Received: Added at production | Revised: Added at production | Accepted: Added at production

DOI: xxx/xxxx

RESEARCH ARTICLE

A double-layer Jacobi method for partial differential equation-constrained nonlinear model predictive control†

Haoyang Deng | Toshiyuki Ohtsuka

Department of Systems Science, Graduate School of Informatics, Kyoto University, Kyoto, Japan

Correspondence
Haoyang Deng, Department of Systems Science, Graduate School of Informatics, Kyoto University, Kyoto, Japan. Email: deng.haoyang@outlook.com

Present Address
Department of Systems Science, Graduate School of Informatics, Kyoto University, Kyoto, Japan

Abstract

This paper presents a real-time optimization method for nonlinear model predictive control (NMPC) of systems governed by partial differential equations (PDEs). The NMPC problem to be solved is formulated by discretizing the PDE system in space and time by using the finite difference method. The proposed method is called the double-layer Jacobi method, which exploits both the spatial and temporal sparsities of the PDE-constrained NMPC problem. In the upper layer, the NMPC problem is solved by ignoring the temporal couplings of either the state or costate (Lagrange multiplier corresponding to the state equation) equations so that the spatial sparsity is preserved. The lower-layer Jacobi method is a linear solver dedicated to PDE-constrained NMPC problems by exploiting the spatial sparsity. Convergence analysis indicates that the convergence of the proposed method is related to the prediction horizon. Results of a numerical experiment of controlling a heat transfer process show that the proposed method can be two orders of magnitude faster than the conventional Newton's method exploiting the banded structure of NMPC problems.

KEYWORDS:
Nonlinear model predictive control; PDE systems; real-time optimization; Jacobi iteration; Gauss-Seidel iteration

1 | INTRODUCTION

Nonlinear model predictive control (NMPC), also referred to as nonlinear receding horizon control or nonlinear moving horizon control, is an optimization-based control method for nonlinear systems. The control input is obtained by solving an optimization problem that usually minimizes a tracking cost under the constraints of the system dynamics. Moreover, general input and state constraints or economic costs can be integrated into the optimization problem, making NMPC a powerful advanced control technique and a popular research topic. On the other hand, the generality of NMPC brings about computational difficulties in solving the underlying optimization problem in real time. Considerable efforts and progress have been made toward the real-time NMPC control of systems described by ordinary differential equations (ODEs) in recent years, ranging over automatic code generation[1], first-order iteration (e.g.,[2]), and parallel computing[3], to name just a few.

Besides the NMPC control of ODE systems, NMPC control of systems described by partial differential equations (PDEs), such as Navier-Stokes equations for fluid flow and heat transfer equations for chemical processes, has gained increasing attention due to the optimal and constraint-handling properties of NMPC. However, PDE-constrained NMPC presents a great challenge for real-time optimization due to the infinite-dimensional state space of PDE systems. The solution methods for PDE-constrained

†This work was partly supported by JSPS KAKENHI Grant Numbers 15H02257 and 22H01510.

NMPC can be generally categorized as indirect or direct. Indirect methods analytically derive the optimality conditions for PDE-constrained NMPC and then solve these conditions numerically. For example, in[4], the analytic optimality conditions for the NMPC control of a class of parabolic PDEs are derived. These optimality conditions are discretized to a set of nonlinear algebraic equations, the solution of which is then traced by using the C/GMRES method[5]. Moreover, the so-called contraction mapping method in[4] can be applied efficiently if the nonlinear algebraic equations satisfy certain structure conditions. In contrast to indirect methods, which "first optimize, then discretize," direct methods need PDE systems to be first discretized both in space and time. The NMPC problem is formulated on the basis of the discretized PDE system, which leads to a nonlinear program (NLP). Since a fine-grained spatial discretization results in a large number of states or optimization variables, model reduction techniques are frequently applied. A common way, e.g.,[6], is to combine the proper orthogonal decomposition method[7] with Galerkin projection to obtain a low-dimensional dynamical system.

Unlike the model reduction methods, this paper deals directly with the discretized PDE system. Although conventional NMPC methods for ODE systems can in principle be applied, they are computationally expensive due to the large number of states. For example, methods exploiting the banded structure of NMPC, e.g.,[8] and[9], have computational complexities of $\mathcal{O}(N(n_u + n_x)^3)$, where N is the number of the temporal discretization grid points and n_u and n_x are the number of the control inputs and states, respectively. Note that sparsity exists both in the spatial and temporal directions of PDE-constrained NMPC problems. Structure-exploiting methods can only make use of the temporal sparsity along the prediction horizon, and the spatial sparsity is destroyed due to Riccati recursion. Consequently, dense matrix factorization, which leads to the cubic computational complexity, needs to be performed during recursion. An alternative approach is to first eliminate the state variables. The elimination procedure is roughly of $\mathcal{O}(N^2 n_x^2)$[10].

In this paper, we present a double-layer Jacobi method that exploits both the temporal and spatial sparsities. The upper-layer Jacobi (or Jacobi-type) method is derived by ignoring the temporal couplings of either the state or costate equations such that the spatial sparsity can be preserved. The upper-layer Jacobi method can be applied to the NMPC control of not only PDE systems, but also general nonlinear systems. The upper-layer Jacobi method can be seen as a block-wise Jacobi method for solving nonlinear equations[11]. Compared with the element-wise Jacobi method whose convergence is difficult to guarantee, the proposed method is shown to converge by choosing a short prediction horizon. Instead of using general sparse solvers to solve the sparse linear equations arising from the upper-layer Jacobi method, the lower-layer Jacobi method solves the sparse equations iteratively by exploiting the spatial sparsity of PDE-constrained NMPC. The performance of the proposed method is assessed by controlling the temperature distribution of a two-dimensional heat transfer process on a thin plate. The proposed method is matrix-free and has a per-iteration complexity of $\mathcal{O}(N(n_u + n_x))$ for the heat transfer example. The numerical example shows that the proposed method can be two orders of magnitude faster than the conventional structure-exploiting method.

This paper is organized as follows. PDE systems and their spatial discretization are described in Section 2. The NMPC problem and its Karush-Kuhn-Tucker (KKT) conditions are given in Section 3. The proposed double-layer Jacobi method is introduced in Section 4. Section 5 demonstrates the performance of the proposed method. Finally, this paper is summarized in Section 6.

1.1 | Notations

Let $v_{(i)}$ be the i-th component of a vector $v \in \mathbb{R}^n$. For a matrix $P \in \mathbb{R}^{n\times n}$, we denote $\rho(P)$ as the spectral radius of P. The symbol $\|\cdot\|$ denotes the Euclidean norm for a vector and the Frobenius norm for a matrix. The weighted norm is defined as $\|v\|_P := \sqrt{v^\top P v}$. For an iteration variable s, we denote s^k as the value of s at the k-th iteration and s^* as the optimal solution or fixed point. For a differentiable function $f(v) : \mathbb{R}^n \to \mathbb{R}^m$, we denote $\nabla_v f \in \mathbb{R}^{m\times n}$ as the transpose of the Jacobian matrix of f. Identity and zero matrices are denoted by I and 0, respectively, and their sizes are indicated by using subscripts if necessary.

2 | PDE SYSTEMS

In this paper, we consider the NMPC control of a general class of PDE systems defined on the spatial domain $\Omega \subset \mathbb{R}^n$ and temporal domain $\Gamma \subset \mathbb{R}$:

$$a(u(t), w(p,t))\frac{\partial^2 w(p,t)}{\partial t^2} + b(u(t), w(p,t))\frac{\partial w(p,t)}{\partial t} = c(u(t), w(p,t))\Delta w(p,t) + d(u(t), w(p,t)), \tag{1}$$

where $p \in \Omega$ is the spatial variable, $u \in \mathbb{U} \subset \mathbb{R}^{n_u}$ is the control input, $w \in \mathbb{W} \subset \mathbb{R}$ is the PDE state, $\Delta w(p,t) := \sum_{i=1}^n \partial^2 w(p,t)/\partial p_{(i)}^2$ denotes the Laplacian of w, and a, b, c and $d : \mathbb{U}\times\mathbb{W} \to \mathbb{R}$ are twice-differentiable nonlinear functions

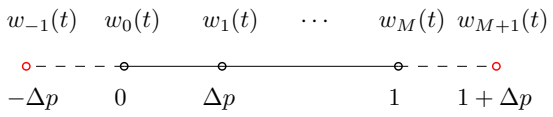


Figure 1 Spatial discretization points and fictitious points

of u and w. The boundary conditions, such as the Dirichlet and Neumann boundary conditions, can be given to be input- and state-dependent, i.e., as functions of u and w. We assume that a is not zero for every $(u, w) \in \mathbb{U} \times \mathbb{W}$ when the second-order time derivative of w is involved.

