Constraint Preconditioning and Parameter Selection for a First-Order Primal-Dual Method applied to Model Predictive Control

Govind M. Chari, Yue Yu, and Behçet Açıkmeşe

Abstract— Many techniques for real-time trajectory optimization and control require the solution of optimization problems at high frequencies. However, ill-conditioning in the optimization problem can significantly reduce the speed of first-order primal-dual optimization algorithms. We introduce a preconditioning technique and step-size heuristic for Proportional-Integral Projected Gradient (PIPG), a first-order primal-dual algorithm. The preconditioning technique, based on the QR factorization, aims to reduce the condition number of the KKT matrix associated with the optimization problem. Our step-size selection heuristic chooses step-sizes to minimize the upper bound on the convergence of the primal-dual gap for the optimization problem. These algorithms are tested on two model predictive control problem examples and show a solve-time reduction of at least 3.6x.

I. INTRODUCTION

Solving optimal control problems in real-time is at the heart of model predictive control (MPC). This control scheme is optimization-based, where we solve a fixed time horizon optimal control problem recursively to generate state and control trajectories which satisfy relevant system constraints while minimizing a predefined cost function.

In this paper, we consider optimal control problems in the form

$$\underset{z}{\text{minimize}} \quad \frac{1}{2}z^\top P z + q^\top z \tag{1a}$$

$$\text{subject to} \quad Hz - g = 0 \tag{1b}$$

$$z \in \mathbb{D} \tag{1c}$$

where $H \in \mathbb{R}^{m \times n}$, $m \leqslant n$, and $P > 0$. Constraint 1b contains the linear dynamics and set $\mathbb{D}$ contains state and control constraints. We assume that set $\mathbb{D}$ is the Cartesian product of sets that have closed-form projections, such as balls, boxes, second-order cones, half-spaces, and the intersection of a ball and second-order cone [1, 2]. This assumption applies to many typical state and control constraints in MPC [3].

First-order optimization algorithms are popular for solving MPC problems due to their ability to warm-start from any prescribed solution (e.g., one that can be obtained from

G. Chari is a PhD student in the Department of Aeronautics & Astronautics at the University of Washington, Seattle, WA 98195 gchari@uw.edu

Y. Yu is an assistant professor in the Department of Aerospace Engineering and Mechanics at the University of Minnesota Twin Cities, Minneapolis, MN 55455. yuey@umn.edu

B. Açıkmeşe is a professor in the Department of Aeronautics & Astronautics at the University of Washington, Seattle, WA 98195 behcet@uw.edu

the previous solution in a recursive MPC scheme), which greatly reduces solve-times and per-iteration computational cost. Unlike second-order methods, first-order methods do not require expensive matrix factorizations, making them attractive for resource constrained embedded systems [4].

However, ill-conditioned optimization problems degrade the performance of methods in this class. Consequently, first-order optimization algorithms use preconditioning and parameter selection to improve their convergence behavior. Preconditioning transforms the given optimization problem into an equivalent problem which is able to be solved in fewer iterations. This is achieved by performing a change of variables or transforming the constraints of the problem to reduce the condition number of a matrix related to the optimization problem, such as the KKT matrix. Whereas preconditioning modifies the given optimization problem, parameter selection modifies the optimization algorithm itself. Parameter selection involves choosing parameters of an optimization algorithm such as step-size or relaxation parameters for fastest convergence [5].

Modified Ruiz equilibration [6] and the hypersphere preconditioner [7] are two existing preconditioning techniques for first-order primal-dual algorithms. These preconditioners aim to reduce the condition number of the KKT matrix. Both perform a change of variables and additionally, modified Ruiz equilibration scales the constraints with a diagonal matrix. These two preconditioner transform set $\mathbb{D}$ in Problem 1 since they both employ a change of variables. This requires special structure in the matrices defining the change of variables in order to preserve closed-form projections onto $\mathbb{D}$, which is needed to solve Problem 1 by using Proportional-Integral Projected Gradient (PIPG), a first-order primal-dual optimization algorithm [8, 9, 10].

Our first contribution is the development of the *QR preconditioner*. This proposed preconditioner does not perform a change of variables, meaning set $\mathbb{D}$ is not transformed and consequently closed-form projections are preserved. Instead, the QR preconditioner transforms the equality constraints by scaling them with a triangular matrix to minimize the condition number of the KKT matrix. Since a triangular matrix is used for transforming the constraints as opposed to a diagonal matrix, the QR preconditioner has more degrees of freedom to transform the constraints than modified Ruiz equilibration. The QR preconditioner requires a matrix factorization to perform this constraint transformation, unlike the aforementioned preconditioners, making it more computationally expensive. However, in our numerical examples the factorization time was a small percentage of

the overall solve-time. Additionally, in settings requiring the solution of multiple optimization problems with identical equality constraints, such as quasi-convex optimization and MPC, performing the preconditioning offline and solving the preconditioned problem online will result in improved real-time performance.

Our second contribution is a step-size selection heuristic for PIPG. For fastest convergence, PIPG requires carefully selecting step sizes, a process that is currently manually tuned. We eliminate manual tuning of step-sizes by introducing a step-size selection heuristic based on choosing step-sizes to minimize the upper bound on the primal-dual gap.

A. Related work

In this section we will discuss two existing preconditioning methods.

The hypersphere preconditioner [7] performs a change of variables to make the Hessian of the objective function some positive multiple of the identity matrix, making the objective function perfectly conditioned. This preconditioner avoids transforming the constraints. It is applicable to PIPG when the Hessian of the objective function is diagonal or has an easy to compute Cholesky factorization, and the change of variables induced by the preconditioner preserves the ability to project onto set $\mathbb{D}$ in closed-form.

