetric equilibration in the 1-norm, 2-norm and ∞-norm. These methods do not guarantee that $I \succeq EQE^T$, which is required to get convergence of the fast dual forward-backward splitting method when metric $L = (E^TE)^{-1}$ is used. However, this is achieved by appropriately scaling E afterward using a norm computation.

6.3.1 Equilibration in 1-norm and 2-norm

For a symmetric matrix, the ith row and column are the same, hence also their norms. Therefore, we need only equilibrate either the rows or the columns. The 1-norm of row i of EQE^T is given by

$$\left\|[EQE^T]_{i,\cdot}\right\|_1 = \sum_{j=1}^m |E_{ii}Q_{ij}E_{jj}| = E_{ii}\sum_{j=1}^m |Q_{ij}|\,E_{jj}$$

since E is diagonal with $E_{ii} > 0$. Similarly, the squared 2-norm is given by

$$\left\|[EQE^T]_{i,\cdot}\right\|_2^2 = \sum_{j=1}^m (E_{ii}Q_{ij}E_{jj})^2 = E_{ii}^2\sum_{j=1}^m Q_{ij}^2E_{jj}^2.$$

Thus, by introducing the matrices $T_1 = |Q|$ (where $|\cdot|$ denotes element-wise absolute value) and $T_2 = (Q)^{(2)}$ (where $(\cdot)^{(2)}$ denotes element-wise square), and by letting $E = \operatorname{diag}(e)$, symmetric equilibration can be stated as finding E (and e) such that

$$ET_1e = \mathbf{1} \tag{20}$$

in the 1-norm case and

$$E^{(2)}T_2e^{(2)} = \mathbf{1} \tag{21}$$

in the 2-norm case. We treat these cases simultaneously by introducing $\widetilde{E} = \mathrm{diag}(\widetilde{e})$ and T that satisfies $\widetilde{E} = E$ and $T = T_1$ in the 1-norm case, and $\widetilde{E} = E^{(2)}$ and $T = T_2$ in the 2-norm case. The conditions (20) and (21) can then be written as

$$0 = T\widetilde{e} - \widetilde{E}^{-1}\mathbf{1}.$$

This is indeed the gradient of the function

$$\phi(\widetilde{e}) = \tfrac{1}{2}\widetilde{e}^T T\widetilde{e} - \sum_{i=1}^{n} \log(\widetilde{e}_i). \tag{22}$$

Since $\widetilde{e}^T T\widetilde{e} \geq \sum_i (T_{ii}\widetilde{e}_{ii}^2) \geq (\min_i T_{ii})\|\widetilde{e}\|_2^2$ for all $\widetilde{e} \in \mathrm{dom}\phi$, i.e. for all $\widetilde{e} > 0$, and since $-\log$ is convex, ϕ is convex on its domain. If in addition $\min_i T_{ii} > 0$ (which is the case we are interested in), then ϕ is strongly convex. Since $\phi(\widetilde{e}) < \infty$ for all $\widetilde{e} \in \mathrm{int}(\mathrm{dom}\phi)$ and since it is coercive, we conclude that ϕ has a unique minimizer $\widetilde{e}^\star \in \mathrm{int}(\mathrm{dom}\phi)$. This unique minimizer can be found in various ways.

One approach is to perform element-wise optimization and cycle through the elements until convergence. Another classic method is the (symmetric) Sinkhorn-Knopp algorithm, [46], which was originally developed to generate doubly stochastic matrices from positive matrices. The symmetric Sinkhorn-Knopp algorithm is given by the iteration

$$\widetilde{e}^{k+1} = (T\widetilde{e}^{k})^{-1}$$

where $(\cdot)^{-1}$ denotes element-wise reciprocal. This is known to converge, see [46] under some technical conditions on T, see [46, 8] for details. Also other equilibration methods exist, see [45, 8, 26]. Common for all these methods is that they are computationally very cheap and that two to five passes over the data are usually sufficient to obtain a close to equilibrated matrix.

6.3.2 Equilibration in ∞-norm

In ∞-norm equilibration of general symmetric matrices, the magnitude of the largest element in each row (or column) is set to 1. For positive semi-definite matrices $S \in \mathbb{S}_+^n$, we have $S_{ii} \geq 0$ and $\max_i S_{ii} \geq \max_{i\neq j} |S_{ij}|$, see [23, p. 398]. Therefore, if $S_{ii} > 0$ for all i, making $S_{ii} = 1$ gives an ∞-norm equilibrated matrix. For $S = EQE^T$ with Q positive semi-definite with positive diagonal, this scaling (which is also called Jacobi scaling) is obtained by letting $E_{ii} = 1/\sqrt{Q_{ii}}$, which is computationally very cheap.

7 Applications

We consider two quadratic programming formulations; one with no specific structure that we solve in two different ways, and one that has a separable structure that allows for distributed implementation. We show how the algorithms look when applied to solve these problems and we discuss how to select metric matrix $L = K^{-1}$.

7.1 Two QP splittings

Here, we consider the following quadratic program

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}x^THx + \xi^Tx \\ \text{subject to} & Bx = b \\ & \underline{d} \leq Cx \leq \bar{d} \end{array}$$

where $H \in \mathbb{S}^n_{++}$, $\xi \in \mathbb{R}^n$, $B \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, $C \in \mathbb{R}^{p\times n}$, and $\underline{d}, \bar{d} \in \mathbb{R}^p$. Below, we present two different splitting schemes for this problem.

