

State-Defect Constraint Pairing Graph Coarsening Method for Karush-Kuhn-Tucker Matrices Arising in Orthogonal Collocation Methods for Optimal Control

Begüm Şenses Cannataro ^{*}
Anil V. Rao [†]
University of Florida
Gainesville, FL 32611-6250

Timothy A. Davis [‡]
Texas A&M University
College Station, TX 77843

Abstract

A state-defect constraint pairing graph coarsening method is described for improving computational efficiency during the numerical factorization of large sparse Karush-Kuhn-Tucker matrices that arise from the discretization of optimal control problems via an Legendre-Gauss-Radau orthogonal collocation method. The method takes advantage of the particular sparse structure of the Karush-Kuhn-Tucker matrix that arises from the orthogonal collocation method. The state-defect constraint pairing graph coarsening method pairs each component of the state with its corresponding defect constraint and forces paired rows to be adjacent in the reordered Karush-Kuhn-Tucker matrix. Aggregate state-defect constraint pairing results are presented using a wide variety of benchmark optimal control problems where it is found that the proposed state-defect constraint pairing graph coarsening method significantly reduces both the number of delayed pivots and the number of floating point operations and increases the computational efficiency by performing more floating point operations per unit time. It is then shown that the state-defect constraint pairing graph coarsening method is less effective on Karush-Kuhn-Tucker matrices arising from Legendre-Gauss-Radau collocation when the optimal control problem contains state and control equality path constraints because such matrices may have delayed pivots that correspond to both defect and path constraints. An alternate graph coarsening method that employs maximal matching is then used to attempt to further reduce the number of delayed pivots. It is found, however, that this alternate graph coarsening method provides no further advantage over the state-defect constraint pairing graph coarsening method.

1 Introduction

Over the past two decades, direct collocation methods have become popular in the numerical solution of nonlinear optimal control problems. In a direct collocation method, the state and control are

^{*}PhD Candidate, Department of Mechanical and Aerospace Engineering. Email: bgmsenses@ufl.edu.

[†]Associate Professor, Department of Mechanical and Aerospace Engineering. E-mail: anilvr Rao@ufl.edu. Corresponding Author.

[‡]Professor, Department of Computer Science & Engineering Information & Science Engineering. E-mail: davis@tamu.edu.

discretized at a set of appropriately chosen points in the interval of interest. The continuous-time optimal control problem is then transcribed to a large sparse finite-dimensional nonlinear programming problem (NLP) and the NLP is solved using a well known NLP solver.^{1,2} While all NLPs arising from direct collocation are sparse, every discretization method has a unique sparse structure that can differ significantly from other discretization methods. In particular, this sparse structure manifests itself in a sparse symmetric indefinite Karush-Kuhn-Tucker (KKT) linear system that needs to be solved repeatedly as the NLP is being solved. Because the largest portion of time required to solve the NLP is the time required to solve the KKT system, it is essential that the KKT system be solved as efficiently as possible to reduce the computation time required to solve the NLP. Consequently, a great deal of work has been done on the development of robust and efficient sparse symmetric indefinite linear solvers and has led to well-known solvers including MA47,³ MA57,⁴ Mumps,⁵ and Pardiso.⁶

In recent years a particular class of direct collocation methods that has received a great deal of attention is the class of *Gaussian quadrature orthogonal collocation* methods.^{7–25} In a Gaussian quadrature collocation method, the time domain is divided into a mesh and the state is approximated in each mesh interval using Lagrange polynomials where the support points of the Lagrange polynomials are chosen to be points associated with a Gaussian quadrature and a noncollocated end point. The order of the approximation in each interval can then be different from the order in every other interval, thus leading to a *variable-order* method. As it turns out, Gaussian quadrature orthogonal collocation methods have a great deal of structure that can be exploited to improve the computational efficiency with which the NLP is solved. A key aspect of the structure of a Gaussian quadrature method is the structure of the KKT matrix itself. In this paper the previously developed Legendre-Gauss-Radau (LGR) orthogonal collocation method^{19–22, 24, 25} is used as the basis for solving a continuous optimal control problem. While the aforementioned linear solvers take advantage of the more generic fact that all KKT matrices are symmetric, none of these linear solvers take advantage of the particular sparse structure of the KKT matrices that arise from LGR orthogonal collocation.

The objective of this research is to develop a method that improves the computational efficiency during the solution of KKT systems arising from the LGR collocation method. As it turns out, this objective can be achieved by performing a graph coarsening method that utilizes the unique sparsity pattern of the KKT systems that arise from LGR collocation. Because graph coarsening methods in the literature are performed as a step in graph partitioning, graph coarsening methods are discussed after a brief description of graph partitioning methods. It is noted that, with minor

modifications, an approach similar to state-defect constraint pairing graph coarsening can be applied to any collocation method that has a similar block structure (for example, the Legendre-Gauss¹¹ collocation method).

Graph partitioning methods compute a fill reducing ordering which leads to a high degree of concurrency in the factorization phase of parallel direct methods or decompose matrices into smaller systems of approximately the same size and solve these smaller systems in parallel. There are three approaches to graph partitioning:²⁶ spectral partitioning^{27,28} (where the eigenvectors of the graphs are used to find the partitioning), geometric partitioning^{29,30} (where the geometrical characterization of the graphs is used to find a small subset of vertices whose removal divides the rest of the graph into two disconnected pieces of approximately equal size), and multilevel (k -way) partitioning^{31–34} (where a sequence of successively smaller hypergraphs is used to find the partitioning). One of the steps of a multilevel (k -way) partitioning method is graph coarsening. Graph coarsening methods fall into two categories: algebraic multigrid (AMG)-based and matching-based coarsening methods. The AMG-based graph coarsening method is beyond the scope of this research. For a detailed explanation of AMG-based coarsening methods see Ref. 35. In matching-based graph coarsening coarser graphs are obtained by collapsing pair of vertices and their edges. The goal of the matching-based graph coarsening is obtaining the coarsest graph possible by matching (collapsing) as many vertices as possible. This goal can be achieved by performing a maximal matching algorithm. There are various maximal matching algorithms with different matching criteria. Random matching, heavy edge matching, and light edge matching are some examples of maximal matching algorithms.²⁶

While the solution of general large-sparse KKT systems has received a great deal of attention, significantly less attention has been paid to the development of tailored solution methods for KKT systems arising from LGR collocation. One particular study of KKT systems arising from LGR collocation is that found in Ref. 36 where a decomposition strategy is presented that exploits the structure of the KKT systems in order to compute the necessary linear algebra operations on multiple processors. In Ref. 36 it is shown that the proposed method has the potential for significant speedup for problems with few states and many algebraics. Although both this paper and Ref. 36 focus on increasing computational efficiency during the solution of KKT systems arising from LGR collocation, the method that is developed in this paper increases the computational efficiency by decreasing the number of delayed pivots, while the method that is developed in Ref. 36 increases computational efficiency by solving KKT systems in parallel.

In this paper a new state-defect constraint pairing graph coarsening method is described for im-

proving computational efficiency during the numerical factorization of large sparse KKT matrices that arise from the discretization of optimal control problems via LGR collocation. The purpose of this graph coarsening is to find a pivot pre-ordering that reduces the number of delayed pivots that occur during numerical factorization. In this new method, each state row is paired with its corresponding defect constraint row and paired rows are forced to be adjacent in the reordered KKT matrix. Aggregate state-defect constraint pairing results are presented using a wide variety of benchmark optimal control problems where it is found that the proposed state-defect constraint pairing method significantly reduces both the number of delayed pivots and the number of floating point operations and increases the computational efficiency by performing more floating point operations per unit time. Furthermore, it is found that the state-defect constraint pairing graph coarsening method is less effective when the continuous optimal control problem contains state and control equality path constraints because the corresponding KKT matrices of the LGR collocation method may have delayed pivots due to both defect and path constraints. In order to attempt further reduce the number of delayed pivots beyond that of the state-defect constraint pairing graph coarsening method, an alternate graph coarsening method is described. It is found, however, that this alternate graph coarsening method provides no further advantage over the state-defect constraint pairing graph coarsening method.

This paper is organized as follows. In Section 2 a general single-phase optimal control problem is described. In Section 3 this optimal control problem is transcribed to an NLP using the aforementioned LGR collocation method. In Section 4 the phases of a general purpose sparse linear solver are described and details are provided on the analysis and numerical factorization phases of an off-the-shelf sparse linear solver MA57.⁴ In Section 5 the properties of the KKT matrices arising from LGR collocation and general trends in delayed pivots that arise during the numerical factorization of KKT matrices arising from LGR collocation are discussed. In Section 6 the state-defect constraint pairing graph coarsening method is described. In Section 7 the benefits of the state-defect constraint graph coarsening method are demonstrated on a wide variety of benchmark optimal control problems. Additionally, in Section 8 a discussion is given on problems with state and control equality path constraints and an attempt is made to improve upon the state-defect constraint pairing graph coarsening method via an alternate graph coarsening method. Finally, in Section 9 conclusions are provided on the research.

2 Continuous Optimal Control Problem

Without loss of generality, consider the following one-phase continuous optimal control problem stated in a form similar to that given in Ref. 23 as follows. Determine the state, $\mathbf{y}(\tau) \in \mathbb{R}^{n_y}$, and the control, $\mathbf{u}(\tau) \in \mathbb{R}^{n_u}$, on $\tau \in [-1, +1]$ that minimize the cost functional

$$J = \phi(\mathbf{y}(-1), t_0, \mathbf{y}(+1), t_f, \mathbf{q}) \quad (1)$$

subject to dynamic constraints

$$\dot{\mathbf{y}}(t) - \frac{t_f - t_0}{2} \mathbf{a}(\mathbf{y}(\tau), \mathbf{u}(\tau)) = \mathbf{0}, \quad (2)$$

the algebraic path constraints

$$\mathbf{c}_{\min} \leq \mathbf{c}(\mathbf{y}(\tau), \mathbf{u}(\tau)) \leq \mathbf{c}_{\max}, \quad (3)$$

the integral constraints

$$q_i - \frac{t_f - t_0}{2} \int_{-1}^1 Q_i(\mathbf{y}(\tau), \mathbf{u}(\tau), t) d\tau = 0, \quad (i = 1, \dots, n_q), \quad (4)$$

and the event constraints

$$\mathbf{b}_{\min} \leq \mathbf{b}(\mathbf{y}(-1), t_0, \mathbf{y}(+1), t_f, \mathbf{q}) \leq \mathbf{b}_{\max}. \quad (5)$$

While Eqs. (1)–(5) describe a one-phase optimal control problem, the optimal control problem of Eqs. (1)–(5) can be extended to multiple phases with minor modifications (see Ref. 23 for details).

Suppose now that the interval $\tau \in [-1, +1]$ is divided into a *mesh* consisting of K *mesh intervals* $[T_{k-1}, T_k]$, $k = 1, \dots, K$, where $(T_0, \dots, T_K)$ are the *mesh points*. The mesh points have the property that $-1 = T_0 < T_1 < T_2 < \dots < T_K = T_f = +1$. Next, let $\mathbf{y}^{(k)}(\tau)$ and $\mathbf{u}^{(k)}(\tau)$ be the state and control in mesh interval k . The optimal control problem of Eqs. (1)–(5) can then be written as follows. First, the cost functional of Eq. (1) can be written as

$$J = \phi(\mathbf{y}^{(1)}(-1), t_0, \mathbf{y}^{(K)}(+1), t_f, \mathbf{q}), \quad (6)$$

Next, the dynamic constraints of Eq. (2) in mesh interval k can be written as

$$\frac{d\mathbf{y}^{(k)}(\tau^{(k)})}{d\tau^{(k)}} = \frac{t_f - t_0}{2} \mathbf{a}(\mathbf{y}^{(k)}(\tau^{(k)}), \mathbf{u}^{(k)}(\tau^{(k)}), \tau^{(k)}; t_0, t_f), \quad (k = 1, \dots, K). \quad (7)$$

Furthermore, the path constraints of Eq. (3) in interval k are given as

$$\mathbf{c}_{\min} \leq \mathbf{c}(\mathbf{y}^{(k)}(\tau^{(k)}), \mathbf{u}^{(k)}(\tau^{(k)}), \tau^{(k)}; t_0, t_f) \leq \mathbf{c}_{\max}, \quad (k = 1, \dots, K). \quad (8)$$

the integral constraints of Eq. (4) are given as

$$q_j - \frac{t_f - t_0}{2} \sum_{k=1}^K \int_{T_{k-1}}^{T_k} Q_j(\mathbf{y}^{(k)}(\tau^{(k)}), \mathbf{u}^{(k)}(\tau^{(k)}), \tau^{(k)}; t_0, t_f) d\tau = 0, \quad (j = 1, \dots, n_q, k = 1 \dots, K). \quad (9)$$

Finally, the event constraints of Eq. (5) are given as

$$\mathbf{b}_{\min} \leq \mathbf{b}(\mathbf{y}^{(1)}(-1), t_0, \mathbf{y}^{(K)}(+1), t_f, \mathbf{q}) \leq \mathbf{b}_{\max}, \quad (10)$$

where

$$\mathbf{q} = \begin{bmatrix} q_1 \\ \vdots \\ q_{n_q} \end{bmatrix}.$$

Because the state must be continuous at each interior mesh point, it is required that the condition $\mathbf{y}^{(k)}(T_k) = \mathbf{y}^{(k+1)}(T_k)$ is satisfied at the interior mesh points $(T_1, \dots, T_{K-1})$.

