Convex optimization techniques in compliant assembly simulation

**Maria Stefanova[1] · Olga Minevich[2] · Stanislav Baklanov[1] ·
Margarita Petukhova[1] · Sergey Lupuleac[1] · Boris Grigor'ev[1] ·
Michael Kokkolaras[3]**

Abstract

A special class of quadratic programming (QP) problems is considered in this paper. This class emerges in simulation of assembly of large-scale compliant parts, which involves the formulation and solution of contact problems. The considered QP problems can have up to 20,000 unknowns, the Hessian matrix is fully populated and ill-conditioned, while the matrix of constraints is sparse. Variation analysis and optimization of assembly process usually require massive computations of QP problems with slightly different input data. The following optimization methods are adapted to account for the particular features of the assembly problem: an interior point method, an active-set method, a Newton projection method, and a pivotal algorithm for the linear complementarity problems. Equivalent formulations of the QP problem are proposed with the intent of them being more amenable to the considered methods. The methods are tested and results are compared for a number of aircraft assembly simulation problems.

✉ Maria Stefanova
stefanova.m@list.ru

✉ Olga Minevich
minewitsch@rambler.ru

1   Peter the Great St. Petersburg Polytechnic University, Saint Petersburg, Russia

2   Technische Universität München, Munich, Germany

3   McGill University, Montreal, Canada

1 Introduction

In the last decade a new modeling approach has been developed and applied to variation simulation and assembly optimization in aerospace and automotive industry (see Lupuleac et al. 2010, 2011, 2019b; Dahlström and Lindkvist 2007; Lindau et al. 2016; Yang et al. 2016). This approach considers the contact interaction between compliant parts. By using the variational formulation (Galin 1961; Tu and Gazis 1964; Lions and Stampacchia 1967; Kinderlehrer and Stampacchia 1980) and the substructuring (Turner et al. 1956; Guyan 1965; Wriggers 2006; Petukhova et al. 2014), contact detection is reduced to a quadratic programming problem that allows for large-scale computations of contact problems during variation simulation and assembly optimization. Since the accuracy of the assembly model influences the simulation results, it is important to use fine FEM meshes that can reflect the most essential features of assemblies. Implementation of highly refined computational meshes and need for massive computations of problems with slightly different input data require choosing appropriately tailored solvers for the QP problem; that is the primary objective of the present paper.

The paper is organized as follows: in Sect. 2 the contact problem arising in assembly simulation is presented and its features are described; task-level parallelization is also discussed. Section 3 is dedicated to the adaptation of numerical methods for solving the described contact problem. In Sect. 4 computational results for the adapted methods are provided and discussed for several problems of aircraft assembly simulation. Conclusions are presented in Sect. 5.

2 Formulation of contact problem

A finite element model of several parts fixed in an assembly jig is schematically shown in Fig. 1. This kind of assembly is of *sandwich* type, i.e., at least three parts are lap-joint. The assemblies with no intermediate parts are called *simple* (for instance, the assemblies consisting of two parts). The area where the parts overlap is called the *junction* area. The finite element meshes of the parts are assumed to be conformal. The nodes of the finite element model that are located in the junction area are denoted as *computational* nodes. These nodes are highlighted in Fig. 1. The degrees of freedom in the computational nodes considered in the analysis are arranged into the vector $x \in \mathbb{R}^n$, where n is the dimension of the problem. Although
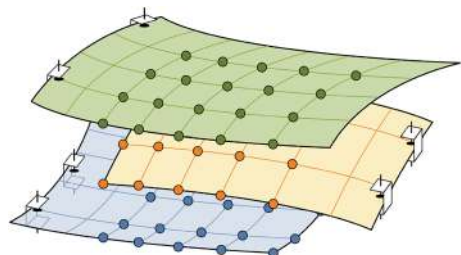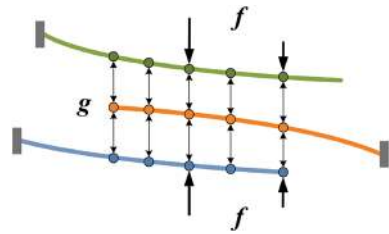
Fig. 1 Schematic outline of the assembly

Fig. 2 Cross section of the assembly

x usually contains only normal displacements, tangential displacements and rotations can also be included.

Relative tangential displacements are assumed to be small; therefore, a node-to-node contact model is used. According to this model only the predefined pairs of nodes can come in contact. Figure 2 depicts the corresponding pairs of nodes in the junction area connected with arrows. In sandwich-type assemblies some nodes can belong to two pairs simultaneously. The initial distance between corresponding computational nodes is given by the vector of initial gap $g \in \mathbb{R}^m$, where m is the total number of contact pairs.

As a rule, the fasteners are modeled in two ways depending of their type. The temporary fasteners used in the aircraft assembly usually provide the fastening load by means of prestressed spring. Such fasteners are modelled by the loads applied to computational nodes. The permanent fastening elements (e.g., rivets or bolts) are modelled by constraining the relative displacements of parts in the corresponding computational nodes. In some cases, these two ways are combined by applying the normal fastening load while also constraining the relative tangential displacement of the assembled parts.

By using the substructuring the contact problem can be reformulated in QP form (see Lupuleac et al. 2011, 2019a for details):

$$\min \ \frac{1}{2}x^T K x - f^T x,$$
$$\text{s.t.} \ \ A^T x - g \leq 0, \tag{1}$$

where $f \in \mathbb{R}^n$ is the vector of loads (for example, loads from fastening elements), $K \in \mathbb{R}^{n \times n}$ is the reduced stiffness matrix computed using finite element analysis, and $A \in \mathbb{R}^{n \times m}$ ($n \geq m$) is a linear operator that defines the contact pairs.

As a rule, contact problems arising in assembly simulation have a number of unknowns that varies from 1000 to 20,000. The mechanical properties of the structure, the influence of assembly jig type, and the configuration of the joining parts are described by the stiffness matrix K and the constraint matrix A. When performing a series of computations, these matrices do not change. This feature is exploited to define some auxiliary matrices based on matrices K and A once for every assembly model. Matrix K is symmetric, positive-definite and ill-conditioned. It has block-diagonal structure with fully populated blocks. The number and the size of the blocks are defined by the number of assembled parts and the number of considered degrees of freedom in computational nodes in each part. Matrix A, which defines

the pairs of nodes that can come in contact, is sparse in the considered problems: every column of the matrix contains 1 or 2 non-zero elements.

The analysis of assembly quality involves the simulation of shape variations of the joined parts. The shape variations are caused by tolerances in manufacturing, positioning, fixating, etc. In industrial applications (e.g., see Lupuleac et al. 2019b) the variation simulation is realized by generation of the large number of initial gap vectors g (cloud of gaps) and subsequent massive solving of contact problem (1). Such an approach can be considered as generalization of the Method of Influence Coefficients proposed in Liu and Hu 1997.

2.1 Equivalent formulations of the QP problem

Considering that the matrices K and A are usually the same for the series of computations during aircraft assembly simulation and optimization, problem (1) can be reformulated in a form that is more attractive from a computational point of view. Specifically, the size of the QP problem can be reduced and the form of the constraint matrix A can be simplified. Consider two different reformulations of problem (1): dual problem and relative problem.

