Newton-PIPG: A Hybrid Algorithm for Optimal Control Quadratic Programming Optimization [⋆]

Dayou Luo [a], Yue Yu [b], Maryam Fazel [c] and Behçet Açıkmeşe [d]

[a] *Department of Applied Mathematics, University of Washington, Seattle, WA, 98195*

[b] *Department of Aerospace Engineering and Mechanics, University of Minnesota Twin Cities, Minneapolis, MN 55455.*

[c] *Department of Electrical and Computer Engineering, University of Washington, Seattle, WA, 98195*

[d] *William E. Boeing Department of Aeronautics and Astronautics, University of Washington, Seattle, WA, 98195*

Abstract

We propose Newton-PIPG, an efficient method for solving quadratic programming (QP) problems arising in optimal control, subject to additional set constraints. Newton-PIPG integrates the Proportional-Integral Projected Gradient (PIPG) method with the Newton method, thereby achieving both global convergence and local quadratic convergence. The PIPG method, an operator-splitting algorithm, seeks a fixed point of the PIPG operator. Under mild assumptions, we demonstrate that this operator is locally smooth, enabling the application of the Newton method to solve the corresponding nonlinear fixed-point equation. Furthermore, we prove that the linear system associated with the Newton method is locally nonsingular under strict complementarity conditions. To enhance efficiency, we design a specialized matrix factorization technique that leverages the typical sparsity of optimal control problems in such systems. Numerical experiments demonstrate that Newton-PIPG achieves high accuracy and reduces computation time, particularly when feasibility is easily guaranteed.

1 Introduction

The development of techniques for solving Quadratic programming (QP) problems is a key enabler for advancing optimal control research, as efficiently solving these problems is often essential and a bottleneck in real-world optimal control applications. For example, Model Predictive Control (MPC), a framework for implementing optimal control, often relies on solving a series of QP problems [Houska et al., 2011, Jones et al., 2012, Zometa et al., 2013].In addition, Sequential Convex Programming (SCP), a widely used algorithm for nonlinear optimal control, also relies on solving QP problems [Neunert et al., 2016, Szmuk et al., 2020, Malyuta et al., 2022, Elango et al., 2024]. Fast QP solvers are particularly beneficial in these applications, especially in scenarios requiring real-time implementation [Malyuta et al., 2022, Kamath et al., 2023].

We consider the following QP problem, which includes additional set constraints and frequently arises in optimal control applications:

Problem 1:

$$
\begin{array}{ll}
\text{minimize} & \sum_{i=0}^{N+1}\sum_{j=1}^{m_i}\frac{1}{2}\rho_{ij}\left\|z_{ij}\right\|^2+q^Tz\\
\text{Subject to} & H_Ez+g_E=0,\\
& H_Iz+g_I\succeq 0,\\
& z_{ij}\in\mathbb{D}_{ij}=\left\{x\in R^{n_{ij}}\mid \Phi_{ij}(x)\preceq 0\right\},
\end{array}
\tag{1}
$$

where $z_{ij}\in\mathbb{R}^{n_{z_{ij}}}$, $z_i=(z_{i1},z_{i2},\cdots,z_{im_i})\in\mathbb{R}^{n_{z_i}}$, and $z=(z_0,z_1,\ldots,z_{N+1})\in\mathbb{R}^{n_z}$. Here, z_i represents the combination of state and input variables at the i-th time grid point. The vector z_i consists of m_i components, with each component z_{ij} has a consistent quadratic weight $\rho_{ij}>0$, The notations $\preceq$ and $\succeq$ denote element-wise inequalities for vectors. Vectors $q\in\mathbb{R}^{n_z}$, $g_E\in\mathbb{R}^{n_E}$, and $g_I\in\mathbb{R}^{n_I}$, as well as matrices $H_E\in\mathbb{R}^{n_E\times n_z}$ and $H_I\in\mathbb{R}^{n_I\times n_z}$, are constants. The function Φ_{ij} is a smooth vector-valued function on $\mathbb{R}^{n_{z_{ij}}}$, defining the convex set $\mathbb{D}_{ij}$ for additional constraints. We will later add requirements on Φ_{ij} to include commonly used sets, such as balls, second-order cones, boxes, and half-spaces.

Additionally, we require that both H_E and H_I follows the pattern below:

⋆ Corresponding author Dayou Luo.

$$\begin{pmatrix} A_0 & B_0 & 0 & \cdots & \cdots & 0 & 0 \\ 0 & A_1 & B_1 & 0 & & \vdots & \vdots \\ 0 & 0 & A_2 & B_2 & \ddots & \vdots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & 0 & 0 \\ \vdots & & & \ddots & A_{N-1} & B_{N-1} & 0 \\ 0 & \cdots & \cdots & 0 & 0 & A_N & B_N \end{pmatrix}$$

where A_i and B_i are constant matrices. The number of columns of A_i is n_{z_i} and that of B_i is $n_{z_{i+1}}$. The number of rows for A_i and B_i are determined by the linear equality $H_E z + g_E = 0$ and the inequality $H_I z + g_I \geq 0$. We denote these numbers as n_{E_i} for the equalities and n_{I_i} for the inequalities in the i-th row of the block matrices H_E and H_I, respectively.

The sparsity patterns of H_E and H_I are typical for optimal control optimization problems. These matrices define equality and inequality constraints among consecutive time grid points, while the set $\mathbb{D}_{ij}$ includes constraints for states and inputs at each time grid point.

There are a few differences between Problem 1 and general optimal control QP problems. First, our objective function requires strong convexity, and the quadratic term must be homogeneous with respect to the constraint $\mathbb{D}_{ij}$. Specifically, for dimensions of z constrained by the same set $\mathbb{D}_{ij}$, the corresponding quadratic cost coefficients must be identical. Second, we restrict the set $\mathbb{D}_{ij}$ to specific families of convex sets. Possible relaxations of these requirements are discussed in Section 5.

Widely used methods for solving Problem 1 fall into two primary categories: second-order methods and operator-splitting methods, each with their own advantages and disadvantages. Second-order methods, such as interior-point methods and active-set methods, utilize the curvature information of the optimization problem. These methods typically converge in a modest number of iterations and are robust when the problem approaches infeasibility. Commonly used software in this category includes MOSEK[Andersen and Andersen, 2000], Gurobi[Gurobi Optimization, LLC, 2024], and ECOS[Domahidi et al., 2013]. For further theoretical discussions on second-order methods, we refer the reader to [Andersen and Andersen, 2000, Boyd and Vandenberghe, 2004, Nocedal and Wright, 1999].

One drawback of second-order methods lies in their high computational cost per iteration, as each step requires solving a linear system. This issue becomes significant as the dimension of the optimization variables increases. Additionally, while interior-point solvers are generally effective, they typically cannot directly handle Problem 1 in its original form and require the use of parsers, such as Yamlip [Löfberg, 2004] or CVX [Grant and Boyd, 2014], to transform the problem into a solver-compatible form. This transformation process adds overhead and increases the effort needed for verification, validation, and maintenance.

Another approach to solving Problem 1 is the operator-splitting method, which constructs an update operator applied at each iteration, ensuring that each iteration converges to the fixed point of this operator. Unlike interior-point methods, operator-splitting methods eliminate the need for matrix factorization, resulting in significantly lower per-iteration computational costs. Additionally, these methods do not require external parsers and are typically implemented with a smaller codebase [Kamath et al., 2023]. For a comprehensive review of operator-splitting methods, see [Jiang and Vandenberghe, 2023].

However, one drawback of operator-splitting methods is that they often require a large number of iterations to converge, especially when the problem is ill-conditioned.

Our main contribution is the development of Newton-PIPG, a hybrid method that integrates the Proportional-Integral Projected Gradient (PIPG) method [Yu et al., 2020, Yu et al., 2022], a representative operator-splitting approach, with the Newton method. This approach is motivated by the insight that finding a fixed point for the updating operator in operator-splitting methods is equivalent to solving a nonlinear fixed point equation. Given the Newton method's effectiveness in solving such equations, we design a Newton step for this fixed-point problem, with PIPG providing a warm start to ensure global convergence. Compared to second-order methods, Newton-PIPG reduces the need for matrix factorization and avoids the requirement for additional parsers. Additionally, in contrast to operator-splitting methods, the Newton step reduces the number of iterations required, enhancing both accuracy and speed.

We also provide theoretical guarantees for the Newton-PIPG method. Specifically, we demonstrate that under an extended linear independence constraint qualification and strict complementarity condition, the Newton step is well-defined, the associated matrix is nonsingular, and the Newton-PIPG algorithm enjoys global convergence with local quadratic convergence behavior.

Furthermore, we tailor our algorithm for optimal control problems. Recognizing that the most computationally expensive step is solving a large linear system in the Newton method, we exploit the sparsity inherent in optimal control problems to design an efficient factorization strategy, significantly reducing computation time.

1.1 Related Work

Several studies have focused on integrating operator-splitting methods with Newton-type methods. These approaches include directly using BFGS [Dennis Jr and Schnabel, 1996] to solve the fixed-point equation from the proximal point method [Chen and Fukushima, 1999], and employing the ADMM method in conjunction with a semi-smooth Newton method [Ali et al., 2017], where linear systems are solved using GMRES

[Saad and Schultz, 1986]. Other approaches involve hybridizing operator-splitting methods with quasi-Newton methods to balance global convergence and local acceleration. Popular choices for quasi-Newton methods include BFGS [Themelis and Patrinos, 2019, Sopasakis et al., 2019] and Anderson acceleration [Zhang et al., 2020].

The main difference between our work and others is that we focus on a specific type of optimal control QP problem. This focus provides several advantages for our method. First, we demonstrate that, under mild assumption, the update operator in PIPG is differentiable in a neighborhood of the solution. This suggests that the semi-smooth Newton method essentially acts as a Newton method, offering a quadratic convergence guarantee. Details of this differentiability are presented in Theorem 14.

Second, rather than using quasi-Newton methods or iterative solvers, we directly handle the linear system in the Newton step using an efficient matrix factorization technique. This approach leverages the sparsity pattern of the matrix, resulting in faster computation.

Third, our algorithm offers a stronger theoretical guarantee. We prove, rather than assume as in [Ali et al., 2017, Assumption A6], that the matrix involved in the Newton step is nonsingular in a local neighborhood of the solution under a generalized Linearly Independent Constraint Qualification (LICQ) assumption, which ensures the robustness of the algorithm. Moreover, our algorithm guarantees convergence even when certain key assumptions are not satisfied.

In addition to its relationship with previously discussed hybrid methods, we view Newton-PIPG as a generalization of the algorithm in [Wang and Boyd, 2009], as the matrix factorization technique used in the Newton step is inspired by [Wang and Boyd, 2009]. Other methods, such as [Wright, 1993], address similar matrices using different linear algebra techniques while maintaining the same complexity order. A key distinction between our approach and those in [Wang and Boyd, 2009, Wright, 1993] is our use of operator-splitting methods. This strategy not only ensures the applicability of our method to QPs with additional set constraints but also reduces the amount of matrix factorizations required for convergence.

1.2 Notation

We adopt the following notation in the remainder of the discussion. We denote the set of real numbers by $\mathbb{R}$, the set of nonnegative real numbers by $\mathbb{R}_+$, the set of nonpositive real numbers by $\mathbb{R}_-$, the set of real $n \times m$ matrices by $\mathbb{R}^{n\times m}$, and the set of real n-dimensional vectors by $\mathbb{R}^n$. We denote the identity matrix by I and the origin of $\mathbb{R}^n$ by 0^n. For two matrices/vectors A and B, we use the notation $[A, B]$ for a matrix formed by placing A and B horizontally, given that A and B have the same number of rows. We use $[A; B] = (A^\top, B^\top)^\top$ for stacking A and B vertically. Such notation naturally extends to cases of combining multiple matrices. For vectors, we further simplify the notation and denote $(v_1, v_2, \ldots, v_n) = [v_1; v_2; \ldots; v_n]$ as the vertical stacking of vectors $v_1, v_2, \ldots, v_n$ to form one vector. For the i-th element of a vector v, we use the notation $v_{(i)} \in \mathbb{R}$. For the element on the i-th row and the j-th column of a matrix M, we denote it as $M_{(i,j)} \in \mathbb{R}$. We define vector inequalities as element-wise comparisons. For example, for vectors a and b, $a \preceq b$ indicates that $a_{(i)} \leq b_{(i)}$ for all i, and $a \succeq b$ indicates that $a_{(i)} \geq b_{(i)}$ for all i. We use $\operatorname{blkdiag}(A_1, A_2, \cdots, A_n)$ to build a block diagonal matrix with A_1 to A_n as the diagonal block matrices. $\|\cdot\|$ denotes the Euclidean norm for vectors and matrices. For a convex set $\mathbb{D}$, we denote $\Pi_{\mathbb{D}}(z)$ as the projection of z onto set $\mathbb{D}$ and for $z \in \mathbb{D}$, we denote the normal cone of $\mathbb{D}$ at the point z as $N_{\mathbb{D}}(z)$. For the definition of the normal cone, we refer to [Borwein and Lewis, 2006, Chapter 2.1]. For general closed set $\mathbb{D}$, we denote the distance of a point z_0 to $\mathbb{D}$ as $d_{\mathbb{D}} = \min\{\|z - z_0\| | z \in \mathbb{D}\}$, and we use $\operatorname{ri}\mathbb{D}$ as the relative interior of $\mathbb{D}$. We denote $\mathbb{D}_1 \times \mathbb{D}_2$ as the direct product of set $\{\mathbb{D}_1, \mathbb{D}_2\}$, and $\prod_{i=0}^N \mathbb{D}_i$ as the direct product of set $\{\mathbb{D}_1, \mathbb{D}_2, \cdots, \mathbb{D}_N\}$. We use the notation $o(\epsilon)$ to denote a function $f(\epsilon)$ that $\lim_{\epsilon\to 0} \frac{f(\epsilon)}{\epsilon} = 0$. We use $[1, n]_{\mathbb{N}}$ for the integer set $\{1, 2, \cdots, n\}$. We denote $\operatorname{span}(v)$ as the set of all linear combinations of the vectors in the set v.

2 Background

To simplify the notation, we rewrite Problem 1 as follows **Problem 2**:

$$\begin{aligned} \underset{z}{\text{minimize}} \quad & \tfrac{1}{2} z^\top P z + q^\top z \\ \text{subject to} \quad & Hz - g \in \mathbb{K}, \quad z \in \mathbb{D}, \end{aligned}$$

Here, the matrix H is constructed by rearranging the rows of the matrix $[H_E; H_I]$, resulting in the following form:

$$H = \begin{pmatrix} A_0 & B_0 & 0 & \cdots & \cdots & 0 & 0 \\ 0 & A_1 & B_1 & 0 & & \vdots & \vdots \\ 0 & 0 & A_2 & B_2 & \ddots & \vdots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & 0 & 0 \\ \vdots & & & \ddots & A_{N-1} & B_{N-1} & 0 \\ 0 & \cdots & \cdots & 0 & 0 & A_N & B_N \end{pmatrix}. \tag{2}$$

In this formulation, the notation (A_i, B_i) is reused to represent the block matrices that combine the original blocks from H_E and H_I, corresponding to the equality and inequality constraint coefficients at the i-th time grid point. Setting $n_i = n_{E_i} + n_{I_i}$, the matrices $A_i \in \mathbb{R}^{n_i \times n_{z_i}}$ and $B_i \in \mathbb{R}^{n_i \times n_{z_{i+1}}}$. The cone $\mathbb{K}$ is defined as

$\prod_{i=0}^{N}(0^{n_{E_i}} \times \mathbb{R}_+^{n_{I_i}})$, and the vector g is permuted from $[g_E; g_I]$, ensuring that $Hz - g \in \mathbb{K}$ enforces both $H_E z + g_E = 0$ and $H_I z + g_I \geq 0$. From now on, A_i and B_i refer to the combined blocks in H.