3 Legendre-Gauss-Radau Orthogonal Collocation Method

In this section, the continuous optimal control problem given in Section 2 is approximated by a finite-dimensional nonlinear programming problem (NLP) using the previously developed Legendre-Gauss-Radau (LGR) orthogonal collocation method.^{19-22,24,25} In the LGR collocation method the state of continuous optimal control problem is approximated in each mesh interval $k \in [1, \dots, K]$ as

$$\mathbf{y}^{(k)}(\tau) \approx \mathbf{Y}^{(k)}(\tau) = \sum_{j=1}^{N_k+1} \mathbf{Y}_j L_j(\tau), \quad L_j^{(k)}(\tau) = \prod_{l=1, l \neq j}^{N_k+1} \frac{\tau - \tau_l^{(k)}}{\tau_j^{(k)} - \tau_l^{(k)}}, \quad (11)$$

where $L_j^{(k)}(\tau)$, $(j = 1, \dots, N_k+1)$ is a basis of Lagrange polynomials, N_k is the number of Legendre-Gauss-Radau collocation points in mesh interval k , $N_k + 1$ is a noncollocated point. The derivative of the state approximation is then given as

$$\frac{d\mathbf{Y}^{(k)}(\tau)}{d\tau} = \sum_{j=1}^{N_k+1} \mathbf{Y}_j^{(k)} \frac{dL_j^{(k)}(\tau)}{d\tau}. \quad (12)$$

Furthermore, the cost functional of Eq. (6) is approximated by the following cost function:

$$J \approx \mathcal{J} = \phi(\mathbf{Y}_1^{(1)}, t_0, \mathbf{Y}_{N_K+1}^{(K)}, t_f, \mathbf{q}), \quad (13)$$

where $\mathbf{Y}_1^{(1)}$ is the approximation of $\mathbf{y}(T_0 = -1)$, and $\mathbf{Y}_{N_K+1}^{(K)}$ is the approximation of $\mathbf{y}(T_K = +1)$. Next, the dynamic constraints of Eq. (7) are approximated by the following algebraic *defect*

constraints that are enforced at the N_k LGR points in mesh interval k ,

$$\Delta_i^{(k)} = \sum_{j=1}^{N_k+1} D_{ij}^{(k)} \mathbf{Y}_j^{(k)} - \frac{t_f - t_0}{2} \mathbf{a}(\mathbf{Y}_i^{(k)}, \mathbf{U}_i^{(k)}, \tau_i^{(k)}; t_0, t_f) = \mathbf{0}, \quad (i = 1, \dots, N_k), \quad (14)$$

where

$$D_{ij}^{(k)} = \left[\frac{dL_k^{(k)}(\tau)}{d\tau} \right]_{\tau_i^{(k)}}, \quad (i = 1, \dots, N_k, j = 1, \dots, N_k + 1) \quad (15)$$

is the $N_k \times (N_k + 1)$ Legendre-Gauss-Radau differentiation matrix in mesh interval $k \in [1, \dots, K]$.

The structure of the LGR differentiation matrix $\mathbf{D}$ is shown in Fig. 1. Furthermore, the path

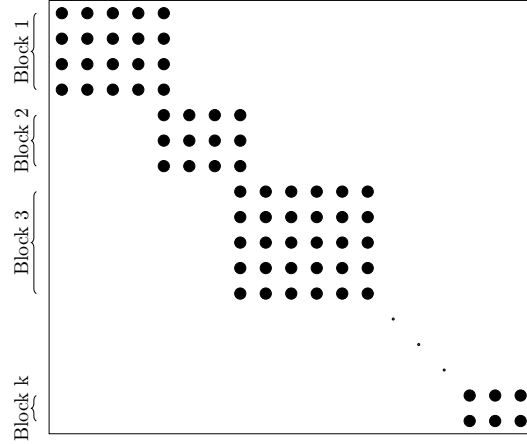


Figure 1: Legendre-Gauss-Radau differentiation matrix.

constraints given in Eq. (8) are approximated as

$$\mathbf{c}_{\min} \leq \mathbf{c}(\mathbf{Y}_i^{(k)}, \mathbf{U}_i^{(k)}, \tau_i^{(k)}; t_0, t_f) \leq \mathbf{c}_{\max}, \quad (i = 1, \dots, N_k). \quad (16)$$

The integral constraints of Eq. (9) are then approximated as

$$q_j - \sum_{k=1}^K \sum_{i=1}^{N_k} \frac{t_f - t_0}{2} w_i^{(k)} Q_j(\mathbf{Y}_i^{(k)}, \mathbf{U}_i^{(k)}, \tau_i^{(k)}; t_0, t_f) = 0, \quad (i = 1, \dots, N_k, j = 1, \dots, n_q) \quad (17)$$

where $w_i^{(k)}$ is the weight that corresponds to i^{th} collocation point in the k^{th} mesh interval. Finally, the event constraints of Eq. (10) are approximated as

$$\mathbf{b}_{\min} \leq \mathbf{b}(\mathbf{Y}_1^{(1)}, t_0, \mathbf{Y}_{N_k+1}, t_f, \mathbf{q}) \leq \mathbf{b}_{\max}, \quad (18)$$

where continuity in the state at the interior mesh points is enforced with the constraint $\mathbf{Y}_{N_k+1}^{(k)} = \mathbf{Y}_1^{(k+1)}$.

The cost function given in Eq. (13) together with the algebraic constraints of Eqs. (14), (16), (17), and (18) form a nonlinear programming problem (NLP) of the following form. Determine the

decision vector $\mathbf{x} \in \mathbb{R}^n$ that minimizes the cost function

$$f(\mathbf{x}) \tag{19}$$

subject to the algebraic constraints

$$\begin{aligned} \mathbf{x}_{\min} &\leq \mathbf{x} \leq \mathbf{x}_{\max}, \\ \mathbf{g}_{\min} &\leq \mathbf{g}(\mathbf{x}) \leq \mathbf{g}_{\max}. \end{aligned} \tag{20}$$

The constraint Jacobian and Lagrangian Hessian of the NLP given in Eqs. (19) and (20) are defined, respectively, as

$$\mathbf{J} = \frac{\partial \mathbf{g}(\mathbf{x})}{\partial \mathbf{x}}, \quad \mathbf{H} = \frac{\partial^2 \mathcal{L}}{\partial \mathbf{x}^2}, \tag{21}$$

where $\mathcal{L} = \sigma f(\mathbf{x}) + \boldsymbol{\lambda}^\top \mathbf{g}(\mathbf{x})$ is the Lagrangian, $\sigma \in \mathbb{R}$ and $\boldsymbol{\lambda} \in \mathbb{R}^m$ are the Lagrange multipliers associated with the cost function and the constraints, respectively. The sparsity pattern of the constraint Jacobian and the Lagrangian Hessian that arise from the NLP associated with the LGR collocation method are shown in Figs. 2 and 3, respectively. This sparse structure is obtained by mapping the derivatives of the optimal control problem to the first and second derivatives of the NLP functions as described in Ref. 37.

Throughout this study, optimal control problems are transcribed into an NLP by a general purpose MATLAB software GPOPS-II.²³ The NLP can then be solved using a wide variety of well established NLP solvers (for example, SNOPT¹ or IPOPT²). While all Newton-based NLP solvers solve a large KKT system at each NLP iteration, the focus of this paper is on solving a KKT system arising from a full-Newton NLP solver where the constraint Jacobian and the Lagrangian Hessian are sparse. The KKT matrix at the iteration k of an NLP is given as follows:

$$\mathbf{K} = \begin{bmatrix} \mathbf{H}^{(k)} & [\mathbf{J}^{(k)}]^\top \\ \mathbf{J}^{(k)} & \mathbf{0} \end{bmatrix} \tag{22}$$

where $\mathbf{J}^{(k)}$ and $\mathbf{H}^{(k)}$ are the constraint Jacobian and the Lagrangian Hessian at iteration k , respectively. In order to guarantee a descent direction at every NLP iteration, $\mathbf{H}^{(k)}$ must be positive definite on the null space of the constraint Jacobian and the constraint Jacobian has to be full rank.³⁸ A KKT matrix that satisfies these conditions must have exactly as many positive eigenvalues as the number of variables and as many negative eigenvalues as the constraints. Fortunately, it is not necessary to find eigenvalues of the KKT matrix at every iteration; instead, only the inertia³⁹ of the KKT matrix needs to be monitored. If the inertia of the KKT matrix is not correct in an iteration, the KKT matrix is modified using a Levenberg parameter.⁴⁰ The modified KKT matrix

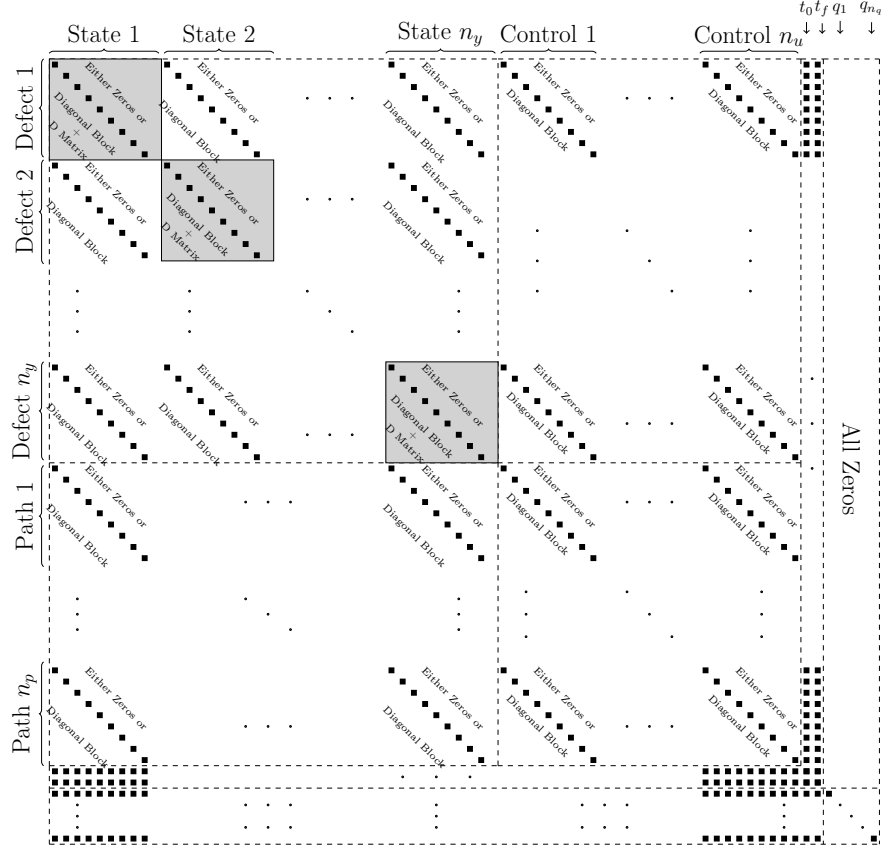


Figure 2: One-phase differential LGR collocation NLP constraint Jacobian.

is given as follows:

$$\tilde{\mathbf{K}} = \begin{bmatrix} \mathbf{H}^{(k)} + \tau \mathbf{I} & [\mathbf{J}^{(k)}]^\top \\ \mathbf{J}^{(k)} & \mathbf{0} \end{bmatrix}, \quad (23)$$

where $\mathbf{I}$ is the identity matrix of proper size and τ is the Levenberg parameter that guarantees a positive definite reduced modified Lagrangian Hessian, $\mathbf{H}^{(k)} + \tau \mathbf{I}$. The only necessary condition for the successful state-defect constraint pairing graph coarsening method is a Lagrangian Hessian with nonzero diagonals. This condition is always satisfied for all second-order Newton-based NLP methods. Without loss of generality, the NLPs are solved using the open-source NLP solver IPOPT,² where IPOPT uses an interior-point method with a filter-based line search (details found Ref. 41), in full Newton (second derivative) mode. At every iteration of IPOPT a large sparse symmetric indefinite KKT system is solved using the sparse symmetric indefinite linear solver MA57⁴ which is the default linear solver for IPOPT.