The dual formulation of problem (1) is given by

$$\begin{aligned} \max \quad & -\tfrac{1}{2}\lambda^T Q \lambda + p^T \lambda + s, \\ \text{s.t.} \quad & \lambda \geq 0, \end{aligned} \tag{2}$$

where $\lambda \in \mathbb{R}^m$ is the vector of Lagrange multipliers corresponding to contact forces in the junction area, $Q = A^T K^{-1} A \in \mathbb{R}^{m \times m}$ is a symmetric positive-definite fully populated matrix, $p = A^T K^{-1} f - g \in \mathbb{R}^m$ and $s = -\tfrac{1}{2} f^T K^{-1} f \in \mathbb{R}^1$ (Lupuleac et al. 2013).

The relative formulation of problem (1) is based on a vector of relative displacements in the junction area defined as $u = A^T x \in \mathbb{R}^m$ and may be obtained through the minimax form of the problem. First, an additional relation should be derived from optimality conditions. By multiplying the stationarity condition $Kx - f + A\lambda = 0$ with $A^T K^{-1}$ we get

$$u - A^T K^{-1} f + Q\lambda = 0.$$

Excluding λ the relation that binds variable x to u is obtained:

$$x = K^{-1} f - K^{-1} A Q^{-1} (A^T K^{-1} f - u) = K^{-1} A Q^{-1} u.$$

Substutuing x to the Lagrangian function $L(x, \lambda) = \tfrac{1}{2} x^T K x - f^T x + (A^T x - g)^T \lambda$ yields:

$$L(u, \lambda) = \frac{1}{2} u^T Q^{-1} u - f^T K^{-1} A Q^{-1} u + (u - g)^T \lambda.$$

Then the minimax form of the problem is

$$\min_{u \leq g} \max_{\lambda \geq 0} L(u, \lambda).$$

Calculating the maximum, we obtain a relative formulation of problem (1):

$$\min \ \tfrac{1}{2} u^T \tilde{K} u - \tilde{f}^T u,$$
$$\text{s.t.} \ \ u - g \leq 0, \tag{3}$$

where $\tilde{K} = (A^T K^{-1} A)^{-1} \in \mathbb{R}^{m \times m}$ is a symmetric positive-definite fully populated matrix and $\tilde{f} = \tilde{K} A^T K^{-1} f \in \mathbb{R}^m$.

All the listed formulations of the QP problem are equivalent and their solutions are connected by the following formulas:

$$x = K^{-1}(f - A\lambda),$$
$$\lambda = \tilde{f} - \tilde{K} u.$$

Two more formulations arise when applying the Newton projection method and pivotal algorithm for aircraft assembly problems. The Newton projection method is only applicable if the constraint matrix is the identity (see Sect. 3.3 for details). To solve the primal problem (1) with general constraints $A^T x - g \leq 0$ a change of variables is proposed. If the total number of constraints is less than the number of variables, extra rows are added to the matrix A^T to make it square. The proposed idea is to add unit vector rows in such a way that the new matrix becomes nondegenerate. This technique allows to keep the constraint matrix sparse. Thus, a new variable $\tilde{x} = \tilde{A}^T x = [A, a_{m+1}, \ldots, a_n]^T x$ is introduced and a change of variables in the functional $F(x) = F(\tilde{A}^{-1} \tilde{x})$ is made. Finally, fictitious constraints $[a_{m+1}, \ldots, a_n]^T x \leq +\infty$ are introduced, and problem (1) is reformulated as follows:

$$\min \ \tfrac{1}{2} \tilde{x}^T \tilde{A}^{-1} K \tilde{A}^{-T} \tilde{x} - f^T \tilde{A}^{-T} \tilde{x},$$
$$\text{s.t.} \ \ \tilde{x} - \tilde{g} \leq 0, \tag{4}$$

where $\tilde{g} = [g^T, +\infty]^T$.

The pivotal algorithm considered in the paper is used to solve linear complementarity problems LCP (q, M) that consists in finding the vectors $w, z \in \mathbb{R}^N$ such that

$$w = q + Mz,$$
$$z^T w = 0, \tag{5}$$
$$w \geq 0, z \geq 0$$

for given a vector $q \in \mathbb{R}^N$ and a matrix $M \in \mathbb{R}^{N \times N}$ or in concluding that no such pair of vectors w, z exists; N denotes the dimension of the LCP (q, M). How the problems (1)–(3) can be formulated as LCP the reader may find in Sect. 3.4.

The basic information about the formulations (1)–(4) such as the number of variables, the number of constraints, type of constraints and structure of Hessian, is presented in Table 1. The details of the LCP formulations are described in Sect. 3.4.

Table 1 Comparison of equivalent formulations

	Number of variables	Number of constraints	Type of constraints	Number of blocks in Hessian
Primal (1)	n	m	Linear inequality	*
Primal (4)	n	n	Bound	1
Dual (2)	m	m	Bound	1
Relative (3)	m	m	Bound	1

*Equal to the number of joined parts

2.2 Specifics of parallelization for considered problems

The most important computational feature of the contact problem arising in variation simulation is the need to carry out many similar computations for the same large-sized assembly model; namely, to calculate the displacements for different sets of forces $F = \{f_j\}^p$ and for different variants of the initial gap $G = \{g_i\}^k$. The total number of the serial computations can reach up to 10^6 runs during variation simulation and assembly optimization for each assembly model. Usually, both variation simulation and assembly optimization can be parallelized by tasks (Pogarskaia et al. 2018): each task refers to solving a particular QP problem (see Fig. 3). Thus, parallelization of QP solvers is not needed and is therefore not considered in this paper.

Task parallelization gives a rise to another computational feature. Namely, the effect of the dependence of computation time on input data for different solvers is enhanced. For some numerical methods the number of active constraints at the optimum n_{act} affects computation time significantly, thus becoming an essential characteristic of optimization problem when computation time is analysed.
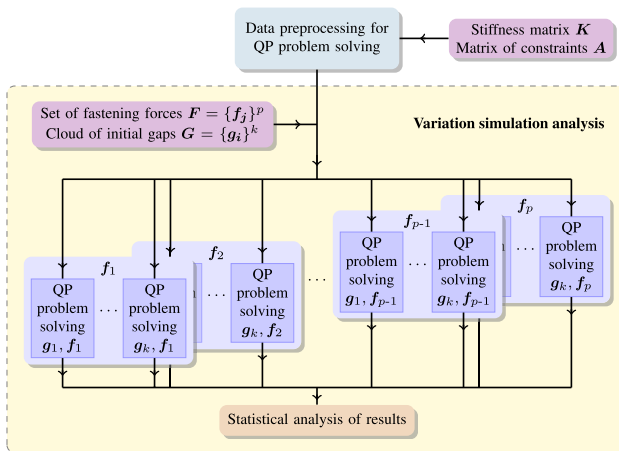

Fig. 3 Task parallelization of computational process for an assembly model

Table 2 The dependence of number of active constraints n_{act} on input data

(a) Different fastening forces and constant initial gap 5mm

Vectors of forces $\boldsymbol{F}$	Percent of fasteners (%)	n_{act} (%)
f_1	5	3.1
f_2	31	21.1
f_3	58	33.3
f_4	72	38.6
f_5	90	47.4
f_6	100	52.3

(b) Different initial gap vector and 50% of installed fasteners

Vectors of gap $\boldsymbol{G}$	Initial gap field	n_{act} (%)
g_1		45.9
g_2		46.1
g_3		44.2
g_4		47.9
g_5		47.4
g_6		46.4

Table 2 presents the dependence of the number of active constraints n_{act} for problem (1) on input data. An increase in the number of installed fasteners leads to an increase in the number of active constraints n_{act}. Variation of the vector of initial gap generally does not affect n_{act} much. Note that in the remainder of the text n_{act} is the number of active constraints for problem (1), (3) and (4) and for problem (2) is equal to $m - n_{act}$.