Note that (1) is a very general description of PDE systems. Many of the PDE systems, such as the heat transfer equation and wave equation, fall into the form of (1). For PDE systems that are not in this form, e.g., the Navier-Stokes equations including both gradients and algebraic variables, we will discuss later in Remark 2 that the results of this paper can in principle be extended.

2.1 | Spatial discretization

We first introduce the spatial discretization of (1) by using the finite difference method. Without loss of generality, we demonstrate the discretization by using a one-dimensional system on an unit space interval $\Omega := [0, 1]$ satisfying the following Neumann boundary condition:

$$\frac{\partial w(p,t)}{\partial p} = e(u(t), w(p,t)), \ p = 0 \text{ and } 1, \tag{2}$$

where e is a given function of u and w. Let $M + 1$ be the number of the spatial discretization grid points and Δp be the corresponding step size. The finite difference method consists in approximating derivatives by using finite differences, i.e., for $j \in \{0, \cdots, M\}$,

$$\Delta w(j\Delta p, t) \approx \frac{w_{j+1}(t) - 2w_j(t) + w_{j-1}(t)}{\Delta p^2},$$

where $w_j(t) := w(j\Delta p, t)$. At $j = 0$ and M, two fictitious points $w_{-1}(t)$ and $w_{M+1}(t)$, as illustrated in Fig. 1, are introduced to deal with the boundary condition. By using the finite difference method, the Neumann boundary condition (2) translates into the difference equations as follows.

$$\frac{w_1(t) - w_{-1}(t)}{2\Delta p} = e(u(t), w_0(t)), \ \frac{w_{M+1}(t) - w_{M-1}(t)}{2\Delta p} = e(u(t), w_M(t)). \tag{3}$$

The PDE system (1) is then discretized into

$$a(u(t), w_j(t))\ddot{w}_j(t) + b(u(t), w_j(t))\dot{w}_j(t) = c(u(t), w_j(t))\frac{w_{j+1}(t) - 2w_j(t) + w_{j-1}(t)}{\Delta p^2} + d(u(t), w_j(t)), \ j \in \{0, \cdots, M\}, \tag{4}$$

where the fictitious points $w_{-1}(t)$ and $w_{M+1}(t)$ can be eliminated by using the discretized boundary conditions (3). Note that the discretized PDE system (4) is described by a finite number of states:

$$x(t) := (W(t), \dot{W}(t)) := (w_0(t), \cdots, w_M(t), \dot{w}_0(t), \cdots, \dot{w}_M(t)) \in \mathbb{R}^{n_x},$$

where $n_x = 2(M + 1)$. The dynamics of the discretized PDE system are given by

$$\dot{x}(t) = \begin{bmatrix} \dot{W}(t) \\ g(u(t), x(t)) \end{bmatrix} =: f(u(t), x(t)), \tag{5}$$

where $g(u(t), x(t))$ denotes the expression of $\ddot{W}(t)$ obtained from (4).

3 | NMPC

The spatial discretization approximates the PDE system (1) into an ODE system (5), which is used to design the NMPC controller. For a prediction horizon $T > 0$, we consider the following N-stage NMPC problem based on the backward Euler's method with

a temporal discretization step size of $h := T/N$:

$$\begin{aligned} \min_{\substack{u_1,\cdots,u_N \\ x_0,\cdots,x_N}} \ & \sum_{i=1}^{N} l_i(u_i, x_i) \\ \text{s.t.} \quad & x_0 = \bar{x}_0, \\ & x_i = x_{i-1} + hf(u_i, x_i), \quad i \in \{1, \cdots, N\}, \\ & u_i \in \mathbb{U},\ x_i \in \mathbb{X}, \quad i \in \{1, \cdots, N\}, \end{aligned} \tag{6}$$

where $\bar{x}_0$ is the initial state, $\mathbb{U} := [\underline{u}, \bar{u}]$ and $\mathbb{X} := [\underline{x}, \bar{x}]$ are the admissible sets of u and x (the boundaries $\underline{u}$, $\bar{u}$, $\underline{x}$, and $\bar{x}$ are given), respectively, and $l_i(u, x) : \mathbb{U} \times \mathbb{X} \to \mathbb{R}$ is the stage cost function, which is assumed to be twice differentiable. Discretization using the backward Euler's method makes notations easy throughout the paper and leads to succinct algorithms. Discretization using other methods is discussed in Remark 3.

3.1 | Relaxation

We adopt the interior-point method to relax the NMPC problem by transferring the bound inequality constraints into a logarithmic barrier function added to the cost. To simplify the notation, the bound inequality constraints are put into a single vector-valued function

$$G(u, x) \geq 0.$$

We obtain the following relaxed NMPC problem:

$$\begin{aligned} \min_{\substack{u_1,\cdots,u_N \\ x_0,\cdots,x_N}} \ & \sum_{i=1}^{N} \bar{l}_i(u_i, x_i) \\ \text{s.t.} \quad & x_0 = \bar{x}_0, \\ & x_i = x_{i-1} + hf(u_i, x_i), \quad i \in \{1, \cdots, N\}, \end{aligned} \tag{7}$$

where $\bar{l}_i(u, x) := l_i(u, x) - \tau \sum_j \ln G_{(j)}(u, x)$ ($\tau > 0$ is the barrier parameter). The set of solutions to the relaxed NMPC problem (7) approaches the set of solutions to the original NMPC problem (6) when $\tau \to 0$.

For the consideration of feasibility, the state constraints usually need to be softened. The softened NMPC problem can also be relaxed and written in the form of (7).

3.2 | KKT conditions

Let $\lambda_i \in \mathbb{R}^{n_x}$ be the Lagrange multiplier (costate) corresponding to the i-th state equation. For the sake of brevity, we define

$$s := (x, u, \lambda) \text{ and } S := (s_1, \cdots, s_N).$$

Let $H_i(s)$ be the Hamiltonian defined by

$$H_i(s) := \bar{l}_i(u, x) + h\lambda^\top f(u, x).$$

Let $\mathcal{K}_i(x_{i-1}, s_i, \lambda_{i+1})$ be defined by

$$\mathcal{K}_i(x_{i-1}, s_i, \lambda_{i+1}) := \begin{bmatrix} x_{i-1} - x_i + hf(u_i, x_i) \\ \nabla_u H_i(s_i) \\ \lambda_{i+1} - \lambda_i + \nabla_x H_i(s_i) \end{bmatrix}$$

with $x_0 = \bar{x}_0$ and $\lambda_{N+1} = 0$. The KKT conditions for the relaxed NMPC problem (7) are

$$\mathcal{K}_i(x_{i-1}^*, s_i^*, \lambda_{i+1}^*) = 0,\ i \in \{1, \cdots, N\}, \tag{8}$$

which are the first-order conditions for optimality of (7). Although the KKT conditions (8) are only the necessary conditions for optimality and globalization strategies are needed to find a local minimizer in general, the proposed method focuses only on solving the KKT conditions (8).