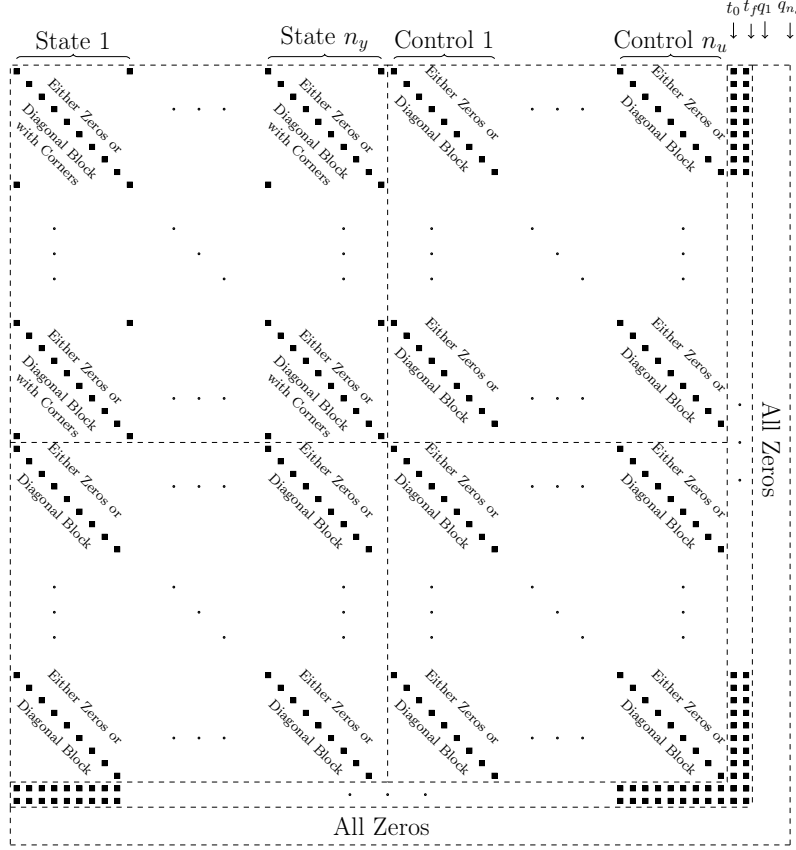


Figure 3: One-phase differential LGR collocation NLP Lagrangian Hessian.

4 Solution of Sparse Symmetric Indefinite Linear Systems

In this section, the phases of a general-purpose sparse linear solver are given and the multifrontal methods are described. The first phase of a general-purpose sparse linear solver is the *analysis phase*. In this phase, the linear solver permutes the given sparse matrix using a fill-reducing algorithm and determines the elimination tree structure of the permuted sparse matrix. Moreover, it postorders the elimination tree to reduce the working storage. Finally, it forecasts the storage needs for the second phase called *numerical factorization phase*. In the numerical factorization phase, the linear solver factorizes the permuted matrix as follows:

$$\mathbf{PAP}^\top = \mathbf{LDL}^\top, \quad (24)$$

where $\mathbf{P}$ is a permutation matrix and $\mathbf{L}$ is a lower triangular matrix. The last phase of a general-purpose sparse linear solver is the solution phase. In this phase the linear solver solves the factorized sparse matrix using forward and/or backward substitutions. As stated earlier, in this research, sparse symmetric indefinite linear systems are solved using the multifrontal linear solver MA57.⁴ MA57 was chosen for use in this research because it is widely-used (for example, it is the built-

in solver in MATLAB for the backslash operation $\mathbf{x} = \mathbf{A} \backslash \mathbf{b}$ when $\mathbf{A}$ is symmetric, sparse, and indefinite). While a detailed description of the methods used in MA57 is beyond the scope of this paper (see Refs. 42–44), the MA57 analysis and numerical factorization phases are briefly described. In addition to the analysis phase steps of a general-purpose sparse linear solver, MA57 finds frontal matrices using the number of nonzeros in the $\mathbf{L}$ matrix given in Eq. (24) together with the structure of the elimination tree found in the earlier stages of the analysis phase. The basic idea behind the numerical factorization phase of MA57 is Gaussian elimination, but MA57 does not perform Gaussian elimination on a large sparse matrix at once. Instead, MA57 performs Gaussian elimination on small dense frontal matrices. A frontal matrix, $\mathbf{F}$, is given as follows:

$$\mathbf{F} = \begin{bmatrix} \mathbf{F}_{11} & \mathbf{F}_{12} \\ \mathbf{F}_{12}^\top & \mathbf{F}_{22} \end{bmatrix}, \quad (25)$$

where the rows and the columns of $\mathbf{F}_{11}$ block correspond to the fully summed entries⁴ where an entry is called fully summed when contributions on this entry from every pivot is summed.

The main difference between the solution methods of definite and indefinite systems occurs in the numerical factorization phase. If the linear system is symmetric indefinite, a pivoting strategy that preserves the symmetry and prevents the element growth caused by the propagation of the round-off errors has to be performed during the numerical factorization phase. MA57 examines the numerical stability of pivots in a frontal matrix using the following pivot threshold test. The diagonal entry of the row k , a_{kk} , is accepted as the 1×1 pivot if

$$|a_{kk}^{-1}| \alpha_k \leq u^{-1} \quad (26)$$

is satisfied or $\mathbf{B}_{kk}$ is accepted as the 2×2 pivot if

$$|\mathbf{B}_{kk}^{-1}| \begin{bmatrix} \beta_k \\ \beta_{k+i} \end{bmatrix} \leq \begin{bmatrix} u^{-1} \\ u^{-1} \end{bmatrix}, \quad i > 1 \quad (27)$$

is satisfied ($0 < u < 0.5$). The quantities α_k , β_k and $\mathbf{B}_{kk}$ are given as follows:

$$\alpha_k = \max_{j \geq k+1} |a_{kj}|, \quad (28)$$

$$\beta_k = \max_{j \geq k+2} |a_{kj}| \quad (29)$$

and

$$\mathbf{B}_{kk} = \begin{bmatrix} a_{kk} & a_{k,k+i} \\ a_{k+i,k} & a_{k+i,k+i} \end{bmatrix}, \quad i > 1. \quad (30)$$

MA57 provides accurate 2×2 pivoting, with arbitrary interchanges to improve numerical stability. As a result, MA57 can handle ill-conditioned matrices that other solvers cannot handle (for example, PARDISO (in the Intel library) only allows for static 2×2 pivoting, which means that it can fail for very ill-conditioned matrices).

A pivot that does not satisfy the 1×1 or the 2×2 pivot criteria is called a *delayed pivot*. A delayed pivot is passed to its father node to be eliminated later in the factorization phase and the pivot search moves on to the next diagonal. Delayed pivots increase the fill-in (number of nonzeros in $\mathbf{L}$), the storage needs of the factorization phase, and number of floating point operations. When the MA57 analysis phase predicts the storage needs of the numerical factorization phase, it does not take into account the delayed pivots. Therefore, if there is a large number of delayed pivots, MA57 may fail due to insufficient memory allocation. In that case, MA57 stops the numerical factorization phase and reallocates a larger memory. Additionally, MA57 analysis phase forecasts

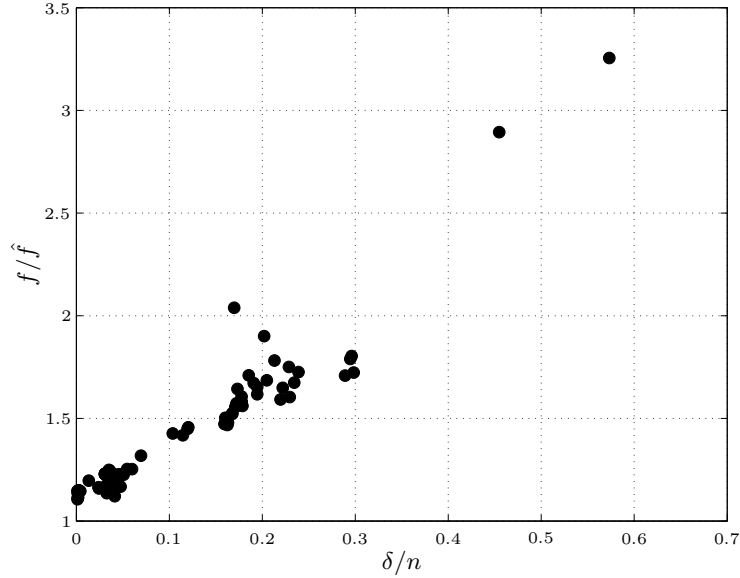


Figure 4: Ratio of the number of floating point operations in MA57 numerical factorization phase to forecast of the number of floating point operations that is obtained in MA57 symbolic analysis phase, $f/\hat{f}$, vs. ratio of the number of delayed pivots to size of matrices, δ/n .

the number of floating point operations that are required in numerical factorization phase, $\hat{f}$, without taking into account the delayed pivots that may occur during the numerical factorization phase. Figure 4 shows the ratio of actual number of floating point operations that are performed during the numerical factorization phase, f to $\hat{f}$ vs. the ratio of the number of delayed pivots, δ , to the size of the Karush-Kuhn-Tucker matrices that are solved, n . As seen in Fig. 4 when δ/n increases, $f/\hat{f}$ increases which shows that delayed pivots increase the number of floating point operations. Because

delayed pivots may cause insufficient memory allocation and increase the number of floating point operations this research focuses on developing a method that is tailored to decrease the number of delayed pivots during the numerical factorization of KKT matrices arising from LGR collocation.

5 KKT Matrices Arising from LGR Collocation

In this section, properties of KKT matrices arising from LGR collocation are explained. Then, general trends in delayed pivots that occur during the numerical factorization of these KKT matrices are given and possible reasons for these general trends are discussed. Finally, a way to decrease the number of delayed pivots is proposed.

5.1 Properties of LGR Collocation KKT Matrices

The KKT matrices arising from LGR collocation have a known sparse structure based on the Lagrangian Hessian and the constraint Jacobian as described in Section 3. Figure 5 shows an example of the sparsity pattern of a KKT matrix arising from LGR collocation. As seen in Fig. 5, the

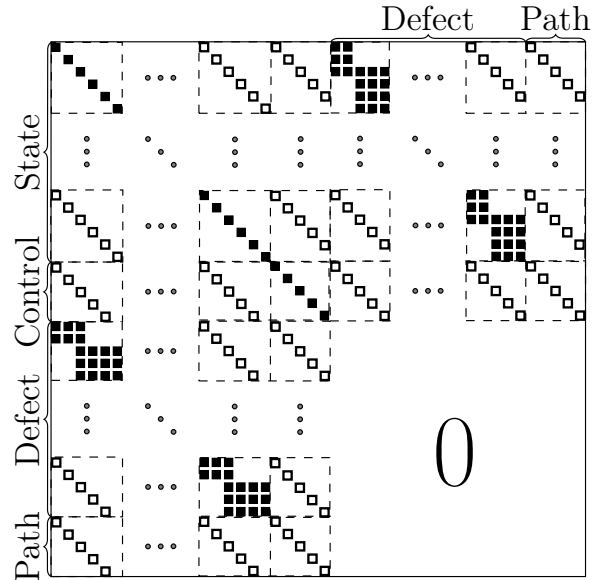


Figure 5: Sparsity pattern of KKT matrices arising from LGR collocation.

Lagrangian Hessian and the off-diagonal part of the constraint Jacobian consist of purely diagonal blocks. In addition, due to discretization separability,⁴⁵ computing the constraint Jacobian of the NLP can be reduced to computing the first derivatives of the continuous optimal control problem constraints with respect to the continuous optimal control problem variables, while computing the Lagrangian Hessian of the NLP can be reduced to computing the second derivatives of the

continuous optimal control problem functions with respect to the continuous optimal control problem variables.³⁷ As a result of discretization separability each optimal control problem derivative that is equal to zero corresponds to a zero block in the KKT matrix. Although, Fig. 5 provides an overestimate of the actual sparsity pattern of a KKT matrix, dense blocks in $\mathbf{J}$ and $\mathbf{J}^\top$ and the main diagonal blocks in $\mathbf{H}$ [see Eq. (22)], which are represented by black squares in Fig. 5, always consist of nonzeros in the KKT matrices because of the nonzero differentiation matrix arising from the differential form of LGR collocation and the Levenberg parameter that guarantees a positive definite $\mathbf{H}$, respectively. In addition, because KKT matrices arising from LGR collocation method has discretization separability, throughout this paper the Lagrangian Hessian rows are referred as the continuous optimal control problem variable rows (such as state or control rows) and the constraint Jacobian rows are referred as the continuous optimal control problem constraint rows (such as defect constraint or path constraint rows).

5.2 Properties of Delayed Pivots in LGR Collocation

In this paper, KKT systems of various sizes are solved where the KKT systems arise from the NLP associated with LGR collocation of optimal control problems that are given in Refs. 45–51. The following three general trends are observed in most of the delayed pivots that occur during the numerical factorization of these KKT matrices: (1) most of the delayed pivots correspond to the defect constraint rows; (2) most of the delayed defect constraint pivots reside in a frontal matrix with one fully summed row and column; and (3) most of the delayed defect constraints satisfy the MA57 pivot threshold test when they reside in the same frontal matrix as their corresponding state. Each of these trends is described in more detail below.