7.1.1 QP splitting 1

In the first splitting, which has been used in [42] in the context of fast dual forward-backward splitting and in [41] in the context of ADMM, we set f and g in (4) to

$$\begin{aligned} f(x) &= \tfrac{1}{2}x^THx + \xi^Tx + I_{Bx=b}(x) \\ g(y) &= I_{\underline{d}\leq y\leq\bar{d}}(y) \end{aligned}$$

which gives equality constraint $Cx = y$. (Here, it is enough to assume that H is positive definite on the null-space of B.) We form the dual as in (5) and restrict $L = K^{-1}$ in the algorithm to be diagonal. After some simplification, the resulting fast dual forward-backward splitting algorithm on $\mathbb{E}_L = \mathbb{E}_{K^{-1}}$ becomes

$$\nu^k = \mu^k + \alpha_k(\mu^k - \mu^{k-1}) \tag{23}$$

$$x^k = \underset{x}{\operatorname{argmin}} \left\{ \tfrac{1}{2}x^THx + \xi^Tx + I_{Bx=b}(x) + \nu^TCx \right\} \tag{24}$$

$$\mu^k = \min\Big(\nu^k + L^{-1}(Cx^k - \underline{d}), \max\big(\nu^k + L^{-1}(Cx^k - \bar{d}), 0\big)\Big). \tag{25}$$

The restriction that L is diagonal implies that the prox-operation becomes a min-max operation, hence very cheap. The matrix L should be computed in accordance with the suggestions in Section 6. Specifically, according to Proposition 4, we need $L \succeq CM_{11}C^T$, where M_{11} is defined in (13). Obviously also $L \succeq CH^{-1}C^T$ holds since $M_{11} \preceq H^{-1}$.

Equation (24) can be efficiently implemented since it is an equality constrained quadratic problem. It can be solved by forming and storing M_{11} and M_{12} in (13), and reuse these in all iterations. Another option, that might be beneficial if $\left[\begin{smallmatrix} H & B^T \\ B & 0 \end{smallmatrix}\right]$ (from (13)) is sparse, is to compute and store a sparse LDL factorization of $\left[\begin{smallmatrix} H & B^T \\ B & 0 \end{smallmatrix}\right]$ and perform forward and backward substitution on the precomputed factors in each subsequent iteration.

7.1.2 QP splitting 2

The second splitting, which has been used in [43], is obtained by letting f and g in (4) be

$$f(x) = \tfrac{1}{2}x^T Hx + \xi^T x + I_{\underline{d}\leq Cx\leq\bar{d}}(x)$$
$$g(y) = I_{y=b}(y)$$

which gives equality constraint $Bx = y$. Without structural restrictions on $L = K^{-1}$, the resulting fast dual forward backward splitting method on $\mathbb{E}_L = \mathbb{E}_{K^{-1}}$ becomes after some simplification:

$$\nu^{k+1} = \mu^k + \alpha_k(\mu^k - \mu^{k-1}) \tag{26}$$
$$x^k = \underset{x}{\operatorname{argmin}}\left\{\tfrac{1}{2}x^T Hx + \xi^T x + I_{\underline{d}\leq Cx\leq\bar{d}}(x) + \nu^T Bx\right\} \tag{27}$$
$$\mu^k = \nu^k + L^{-1}(Bx^k - b) \tag{28}$$

Since we have no structural constraints on L, we can choose any $L \succeq BH^{-1}B^T$. If $BH^{-1}B^T$ is sparse, an efficient choice is to let $L = BH^{-1}B^T$ and compute and store a sparse Cholesky factorization of $BH^{-1}B^T$. Updating μ^k then reduces to a forward and backward solve in each subsequent iteration.

The complexity of solving (27) depends highly on the structures of H and C. If H and C are block-diagonal with sufficiently small blocks, then (27) can be solved efficiently and exactly in parallel using, e.g., the MPT toolbox, [22]. If H and C are diagonal, solving (27) reduces to an element-wise clip operation. For problems where solving (27) is computationally expensive and no exact solutions can be obtained easily or fast, we suggest to instead use QP splitting 1.

7.2 The distributed case

We consider separable optimization problems of the form:

$$\begin{array}{ll} \text{minimize} & \displaystyle\sum_{i=1}^{M}(f_i(x_i) + g_i(y_i)) \\ \text{subject to} & Ax = y \end{array} \tag{29}$$

where $f_i : \mathbb{E}_{H_i} \to \overline{\mathbb{R}}$ is 1-strongly convex (w.r.t. $\mathbb{E}_{H_i}$), $g_i \in \Gamma_0(\mathbb{E}_{K_i})$, $x = (x_1, \ldots, x_M)$, $y = (y_1, \ldots, y_M)$, and

$$A = \begin{bmatrix} A_{11} & \ldots & A_{1M} \\ \vdots & \ddots & \vdots \\ A_{M1} & \ldots & A_{MM} \end{bmatrix}.$$