Modified Ruiz equilibration [6] preconditions the objective function and transforms the constraints by performing a linear change of variables with a diagonal matrix and scaling the constraints with a diagonal matrix to make the preconditioned KKT matrix have equal row and column norms, a heuristic for good conditioning. Similar to the hypersphere preconditioner, to be applicable to PIPG the change of variables induced must preserve the ability to project onto set $\mathbb{D}$ in closed-form.

B. Notation

We denote the $n \times 1$ vector of zeros as 0_n, the $m \times n$ matrix of zeros as $0_{m \times n}$, the $n \times n$ identity matrix as I_n, the Euclidean projection of point z onto the convex set $\mathbb{D}$ as $\Pi_{\mathbb{D}}[z]$, the concatenation of vectors u and v as (u, v), the vector formed from the ith through the jth component of z as $z^{i:j}$, and the Euclidean norm of a vector z as $\|z\| := \sqrt{z^\top z}$.

II. PROPORTIONAL-INTEGRAL PROJECTED GRADIENT

Proportional-Integral Projected Gradient (PIPG), shown in Algorithm 1, is a first-order primal-dual conic optimization algorithm capable of infeasibility detection. This algorithm is specialized to handle common constraints sets which arise in optimal control problems, such as boxes, balls, and second-order cones, via closed-form projections [8, 9, 10]. PIPG has been demonstrated to be faster than many state-of-the art optimization solvers such as ECOS, MOSEK, GUROBI, and OSQP [10]. It has also been extensively applied to trajectory optimization problems including three-degree-of-freedom (DoF) powered-descent guidance and has been used as a subproblem solver within sequential convex

programming for multi-phase powered-descent guidance, 6-DoF powered-descent-guidance, and spacecraft rendezvous problems [7, 11, 12, 13, 14].

Algorithm 1 PIPG

Require: $k_{\max}, \alpha, \beta, z^1 \in \mathbb{D}, v^1$.
Ensure: z^k.
 1: **for** $k = 1, 2, \ldots, k_{\max} - 1$ **do**
 2: $\quad w^{k+1} = v^k + \beta(Hz^k - g)$
 3: $\quad z^{k+1} = \Pi_{\mathbb{D}}[z^k - \alpha(Pz^k + q + H^\top w^{k+1})]$
 4: $\quad v^{k+1} = w^{k+1} + \beta H(z^{k+1} - z^k)$

In Algorithm 1, z^k and w^k are the iterates of the primal and dual variables respectively. Thus, we refer to α and β as the primal and dual step-sizes. For the convergence results to hold for PIPG, the primal and dual step-sizes must satisfy the algebraic relationship

$$\alpha(\lambda_{\max} + \sigma\beta) < 1.$$

We parameterize this family of step-sizes with $\gamma > 0$ as

$$\alpha < \frac{1}{\lambda_{\max} + \gamma}, \quad \beta = \frac{\gamma}{\sigma} \tag{2}$$

where $\lambda_{\max}$ is the maximum eigenvalue of P and σ is the maximum eigenvalue of $H^\top H$. In practice we choose α to be equal to the upper bound and do not observe convergence issues. We will discuss how to choose γ in Section IV.

III. QR PRECONDITIONER

In this section, we derive and provide a geometric interpretation for the QR preconditioner.

A. Derivation

First, we derive a preconditioner that only modifies Constraint 1b of Problem 1 and minimizes the condition number of the KKT matrix. First, we need the following assumptions on the constraint matrix H and the Hessian of the objective function P.

Assumption 1: $H \in \mathbb{R}^{m \times n}$ has full row rank.

Assumption 2: P is positive definite and has maximum and minimum eigenvalues $\lambda_{\max}$ and $\lambda_{\min}$ respectively.

The KKT conditions of Problem 1 are as follows

$$\underbrace{\begin{bmatrix} P & H^\top \\ H & 0_{m \times m} \end{bmatrix}}_{\mathcal{K}} \begin{bmatrix} z^\star \\ w^\star \end{bmatrix} + \begin{bmatrix} q \\ -g \end{bmatrix} \in \begin{bmatrix} -\mathbb{N}_{\mathbb{D}}(z^\star) \\ \{0_m\} \end{bmatrix} \tag{3}$$

where $z^\star$ and $w^\star$ are the optimal primal and dual solutions respectively and $\mathbb{N}_{\mathbb{D}}(z^\star)$ is the normal cone of set $\mathbb{D}$ at $z^\star$. We denote the KKT matrix as $\mathcal{K}$.

If Assumptions 1 and 2 hold and $\sigma_{\max}$ and $\sigma_{\min}$ are the maximum and minimum singular values of H, the eigenvalues of $\mathcal{K}$ lie in the following set [15]

$$\lambda(\mathcal{K}) \subset \mathcal{I}^- \cup \mathcal{I}^+$$

where

$$\mathcal{I}^- = \left[\frac{1}{2} \left(\lambda_{\min} - \sqrt{\lambda_{\min}^2 + 4\sigma_{\max}^2} \right), \right.$$
$$\left. \frac{1}{2} \left(\lambda_{\max} - \sqrt{\lambda_{\max}^2 + 4\sigma_{\min}^2} \right) \right]$$

$$\mathcal{I}^+ = \left[\lambda_{\min}, \frac{1}{2} \left(\lambda_{\max} + \sqrt{\lambda_{\max}^2 + 4\sigma_{\max}^2} \right) \right].$$

The spectral condition number of $\mathcal{K}$ is defined as [15]

$$\kappa(\mathcal{K}) = \frac{\max |\lambda(\mathcal{K})|}{\min |\lambda(\mathcal{K})|}.$$

The condition number quantifies the difficulty in terms of the number of iterations to solve Equation 3 with an iterative method. Thus, a smaller condition number for $\mathcal{K}$ results in faster convergence for PIPG which solves the KKT conditions in Equation 3.