Most Delayed Pivots Correspond to Defect Constraint Rows. As shown in Fig. 6, the ratio of the delayed pivots that correspond to the defect constraints, δ^d , to the total number of delayed pivots, δ , is larger than 0.5 for almost all of the continuous optimal control problems studied. As mentioned before, delayed pivots do not satisfy either the 1×1 or the 2×2 pivot criterion. Because the defect constraint rows always have zero diagonals, these rows do not satisfy the 1×1 pivot criterion given in Eq. (26) unless the fill-in caused by the state and/or control rows is large enough. Although not every nonzero diagonal is guaranteed to satisfy the 1×1 pivot criterion, the defect constraint rows are more likely to fail the 1×1 pivot criterion than the state and control rows. Even if a defect constraint row does not satisfy the 1×1 pivot criterion, it may form a 2×2 pivot with the state or control rows. The second general trend in delayed pivots explains why most of the defect constraint rows do not satisfy the 2×2 pivot criterion.

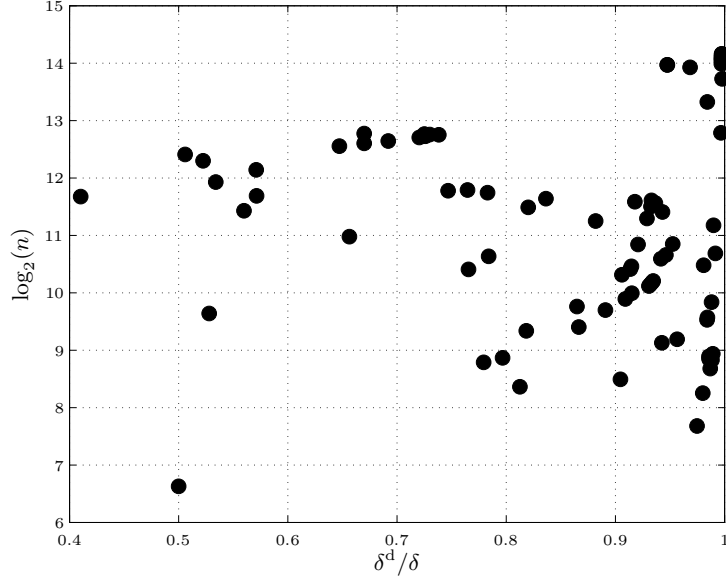


Figure 6: Base two logarithm of size of KKT matrices, $\log_2(n)$, vs. ratio of the delayed defect constraint pivots to the total number of delayed pivots, δ^d/δ .

Most Delayed Pivots Reside in a Frontal Matrix with One Fully Summed Row and Column. Figure 7 shows base two logarithm of size of KKT matrices, $\log_2(n)$, vs. the ratio of delayed defect constraint pivots that belong to a frontal matrix with a single fully summed row and column, $\delta^d(\text{single})/\delta^d$. As seen in Fig. 7, the ratio $\delta^d(\text{single})/\delta^d$ is larger than 0.5 for most of the continuous optimal control problems. For a defect constraint row to be able form a 2×2 pivot, a frontal matrix must contain at least two pivots. Therefore a defect constraint row that belongs to a frontal matrix with a single fully summed row and column cannot form a 2×2 pivot.

Most Delayed Defect Constraints Satisfy MA57 Pivot Threshold Test When Residing in Same Frontal Matrix as Corresponding State. Figure 8 compares the ratios of the state-defect constraint pairs that are in the same frontal matrix during the symbolic factorization, $\hat{\mu}$, and numerical factorization, μ , to the total number of state-defect constraint pairs, μ_{total} . Figure 8(a) shows that $\hat{\mu}/\mu_{\text{total}}$ changes from 0 to 0.3. On the other hand, Figs. 8(b) shows that μ/μ_{total} is much larger than $\hat{\mu}/\mu_{\text{total}}$ which means that if the state and its corresponding defect constraint are not in the same frontal matrix during the symbolic factorization, defect constraints get delayed and ultimately satisfy the MA57 pivot threshold test when they are in the same frontal matrix as their corresponding state.

The observations on delayed pivots lead to the conclusion that forcing a state or a control pivot to be in the same frontal matrix with a defect constraint may increase the possibility of this defect constraint satisfying the 1×1 or the 2×2 pivot criterion. The state is a better pair for defect constraints as compared with the control because the number of defect constraint rows are equal to

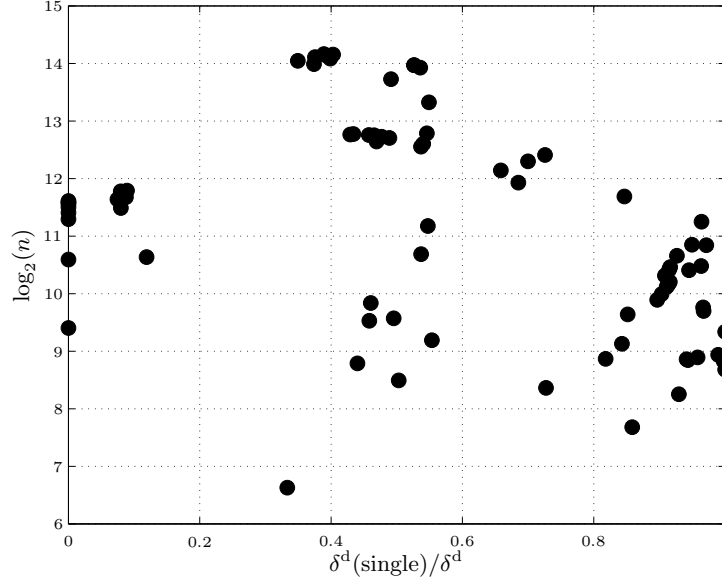


Figure 7: Base two logarithm of size of the KKT matrices, $\log_2(n)$, vs. ratio of delayed defect constraint pivots that belong to a frontal matrix with a single fully summed row and column to the delayed defect constraint pivots, $\delta^d(\text{single})/\delta^d$.

the number of state rows so there is a state row pair for each defect constraint row. On the other hand, the number of control rows is not necessarily equal to the number of defect constraint rows. If the number of control rows is less than the number of defect constraint rows, some of the defect constraint rows remain without a pair. Additionally, it is not guaranteed that every state or control row that is in the same frontal matrix with a defect constraint row increases the possibility of this defect constraint row satisfying the 1×1 pivot criterion by filling in its diagonal. As mentioned before, if the derivative of a dynamic constraint [Eq. (2)] with respect to a continuous optimal control problem variable is equal to zero, this zero derivative corresponds to a zero block in the defect constraint Jacobian. Because of this zero block the variable row cannot cause fill-in on the defect constraint row. Moreover, it is not guaranteed that every defect constraint row can form a nonsingular 2×2 pivot with every state and control row. On the other hand, because of the nonzero dense blocks in $\mathbf{J}$ and main diagonal blocks in $\mathbf{H}$, every state row causes fill-in on its corresponding defect constraint row (if the state row is eliminated before its corresponding defect constraint) and a defect constraint row, together with its corresponding state row, may form a nonsingular 2×2 pivot of the following form:

$$\begin{pmatrix} \times & \times \\ \times & 0 \end{pmatrix}.$$

Additionally, Fig. 8 shows that most of the delayed defect constraint rows satisfy the MA57 pivot threshold test while they are in the same frontal matrix with their corresponding state row. Thus, if

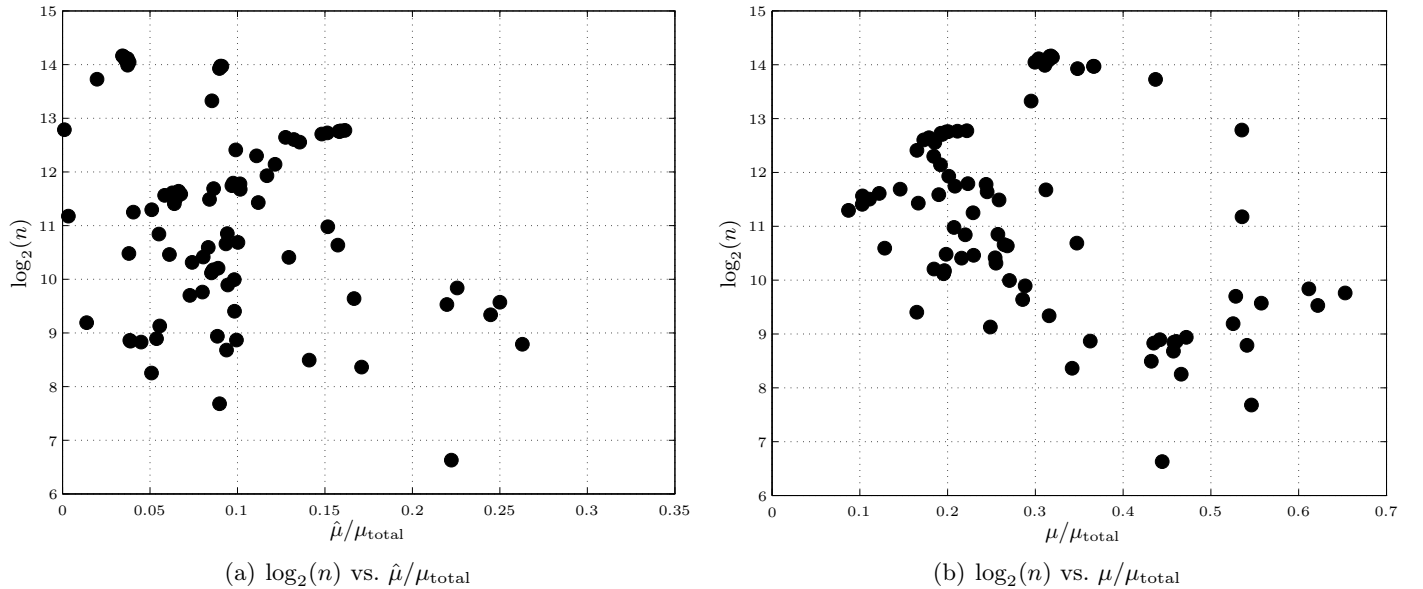


Figure 8: Base two logarithm of size of the KKT matrices, $\log_2(n)$, vs. ratio of the state-defect constraint pairs that are in the same frontal matrix during the MA57 symbolic factorization phase to the total number of state-defect constraint pairs, $\hat{\mu}/\mu_{\text{total}}$, and ratio of the state-defect constraint pairs that are in the same frontal matrix during the MA57 numerical factorization phase to the total number of state-defect constraint pairs, μ/μ_{total} .

a defect constraint and its corresponding state are in the same frontal matrix, that defect constraint is less likely to be delayed. In Section 6 we describe a graph coarsening method that forces state-defect constraint pairs to be adjacent in the reordered KKT matrix. Although two consecutive rows are not guaranteed to be in the same frontal matrix during the numerical factorization phase, forcing state-defect constraint pairs to be consecutive increases the possibility of these pairs being in the same frontal matrix.

6 State-Defect Constraint Pairing Graph Coarsening

In this section, a state-defect constraint pairing graph coarsening method, SGC method, that is tailored for KKT matrices arising from LGR collocation is described. The method consists of four steps: (1) finding the state-defect constraint pairs; (2) obtaining the coarsened matrix by collapsing the vertices of the state-defect constraint pairs into a multinode; (3) performing a fill-reducing ordering on the coarsened matrix; and (4) uncoarsening the reordered coarsened matrix. The state-defect constraint pairs are found using the properties of the continuous optimal control problem, the number of collocation points, and the known sparsity pattern of KKT matrices resulting from LGR collocation. The coarsened matrix is obtained by collapsing the vertices of the state-

defect constraint pairs and representing each pair with a multinode in the coarsened matrix. Each multinode is the union of the sparsity pattern of the state and defect constraint nodes. It is noted that the coarsened matrix is smaller and denser than the original matrix. For example, consider a KKT matrix of size $n \times n$ that has k state-defect constraint pairs. The coarsened matrix is then size of $(n - k) \times (n - k)$. The third step is to perform a fill-reducing ordering on the coarsened matrix. Fill-reducing orderings^{52–56} improve computational efficiency by maintaining the sparsity of a matrix after numerical factorization. In this paper, we employ the approximate minimum degree algorithm of Ref. 57 (one of the most widely used fill-reducing orderings) on the coarsened matrix. In the uncoarsening step each multinode that consists of a state-defect constraint pair in the coarsened matrix is projected back to the separate state and defect constraint nodes. As a result of these four steps, paired rows are forced to be adjacent in the reordered KKT matrix.