We introduce the following shorthand at the k-th iteration:

$$\mathcal{K}_i^k := \mathcal{K}_i(x_{i-1}^k, s_i^k, \lambda_{i+1}^k),\ \mathcal{K}^k := (\mathcal{K}_1^k, \cdots, \mathcal{K}_N^k), \nabla_{s_i}\mathcal{K}_i^k := \nabla_{s_i}\mathcal{K}_i(x_{i-1}^k, s_i^k, \lambda_{i+1}^k),\ \text{etc.} \tag{9}$$

3.3 | Newton's method

In solving the KKT conditions (8) by using the Newton's method, the search direction $\Delta S^k := (\Delta s_1^k, \cdots, \Delta s_N^k)$ is obtained by solving the following KKT system:

$$\begin{bmatrix} \ddots & & & & \\ \cdots & D_{i-1}^k & M_U & & \\ & M_L & D_i^k & M_U & \\ & & M_L & D_{i+1}^k & \cdots \\ & & & & \ddots \end{bmatrix} \begin{bmatrix} \vdots \\ \Delta s_{i-1}^k \\ \Delta s_i^k \\ \Delta s_{i+1}^k \\ \vdots \end{bmatrix} = \begin{bmatrix} \vdots \\ \mathcal{K}_{i-1}^k \\ \mathcal{K}_i^k \\ \mathcal{K}_{i+1}^k \\ \vdots \end{bmatrix}. \tag{10}$$

Here, $D_i^k := \nabla_{s_i}^\top \mathcal{K}_i^k$ (the expression can be found in (17)) and the constant matrices M_L and M_U given as follows show the linear couplings of the state and costate equations, respectively.

$$M_L := \begin{bmatrix} I_{n_x} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \; M_U := \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & I_{n_x} \end{bmatrix}.$$

After the search direction is calculated, a line search is performed to guarantee the primal feasibility $G(u, x) \geq 0$, i.e.,

$$S^{k+1} = S^k - \alpha^{\max} \Delta S^k,$$

where $\alpha^{\max}$ is obtained from the fraction-to-the-boundary rule[12]:

$$\alpha^{\max} = \max\{\alpha^{\max} \in (0, 1] : \; G_i^{k+1} \geq 0.005 G_i^k, \; i \in \{1, \cdots, N\}\}. \tag{11}$$

4 | DOUBLE-LAYER JACOBI METHOD

This section introduces the proposed double-layer Jacobi method, which makes use of both the sparsities in the upper-level KKT matrix in (10) and the lower-level Jacobian matrices D_i^k, $i \in \{1, \cdots, N\}$. The upper-layer Jacobi method is a general NMPC optimization method that is not limited to a particular class of dynamical systems. Convergence for the upper-layer Jacobi method is analyzed, and some variants of the Jacobi method are given. The lower-layer Jacobi method is a linear solver dedicated to the NMPC control of PDE systems by exploiting their particular structures after spatial discretization. We first review some general results on the convergence of iterative methods in the following subsection.

4.1 | Preliminaries

We first review the Jacobi method (see, e.g.,[11]) for solving linear equations as follows. Lemma 1 states the iteration and the convergence condition of the Jacobi method. The convergence condition for the general iteration (13), of which the Jacobi iteration is a special case, is given in Lemma 2. In order to indicate how fast the iteration (13) converges to a point of attraction, the convergence factor and rate are defined in Lemma 3.

Lemma 1. Let

$$Av = b, \tag{12}$$

where $A \in \mathbb{R}^{n \times n}$ and $b \in \mathbb{R}^n$. Let A be decomposed into $A = D + L + U$, where D, L, and U are the diagonal, strict lower triangular, and strict upper triangular elements (blocks) of A, respectively. Assume that D is invertible. Then, the Jacobi method for solving (12) is described by

$$v^{k+1} = D^{-1}(b - (L + U)v^k).$$

The Jacobi method converges if and only if

$$\rho(D^{-1}(L + U)) < 1.$$

Likewise, for the element-wise decomposition, the Jacobi method converges if the matrix A is strictly diagonally dominant.

Some results on the convergence of general iterations are shown as follows.

Definition 1. (Point of attraction[13]). Consider the iteration

$$v^{k+1} = T(v^k), \tag{13}$$

where $v \in \mathbb{R}^n$ and $T : P \to \mathbb{R}^n$ for a subset $P \subset \mathbb{R}^n$. Let v^* be an interior point of P and a fixed point of the iteration (13), i.e., $v^* = T(v^*)$. Then, v^* is said to be a point of attraction of the iteration (13) if there is an open neighborhood $O \subset P$ of v^* such that the iterates defined by (13) all lie in O and converge to v^* for any $v^0 \in O$.

Lemma 2. [13] Assume that T is continuously differentiable. Consider the iteration (13). Then, v^* is a point of attraction of the iteration (13) if the following condition holds:

$$\rho(\nabla_v^\top T(v^*)) < 1.$$

Lemma 3. (Convergence factor and rate [13]). If v^* is a point of attraction of the iteration (13), the following holds:

$$\rho(\nabla_v^\top T(v^*)) = \lim_{k\to\infty} \sup \|v^k - v^*\|^{1/k},$$

and $\rho(\nabla_v^\top T(v^*))$ is called the convergence factor. The convergence rate is defined by $-\ln \rho(\nabla_v^\top T(v^*))$.

4.2 | Upper-layer Jacobi method

Let D^k, L, and U be the diagonal, strict lower triangular, and strict upper triangular blocks of the KKT matrix in (10) as follows.

$$\begin{aligned} D^k &:= \text{block-diag}(D_1^k, \cdots, D_N^k), \\ L &:= \text{lower-block-diag}(M_L, \cdots, M_L), \\ U &:= \text{upper-block-diag}(M_U, \cdots, M_U). \end{aligned} \tag{14}$$

Since each block D_i^k in D^k corresponds to the KKT matrix of a single-stage NMPC problem, the block-diagonal matrix D^k can be assumed to be invertible without loss of generality. The upper-layer Jacobi method for solving the KKT conditions (8) is given by

$$S^{k+1} = S^k - \alpha^{\max}(D^k)^{-1}\mathcal{K}^k, \tag{15}$$

where $\alpha^{\max} \in (0, 1]$ is a scalar obtained from the fraction-to-the-boundary rule (11) and S^0 is chosen such that the primal feasibility condition $G(u_i, x_i) > 0$ is satisfied for all $i \in \{1, \cdots, N\}$. Note that since only bound constraints are introduced, an initial guess S_0 that satisfies the primal feasibility condition $G(u, x) > 0$ can be easily found. By introducing slack input variables, general inequality constraints can be transferred into equality constraints and bound inequality constraints on the slack variables. Therefore, the barrier function for bound constraints can also be applied and an feasible initial guess can also be found, and the results of this paper can in principle be extended to NMPC problems with general inequality constraints.

The upper-layer Jacobi method exploits the banded structure (temporal sparsity) of the KKT matrix by ignoring its off-diagonal blocks. Since D^k is a block-diagonal matrix, the iteration (15) can be performed block-wisely. With the particular stage-wise ordering of the variables, the couplings of the neighboring stages are reduced to be linear, i.e., the off-diagonal blocks are constant matrices. Although the Jacobi method in (15) can be regarded as Newton's method ignoring constant off-diagonal blocks and Newton's method is known to be locally quadratically convergent under mild assumptions, the convergence property might not be preserved for the Jacobi method. The convergence of the upper-layer Jacobi method is analyzed in the following subsection.

4.2.1 | Convergence

We first give a general convergence condition for the upper-layer Jacobi method.

Theorem 1. S^* is a point of attraction of the iteration (15) if the following condition holds:

$$\rho((D^*)^{-1}(L + U)) < 1, \tag{16}$$

where $D^* := D(S^*)$.