The upper bound for the $\mathcal{I}^+$ interval is larger in magnitude than the lower bound for the $\mathcal{I}^-$ interval since $\lambda_{\max} \geqslant \lambda_{\min} > 0$ and $\sigma_{\max} > 0$ by Assumptions 1 and 2. We can construct an upper bound on the condition number of $\mathcal{K}$ as follows

$$\kappa(\mathcal{K}) \leqslant \max \left(\frac{\lambda_{\max} + \sqrt{\lambda_{\max}^2 + 4\sigma_{\max}^2}}{\sqrt{\lambda_{\max}^2 + 4\sigma_{\min}^2} - \lambda_{\max}}, \right. $$
$$\left. \frac{\lambda_{\max} + \sqrt{\lambda_{\max}^2 + 4\sigma_{\max}^2}}{2\lambda_{\min}} \right). \quad (4)$$

Lemma 1: The singular values for H which minimize the condition number of the KKT matrix are $\sigma_{\min} = \sigma_{\max} = \sqrt{\lambda_{\max}\lambda_{\min} + \lambda_{\min}^2}$

Proof: We will first prove by contradiction that $\sigma_{\max}$ and $\sigma_{\min}$ which minimize the bound on the condition number given by Equation 4 must be equal.

Suppose that we have $\sigma_{\min} < \sigma_{\max}$ which minimize the bound on the condition number. Both arguments of the max operator are monotonically increasing in $\sigma_{\max}$, so we can reduce $\sigma_{\max}$ and further reduce the bound on the condition number which produced a contradiction.

Since $\sigma_{\min} = \sigma_{\max}$ at optimality, the first argument of the max operator and its derivative are

$$h(\sigma_{\max}) = \frac{\lambda_{\max} + \sqrt{\lambda_{\max}^2 + 4\sigma_{\max}^2}}{\sqrt{\lambda_{\max}^2 + 4\sigma_{\max}^2} - \lambda_{\max}}$$

$$h'(\sigma_{\max}) = \frac{-8\lambda_{\max}\sigma_{\max}\sqrt{\lambda_{\max}^2 + 4\sigma_{\max}^2}}{\left(\sqrt{\lambda_{\max}^2 + 4\sigma_{\max}^2} - \lambda_{\max} \right)^2}.$$

Since $\lambda_{\max} > 0$ and $\sigma_{\max} > 0$, $h'(\sigma_{\max})$ is negative and thus $h(\sigma_{\max})$ is monotonically decreasing in $\sigma_{\max}$.

It is also clear that the second argument of the max operator in Equation 4 is monotonically increasing in $\sigma_{\max}$. To minimize the maximum of a monotonically decreasing and monotonically increasing function, we must set the two arguments equal. By setting the arguments equal we obtain

the minimum and maximum singular values of H which minimize the KKT condition number

$$\sigma_{\max} = \sigma_{\min} = \sqrt{\lambda_{\max}\lambda_{\min} + \lambda_{\min}^2}.$$

$\blacksquare$

If $HH^\top = (\lambda_{\min}\lambda_{\max} + \lambda_{\min}^2)I_m$, then the singular values of H are $\sqrt{\lambda_{\max}\lambda_{\min} + \lambda_{\min}^2}$ which minimize the condition number of the KKT matrix by Lemma 1. For $HH^\top$ to be some scaling of the identity matrix, it is sufficient for the rows of H to be orthogonal. To do this, we will apply a QR factorization to $H^\top$, and transform Constraint 1b into Constraint 5e as follows

$$Hz - g = 0 \quad (5a)$$
$$\Rightarrow R^\top Q^\top z - g = 0 \quad (5b)$$
$$\Rightarrow Q^\top z - R^{-\top}g = 0 \quad (5c)$$
$$\Rightarrow \eta(Q^\top z - R^{-\top}g) = 0 \quad (5d)$$
$$\Rightarrow \hat{H}z - \hat{g} = 0 \quad (5e)$$

where $H^\top = QR$ is the thin or economy QR factorization of $H^\top$, $\hat{H} = \eta Q^\top$, $\hat{g} = \eta R^{-\top}g$, and Assumption 1 ensures invertibility of $R^\top$ [16]. Note that Assumption 1 is equivalent to requiring that linear independence constraint qualification (LICQ) of Constraint 1b holds for Problem 1 [17]. We can arbitrarily scale the constraint in Equation 5d by some $\eta > 0$ and the rows of $\hat{H}$ will still be orthogonal. If we choose $\eta = \sqrt{\lambda_{\max}\lambda_{\min} + \lambda_{\min}^2}$, then $\hat{H}\hat{H}^\top = (\lambda_{\min}\lambda_{\max} + \lambda_{\min}^2)I_m$, thus minimizing the condition number of the KKT matrix. The QR preconditioner is given by Algorithm 2.

Algorithm 2 QR Preconditioner

1: **function** QR_PRECONDITIONER(H, g, $\lambda_{\max}$, $\lambda_{\min}$)
2: $\quad \eta = \sqrt{\lambda_{\max}\lambda_{\min} + \lambda_{\min}^2}$
3: $\quad Q, R = \text{qr}(H^\top) \quad \triangleright$ *economy QR factorization*
4: $\quad \hat{H} = \eta Q^\top$
5: $\quad \hat{g} = \eta R^{-\top}g$
6: $\quad$ **return** $\hat{H}, \hat{g}$

B. Geometric Interpretation

To interpret this preconditioner geometrically, we will consider the simplified problem of using PIPG to minimize a bivariate quadratic function subject to two equality constraints with nearly parallel normal vectors. Figures 1 and 2 depict the level curves of the objective function and the PIPG iterates before and after the QR preconditioner is applied.