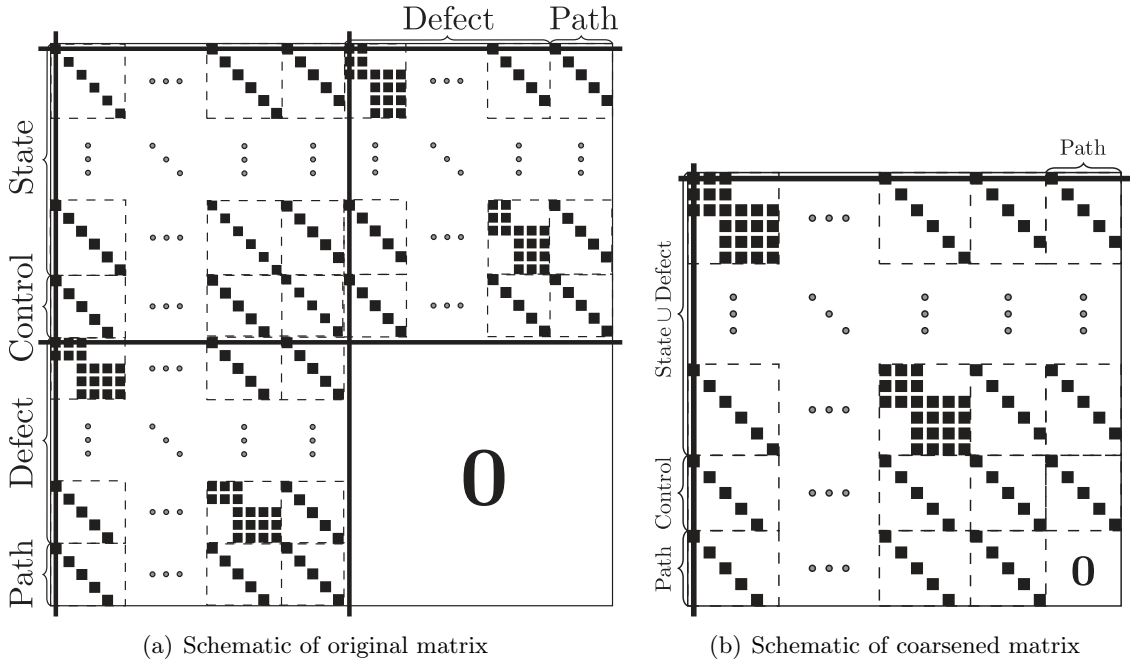


Figure 9: Schematics of original and coarsened matrices.

In order to visualize the graph coarsening step, consider the following example as shown in Figs. 9 and 10. Figure 9 illustrates the state-defect constraint pairing graph coarsening in matrix form. In Fig. 9(a), without loss of generality, thick solid vertical lines and horizontal lines show the columns and rows of a state-defect constraint pair in the original matrix, respectively. As seen in Fig. 9(b), the state-defect constraint pair in the original matrix is collapsed into one row and one column in the coarsened matrix. Furthermore, the sparsity pattern of the row and the column in the coarsened matrix is equal to the union of the sparsity pattern of the state-defect constraint

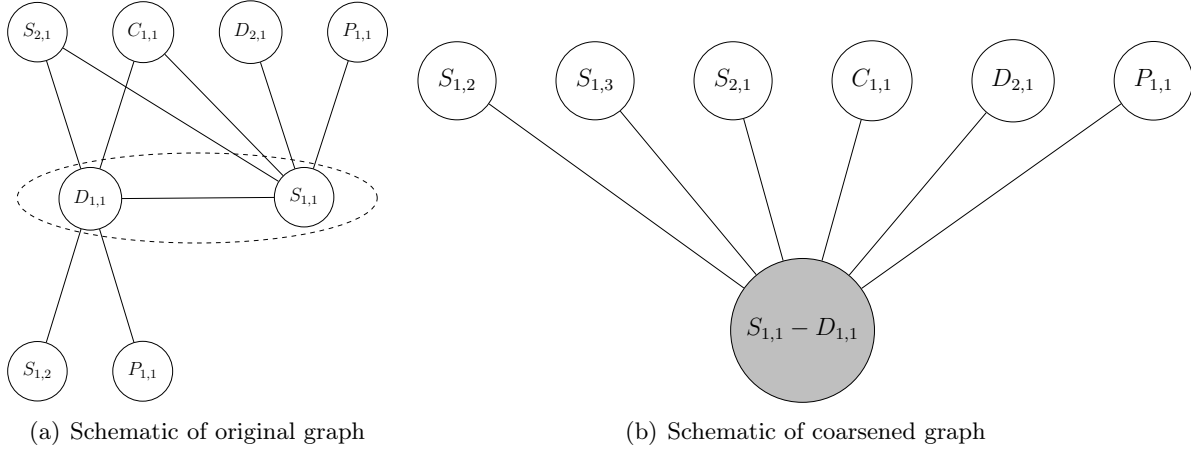


Figure 10: Schematic of original and coarsened graphs.

pair in the original matrix. Figure 10(a) illustrates the state-defect constraint pair that is shown in Fig 9(a) in the graph form, while Fig. 10(b) illustrates the multinode that consists of the union of the state and defect constraint nodes (where the symbols S , C , D , and P represent the state, control, defect constraint, and path constraint, respectively and the subscripts i, j refer to the i^{th} component of a variable or constraint at j^{th} collocation point).

7 State-Defect Constraint Pairing Graph Coarsening Results

In this section, the performance of the MA57 numerical factorization phase before and after applying the state-defect constraint pairing graph coarsening method (SGC method) are compared for sparse KKT matrices of various sizes that arise from the following ten benchmark optimal control problems. The first eight of these optimal control problems are taken from Ref. 45 and are given as follows: the maximum ascent of a one-dimensional sounding rocket (on page 213), the space shuttle reentry problem (on page 247), the low-thrust orbit transfer problem (on page 265), the range maximization of a hang glider (on page 282), the space station attitude reorientation problem (on page 293), the reorientation of an asymmetric rigid body (on page 299), the free flying robot (on page 326), the tumor anti-angiogenesis problem (on page 348). The ninth of these optimal control problems is the dynamic soaring problem taken from Ref. 46. The tenth of these optimal control problems is the orbit-raising problem taken from Ref. 37. KKT matrices that correspond to these ten benchmark optimal control problems can be found at the University of Florida Matrix Collection⁵⁸ under the group name UF-VDOL.

Table 1 provides the terminology used throughout the analysis. Additionally, the KKT matrices

Table 1: Terminology used to describe results of state-defect constraint pairing graph coarsening (SGC) method.

$\hat{\mu}$	=	Number of state-defect constraint pairs that are in the same frontal matrix during MA57 symbolic factorization
μ_{total}	=	Total number of state-defect constraint pairs
n	=	Matrix size
δ	=	Number of delayed pivots
f	=	Number of floating point operations in MA57 numerical factorization phase
t	=	Duration of MA57 numerical factorization phase
r	=	Number of floating point operations per unit time in MA57 numerical factorization phase
$nnz(\mathbf{A})$	=	Number of nonzeros in KKT matrices
$nnz(\mathbf{L})$	=	Number of nonzeros in $\mathbf{L}$

that are factorized without a graph coarsening method are referred to as the *original matrices* and the KKT matrices that are factorized after employing the SGC method are referred to as the *coarsened matrices*. The subscript “s” is added to quantities used to denote matrices that are coarsened via the SGC method in order to distinguish these quantities from the quantities corresponding to the original matrices. Figure 11 shows the ratio $\hat{\mu}_s/\mu_{\text{total}}$ as a function of n . Although the SGC method aims to decrease δ by increasing the number of state-defect constraint pairs that are in the same frontal matrix (thereby increasing $\hat{\mu}_s/\mu_{\text{total}}$), this method does not guarantee that every state-defect constraint pair will be in the same frontal matrix. As observed in Fig. 11, however, the SGC method places almost all of the state-defect constraint pairs in the same frontal matrix during the symbolic factorization. Figure 12 compares δ with δ_s . Figure 12(a) shows that δ varies from 400 to 4000 (without loss of generality, Fig. 12(a) shows the matrices with more than 400 delayed pivots) and almost all of the delayed pivots correspond to the defect constraint rows. As seen in Fig. 12(b), the SGC method significantly decreases the number of delayed pivots. Figure 13 compares the MA57 numerical factorization phase performance of the coarsened matrices with the MA57 numerical factorization phase performance of the original matrices in terms of f , t , and $r = f/t$. Figure 13(a) shows the ratio f_s/f as a function of $nnz(\mathbf{A})/\delta$. As seen in Fig. 13(a) when $nnz(\mathbf{A})/\delta$ is less than 50, employing the SGC method almost always decreases f . Next, Fig. 13(b) shows the ratio t_s/t as a function of $nnz(\mathbf{A})/\delta$ where, consistent with the flop ratios shown in Fig. 13(a), employing the SGC method decreases t if $nnz(\mathbf{A})/\delta < 50$. Additionally, the method decreases t even if f_s is greater than f (that is, when $nnz(\mathbf{A})/\delta > 50$ and $t_s/t < 1$). The reason for the decrease in t is the increase in the computational efficiency (that is, an increase in the number of floating point operations per unit time that can be performed). Finally, Fig. 13(c) shows

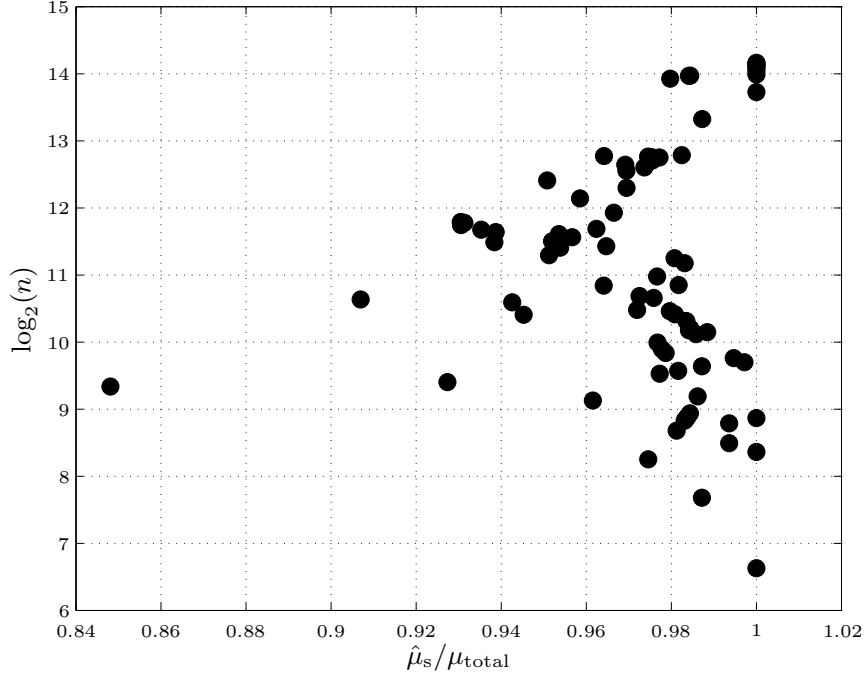


Figure 11: Base two logarithm of size of KKT matrices, $\log_2(n)$, vs. ratio of the number of state-defect constraint pairs that are in the same frontal matrix during the MA57 symbolic factorization phase to the total number of state-defect constraint pairs, $\hat{\mu}_s/\mu_{\text{total}}$, after applying the state-defect constraint pairing graph coarsening method.

r_s/r as a function of $\text{nnz}(\mathbf{A})/\delta$ where performing the method increases r by up to a factor of five. As a result, the SGC method increases the computational efficiency of the numerical factorization phase.

8 Problems with State and Control Equality Path Constraints

All of the optimal control problems that were analyzed in Section 7 in terms of the MA57 numerical factorization phase performance have either no equality path constraints or have only *control* equality path constraints. Therefore, KKT matrices arising from these problems have delayed pivots due primarily to the delayed defect constraints. In this section a discussion is provided regarding the application of the SGC method to problems with *state and control* equality path constraints. Instead of providing a general discussion on how the method applies to problems that have state and control equality path constraints, in Section 8.1 the performance of the SGC applied is shown on the kinetic batch reactor problem given on pages 331-336 of Ref. 45. This problem contains six state components, five control components, and five *state and control* equality path constraints in each three phases. In order to meet the specified accuracy tolerance of 10^{-6} , eleven mesh refine-