Proof. Let $w_i \in \mathbb{C}$, $i \in \{1, \cdots, 2n_x + n_u\}$ be the eigenvalues of $(D^*)^{-1}(L + U)$, where D, L, and U are defined in (14). From the definition of spectral radius, the condition (16) is equivalent to $\max_i |w_i| < 1$. Together with $\alpha^{\max} \in (0, 1]$, the inequality $\max_i |1 - \alpha^{\max} - \alpha^{\max} w_i| \le 1 - \alpha^{\max} + \alpha^{\max} \max_i |w_i| < 1$ holds. From the fact that the KKT conditions defined in (8) and (9) satisfy $\mathcal{K}^* = 0$ and $\nabla_S^\top \mathcal{K}^* = D^* + L + U$, the convergence factor for the iteration (15) is given by $\rho((1 - \alpha^{\max})I - \alpha^{\max}(D^*)^{-1}(L + U))$. The result then follows by applying Lemma 2 with the fact that $\rho((1 - \alpha^{\max})I - \alpha^{\max}(D^*)^{-1}(L + U)) = \max_i |1 - \alpha^{\max} - \alpha^{\max} w_i| < 1$. □

Note that (16) corresponds to the convergence condition for the iteration (15) when the fraction-to-the-boundary rule (11) is not triggered, i.e., $\alpha^{\max} = 1$. Theorem 1 indicates that if the convergence condition holds for $\alpha^{\max} = 1$, it also holds for $\alpha^{\max} \in (0, 1]$. Theorem 1 gives a general sufficient condition for convergence. However, the condition (16) can only be verified afterward and does not provide significant insights related to the NMPC problem. We show in the following lemma and theorem that the upper-layer Jacobi method can be guaranteed to converge by tuning the NMPC parameters, e.g., by shortening the prediction horizon T.

Lemma 4. Let D_i^* be decomposed into

$$\begin{aligned} D_i^* &= \begin{bmatrix} h\nabla_x^\top f_i^* - I & 0 & 0 \\ \nabla_{ux}^2 H_i^* & \nabla_{uu}^2 H_i^* & 0 \\ \nabla_{xx}^2 H_i^* & \nabla_{xu}^2 H_i^* & h\nabla_x^\top f_i^* - I \end{bmatrix} + h \begin{bmatrix} 0 & \nabla_u^\top f_i^* & 0 \\ 0 & 0 & \nabla_u f_i^* \\ 0 & 0 & 0 \end{bmatrix} \\ &=: \bar{D}_i^* + h\tilde{D}_i^*. \end{aligned} \tag{17}$$

Let $\bar{D}^*$ and $\tilde{D}^*$ be defined as follows.

$$\begin{aligned} \bar{D}^* &:= \text{block-diag}(\bar{D}_1^*, \cdots, \bar{D}_N^*), \\ \tilde{D}^* &:= \text{block-diag}(\tilde{D}_1^*, \cdots, \tilde{D}_N^*). \end{aligned}$$

Assume that $\bar{D}^*$ is invertible. Then, the following holds:

$$\rho((\bar{D}^*)^{-1}(L + U)) = 0.$$

Proof. The proof can be shown from the inspection of the special sparsity patterns of $\bar{D}_i^*$, M_L, and M_U, i.e., the block lower triangular structure of $\bar{D}_i^*$ and the "corner" structure of M_L and M_U. The details are given in Appendix A. □

Theorem 2. Assume that $\bar{D}^*$ defined in Lemma 4 is invertible. The convergence of the upper-layer Jacobi method (15) is described as follows.

(i) There exists $T_0 > 0$ such that $\rho((D^*)^{-1}(L + U)) < 1$ holds for any $T < T_0$.

(ii) There exists $\epsilon > 0$ such that $\rho((D^*)^{-1}(L + U)) < 1$ holds for any $\tilde{D}_i^*$ satisfying $\|\tilde{D}_i^*\| < \epsilon\|\bar{D}_i^*\|$, $i \in \{1, \cdots, N\}$.

Proof of Theorem 2 (i). From the definitions in Lemma 4, we know that

$$\rho((D^*)^{-1}(L + U)) = \rho((\bar{D}^* + h\tilde{D}^*)^{-1}(L + U)). \tag{18}$$

We first choose $T_0 > 0$ to be sufficiently small. Since $T < T_0$, $h = T/N$ is sufficiently small, and $\bar{D}^*$ is invertible as assumed, $(D^*)^{-1}$ exists and the right-hand side of (18) can be seen as a small perturbation of $\rho((\bar{D}^*)^{-1}(L + U))$ in terms of h. That is, together with Lemma 4, we obtain that

$$\lim_{h \to 0} \rho((D^*)^{-1}(L + U)) = 0.$$

From the continuities of matrix inverse and spectral radius, there always exists $T_0 > 0$ such that $\rho((D^*)^{-1}(L + U)) < 1$ holds for any $T < T_0$. □

Proof of Theorem 2 (ii). Let $\epsilon > 0$ be chosen to be sufficiently small. Since ϵ is a small number and $\|\tilde{D}_i^*\| < \epsilon\|\bar{D}_i^*\|$, $i \in \{1, \cdots, N\}$, $\rho((D^*)^{-1}(L + U)) = \rho((\bar{D}^* + h\tilde{D}^*)^{-1}(L + U))$ can also be seen as a small perturbation of $\rho((\bar{D}^*)^{-1}(L + U)) = 0$ in terms of $\tilde{D}^*$. Similarly to the proof of (i), the result then follows. □

Theorem 2 can be interpreted as follows. Theorem 2 (i) indicates that a short prediction horizon T guarantees the convergence of the upper-layer Jacobi method. However, note that a small $T > 0$ is not a necessary condition for convergence. The upper-layer Jacobi method might still converge for NMPC problems with long prediction horizons. For a problem with a long prediction horizon that the proposed method cannot converge, a small T together with an addition terminal cost[14] can be chosen so that it can behave as if its prediction horizon were long. Since $\tilde{D}_i^*$ consists of the sensitivity $\nabla_u f_i^*$, Theorem 2 (ii) indicates that $\rho((D^*)^{-1}(L + U)) < 1$ holds if the dynamical system (5) is not sensitive to the control input u. Note that Theorem 2 assumes the invertibility of the block-diagonal matrix $\bar{D}^*$. We discuss in Remark 1 the invertibility of $\bar{D}^*$ as follows.

Remark 1. (Invertibility of $\bar{D}^*$ and regularization). Although D^* can be assumed invertible without loss of generality, the invertibility assumption on $\bar{D}^*$ in Theorem 2 might not hold under some problem settings, e.g., when the dynamical system is linear and the stage cost function does not involve all inputs such that $\nabla_{uu}^2 H_i^*$ is singular. Moreover, even $\bar{D}^*$ is invertible, its

inverse $(\bar{D}^*)^{-1}$ might be ill-conditioned and the convergence condition (16) might not hold even when a small perturbation $h\tilde{D}^*$ is introduced. To tackle the problem of singularity, a regularization term γI with $\gamma \geq 0$ can be added to each $\nabla^2_{uu} H_i$ such that $(\bar{D}^*)^{-1}$ exists and is less ill-conditioned (less sensitive to $h\tilde{D}^*$). Note that the introduction of the regularization term γI does not change the solution to the NMPC problem when the iteration converges.

To speed up convergence, the variants (see, e.g., [11]) of the Jacobi method, such as the forward Gauss-Seidel method (FGS), the backward Gauss-Seidel method (BGS), and the symmetric Gauss-Seidel method (SGS), are usually applied in practice. The following are the FGS, BGS, and SGS iterations, respectively:

$$\begin{aligned} S^{k+1} &= S^k - \alpha^{\max}(D^k + L)^{-1}\mathcal{K}^k, \\ S^{k+1} &= S^k - \alpha^{\max}(D^k + U)^{-1}\mathcal{K}^k, \end{aligned} \tag{19}$$

and

$$S^{k+1} = S^k - \alpha^{\max}(D^k + L)^{-1}(\mathcal{K}^k - U(D^k + U)^{-1}\mathcal{K}^k). \tag{20}$$

Note that FGS and BGS have the same amount of computation as the Jacobi method. The difference is the rate of convergence. Since the full-step iteration $\alpha^{\max} = 1$ is typically observed when the iterates near the solution S^* [12], the rate of convergence for full-step iterations is considered in the following corollary.

Corollary 1. Suppose that the iteration is full-step, i.e., $\alpha^{\max} = 1$, in the neighborhood of S^*. If the convergence condition (16) holds for the Jacobi method, then the convergence conditions, which are given by $\rho((D^* + L)^{-1}U) < 1$ and $\rho((D^* + U)^{-1}L) < 1$, also hold for FGS and BGS, respectively. That is, S^* is a point of attraction of the FGS and BGS iterations. Moreover, both the FGS and BGS methods converge twice as fast as the Jacobi method.