3 Adaptation of QP methods for solving the contact problem

The paper is devoted to adaptation and comparison of various types of quadratic programming methods applied to the contact problem. The following four widely used quadratic programming methods are considered: the interior-point method, the active set method, Newton projection method, and complementary pivot algorithm.

The interior-point method is a polynomial-time algorithm that has proved its efficiency in practice (Gondzio and Grothey 2006; Byrd et al. 1999; Mehrotra 1992). It allows to solve a wide range of optimization problems such as linear, convex quadratic-programming, second-order cone programming, and semidefinite programming problems (Wright 2005). This method is known to be one of the main methods used for solving large sparse problems and is implemented in many commercial optimization software packages (MOSEK, CPLEX, IMSL, MATLAB and others).

Another class of optimization methods is active set methods. One of the most commonly used active set methods for solving quadratic programming problems was proposed by Goldfarb and Idnani in (1983). It is a dual active set method which

proved to be fast and efficient for small to medium-sized problems (up to ten thousand unknowns), see Burton and Toint (1992); Gaspero et al. (2011); Marron et al. (1997). This paper focuses on the modification made by Powell (1985) that is preferable for ill-conditioned problems (further referred to as ASM). This method is also implemented in some well-known numerical libraries such as IMSL Numerical Library (Rogue Wave Software 2016) and Scilab (2018).

Newton projection method is a second order modification of the gradient projection method, which is an extension of the steepest-descent method for constrained minimization problems. The gradient projection method was proposed by Goldstein (1964), Levitin and Polyak (1966) and Newton projection method was thoroughly developed by Bertsekas (1976, 1982). It is easily implemented and highly effective if constraints have the form $x - g \leq 0$, as the projection operator is just a minimum operator in this case. Newton projection method works well for large-scale problems, but is not popular with researchers due to the fact that its basic version has a low convergence rate.

Convex quadratic problems can be formulated as linear complementarity problems. One of the most commonly used methods for solving problems of this type is complementary pivot algorithm (CPA) or Lemke's method, introduced by Lemke (1968). Lemke's method is widely used because of such features as finiteness, reliability and efficiency (Cottle et al. 1992; Acary and Brogliato 2008). Several numerical packages include implementations of Lemke's method such as Siconos (Acary and Pérignon 2007) or GAMS PATH (Ferris and Munson 2018).

The theoretical estimations of computational complexity do not allow choosing the indisputable leader among these methods for the contact problem solving (see Table 3). Note that Table 3 presents a comparison for the primal formulation of the contact problem (1). For the dual and relative formulations (problem (2) and (3), respectively), the number of blocks p in the stiffness matrix K is 1 and the number of unknowns n is equal to the number of constraints m.

All algorithms have polynomial time on average with a close degree of the polynomial. In particular, computation time depends on the number of active constraints at the optimum n_{act} and the number of conjugate gradient method iterations n_{cg} when iterative method is used for linear system solving. Typically, the number of active constraints n_{act} can run over $[0, m]$ for an assembly model. It can take small values as well as large ones when solving a set of similar contact problems as n_{act} is defined by the vector of applied forces f and the vector of initial gap g (Table 2). Number of iterations n_{cg} depends on the condition number of the stiffness matrix K and the quality of the applied preconditioner. Typically, the condition number is $10^5 - 10^8$ for the considered type of contact problems.

While solving a set of similar contact problems, some of the time-consuming operations can be performed once. These operations include the Cholesky decomposition of the stiffness matrix K, the inversion of the Cholesky matrix, and the reordering of constraints and stiffness matrices. The time required for such data preprocessing is presented in the third column of Table 3. It differs for particular algorithms depending on required operations. The details of implementation and adaptation to the features of contact problem are discussed in the following subsections.

Table 3 Comparison of algorithms

	Computation time	Required memory	Time for data preparation
Primal-dual interior-point method	$O(n^{2.5} n_{cg} \ln(\frac{1}{\epsilon_\mu}))$	$O(\sum_{i=1}^{p} n_i^2)$	$O(n^3)$
Active set method	$O(n^2 n_{act})$	$O(n^2)$	$O(n^3)$
Newton projection method	$O((n - n_{act})^4)$	$O(n^2)$	$O(n^2)$
Complementarity pivot algorithm	$O((n + m)^3)$	$O(\sum_{i=1}^{p} n_i^2)$	$O(n^3)$

n_{act}, number of active constraints at the optimum ($0 \le n_{act} \le m$); n_{cg}, number of conjugate gradient method iterations; p, number of blocks in the stiffness matrix K; n_i, number of unknowns of i-th part; ϵ_μ, threshold related to desired accuracy

3.1 Interior-point method

The primal-dual interior-point method (IPM) reformulates the problem of constrained minimization (1) as a sequence of unconstrained minimization problems using a logarithmic barrier:

$$\min \ \tfrac{1}{2} x^T K x - f^T x - \mu \sum_{j=1}^{m} \ln(-a_j^T x + g_j), \tag{6}$$

where μ is the barrier parameter and a_j is the j-th column of A. The solution of problem (6) converges to the solution of problem (1) when μ goes to zero. At each iteration IPM solves a system of nonlinear equations (7) that represents the first order optimality conditions for problem (6):

$$S_\mu(x, y, \lambda) = \begin{pmatrix} Kx - f + A\lambda \\ A^T x - g + y \\ \Lambda Y e - \mu e \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \ y \ge 0, \ \lambda \ge 0. \tag{7}$$

In the system above, Λ and Y are diagonal matrices with λ and $y = -A^T x + g$ on their diagonals and $e = 1_m$ is a vector of ones. One step of Newton's method is used to find an approximate solution of system of linear equations (7) and then the duality gap μ is updated. The process is repeated until μ becomes sufficiently small.