The set $\mathbb{D}$ is defined as $\prod_{i=0}^{N+1}\prod_{j=0}^{m_i}\mathbb{D}_{ij}$. The dual cone $\mathbb{K}^\circ$ is $\prod_{i=0}^{N}(\mathbb{R}^{n_{E_i}} \times \mathbb{R}_-^{n_{I_i}})$. The dimension of $\mathbb{K}$ is denoted by $n_w = \sum_{i=0}^{N} n_{E_i} + n_{I_i}$.

2.1 Proportional-Integral Projected Gradient

As discussed in the introduction, we propose a new method that combines the Proportional-Integral Projected Gradient (PIPG) method with the Newton method. In this section, we provide a brief overview of the PIPG method.

Developed in [Yu et al., 2022], the PIPG algorithm's update rule for Problem 2 is defined as:

$$\begin{aligned} z^{k+1} &= \pi_{\mathbb{D}}\left[z^k - \alpha\left(Pz^k + q + H^\top w^k\right)\right] \\ w^{k+1} &= \pi_{\mathbb{K}^\circ}\left[w^k + \beta\left(H\left(2z^{k+1} - z^k\right) - g\right)\right] \end{aligned} \tag{3}$$

where $\pi_{\mathbb{D}} : \mathbb{R}^{n_z} \to \mathbb{R}^{n_z}$ is the projection operator on to set $\mathbb{D}$, and $\pi_{\mathbb{K}^\circ} : \mathbb{R}^{n_w} \to \mathbb{R}^{n_w}$ is the projection operator on to set $\mathbb{K}^\circ = \prod_{i=0}^{N}\mathbb{R}^{n_{E_i}} \times \mathbb{R}_-^{n_{I_i}}$. We denote the operator $T : \mathbb{R}^{n_z} \times \mathbb{R}^{n_w} \to \mathbb{R}^{n_z+n_w}$ for the PIPG operator and express the PIPG update in (3) as

$$(z^{k+1}, w^{k+1}) = T(z^k, w^k). \tag{4}$$

Let $\mathrm{Fix}(T)$ denote the set of fixed points of operator T, that is, $\mathrm{Fix}(T) = \{(z, w) \in \mathbb{R}^{n_z+n_w} \mid (z, w) = T(z, w)\}$. The following theorem for the convergence of the PIPG method is a direct consequence of Theorem 4, Theorem 17, and Theorem 18, discussed later in Section 4:

Theorem 1 *Suppose (z^k, w^k) is computed as (4), $\alpha, \beta > 0$ and $\alpha\|P\| + \alpha\beta\|H\|^2 < 1$. If $\mathrm{Fix}(T)$ is nonempty, the sequence (z^k, w^k) will converge to a point $(z^\star, w^\star) \in \mathrm{Fix}(T)$, and $z^\star$ is a solution of Problem 2.*

For a fixed point $(z^\star, w^\star) \in \mathrm{Fix}(T)$, we denote $z^\star$ as a primal solution and $w^\star$ as a dual solution to Problem 2. The pair $(z^\star, w^\star)$ will be referred to as a primal-dual solution pair to Problem 2.

2.2 Newton Method

Theorem 1 indicates that PIPG solves a nonlinear fixed-point equation $(z^\star, w^\star) - T(z^\star, w^\star) = 0$. Consequently, it is natural to consider applying the Newton method directly to solve this equation, given its fast local convergence rate [Nocedal and Wright, 1999]. In this section, we provide a brief review of the Newton method.

The Newton method is designed to solve

$$R(x) = 0,$$

where $R : \mathbb{R}^n \to \mathbb{R}^n$ is a continuously differentiable function with a Jacobian matrix $J(x)$. At iteration x_k, a linear approximation for $R(x_k)$ is $R(x_k + p) \approx R(x_k) + J(x_k)p$. The Newton method chooses p_k such that the linear approximation of R equals zero, i.e., $p_k = -J(x_k)^{-1}R(x_k)$. The next iteration x_{k+1} is defined as $x_{k+1} = x_k + p_k$.

The Newton method is known for its locally quadratic convergence rate, but it also has three major disadvantages: the requirement of a nonsingular Jacobian matrix, the high computational cost of computing a factorization of the Jacobian matrix, and the lack of a global convergence guarantee. Additionally, in order to apply the Newton method to the fixed-point equation of T, we need to assess the differentiability of the operator T.

In the next section, we will define the Newton step for the PIPG operator T. We will show that the Jacobian of the PIPG exists and that the matrix appearing in the Newton step is invertible in the neighborhood of T's fixed point under some mild assumptions. Hence, the linear equation in the Newton step has a local solution. Additionally, we will provide a factorization strategy for the Jacobian matrix that leverages its sparse structure. The global convergence result is deferred to Section 4.

3 Newton Step for PIPG

We start with a formal definition of the Newton step for PIPG. Consider the case where both $\pi_{\mathbb{D}}$ and $\pi_{\mathbb{K}^\circ}$ are differentiable. In this case, the Jacobian of the PIPG operator can be computed using the chain rule:

$$J_T := \begin{pmatrix} \frac{\partial z^{k+1}}{\partial z^k} & \frac{\partial z^{k+1}}{\partial w^k} \\ \frac{\partial w^{k+1}}{\partial z^k} & \frac{\partial w^{k+1}}{\partial w^k} \end{pmatrix} \tag{5}$$

where

$$\begin{aligned} \frac{\partial z^{k+1}}{\partial z^k} &= J_{\mathbb{D}}(I - \alpha P), \\ \frac{\partial z^{k+1}}{\partial w^k} &= J_{\mathbb{D}}(-\alpha H^\top), \\ \frac{\partial w^{k+1}}{\partial z^k} &= J_{\mathbb{K}^\circ}\beta H(-I + 2J_{\mathbb{D}}(I - \alpha P)) \\ &= J_{\mathbb{K}^\circ}\beta H(-I + 2\frac{\partial z^{k+1}}{\partial z^k}), \\ \frac{\partial w^{k+1}}{\partial w^k} &= J_{\mathbb{K}^\circ} + 2J_{\mathbb{K}^\circ}\beta H J_{\mathbb{D}}(-\alpha H^\top) \\ &= J_{\mathbb{K}^\circ} + 2J_{\mathbb{K}^\circ}\beta H\frac{\partial z^{k+1}}{\partial w^k}, \end{aligned}$$

and $J_{\mathbb{D}}$ and $J_{\mathbb{K}^\circ}$ are the Jacobian of projection $\pi_{\mathbb{D}}$ and $\pi_{\mathbb{K}^\circ}$, evaluated at $z^k - \alpha\left(Pz^k + q + H^\top w^k\right)$ and $w^k + \beta\left(H\left(2z^{k+1} - z^k\right) - g\right)$ respectively.

If $\pi_{\mathbb{D}}$ and $\pi_{\mathbb{K}^\circ}$ are not differentiable, we will simply set J_T as a matrix where all elements are set to zero. We will later show that regardless of the choice of J_T, the global convergence of our algorithm will not be affected and, with mild assumption, it will also not affect the local convergence rate. Also, since projections are functions that are almost everywhere differentiable, it is not likely to have $\pi_{\mathbb{D}}$ and $\pi_{\mathbb{K}^\circ}$ being non-differentiable at iterations of our algorithm.

With J_T being well-defined for all (z, w), we have the definition of the Newton step

Definition 2 *The update rule for the Newton step is defined as:*

$$(z^{k+1}, w^{k+1}) = (z^k, w^k) + (\Delta z, \Delta w),$$

where $(\Delta z, \Delta w)$ *is the solution of the following linear equation:*

$$(I - J_T(z^k, w^k))(\Delta z, \Delta w) = -(z^k, w^k) + T(z^k, w^k), \quad (6)$$

and $J_T(z^k, w^k)$ *is defined as* (5).

3.1 Extended Linearly Independent Constraint Qualification

In this section, we introduce the constraint qualification for our problem, ensuring the matrix in the Newton step is locally nonsingular. This requires the fixed point set of the PIPG operator to be a single point. The strong convexity of Problem (1) ensures a unique primal solution, while additional constraint qualifications are needed for a unique dual solution.

The linearly independent constraint qualification (LICQ) is commonly used to ensure the uniqueness of the dual variable. However, LICQ is trivially violated for some important cases in Problem 1.

One example where LICQ fails is when $\mathbb{D}_{ij}$ is a second-order cone and the optimal solution $z_{ij} = 0$. Note that the second-order cone is defined as

$$\mathbb{D}_{ij} = \{(x, y) \in \mathbb{R}^{n_{z_{ij}}-1} \times \mathbb{R} \mid |x|^2 - y \leq 0, y \geq 0\},$$

When $x = 0$, the gradient of $|x|^2 - y$ is a zero vector, causing LICQ to automatically fail.

Another example is when $\mathbb{D}_{ij}$ is an affine subspace. According to our definition of $\mathbb{D}_{ij}$, an affine subspace is formed as

$$\{x \in \mathbb{R}^{n_{z_{ij}}} \mid 0 \preceq Ax - b \preceq 0\}.$$

For any point $z_{ij} \in \mathbb{D}_{ij}$, LICQ is automatically violated.

To include both types of set constraints in our discussion, we consider the following extended LICQ:

Definition 3 (Extended-LICQ) *Problem 1 satisfies the extended Linearly Independent Constraint Qualification (extended-LICQ) if*

$$\mathrm{range}([H_E^\top, H_{A_I}^\top]) \bigcap \mathrm{span}(N_{\mathbb{D}}(z^\star)) = \{0\},$$

and the matrix $[H_E^\top, H_{A_I}^\top]$ *has full column rank. Here* $z^\star$ *is the optimal solution of Problem 1 and* H_{A_I} *consists of rows from* H_I *that activate (achieve equality) at* $z^\star$.

The extended-LICQ, or its equivalent forms, are commonly used constraint qualifications for problems that involve second-order cones. For instance, the extended LICQ is equivalent to the second-order constraint qualification defined in [Mordukhovich et al., 2014].

Now, we need to show that under extended-LICQ, all solutions to Problem 2 satisfy the Karush-Kuhn-Tucker (KKT) condition:

Theorem 4 *Under the extended-LICQ assumption,* $z^\star$ *is a solution to Problem 2 if and only if* $z^\star$ *is feasible for Problem 2 and there exists a dual variable* $w^\star \in \mathbb{K}^\circ$ *such that*

$$0 \in Pz^\star + q + H^\top w^\star + N_{\mathbb{D}}(z^\star), \quad (7)$$

and

$$Hz^\star - g \in N_{\mathbb{K}^\circ}(w^\star).$$

Moreover, $w^\star$ *is unique, and* $z^\star$ *is the unique solution to Problem 2.*

Proof: By the definition of $N_{\mathbb{K}^\circ}$, $Hz^\star - g \in N_{\mathbb{K}^\circ}(w^\star)$ is equivalent to

$$\begin{aligned} w^\star &\in \mathbb{K}^\circ, \\ Hz^\star - g &\in \mathbb{K}, \\ w^{\star\top}(Hz^\star - g) &= 0. \end{aligned}$$

For the if part, Equation (7) ensures that $z^\star$ is a minimizer of the function

$$\frac{1}{2} z^{\star\top} P z^\star + q^\top z^\star + w^{\star\top}(Hz^\star - g) + I_{\mathbb{D}}(z^\star).$$

Since $w^\star \in \mathbb{K}^\circ$, $w^{\star\top}(Hz - g)$ is non-positive for all z feasible to Problem 2, and thus

$$\begin{aligned} &\frac{1}{2} z^{\star\top} P z^\star + q^\top z^\star \\ &= \frac{1}{2} z^{\star\top} P z^\star + q^\top z^\star + w^{\star\top}(Hz^\star - g) + I_{\mathbb{D}}(z^\star) \\ &\leq \frac{1}{2} z^\top P z + q^\top z + w^{\star\top}(Hz - g) + I_{\mathbb{D}}(z) \\ &\leq \frac{1}{2} z^\top P z + q^\top z, \end{aligned}$$

for all feasible z. Thus, $z^\star$ is the solution to Problem 2.

For the only if part, [Clarke, 2013, Theorem 10.47] ensures the existence of $w^\star \in \mathbb{K}^\circ$, $\eta = 0$ or 1, and $(w^\star, \eta) \neq 0$ such that

$$0 \in \eta(Pz^\star + q) + H^\top w^\star + N_{\mathbb{D}}(z^\star), \quad (8)$$

and

$$w^{\star\top}(Hz^\star - g) = 0.$$

The extended LICQ ensures that $\eta = 1$: If $\eta = 0$, then $w^\star \neq 0$, meaning $H^\top w^\star$ belongs to range$([H_E^\top, H_{A_I}^\top])$, which violates the extended LICQ due to Equation (8).

□

Remark 5 *In the proof of Theorem 4, the extended LICQ is not required for the if part of the proof. Meanwhile, for the fixed point $(z^\star, w^\star)$ of the PIPG operator, we have*

- $-\left(Pz^\star + q + H^\top w^\star\right) \in N_{\mathbb{D}}(z^\star)$
- $H(z^\star) - g \in N_{\mathbb{K}^\circ}(w^\star)$

Therefore, Theorem 4 indicates that all fixed points $(z^\star, w^\star)$ of the PIPG operator are solutions to Problem 2, with or without the extended-LICQ constraints.

3.2 Properties of Projections to Convex Sets

In this section, we discuss the properties of projections onto convex sets. These properties are useful for subsequent proofs. To start, we discuss the eigenvalues of Jacobians of projection functions.

Lemma 6 *For a convex set $\mathbb{D}$ and a point z outside $\mathbb{D}$, let $\pi(z)$ denote the projection of z onto $\mathbb{D}$. If the Jacobian of this projection, denoted $J(z)$, exists and is symmetric, then the eigenvalues of $J(z)$ are real, non-negative, and do not exceed one.*

Proof: Since $J(z)$ is symmetric, all eigenvalues of $J(z)$ are real numbers.

Suppose that λ is an eigenvalue of $J(z)$, and let v be a corresponding eigenvector. We have

$$\lim_{\varepsilon \to 0} \frac{\pi(z+\varepsilon v) - \pi(z)}{\varepsilon} = J(z)v = \lambda v$$

Thus $\pi(z + \varepsilon v) = \pi(z) + \lambda \varepsilon v + o(\varepsilon)$.