We can see that in Figure 1, PIPG takes several iterations to converge. Notice that Line 3 of PIPG takes a step in the negative gradient direction of the objective function in order to minimize the objective, then takes a step in a direction given by a linear combination of the rows of H (the $H^\top w^{k+1}$ term) to satisfy Constraint 1b. The rows of H are the normal vectors of the hyperplanes which define the equality constraints in Problem 1. If we have two equality
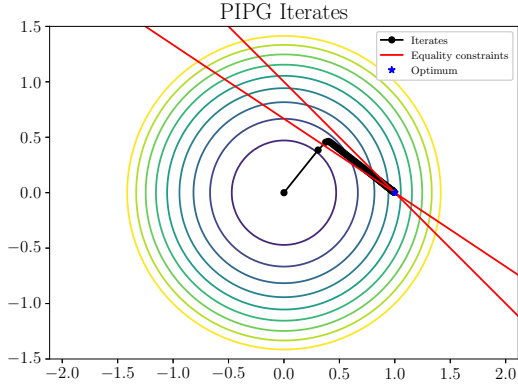
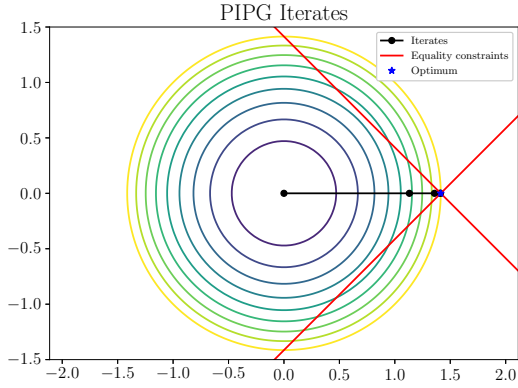Fig. 1: PIPG iterates with nearly parallel equality constraints

Fig. 2: PIPG iterates after applying QR preconditioner

IV. PARAMETER SELECTION

In this section we will discuss how to select the step-sizes α and β to minimize the upper bound on the primal-dual gap for the iterates of PIPG.

For convergence of PIPG, the primal and dual step-sizes must satisfy the relationship

$$\alpha(\lambda_{\max} + \sigma\beta) < 1 \tag{6}$$

where α and β are the primal and dual step-sizes respectively, $\lambda_{\max}$ is the maximum eigenvalue of P, and σ is the maximum eigenvalue of $H^\top H$.

There are infinite choices for α and β which satisfy the above relationship, and it has been observed that for fastest convergence a "proper" choice for α and β must be made. From Proposition 1 of [8], we see that zero primal-dual gap is a sufficient condition for optimality of Problem 1. Thus, we propose to choose step-sizes which minimize the upper bound on the primal-dual gap.

In [10], the authors introduce a positive, scalar parameter, referred to as ω, to select an α and β which satisfy the relationship given by Equation 6. We introduce a different scalar parameter γ, to choose an α and β using Equation 2. Using γ instead of the previously introduced ω makes the analysis in this section tractable.

Theorem 1 from [8] provides an upper bound on the primal-dual gap for Algorithm 1. We will rewrite this theorem to explicitly show dependence on γ

$$\mathcal{L}(\bar{z}^k, w^\star) - \mathcal{L}(z^\star, \bar{w}^k) \leqslant \frac{f(\gamma)}{k} \tag{7}$$

$$f(\gamma) = \frac{\lambda_{\max} + \gamma}{2}\|z^1 - z^\star\|^2 + \frac{\sigma}{2\gamma}\|v^1 - w^\star\|^2 \tag{8}$$

$$\bar{z}^k = \frac{1}{k}\sum_{j=1}^{k} z^{j+1}, \quad \bar{w}^k = \frac{1}{k}\sum_{j=1}^{k} w^{j+1}$$

where $\{z^j, w^j\}_{j=1}^{k}$ are generated by Algorithm 1, $(z^\star, w^\star)$ is the optimal primal-dual solution of Problem 1, and $\mathcal{L}(z, w) = \frac{1}{2}z^\top P z + q^\top z + w^\top(Hz - g)$ is the Lagrangian of Problem 1. Both γ and σ are defined to be positive real numbers. Under this assumption, we see that $f(\gamma)$ given by Equation 8 is convex in γ and has a minimizer given by

$$\gamma^\star = \sqrt{\sigma}\frac{\|v^1 - w^\star\|}{\|z^1 - z^\star\|}.$$

Since $\gamma^\star$ is a function of the optimal primal-dual solution $(z^\star, w^\star)$, we cannot directly choose the optimal step-sizes. Instead, we suggest a heuristic where for every fixed number of iterations of PIPG, we pick the value for γ which minimizes an approximation of Equation 8, where the optimal primal-dual solution, $(z^\star, w^\star)$, is replaced with our current iterates of the primal and dual variables, (z^k, w^k). The value of γ which minimizes this approximation of Equation 8 after the kth iteration of PIPG is

constraints with nearly parallel hyperplane normals (as in Figure 1), we require smaller coefficients in w^{k+1} for the linear combination of the hyperplane normals, given by $H^\top w^{k+1}$, to move the iterates in a direction nearly parallel to hyperplane normals (up and to the right in Figure 1), but require large coefficients for the iterates to move in a direction nearly orthogonal to the hyperplane normals (down and to the right in Figure 1). The coefficients of this linear combination are given by our current iterate of the dual variables w^{k+1}. To get large enough w^{k+1} for the iterates to move nearly orthogonal to the hyperplane normals, we must wait for the integrator given by Line 2 to accumulate enough error, resulting in slow convergence.

The QR preconditioner transforms the equality constraints given by Equation 1b to a equivalent set of equality constraints with the same feasible set as the original constraints, but one whose hyperplane normal vectors are all orthogonal and of the same length, making it easy for iterates to move in all directions. After applying the QR preconditioner, we can see the resulting iterates in Figure 2 where the optimum is achieved in far fewer iterations.

$$\hat{\gamma}^{\star} = \sqrt{\sigma}\frac{\|v^1 - w^k\|}{\|z^1 - z^k\|}.$$

The proposed step-size selection algorithm is given by Algorithm 3. We can combine Algorithms 1, 2, and 3 into Algorithm 4 to show how our proposed algorithms should be used within PIPG.