We further assume that many $A_{ij} = 0$. The non-zero block entries of A are indexed by the sets

$$\mathcal{N}_i = \{j \in \{1, \ldots, M\} \mid A_{ij} \neq 0\}$$
$$\mathcal{M}_j = \{i \in \{1, \ldots, M\} \mid A_{ij} \neq 0\}.$$

We introduce the notation $x_{\mathcal{N}_i} = (\ldots, x_j, \ldots)$ that stacks all x_j with $j \in \mathcal{N}_i$, and $A_{\mathcal{N}_i} = [\ldots, A_{ij}, \ldots]$ that collects all $A_{ij} \neq 0$ in block-row i. This implies that (29) can equivalently be written as

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^{M} (f_i(x_i) + g_i(y_i)) \\ \text{subject to} & A_{\mathcal{N}_i} x_{\mathcal{N}_i} = y_i, \text{ for all } i \in \{1, \ldots, M\}. \end{array}$$

We introduce dual variables μ_i for all equality constraints $A_{\mathcal{N}_i} x_{\mathcal{N}_i} = y_i$ and define the vectors $\mu_{\mathcal{M}_j} = (\ldots, \mu_i, \ldots)$ that stacks all μ_i with $i \in \mathcal{M}_j$. We also define the matrix $A_{\mathcal{M}_j} = [\ldots, A_{ij}^T, \ldots]^T$ that collects all $A_{ij} \neq 0$ in block-column j. This implies that the dual problem can be written as

$$\text{minimize} \quad \sum_{i=1}^{M} (d_i(\mu_{\mathcal{M}_i}) + g_i^\star(\mu_i))$$

where

$$d_i(\mu_{\mathcal{M}_i}) := f_i^\star(-A_{\mathcal{M}_i}^T \mu_{\mathcal{M}_i}).$$

In [1], it is shown how to compute a matrix L that defines a space $\mathbb{E}_L$, using distributed computations only, on which

$$d(\mu) := \sum_{i=1}^{m} d_i(\mu_{\mathcal{M}_i})$$

is 1-smooth. The procedure from [1] is presented below:

$$L_{\mathcal{M}_j} = \tfrac{\|A_{\mathcal{M}_j}\|_2^2}{\lambda_{\min}(H_j)} \tag{30}$$

$$L_i = \sum_{j \in \mathcal{N}_i} L_{\mathcal{M}_j} \tag{31}$$

$$L = \text{blkdiag}(L_1 I, \ldots, L_M I). \tag{32}$$

This metric selection procedure relies on that d_i is $\frac{\|A_{\mathcal{M}_i}\|_2^2}{\lambda_{\min}(H_i)}$-smooth w.r.t $\mathbb{E}_I$. From Proposition 2, we know that d_i is 1-smooth w.r.t. $\mathbb{E}_{L_{\mathcal{M}_i}}$ for any $L_{\mathcal{M}_i} \succeq A_{\mathcal{M}_i} H_i^{-1} A_{\mathcal{M}_i}^T$, which gives a tighter characterization of d_i. Using Proposition 2, the distributed metric selection procedure proposed in [1] can be modified to yield a less conservative set from which L may be chosen:

$$L_{\mathcal{M}_j} = \text{blkdiag}(\ldots, L_{\mathcal{M}_j, i}, \ldots) \succeq A_{\mathcal{M}_j} H_j^{-1} A_{\mathcal{M}_j}^T \tag{33}$$

$$L_i = \sum_{j \in \mathcal{N}_i} L_{\mathcal{M}_j, i} \tag{34}$$

$$L = \text{blkdiag}(L_1, \ldots, L_M) \tag{35}$$

Table 1: Comparison to other first-order methods, all implemented in MATLAB. FDFBS refers to fast dual forward backward splitting, ADMM refers to alternating direction method of multipliers, QPi refers to QP-splitting i (for $i = 1, 2$).

		exec time (ms)		nbr iters	
Algorithm/splitting(/reference)	Parameters	avg.	max	avg.	max
FDFBS/QP1	L diag fr. (19) with $Q = CM_{11}C^T$	1.4	7.1	23.5	128
FDFBS/QP1	L diag fr. (18) with $Q = CH^{-1}C^T$	1.2	5.8	20.0	105
FDFBS/QP1/[42]	$L = \|CM_{11}C^T\|_2 I$	98.5	673.0	1835.9	12686
FDFBS/QP1/[42]	$L = \|CH^{-1}C^T\|_2 I$	98.9	679.4	1850.1	12783
FDFBS/QP2	$L = BH^{-1}B^T$	2.3	12.1	21.7	102
FDFBS/QP2/[43]	$L = \|BH^{-1}B^T\|_2 I$	4713.9	28411	50845	308210
ADMM/QP1/[41]	$\rho = 0.3$	193.9	920.6	3129.5	15037
ADMM/QP1/[41]	$\rho = 3$	29.7	142.2	457.3	2179
ADMM/QP1/[41]	$\rho = 30$	35.1	264.4	556.7	4194

where $L_{\mathcal{M}_j,i}$ are sub-blocks of the same dimension as μ_i. This structure allows for a distributed implementation of the fast dual forward backward splitting algorithm on $\mathbb{E}_L$. Also, the proof in [1] to show 1-smoothness of d w.r.t. $\mathbb{E}_L$ with L from (30)-(32) easily generalized to that d is 1-smooth w.r.t. $\mathbb{E}_L$ with L from (33)-(35). This is needed to guarantee convergence of the distributed fast dual forward-backward splitting algorithm using metric L. The local $L_{\mathcal{M}_j}$ could be chosen using some method from Section 6 to give the algorithm a good approximation of d, which will lead to improved performance.