Since $\pi(z) \in \mathbb{D}$, the definition of projection ensures that

$$\|(z+\varepsilon v) - \pi(z+\varepsilon v)\|^2 \leq \|(z+\varepsilon v) - \pi(z)\|^2,$$

Therefore,

$$\|\pi(z+\varepsilon v) - \pi(z) + \pi(z) - (z+\varepsilon v)\|^2 \leq \|z+\varepsilon v - \pi(z)\|^2,$$

and hence

$$\|\pi(z+\varepsilon v) - \pi(z)\|^2 + 2\langle \pi(z+\varepsilon v) - \pi(z), \pi(z) - z - \varepsilon v\rangle \leq 0$$

Thus

$$\begin{aligned} &\|\lambda\varepsilon v + o(\varepsilon)\|^2 + 2\langle \lambda\varepsilon v + o(\varepsilon), -\varepsilon v\rangle \\ &\leq 2\langle \pi(z+\varepsilon v) - \pi(z), z - \pi(z)\rangle \end{aligned}$$

By the convexity of set $\mathbb{D}$, $\langle \pi(z+\varepsilon v) - \pi(z), z - \pi(z)\rangle \leq 0$([Clarke, 2013, Proposition7.4]). We have

$$\|\lambda\varepsilon v + o(\varepsilon)\|^2 + 2\langle \lambda\varepsilon v + o(\varepsilon), -\varepsilon v\rangle \leq 0$$

Dividing ε^2 from both sides,

$$\|\lambda v + o(1)\|^2 \leq 2\langle \lambda v + o(1)\ , v\rangle$$

Hence, $\lambda \geq 0$.

On the other hand, since projections are Lipschitz functions with Lipschitz constants equal to one[Clarke, 2013, Exercise 7.5], all eigenvalues of $J(z)$ need to be less than or equal to one. Otherwise, we would have

$$\lim_{\varepsilon \to 0} \frac{\pi(z+\varepsilon v) - \pi(z)}{\varepsilon} = J(z)v = \lambda v, \quad \lambda > 1$$

for some λ and v, Then we have $\|\pi(z+\varepsilon v) - \pi(z)\| > \|\varepsilon v\|$ for ε sufficiently small, violating the Lipschitz property of projections.

□

Lemma 6 requires $J(z)$ to be symmetric. This assumption holds for many types of convex sets. For example, projections onto points, boxes, balls, second-order cones, and half-spaces have symmetric Jacobians. These sets cover many applications in optimal control.

Next, we explore the smoothness of projections onto $\mathbb{D}_{ij}$.

Based on Corollary 4.13 in [Lewis, 2002] and Theorem 3.3 in [Hare and Lewis, 2004], we have the following theorem.

Theorem 7 *Consider a point z_0 in a convex set*

$$\mathbb{E} = \{z \in \mathbb{R}^{n_{\mathbb{E}}} : \Phi_i(z) \leq 0, \textit{for all } i \in [1, m_{\mathbb{E}}]_{\mathbb{N}}\},$$

where $\Phi_i : \mathbb{R}^{n_{\mathbb{E}}} \to \mathbb{R}$ are smooth convex functions. Denote $I_{\mathbb{E}}(z_0) = \{k : \Phi_k(z_0) = 0\}$. Assume that $\{\nabla\Phi_k(z_0) : k \in I_{\mathbb{E}}(z_0)\}$ is a linearly independent set or an empty set. For any normal vector $\bar{n}$ in $riN_{\mathbb{E}}(z_0)$, there exists a sufficiently small open neighborhood K of $z_0 + \bar{n}$ such that the projection mapping $\pi_{\mathbb{E}}$ from K to set $\mathbb{E}$ is a smooth function.

Furthermore, we set

$$\mathbb{M} = \begin{cases} \{z : \Phi_k(z) = 0,\ \forall k \in I_{\mathbb{E}}(z_0)\}, & \textit{if } I_{\mathbb{E}}(z_0) \neq \emptyset, \\ \mathbb{R}^{n_{\mathbb{E}}}, & \textit{otherwise.} \end{cases}$$

Let $\pi_{\mathbb{M}}$ be the projection function onto $\mathbb{M}$. It follows that

$$\pi_{\mathbb{M}} = \pi_{\mathbb{E}},$$

for all points in K.

Based on Theorem 7, we have:

Theorem 8 *Consider the set $\mathbb{E}$, point z_0, vector $\bar{n}$, and neighborhood K in Theorem 7. Under assumptions in Theorem 7, the projection $\pi_{\mathbb{E}}(y)$ is a smooth function for $y \in K$. The null space of the Jacobian of $\pi_{\mathbb{E}}(z_0 + \bar{n})$ is a subset of* span $N_{\mathbb{E}}(z_0)$, *the set of all linear combinations of the vectors in the set $N_{\mathbb{E}}(z_0)$.*

Proof: Set $y_0 = z_0 + \bar{n}$. If $I_{\mathbb{E}}(z_0) := \{k \mid \Phi_k(z_0) = 0\}$ is empty, then $z_0 \in \operatorname{int} \mathbb{E}$ and thus $\bar{n} = 0$. Consequently, the Jacobian of the projection is the identity matrix, and the conclusion holds automatically.

When $I_{\mathbb{E}}(z_0)$ is nonempty, without loss of generality, we assume that $I_{\mathbb{E}}(z_0) = \{1, \ldots, m\}$, i.e., only the first m inequalities defining $\mathbb{E}$ are satisfied as equalities at z_0.

For any $y \in K$, Theorem 7 ensures that $\pi_{\mathbb{E}}(y) = \pi_{\mathbb{M}}(y)$. Since $\pi_{\mathbb{M}}(y) = \operatorname{argmin}\{\frac{1}{2}\|\bar{y} - y\|^2 \mid \bar{y} \in \mathbb{M}\}$, the linear independence condition in Theorem 7 ensures the following KKT condition: there exists a dual variable $\lambda(y) = (\lambda_1, \cdots, \lambda_m) \in \mathbb{R}^m$ such that

$$0 = (\pi_{\mathbb{E}}(y) - y) + \sum_{i=1}^{m} \lambda_i(y) \nabla \Phi_i(\pi_{\mathbb{E}}(y)). \tag{9}$$

Denote the matrix $F(y)$ as

$$[\nabla\Phi_1(\pi_{\mathbb{E}}(y)), \quad \nabla\Phi_2(\pi_{\mathbb{E}}(y)), \quad \ldots, \quad \nabla\Phi_m(\pi_{\mathbb{E}}(y))].$$

According to the assumptions in Theorem 7, the columns of $F(y_0)$ are linearly independent. Thus, $F(y_0)$ has full column rank, and the number of rows in $F(y_0)$ is not less than the number of columns. Therefore, we can choose m rows of $F(y)$, denoted as $k_1, k_2, \ldots, k_m$, to form a matrix $\tilde{F}(y)$ such that $\tilde{F}(y_0)$ is invertible.

As $\pi_{\mathbb{E}}$ is a smooth function, there exists an open neighborhood $\tilde{K} \subseteq K$ of y_0 such that for any $y \in \tilde{K}$, $\tilde{F}(y)$ is invertible. Therefore, equation (9) implies that

$$\lambda(y) = -\tilde{F}^{-1}(y)[\pi_{\mathbb{E}}(y)_{k_1} - y_{k_1}, \cdots, \pi_{\mathbb{E}}(y)_{k_m} - y_{k_m}]^\top.$$

Using Cramer's rule, $\tilde{F}^{-1}(y)$, and thus $\lambda(y)$, are smooth functions for $y \in \tilde{K}$. We then compute the Jacobian of equation (9) evaluated at the point y_0, leading to

$$J_{\mathbb{E}} - I + \sum_{i=1}^{m} \lambda_i \nabla^2 \Phi_i J_{\mathbb{E}} + \sum_{i=1}^{m} \nabla\Phi_i \nabla\lambda_i = 0, \tag{10}$$

where $J_{\mathbb{E}}(y)$ is the Jacobian of $\pi_{\mathbb{E}}(y)$. In equation (10), $J_{\mathbb{E}}$, λ_i and $\nabla\lambda_i$ are evaluated at y_0, and $\nabla\Phi_i$ and $\nabla^2\Phi_i$ are evaluated at z_0.

Let v be a vector in the null space of $J_{\mathbb{E}}(y_0)$, i.e. $J_{\mathbb{E}}(y_0)v = 0$. Multiplying both sides of (10) by v, we obtain

$$\sum_{i=1}^{m} \nabla\Phi_i(y_0)\nabla\lambda_i^\top(y_0)v = v.$$

Hence,

$$v \in \operatorname{span}(\{\nabla\Phi_k(z_0) \mid k \in I_{\mathbb{E}}(z_0)\}) = \operatorname{span}(N_{\mathbb{E}}(z_0)).$$

The last equality follows from the fact that $N_{\mathbb{E}}(z_0) = \{\sum_k \eta_k \nabla\Phi_k(z_0) \mid \eta_k \geq 0,\ k \in I_{\mathbb{E}}(z_0)\}$. $\square$

Although Theorem 7 covers most sets of interest, it doesn't include projections onto second-order cones and affine subspaces, as both violate assumptions of Theorem 7. To incorporate these sets into our algorithm, we must demonstrate that the smoothness guaranteed by Theorem 7 applies to projections onto second-order cones and affine subspaces. Since projections onto affine subspaces are linear and can be computed explicitly using projection matrices, these projections are smooth, and thus all conclusions from Theorem 8 hold. For projections onto second-order cones, smoothness is only an issue if the projection results in the origin, leading to the following lemma.

Lemma 9 *Consider a second-order cone*

$$\mathbb{E} = \{(x, y) \in \mathbb{R}^{n_x} \times \mathbb{R} \mid \|x\|^2 - ty \leq 0, y \geq 0\},$$

where $t \in \mathbb{R}_+$ is fixed. Then

$$N_{\mathbb{E}}(0) = \{(x, y) \in \mathbb{R}^{n_x} \times \mathbb{R} \mid t\|x\|^2 \leq -y, y \leq 0\},$$

and $\operatorname{span} N_{\mathbb{E}}(0) = \mathbb{R}^{n_x+1}$. *For any $\bar{n} \in \operatorname{ri} N_{\mathbb{E}}(0)$, there exists an open neighborhood K around $\bar{n}$ such that $\pi_{\mathbb{E}}(v) = 0$ for all $v \in K$, ensuring the smoothness of $\pi_{\mathbb{E}}$ in K.*

Proof: The form of the normal cones can be verified using the definition of normal cones to convex sets. Clearly, $\operatorname{span}(N_{\mathbb{E}}(0))$ encompasses the entire space and $\operatorname{ri} N_{\mathbb{E}}(0) = \operatorname{int}(N_{\mathbb{E}}(0))$. Moreover, if $\bar{n} \in \operatorname{int}(N_{\mathbb{E}}(0))$, there exists a neighborhood K such that for any v in K, $v \in N_{\mathbb{E}}(0)$ and $\pi_{\mathbb{E}}(v) = 0$, according to the properties of the normal cone. $\square$

3.3 Nonsingularity of the Newton Step Matrix

In this section, we will show that the Newton step is well defined in a local neighborhood of $(z^\star, w^\star)$, i.e., T is differentiable and $I - J_T(z^k, w^k)$ is invertible.

We start with the following lemma, which is also a major component of the algorithm. This lemma ensures that the linear system (6), which involves a nonsymmetric matrix, can be transformed into a symmetric linear system. We will later show that this symmetric linear system exhibits a desirable sparsity pattern.

Lemma 10 *Assume that $J_{\mathbb{D}}$ in equation 5 is symmetric and has an eigenvalue decomposition $J_{\mathbb{D}} = Q_{\mathbb{D}}\Lambda_{\mathbb{D}}Q_{\mathbb{D}}^\top$. Then, the matrix $I - \Lambda_{\mathbb{D}} + \alpha\Lambda_{\mathbb{D}}P$ is invertible. Denote*

$$\begin{aligned} V_{\mathbb{D}} &:= Q_{\mathbb{D}}\left(I - \Lambda_{\mathbb{D}} + \alpha\Lambda_{\mathbb{D}}P\right)^{-1} Q_{\mathbb{D}}^\top \\ W_{\mathbb{D}} &:= HQ_{\mathbb{D}}\left(I - \Lambda_{\mathbb{D}} + \alpha\Lambda_{\mathbb{D}}P\right)^{-1} \Lambda_{\mathbb{D}}Q_{\mathbb{D}}^\top H^\top \\ \tilde{W}_{\mathbb{D}} &:= \alpha\beta J_{\mathbb{K}^\circ} W_{\mathbb{D}} J_{\mathbb{K}^\circ} + I - J_{\mathbb{K}^\circ}. \end{aligned}$$

Solving equation (6) is thus equivalent to solving

$$\begin{aligned} \tilde{W}_{\mathbb{D}}\Delta w &= (I - \alpha\beta^2 J_{\mathbb{K}^\circ} W_{\mathbb{D}}(I - J_{\mathbb{K}^\circ}))\bar{R}_w^k, \\ \Delta z &= V_{\mathbb{D}}(\tilde{R}_z^k - \alpha J_{\mathbb{D}} H^\top \Delta w). \end{aligned}$$

Here,

$$\begin{pmatrix} \tilde{R}_z^k \\ \tilde{R}_w^k \end{pmatrix} := \begin{bmatrix} I & 0 \\ -2J_{\mathbb{K}^\circ}\beta H & I \end{bmatrix} (T(z^k, w^k) - (z^k, w^k)),$$

and

$$\bar{R}_w^k = J_{\mathbb{K}^\circ}\beta H V_{\mathbb{D}} \tilde{R}_z^k + \tilde{R}_w^k.$$

Proof: The invertibility of $I - \Lambda_{\mathbb{D}} + \alpha\Lambda_{\mathbb{D}}P$ is a direct consequence of Lemma 6 and the fact that P is a diagonal matrix with positive diagonal elements.

We start by simplifying (6) to show the second result. Using the definition of J_T in (5), one can verify that

$$I - J_T = \begin{bmatrix} I & 0 \\ 2J_{\mathbb{K}^\circ}\beta H & I \end{bmatrix} \begin{bmatrix} I - J_{\mathbb{D}}(I - \alpha P) & \alpha J_{\mathbb{D}} H^\top \\ -J_{\mathbb{K}^\circ}\beta H & I - J_{\mathbb{K}^\circ} \end{bmatrix}.$$

Multiplying

$$\begin{bmatrix} I & 0 \\ 2J_{\mathbb{K}^\circ}\beta H & I \end{bmatrix}^{-1} = \begin{bmatrix} I & 0 \\ -2J_{\mathbb{K}^\circ}\beta H & I \end{bmatrix}$$

on both sides of equation (6), we obtain the following equivalent Newton step:

$$\begin{aligned} &\begin{bmatrix} I & 0 \\ -2J_{\mathbb{K}^\circ}\beta H & I \end{bmatrix} (I - J_T(z^k, w^k)) \begin{pmatrix} \Delta z \\ \Delta w \end{pmatrix} \\ &= \begin{bmatrix} I - J_{\mathbb{D}}(I - \alpha P) & \alpha J_{\mathbb{D}} H^\top \\ -J_{\mathbb{K}^\circ}\beta H & I - J_{\mathbb{K}^\circ} \end{bmatrix} \begin{pmatrix} \Delta z \\ \Delta w \end{pmatrix} \\ &= \begin{bmatrix} I & 0 \\ -2J_{\mathbb{K}^\circ}\beta H & I \end{bmatrix} \left(T(z^k, w^k) - (z^k, w^k)\right) \\ &=: \begin{pmatrix} \tilde{R}_z^k \\ \tilde{R}_w^k \end{pmatrix}, \end{aligned}$$

where $\tilde{R}_z^k \in \mathbb{R}^{n_z}$ and $\tilde{R}_w^k \in \mathbb{R}^{n_w}$.