Proof. Since the KKT matrix in (10) is a block-tridiagonal matrix, the KKT matrix is consistently ordered [15]. It is known from [11] that for a consistently ordered matrix, the spectral radius of FGS is the square of that of the Jacobi method, i.e.,

$$\rho((D^* + L)^{-1}U) = \rho((D^*)^{-1}(L + U))^2.$$

The conclusion above can be shown similarly for BGS that

$$\rho((D^* + U)^{-1}L) = \rho((D^*)^{-1}(L + U))^2.$$

Recall the definition of the convergence rate in Lemma 3. The result then follows. □

As can be seen from Theorem 2 and Corollary 1, a short prediction horizon T can also guarantee the convergence of FGS and BGS. The discussion on the role of the regularization procedure in Remark 1 applies to FGS and BGS as well. As for the SGS iteration (20), its convergence condition is given by $\rho((D^* + L)^{-1}U(D^* + U)^{-1}L) < 1$, and it can be seen as a BGS iteration followed by a FGS iteration, i.e., a backward sweep followed by a forward sweep. The iteration in the previous work [16] is similar to the FGS iteration (19). However, the inequality constraints are kept and the regularization procedure is not introduced, so the convergence is difficult to guarantee.

4.3 | Lower-layer Jacobi method

The upper-layer Jacobi method and its variants essentially consist of solving linear equations with the coefficient matrices D_i^k of the structure in (17) for $i \in \{1, \cdots, N\}$. Since D_i^k is sparse and its structure is fixed, efficient exact or iterative solution methods usually exist. For example, the Jacobi method for solving linear equations can be applied directly when h is sufficiently small. In this subsection, we introduce another iterative method by exploiting the particular structure of D_i^k.

The linear systems are reordered to have the following coefficient matrix (assume that the regularization term γI is introduced):

$$\begin{bmatrix} 0 & h\nabla_x^\top f_i^k - I & h\nabla_u^\top f_i^k \\ h\nabla_x f_i^k - I & \nabla^2_{xx} H_i^k & \nabla^2_{xu} H_i^k \\ h\nabla_u f_i^k & \nabla^2_{ux} H_i^k & \nabla^2_{uu} H_i^k + \gamma I \end{bmatrix}. \tag{21}$$

For the sake of brevity, the linear system with the coefficient matrix (21) is expressed by using the following shorthand:

$$\begin{bmatrix} 0 & F_x & F_u \\ F_x^\top & A_{xx} & A_{xu} \\ F_u^\top & A_{ux} & A_{uu} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}. \tag{22}$$

Equation (22) can be solved by first eliminating (v_1, v_2) and then solving for v_3, i.e., by performing the following two steps:

$$\left(A_{uu} - \begin{bmatrix} F_u^\top & A_{ux} \end{bmatrix} \begin{bmatrix} 0 & F_x \\ F_x^\top & A_{xx} \end{bmatrix}^{-1} \begin{bmatrix} F_u \\ A_{xu} \end{bmatrix}\right) v_3 = b_3 - \begin{bmatrix} F_u^\top & A_{ux} \end{bmatrix} \begin{bmatrix} 0 & F_x \\ F_x^\top & A_{xx} \end{bmatrix}^{-1} \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \tag{23a}$$

and

$$\begin{bmatrix} 0 & F_x \\ F_x^\top & A_{xx} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} - \begin{bmatrix} F_u \\ A_{xu} \end{bmatrix} v_3. \tag{23b}$$

Solving (23) consists of solving several linear equations of the following form:

$$\begin{bmatrix} 0 & F_x \\ F_x^\top & A_{xx} \end{bmatrix} \begin{bmatrix} v_4 \\ v_5 \end{bmatrix} = \begin{bmatrix} b_4 \\ b_5 \end{bmatrix}. \tag{24}$$

The linear equation (24) can be solved by first solving $F_x v_5 = b_4$ and then solving $F_x^\top v_4 = b_5 - A_{xx} v_5$. That is, linear equations with the coefficient matrices F_x and $F_x^\top$ are solved essentially. We show next that these linear equations can be solved efficiently by using the Jacobi method.

Recall that for the discretized PDE system (5), we have

$$F_x = h\nabla_x^\top f_i^k - I = \begin{bmatrix} -I & hI \\ h\nabla_W^\top g_i^k & h\nabla_{\dot{W}}^\top g_i^k - I \end{bmatrix}.$$

Therefore, a linear equation with the coefficient matrix F_x, i.e., the following equation,

$$\begin{bmatrix} -I & hI \\ h\nabla_W^\top g_i^k & h\nabla_{\dot{W}}^\top g_i^k - I \end{bmatrix} \begin{bmatrix} v_6 \\ v_7 \end{bmatrix} = \begin{bmatrix} b_6 \\ b_7 \end{bmatrix}$$

can be solved by first eliminating v_6 with $v_6 = hv_7 - b_6$ and then solving a linear system with the following coefficient matrix:

$$h\nabla_{\dot{W}}^\top g_i^k - I + h^2 \nabla_W^\top g_i^k. \tag{25}$$

It is easy to show that (25) is diagonally dominant if h is sufficiently small. Moreover, the off-diagonal entries of $\nabla_W^\top g_i^k$ are sufficiently small for many of the PDE equations, such as the heat transfer equation and the Navier-Stokes equation with a large Reynolds number. That is, according to Lemma 1, convergence of the Jacobi method can be achieved for solving linear equations with the coefficient matrices (25). The same conclusion can be made for the $F_x^\top$ system.

The lower-layer Jacobi method is concluded as follows. Since $n_x \gg n_u$ for PDE-constrained NMPC problems, the major computational cost for solving (22) comes from solving linear equations with the coefficient matrices (25), which can be solved efficiently by using the Jacobi method if, e.g., the NMPC problem is finely discretized in time, i.e., with a sufficiently small h.

The overall double-layer Jacobi method on the basis of the SGS iteration is summarized in Algorithm 1.

Algorithm 1 Double-layer Jacobi method on the basis of SGS iteration

Input: initial state $\bar{x}_0$, barrier parameter $\tau > 0$, regularization parameter $\gamma \geq 0$, $i \in \{1, \cdots, N\}$, initial guess S^0
Output: S^*

Initialization: $k = 0$; $\Delta\bar{\lambda}_{N+1} = 0$; $\Delta x_0 = 0$
repeat
  Evaluate $\mathcal{K}^k$ and D^k
  for $i = N$ to 1 **do**
    Solve: $D_i^k \Delta\bar{s}_i^k = \mathcal{K}_i^k - (0, 0, \Delta\bar{\lambda}_{i+1}^k)$
  end for
  for $i = 1$ to N **do**
    Solve: $D_i^k \Delta s_i^k = \mathcal{K}_i^k - (\Delta x_{i-1}^k, 0, \Delta\bar{\lambda}_{i+1}^k)$
  end for
  $\alpha^{\max} \leftarrow (11)$
  $S^{k+1} \leftarrow S^k - \alpha^{\max} \Delta S^k$
  $k \leftarrow k + 1$
until termination criterion is met
$S^* \leftarrow S^k$

Remark 2. The major computational cost of Algorithm 1 comes from lines 5 and 8, i.e., solving the sparse system (22). Compared with general sparse solvers, dedicated solvers considering the particular structure of the spatial sparsity, e.g., the introduced lower-layer Jacobi method for the NMPC control of the PDE system (1), are expected to have better performance. For PDE systems that are not in the form of (1) or even general systems, the proposed method can be performed efficiently if a structure-exploiting linear solver can be designed for the sparse system (22).

Remark 3. Since different spatial discretization methods, such as the finite difference method and the finite volume method, only differ in the expression of g in (5) and the diagonally dominant property of (25) does not depend on the particular expression of g, the lower-layer method can be generalized to other spatial discretization methods. Regarding the temporal discretization method, as can be seen from Section 4.2.1 and 4.3, both the convergence result and the lower-layer iteration of the proposed method depend on that the state increment, i.e., $hf(u,x)$ for the Euler's method, is proportional to the step size h, which also holds for, e.g., the Runge-Kutta method. Therefore, the results of the proposed method can in principle be extended to other spatial and temporal discretization methods.