Although choosing γ to minimize the upper bound on the primal-dual gap given by Equation 7 only guarantees improving the worst-case performance of PIPG, we observe improved convergence in our numerical results.

Algorithm 3 Parameter selection for PIPG

1: **function** STEP_SELECTION($z^1, v^1, z^k, w^k, \lambda_{\max}, \sigma$)
2:     $\hat{\gamma}^{\star} = \sqrt{\sigma}(\|z^1 - z^k\|/\|v^1 - w^k\|)$
3:     $\alpha = 1/(\lambda_{\max} + \hat{\gamma}^{\star})$
4:     $\beta = \hat{\gamma}^{\star}/\sigma$
5:     **return** α, β

Algorithm 4 PIPG w/ QR Preconditioner and Parameter Selection

---

Require: $k_{\max}, k_{\text{update}}, \alpha_{\text{init}}, \beta_{\text{init}}, z^1 \in \mathbb{D}, v^1, H, g$.
Ensure: z^k.
1: $\hat{H}, \hat{g} = $ QR_PRECONDITIONER($H, g, \lambda_{\max}, \lambda_{\min}$)
2: $\alpha, \beta = \alpha_{\text{init}}, \beta_{\text{init}}$
3: **for** $k = 1, 2, \ldots, k_{\max} - 1$ **do**
4:     **if** $\mod(k, k_{\text{update}}) == 0$ **then**
5:       $\alpha, \beta = $ STEP_SELECTION($z^1, v^1, z^k, w^k, \lambda_{\max}, \sigma$)
6:     $w^{k+1} = v^k + \beta(\hat{H}z^k - \hat{g})$
7:     $z^{k+1} = \Pi_{\mathbb{D}}[z^k - \alpha(Pz^k + q + \hat{H}^{\top}w^{k+1})]$
8:     $v^{k+1} = w^{k+1} + \beta\hat{H}(z^{k+1} - z^k)$

V. NUMERICAL RESULTS

In this section, we assess the performance of Algorithms 2 and 3 on two MPC problems: regulation of an oscillating mass-spring system and quadrotor obstacle avoidance, a quadratic program (QP) and a second-order cone program (SOCP) respectively. We first define the two problems, discuss implementation details of our algorithms, and finally present the numerical results.

A. Oscillating Mass Control

The first MPC problem considers regulation of a one-dimensional mass-spring system to equilibrium by applying forces to each of the masses [18]. In this problem there are N masses connected to their neighbors by springs with the masses at the end being connected to a wall with springs.

The *state* at time-step t is given by $x_t = (r_t, v_t)$, where $r_t \in \mathbb{R}^N$ is the displacement of the masses from their respective equilibrium positions, and $v_t \in \mathbb{R}^N$ is the velocity of the masses. The *control* at time-step t, $u_t \in \mathbb{R}^N$, is the input forces acting on each of the N masses. We constrain the displacement of the masses (9d), the speed of the masses (9e), and the force applied to each mass (9f). We discretize

the dynamics with a zero-order-hold and sample-time of Δt and can write the problem as follows

$$\underset{x_t, u_t}{\text{minimize}} \quad \frac{1}{2}\left(\sum_{t=1}^{T} x_t^{\top} Q_t x_t + \sum_{t=1}^{T-1} u_t^{\top} R_t u_t\right) \tag{9a}$$

$$\text{subject to} \quad x_{t+1} = Ax_t + Bu_t \quad t \in [1, T-1] \tag{9b}$$

$$x_1 = x_{\text{init}} \tag{9c}$$

$$\|r_t\|_{\infty} \leqslant r_{\max} \qquad t \in [1, T] \tag{9d}$$

$$\|v_t\|_{\infty} \leqslant v_{\max} \qquad t \in [1, T] \tag{9e}$$

$$\|u_t\|_{\infty} \leqslant u_{\max} \qquad t \in [1, T-1]. \tag{9f}$$

All the masses have unit mass, all the springs have unit spring constant, the following non-dimensional parameters are used,

$$T = 30, \ \Delta t = 0.1, \ N = 8,$$
$$r_{\max} = 0.75, \ v_{\max} = 0.75, \ u_{\max} = 0.5$$
$$Q_t = \text{blkdiag}(I_N, 5I_N), \ R_t = I_N.$$

Each component of the initial state x_{init} is drawn from the uniform distribution on the interval [-0.5, 0.5].

The dynamics given by Equation 9b can be written as Constraint 1b with appropriate choice of H and g. Constraints 9d, 9e, and 9f can be written as Constraint 1c, where $\mathbb{D}$ is the Cartesian product of boxes.

B. Quadrotor Path Planning

The second MPC problem considered is the three-degree-of-freedom quadrotor path planning problem with obstacle avoidance. The *state* at time-step t is defined as $x_t = (r_t, v_t)$, where $r_t \in \mathbb{R}^3$ is the position of the quadrotor, and $v_t \in \mathbb{R}^3$ is the velocity of the quadrotor. The *control* at time t, $u_t \in \mathbb{R}^3$, is the thrust produced by the quadrotor's propellers.

Unlike the oscillating masses regulation problem, where the goal is to regulate all the masses to their equilibrium positions, the objective in this problem is to track a reference trajectory given by $\hat{x}_t$ which linearly interpolates the initial state x_{init} and the desired target state x_{target}. However, this reference trajectory intersects the obstacle, forcing the optimizer to compute a trajectory that avoids that obstacle. The dynamics of this system are given by a double integrator in three dimensions subject to gravity. These dynamics are discretized with a zero-order-hold to obtain the discrete-time linear system given by 10b, where

$$A = \begin{bmatrix} I_3 & (\Delta t)I_3 \\ 0_3 & I_3 \end{bmatrix}$$

$$B = \frac{1}{m}\begin{bmatrix} (1/2)(\Delta t)^2 I_3 \\ (\Delta t)I_3 \end{bmatrix}$$

$$c = \begin{bmatrix} 0_{2 \times 1} \\ -(1/2)g(\Delta t)^2 \\ 0_{2 \times 1} \\ -g(\Delta t) \end{bmatrix}$$

$g = 9.8$ is the non-dimensional gravitational acceleration and m is the mass of the quadrotor.