Hence, solving Equation 6 is equivalent to solving the following linear system:

$$\begin{bmatrix} I - J_{\mathbb{D}}(I - \alpha P) & \alpha J_{\mathbb{D}} H^\top \\ -J_{\mathbb{K}^\circ}\beta H & I - J_{\mathbb{K}^\circ} \end{bmatrix} \begin{pmatrix} \Delta z \\ \Delta w \end{pmatrix} = \begin{pmatrix} \tilde{R}_z^k \\ \tilde{R}_w^k \end{pmatrix}. \quad (11)$$

We solve this linear system by eliminating Δz using the first row and obtaining an equation involving only Δw. To do so, we need the inverse of $I - J_{\mathbb{D}}(I - \alpha P)$. Noticing that the specific structure in Problem 1 ensures that $Q_{\mathbb{D}}P = PQ_{\mathbb{D}}$, we have

$$\begin{aligned} (I - J_{\mathbb{D}}(I - \alpha P))^{-1} &= \left(I - Q_{\mathbb{D}}\Lambda_{\mathbb{D}}Q_{\mathbb{D}}^\top (I - \alpha P)\right)^{-1} \\ &= \left(Q_{\mathbb{D}}\left(I - \Lambda_{\mathbb{D}} + \alpha\Lambda_{\mathbb{D}}P\right)Q_{\mathbb{D}}^\top\right)^{-1} \\ &= Q_{\mathbb{D}}\left(I - \Lambda_{\mathbb{D}} + \alpha\Lambda_{\mathbb{D}}P\right)^{-1} Q_{\mathbb{D}}^\top \\ &=: V_{\mathbb{D}}. \end{aligned}$$

Meanwhile

$$V_{\mathbb{D}} J_{\mathbb{D}} = Q_{\mathbb{D}}\left(I - \Lambda_{\mathbb{D}} + \alpha\Lambda_{\mathbb{D}}P\right)^{-1} \Lambda_{\mathbb{D}} Q_{\mathbb{D}}^\top. \quad (12)$$

Solving the first row of equation (11), we have

$$\Delta z = V_{\mathbb{D}}\left(\tilde{R}_z^k - \alpha J_{\mathbb{D}} H^\top \Delta w\right).$$

Substituting this into the second row of equation (11), using equation (12), and applying the definitions of $W_{\mathbb{D}}$ and $\bar{R}_w^k$, we obtain

$$J_{\mathbb{K}^\circ}\alpha\beta W_{\mathbb{D}}\Delta w + (I - J_{\mathbb{K}^\circ})\Delta w = \beta\bar{R}_w^k.$$

Because of the specific form of $\mathbb{K}^\circ$, $J_{\mathbb{K}^\circ}$ is a diagonal matrix with only zeros and ones on the diagonal. Therefore, $(I - J_{\mathbb{K}^\circ})J_{\mathbb{K}^\circ} = 0$, and we have

$$\begin{aligned} (I - J_{\mathbb{K}^\circ})\Delta w &= (I - J_{\mathbb{K}^\circ})\beta\bar{R}_w^k, \\ \alpha\beta J_{\mathbb{K}^\circ} W_{\mathbb{D}} J_{\mathbb{K}^\circ}\Delta w &= J_{\mathbb{K}^\circ}\beta\bar{R}_w^k - \alpha\beta J_{\mathbb{K}^\circ} W_{\mathbb{D}}(I - J_{\mathbb{K}^\circ})\Delta w. \end{aligned}$$

Summing both equations, we have an equivalent linear equation

$$\begin{aligned} &(\alpha\beta J_{\mathbb{K}^\circ} W_{\mathbb{D}} J_{\mathbb{K}^\circ} + I - J_{\mathbb{K}^\circ})\Delta w \\ &= \beta\bar{R}_w^k - \alpha\beta J_{\mathbb{K}^\circ} W_{\mathbb{D}}(I - J_{\mathbb{K}^\circ})\Delta w \\ &= (\beta I - \alpha\beta^2 J_{\mathbb{K}^\circ} W_{\mathbb{D}}(I - J_{\mathbb{K}^\circ}))\bar{R}_w^k. \end{aligned}$$

□

Lemma 10 indicates that the nonsingularity of $I - J_T$ is equivalent to the nonsingularity of $W_{\mathbb{D}}$. To demonstrate the nonsingularity of the latter, we need to make a few assumptions.

Assumption 11 *The solution $z^\star$ of Problem 2 satisfies the extended LICQ condition. Moreover, in Problem 1, each set $\mathbb{D}_{ij}$ is either a second-order cone, an affine subspace, or satisfies the linear independence condition in Theorem 7 at all points.*

Assumption 12 *The Jacobian of projections onto set $\mathbb{D}$, when it exists, is a symmetric matrix.*

Since $\mathbb{D}$ is the direct product of sets $\mathbb{D}_{ij}$, Assumption 12 is satisfied if and only if the Jacobians of projections onto $\mathbb{D}_{ij}$ are symmetric. Many sets of interest in optimal control have this property, including balls, second-order cones, boxes, points, half-spaces, hyperplanes, affine spaces, and polytopes. This ensures the applicability of our algorithm to a wide range of problems.

To ensure the differentiability of the PIPG operator, we need the following assumption:

Assumption 13 (Strict Complementarity) *For the fixed point $(z^\star, w^\star)$ of the PIPG operator, we require*

- $-\left(Pz^\star + q + H^\top w^\star\right) \in \operatorname{ri} N_{\mathbb{D}}(z^\star)$
- $H(z^\star) - g \in \operatorname{ri} N_{\mathbb{K}^\circ}(w^\star)$

Assumption 13 is closely related to the conventional strict complementarity condition [Hare and Lewis, 2004, Example 4.4], which is usually required for a fast convergence rate in Newton-type methods [Nocedal and Wright, 1999, Assumption 19.1].

Additionally, Assumption 13 is generally satisfied in practice. When $(z^\star, w^\star)$ are fixed and the vectors q and g are sampled from a Gaussian distribution, Assumption 13 is violated only if q lies on the relative boundary of the set $N_{\mathbb{D}}(z^\star) + Pz^\star + H^\top w^\star$, or if $-g$ lies on the relative boundary of $N_{\mathbb{K}^\circ}(w^\star) - Hz^\star$. The conditional probability of violating Assumption 13 for fixed $(z^\star, w^\star)$ is zero since the relative boundary has measure zero under the distributions of q and g. Furthermore, integrating over all $(z^\star, w^\star)$ outcomes, the probability of violation remains zero. Of course, q and g may be generated by other types of mechanisms, and it is possible that Assumption 13 may be violated. In that case, our algorithm will still ensure convergence, though the theoretical convergence rate may not be quadratic.

We are now ready to introduce the theorem for the nonsingularity of $I - J_T(z, w)$.

Theorem 14 *Under Assumptions 11, 12, and 13, $I - J_T(z, w)$ is a smooth function of (z, w) in a local neighborhood of $(z^\star, w^\star)$, the fixed point of the PIPG operator. Moreover, $I - J_T(z, w)$ is differentiable for (z, w) in that neighborhood.*

Proof: We start with the differentiability of $I - J_T(z, w)$. Utilizing the following fact [Rockafellar, 1970, Section 6],

$$\operatorname{ri} N_{\mathbb{D}}(z) = \prod_{i=1}^{N} \prod_{j=1}^{m_i} \operatorname{ri} N_{\mathbb{D}_{ij}}(z_{ij}),$$

and according to Theorem 7, Lemma 9, Assumption 11 and Assumption 13, it is ensured that $\pi_{\mathbb{D}}$ is a smooth function in a neighborhood of $z^\star - \alpha(Pz^\star + q + H^\top w^\star)$. Similarly, $\pi_{\mathbb{K}^\circ}$ is a smooth function in a neighborhood of $w^\star + \beta(H(z^\star) - g)$. Hence, $I - J_T(z, w)$ is also a smooth function in a neighborhood of $(z^\star, w^\star)$.

Next, we move to the invertibility. According Lemma 10, $I - J_T(z^\star, w^\star)$ is invertible if and only if $\tilde{W}_{\mathbb{D}} =$

$$\alpha\beta J_{\mathbb{K}^\circ} H \left(Q_{\mathbb{D}}(I - \Lambda_{\mathbb{D}} + \alpha\Lambda_{\mathbb{D}} P)^{-1} \Lambda_{\mathbb{D}} Q_{\mathbb{D}}^\top\right) H^\top J_{\mathbb{K}^\circ} + I - J_{\mathbb{K}^\circ}$$

is invertible. Let v be a vector such that $\tilde{W}_{\mathbb{D}} v = 0$. Since $J_{\mathbb{K}^\circ}$ is a diagonal matrix with zeros and ones on the diagonal, and

$$H\left(Q_{\mathbb{D}}(I - \Lambda_{\mathbb{D}} + \alpha\Lambda_{\mathbb{D}} P)^{-1} \Lambda_{\mathbb{D}} Q_{\mathbb{D}}^\top\right) H^\top$$

is positive semi-definite, we have

$$\begin{aligned} (I - J_{\mathbb{K}^\circ}) v &= 0, \\ \Lambda_{\mathbb{D}} Q_{\mathbb{D}}^\top H^\top J_{\mathbb{K}^\circ} v &= 0. \end{aligned}$$

Observe that the null space of $\Lambda_{\mathbb{D}} Q_{\mathbb{D}}$ coincides with the null space of $J_{\mathbb{D}}$. Additionally, $J_{\mathbb{D}}$ is a block diagonal matrix. The null space for each diagonal block is a subset of $\operatorname{span}(N_{\mathbb{D}_{ij}}(z_{ij}^\star))$, as ensured by Theorem 8 and Lemma 9. Hence, the null space of $J_{\mathbb{D}}$ is a subset of

$$\operatorname{span}(N_{\mathbb{D}}(z)) = \prod_{i=1}^{N} \prod_{j=1}^{m_i} \operatorname{span}(N_{\mathbb{D}_{ij}}(z_{ij})),$$

and $H^\top J_{\mathbb{K}^\circ} v \in \operatorname{span}(N_{\mathbb{D}}(z))$. Meanwhile, $H^\top J_{\mathbb{K}^\circ}$ shares the same column space with $[H_E^\top, H_{A_I}^\top]$, where H_E and H_{A_I} are defined in the extended LICQ assumption. Hence, the extended LICQ ensures that $J_{\mathbb{K}^\circ} v = 0$. Since $(I - J_{\mathbb{K}^\circ}) v = 0$, it follows that $v = 0$ and thus $I - J_T$ is invertible at $(z^\star, w^\star)$. As $I - J_T$ is a smooth function of (z, w) in a neighborhood of $(z^\star, w^\star)$, it remains invertible in some neighborhood of $(z^\star, w^\star)$. $\square$

Remark 15 *$I - J_T$ may not be invertible if the current iteration is not close to the primal-dual solution pair. For methods to handle the singular $I - J_T$, see Section 6. Importantly, this non-invertibility does not compromise the global convergence or the local convergence rate.*

3.4 Efficient Computation of Linear System in Newton Step

To efficiently factorize the invertible matrix

$$\tilde{W}_{\mathbb{D}} := \alpha\beta J_{\mathbb{K}^\circ} W_{\mathbb{D}} J_{\mathbb{K}^\circ} + I - J_{\mathbb{K}^\circ},$$

we use an approach from [Wang and Boyd, 2009]. This approach leverages the sparsity inherent in the optimal control problem, reducing computational costs to a linear function of the total number of time grid points N.

By definition,

$$W_{\mathbb{D}} = HQ_{\mathbb{D}}(I - \Lambda_{\mathbb{D}} + \Lambda_{\mathbb{D}}\alpha P)^{-1} \Lambda_{\mathbb{D}} Q_{\mathbb{D}}^\top H^\top.$$

Since $Q_{\mathbb{D}}(I - \Lambda_{\mathbb{D}} + \Lambda_{\mathbb{D}}\alpha P)^{-1} \Lambda_{\mathbb{D}} Q_{\mathbb{D}}$ is a block diagonal matrix, we can write it as:

$$\operatorname{blkdiag}(U_0, U_1, U_2, \ldots, U_N, U_{N+1}),$$

where $U_i \in \mathbb{R}^{n_{z_i} \times n_{z_i}}$. Hence, $W_{\mathbb{D}}$ will have the following pattern:

$$W_D = \begin{bmatrix} W_{00} & W_{01} & 0 & \cdots & 0 & 0 \\ W_{10} & W_{11} & W_{12} & \cdots & 0 & 0 \\ 0 & W_{21} & W_{22} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & W_{N-1,N-1} & W_{N-1,N} \\ 0 & 0 & 0 & \cdots & W_{N,N-1} & W_{N,N} \end{bmatrix},$$

where $W_{i,i} \in \mathbb{R}^{n_i \times n_i}$, $W_{i,i+1} \in \mathbb{R}^{n_i \times n_{i+1}}$, $W_{i+1,i} \in \mathbb{R}^{n_{i+1} \times n_i}$ given that $n_i = n_{E_i} + n_{I_i}$. Using the definition of H in equation (2), we have

$$\begin{aligned} W_{i,i} &= A_i U_i A_i^\top + B_i U_{i+1} B_i^\top, \ i \in [0, N]_{\mathbb{N}}, \\ W_{i,i+1} &= W_{i+1,i}^\top = B_i U_{i+1} A_i^\top, \quad i \in [0, N-1]_{\mathbb{N}}, \end{aligned}$$

Since $J_{\mathbb{K}^\circ}$ is a diagonal matrix with only zeros and ones on its diagonal, $\tilde{W}_{\mathbb{D}}$ will follow exactly the same pattern as $W_{\mathbb{D}}$. Denote $m_k = \sum_{i=0}^{k-1} n_i$ for $k = [1, N]_{\mathbb{N}}$, $m_0 = 0$, and $J_{\mathbb{K}^\circ i} \in \mathbb{R}^{n_i \times n_i}$ being a diagonal matrix whose diagonal elements equal the m_i to m_{i+1} elements of $J_{\mathbb{K}^\circ}$. Then we have

$$\tilde{W}_{\mathbb{D}} = \begin{bmatrix} \tilde{W}_{00} & \tilde{W}_{01} & 0 & \cdots & 0 & 0 \\ \tilde{W}_{10} & \tilde{W}_{11} & \tilde{W}_{12} & \cdots & 0 & 0 \\ 0 & \tilde{W}_{21} & \tilde{W}_{22} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \tilde{W}_{N-1,N-1} & \tilde{W}_{N-1,N} \\ 0 & 0 & 0 & \cdots & \tilde{W}_{N,N-1} & \tilde{W}_{N,N} \end{bmatrix},$$

where

$$\begin{aligned} \tilde{W}_{i,i} &= \alpha\beta J_{\mathbb{K}^\circ i} W_{i,i} J_{\mathbb{K}^\circ i} + I - J_{\mathbb{K}^\circ i}, i \in [0, N]_{\mathbb{N}} \\ \tilde{W}_{i,i+1} &= \tilde{W}_{i,i+1}^\top = \alpha\beta J_{\mathbb{K}^\circ i} W_{i,i+1} J_{\mathbb{K}^\circ i+1}, i \in [0, N-1]_{\mathbb{N}}. \end{aligned}$$