Remark 5 *Note that $L_{\mathcal{M}_j}$ should be block-diagonal to facilitate a distributed implementation. However, we could superimpose additional internal structural constraints on each sub-block $L_{\mathcal{M}_j,i}$ (e.g., diagonal, sparse) that may differ from one block to the next.*

8 Numerical examples

To evaluate the proposed methods, we apply them on a (small-scale) aircraft control problem and on large-scale separable randomly generated problems that are solved in distributed fashion. We also compare to other methods in the literature.

8.1 Aircraft control

Here, we apply QP-splitting 1 and QP-splitting 2 from Section 7.1 to the AFTI-16 aircraft model in [25, 3]. As in [3], the continuous time model from [25] is sampled using zero-order hold every 0.05 s. The system has four states $x = (x_1, x_2, x_3, x_4)$, two outputs $y = (y_1, y_2)$, two inputs $u = (u_1, u_2)$, and obeys the

Table 2: Comparison to other solvers, all implemented in C. We report average and worst case execution times, code size for methods that generate problem specific code, and what type of algorithm that is used.

		exec time (ms)			
Alg. or software (/split.)	Comments	avg.	max	code size	algorithm type
FDFBS/QP1 (QPgen)	L diag fr. (19) with $Q = CH^{-1}C^T$	0.083	0.212	36 kB	First order
FDFBS/QP2	$L = BH^{-1}B^T$	0.079	0.232	54 kB	First order
FORCES		0.347	0.592	109 kB	Interior-point
CVXGEN		0.639	0.760	404 kB	Interior-point
MPT toolbox	$N = 3$ and no tracking	0.22	0.31	9.8 MB	Explicit
qpOASES	warm-starting version	0.189	5.8	-	Online active set
qpOASES	cold-starting version	4.7	6.0	-	Active set
DuQuad	alg: inexact fast dual grad. method	27.2s	56.5s	-	First order
FiOrdOs	alg: [9]	38.4	58.2	27 kB	First order
MOSEK		4.6	8.1	-	Interior-point

following dynamics

$$x(t+1) = \begin{bmatrix} 0.999 & -3.008 & -0.113 & -1.608 \\ -0.000 & 0.986 & 0.048 & 0.000 \\ 0.000 & 2.083 & 1.009 & -0.000 \\ 0.000 & 0.053 & 0.050 & 1.000 \end{bmatrix} x(t) + \begin{bmatrix} -0.080 & -0.635 \\ -0.029 & -0.014 \\ -0.868 & -0.092 \\ -0.022 & -0.002 \end{bmatrix} u(t),$$

$$y(t) = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} x(t)$$

The system is unstable, the magnitude of the largest eigenvalue of the dynamics matrix is 1.313. The outputs are the attack and pitch angles, while the inputs are the elevator and flaperon angles. The inputs are physically constrained to satisfy $|u_i| \leq 25^\circ$, $i = 1, 2$. The outputs are soft constrained to satisfy $-s_1 - 0.5 \leq y_1 \leq 0.5 + s_2$ and $-s_3 - 100 \leq y_2 \leq 100 + s_4$ respectively, where $s = (s_1, s_2, s_3, s_4) \geq 0$ are slack variables. The cost in each time step is

$$\ell(x, u, s) = \frac{1}{2}\big((x - x_r)^T Q(x - x_r) + u^T Ru + s^T Ss\big)$$

where x_r is a reference, $Q = \text{diag}(10^{-4}, 10^2, 10^{-3}, 10^2)$, $R = 10^{-2}I$, and $S = 10^6 I$. This gives a condition number of 10^{10} of the full cost matrix. Further, the terminal cost is Q, and the control and prediction horizon is $N = 10$. The numerical data in Tables 7.2 and 7.2 are obtained by following a reference trajectory on the output. The objective is to change the pitch angle from 0° to 10° and then back to 0° while the angle of attack satisfies the output constraints $-0.5^\circ \leq y_1 \leq 0.5^\circ$. The full optimization problem can be written on the form

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}z^T Hz + r_t^T z \\ \text{subject to} & Bz = bx_t \\ & \underline{d} \leq Cz \leq \bar{d} \end{array} \tag{36}$$

Table 3: Numerical evaluation for fast dual forward-backward splitting (FDFBS) applied on different spaces $\mathbb{E}_L$, including the space selection from [1] and the Euclidean space. The comparison also includes the dual Newton CG method in [27].