We impose two control constraints: an upper bound on thrust (10f) and a thrust pointing constraint (10g). The thrust pointing constraint is a proxy to constrain the tilt angle of the quadrotor, since we are using a three-degree-of-freedom model and cannot directly constrain the quadrotor's tilt.

A cylindrical keep-out zone can be written as

$$\|r_t^{1:2} - r_c\|_2 \geqslant \rho$$

where r_c is the position of the center of the obstacle and ρ is the radius of the obstacle. This constraint, however, is nonconvex.

This nonconvex keep-out-zone constraint can be replaced with a convex rotating half-space constraint, as in [9]. This rotating half-space constraint is written as 10d with

$$a_t = [\cos(\psi t + \phi) \ -\sin(\psi t + \phi)]^\top$$
$$b_t = -a_t^\top r_c - \rho$$

where ψ is the rotation rate of the half-space and ϕ is the initial phase offset of the half-space. This problem can be written as

$$\underset{x_t, u_t}{\text{minimize}} \ \frac{1}{2} \left(\sum_{t=1}^{T} (x_t - \hat{x}_t)^\top Q_t (x_t - \hat{x}_t) \right. \tag{10a}$$
$$\left. + \sum_{t=1}^{T-1} u_t^\top R_t u_t \right)$$

$$\text{subject to } x_{t+1} = Ax_t + Bu_t + c \quad t \in [1, T-1] \ \text{(10b)}$$
$$x_1 = x_{\text{init}} \tag{10c}$$
$$a_t^\top r_t \leqslant b_t \quad t \in [1, T] \ \text{(10d)}$$
$$\|v_t\|_2 \leqslant v_{\max} \quad t \in [1, T] \ \text{(10e)}$$
$$\|u_t\|_2 \leqslant u_{\max} \quad t \in [1, T-1] \ \text{(10f)}$$
$$\cos(\theta_{\max})\|u_t\|_2 \leqslant u_t^\top e \quad t \in [1, T-1] \ \text{(10g)}$$

where $e = [0 \ 0 \ 1]^\top$. Additionally, the following non-dimensional parameters are used

$$T = 30, \ \Delta t = 0.2, \ m = 3, \ \psi = -0.5$$
$$\phi = -\pi/4, \ r_c = [2.5 \ 2.5]^\top, \ \rho = 0.25$$
$$v_{\max} = 1.5, \ u_{\max} = 35, \ \theta_{\max} = 0.1745$$
$$Q_t = \text{blkdiag}(2I_3, I_3), \ R_t = 0.5I_3$$
$$x_{\text{init}} = [0 \ 0 \ 5 \ 0 \ 0 \ 0], \ x_{\text{target}} = [5 \ 5 \ 5 \ 0 \ 0 \ 0].$$

With proper choice for H and g, we can rewrite the dynamics given by Constraint 10b in the form of Constraint 1b. Constraint 10d represents a half-space, Constraint 10e represents a ball, and Constraints 10f and 10g represent the intersection of a ball and a second-order cone. We can write all of these constraints as Constraint 1c, where $\mathbb{D}$ is the Cartesian product of the aforementioned sets.

C. Implementation Details

We generate our numerical results using MATLAB. From Algorithm 2, we determine that the maximum eigenvalue of $H^\top H$ is η^2. If Algorithm 2 is not used, we use power iteration to compute the maximum eigenvalue of $H^\top H$ [16]. To compute maximum and minimum eigenvalues of P, $\lambda_{\max}$ and $\lambda_{\min}$ respectively, we simply take the maximum and minimum diagonal elements of P since P is diagonal in our examples. For general $P > 0$, we can compute the maximum eigenvalue with power iteration and minimum eigenvalue with either shifted power iteration or inverse power iteration. In all examples, we use $k_{\text{update}} = 25$ when testing Algorithm 3. When running examples without using Algorithm 3, we set $\gamma = \sigma$ which makes the dual step-size $\beta = 1$ consistent with the implementation in [8]. We run the oscillating-mass problem 50 times with 50 different initial conditions drawn from the distribution described in the previous section.

Figures 3 and 4 depict the sparsity pattern of H for MPC problems before and after applying the QR preconditioner. Applying the preconditioner results in significant fill-in, i.e., the introduction of nonzero elements in a sparse matrix where they did not exist previously. In the PIPG implementation, H is stored as a sparse matrix when the QR preconditioner is not applied, but H is stored as a dense matrix when the preconditioner is applied. This choice of data structure results in faster matrix-vector multiplication for each case. Without preconditioning, the percentage of nonzero elements in H is small enough that sparse matrix vector multiplication is faster than dense matrix vector multiplication. However, after applying the QR preconditioner, H contains too high of a percentage of nonzero elements for sparse matrix-vector multiplication to be faster than dense matrix-vector multiplication. Consequently, the QR preconditioner is limited to problems and embedded systems where the extra storage requirement for the preconditioned, dense H matrix is not restrictive. Before preconditioning, when we store H for both problems in compressed sparse column (CSC), it takes roughly 187 kilobytes for the oscillating masses problem and 11.6 kilobytes for the quadrotor problem. After applying our preconditioner and storing the preconditioned matrix as a dense matrix, it takes roughly 2581 kilobytes for the oscillating masses problem and 363 kilobytes of storage for the quadrotor problem. As the problem size increases, the storage requirements for the dense matrix become more demanding.