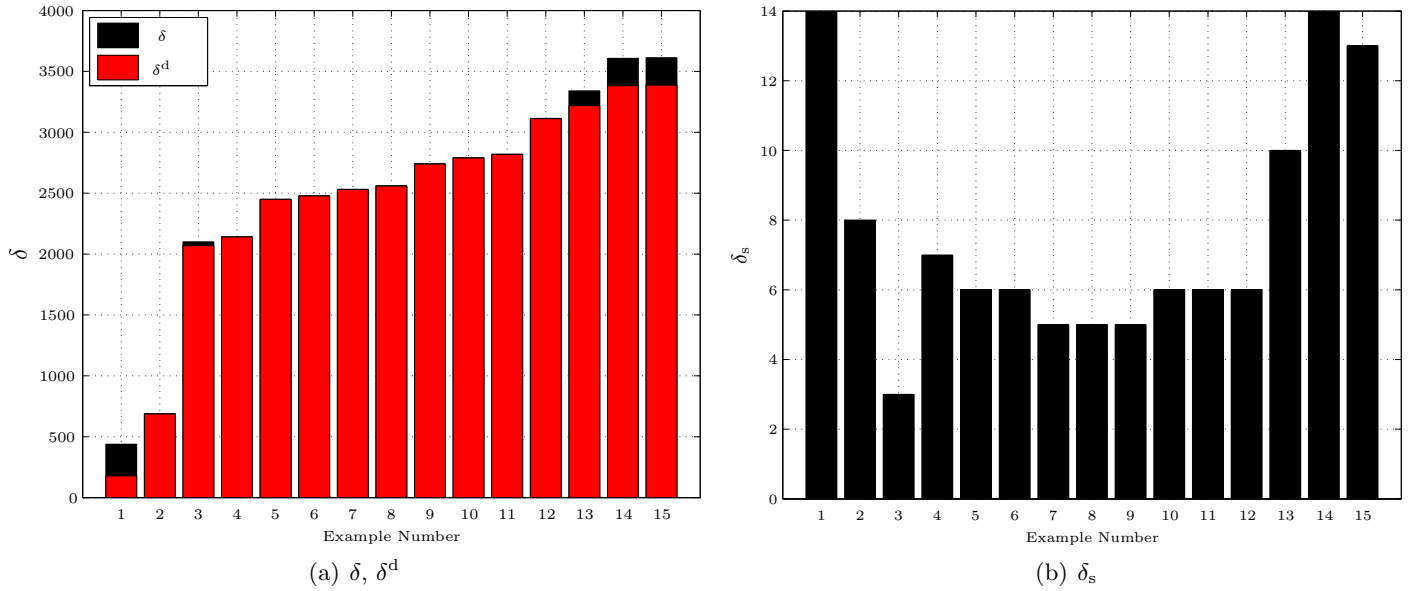


Figure 12: Comparison of total delayed pivots, δ , and delayed pivots that correspond to defect constraints, δ^d , before applying the state-defect constraint pairing graph coarsening method with the total delayed pivots, δ_s , after applying the state-defect constraint pairing graph coarsening method.

ment iterations were required by GPOPS-II. As the mesh refinement proceeds, the size of the KKT matrices increases from 2000 to 9000. After describing the performance of the SGC method on this example, in Section 8.2 an alternate graph coarsening method is described to further reduce the number of delayed pivots over the SGC method.

8.1 Application of SGC Method

Consider the numerical factorization of KKT matrices arising from the kinetic batch reactor problem given on pages 331-336 of Ref. 45 after performing the SGC method. Figure 14 shows three key properties: the nature of the delayed pivots, a comparison of the number of delayed pivots with and without repetition, and a comparison of number of delayed pivots before and after employing the SGC method. First, Fig. 14 shows that the delayed pivots correspond to both defect and equality path constraints. Second, the total number of delayed pivots with and without repetition are shown in Figs. 14(a) and 14(c). Delayed pivots with repetition gives the number of delayed pivots that is given as an output in MA57. Every time a delayed pivot (that is, a fully-summed variable that is passed to its father node in the elimination tree because of pivoting considerations) is passed further up the elimination tree, MA57 adds that variable to the total number of delayed pivots again. Furthermore, Figs. 14(a) and 14(c) show that the number of repetitions in the original matrices is greater than the number of repetitions in the coarsened matrices. Third, Fig. 14 shows

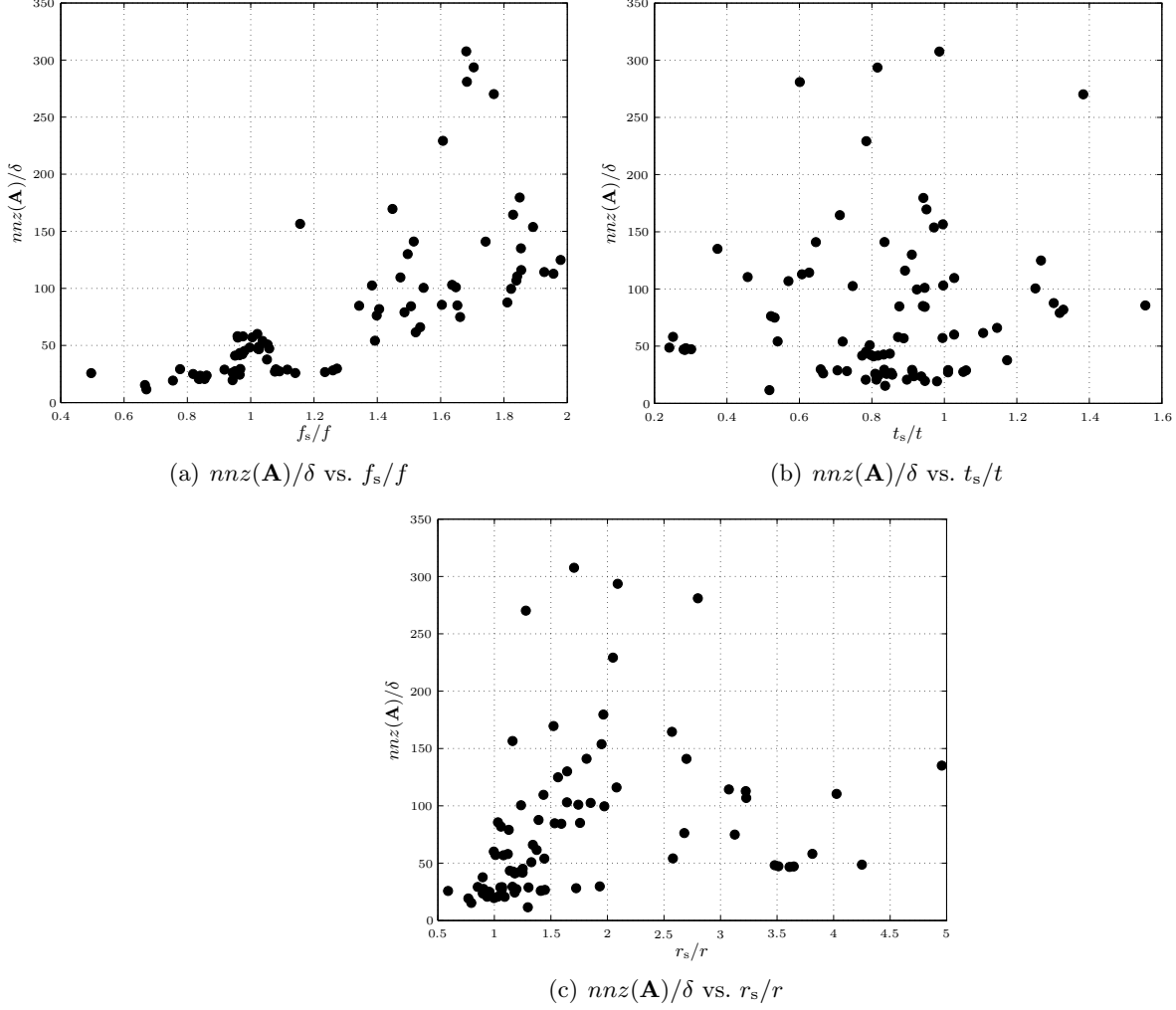


Figure 13: Ratio of number of nonzeros in KKT matrices to delayed pivots, $nnz(\mathbf{A})/\delta$, vs. ratio of average floating point operations, f_s/f , ratio of average time, t_s/t , ratio of average floating point operations per unit time, r_s/r , before and after applying the state-defect constraint pairing graph coarsening method.

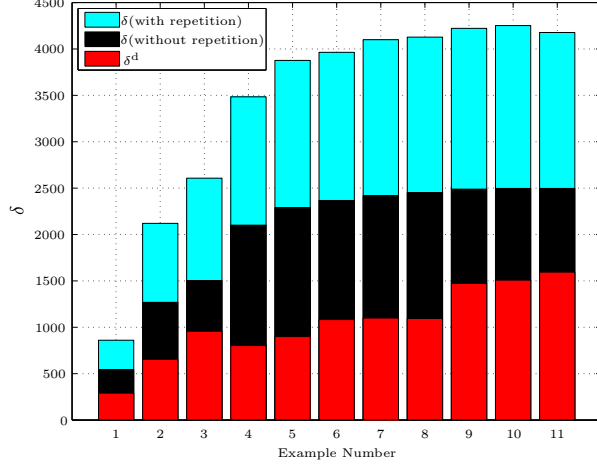
that employing the SGC method decreases δ , but even with this decrease, δ still varies between approximately 300 and 1200.

Figure 15 compares the MA57 numerical factorization phase performance of the coarsened matrices with the MA57 numerical factorization phase performance of the original matrices in terms of f , t , and $r = f/t$. Recall that in Section 7 the same comparison was made on the matrices that arise from the optimal control problems without any equality path constraints or with only control equality path constraints. The two main results of this comparison were as follows: when $nnz(\mathbf{A})/\delta < 50$, employing the SGC method almost always decreases f and the SGC method decreases t even if f_s is greater than f because the method increases r . The ratio $nnz(\mathbf{A})/\delta$ of the matrices that correspond to the kinetic batch reactor problem varies between 14 and 17, where both

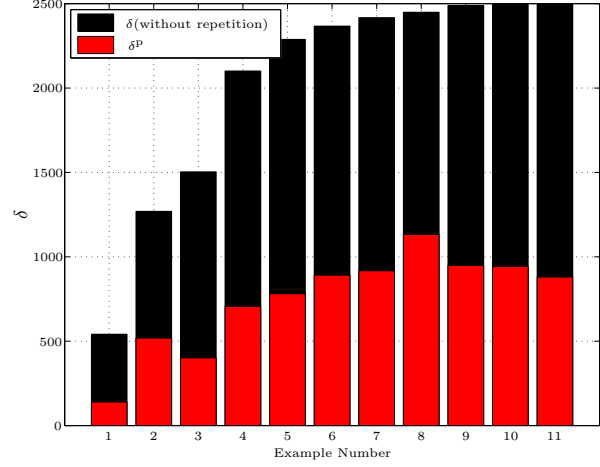
are less than 50. Therefore, as expected, employing the SGC method decreases f [Fig. 15(a)] and decreases t [Fig. 15(b)], but interestingly employing the SGC method decreases r [Fig. 15(c)]. In the kinetic batch reactor example, r decreases because the number of repetitions in the original matrices is greater than the number of repetitions in the coarsened matrices that also means that the original matrices pass more delayed pivots further up the tree compared to the coarsened matrices.

8.2 Alternate Graph Coarsening Method

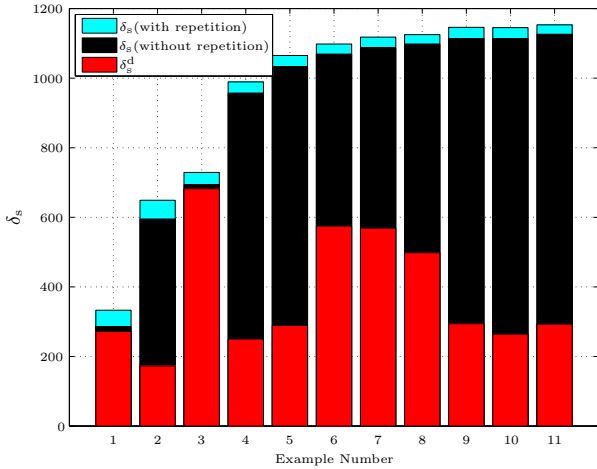
Consider again the numerical factorization of KKT matrices arising from the kinetic batch reactor problem given on pages 331-336 of Ref. 45. In this section it is attempted to further reduce the number of delayed pivots that occur during the numerical factorization of KKT matrices for this problem using an *alternate graph coarsening method* (AGC method), where the AGC method pairs defect and equality path constraints with the state and control components via a maximal matching algorithm.⁵⁹ In order to distinguish between the SGC and AGC methods, the subscript “a” is used to denote the AGC method. As shown in Fig. 16, the AGC method significantly decreases δ . Figure 17(a) shows $nnz(\mathbf{L}_a)/nnz(\mathbf{L}_s)$ as a function of $nnz(\mathbf{A})/\delta$. Although employing the AGC method decreases $nnz(\mathbf{L})$ by decreasing the number of delayed pivots, it increases $nnz(\mathbf{L})$ by altering the ordering that is chosen by the approximate minimum degree algorithm.⁵⁷ As seen in Fig. 17(a), performing the AGC method does not change $nnz(\mathbf{L})$ significantly because trade-off exists between decreasing the number of delayed pivots and altering the approximate minimum degree ordering. Figures 17(b)-17(d) compare the MA57 numerical factorization phase performance of the matrices that are coarsened using the AGC method with the MA57 numerical factorization phase performance of the matrices that are coarsened using the SGC method in terms of f , t , and $r = f/t$. As seen in Figs. 17(b)-17(d), employing the AGC method does not change f , t or r significantly because it does not change $nnz(\mathbf{L})$ significantly.