			# communication rounds				
			local		global		avg. exec. time
Algorithm	Parameters	# ss/vars./constr.	avg.	max	avg.	max	12 cores [mm:ss.s]
FDFBS	L fr. (33)-(35), min trace	500/87k/246k	523.7	774	-	-	3.2
FDFBS	L fr. (33)-(35), $\|\cdot\|_2$-equil.	500/87k/246k	1912.5	3022	-	-	10.1
[1]	L fr. (30)-(32)	500/87k/246k	4789.8	7558	-	-	25.4
FDFBS	$L = \|AH^{-1}A^T\|_2 I$	500/87k/246k	6114.7	6556	-	-	32.4
[27]	$\epsilon_i = 10^{-4}, \mu = 0.8, \sigma = 0.3$	500/87k/246k	6661.1	28868	4082.6	17694	2:06.0
FDFBS	L fr. (33)-(35), min trace	2000/351k/993k	356.8	652	-	-	15.6
FDFBS	L fr. (33)-(35), $\|\cdot\|_2$-equil.	2000/351k/993k	1138.0	1666	-	-	33.0
[1]	L fr. (30)-(32)	2000/351k/993k	2530.5	3218	-	-	1:13.5
FDFBS	$L = \|AH^{-1}A^T\|_2 I$	2000/351k/993k	4474.9	4608	-	-	2:09.9
[27]	$\epsilon_i = 10^{-4}, \mu = 0.8, \sigma = 0.3$	2000/351k/993k	6464.1	20624	3961.9	12641	41:28.0
FDFBS	L (33)-(35), min trace	8000/1.41M/3.98M	340.2	426	-	-	44.6
FDFBS	L fr. (33)-(35), $\|\cdot\|_2$-equil.	8000/1.41M/3.98M	1350.1	1776	-	-	2:10.8
[1]	L fr. (30)-(32)	8000/1.41M/3.98M	3050.2	3458	-	-	4:55.5
FDFBS	$L = \|AH^{-1}A^T\|_2 I$	8000/1.41M/3.98M	10583.4	10688	-	-	17:05.3

where x_t and r_t may change between sampling instants.

In Table 7.2 we compare the performance of QP-splitting 1 and QP-splitting 2 from Section 7.1, when solving (36). We compare the performance when applied in the Euclidean setting and when using an appropriated metric L. Since QP-splitting 1 in the Euclidean space is the algorithm in [42], and QP-splitting 2 in the Euclidean space is the algorithm in [43], we compare to those methods. We also compare to the performance of the ADMM-based method in [41]. Since this method is based on ADMM, the ρ-parameter much be chosen. Table 7.2 provides results for the best performing ρ, namely $\rho = 3$, and two other choices. All algorithms are implemented in MATLAB and the numerical results in Table 7.2 are obtained by running the simulations on a Linux machine using a single core running at 2.9 GHz. We use termination criterion $\|z^k - z^\star\|_2/\|z^\star\|_2 \leq 0.005$, where z^k is the primal iterate in the algorithm at iteration k and $z^\star$ is the solution computed to high accuracy with an interior point method. Table 7.2 shows that the proposed metric selection improves the performance for QP-splitting 1 with one to two orders of magnitude, and for QP-splitting 2 with three orders of magnitude. Also, the proposed methods outperform the method in [41].

In Table 7.2, we compare different solvers implemented in C. For QP-splitting 1 we use QPgen [19] which implements this methods. For QP-splitting 2, we generate C code that take the reference trajectory and the initial state as inputs. We see that the two QP-splittings perform similarly, and we see improvements of more than a factor 20 compared to the MATLAB implementations in Table 7.2. The other pieces optimization software in our comparison are

FORCES, CVXGEN, qpOASES, FiOrdOs, DuQuad, the MPC toolbox, and MOSEK. FORCES [12] and CVXGEN [30] are based on interior point methods. We see that our methods are three to five times faster for this problem. DuQuad [34] and FiOrdOs [48] are based on first order methods with inexact (DuQuad) or exact (FiOrdOs) inner minimizations. In DuQuad, the fast gradient method is chosen, and in FiOrdOs, the primal-dual method in [9] is chosen. Table 7.2 shows that our methods outperform these methods on this example. This is largely due to the developed preconditioning techniques. We also compare to qpOASES, which performs similarly as our methods for this problem in the warm-starting case. The problem is too big for the MPT toolbox. However, we have compared to the MPT toolbox when the horizon is shortened to three (instead of ten), and when no reference tracking is used. This gives 30 instead of 100 decision variables and 4 instead of 64 parameters. Even in this reduced setting, the execution time is not better than for our methods and the code size is much larger. Finally, we compare to the commercial solver MOSEK, which is more than one order of magnitude slower than our methods.

8.2 Distributed examples

Here, we apply the fast dual forward-backward splitting method to solve randomly generated DMPC optimization problems. The systems to be controlled have a sparse dynamic interaction structure, which is decided using the method in [28, §6.1]. The resulting average degree of the generated interconnection structures are 2.27, 2.23, and 2.23 respectively and the number of subsystems are 500, 2000, and 8000 respectively. The number of states in each subsystem is between 10 and 20, the number of inputs are three or four, and the control horizon is $N = 10$. This gives a total number of 87060, 350860, and 1405790 decision variables respectively. The entries in the dynamics and input matrices are randomly chosen from the intervals $[-0.7\ 1.3]$ and $[-1\ 1]$ respectively. Then the dynamics matrix is re-scaled to get a spectral radius of 1.15. The states and inputs are upper and lower bounded by random bounds generated from the intervals $[0.4\ 1]$ and $[-1\ -0.4]$ respectively. The state and input cost matrices are diagonal and each diagonal entry is randomly chosen from the interval $[1\ 10^6]$.