Next, we compute the Cholesky decomposition $\tilde{W}_{\mathbb{D}} = LL^\top$ using the factorization technique from [Wang and Boyd, 2009], where L is lower triangular. The Cholesky factor L has a lower bidiagonal block structure.

$$L = \begin{bmatrix} L_{00} & 0 & 0 & \cdots & 0 & 0 \\ L_{01} & L_{11} & 0 & \cdots & 0 & 0 \\ 0 & L_{21} & L_{22} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & L_{N-1,N-1} & 0 \\ 0 & 0 & 0 & \cdots & L_{N,N-1} & L_{N,N} \end{bmatrix},$$

where $L_{i,i} \in \mathbb{R}^{n_i \times n_i}$, $L_{i+1,i} \in \mathbb{R}^{n_{i+1} \times n_i}$. Specifically, L_{ii} are lower triangular matrices. Since $LL^\top = \tilde{W}_{\mathbb{D}}$, we have

$$\begin{aligned} L_{00} L_{00}^\top &= \tilde{W}_{00} \\ L_{i,i} L_{i+1,i}^\top &= \tilde{W}_{i,i+1} \quad i \in [0, N-1]_{\mathbb{N}}, \\ L_{i,i} L_{i,i}^\top &= \tilde{W}_{ii} - L_{i,i-1} L_{i,i-1}^\top, \quad i \in [1, N]_{\mathbb{N}}. \end{aligned} \tag{13}$$

To compute L_{00}, we employ Cholesky decomposition on $\tilde{W}_{00}$. Subsequently, L_{10} is determined through forward substitution from the equation $L_{00} L_{10}^\top = \tilde{W}_{01}$. Next, L_{11} is obtained by performing Cholesky decomposition on the matrix $\tilde{W}_{11} - L_{10} L_{10}^\top$. This procedure is repeated iteratively to construct the entire L matrix.

We summarize the procedure for computing the Newton step in Algorithm 1. It combines the insights from Lemma 10 with the Cholesky factorization. Details of implementations are left to Section 6.

Algorithm 1 Newton step for PIPG

```
1: Require $H, \alpha, \beta, J_{\mathbb{D}}, J_{\mathbb{K}^\circ}, P, z^k, w^k$
2: Compute $R^k = T(z^k, w^k) - (z^k, w^k)$
3: Compute $\begin{pmatrix} \tilde{R}_z^k \\ \tilde{R}_w^k \end{pmatrix} = \begin{bmatrix} I & \\ -2\beta H & I \end{bmatrix} R^k$
4: Compute decomposition $J_{\mathbb{D}} = Q_{\mathbb{D}} \Lambda_{\mathbb{D}} Q_{\mathbb{D}}^\top$
5: Compute $\bar{R}_w^k = J_{\mathbb{K}^\circ} \beta H V_{\mathbb{D}} \tilde{R}_z^k + \tilde{R}_w^k$.
6: Compute matrix $\tilde{W}_{\mathbb{D}}$ using $Q_{\mathbb{D}}$ and $\Lambda_{\mathbb{D}}$
7: Compute $L$ for $\tilde{W}_{\mathbb{D}}$ using equation (13)
8: Compute
       $\Delta w = \left(L^\top\right)^{-1} L^{-1} (I - \alpha\beta^2 J_{\mathbb{K}^\circ} W_{\mathbb{D}} (I - J_{\mathbb{K}^\circ})) \bar{R}_w^k$
9: Compute $\Delta z = V_{\mathbb{D}} \left(\tilde{R}_z^k - \alpha J_{\mathbb{D}} H^\top \Delta w\right)$
```

4 Global Convergence for Newton-PIPG

In this section, we introduce the Newton-PIPG algorithm, a hybrid of the PIPG operator and the Newton step.

We define the residual of the PIPG operator as

$$R(z, w) := T(z, w) - (z, w),$$

and we denote Fix(T) the fixed point set of T. Rather than employing the conventional Euclidean norm as the convergence criterion, we adopt an alternative norm: $\|z\|_M := \sqrt{\langle z, Mz\rangle}$, where M is a positive definite matrix expressed as

$$M = \begin{bmatrix} \frac{1}{\alpha}I - P & -H^\top \\ -H & \frac{1}{\beta}I \end{bmatrix}.$$

The positive definiteness of M is proved later in Lemma 16. The associated inner product with this matrix is given by:

$$\langle x, y\rangle_M = \langle x, My\rangle.$$

and the norm $\|\cdot\|_M$ for a given matrix A is defined as

$$\|A\|_M = \max_{\|x\|_M \leq 1} \|Ax\|_M.$$

The norm $\|\cdot\|_M$ is intrinsic to the proof of PIPG's convergence, rendering it a convenient choice for establishing convergence for the Newton-PIPG algorithm. Due to the equivalence among norms on finite-dimension problems, the convergence in $\|\cdot\|_M$ is equivalent to that in the Euclidean norm.

In order to ensure the convergence of the Newton method, we design Algorithm 2 that combines the PIPG with the Newton step, inspired by [Themelis and Patrinos, 2019]. In this algorithm, the PIPG algorithm will be a safeguard to ensure global convergence, while the Newton step will be used to accelerate PIPG locally. Algorithm 2 provides flexibility in choosing the update

Algorithm 2 Newton-PIPG

```
 1: Require $c \in [0,1), e \geq 0, t > 0$, Initial iteration $(z^0, w^0)$
 2: Initialize $k = 0$
 3: while termination criteria not satisfied do
 4:     Compute an update $p^k = (\Delta z^k, \Delta w^k)$
 5:     if
            $\left\|R(z^k + \Delta z^k, w^k + \Delta w^k)\right\|_M \leq c\left\|R(z^k, w^k)\right\|_M$
        and
            $\|p^k\|_M < \sigma\|R(z^k, w^k)\|_M$
        then
 6:         Newton update :
                $(z^{k+1}, w^{k+1}) = (z^k + \Delta z^k, w^k + \Delta w^k)$
 7:     else
 8:         PIPG update : $(z^{k+1}, w^{k+1}) = \mathrm{T}(z^k, w^k)$
 9:     end
10:     k = k+1
11: end while
```

p^k while ensuring global convergence, as demonstrated in Theorem 17. In practice, p^k can be set to zero or to the solution of the Newton step. When p^k is zero, the Newton update is disabled. To achieve fast computation, we periodically activate the Newton step instead of using it at each iteration. Details on selecting p^k are available in Section 6.

The termination criteria for Algorithm 2 is discussed in section 6.2. The coefficient σ in Algorithm 2 is a safeguard ensuring that the Newton update does not move (z^k, w^k) to a point far away from the current iteration. The use of such a coefficient σ is inspired by [Themelis and Patrinos, 2019].

4.1 Convergence Analysis of Newton-PIPG

Next, we start the convergence analysis of Algorithm 2. First, we show the positive definiteness of M.

Lemma 16 *Assuming $\alpha, \beta > 0$, and $\alpha(\|P\| + \beta\|H\|^2) < 1$, M is a positive definite matrix.*

Proof:

Consider any vector $v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$, where $v_1 \in \mathbb{R}^{n_z}$ and $v_2 \in \mathbb{R}^{n_x(N+1)}$. We have

$$\begin{aligned}\|v\|_M &= \left[v_1^\top, v_2^\top\right] \begin{bmatrix} \frac{1}{\alpha}I - P & -H^\top \\ -H & \frac{1}{\beta}I \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \\ &= \frac{1}{\alpha}\|v_1\|_2^2 + \frac{1}{\beta}\|v_2\|_2^2 - 2v_2^\top H v_1 - v_1^\top P v_1.\end{aligned}$$

Note that

$$2v_2^\top H v_1 \leq 2\|v_1\|_2 \|v_2\|_2 \|H\|_2 \leq \frac{1}{\beta}\|v_2\|_2^2 + \beta\|v_1\|_2^2 \|H\|_2^2.$$

Thus

$$\|v\|_M \geqslant \frac{1}{\alpha}\|v_1\|_2^2 - \|P\|\|v_1\|_2^2 - \beta\|H\|_2^2\|v_1\|_2^2 \geqslant 0,$$

according to the assumption $\alpha, \beta > 0$ and $\alpha(\|P\| + \beta\|H\|^2) < 1$. The equality holds only when $v = 0$.

□

To simplify the notation, we set $\tilde{z}^k = (z^k, w^k)$, $T\tilde{z} = T(z, w)$ and $R\tilde{z} = R(z, w)$. The following theorem, taken from [Yu et al., 2022, Lemma 2], indicates a key property for the PIPG operator.

Theorem 17 *Under the assumption in Lemma 16, for the PIPG operator T, there exists a non-expansive operator $\tilde{T}$, i.e. $\|\tilde{T}x - \tilde{T}y\|_M \leq \|x - y\|_M$, such that $T = \frac{1}{2}I + \frac{1}{2}\tilde{T}$.*

We present the following theorem, which is a standard result concerning the convergence of algorithms based on nonexpansive operators. The proof, originally from [Bauschke et al., 2011, Theorem 5.14], is included here for completeness.

Theorem 18 *Suppose that T satisfies $T = \lambda I + (1-\lambda)\tilde{T}$, where $\tilde{T}$ is nonexpansive, i.e. $\|\tilde{T}x - \tilde{T}y\|_M \leq \|x - y\|_M$. Considering an algorithm that $\tilde{z}^{k+1} = T\tilde{z}^k$. Then, the operator $R = I - T$ is continuous, and $\|R\tilde{z}^k\|_M$ is decreasing. Furthermore, if the fixed point set of T is nonempty, the following hold:*

- $\|\tilde{z}^{k+1} - y\|_M^2 \leq \|\tilde{z}^k - y\|_M^2 - \lambda(1-\lambda)\|R\tilde{z}^k\|_M^2$ *for any* $y \in \mathrm{Fix}(T)$
- $\|R\tilde{z}^k\|_M$ *converges to zero.*
- $\tilde{z}^k$ *will converge to a fixed point of* T.

Proof: We start with some facts related to T and $\tilde{\mathrm{T}}$. Firstly, T is nonexpansive as $\|Tx - Ty\|_M = \|(1-\lambda)(x-y) + \lambda(\tilde{T}x - \tilde{T}y)\|_M \leq (1-\lambda)\|x-y\|_M + \lambda\|\tilde{T}x - \tilde{T}y\|_M \leqslant \|x-y\|_M$. Secondly, $y \in \mathrm{Fix}(T)$ if and only if $y \in \mathrm{Fix}(\tilde{T})$. This is proved by plugging y into $T = \lambda I + (1-\lambda)\tilde{T}$.

For the first conclusion, we know that, by the convexity of norm function, $\|Rx - Ry\|_M = \|\lambda(x-y) + (1-\lambda)(\tilde{T}x - \tilde{T}y)\|_M \leq \lambda\|x-y\|_M + (1-\lambda)\|\tilde{T}x - \tilde{T}y\|_M \leqslant \|x-y\|_M$. Therefore, R is a Lipschitz continuous operator.

Meanwhile, since T is nonexpansive,

$$\begin{aligned}\left\|R\tilde{z}^{k+1}\right\|_M &= \|\ \tilde{z}^{k+1} - T\tilde{z}^{k+1}\ \|_M = \|\ T\tilde{z}^k - T\tilde{z}^{k+1}\|_M \\ &\leq \left\|\tilde{z}^{k+1} - \tilde{z}^k\right\|_M = \left\|T\tilde{z}^k - \tilde{z}^k\right\|_M \\ &= \left\|R\tilde{z}^k\right\|_M.\end{aligned}$$

This proves the second half of the conclusion.

Now we work on the other conclusions. For arbitrary y,

$$\|\tilde{z}^{k+1} - y\|_M^2 = \|\ (1-\lambda)\left(\tilde{z}^k - y\right) + \lambda\left(\tilde{T}\tilde{z}^k - y\right)\|_M^2.$$

We can expand and simplify this equation:

$$\begin{aligned}\|\tilde{z}^{k+1} - y\|_M^2 &= \left\|(1-\lambda)\left(\tilde{z}^k - y\right)\right\|_M^2 \\ &\quad + 2\lambda(1-\lambda)\left\langle \tilde{z}^k - y, \tilde{T}\tilde{z}^k - y\right\rangle_M + \left\|\lambda\left(\tilde{T}\tilde{z}^k - y\right)\right\|_M^2 \\ &= (1-\lambda)\left\|\tilde{z}^k - y\right\|_M^2 + \lambda\left\|\tilde{T}\tilde{z}^k - y\right\|_M^2 \\ &\quad - \lambda(1-\lambda)\left\|\tilde{z}^k - \tilde{T}\tilde{z}^k\right\|_M^2.\end{aligned}$$

Here, we have used the fact that

$$\begin{aligned}\left\langle \tilde{z}^k - y, \tilde{T}\tilde{z}^k - y\right\rangle_M = \frac{1}{2}\Big(&\left\|\tilde{T}\tilde{z}^k - y\right\|_M^2 \\ &+ \left\|\tilde{z}^k - y\right\|_M^2 - \left\|\tilde{z}^k - \tilde{T}\tilde{z}^k\right\|_M^2\Big).\end{aligned}$$

If $\mathrm{Fix}(T)$ is nonempty, we plug in arbitrary y from this fixed point set. Since $y \in \mathrm{Fix}(\tilde{T})$ as well, we have

$$\left\|\tilde{T}\tilde{z}^k - y\right\|_M^2 = \left\|\tilde{T}\tilde{z}^k - \tilde{T}y\right\|_M^2 \leq \left\|\tilde{z}^k - y\right\|_M^2.$$

Thus

$$\begin{aligned}\left\|\tilde{z}^{k+1} - y\right\|_M^2 &\leq \left\|\tilde{z}^k - y\right\|_M^2 - \lambda(1-\lambda)\left\|\tilde{T}\tilde{z}^k - \tilde{z}^k\right\|_M^2 \\ &= \left\|\tilde{z}^k - y\right\|_M^2 - \lambda\left\|R\tilde{z}^k\right\|_M^2.\end{aligned}$$

A summation of both sides of this inequality for all iterations will show that $\|R\tilde{z}^k\|^2$ is summable and thus it will converge to zero. At the meantime $\left\|\tilde{z}^k - y\right\|_M^2$ is decreasing in k and thus $\tilde{z}^k$ is bounded. We can therefore find a subsequence $\tilde{z}^{k_i}$ converging to a point $\tilde{z}^\star$ by compactness. As $\|R\tilde{z}^{k_i}\|_M$ convergence to zero and R is continuous, we have that $T\tilde{z}^\star = \tilde{z}^\star$. Since $\tilde{z}^\star$ is a fixed point, we plug in $\tilde{z}^\star$ in the place of y and get $\left\|\tilde{z}^{k+1} - \tilde{z}^\star\right\|_M^2 \leq \left\|\tilde{z}^k - \tilde{z}^\star\right\|_M^2$ for all k, and $\lim_i \left\|\tilde{z}^{k_i} - \tilde{z}^\star\right\|_M^2 = 0$. Thus $\lim_n \tilde{z}^k = \tilde{z}^\star$ □

We now proceed to establish the convergence of Newton-PIPG and start with the local convergence result.