There are two challenging issues related to the implementation of the interior-point method when solving the contact problem: the selection of a starting point and the solution of the linear system for determining the Newton direction. There are two variants of the algorithm, namely feasible and infeasible, which call for the fulfilment of different conditions on the starting point. Feasible IPM requires starting points that satisfy the following conditions:

$$Kx - f + A\lambda = 0, \tag{8}$$

$$A^T x - g + y = 0, \tag{9}$$

$$y \geq 0, \lambda \geq 0, \tag{10}$$

whereas for infeasible IPM, a starting point needs only satisfy the inequality conditions (10). So, for the infeasible IPM the starting point search process is simplified considerably. Note, that the theoretical worst case complexity is better for the feasible IPM, although the infeasible IPM is known to have practically more efficient implementations (Gondzio and Terlaky 1995). The choice of starting point for solving contact problem is discussed in Stefanova et al. (2018), where the starting point for feasible IPM is determined based on the physical interpretation of the QP problem (1), and is compared to the infeasible one with the starting point from D'Apuzzo et al. (2010). When applying IPM to assembly problems, a feasible IPM is preferred for assembly models with junction area of simple type, whereas for more complicated sandwich type joints the algorithm described in Stefanova et al. (2018) does not have direct physical justification; an infeasible IPM yields better results in this case (see Table 4).

The most time consuming IPM step is the solution of the linear system of equations for determining the Newton direction:

$$\nabla S_\mu(\boldsymbol{x}, \boldsymbol{y}, \lambda) = \begin{pmatrix} \boldsymbol{K} & \boldsymbol{0} & \boldsymbol{A} \\ \boldsymbol{A}^T & \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\Lambda} & \boldsymbol{Y} \end{pmatrix} \begin{pmatrix} \Delta\boldsymbol{x} \\ \Delta\boldsymbol{y} \\ \Delta\lambda \end{pmatrix} = \begin{pmatrix} \boldsymbol{s}_1 \\ \boldsymbol{s}_2 \\ \boldsymbol{s}_3 \end{pmatrix}, \tag{11}$$

Table 4 Number of IPM iterations for different starting point approaches

Simple type assembly		
Dual (2)		
n_{act}	Feasible	Infeasible
$n = 6112, m = 3056$		
333	22	24
479	21	25
591	22	25
780	21	25
1017	21	25
1353	20	24
Sandwich type assembly		
Dual (2)		
n_{act}	Feasible	Infeasible
$n = 4268, m = 2716$		
128	42	27
272	38	27
543	39	28
876	42	27
1103	38	27
2004	38	25

where I is the identity matrix, $s_1 = -(Kx - f + A\lambda)$, $s_2 = -(A^T x - g + y)$, $s_3 = -(\Lambda Ye - \sigma \mu e)$, and σ is the centering parameter chosen adaptively (Mehrotra 1992). A challenging numerical issue is the increase of condition number as the optimal solution is approached. To solve the system of equations (11), one can use direct or iterative approaches. Direct methods are based on decomposition techniques (Gondzio and Grothey 2006) and can be efficient if the problem is sparse or parallelization is possible. When the problem is large-scale, an iterative approach with the use of an appropriate preconditioner is preferable. Several different approaches for a set of contact problems arising in assembly simulation are discussed further and a preconditioner attractive for the considered type of problems is proposed.

The system of equations (11) is first transformed to an augmented system (12) and then to a normal system (13):

$$\begin{pmatrix} K & A \\ A^T & -\Lambda^{-1}Y \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta \lambda \end{pmatrix} = \begin{pmatrix} s_1 \\ s_2 - \Lambda^{-1}s_3 \end{pmatrix}, \tag{12}$$

$$H\Delta\lambda = p, \tag{13}$$

where $H = A^T K^{-1} A + \Lambda^{-1} Y$, $p = A^T K^{-1} s_1 + \Lambda^{-1} s_3 - s_2$, Δx and Δy can be computed using the relations $\Delta x = K^{-1}(s_1 - A\Delta\lambda)$, $\Delta y = \Lambda^{-1}(s_3 - Y\lambda)$. The augmented system is symmetric indefinite. For the class of problems considered in the paper, solving the system of equations (12) turned out to be significantly less efficient compared to solving the normal system, therefore it is not considered in the paper. The normal system has smaller dimension; it is symmetric, positive definite and fully populated. The system of equations (13) is solved directly using Cholesky decomposition and iteratively using preconditioned conjugate gradient method (CG).

Three preconditioners are considered and compared in this paper. The first one is a Jacobi preconditioner $P_1 = \text{diag}(H)$ that is based on the diagonal elements of matrix H. The second one is an incomplete Cholesky preconditioner $P_2 = L_H L_H^T$. The incomplete Cholesky factorisation is applied to the matrix $H_r = H_s + \alpha I$, where α is a regularization parameter and H_s is a sparse approximation of matrix H. Paper (Lin and Saigal 2000) presents a brief review of several existing approaches to define sparsity pattern for dense matrices based on dropping elements with small magnitude from H. The incomplete Cholesky preconditioner uses more information about matrix structure compared to P_1; however its efficiency depends on the parameters such as the number of non-zeros in H_s and the value of regularization parameter α. While the number of CG iterations is reduced with the increase of the amount of non-zeros in H_s, the time required to build the preconditioner increases. By varying the number of non-zeros, the preconditioner with 50% of the largest elements of the original matrix H (and all other elements substituted by zeros) was chosen as the fastest. The regularization parameter α is chosen adaptively starting from $\alpha = 0$; the regularization parameter is increased if the incomplete Cholesky factorization fails.

The third preconditioner proposed for assembly problems (Stefanova et al. 2018) is a split preconditioner $P_3 = (L_Q + \sqrt{\Lambda^{-1}Y})(L_Q + \sqrt{\Lambda^{-1}Y})^T$, where L_Q is the Cholesky decomposition of the matrix $Q = A^T K^{-1} A$. The strong point of the

proposed preconditioner $\boldsymbol{P}_3$ is the combination of the possibility to be updated quickly at each IPM iteration with a good approximation of $\boldsymbol{H}$. The matrix $\boldsymbol{L_Q}$ does not change from iteration to iteration, so only the diagonal terms $\sqrt{\boldsymbol{\Lambda}^{-1}\boldsymbol{Y}}$ must be modified to update the preconditioner. Moreover, the Cholesky decomposition $\boldsymbol{L_Q}$ can be performed once per assembly model.

All of the described approaches are compared in Table 5. The table presents the time necessary to solve contact problem. The computations were done on an Intel Core i5 CPU 3.30GHz computer with 16 GB of RAM. Two sets of problems with different vectors of forces $\boldsymbol{f}$ are considered for wing assembly models with 4386 and 6112 variables. The computational results show that the iterative approach with preconditioner $\boldsymbol{P}_3$ is considerably faster.

Thus, to solve the system of linear equations the adapted IPM uses CG method with the combined preconditioner $\boldsymbol{P}_3$ and an appropriate starting point is chosen depending on the assembly model.

3.2 Active set method

The main goal of active set methods is to find the set of constraints that are active at the solution point:

$$\mathrm{I}_{opt} = \left\{ i = 1, 2, \ldots, n \mid \left(\boldsymbol{A}^T \boldsymbol{x}_{opt} - \boldsymbol{g} \right)_i = 0 \right\}.$$

An initial approximation of the active set I_{act}^0 is chosen and then one constraint is added or removed at each iteration forming the current active set I_{act}^k. Since the number of constraints is finite, iteration process finishes after finite number of steps (Boland 1996).