We evaluate the distributed fast dual forward backward splitting when applied on the Euclidean space, when applied on $\mathbb{E}_L$ where L is computed as in (30)-(32) (this is the method proposed in [1]), and when applied on $\mathbb{E}_L$ where L is computed by the procedure proposed in this paper, (33)-(35). These methods are compared to the dual Newton conjugate gradient (CG) method proposed in [27].

The evaluation in Table 8.1 is obtained by generating 200 feasible random initial conditions for each system. The corresponding optimal control problems are solved using the different algorithms. For each problem size, we compare the performance when the L-matrix (that defines $\mathbb{E}_L$) is computed using (33)-(35). We include the cases where the local metric matrices $L_{\mathcal{M}_j}$ in (33) are computed by trace-minimization and 2-norm equilibration only. (Other options are computationally too expensive or give compatible performance.) These methods are compared to the performance of the method from [1], i.e., when the L-matrix is computed using (30)-(32), and to fast dual decomposition with the optimal parameter selection given by $L = \|AH^{-1}A^T\|_2 I$. Table 8.1 shows that the algorithm with L computed using trace minimization has the fewest communications rounds, then L computed using equilibration, thereafter the method from [1], and finally fast dual decomposition with a global step-size. This is expected since, as we traverse up the list, the approximations of the smooth part of the dual function used in the algorithm become better. For the algorithm where the L-matrix is computed using local trace-minimization problems, the L-matrix is block-diagonal, while the remaining algorithms have diagonal L-matrices. This added flexibility enables for a reduced number of iterations. We also note that the dual Newton CG method in [27] is outperformed on this example.

The execution times in Table 8.1 are pure execution times for the DMPC scheme when solved on 12 cores, i.e., without the metric computation step. To solve a semi-definite program, to find the metric L is often only computationally beneficial if it can be computed offline and used to solve many optimization problems, as in DMPC. Thus the first row in Table 8.1 of every problem size are merely for DMPC applications, or other applications with the same characteristic. The computational complexity for equilibration is only slightly higher than for the method in [1]. Both these methods are considerably less computationally demanding than computing $\|AH^{-1}A^T\|_2$ which is required to compute a global step-size. Therefore, these methods are useful for metric selection also in general distributed optimization.

9 Conclusions

We have proposed several methods, with different computational complexity, to compute spaces on which to perform fast dual forward-backward splitting. We have evaluated these methods by applying them to an aircraft control problem and to randomly generated DMPC problems. For the most ill-conditioned problem, the numerical evaluations show improvements of up to three orders of magnitude compared to if the standard Euclidean metric is used.

10 Acknowledgements

The first author is financially supported by the Swedish Foundation for Strategic Research and he is a member of the LCCC Linnaeus Center at Lund University. Financial support from the Swedish Research Council for the first author's Postdoctoral studies at Stanford University is also gratefully acknowledged. Further, Alexander Domahidi is gratefully acknowledged for suggesting the AFTI-16 control problem as a benchmark and providing FORCES code for the same. Finally, the authors acknowledge the anonymous reviewers for constructive feedback.

References

[1] A. Beck, A. Nedic, A. Ozdaglar, and M. Teboulle. Optimal distributed gradient methods for network resource allocation problems. Submitted for publication, 2013.

[2] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imaging Sciences*, 2(1):183–202, October 2009.

[3] A. Bemporad, A. Casavola, and E. Mosca. Nonlinear control of constrained linear systems via predictive reference management. *IEEE Transactions on Automatic Control*, 42(3):340–349, 1997.

[4] J. F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4(1):238–252, 1962.

[5] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear Matrix Inequalities in System and Control Theory*, volume 15 of *Studies in Applied Mathematics*. SIAM, Philadelphia, PA, June 1994.

[6] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

[7] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, 2004.

[8] A. Bradley. *Algorithms for the Equilibration of Matrices and Their Application to Limited-Memory Quasi-Newton Methods*. PhD thesis, Stanford University, 2010.

[9] A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems withapplications to imaging. *Journal of Mathematical Imaging and Vision*, 40(1):120–145, 2011.

[10] G. B. Danzig and P. Wolfe. The decomposition algorithm for linear programming. *Econometrica*, 4:767–778, 1961.

[11] M. D. Doan, T. Keviczky, and B. De Schutter. An iterative scheme for distributed model predictive control using Fenchel's duality. *Journal of Process Control*, 21(5):746–755, June 2011. Special Issue on Hierarchical and Distributed Model Predictive Control.

[12] A. Domahidi, A. Zgraggen, M.N. Zeilinger, M. Morari, and C.N. Jones. Efficient interior point methods for multistage problems arising in receding horizon control. In *51st IEEE Conference on Decision and Control*, pages 668–674, Maui, HI, USA, December 2012.

[13] H. Everett. Generalized Lagrange multiplier method for solving problems of optimum allocation of resources. *Operations Research*, 11:399–417, 1963.

[14] H. J. Ferreau, H. G. Bock, and M. Diehl. An online active set strategy to overcome the limitations of explicit mpc. *International Journal of Robust and Nonlinear Control*, 18(8):816–830, 2008.

[15] E. Ghadimi, I. Shames, and M. Johansson. Multi-step gradient methods for networked optimization. *Signal Processing, IEEE Transactions on*, 61(21):5417–5429, Nov 2013.