Theorem 19 *With Assumptions 11, 12, and 13, consider the unique fixed point $\tilde{z}^\star$ of the operator T. There exists a neighborhood K of $\tilde{z}^\star$ such that $\nabla R(\tilde{z}) = I - J_T(\tilde{z})$ for any $\tilde{z} \in K$. Consider vector p for $\tilde{z} \in K$ and*

$$\nabla R(\tilde{z})p = -R(\tilde{z}).$$

There exist constants C_0 and C_1 such that

$$\|R(\tilde{z} + p)\|_M \leq C_0\|R(\tilde{z})\|_M^2 \tag{14}$$

$$\|\tilde{z} + p - \tilde{z}^\star\|_M \leq C_1\|\tilde{z} - \tilde{z}^\star\|_M^2 \tag{15}$$

Proof: According to Theorem 14, there exists a compact neighborhood K such that $\nabla R(\tilde{z}) = I - J_T$ exists and is invertible. Within this neighborhood, $\nabla R(\tilde{z})$ is a smooth function and hence a γ-Lipschitz function for some constant $\gamma > 0$. The invertibility and smoothness of ∇R ensure that there exist constants $\mu > 0$ such that for any vector $\xi \in \mathbb{R}^{n_z \times n_w}$ and any $\tilde{z} \in K$, we have

$$\mu\|\xi\|_M \leq \|\nabla R(\tilde{z})\xi\|_M. \tag{16}$$

Then for any $\tilde{z} \in K$ and $p \in \mathbb{R}^{n_z + n_w}$ such that $\nabla R(\tilde{z})p = -R(\tilde{z})$, we have

$$\begin{aligned}R(\tilde{z} + p) &= R(\tilde{z}) + \int_0^1 \nabla R(\tilde{z} + \theta p)\, p\, d\theta \\ &= \int_0^1 (\nabla R(\tilde{z} + \theta p) - \nabla R(\tilde{z}))\, p\, d\theta\end{aligned}$$

Hence,

$$\begin{aligned}\|R(\tilde{z}+p)\|_M &\leq \int_0^1 \|p\|_M \|\nabla R(\tilde{z}+\theta p) - \nabla R(\tilde{z})\|_M d\theta \\ &\leq \int_0^1 \|p\|_M \gamma \|\theta p\|_M d\theta = \frac{\gamma}{2}\|p\|_M^2 \\ &\leq \frac{\gamma}{2\mu^2}\|R(\tilde{z})\|_M^2,\end{aligned}$$

where the last inequality is a consequence of $\|R(z)\|_M = \|\nabla R(\tilde{z})p\|_M \geqslant \mu\|p\|_M$. Therefore, equality (14) is satisfied with $C_0 = \gamma/2\mu^2$.

Next, we show the validity of equality (15). Setting $\eta = \tilde{z} - \tilde{z}^\star$ and using that $R(\tilde{z}^\star) = 0$, we have

$$\begin{aligned}&R(\tilde{z}) = R(\tilde{z}) - R(\tilde{z}^\star) \\ &= \int_0^1 \nabla R(\tilde{z} - \theta\eta)\,\eta d\theta \\ &= \int_0^1 (\nabla R(\tilde{z} - \theta\eta) - \nabla R(\tilde{z}))\eta d\theta + \nabla R(\tilde{z})\,\eta\end{aligned}$$

Hence

$$\begin{aligned}p &= -(\nabla R(\tilde{z}))^{-1} R(\tilde{z}) \\ &= -\eta - \int_0^1 (\nabla R(\tilde{z}))^{-1} (\nabla R(\tilde{z} - \theta\eta) - \nabla R(\tilde{z}))\,\eta d\theta.\end{aligned}$$

Using $\eta = \tilde{z} - \tilde{z}^\star$, we have

$$\begin{aligned}&\|\tilde{z} + p - \tilde{z}^\star\|_M \\ &\leqslant \int_0^1 \left\|(\nabla R(\tilde{z}))^{-1} (\nabla R(\tilde{z} - \theta\eta) - \nabla R(\tilde{z}))\,\eta\right\|_M d\theta \\ &\leqslant \mu^{-1} \int_0^1 \|\nabla R(\tilde{z} - \theta\eta) - \nabla R(\tilde{z})\|_M \|\eta\|_M d\theta \\ &\leqslant \mu^{-1} \int_0^1 \gamma\theta\|\eta\|_M^2 d\theta = \frac{\gamma}{2\mu}\|\tilde{z} - \tilde{z}^\star\|_M^2\end{aligned}$$

which proves equality (15). Here, the second inequality follows from (16), while the last inequality is a result of the Lipschitz continuity of $\nabla R(\tilde{z})$.

□

Now we prove the global convergence of Newton-PIPG:

Theorem 20 *Suppose the conditions of Lemma 16 hold and* Fix(T) *is non-empty. If the algorithm is run indefinitely, disregarding the termination criteria in line 3 of Algorithm 2, the following properties hold:*

(1) $R\tilde{z}^k \to 0$.
(2) Given Assumptions 11, 12, and 13, if σ is sufficiently large and p_k is computed using equation (6) *for each iteration unless $I - J_T$ is singular, Newton-PIPG converges to the unique fixed point $\tilde{z}^\star$ of T, with a quadratic local convergence rate.*

Proof:

For the first result, consider a proof by contradiction. Assume that $R\tilde{z}^k$ does not converge to zero. Given Theorem 18 and the restriction on the Newton step, $\|R\tilde{z}^k\|_M$ is monotonically decreasing. If $R\tilde{z}^k$ doesn't converge to zero, it implies that the Newton step is disabled, leading us to the same algorithm as described in Theorem 18. The convergence results from Theorem 17 and Theorem 18 then imply a contradiction. Hence, $R\tilde{z}^k \to 0$.

For the second result, let $\tilde{z}^\star$ be the unique fixed point of the PIPG operator. The uniqueness of the fixed point is guaranteed by extended-LICQ and the strong convexity of Problem 1. To start, we aim to show that $\|\tilde{z}^k - \tilde{z}^\star\|_M$ is bounded.

In the k-th iteration, if the PIPG update is used, then $\left\|\tilde{z}^{k+1} - \tilde{z}^\star\right\|_M \leq \left\|\tilde{z}^k - \tilde{z}^\star\right\|_M$ as per Theorem 18. If the Newton update is accepted, the criteria in line 5 of Algorithm 2 ensures that

$$\begin{aligned}\left\|\tilde{z}^{k+1} - \tilde{z}^\star\right\|_M - \left\|\tilde{z}^k - \tilde{z}^\star\right\|_M &\leq \left\|\tilde{z}^{k+1} - \tilde{z}^k\right\|_M \\ &\leq \sigma\left\|R\tilde{z}^k\right\|_M \leq \sigma c^{q_k}\left\|R\tilde{z}^0\right\|_M.\end{aligned}$$

Here, q_k is the number of times the Newton step has been activated since the first iteration. Coefficients c and σ are defined in Algorithm 2.

Summing over k, we obtain

$$\|\tilde{z}^n - \tilde{z}^\star\|_M \leq \left\|\tilde{z}^0 - \tilde{z}^\star\right\|_M + (\sigma\sum_{i=1}^{\infty} c^i)\left\|R\tilde{z}^0\right\|_M.$$

Given that $c < 1$, we infer that $\tilde{z}^k$ is bounded.

According to Theorem 19, there exists a neighborhood K such that equations (14) and (15) hold. Therefore, we can find an M-norm ball $K' \subset K$ such that for any $\tilde{z} \in K'$ and vector p such that $\nabla R(\tilde{z})p_k = -R(\tilde{z})$, we have

$$\begin{aligned}\|R(\tilde{z}+p_k)\|_M &\leq c\|R(\tilde{z})\|_M \\ \|\tilde{z} + p_k - \tilde{z}^\star\|_M &\leq \|\tilde{z} - \tilde{z}^\star\|_M,\end{aligned}$$

where c is defined in Algorithm 2. Hence, for any $\tilde{z}^k \in K'$, we have $\tilde{z}^k + p_k \in K'$.

Therefore, if σ is sufficiently large, specifically $\sigma \geq \sup_{z\in K}\|\nabla R^{-1}\|_M$, we will have $\|p_k\|_M \leq \sigma\|R(\tilde{z}^k)\|_M$ for all $\tilde{z}^k \in K$.

By the compactness of $\tilde{z}^k$, the continuity of R, and that $R\tilde{z}^k \to 0$, there exists a converging subsequence of $\tilde{z}^k$ that converges to the unique fixed point $\tilde{z}^\star$. Then, when $\tilde{z}^k \in K'$, the Newton step will be accepted as it leads to a sufficient amount of decrease in $R(\tilde{z}^k)$, and $\|p_k\|_M \leq \sigma\|R(\tilde{z}^k)\|_M$. Further, $\tilde{z}^{k+1} \in K'$, ensuring that the PIPG will not be activated again. In that case, Equation (15) in Theorem 19 ensures a quadratic convergence rate.

□

Remark 21 *The quadratic convergence result in Theorem 20 can be further strengthened to convergence within finite iterations if $J_{\mathbb{D}}$ and $J_{\mathbb{K}^\circ}$ are piecewise constant functions. This scenario occurs when $J_{\mathbb{D}}$ consists of boxes, halfspaces, and affine subspaces. In these cases, J_T will also be a piecewise constant function of the primal and dual variables. Within a local neighborhood of the fixed point $(z^\star, w^\star)$, the PIPG operator will be a linear function, allowing the Newton step to find the fixed point within one iteration.*

Remark 22 *The proof method in Theorem 20 is inspired by [Themelis and Patrinos, 2019]. The difference between the proof in Theorem 20 and the convergence analysis in [Themelis and Patrinos, 2019] is that we establish convergence for a hybrid approach combining the Newton method, instead of the quasi-Newton method, with the operator splitting method, thus achieving a better convergence rate. Furthermore, the invertibility of $I - J_T$ guarantees the existence of the coefficient σ in our proof (corresponding to D in [Themelis and Patrinos, 2019, Assumption 2]), eliminating the requirement of [Themelis and Patrinos, 2019, Assumption 2], which is not commonly used in optimization research.*

5 Restrictions and Relaxation

In this section, we point out several restrictions in Problem (2) compared to general optimal control quadratic programming problems, and we discuss the possibility of relaxing such restrictions.

In Problem (2), we require P to be a diagonal matrix with positive diagonal elements. This requirement ensures the invertibility of $I - J_{\mathbb{D}} + \alpha J_{\mathbb{D}} P$, which is necessary for the linear algebra technique outlined in Lemma 10. Although this choice of P is suitable for many optimal control problems, it does not encompass all scenarios. In particular, it does not cover QPs arising in the sequential convexification of nonconvex optimal control problems that utilize the L_1-exact penalty [Malyuta et al., 2022, Szmuk et al., 2020, Elango et al., 2024]. When applying the L_1 exact penalty, transforming the linearized subproblems into QP form requires the introduction of slack variables. Consequently, the resulting QP no longer has a positive definite quadratic term; the quadratic matrix contains zeros on the diagonal for positions that correspond to these slack variables. To address this issue, we propose adding an additional squared L_2-norm to the penalty. This modification not only maintains the exactness but also ensures our algorithm's compatibility with the subproblems from the sequential convexification.

Besides the form of P, we also require $J_{\mathbb{D}} = J_{\mathbb{D}}^\top$, $Q_{\mathbb{D}} P = P Q_{\mathbb{D}}$, and $\mathbb{K} = \prod_{i=0}^{N} (0^{n_{E_i}} \times \mathbb{R}_+^{n_{I_i}})$. Generalizing these constraints introduces several challenges. If $J_{\mathbb{D}}$ is not symmetric or $Q_{\mathbb{D}} P \neq P Q_{\mathbb{D}}$, the relationship $(I - J_{\mathbb{D}}(I - \alpha P))^{-1} = V_{\mathbb{D}}$ in Lemma 10 will no longer hold. Furthermore, if we generalize $\mathbb{K}$ to a general cone, $J_{\mathbb{K}^\circ}(I - J_{\mathbb{K}^\circ}) = 0$ will not be satisfied. In either case, the linear matrix appearing in the Newton step will not be symmetric. Consequently, the Cholesky decomposition will not be applicable, and the proof for the nonsingularity of $I - J_T$ will be invalid.

If generalization is necessary, one can directly compute the factorization for $I - J_T$ at some Newton steps and use iterative methods such as GMRES[Saad and Schultz, 1986] for other Newton steps. In this approach, previous factorizations can be used as preconditioners for subsequent iterations.

6 Implementation and Numerical Experiments

We demonstrate the implementation details of the Newton-PIPG algorithm and compare its performance with other state-of-the-art solvers.

6.1 Implementation Details for the Newton-PIPG Algorithm

Algorithm 2 provides a framework for the Newton-PIPG algorithm, allowing flexibility in choosing the form of p^k. This flexibility leads to different strategies for implementing the Newton-PIPG algorithm. Here are some strategies we use to optimize computation speed.

First, we reduce the use of Newton steps by applying them only when PIPG iterations stagnate. A heuristic waiting period, typically 3-10 PIPG iterations, is employed. If both $J_{\mathbb{K}^\circ}$ and the active manifold of $\pi_{\mathbb{D}}$ remain unchanged during this period, the algorithm switches to the Newton step.

Second, after each computationally intensive Newton step, a line search is performed to monitor the reduction in the PIPG residual $R(z^k, w^k)$. This line search, typically executed 3-5 times per Newton step, improves practical convergence speed without affecting the convergence properties of the Newton-PIPG algorithm. The line search result that reduces the residual norm by a factor of c, as defined in Algorithm 2, is accepted. The coefficient c is generally close to 1; for instance, we use $c = 0.99$. If the line search fails to sufficiently reduce the residual, the result is discarded, and a PIPG update is performed instead. Moreover, in the case of a failed line search, the waiting period is extended until either $J_{\mathbb{K}^\circ}$ or the active manifold of $\pi_{\mathbb{D}}$ is updated.

Third, to improve the conditioning of the optimization problem, we rescale the matrix H so that the norm of each row is of the same magnitude. This rescaling does not affect the optimal solution. For a detailed discussion, we refer interested readers to [Kamath et al., 2023].

Fourth, $I - J_T$ can become singular when the current iteration is not in a local neighborhood of the primal-dual solution pair $(z^\star, w^\star)$. To obtain useful results from the Newton step, we add an additional perturbation to the matrix $I - J_T$. The perturbation we choose is an identity matrix multiplied by a coefficient proportional to the norm of the residual $R(z^k, w^k)$, so that the perturbation becomes negligible when the current iteration is close to $(z^\star, w^\star)$.