D. Analysis of results

We assess the effect of the our algorithms using two metrics: relative distance to optimum and feasibility as defined below

$$\text{error}_{\text{opt}} = \frac{\|z^k - z^\star\|_\infty}{\|z^\star\|_\infty}, \ \text{error}_{\text{feas}} = \frac{\|Hz^k - g\|_\infty}{\|z^\star\|_\infty}$$

where $z^\star$ is the solution to 1 computed by Gurobi [19] using YALMIP [20]. We terminate PIPG when the relative error,
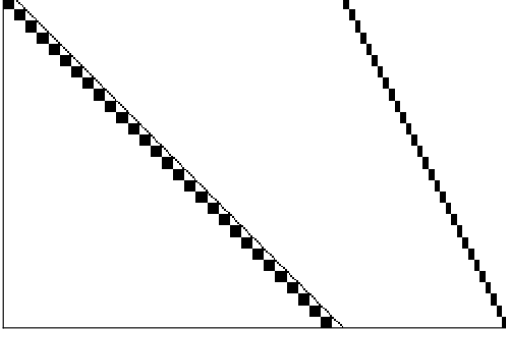
Fig. 3: Sparsity plot of H for the oscillating mass problem

Fig. 4: Sparsity plot of $\hat{H}$ for the oscillating mass problem

error$_{\mathrm{opt}}$, decreases below 1.0×10^{-4}, indicating the solution has a relative error of 0.01%.

For the oscillating masses problem, we test the QR preconditioner against modified Ruiz equilibration. For this problem, modified Ruiz equilibration preserves closed-form projection onto set $\mathbb{D}$ since set $\mathbb{D}$ is a box before and after transformation. Additionally, we test the hypersphere preconditioner (HS) on the quadrotor problem, since in this case, the transformed set $\mathbb{D}$ still has a closed-form projection. If our state cost matrix Q_t penalized each component of velocity by different weights or the control cost matrix R_t penalized each component of thrust with different weights, the HS preconditioner will induce a change of variables which will result in the transformed set $\mathbb{D}$ containing ellipsoids and no longer having closed-forms projections. The hypersphere preconditioner involves multiplying the preconditioned cost function by some $\lambda > 0$ similar to how the QR preconditioner multiplies the equality constraints by some $\eta > 0$, however the authors do not provide a way of selecting this λ [7]. In our numerical results, we choose the λ for the hypersphere preconditioner such that the condition number of the KKT matrix given by Equation 4 is minimized.

Figure 5 depicts the distribution of the absolute value of the eigenvalues for the KKT matrix of the oscillating masses problem and quadrotor problem before and after applying the QR preconditioner. Applying the preconditioner shrinks the interval on which the eigenvalues are distributed, making the maximum and minimum eigenvalues closer in magnitude to each other, reducing the condition number of the KKT matrix.

Figures 6 and 7 show that the QR preconditioner and the step-size heuristic cause PIPG to converge in fewer iterations when applied independently. When applied together, we observed a further reduction in iteration count. Applying the QR preconditioner also results in fewer iterations to convergence than applying modified Ruiz equilibration or the hypersphere preconditioner. Note that for the oscillating-mass problem, PIPG without step-selection and the QR preconditioner requires many more than 1500 iterations to meet our stopping criteria, but we only plotted the first 1500 iterations for clarity.

Tables I and II contain the solve-times for our numerical examples. The solve-times in the table do not include preconditioning time for any of the preconditioners, since for MPC problems we can perform the preconditioning offline and only have to incur cost of solving the preconditioned problem. We observe that the time to perform the QR preconditioning was very small for the problems we considered: 11.9 and 1.75 milliseconds for the oscillating masses problem and quadrotor problem respectively. Our two proposed algorithms result in solve-time reductions, and we see that the QR preconditioner is more effective at reducing solve-times than modified Ruiz equilibration and the hypersphere preconditioner.
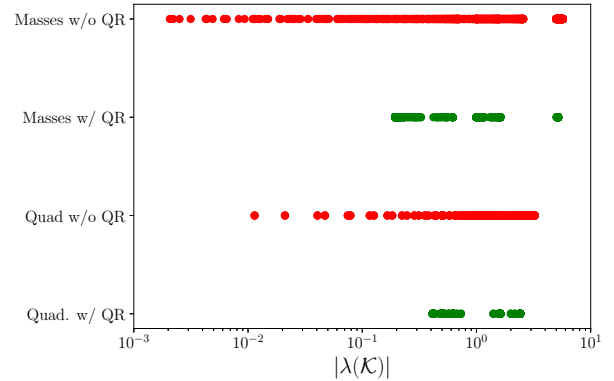

Fig. 5: Eigenvalue distribution of the KKT matrix for the oscillating masses and quadrotor problems with and without applying the QR preconditioner

	Without Precond.	With Ruiz	With QR
Without Step Select.	651.73	562.80	23.96
With Step Select.	50.85	60.71	**18.69**

TABLE I: Average solve-time in milliseconds over 50 runs for the oscillating masses problem
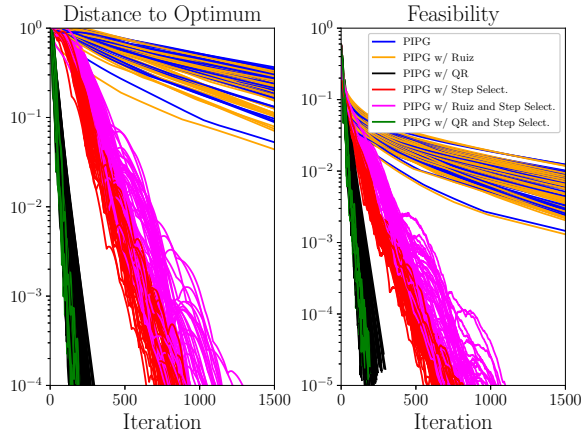
Fig. 6: $\mathrm{error}_{\mathrm{opt}}$ (left) and $\mathrm{error}_{\mathrm{feas}}$ (right) for the oscillating masses problem
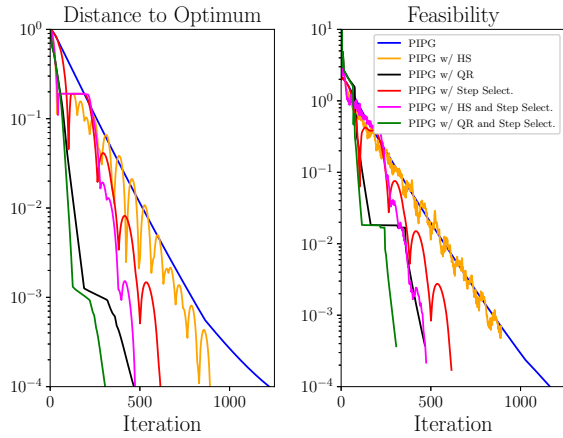