5 | NUMERICAL EXPERIMENT

In this section, we demonstrate the performance of the double-layer Jacobi method in terms of the computation time, number of iterations, and convergence factor by using a heat transfer closed-loop control example. The proposed method was implemented in C and the experiment was performed on a 3.9-GHz (turbo boost frequency) Intel Core i5-8265U laptop computer. To reduce the effect of the computing environment, the computation time at each time step was measured by taking the minimum one of ten runs of the closed-loop simulation. All of the comparison scripts can be found online (https://github.com/deng-haoyang/PDE-NMPC).

5.1 | System description

We consider a nonlinear heat transfer process in a thin copper plate[17]. Because the plate is relatively thin compared with the planar dimensions, temperature can be assumed constant in the thickness direction. The system is described by the following two-dimensional PDE:

$$\rho C_p t_z \frac{\partial w(p,t)}{\partial t} - k t_z \Delta w(p,t) + 2Q_c + 2Q_r = 0,$$

where w is the plate temperature, $p \in \Omega := \{(x,y)|x,y \in [0,1]\}$ (x here stands for the horizontal axis) and Q_c and Q_r are, respectively, the convection and radiation heat transfers defined as follows.

$$Q_c := h_c(w(p,t) - T_a),$$
$$Q_r := \epsilon\delta(w(p,t)^4 - T_a^4).$$

The boundary conditions are the zero Neumann boundary conditions. The parameters of the heat transfer process are given in Table 1.

Table 1 Parameters in the heat transfer process

ρ	8960	Density of copper [$\mathrm{kgm^{-3}}$]
C_p	386	Specific heat of copper [$\mathrm{Jkg^{-1}K^{-1}}$]
t_z	0.01	Plate thickness [m]
k	400	Thermal conductivity of copper [$\mathrm{Wm^{-1}K^{-1}}$]
h_c	1	Convection coefficient [$\mathrm{Wm^{-2}K^{-1}}$]
T_a	300	Ambient temperature [K]
ϵ	0.5	Emissivity of the plate surface
δ	$5.67 \cdot 10^{-8}$	Stefan-Boltzmann constant [$\mathrm{Wm^{-2}K^{-4}}$]

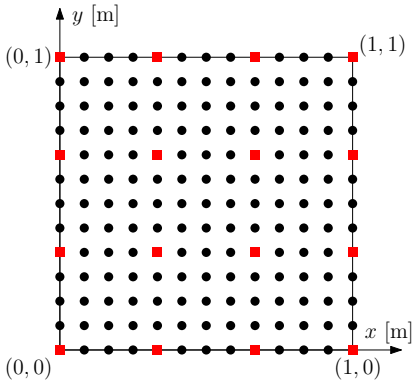


Figure 2 Spatial discretization grid points on plate (red squared points: actuators' positions)

There are 16 actuators distributed uniformly under the plate to heat or cool the plate above the ambient temperature in the range of $[T_a, T_a + 400]$ K. The temperatures of the plate at the positions of the actuators can be controlled directly. We assume that the actuators negligibly impact the convection and radiation heat transfer processes. We are interested in controlling the temperature distribution across the plate under constraints, which is a typical control problem that arises in semiconductor manufacturing. For example, a temperature gradient needs to be maintained within a wafer to ensure catalytic activation[18].

5.2 | NMPC description

The plate was uniformly discretized into 13×13 spatial grid points as shown in Fig. 2. Since the temperatures at the positions of the actuators can be controlled directly, the temperatures of the red squared points are regard as control inputs. We obtain a system with 16 inputs and 153 states. The inputs are constrained by

$$G(u, x) = \begin{bmatrix} u - T_a e \\ -u + (T_a + 400)e \end{bmatrix} \geq 0,$$

where $e = [1, \cdots, 1]^\top$. We chose the cost function to be quadratic as

$$l_i(u, x) := \frac{1}{2}(\|x - x_{ref}\|_Q^2 + \|u - u_{ref}\|_R^2), \ i \in \{1, \cdots, N\},$$

where x_{ref} and u_{ref} encoded the temperature distribution reference and the weighting matrices were $Q = I$ and $R = 0.1 \times I$. The generalized Gauss-Newton method[19] was used to approximate the Hessian of the Hamiltonian H_i, i.e., by using the Hessian of the cost function l_i.

Note that the lower-layer Jacobi method converged fast and needed only two iterations due to the small thermal diffusivity $k\rho^{-1}C_p^{-1}$. Moreover, we noticed that the coefficient matrix in (23a) was dominated by the diagonal matrix A_{uu}. The linear equation (23a) in the lower-layer Jacobi method was solved iteratively by performing two of the following iterations:

$$A_{uu}v_3^{k+1} = b_3 + \begin{bmatrix} F_u^\top & A_{ux} \end{bmatrix} \begin{bmatrix} 0 & F_x \\ F_x^\top & A_{xx} \end{bmatrix}^{-1} \left(\begin{bmatrix} F_u \\ A_{xu} \end{bmatrix} v_3^k - \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \right),$$

which can be solved efficiently due to the diagonal property of A_{uu}. The parameters of the NMPC controller are given in Table 2.

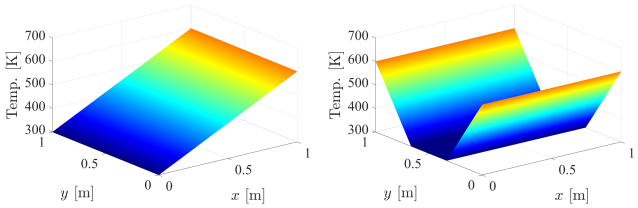
Since all of the matrices during iteration were sparse, the expressions of the matrix-vector multiplications were pre-computed offline symbolically, and the multiplications were performed by evaluating the corresponding expressions, which made the proposed method matrix-free. The computational complexity of the proposed method for the heat transfer example is $\mathcal{O}(N(n_x + n_u))$.

Table 2 NMPC parameters

Name	Value
Prediction horizon T	100 [s]
# of temporal discretization points N	20
Barrier parameter τ	100
Regularization parameter γ	0.5
Stopping criterion	$\|\mathcal{K}^k\|_\infty < 1$
Upper-layer method	SGS (20)
Lower-layer method	Two Jacobi iterations

5.3 | Closed-loop simulation

The system was started from an initial state of $\bar{x}_0 = [T_a, \cdots, T_a]^\top$. The simulation was performed for 1000 s with a sampling period of 5 s. The temperature distribution reference, as shown in Fig. 3, was set to a slope shape for the first 500 seconds and a V-like shape for the last 500 seconds. The second reference was fed to the controller by changing continuously from the first



(a) $0 \le t \le 500$ s **(b)** $500 < t \le 1000$ s

Figure 3 Temperature distribution references at different time periods

reference within 50 seconds.

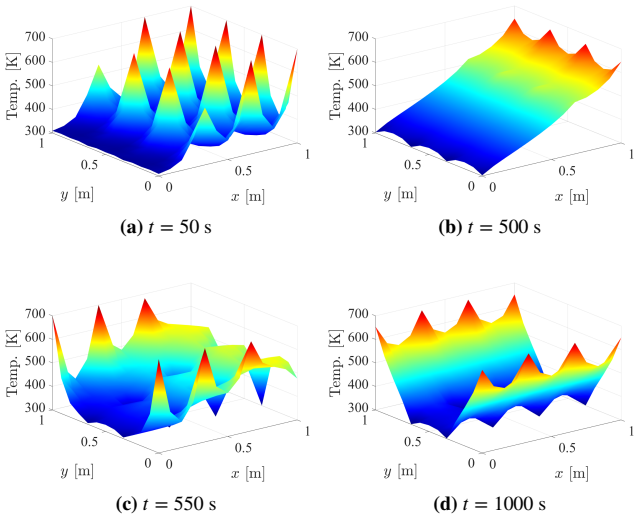
For tracking the first reference of the closed-loop simulation, two sampled plots at $t = 50$ s and $t = 500$ s are shown in Fig. 4 (a) and (b). For the second reference, two sampled plots at $t = 550$ s and $t = 1000$ s are shown in Fig. 4 (c) and (d). As shown by these plots, the references were tracked well by using the NMPC controller. The time histories of the control inputs are shown in Fig. 5. A high accuracy was still achieved for the chosen barrier parameter $\tau = 100$ and the stopping criterion $\|\mathcal{K}^k\|_\infty < 1$. To prove this claim, we measured the normalized suboptimality during the closed-loop simulation as:

$$e^\tau(t) := \frac{\sum_{i=1}^N |l_i(x_i^\tau(t), u_i^\tau(t)) - l_i(x_i^*(t), u_i^*(t))|}{\max_t \sum_{i=1}^N l_i(x_i^*(t), u_i^*(t))} \times 100\%,$$

where $(x_i^\tau(t), u_i^\tau(t))$ and $(x_i^*(t), u_i^*(t))$ are the optimal solutions to the relaxed NMPC problem (7) under the chosen settings and the original NMPC problem (6), respectively. The suboptimality $e^\tau(t)$ was less than 0.26% during simulation.