Fifth, we monitor the convergence of the residual using the Euclidean norm instead of $\| \cdot \|_M$, as the Euclidean norm is more efficient to compute. In our experiments, using the Euclidean norm did not compromise the algorithm's convergence. However, in situations where it might, one may consider using $\| \cdot \|_M$ to validate the Newton step for a more robust implementation.

Finally, we optimize the matrix-vector multiplication by partitioning the sparse matrix H into zero blocks and dense, nonzero blocks. Instead of performing sparse matrix multiplication, we treat H as a collection of dense block matrices. Using for-loops, we compute the matrix-vector product as a series of dense matrix-vector multiplications. This approach is also applied when computing the factorization in the Newton step. A similar method is used in [Yu et al., 2020].

6.2 Termination Criteria

In this section, we propose the termination criteria for both PIPG and Newton-PIPG. We claim that monitoring the magnitude of $\|R(z^k, w^k)\| = \|(z^{k+1}, w^{k+1}) - (z^k, w^k)\|$ provides a good criterion for stopping the PIPG algorithm. Here, (z^k, w^k) and (z^{k+1}, w^{k+1}) are consecutive PIPG iterations. Since Newton-PIPG requires iteratively calling the PIPG operator, we can use the same termination criteria for the Newton-PIPG algorithm as well.

To start, we show that $\|(z^{k+1}, w^{k+1}) - (z^k, w^k)\|$ is a good indicator of the quality of (z^{k+1}, w^{k+1}) in terms of the KKT condition in Theorem 4. Specifically, we set $\Delta z = z^{k+1} - z^k$ and $\Delta w = w^{k+1} - w^k$ and show that:

Theorem 23 *When both $\|\Delta z\| \leq \epsilon$ and $\|\Delta w\| \leq \epsilon$ for $\epsilon > 0$, we have the following relationship:*

$$\begin{aligned}
&d_{N_{\mathbb{D}}(z^{k+1})}\left(-Pz^{k+1} - q - H^\top w^{k+1}\right)\\
&\quad\leq \left(\frac{1}{\alpha} + \|P\| + \|H^\top\|\right)\epsilon\\
&d_{N_{\mathbb{K}^\circ}(w^{k+1})}\left(Hz^{k+1} - g\right) \leq \left(\frac{1}{\beta} + \|H\|\right)\epsilon\\
&z^{k+1} \in \mathbb{D} \text{ and } w^{k+1} \in \mathbb{K}^\circ
\end{aligned}$$

Proof:

The PIPG iterations (3) trivially guarantees $z^{k+1} \in \mathbb{D}$ *and* $w^{k+1} \in \mathbb{K}^\circ$. The PIPG iteration (3) ensures that

$$\begin{aligned}
z^{k+1} =& \pi_{\mathbb{D}}[z^{k+1} - \Delta z - \alpha P(z^{k+1} - \Delta z)\\
&- \alpha\left(q + H^\top\left(w^{k+1} - \Delta w\right)\right)]\\
w^{k+1} =& \pi_{\mathbb{K}^\circ}\left[w^{k+1} - \Delta w + \beta\left(H\left(z^{k+1} + \Delta z\right) - g\right)\right]
\end{aligned}$$

Using properties of projections to a convex set, we obtain

$$\begin{aligned}
&P(\Delta z - z^{k+1}) - q + H^\top(\Delta w - w^{k+1}) - \frac{\Delta z}{\alpha} \in N_{\mathbb{D}}\left(z^{k+1}\right)\\
&Hz^{k+1} - g + H\Delta z - \frac{1}{\beta}\Delta w \in N_{\mathbb{K}^\circ}\left(w^{k+1}\right).
\end{aligned}$$

Using the fact that $\|\Delta z\| \leq \epsilon$ and $\|\Delta w\| \leq \epsilon$

$$\begin{aligned}
&d_{N_{\mathbb{D}}(z^{k+1})}(-Pz^{k+1} - q - H^T w^{k+1})\\
&\leq \|P\Delta z + H^\top \Delta w - \frac{\Delta z}{\alpha}\|\\
&\leq \left(\frac{1}{\alpha} + \|P\| + \|H^\top\|\right)\varepsilon\\
&d_{N_{\mathbb{K}^\circ}(w^{k+1})}\ \left(Hz^{k+1} - g\right)\\
&\leq \|H\Delta z - \frac{\Delta w}{\beta}\| \leq \left(\frac{1}{\beta} + \|H\|\right)\varepsilon.
\end{aligned}$$

□

Setting $\gamma_p = \frac{1}{\alpha} + \|P\| + \|H^\top\|$ and $\gamma_d = \frac{1}{\beta} + \|H\|$, Theorem 23 indicates that $\gamma_p\|\Delta z\|$ and $\gamma_d\|\Delta w\|$ measure the KKT satisfaction for the current iteration (z^k, w^k). Hence, we propose the following termination condition for both PIPG and Newton-PIPG: given an absolute tolerance ϵ_{abs} and a relative accuracy tolerance ϵ_{rel}, the algorithm will be terminated if

$$\begin{aligned}
\left\|z^{k+1} - z^k\right\| &\leq \frac{\epsilon_{\text{abs}} + \epsilon_{\text{rel}}\left\|Pz^{k+1} + q + H^\top w^{k+1}\right\|}{\gamma_p}\\
\left\|w^{k+1} - w^k\right\| &\leq \frac{\epsilon_{\text{abs}} + \epsilon_{\text{rel}}\|Hz^{k+1} - g\|}{\gamma_d},
\end{aligned}$$

Later in the Appendix 8.1, we will use a toy model to demonstrate that the current termination criteria achieve the desired accuracy.

6.3 Numerical Experiment Introduction

We test the performance of the Newton-PIPG algorithm via two example problems. The first example is an oscillating masses problem that involves only linear constraints. The second example is a powered-descent guidance problem that includes various types of constraints, such as ball constraints and second-order cones. We compare the results against benchmark algorithms such as ECOS [Domahidi et al., 2013, Andersen and Andersen, 2000] and OSQP [Stellato et al., 2020], among others.

The numerical experiment is conducted using MATLAB. The Newton-PIPG code was initially written in MATLAB and then converted into executable C code using MATLAB Coder. The computational experiments were carried out on a computer equipped with an AMD Ryzen 7 5700G 8-Core CPU and 16GB of RAM.

6.4 Oscillating Masses

We use a benchmark oscillating masses problem [Wang and Boyd, 2009, Jerez et al., 2014] to evaluate the performance of our algorithm in comparison to other state-of-the-art algorithms. Consider a mechanical system comprising $m = 16$ masses connected in a one-dimensional line, shown in figure 1. Each mass is defined by its 1-D velocity and position, resulting in a total of $n_x = 16$ state variables. To control this mechanical system, we assign one control variable to each mass. Hence the dimension of control at each time grid point is $n_u = 8$. Note that $n_x = 8$ is a typical parameter setting for a 6-degree-of-freedom powered descent guidance problem[Kamath et al., 2023, Szmuk et al., 2020]. Time is discretized using a step size of Δ. The total number of time grids is denoted by N, and we set $\Delta = 3/N$, i.e. this equation describes a system among $[0, 3]$. In our experiment, we set $N = 20, 50, 100$ to show the scalability of our algorithm.

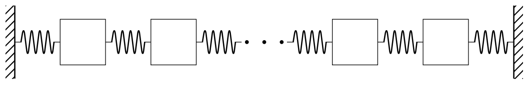

Fig. 1. The oscillating masses system

After discretization, we have the following optimization problem:

$$
\begin{aligned}
\underset{u,x}{\text{minimize}} \quad & \tfrac{1}{2}\textstyle\sum_{i=0}^{N+1} x_i^\top P x_i + \tfrac{1}{2}\sum_{i=0}^{N} u_i^\top Q u_i \\
\text{subject to} \quad & x_{i+1} = Ax_i + Bu_i, \quad i \in [0, N]_{\mathbb{N}}\,, \\
& u_{min} \le u_i \le u_{max}, \quad i \in [0, N]_{\mathbb{N}} \\
& x_{min} \le x_i \le x_{max}, \quad i \in [0, N+1]_{\mathbb{N}} \\
& x_0 = \hat{x}_0.
\end{aligned}
$$

Here, the matrices P and Q are identity matrices and matrices A and B are defined as

$$
A = \exp\left(\Delta \begin{bmatrix} 0_{m\times m} & I_m \\ -\mathbb{L} & 0_{m\times m} \end{bmatrix}\right),
$$
$$
B = \int_0^\Delta \exp\left(s \begin{bmatrix} 0_{m\times m} & I_m \\ -\mathbb{L} & 0_{m\times m} \end{bmatrix}\right) \mathrm{d}s \begin{bmatrix} 0_{m\times m} \\ I_m \end{bmatrix},
$$

where the matrix $\mathbb{L} \in \mathbb{R}^{m\times m}$ is a tridiagonal matrix with 2's on the diagonal and -1's on the subdiagonal and superdiagonal entries.

The initial conditions $\hat{x}_0$ are randomly generated. In our experiments, we set $x_{\min} = -1$ and $x_{\max} = 1$. The initial condition is generated from a normal distribution with mean zero and a standard deviation of 0.3 then projected to $[x_{\min}, x_{\max}]$. This ensures that the values of x are spread within the range of $[-1, 1]$, without too many values equal to -1 or 1.

In our numerical experiments, we compared our algorithm with several prominent solvers: OSQP [Stellato et al., 2020], SCS [O'Donoghue, 2021], SCS-Anderson [Zhang et al., 2020], PIPG [Yu et al., 2022], and ECOS [Domahidi et al., 2013], using their respective MATLAB packages. OSQP and SCS are ADMM-based methods. SCS-Anderson represents methods that combine operator-splitting operators with quasi-Newton methods. Lastly, ECOS stands as an example of an interior-point method.

Since our current algorithm does not have feasibility detection capacity, we only consider problems with feasible solutions. In practice, we run OSQP and SCS before the Newton-PIPG and Newton-PIPG will only be used to solve problems that are characterized as feasible by both OSQP and SCS.

For our numerical experiments, we set the tolerance level of all competing solvers to 10^{-8} in their respective software settings, and for PIPG we set $\epsilon_{abs} = 10^{-8}$, and $\epsilon_{rel} = 0$. The Newton-PIPG algorithm does not require a specific termination criterion, as it naturally converges within a finite number of iterations. Post-computation, we measure and report the difference between each solver's solution and that of Newton-PIPG using the L_2 norm.

We set $(u_{\min}, u_{\max}) \in \{(-1, 1), (-0.4, 0.4)\}$ and then randomly generate 100 initial conditions for each combination of N and $(u_{\min}, u_{\max})$. The average time consumption is shown in Table 1, measured in milliseconds. These time consumption are reported by the Matlab package of each solver. When the maximum control magnitude is set to 1, we observed that all randomly generated problems are feasible. When the control magnitude is set to 0.4, approximately 95% of the randomly generated problems are feasible, and we only apply the Newton-PIPG algorithm to these problems.

When set to $(u_{\min}, u_{\max}) = (-1, 1)$ and $N = 20$, our algorithm outperforms others by a factor of five. Such a ratio persists with larger problem sizes. With respect to the post-computation error tolerance, ECOS attains 10^{-5} in L_2 norm in all cases. At $N = 20$, all other solver achieves 10^8 in accuracy. At $N = 50$ and $N = 100$, algorithms other than PIPG and ECOS achieve accuracy of 10^{-6}. PIPG reaches the target accuracy for all cases.

The Newton-PIPG algorithm achieves rapid QP solution primarily because the PIPG step rapidly identifies the correct active constraints of $\mathbb{D}$ in situations where the constraints are not tight. Thus, it usually requires a single Newton step and a small number of PIPG steps to solve the problem. To evaluate scenarios where more Newton steps are needed, we set $(u_{\min}, u_{\max}) = (-0.4, 0.4)$. With more inequality constraints activated, more Newton steps are expected. As predicted, Newton-PIPG slows down but is still 30% faster than the best of the other algorithms.

We also compare Newton-PIPG with SCS-Anderson, as both methods hybridize operator-splitting and second-order approaches. SCS-Anderson does speed up SCS, but the increase is limited. However, adding the Newton step to PIPG dramatically improves its speed, both at $u = 1$ and $u = 0.4$. This indicates that the Newton step has better performance compared to quasi-Newton methods.

We presented a typical plot (Figure 2) comparing the computation time between the Newton-PIPG method and the PIPG method. The plot tracks the computation time and the norm of the residual after each PIPG and Newton step, with $N = 20$ and $(u_{\min}, u_{\max}) = (-0.4, 0.4)$. The y-axis represents the logarithm of the norm of the difference between two consecutive iterations, and the x-axis represents the solve time in milliseconds. Figure 2 shows that the Newton step is activated twice and is more efficient compared to PIPG iterations.

6.5 Powered-Descent Guidance Problem

Our second numerical example considers a powered-descent guidance (PDG) problem for soft-landing a rocket-powered vehicle on a planetary body. Specifically, we consider the reference trajectory tracking problem from [Elango et al., 2022]. This problem can be formulated to fit within the form of Problem 1 and can be solved using the Newton-PIPG method. We use this problem to demonstrate the efficiency of Newton-PIPG on complex problems with various types of constraints and to compare Newton-PIPG with interior-point solvers.

In the PDG problem, the rocket-powered vehicle is initially located at $r_{\text{init}} \in \mathbb{R}^3$ with an initial velocity $v_{\text{init}} \in \mathbb{R}^3$. The goal is to land this vehicle at the origin with a final velocity of zero. There are seven state variables describing the problem at each time grid point: three variables for location, three variables for velocity, and one variable for the log of the vehicle mass. Meanwhile, there are four control variables at each time grid point: three variables for a three-dimensional thrust and one slack variable to implement the lossless convexification technique. In our numerical problem, we choose the amount of time grid points as 30.

Constraints of the PDG problem include second-order cone constraints on the four-dimensional control variable and the location variables separately, ball constraints on velocity, linear equalities for the dynamics, linear inequalities for the relationship between log-mass and thrust, and box constraints on log-mass. For the exact form and discussion of this problem, refer to [Elango et al., 2022, Problem 3].

Compared to the original form of [Elango et al., 2022, Problem 3], we made several modifications. First, since some dimensions of the optimal solution are a few magnitudes larger than others, we adjusted the quadratic coefficients in the objective function to ensure that each dimension of the optimal solution contributes similarly to the objective function. This adjustment is primarily to enhance the performance of the interior-point method. Without this change, the solution obtained by the interior-point method exhibited large relative errors on some dimensions of the solution, making the comparison between the interior-point method and Newton-PIPG less meaningful.

Secondly, we retained most of the coefficients listed in [Elango et al., 2022, Table 3], modifying only r_{init}, the initial location of the vehicle. We set $r_{\text{init}_{(1)}} = 0\,\text{m}$ and $r_{\text{init}_{(3)}} = 2000\,\text{m}$, while $r_{\text{init}_{(2)}}$ follows an arithmetic sequence from 0 m to 2900 m, with a common difference of 50 m. The optimization problem becomes infeasible when $r_{\text{init}_{(2)}}$ exceeds 2950 m. As the second component of r_{init} increases, the problem approaches infeasibility, allowing us to test our algorithm in both scenarios where feasibility is easily guaranteed and where the problem is nearly infeasible.