Table 5 Computation time (in seconds) of IPM with direct and iterative approaches for solving system of equations (13)

Dual (2)				
n_{act}	Direct	Iterative P_1	Iterative P_2	Iterative P_3
$n = 4386, m = 2193$				
20	663	55	217	48
378	736	126	444	59
695	771	210	489	68
1830	502	279	597	43
2193	502	337	532	14
$n = 6112, m = 3056$				
333	2119	929	2830	496
479	2017	1050	3068	472
591	2119	1241	3530	501
780	2025	1316	3525	440
1017	2031	1446	3700	412
1353	1929	1541	3524	361

The dual Goldfarb-Idnani active set method, considered in this paper, starts at the point of the unconstrained minimum of problem (1), i.e., $x_0 = K^{-1}f$, $I_{act}^0 = \emptyset$.

If the non-penetration conditions are violated, the iteration procedure starts. At each step, current point x_k is forced to satisfy one of the violated constraints in such a way that $(A^T x_k - g)_{i^k} = 0$, and the objective function of problem (1) is minimized over the set $I_{act}^k \cup \{i^k\}$. The process continues until the point x_k is in the admissible set, meaning that the solution is found.

In the case that it is not possible to satisfy any constraint during the iteration step, the rollback is performed: the point is shifted in such a way that one of the current active constraints is violated again.

It is worth mentioning that the iteration step is made in both primary and dual spaces (not only the point is shifted but also the Lagrange multipliers).

In order to speed up the computations several modifications of the described algorithm are proposed that make use of the input data structure given in the paragraph 2 and take the algorithm specifics into account.

1. Storing only the non-zero blocks of the matrix K allows the reduction of the time necessary for the calculation of the objective function and its gradient, because the number of multiplications is decreased from $(n \cdot p)^2$ to $p \cdot n^2$, where p is the number of blocks and n is the number of variables in the block.
2. At each iteration step, the next constraint to be satisfied is chosen. In the original version of algorithm, the first most violated constraint is taken into consideration: $i_{next} = \arg\max_{i=\overline{1,m}}(a_i^T x - g_i)$. A more complicated procedure is introduced for the described algorithm relying on the fact that if a force is applied to a computational node, it means that a fastening element is installed. The gap is then assumed to be closed, and the corresponding constraint must be added to the current active set.
3. As it was mentioned earlier, matrix A^T is sparse: each row contains either 1 or 2 elements. Therefore, it is not necessary to process the entire matrix, but only the indices of the non-zero elements, which leads to the decrease of the number of multiplications by a factor of 10: multiplication of matrix A^T by a vector takes $O(n)$ operations.

The computation time after the implementation of all the proposed modifications is given in Table 6 for a test problem with 3916 variables. The computations were done on an Intel Core i7 CPU 4.00GHz computer with 32 GB of RAM.

Another important feature that can be implemented with the help of the active set method is a "warm-start" technology (Goswami et al. 2012). When a series of similar problems is solved, it is reasonable to start not from the point of the unconstrained minimum but instead from the solution obtained for the previous problem. Thus the iterations that were already done can be skipped and the speed of the computations is significantly increased.

Table 7 illustrates how the computation time changes depending on different starting points.

Table 6 Computation time (in seconds) of ASM

Primal (1)

n_{act}	Original ASM	Modif. 1	Modif. 1 and 2	Modif. 1, 2 and 3
$n = 3916, m = 2504$				
19	181	55	54	48
834	332	210	202	108
1352	371	257	246	131
1987	438	283	269	145

Table 7 Computation time (in seconds) of "warm-start" by modified ASM given different starting points

Primal (1)

n_{act}	Number of active constraints in the previous problem used for "warm-start"			
	n = 0	n = 668	n = 673	n = 690
668	74	–	–	–
673	74	25	–	–
690	76	37	14	–
692	79	47	21	16

Therefore, the Goldfarb-Idnani active set method can be easily adapted to the peculiarities of the considered problem. Implementation of the modifications listed above allows to reduce computation time drastically. The technology of "warm-start" can be efficiently applied when optimizing the assembly.

3.3 Newton projection method

Newton projection method (NPM) is an iterative technique based on gradient descent for solving constrained optimization problems. Each iteration consists of a step in a search direction and a projection of the new point to the feasible region. The process continues until the gradient projection is close enough to zero. An iteration of NPM is defined as

$$x^{(k+1)} = P\big(x^{(k)} - \alpha_k d^{(k)}\big), \tag{14}$$

where $d^{(k)}$ is the search direction, α_k is the step size, P(x) is the projection operator.

Constraints with an identity matrix are considered:

$$A = I, \quad x - g \leq 0.$$

Relative and dual problems satisfy this condition as well as primal problem (4). The projection operator is simple:

$$\mathrm{P}_i(\boldsymbol{x}) = \min(x_i, g_i).$$

The method for choosing the direction $\boldsymbol{d}^{(k)}$ with quadratic rate of convergence was introduced in Bertsekas (1982). It is based on Newton's method. For the algorithm specification the set of indices is introduced

$$\mathrm{I}^-(\boldsymbol{x}^{(k)}) = \left\{ i \in \overline{1,n} \mid x_i = g_i, \ \frac{\partial F(\boldsymbol{x}^{(k)})}{\partial x_i} < 0 \right\},$$

where

$$F(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T \boldsymbol{K}\boldsymbol{x} - \boldsymbol{f}^T\boldsymbol{x}.$$

The set $\mathrm{I}^-(\boldsymbol{x}^{(k)})$ includes such indices i that x_i would not change if the search was made along the antigradient. These indices are excluded from the search and the Newton's method is used in the subspace defined by the remaining indices to reach the quadratic rate of convergence. Thus, the search direction $\boldsymbol{d}^{(k)}$ can be defined from

$$\boldsymbol{K}_{\mathrm{P}}^{(k)}\boldsymbol{d}^{(k)} = \nabla F(\boldsymbol{x}^{(k)}), \tag{15}$$

where

$$\nabla F(\boldsymbol{x}) = \boldsymbol{K}\boldsymbol{x} - \boldsymbol{f},$$
$$(\boldsymbol{K}_{\mathrm{P}}^{(k)})_{ij} = \begin{cases} K_{ij}, & \text{if } i = j \text{ or both } i,j \notin \mathrm{I}^-(\boldsymbol{x}^{(k)}), \\ 0, & \text{otherwise.} \end{cases}$$

Initial estimate $\boldsymbol{x}^{(0)}$ can be set equal to $\boldsymbol{g}$ in order to make the set $\mathrm{I}^-(\boldsymbol{x}^{(0)})$ not empty and reduce the dimension of $\boldsymbol{K}_{\mathrm{P}}^{(0)}$. Solving the Eq. (15) is a computationally expensive part of NPM. Thus, choosing an effective method for solving (15) is a key point in NPM adaptation to assembly problems. Notice that it is possible to avoid storing of zero rows and columns of matrix $\boldsymbol{K}_{\mathrm{P}}^{(k)}$. Thus, the dimension of the system of equations is reduced by $n_{act}^{(k)} = \left| \mathrm{I}^-(\boldsymbol{x}^{(k)}) \right|$. Three approaches based on Cholesky decomposition, conjugate gradient (CG) method and their combination are compared.