To demonstrate the performance of the proposed method, we set up the following experiments for comparison (all with the same barrier parameter and stopping criterion):

- Ipopt (multiple shooting): the original NMPC problem (6) was formulated in the CasADi[20] environment and solved by using Ipopt[21] with the direct sparse linear solver MUMPS[22].
- Ipopt (single shooting): the state variables were eliminated by forward simulating the state equations obtained by using the forward Euler's method. The settings were the same as the settings of Ipopt (multiple shooting).
- ParNMPC[23]: The conventional Newton's method introduced in Section 3.3 was implemented, and the KKT system (10) was solved by using the block Gaussian elimination method[23]. Compared with the proposed method, the recursion of elimination in ParNMPC destroys the lower-level spatial sparsity so that dense matrix factorization needs to be performed.



(a) $t = 50$ s **(b)** $t = 500$ s

(c) $t = 550$ s **(d)** $t = 1000$ s

Figure 4 Temperature distributions at different time t

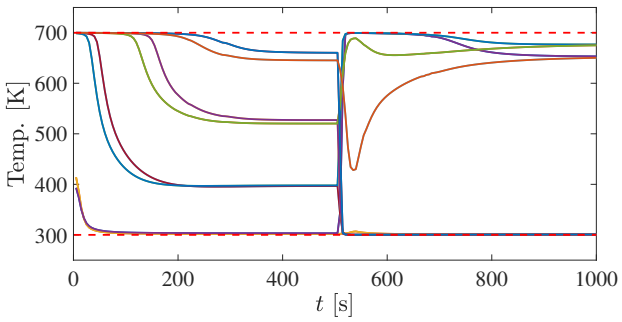


Figure 5 Time histories of inputs (some inputs coincide with each other)

Note that since all these methods were based on the interior-point method, their computation times per iteration were consistent throughout the closed-loop simulation. The mean computation time per iteration for Ipopt (multiple shooting), Ipopt (single shooting), ParNMPC, and the proposed method were 0.081, 0.046, 0.18, and 0.0041 seconds, receptively. Since the states were eliminated in Ipopt (single shooting) and the number of states was much more than the number of inputs, linear systems with smaller sizes were solved in Ipopt (single shooting) than in Ipopt (multiple shooting). Therefore, the mean computation time per iteration for Ipopt (single shooting) was smaller than Ipopt (multiple shooting). Considering that their numbers of iterations shown in Fig. 6 were in the same range, the proposed method was much faster than the other methods in terms of the computation time per time step shown in Fig. 7.

Lastly, we discuss the effects of the prediction horizon and regularization. In the numerical experiment, the proposed method could not converge without regularization ($\gamma = 0$). Since $\bar{D}^*$ is invertible under the problem setting and according to Theorem 2, the convergence condition did not hold after perturbation and the convergence can be guaranteed by shortening the prediction horizon. Also, according to Remark 1, the convergence condition can be made less sensitive to perturbation by introducing the regularization term γI with $\gamma > 0$. We compared the convergence factor $\rho((D^* + L)^{-1}U(D^* + U)^{-1}L)$ for the upper layer's SGS iteration along the closed-loop simulation under different prediction horizons ($T = 20$ and 100) and regularization parameters ($\gamma = 0$ and 0.5) in Fig. 8. It can be seen that the convergence condition $\rho((D^* + L)^{-1}U(D^* + U)^{-1}L) < 1$ for SGS was satisfied with either regularization or a short prediction horizon. Recall that the conditions provided by Theorem 2 are sufficient conditions. Aside from the prediction horizons in the numerical example (20 and 100 s), the proposed method still converged

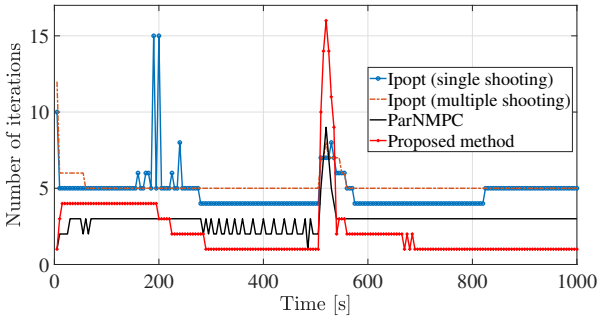


Figure 6 Time histories of numbers of iterations

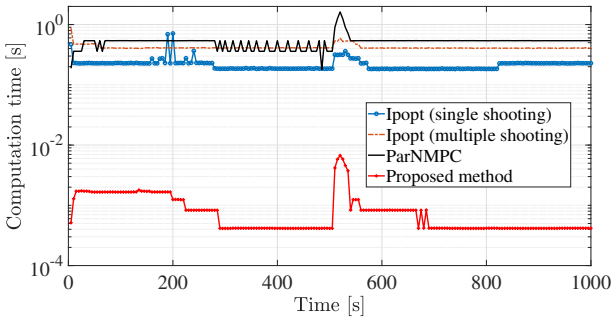


Figure 7 Time histories of computation times per time step

at every time step during simulation for a prediction horizon of 1000 s ($\gamma = 0.5$), which is sufficiently long for the heat transfer example.

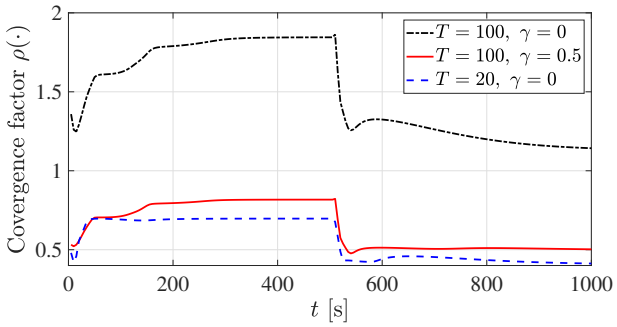


Figure 8 Time histories of convergence factors of SGS under different settings

6 | CONCLUSION

This paper presents a double-layer Jacobi method for the NMPC control of PDE systems. The NMPC problem is formulated on the basis of the spatially and temporally discretized PDE system and then relaxed. The proposed method performs simple Jacobi-type iterations to solve the KKT conditions and the underlying linear systems to make full use of the sparsities exist in both the spatial and temporal directions. Furthermore, the convergence of the proposed method can be guaranteed by adjusting the prediction horizon and regularization parameter. The results of the numerical experiment show that the proposed method can significantly reduce the computation time.

Future research directions include extending the proposed method to the NMPC control of other large-scale systems and applying the finite element method to discretize the PDE system.

□

APPENDIX

A PROOF OF LEMMA 4

Proof. If $N = 1$, the upper-layer Jacobi method is exactly identical to Newton's method. The result $\rho((\bar{D}^*)^{-1}(L+U)) = 0$ can be easily obtained from $L = U = 0$. We discuss the case of $N \geq 2$. The proof is done by showing that the eigenvalues of $(\bar{D}^*)^{-1}(L+U)$ are all zero. In fact, for a variable $\sigma \in \mathbb{C}$, the expression $\det(\sigma I - (\bar{D}^*)^{-1}(L+U)) = \sigma^{N(2n_x+n_u)}$ is obtained by using the Schur complement recursively as shown below.