In the numerical experiment, we compared the performance of Newton-PIPG with ECOS and PIPG. To use ECOS, we employed Yamlip [Löfberg, 2004] to transform the original problem into a cone programming problem where ECOS is applicable. For the termination conditions, the termination criteria were set to $\epsilon_{\text{abs}} = 10^{-12}$ and $\epsilon_{\text{rel}} = 0$ for Newton-PIPG. Similarly, we set the accuracy to 10^{-12} for ECOS. The post-computation error for ECOS, compared to the Newton-PIPG result, ranged from 3×10^{-6} to 5×10^{-7} for different choices of r_{init}. For PIPG, we used two different termination criteria: $\epsilon_{\text{abs}} = 10^{-4}$ and $\epsilon_{\text{abs}} = 10^{-8}$, with $\epsilon_{\text{rel}} = 0$. The computed errors for PIPG showed that the algorithm achieved the assigned accuracy.

The wall time of the three algorithms under different initial conditions is shown in Figure 4, with a logarithmic time axis. In this figure, the line labeled "PIPG low accuracy" represents the results of the PIPG algorithm with a tolerance of 10^{-4}, while "PIPG high accuracy" corresponds to a tolerance of 10^{-8}. For r_{init} values greater than 2250 meters in the second dimension, PIPG did not converge within 50,000 iterations. As a result, we used hollow markers to represent these non-converging points, which are not explicitly mentioned in the legend. The line labeled "ECOS" shows the wall time reported by Yamlip, excluding Yamlip's compile time.

Comparing the computation speed between Newton-PIPG and ECOS, we observed that Newton-PIPG is notably faster than ECOS when the problem is not close to infeasibility. However, as the problem approaches infeasibility, especially when the second dimension of r_{init} is larger than 2400 meters, the computation time of Newton-PIPG increases dramatically, while the speed of the interior-point methods appears unaffected by the proximity to infeasibility. Therefore, our numerical experiments suggest that Newton-PIPG is preferred when a highly accurate solution is required or when the problem is not close to infeasibility.

Figure 3 compares the residuals of Newton-PIPG and PIPG versus wall time, with $r_{\text{init}} = (0, 0, 2000)$ meters, using the same method as Figure 4.Newton-PIPG requires six Newton steps to converge below the threshold of 10^{-12}, with the last three Newton steps contributing most to the significant decrease in the residual norm, consistent with theoretical expectations.

7 Conclusion and Future Work

In conclusion, we introduced Newton-PIPG for solving optimal control quadratic programming problems, which combines an operator-splitting method, PIPG, with second-order Newton steps. We demonstrated the

Table 1
Performance and speedup comparison for the oscillating mass example (All times in milliseconds.)

Umax	N	SCS-Anderson	OSQP	SCS	ECOS	PIPG	Newton-PIPG
1	20	3.48	3.47	3.93	16.84	9.42	**0.61**
0.4	20	6.97	3.74	7.22	17.30	8.75	**1.53**
1	50	14.88	11.48	14.25	40.80	33.73	**2.10**
0.4	50	21.02	10.41	20.96	40.91	43.14	**7.68**
1	100	46.72	72.82	73.11	83.45	149.36	**4.88**
0.4	100	51.50	25.47	52.71	84.05	161.99	**16.28**

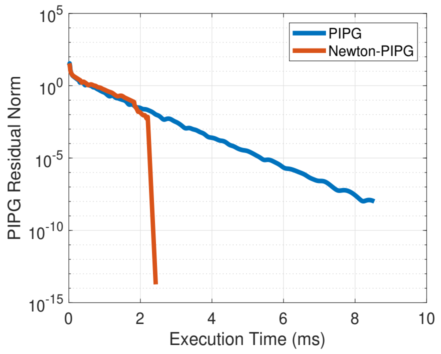

Fig. 2. Comparison of residuals versus solve time for Newton-PIPG and PIPG algorithms for the oscillating masses example.

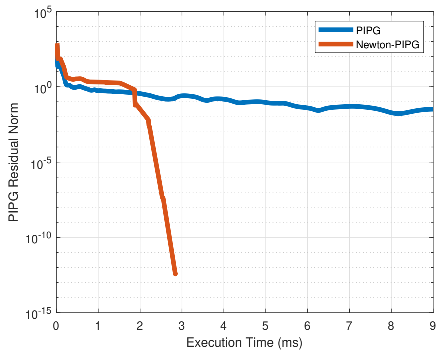

Fig. 3. Comparison of residuals versus solve time for Newton-PIPG and PIPG algorithms for the PDG example, with $r_{\text{init}} = (0, 0, 2000)$ meters

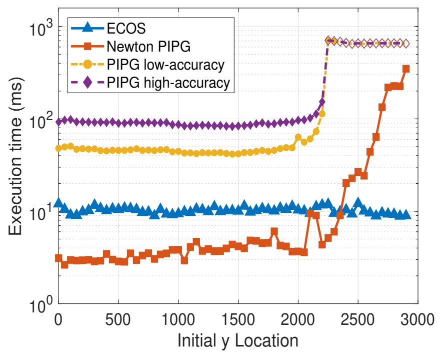

Fig. 4. Comparison of execution times for ECOS, Newton-PIPG, and PIPG with low accuracy (tolerance 10^{-4}) and high accuracy (tolerance 10^{-8}) across different initial y locations. Hollow markers represent non-converged points, which are omitted from the legend.

convergence of this algorithm and provided an efficient technique for solving the linear system in the Newton step. Our numerical experiments showed that our algorithm performs well compared to other state-of-the-art algorithms in solving quadratic optimal control problems.

For future research, we will incorporate infeasibility detection for Newton-PIPG and extend our algorithm to handle more general constraints. Additionally, the current Newton-PIPG software is written in Matlab, and we expect that a pure C/C++ implementation could further reduce computation time.

8 Appendix

8.1 PIPG Termination Criteria

We use a toy problem to show that the current termination criteria provide a solution with the desired accuracy. Consider the following problem:

$$\min_{x,y} (x-1)^2 + (y-1)^2$$
$$\text{s.t. } (x-x_0)^2 + (y-y_0)^2 \leq 1.$$

The analytical solution can be found by projecting the point $(1, 1)$ onto the unit circle centered at (x_0, y_0). In our example, we choose (x_0, y_0) to be $(0, 0)$, $(0, 1)$, $(0, 10)$, and $(0, 100)$. For each choice of (x_0, y_0), we select the coefficient α in the PIPG operator as a geometric sequence of 50 terms ranging from 5×10^{-4} to 0.5, and set ϵ_{abs} and ϵ_{rel} to 10^{-8}. Since there are no linear equality or inequality constraints, we only need to update the primal variable. With the converged result, we compute the absolute error of the PIPG algorithm's solution compared to the analytical solution.

For all α tested, the absolute error for the case of $(0, 0)$ lies in $[1.0 \times 10^{-8}, 1.7 \times 10^{-8}]$, for $(0, 1)$ it lies in $[9.4 \times 10^{-9}, 2.0 \times 10^{-8}]$, for $(0, 10)$ it lies in $[5.3 \times 10^{-9}, 3.0 \times 10^{-8}]$, and for $(0, 100)$ it lies in $[5.8 \times 10^{-9}, 1.2 \times 10^{-7}]$. Hence, the numerical results indicate that our termination criteria achieve the expected absolute error. For those who prioritize relative error, the termination criteria can be adjusted accordingly.

References

[Ali et al., 2017] Ali, A., Wong, E., and Kolter, J. Z. (2017). A semismooth newton method for fast, generic convex programming. In *International Conference on Machine Learning*, pages 70–79. PMLR.

[Andersen and Andersen, 2000] Andersen, E. D. and Andersen, K. D. (2000). The mosek interior point optimizer for linear programming: an implementation of the homogeneous algorithm. In *High performance optimization*, pages 197–232. Springer.

[Bauschke et al., 2011] Bauschke, H. H., Combettes, P. L., et al. (2011). *Convex analysis and monotone operator theory in Hilbert spaces*, volume 408. Springer.

[Borwein and Lewis, 2006] Borwein, J. and Lewis, A. (2006). *Convex Analysis*. Springer.

[Boyd and Vandenberghe, 2004] Boyd, S. P. and Vandenberghe, L. (2004). *Convex optimization*. Cambridge university press.

[Chen and Fukushima, 1999] Chen, X. and Fukushima, M. (1999). Proximal quasi-newton methods for nondifferentiable convex optimization. *Mathematical Programming*, 85:313–334.

[Clarke, 2013] Clarke, F. (2013). *Functional analysis, calculus of variations and optimal control*, volume 264. Springer.

[Dennis Jr and Schnabel, 1996] Dennis Jr, J. E. and Schnabel, R. B. (1996). *Numerical methods for unconstrained optimization and nonlinear equations*. SIAM.

[Domahidi et al., 2013] Domahidi, A., Chu, E., and Boyd, S. (2013). Ecos: An socp solver for embedded systems. In *2013 European control conference (ECC)*, pages 3071–3076. IEEE.

[Elango et al., 2022] Elango, P., Kamath, A. G., Yu, Y., Acikmese, B., Mesbahi, M., and Carson, J. M. (2022). A customized first-order solver for real-time powered-descent guidance. In *AIAA SciTech 2022 Forum*, page 0951.

[Elango et al., 2024] Elango, P., Luo, D., Uzun, S., Kim, T., and Acikmese, B. (2024). Successive convexification for trajectory optimization with continuous-time constraint satisfaction. *arXiv preprint arXiv:2404.16826*.

[Grant and Boyd, 2014] Grant, M. and Boyd, S. (2014). CVX: Matlab software for disciplined convex programming, version 2.1. https://cvxr.com/cvx.

[Gurobi Optimization, LLC, 2024] Gurobi Optimization, LLC (2024). Gurobi Optimizer Reference Manual.

[Hare and Lewis, 2004] Hare, W. L. and Lewis, A. S. (2004). Identifying active constraints via partial smoothness and prox-regularity. *Journal of Convex Analysis*, 11(2):251–266.

[Houska et al., 2011] Houska, B., Ferreau, H. J., and Diehl, M. (2011). An auto-generated real-time iteration algorithm for nonlinear mpc in the microsecond range. *Automatica*, 47(10):2279–2285.

[Jerez et al., 2014] Jerez, J. L., Goulart, P. J., Richter, S., Constantinides, G. A., Kerrigan, E. C., and Morari, M. (2014). Embedded online optimization for model predictive control at megahertz rates. *IEEE Transactions on Automatic Control*, 59(12):3238–3251.

[Jiang and Vandenberghe, 2023] Jiang, X. and Vandenberghe, L. (2023). Bregman three-operator splitting methods. *Journal of Optimization Theory and Applications*, 196(3):936–972.

[Jones et al., 2012] Jones, C. N., Domahidi, A., Morari, M., Richter, S., Ullmann, F., and Zeilinger, M. (2012). Fast predictive control: Real-time computation and certification. *IFAC Proceedings Volumes*, 45(17):94–98.

[Kamath et al., 2023] Kamath, A. G., Elango, P., Kim, T., Mceowen, S., Yu, Y., Carson, J. M., Mesbahi, M., and Acikmese, B. (2023). Customized real-time first-order methods for onboard dual quaternion-based 6-dof powered-descent guidance. In *AIAA SciTech 2023 Forum*, page 2003.

[Lewis, 2002] Lewis, A. S. (2002). Active sets, nonsmoothness, and sensitivity. *SIAM Journal on Optimization*, 13(3):702–725.

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

[Malyuta et al., 2022] Malyuta, D., Reynolds, T. P., Szmuk, M., Lew, T., Bonalli, R., Pavone, M., and Açıkmeşe, B. (2022). Convex optimization for trajectory generation: A tutorial on generating dynamically feasible trajectories reliably and efficiently. *IEEE Control Systems Magazine*, 42(5):40–113.

[Mordukhovich et al., 2014] Mordukhovich, B. S., Outrata, J. V., and Sarabi, M. E. (2014). Full stability of locally optimal solutions in second-order cone programs. *SIAM Journal on Optimization*, 24(4):1581–1613.

[Neunert et al., 2016] Neunert, M., De Crousaz, C., Furrer, F., Kamel, M., Farshidian, F., Siegwart, R., and Buchli, J. (2016). Fast nonlinear model predictive control for unified trajectory optimization and tracking. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 1398–1404. IEEE.

[Nocedal and Wright, 1999] Nocedal, J. and Wright, S. J. (1999). *Numerical optimization*. Springer.

[O'Donoghue, 2021] O'Donoghue, B. (2021). Operator splitting for a homogeneous embedding of the linear complementarity problem. *SIAM Journal on Optimization*, 31:1999–2023.

[Rockafellar, 1970] Rockafellar, R. T. (1970). *Convex Analysis*, volume 28 of *Princeton Mathematical Series*. Princeton University Press, Princeton, NJ.

[Saad and Schultz, 1986] Saad, Y. and Schultz, M. H. (1986). Gmres: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on scientific and statistical computing*, 7(3):856–869.

[Sopasakis et al., 2019] Sopasakis, P., Menounou, K., and Patrinos, P. (2019). Superscs: fast and accurate large-scale conic optimization. In *2019 18th European Control Conference (ECC)*, pages 1500–1505. IEEE.

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

[Szmuk et al., 2020] Szmuk, M., Reynolds, T. P., and Açıkmeşe, B. (2020). Successive convexification for real-time six-degree-of-freedom powered descent guidance with state-triggered constraints. *Journal of Guidance, Control, and Dynamics*, 43(8):1399–1413.

[Themelis and Patrinos, 2019] Themelis, A. and Patrinos, P. (2019). Supermann: a superlinearly convergent algorithm for finding fixed points of nonexpansive operators. *IEEE Transactions on Automatic Control*, 64(12):4875–4890.

[Wang and Boyd, 2009] Wang, Y. and Boyd, S. (2009). Fast model predictive control using online optimization. *IEEE Transactions on control systems technology*, 18(2):267–278.

[Wright, 1993] Wright, S. J. (1993). Interior point methods for optimal control of discrete time systems. *Journal of Optimization Theory and Applications*, 77(1):161–187.

[Yu et al., 2020] Yu, Y., Elango, P., and Açıkmeşe, B. (2020). Proportional-integral projected gradient method for model predictive control. *IEEE Control Systems Letters*, 5(6):2174–2179.

[Yu et al., 2022] Yu, Y., Elango, P., Açıkmeşe, B., and Topcu, U. (2022). Extrapolated proportional-integral projected gradient method for conic optimization. *IEEE Control Systems Letters*, 7:73–78.

[Zhang et al., 2020] Zhang, J., O'Donoghue, B., and Boyd, S. (2020). Globally convergent type-i anderson acceleration for nonsmooth fixed-point iterations. *SIAM Journal on Optimization*, 30(4):3170–3197.

[Zometa et al., 2013] Zometa, P., Kögel, M., and Findeisen, R. (2013). μao-mpc: A free code generation tool for embedded real-time linear model predictive control. In *2013 American Control Conference*, pages 5320–5325. IEEE.