Fig. 7: $\mathrm{error}_{\mathrm{opt}}$ (left) and $\mathrm{error}_{\mathrm{feas}}$ (right) for the quadrotor problem

	Without Precond.	With HS	With QR
Without Step Select.	226.92	169.50	93.68
With Step Select.	114.91	91.80	**62.04**

TABLE II: Average solve-time in milliseconds over 100 runs for the quadcopter problem

VI. CONCLUSION

We develop the *QR preconditioner* to speed up PIPG, a first-order primal-dual algorithm, by applying a transformation to equality constraints which minimizes the condition number of the KKT matrix for the optimization problem. We then introduce a step-size heuristic for PIPG based on minimizing the upper bound on the primal-dual gap. The effectiveness of these algorithms are demonstrated on two MPC problems, where we provide comparisons to existing preconditioning algorithms.

REFERENCES

[1] Heinz H. Bauschke and Patrick L. Combettes. Convex analysis and monotone operator theory in hilbert spaces. In *CMS Books in Mathematics*, 2011.

[2] Heinz H. Bauschke, Minh N. Bui, and Xianfu Wang. Projecting onto the intersection of a cone and a sphere. *SIAM Journal on Optimization*, 28(3):2158–2188, 2018.

[3] Juan L. Jerez, Paul J. Goulart, Stefan Richter, George A. Constantinides, Eric C. Kerrigan, and Manfred Morari. Embedded online optimization for model predictive control at megahertz rates. *IEEE Transactions on Automatic Control*, 59(12):3238–3251, 2014.

[4] Goran Banjac, Bartolomeo Stellato, Nicholas Moehle, Paul Goulart, Alberto Bemporad, and Stephen Boyd. Embedded code generation using the osqp solver. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 1906–1911, 2017.

[5] Euhanna Ghadimi, André Teixeira, Iman Shames, and Mikael Johansson. Optimal parameter selection for the alternating direction method of multipliers (admm): Quadratic problems. *IEEE Transactions on Automatic Control*, 60(3):644–658, 2015.

[6] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd. OSQP: an operator splitting solver for quadratic programs. *Mathematical Programming Computation*, 12(4):637–672, 2020.

[7] Abhinav G Kamath, Purnanand Elango, Taewan Kim, Skye Mceowen, Yue Yu, John M Carson III, Mehran Mesbahi, and Behçet Açıkmeşe. Customized real-time first-order methods for onboard dual quaternion-based 6-dof powered-descent guidance. In *AIAA SciTech 2023 Forum*, 2023.

[8] Yue Yu, Purnanand Elango, Ufuk Topcu, and Behçet Açıkmeşe. Proportional–integral projected gradient method for conic optimization. *Automatica*, 142:110359, 2022.

[9] Yue Yu, Purnanand Elango, and Behçet Açıkmeşe. Proportional-integral projected gradient method for model predictive control. *IEEE Control Systems Letters*, 5(6):2174–2179, 2021.

[10] Yue Yu, Purnanand Elango, Behçet Açıkmeşe, and Ufuk Topcu. Extrapolated proportional-integral projected gradient method for conic optimization. *IEEE Control Systems Letters*, 7:73–78, 2023.

[11] Purnanand Elango, Abhinav G Kamath, Yue Yu, John M Carson III, Mehran Mesbahi, and Behçet Açıkmeşe. A customized first-order solver for real-time powered-descent guidance. In *AIAA SciTech 2022 Forum*, page 0951, 2022.

[12] Abhinav G. Kamath, Purnanand Elango, Yue Yu, Skye Mceowen, Govind M. Chari, John M. Carson III, and Behçet Açıkmeşe. Real-time sequential conic optimization for multi-phase rocket landing guidance. *IFAC-PapersOnLine*, 56(2):3118–3125, 2023. 22nd IFAC World Congress.

[13] Govind M. Chari, Abhinav G. Kamath, Purnanand Elango, and Behcet Acikmese. Fast monte carlo analysis for 6-dof powered-descent guidance via gpu-accelerated sequential convex programming. In *AIAA SciTech 2024 Forum*, 2024.

[14] Govind M Chari and Behçet Açıkmeşe. Spacecraft rendezvous guidance via factorization-free sequential convex programming using a first-order method. *arXiv preprint arXiv:2402.04561*, 2024.

[15] Michele Benzi, Gene H. Golub, and Jörg Liesen. Numerical solution of saddle point problems. *Acta Numerica*, 14:1–137, 2005.

[16] Gene H Golub and Charles F Van Loan. *Matrix Computations*. Johns Hopkins Series in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD, 3 edition, October 1996.

[17] Jorge Nocedal and Stephen Wright. *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer, New York, NY, 2 edition, July 2006.

[18] Yang Wang and Stephen Boyd. Fast model predictive control using online optimization. *IEEE Transactions on Control Systems Technology*, 18(2):267–278, 2010.

[19] Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2023.

[20] J. Löfberg. Yalmip : A toolbox for modeling and optimization in matlab. In *In Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004.