Define a set of matrices

$$\mathbb{A} := \left\{ \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ P & 0 & 0 \end{bmatrix} \in \mathbb{C}^{(2n_x+n_u)\times(2n_x+n_u)},\ P \in \mathbb{C}^{n_x \times n_x} \right\}.$$

Define the following shorthand:

$$\bar{D}_i^L := -(\bar{D}_i^*)^{-1} M_L \text{ and } \bar{D}_i^U := -(\bar{D}_i^*)^{-1} M_U$$

so that

$$(\bar{D}^*)^{-1}(L+U) = -\begin{bmatrix} 0 & \bar{D}_1^U & & & \\ \bar{D}_2^L & 0 & \bar{D}_2^U & & \\ & \bar{D}_3^L & \ddots & \ddots & \\ & & \ddots & 0 & \bar{D}_{N-1}^U \\ & & & \bar{D}_N^L & 0 \end{bmatrix}.$$

For any $A \in \mathbb{A}$ and $i \in \{2, \cdots, N\}$, it can be examined that

$$\bar{D}_{i-1}^U (\sigma I - A)^{-1} \bar{D}_i^L \in \mathbb{A}. \tag{A1}$$

Let $K \in \{2, \cdots, N\}$ and $A_K \in \mathbb{A}$. We define a K-size (K blocks of rows and columns) block-tridiagonal matrix W_K by

$$\left[\begin{array}{cccc|c} \sigma I & \bar{D}_1^U & & & \\ \bar{D}_2^L & \sigma I & \bar{D}_2^U & & \\ & \bar{D}_3^L & \ddots & \ddots & \\ & & \ddots & \sigma I & \bar{D}_{K-1}^U \\ \hline & & & \bar{D}_K^L & \sigma I - A_K \end{array}\right] =: \begin{bmatrix} M_A & M_B \\ M_C & M_D \end{bmatrix}. \tag{A2}$$

The determinant of W_K is given by

$$\det W_K = \det(\sigma I - A_K) \det(\mathrm{schur}(W_K, \sigma I - A_K)), \tag{A3}$$

where $\mathrm{schur}(W_K, \sigma I - A_K)$ denotes the Schur complement of the block $\sigma I - A_K$ of W_K, i.e.,

$$\mathrm{schur}(W_K, \sigma I - A_K) = M_A - M_B M_D^{-1} M_C.$$

Let us then calculate the right hand side of (A3). It can be shown that

$$\det(\sigma I - A_K) = \sigma^{2n_x+n_u}. \tag{A4}$$

Since $A_K \in \mathbb{A}$, we know from (A1) that the only nonzero block (lower right corner) of $M_B M_D^{-1} M_C$ belongs to $\mathbb{A}$, i.e,

$$\bar{D}_{K-1}^{U}(\sigma I - A_K)^{-1}\bar{D}_K^{L} \in \mathbb{A}. \tag{A5}$$

By choosing A_{K-1} to be the left hand side of (A5), the Schur complement $\mathrm{schur}(W_K, \sigma I - A_K)$ can be seen as a $(K-1)$-size block-tridiagonal matrix in the form of (A2), i.e.,

$$\mathrm{schur}(W_K, \sigma I - A_K) =: W_{K-1}. \tag{A6}$$

By substituting (A4) and (A6) into (A3), we obtain the following recursion:

$$\det W_K = \sigma^{2n_x+n_u} \det W_{K-1}.$$

Following the procedures above and together with $W_1 = \sigma I_{2n_x+n_u}$, we obtain

$$\det W_K = \sigma^{K(2n_x+n_u)},$$

which holds for any $K \in \{2, \cdots, N\}$ and $A_K \in \mathbb{A}$. Then, by choosing $K = N$ and $A_N = 0 \in \mathbb{A}$, we have

$$\det W_N = \det(\sigma I - (\bar{D}^*)^{-1}(L+U)) = \sigma^{N(2n_x+n_u)}. \tag{A7}$$

From (A7), we know that $(\bar{D}^*)^{-1}(L+U)$ has only zero eigenvalues. The conclusion $\rho((\bar{D}^*)^{-1}(L+U)) = 0$ then follows. □

References

1. Ohtsuka T, Kodama A. Automatic code generation system for nonlinear receding horizon control. *Transactions of the Society of Instrument and Control Engineers* 2002; 38(7): 617–623.
2. Englert T, Völz A, Mesmer F, Rhein S, Graichen K. A software framework for embedded nonlinear model predictive control using a gradient-based augmented Lagrangian approach (GRAMPC). *Optimization and Engineering* 2019; 20(3): 769–809.
3. Deng H, Ohtsuka T. A parallel Newton-type method for nonlinear model predictive control. *Automatica* 2019; 109: 108560.
4. Hashimoto T, Yoshioka Y, Ohtsuka T. Receding horizon control with numerical solution for nonlinear parabolic partial differential equations. *IEEE Transactions on Automatic Control* 2013; 58(3): 725–730.
5. Ohtsuka T. A continuation/GMRES method for fast computation of nonlinear receding horizon control. *Automatica* 2004; 40(4): 563–574.
6. Ou Y, Schuster E. Model predictive control of parabolic PDE systems with dirichlet boundary conditions via Galerkin model reduction. In: Proceedings of the 2009 American Control Conference. ; 2009; St. Louis, USA: 1–7.
7. Sirovich L. Turbulence and the dynamics of coherent structures. Part I: Coherent structures. *Quarterly of Applied Mathematics* 1987; 45(3): 561–571.
8. Steinbach MC. A structured interior point SQP method for nonlinear optimal control problems. In: Birkhäuser Basel. 1994 (pp. 213–222).
9. Zanelli A, Domahidi A, Jerez J, Morari M. FORCES NLP: an efficient implementation of interior-point methods for multistage nonlinear nonconvex programs. *International Journal of Control* 2020; 93(1): 13–29.
10. Andersson JA, Frasch JV, Vukov M, Diehl M. A condensing algorithm for nonlinear MPC with a quadratic runtime in horizon length. *Optimization Online* 2017.
11. Saad Y. *Iterative Methods for Sparse Linear Systems, Second Edition*. SIAM . 2003.
12. Nocedal J, Wright SJ. *Numerical Optimization, Second Edition*. Springer Science and Business Media . 2006.
13. Ortega JM, Rheinboldt WC. *Iterative Solution of Nonlinear Equations in Several Variables*. Academic Press . 1970.
14. Rawlings J, Mayne D. *Model Predictive Control: Theory and Design*. Nob Hill Publishing, Madison, WI . 1999.

15. Hageman LA, Young DM. *Applied Iterative Methods*. Academic Press . 1981.

16. Zavala VM. New architectures for hierarchical predictive control. *IFAC-PapersOnLine* 2016; 49(7): 43–48.

17. Mathworks . Nonlinear Heat Transfer in Thin Plate. 2020.

18. Bleris LG, Garcia J, Kothare MV, Arnold MG. Towards embedded model predictive control for system-on-a-chip applications. *Journal of Process Control* 2006; 16(3): 255–264.

19. Bock HG. Recent advances in parameter identification techniques for ODE. In: Deuflhard P, Hairer E., eds. *Numerical Treatment of Inverse Problems in Differential and Integral Equations*. 2 of *Progress in Scientific Computing*. Birkhäuser Boston. 1983.

20. Andersson JA, Gillis J, Horn G, Rawlings JB, Diehl M. CasADi – A software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation* 2019; 11(1): 1–36. doi: 10.1007/s12532-018-0139-4

21. Wächter A, Biegler LT. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming* 2006; 106(1): 25–57.

22. Amestoy P, Duff IS, Koster J, L'Excellent JY. A Fully Asynchronous Multifrontal Solver Using Distributed Dynamic Scheduling. *SIAM Journal on Matrix Analysis and Applications* 2001; 23(1): 15-41.

23. Deng H, Ohtsuka T. A parallel code generation toolkit for nonlinear model predictive control. In: Proceedings of the 57th IEEE Conference on Decision and Control. ; 2018; Miami, USA: 4920–4926.