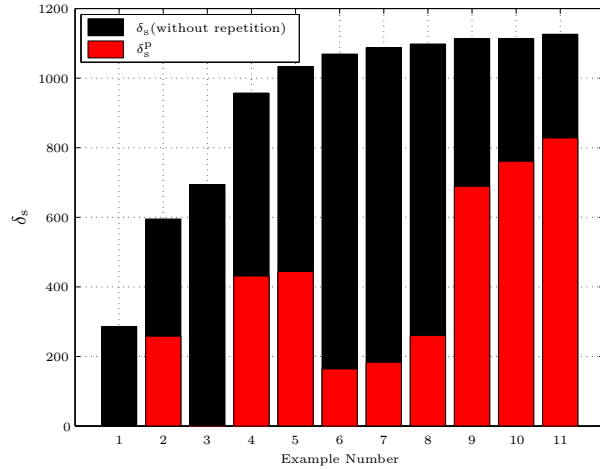
(a) δ (with repetition), δ (without repetition), and δ^d



(b) δ (without repetition) and δ^P

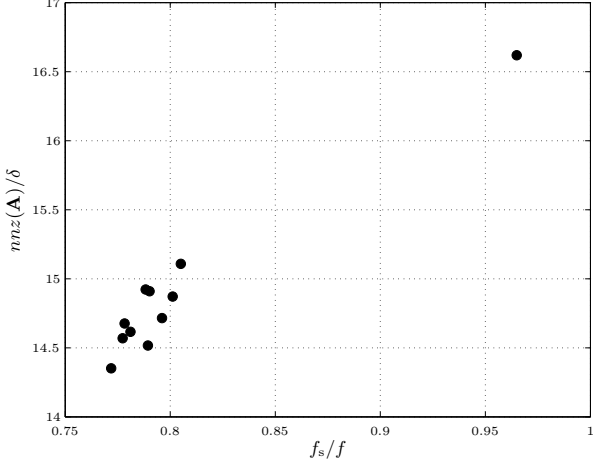


(c) δ_s (with repetition), δ_s (without repetition), and δ_s^d

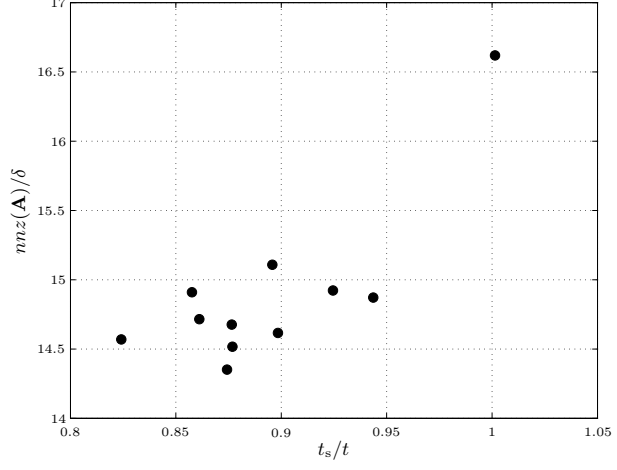


(d) δ_s (without repetition) and δ_s^P

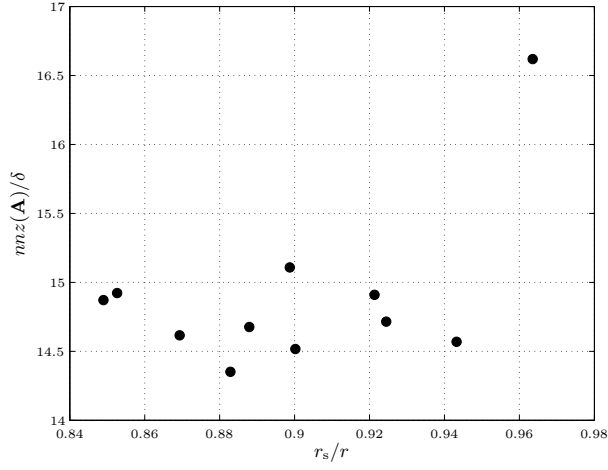
Figure 14: Comparison of total delayed pivots, δ (with and without repetition), delayed pivots that correspond to defect constraints, δ^d , and delayed pivots that correspond to equality path constraints, δ^P , before applying the state-defect constraint pairing graph coarsening method with the total delayed pivots, δ_s (with and without repetition), delayed pivots that correspond to defect constraints, δ_s^d , delayed pivots that correspond to equality path constraints, δ_s^P , after applying the state-defect constraint pairing graph coarsening method to example on pages 331-336 of Ref. 45.



(a) $nnz(\mathbf{A})/\delta$ vs. f_s/f



(b) $nnz(\mathbf{A})/\delta$ vs. t_s/t



(c) $nnz(\mathbf{A})/\delta$ vs. r_s/r

Figure 15: Ratio of number of nonzeros in KKT matrices to delayed pivots, $nnz(\mathbf{A})/\delta$, vs. ratio of average floating point operations, f_s/f , ratio of average time, t_s/t , ratio of average floating point operations per unit time, r_s/r , before and after applying the state-defect constraint pairing graph coarsening method to example on pages 331-336 of Ref. 45.

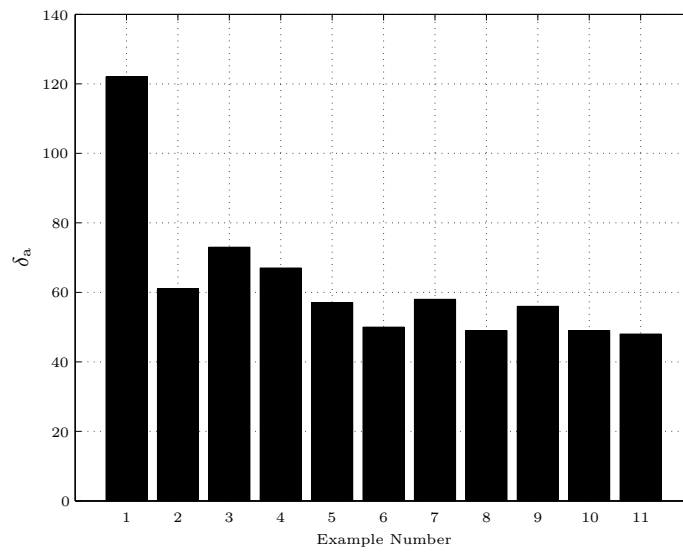


Figure 16: Number of delayed pivots after the alternate graph coarsening method, δ_a , applied to example on pages 331-336 of Ref. 45.

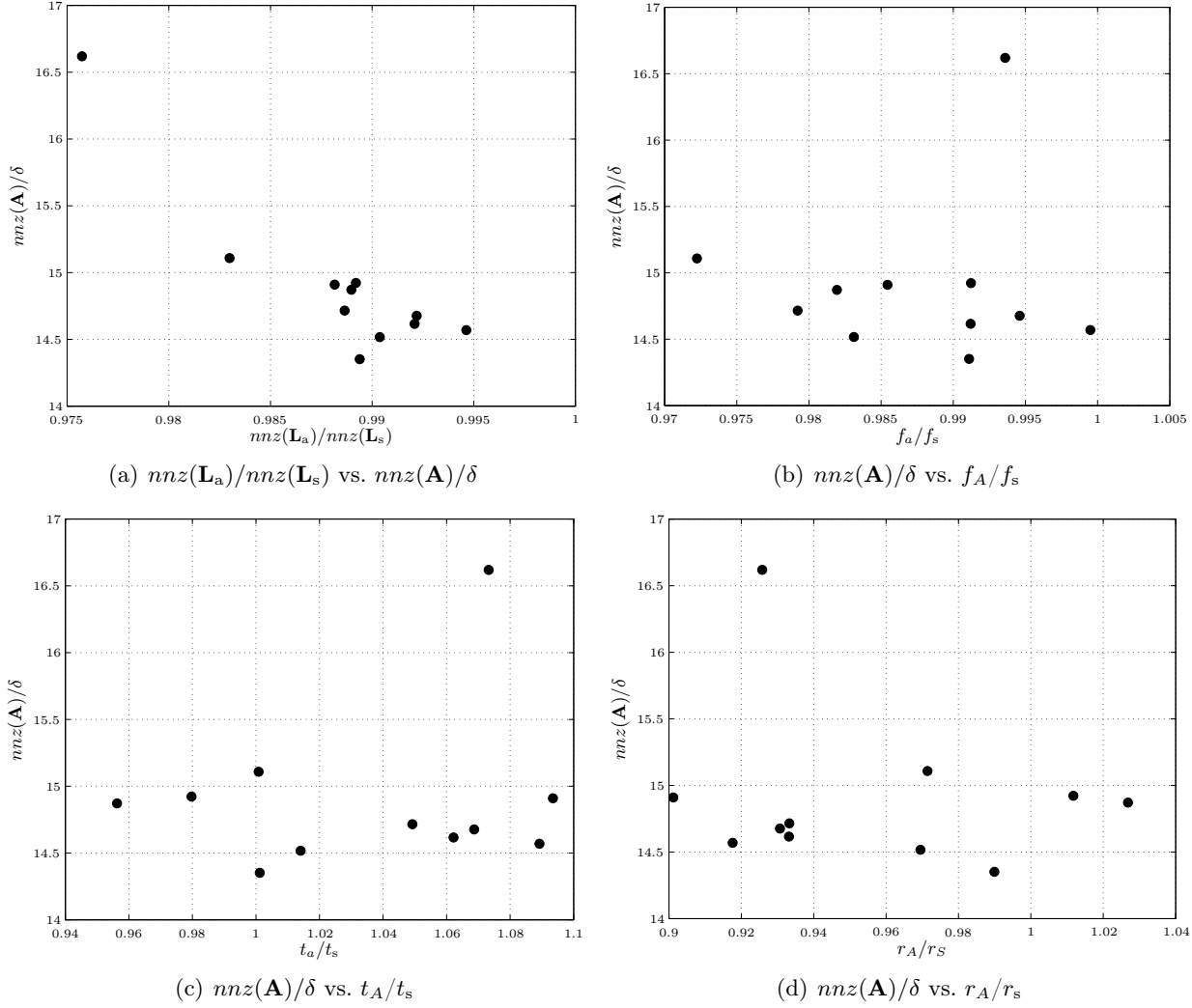


Figure 17: Ratio of number of nonzeros in KKT matrices to delayed pivots, $nnz(\mathbf{A})/\delta$, vs. ratio of number of nonzeros in $\mathbf{L}$, $nnz(\mathbf{L}_a)/nnz(\mathbf{L}_s)$, ratio of average floating point operations, f_a/f_s , ratio of average time, t_a/t_s , ratio of average floating point operations per unit time, r_a/r_s , after the alternate graph coarsening method and after the state-defect constraint pairing graph coarsening method applied to example on pages 331-336 of Ref. 45.

9 Conclusions

A state-defect constraint pairing graph coarsening method has been described for large sparse KKT matrices that arise from the discretization of optimal control problems via a Legendre-Gauss-Radau collocation method. Aggregate results for a wide set of benchmark optimal control problems have been presented. The results obtained in this study show that the state-defect constraint pairing graph coarsening method significantly reduces both the number of delayed pivots and the number of floating point operations and increases the computational efficiency by performing more floating point operations per unit time. It was also shown that the state-defect constraint pairing graph coarsening method is less effective when the continuous optimal control problem contains state and control equality path constraints because such matrices may have delayed pivots that correspond to both defect and path constraints. In order to further reduce the number of delayed pivots that occur during the numerical factorization of KKT matrices arising from optimal control problems with state and control equality path constraints, an alternate graph coarsening method was introduced that employed a maximal matching algorithm. It was shown that the alternate method significantly decreases the number of delayed pivots, but this decrease in the number of delayed pivots is offset by an increase in the number of nonzeros in the factorized matrix that is caused by altering the ordering that arises from the approximate minimum degree algorithm. Thus, the alternate graph coarsening method provides no further advantage over the state-defect constraint pairing graph coarsening method.

Acknowledgments

The authors gratefully acknowledge support for this research from the U.S. Office of Naval Research under grants N00014-11-1-0068 and N00014-15-1-2048, from the U.S. National Science Foundation under grants CBET-1404767 and DMS-1522629, and from the U.S. Space and Naval Warfare Systems Command under grant N65236-13-1-1000.

References

- [1] Gill, P. E., Murray, W., and Saunders, M. A., “SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization,” *SIAM Journal on Optimization*, Vol. 12, No. 4, 2002, pp. 979–1006.
- [2] Biegler, L. T. and Zavala, V. M., “Large-Scale Nonlinear Programming Using IPOPT: An Integrating Framework for Enterprise-Wide Dynamic Optimization,” *Computers & Chemical Engineering*, Vol. 33, No. 3, 2009, pp. 575–582.