[16] P. Giselsson. A generalized distributed accelerated gradient method for DMPC with iteration complexity bounds. In *Proceedings of 2013 American Control Conference*, pages 327–333, Washington D.C., June 2013.

[17] P. Giselsson. Improved dual decomposition for distributed model predictive control. In *Proceedings of 2014 IFAC World Congress*, pages 1203–1209, Cape Town, South Africa, August 2014.

[18] P. Giselsson. Improved fast dual gradient methods for embedded model predictive control. In *Proceedings of 2014 IFAC World Congress*, pages 2303–2309, Cape Town, South Africa, August 2014.

[19] P. Giselsson. QPgen: A C code generator for quadratic optimization problems, 2014. `http://www.control.lth.se/user/pontus.giselsson/qpgen/`.

[20] P. Giselsson and S. Boyd. Preconditioning in fast dual gradient methods. In *Proceedings of the 53rd Conference on Decision and Control*, pages 5040–5045, Los Angeles, CA, December 2014.

[21] P. Giselsson, M. D. Doan, T. Keviczky, B. De Schutter, and A. Rantzer. Accelerated gradient methods and dual decomposition in distributed model predictive control. *Automatica*, 49(3):829–833, 2013.

[22] M. Herceg, M. Kvasnica, C. N. Jones, and M. Morari. Multi-parametric toolbox 3.0, 2013.

[23] R. A. Horn and C. R. Johnson. *Matrix Analysis.* Cambridge University Press, February 1990.

[24] J. L. Jerez, P. J. Goulart, S. Richter, G. A. Constantinides, E. C. Kerrigan, and M. Morari. Embedded online optimization for model predictive control at megahertz rates. *IEEE Transactions on Automatic Control*, 2013. Submitted.

[25] P. Kapasouris, M. Athans, and G. Stein. Design of feedback control systems for unstable plants with saturating actuators. In *Proceedings of the IFAC Symposium on Nonlinear Control System Design*, pages 302–307. Pergamon Press, 1990.

[26] P. A. Knight, D. Ruiz, and B. Uçar. A Symmetry Preserving Algorithm for Matrix Scaling. Research Report RR-7552, INRIA, February 2011.

[27] A. Kozma, E. Klintberg, S. Gros, and M. Diehl. An improved distributed dual newton-cg method for convex quadratic programming problems. In *Proceedings of 2014 American Control Conference*, 2014. Submitted.

[28] M. Kraning, E. Chu, J. Lavaei, and S. Boyd. Dynamic network energy management via proximal message passing. *Foundations and Trends in Optimization*, 1(2):70–122, 2013.

[29] Z. Lu and T. K. Pong. Minimizing condition number via convex programming. *SIAM Journal on Matrix Analysis and Applications*, 32(4):1193–1211, 2011.

[30] J. Mattingley and S. Boyd. Cvxgen: a code generator for embedded convex optimization. *Optimization and Engineering*, 13(1):1–27, 2012.

[31] Mosek. Optimization software. `http://www.mosek.com/`.

[32] I. Necoara and V. Nedelcu. Rate analysis of inexact dual first-order methods application to dual decomposition. *IEEE Transactions on Automatic Control*, 59(5):1232–1243, May 2014.

[33] I. Necoara and V. Nedelcu. On linear convergence of a distributed dual gradient algorithm for linearly constrained separable convex problems. *Automatica*, 2015. To appear.

[34] I. Necoara and A. Patrascu. DuQuad: An inexact (augmented) dual first order algorithm for quadratic programming. Available: `http://arxiv.org/abs/1504.05708`, March 2015.

[35] V. Nedelcu, I. Necoara, and Q. Tran-Dinh. Computational complexity of inexact gradient augmented lagrangian methods: Application to constrained mpc. *SIAM Journal on Control and Optimization*, 52(5):3109–3134, 2014.

[36] R. R. Negenborn. *Multi-Agent Model Predictive Control with Applications to Power Networks*. PhD thesis, TU Delft, 2007.

[37] Y. Nesterov. A method of solving a convex programming problem with convergence rate O $(1/\mathrm{k}^2)$. *Soviet Mathematics Doklady*, 27(2):372–376, 1983.

[38] Y. Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Springer Netherlands, 1st edition, 2003.

[39] Y. Nesterov. Smooth minimization of non-smooth functions. *Math. Program.*, 103(1):127–152, May 2005.

[40] Y. Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming*, 140(1):125–161, 2013.

[41] B. O'Donoghue, G. Stathopoulos, and S. Boyd. A splitting method for optimal control. *IEEE Transactions on Control Systems Technology*, 21(6):2432–2442, 2013.

[42] P. Patrinos and A. Bemporad. An accelerated dual gradient-projection algorithm for embedded linear model predictive control. *IEEE Transactions on Automatic Control*, 59(1):18–33, 2014.

[43] S. Richter, C. N. Jones, and M. Morari. Certification aspects of the fast gradient method for solving the dual of parametric convex programs. *Mathematical Methods of Operations Research*, 77(3):305–321, 2013.

[44] R. T. Rockafellar. *Convex Analysis*, volume 28. Princeton Univercity Press, Princeton, NJ, 1970.