First, Cholesky factorization method is considered. Since every Eq. (15) is a reduction of the full system where some rows and columns are disposed, it is suggested to not recalculate the factorization at each iteration but instead update it from the previous iteration using relations presented in Stewart (1998) and Osborne (2010). This method can significantly reduce the time needed for solving the system of equations (15). The system of equations (15) can also be solved with iterative CG method (van der Vorst 2003). Another proposed idea is to use a combined approach, i.e., to use the CG method with low accuracy at the first iterations of NPM, then increase the accuracy during later iterations, and switch to the Cholesky decomposition when the CG method starts slowing down.

The comparison of the described approaches is presented in Table 8. The computations were done on an Intel Core i5 CPU 3.30GHz computer with 16 GB of RAM. Dimension of the Eq. (15) is $n - n_{act}^{(k)} \approx n - n_{act}$. Thus, the total time for obtaining

Table 8 Computation time (in seconds) of NPM for different methods for solving the system of equations (15)

n_{act}	Dual (2)				Relative (3)			
	Cholesky decomp.	Cholesky update	CG	Comb.	Cholesky decomp.	Cholesky update	CG	Comb.
$n = 6190, m = 3001$								
254	15	15	52	16	155	62	57	43
449	28	27	76	20	135	50	39	28
686	34	29	186	27	77	37	17	16
841	42	36	171	36	64	30	14	13
963	60	48	234	37	58	25	12	12
1105	61	50	165	44	47	20	11	9.7
$n = 5206, m = 2603$								
82	1.7	1.7	1.1	0.98	236	25	99	18
550	11	9.9	5.5	5.7	57	20	16	16
868	15	12	7.3	8.0	36	9.2	9.8	7.1
1006	25	19	8.8	15	28	6.2	7.7	5.6
1235	26	21	10	16	7.7	2.6	1.5	1.9
1362	31	21	11	23	3.2	1.8	0.53	1.1

the solution is $O\big((n - n_{act})^3\big)$. Cholesky decomposition update reduces NPM solution time, especially for the relative problem. CG method can work either faster or slower than Cholesky update depending on the matrix. Combined method commonly works faster than both of them.

One of the challenges of NPM implementation is the step size selection. There are several common methods for this problem. The first method is a constant step method (Goldstein 1964; Levitin and Polyak 1966), which requires knowledge of the Lipchitz constant and has a poor convergence rate. The second one is Armijo step size rule (Bertsekas 1982). The third method is the one-dimensional functional minimization (McCormick and Tapia 1972):

$$\min_{\alpha \in (0,1]} F(P(x - \alpha d)),$$

which can be performed by some one-dimensional minimization algorithm. The golden section search method can be used for the minimization. Practically, the minimum is usually either 1 or close to 0, so the golden section method is used for the logarithm of α in order to reach values close to zero in less iterations. Another approach is to use the projected line search (PLS) (Šantin et al. 2016), which is actually a piecewise quadratic minimization. The comparison between the Armijo rule the golden section method and PLS is presented in Table 9. PLS works faster than golden section method and much faster than the Armijo step size rule.

The comparison between the primal, relative and dual problems is presented in Table 10. The number of iterations of NPM for primal and relative problems decreases linearly with respect to n_{act} and has a linear dependence on the problem

Table 9 Computation time (in seconds) of NPM for different methods of step-size selection

Dual (2)				Dual (2)			
n_{act}	Armijo	Golden sec.	PLS	n_{act}	Armijo	Golden sec.	PLS
$n = 6190, m = 3001$				$n = 5206, m = 2603$			
94	119	21	12	82	2	2	1.1
254	152	38	25	550	10	6.7	5.8
449	136	43	27	868	12	9.3	8.3
686	157	56	35	1006	20	16	14
841	244	52	56	1235	17	17	17
1105	247	79	45	1362	33	21	23

Table 10 Computation time (in seconds) of NPM for different problem formulations

	Primal (4)		Relative (3)		Dual (2)	
n_{act}	NPM time	number of iter.	NPM time	number of iter.	NPM time	number of iter.
$n = 6190, m = 3001$						
94	564	36	48	29	10	67
254	510	30	43	22	16	76
449	494	29	23	20	20	69
686	368	24	16	17	27	66
841	315	20	13	16	36	79
963	327	25	12	17	37	64
1105	309	25	9.7	17	44	63
82	317	34	18	34	0.98	19
550	264	19	16	17	5.7	29
868	159	23	7.1	18	8.0	27
1006	108	22	5.6	18	15	33
1235	69	11	1.9	8	16	18
1362	55	7	1.1	7	23	21

size (Šantin and Havlena 2011). Thus, the number of iterations can be estimated as $O(n - n_{act})$ and the time needed to solve the problem as $O((n - n_{act})^4)$. Primal and relative problems are solved faster when the number of active constraints n_{act} is large, while dual problems are solved faster when n_{act} is small. Moreover, relative problems are almost always solved much faster than the primal ones as the size of the relative problem is smaller. Unfortunately, the dual problem has a worse convergence rate than primal and relative problems due to the different physical meaning of the variables and the lack of diagonal dominance of the inverse stiffness matrix for the dual problem.

In the implemented method the combination of Cholesky factorization, Cholesky update and CG method is used for solution of (15) and the golden section method is used for the step selection. The dual problem should be solved if the number of active constraints is relatively small (i.e., a small number of fasteners is applied), and the relative problem should be solved otherwise.

3.4 Complementary pivot algorithm

Lemke's method or complementary pivot algorithm (CPA) is a finite numerical procedure for solving LCPs. The implementation of the CPA is based on the idea of pivots. First, an artificial variable is introduced to the system to replace one of the basic ones. Then, pivots between the variables are performed until the artificial variable is dropped out of the basis. The resulting vector of basic variables is then considered to be the solution.

Convex quadratic programming problems can be reformulated as LCPs (5). The Karush–Kuhn–Tucker conditions (KKT) for the primal (1) and dual (2) problems can be written with an appropriate transformation of variables as follows:

$$
\begin{aligned}
&u = -f + Kx + A\lambda \geq 0, \quad x \geq 0, \ x^T u = 0, \\
&v = g - A^T x \geq 0, \qquad\qquad \lambda \geq 0, \ \lambda^T v = 0.
\end{aligned} \tag{16}
$$

Conditions (16) define an LCP (q, M) where

$$
M = \begin{bmatrix} K & A \\ -A^T & 0 \end{bmatrix} \quad \text{and} \quad q = \begin{bmatrix} -f \\ g \end{bmatrix}. \tag{17}
$$

Another LCP can be constructed from the original quadratic problem if a block pivot is performed on (16) such that

$$
\begin{aligned}
&x = K^{-1}f + K^{-1}u - K^{-1}A\lambda \geq 0, \qquad\qquad x \geq 0, \ x^T u = 0, \\
&v = g - A^T K^{-1} f - A^T K^{-1} u + A^T K^{-1} A\, \lambda \geq 0, \quad \lambda \geq 0, \ \lambda^T v = 0.
\end{aligned} \tag{18}
$$