- [3] Duff, I. S. and Reid, J. K., “MA47, A Fortran Code for Direct Solution of Indefinite Sparse Symmetric Linear Systems,” *Report RAL*, 1995, pp. 95–001.
- [4] Duff, I. S., “MA57– A Code for the Solution of Sparse Symmetric Definite and Indefinite Systems,” *ACM Transactions on Mathematical Software*, Vol. 30, No. 2, 2004, pp. 118–144.
- [5] Amestoy, P. R., Duff, I. S., L’Excellent, J.-Y., and Koster, J., *MUMPS: A General Purpose Distributed Memory Sparse Solver*, Springer, 2001.
- [6] Schenk, O. and Gärtner, K., “Two-Level Dynamic Scheduling in PARDISO: Improved Scalability on Shared Memory Multiprocessing Systems,” *Parallel Computing*, Vol. 28, No. 2, 2002, pp. 187–197.
- [7] Elnagar, G., Kazemi, M., and Razzaghi, M., “The Pseudospectral Legendre Method for Discretizing Optimal Control Problems,” *IEEE Transactions on Automatic Control*, Vol. 40, No. 10, 1995, pp. 1793–1796.
- [8] Elnagar, G. and Razzaghi, M., “A Collocation-Type Method for Linear Quadratic Optimal Control Problems,” *Optimal Control Applications and Methods*, Vol. 18, No. 3, 1998, pp. 227–235.
- [9] Fahroo, F. and Ross, I. M., “Costate Estimation by a Legendre Pseudospectral Method,” *Journal of Guidance, Control, and Dynamics*, Vol. 24, No. 2, 2001, pp. 270–277.
- [10] Fahroo, F. and Ross, I. M., “Direct Trajectory Optimization by a Chebyshev Pseudospectral Method,” *Journal of Guidance, Control, and Dynamics*, Vol. 25, No. 1, 2002, pp. 160–166.
- [11] Benson, D. A., Huntington, G. T., Thorvaldsen, T. P., and Rao, A. V., “Direct Trajectory Optimization and Costate Estimation via an Orthogonal Collocation Method,” *Journal of Guidance, Control, and Dynamics*, Vol. 29, No. 6, November-December 2006, pp. 1435–1440.
- [12] Huntington, G. T., Benson, D. A., and Rao, A. V., “Optimal Configuration of Tetrahedral Spacecraft Formations,” *The Journal of the Astronautical Sciences*, Vol. 55, No. 2, April-June 2007, pp. 141–169.
- [13] Huntington, G. T. and Rao, A. V., “Optimal Reconfiguration of Spacecraft Formations Using the Gauss Pseudospectral Method,” *Journal of Guidance, Control, and Dynamics*, Vol. 31, No. 3, May-June 2008, pp. 689–698.
- [14] Gong, Q., Fahroo, F., and Ross, I. M., “Spectral Algorithm for Pseudospectral Methods in Optimal Control,” *Journal of Guidance, Control, and Dynamics*, Vol. 31, No. 3, May-June 2008.
- [15] Gong, Q., Ross, I. M., Kang, W., and Fahroo, F., “Connections Between the Covector Mapping Theorem and Convergence of Pseudospectral Methods,” *Computational Optimization and Applications*, Vol. 41, No. 3, December 2008, pp. 307–335.
- [16] Rao, A. V., Benson, D. A., Darby, C. L., Francolin, C., Patterson, M. A., Sanders, I., and Huntington, G. T., “Algorithm 902: GPOPS, A Matlab Software for Solving Multiple-Phase Optimal Control Problems Using the Gauss Pseudospectral Method,” *ACM Transactions on Mathematical Software*, Vol. 37, No. 2, April-June 2010, Article 22, 39 pages.
- [17] Darby, C. L., Hager, W. W., and Rao, A. V., “An *hp*-Adaptive Pseudospectral Method for Solving Optimal Control Problems,” *Optimal Control Applications and Methods*, Vol. 32, No. 4, July-August 2011, pp. 476–502.

- [18] Darby, C. L., Hager, W. W., and Rao, A. V., “Direct Trajectory Optimization Using a Variable Low-Order Adaptive Pseudospectral Method,” *Journal of Spacecraft and Rockets*, Vol. 48, No. 3, May–June 2011, pp. 433–445.
- [19] Kameswaran, S. and Biegler, L. T., “Convergence Rates for Direct Transcription of Optimal Control Problems Using Collocation at Radau Points,” *Computational Optimization and Applications*, Vol. 41, No. 1, 2008, pp. 81–126.
- [20] Garg, D., Patterson, M. A., Darby, C. L., Francolin, C., Huntington, G. T., Hager, W. W., and Rao, A. V., “Direct Trajectory Optimization and Costate Estimation of Finite-Horizon and Infinite-Horizon Optimal Control Problems via a Radau Pseudospectral Method,” *Computational Optimization and Applications*, Vol. 49, No. 2, June 2011, pp. 335–358. DOI: 10.1007/s10589-00-09291-0.
- [21] Garg, D., Patterson, M. A., Hager, W. W., Rao, A. V., Benson, D. A., and Huntington, G. T., “A Unified Framework for the Numerical Solution of Optimal Control Problems Using Pseudospectral Methods,” *Automatica*, Vol. 46, No. 11, November 2010, pp. 1843–1851. DOI: 10.1016/j.automatica.2010.06.048.
- [22] Garg, D., Hager, W. W., and Rao, A. V., “Pseudospectral Methods for Solving Infinite-Horizon Optimal Control Problems,” *Automatica*, Vol. 47, No. 4, April 2011, pp. 829–837. DOI: 10.1016/j.automatica.2011.01.085.
- [23] Patterson, M. A. and Rao, A. V., “GPOPS- II: A MATLAB Software for Solving Multiple-Phase Optimal Control Problems Using hp-Adaptive Gaussian Quadrature Collocation Methods and Sparse Nonlinear Programming,” *ACM Transactions on Mathematical Software*, Vol. 41, No. 1, October 2014, pp. 1:1–1:37.
- [24] Patterson, M. A., Hager, W. W., and Rao, A. V., “A *ph* Mesh Refinement Method for Optimal Control,” *Optimal Control Applications and Methods*, Vol. 36, No. 4, July–August 2015, pp. 398–421.
- [25] Francolin, C. C., Hager, W. W., and Rao, A. V., “Costate Approximation in Optimal Control Using Integral Gaussian Quadrature Collocation Methods,” *Optimal Control Applications and Methods*, Published Online in Early View, January 2014. DOI 10.1002/oca2112.
- [26] Karypis, G. and Kumar, V., “A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs,” *SIAM Journal on Scientific Computing*, Vol. 20, No. 1, 1998, pp. 359–392.
- [27] Pothen, A., Simon, H. D., Wang, L., and Barnard, S. T., “Towards a Fast Implementation of Spectral Nested Dissection,” *Proceedings of the 1992 ACM/IEEE conference on Supercomputing*, IEEE Computer Society Press, 1992, pp. 42–51.
- [28] Pothen, A., Simon, H. D., and Liou, K.-P., “Partitioning sparse matrices with eigenvectors of graphs,” *SIAM Journal on Matrix Analysis and Applications*, Vol. 11, No. 3, 1990, pp. 430–452.
- [29] Miller, G. L., Teng, S.-H., Thurston, W., and Vavasis, S. A., *Automatic mesh Partitioning*, Springer, 1993.
- [30] Miller, G. L., Teng, S.-H., and Vavasis, S. A., “A unified geometric approach to graph separators,” *Foundations of Computer Science, 1991. Proceedings., 32nd Annual Symposium on*, IEEE, 1991, pp. 538–547.

- [31] Cheng, C.-K. and Wei, Y.-C., “An improved two-way partitioning algorithm with stable performance [VLSI],” *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, Vol. 10, No. 12, 1991, pp. 1502–1511.
- [32] Hendrickson, B. and Leland, R. W., “A Multi-Level Algorithm For Partitioning Graphs.” *SC*, Vol. 95, 1995, pp. 28.
- [33] Mansour, N., Ponnusamy, R., Choudhary, A., and Fox, G., “Graph Contraction for Physical Optimization Methods: a Quality-Cost Tradeoff for Mapping Data on Parallel Computers,” *Proceedings of the 7th International Conference on Supercomputing*, ACM, 1993, pp. 1–10.
- [34] Cong, J. and Shinnerl, J. R., *Multilevel Optimization in VLSICAD*, Springer, 2003.
- [35] Safo, I., Sanders, P., and Schulz, C., “Advanced Coarsening Schemes for Graph Partitioning,” *Experimental Algorithms*, Springer, 2012, pp. 369–380.
- [36] Word, D. P., Kang, J., Akesson, J., and Laird, C. D., “Efficient Parallel Solution of Large-Scale Nonlinear Dynamic Optimization Problems,” *Computational Optimization and Applications*, Vol. 59, 2014, pp. 667–688.
- [37] Patterson, M. A. and Rao, A. V., “Exploiting Sparsity in Direct Collocation Pseudospectral Methods for Solving Optimal Control Problems,” *Journal of Spacecraft and Rockets*, Vol. 49, No. 2, March–April 2012, pp. 364–377.
- [38] Nocedal, J. and Overton, M. L., “Projected Hessian updating algorithms for nonlinearly constrained optimization,” *SIAM Journal on Numerical Analysis*, Vol. 22, No. 5, 1985, pp. 821–850.
- [39] Sylvester, J. J., “XIX. A Demonstration of the Theorem That Every Homogeneous Quadratic Polynomial Is Reducible by Real Orthogonal Substitutions to the Form of a Sum of Positive and Negative Squares,” *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, Vol. 4, No. 23, 1852, pp. 138–142.
- [40] Levenberg, K., “A method for the Solution of Certain Problems in Least Squares,” *Quarterly of Applied mathematics*, Vol. 2, 1944, pp. 164–168.
- [41] Wächter, A. and Biegler, L. T., “On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming,” *Mathematical programming*, Vol. 106, No. 1, 2006, pp. 25–57.
- [42] Duff, I. S. and Reid, J. K., *MA27-a set of Fortran subroutines for solving sparse symmetric sets of linear equations*, UKAEA Atomic Energy Research Establishment, 1982.
- [43] Duff, I. S. and Reid, J. K., “The multifrontal solution of indefinite sparse symmetric linear,” *ACM Transactions on Mathematical Software (TOMS)*, Vol. 9, No. 3, 1983, pp. 302–325.
- [44] Liu, J. W., “The multifrontal method for sparse matrix solution: Theory and practice,” *SIAM review*, Vol. 34, No. 1, 1992, pp. 82–109.
- [45] Betts, J. T., *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*, Vol. 19, SIAM Press, Philadelphia, Pennsylvania, 2010.
- [46] Zhao, Y. J., “Optimal Patterns of Glider Dynamic Soaring,” *Optimal Control applications and methods*, Vol. 25, No. 2, 2004, pp. 67–89.

- [47] Goddard, R. H., “A Method of Reaching Extreme Altitudes.” *Nature*, Vol. 105, 1920, pp. 809–811.
- [48] Ledzewicz, U. and Schättler, H., “Analysis of Optimal Controls for a Mathematical Model of Tumour Anti-Angiogenesis,” *Optimal Control Applications and Methods*, Vol. 29, No. 1, 2008, pp. 41–57.
- [49] Bryson, A. E., *Applied Optimal Control: Optimization, Estimation, and Control*, CRC Press, 1975.
- [50] Bulirsch, R., Nerz, E., Pesch, H. J., and von Stryk, O., *Combining Direct and Indirect Methods in Optimal Control: Range Maximization of a Hang Glider*, Springer, 1993.
- [51] Sakawa, Y., “Trajectory Planning of a Free-Flying Robot by Using the Optimal Control,” *Optimal Control Applications and Methods*, Vol. 20, No. 5, 1999, pp. 235–248.
- [52] Tinney, W. F. and Walker, J. W., “Direct Solutions of Sparse Network Equations by Optimally Ordered Triangular Factorization,” *Proceedings of the IEEE*, Vol. 55, No. 11, 1967, pp. 1801–1809.
- [53] Davis, T. A., Amestoy, P., and Duff, I. S., “An Approximate Minimum Degree Ordering Algorithm,” *SIAM Journal on Matrix Analysis and Applications*, Vol. 17, No. 4, December 1996, pp. 886–905.
- [54] George, A., “Nested Dissection of a Regular Finite Element Mesh,” *SIAM Journal on Numerical Analysis*, Vol. 10, No. 2, 1973, pp. 345–363.
- [55] Gibbs, N. E., Poole, Jr, W. G., and Stockmeyer, P. K., “An Algorithm for Reducing the Bandwidth and Profile of a Sparse Matrix,” *SIAM Journal on Numerical Analysis*, Vol. 13, No. 2, 1976, pp. 236–250.
- [56] Chan, W.-M. and George, A., “A Linear Time Implementation of the Reverse Cuthill-McKee Algorithm,” *BIT Numerical Mathematics*, Vol. 20, No. 1, 1980, pp. 8–14.
- [57] Amestoy, P. R., Davis, T. A., and Duff, I. S., “Algorithm 837: AMD, An Approximate Minimum Degree Ordering Algorithm,” *ACM Transactions on Mathematical Software*, Vol. 30, No. 3, 2004, pp. 381–388.
- [58] Davis, T. A. and Hu, Y., “The University of Florida Sparse Matrix Collection,” *ACM Transactions on Mathematical Software (TOMS)*, Vol. 38, No. 1, 2011, pp. 1.
- [59] Davis, T. A., *Direct Methods for Sparse Linear Systems*, Vol. 2, SIAM Press, Philadelphia, Pennsylvania, 2006.