[45] D. Ruiz. A scaling algorithm to equilibrate both rows and columns norms in matrices. Technical report, Rutherford Appleton Laboratories, 2001.

[46] R. Sinkhorn and P. Knopp. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*, 21(2):343–348, 1967.

[47] P. Tseng. On accelerated proximal gradient methods for convex-concave optimization. Technical report. Available: `http://www.csie.ntu.edu.tw/~b97058/tseng/papers/apgm.pdf`, May 2008.

[48] F. Ullmann and S. Richter. FiOrdOs: Code generation for first order methods, 2012. `http://fiordos.ethz.ch/`.

[49] C. Zalinescu. *Convex analysis in general vector spaces.* World Scientific, 2002.

[50] W. Zuo and Z. Lin. A generalized accelerated proximal gradient approach for total-variation-based image restoration. *IEEE Transactions on Image Processing*, 20(10):2748–2759, October 2011.

A Proof to Proposition 3

The function $f - \frac{1}{2}\|\cdot\|_H^2$ is linear on $\mathcal{B}_r^n(x^\star(\bar{\mu}))$ if and only if $f(x) = \frac{1}{2}\|x\|_H^2 + \langle \xi, x\rangle_2 + \theta$ for some $\xi \in \mathbb{E}_{H^{-1}}$, $\theta \in \mathbb{R}$ and all $x \in \mathcal{B}_r^n(x^\star(\bar{\mu}))$. Since $x^\star(\bar{\mu}) \in \mathcal{B}_r^n(x^\star(\bar{\mu}))$, we have

$$\begin{aligned} x^\star(\bar{\mu}) &= \operatorname*{argmin}_x \left\{ \tfrac{1}{2}\|x\|_H^2 + \langle \xi, x\rangle_2 + \theta + \langle A^T\bar{\mu}, x\rangle_2 \right\} \\ &= -H^{-1}(\xi + A^T\bar{\mu}). \end{aligned}$$

Due to continuity of $H^{-1}A^T$ we also have $x^\star(\bar{\mu}+\mu_d) = -H^{-1}(\xi + A^T(\bar{\mu}+\mu_d)) \in \mathcal{B}_r(x^\star(\bar{\mu}))$ for sufficiently small $\mu_d \in \mathbb{R}^m$ pointing in any direction. Thus, there exists a ball $\mathcal{B}_\epsilon^m(\bar{\mu})$ such that for each $\mu \in \mathcal{B}_\epsilon^m(\bar{\mu})$

$$\begin{aligned} d(\mu) &= -\min_x \left\{ \tfrac{1}{2}\|x\|_H^2 + \langle \xi, x\rangle_2 + \theta + \langle A^T\mu, x\rangle_2 \right\} \\ &= \tfrac{1}{2}\|A^T\mu + \xi\|_{H^{-1}}^2 - \theta. \end{aligned}$$

That is, d is a quadratic with Hessian $AH^{-1}A^T$ on $\mathcal{B}_\epsilon^m(\bar{\mu})$. This implies that for any $L \not\succeq AH^{-1}A^T$, where exist $\nu, \mu \in \mathcal{B}_\epsilon^m(\bar{\mu})$ (since $\mathcal{B}_\epsilon^m(\bar{\mu})$ has non-empty interior) such that (11) does not hold. This concludes the proof.

B Proof to Proposition 4

Since H is positive definite on the null-space of A, the inverse in (13) exists, see [7, p. 523]. Thus $x^\star(\mu) = -M_{11}(A^T\mu + \xi) + M_{12}b$, where $x^\star$ is defined in (8). Further

$$\begin{aligned} d(\mu) &= \frac{1}{2}\mu^T A(2M_{11} - M_{11}HM_{11})A^T\mu + \zeta^T\mu + \theta \\ &= \frac{1}{2}\mu^T AM_{11}A^T\mu + \zeta^T\mu + \theta \end{aligned}$$

where $\zeta \in \mathbb{R}^{m+p}$ and $\theta \in \mathbb{R}$ collect the linear and constant terms, and where $M_{11}HM_{11} = M_{11}$ is used in the second equality. This identity follows from the upper left block of $\left[\begin{smallmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{smallmatrix}\right]\left[\begin{smallmatrix} H & B^T \\ B & 0 \end{smallmatrix}\right]\left[\begin{smallmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{smallmatrix}\right] = \left[\begin{smallmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{smallmatrix}\right]$ and using $M_{11}^T B = M_{11}B = BM_{11} = 0$, where $BM_{11} = 0$ follows from the lower left block of $\left[\begin{smallmatrix} H & B^T \\ B & 0 \end{smallmatrix}\right]\left[\begin{smallmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{smallmatrix}\right] = \left[\begin{smallmatrix} I & 0 \\ 0 & I \end{smallmatrix}\right]$. This implies that (12) holds with $L = AM_{11}A^T$ and obviously for any $L \succeq AM_{11}A^T$. Further, since d is a quadratic function with Hessian $AM_{11}A^T$, (12) is tight for $L = AM_{11}A^T$ for all $\mu, \nu \in \mathbb{E}_{K^{-1}}$.Thus, no $L \not\succeq AM_{11}A^T$ exists such that (12) holds for all $\mu, \nu \in \mathbb{E}_{K^{-1}}$. This concludes the proof.