The modified conditions (18) define an equivalent LCP (q, M):

$$
M = \begin{bmatrix} K^{-1} & -K^{-1}A \\ -A^T K^{-1} & A^T K^{-1} A \end{bmatrix} \quad \text{and} \quad q = \begin{bmatrix} K^{-1}f \\ g - A^T K^{-1} f \end{bmatrix}. \tag{19}
$$

Both (17) and (19) LCPs incorporate primal (1) and dual (2) problems. The dimension of the constructed problems $N = n + m$. The CPA, being a pivoting method, is highly sensitive to the dimension of the problem. Therefore another way of LCP construction is suggested in (Yassine 2008). The quadratic problem without positivity constraints can be transformed into an LCP (q, M) where

$$
M = A^T K^{-1} A \quad \text{and} \quad q = A^T K^{-1} f - g. \tag{20}
$$

This LCP can also be obtained if the dual problem (2) is chosen as a base for KKT conditions and hence this LCP is further referred to as *dual* LCP, while the previous

Table 11 Computation time (in seconds) of the CPA

n_{act}	Primal		Dual	Relative	
	LCP (17)	LCP (19)	LCP (20)	LCP (17)	LCP (19)
$n = 6112, m = 3056$					
38	14,943	1522	88	4260	405
60	19,844	2225	131	5772	601
120	29,943	3540	218	8426	1004
164	37,504	4618	289	10,794	1327
228	50,011	6258	410	15,073	1885

Fig. 4 Parts of wing-to-fuselage assembly for Airbus A350-900. Outer wing box (left) and central wing box (right). Photos courtesy of Airbus SAS

LCPs (LCP (17) and LCP (19)) are referred to as *primal*. The dimension of the dual LCP (20) $N = m$. Since the average computation time for the CPA is $O(N^3)$, the dual LCP has an important advantage over the primal ones.

The relative problem (3) can also be reformulated as an LCP. The basic version of the primal LCP (17) and the modified version (19) for the relative problem have a dimension $N = 2m$, and the dual LCP (20) for the relative problem is the same as for the primal, because the dual quadratic problems for them are identical.

Comparison of computation time of CPA application to primal, dual and relative problems is presented in Table 11. The computations were done on an Intel Core i5 CPU 3.30GHz computer with 16 GB of RAM. Results provided in the table confirm that Lemke's method for the dual LCP (20) works decisively faster than for the relative or primal problems.

4 Computational results for aircraft assembly

In this section, the computation time of the optimization methods is compared for a set of typical aircraft assembly problems. The wing-to-fuselage assembly is chosen for this purpose. The details of wing-to-fuselage assembly simulation are given in Lupuleac et al. (2018, 2019b). The upper and lower outer wing box panels are joined with the corresponding central wing box (CWB) panels (see Fig. 4). The CWB is nested within the central fuselage section.
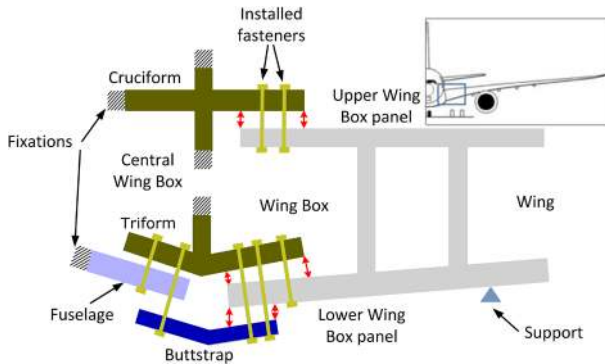
Fig. 5 Schematic outline of wing-to-fuselage junction

A schematic outline of the wing-to-fuselage assembly is presented in Fig. 5. The CWB is designed to be the most rigid part of the aircraft, and therefore the elements of the assembly joined with CWB can be regarded as fixed. The assembly jig supports the wing at several points. The additional parts (buttstraps) are used to ensure the tightness of the joint. The assemblies for the upper and lower wing box panels are performed independently and interinfluence between these processes is minimal. That is why independent models for the upper and lower wing-to-fuselage assemblies are considered (see Fig. 6).

The upper wing-to-fuselage joint consists of just two parts: the cruciform and the upper wing box panel. The cruciform is already fastened to the CWB, so it can be assumed to be fixed along the edges as it is shown in Fig. 5. The model for the upper wing-to-fuselage joint is of simple type and has one junction area between the wing and the cruciform.
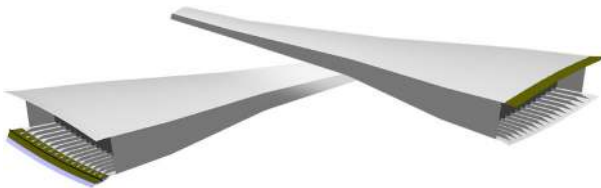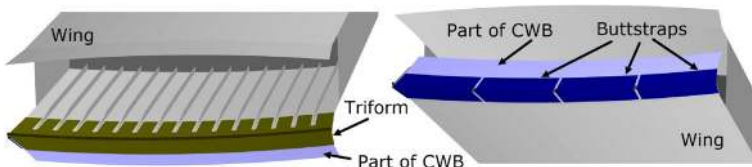

Fig. 6 Lower and upper assembly wing models
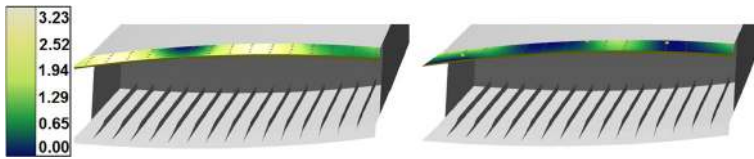

Fig. 7 Lower wing-to-fuselage joint

Fig. 8 Gap in mm before (left) and after (right) installation of the fasteners

Table 12 Test problems description

Model name	Number of parts	Joint type	Number of unknowns	Number of constraints	Condition number of stiffness matrix
UJM1	2	Simple	5206	2603	$2.47 \cdot 10^6$
UJM2	2	Simple	11,308	5654	$1.77 \cdot 10^7$
LJM	7	Sandwich	6190	3001	$1.46 \cdot 10^8$

The lower wing-to-fuselage joint is more complex compared to the upper one and consists of a lower wing panel, a triform, a part of the CWB, and several buttstraps (Fig. 7). The triform and the buttstraps are simultaneously fastened to the wing and to the CWB panel. Hence, the lower wing-to-fuselage assembly model is of sandwich-type where the lower wing panel and the part of CWB are located above the buttstraps and below the triform (Fig. 5). The triform is joined with the vertical CWB panel, and thus it is fixed along the upper edge. The buttstraps are not fixed on the assembly jig before fastening.

An example of computational results for an upper wing assembly model is shown in Fig. 8. The left figure presents the initial gap field between the cruciform and the upper wing box panel. The right figure shows the residual gap after the installation of 3 fasteners.

Three test problems used for comparison of algorithms are described in Table 12. Two models with 5206 and 11308 unknowns correspond to the upper joint model (UJM). The model with the 6190 unknowns refers to the lower joint model (LJM). The condition number of stiffness matrix K is larger for LJM than for UJM due to the features of the assembly jig fixing. The algorithms are implemented in C++ using MSVS 2013 and Visual C++ 12.0 compiler. All computations presented in this section were done on an Intel Core i5 CPU 3.30GHz computer with 16 GB of RAM. As seen in Sect. 3, each algorithm used its own stopping criteria. However, these criteria are chosen so that the difference between the displacements obtained by the two algorithms does not exceed 10e−7 mm.

Figure 9 presents comparison of the optimization methods for UJM and LJM in terms of computation speed. The computation time is shown in logarithmic scale as a function of the number of active constraints. The results are presented only for dual and relative formulations since for the primal formulation the time is much longer due to the size of the problem. Each computation was repeated 10 times and the average value is shown in the Fig. 9. The time deviation varies 4–15% in average.
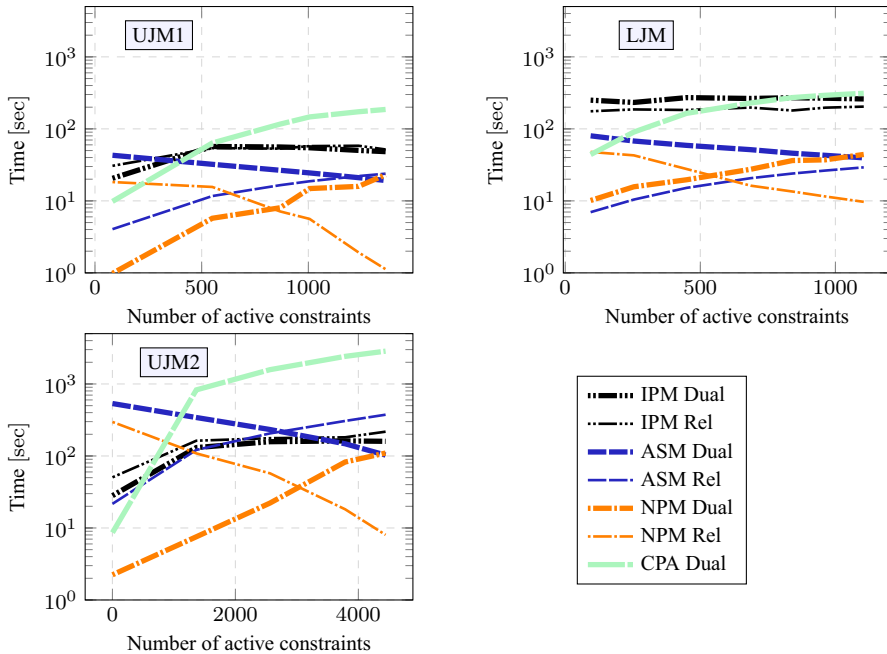
Fig. 9 Comparison of implemented methods. Solid and dashed lines refer to dual and relative problem formulations respectively

Let us first consider the results for UJM1 problem. Computation time of NPM and CPA for the dual problem and ASM for the relative problem increases with the increase in the number of active constraints at the optimum, i.e., in the number of installed fastening elements. Time of NPM for the relative problem and ASM for the dual one has the opposite dependency. IPM computation time does not depend strongly on the number of active constraints. Note that this behaviour agrees with the theoretical complexity presented in Table 3. These basic time trends are also present in the results of UJM2 and LJM problems.

The UJM1 and UJM2 differ mainly in the number of variables n. Thus, its influence is clearly observed in the results. With an increase in the number of variables, the time increases significantly for the most of the methods with the exception of IPM. NPM applied to the dual problem is generaly the fastest in solving the contact problem for upper wing models. However, NPM for the relative problem is less time consuming than for the dual one when the number of active constraints is large.

The UJM1 and LJM have similar number of unknowns, but the condition number for the lower wing model is larger. Comparing these two models, ASM is revealed to be the least affected by the condition number of the stiffness matrix. ASM applied to the relative problem is usually the fastest for LJM with the exception of problems with a large number of active constraints, where NPM for the relative problem is the leader.

Finally, let us consider the computation time distribution for variation simulation analysis of an Airbus A350-900 wing-to-fuselage assembly. The aim of the

Fig. 10 Distribution of number of active constraints for the solution of contact problem with different measured initial gaps and 50% of installed fasteners
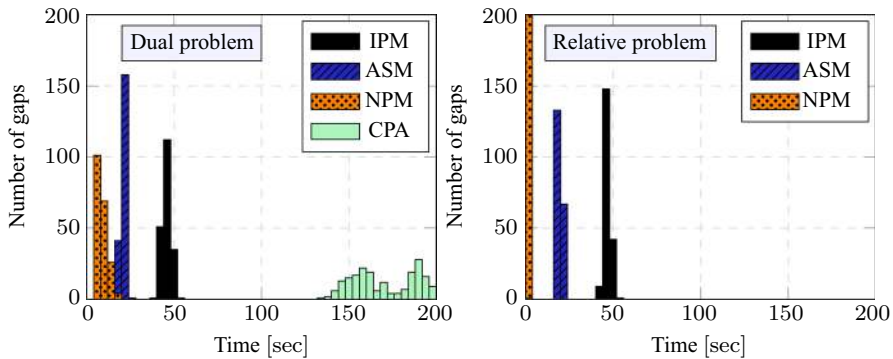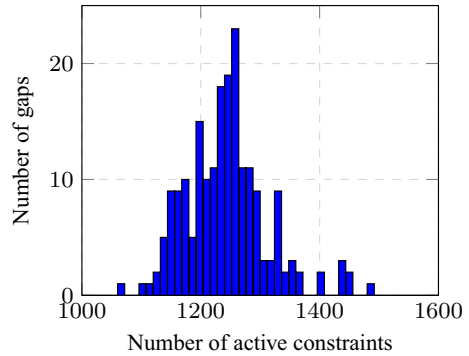

Fig. 11 Time distribution for the solution of contact problem with different measured initial gaps and 50% of installed fasteners

analysis is to verify that a given configuration of fastening elements provides a sufficient fastening level to close the gap between the assembled parts (Lupuleac et al. 2019b). Such an analysis requires to solve the contact problems for a set of measured initial gaps. In total about 200 sets of measurements of aircraft produced in 2013–2016 were used for the analysis. The distribution of the number of active constraints for the measured initial gaps is shown in Fig. 10. UJM1 with 50% of installed fasteners was used for variation simulation analysis, i.e the contact problem was solved with the same stiffness matrix, matrix of constraints and vector of forces for all of the measured initial gaps.

Figure 11 presents computation time distribution for different solvers. The results presented in Fig. 11 agree with those presented in Fig. 9: the order of the bars in the histograms follows that of the curves for the corresponding number of active constraints. Since the computation time of the CPA for relative problems is considerably greater than for dual ones (see Sect. 3.4), it was only applied to the dual formulation of the test problems.

Note that some problems arising in fastener pattern optimization can require massive computations of up to 10^6 runs and thus task parallelization is needed

(Pogarskaia et al. 2018). Therefore, to reduce the total computation time of assembly simulation, not only the *average* computation time for one run has to be reduced, but also the *maximal* computation time. According to the obtained results (see Fig. 11), for both dual and relative problem, NPM is revealed to be the leader both in terms of average and maximal computation time. The CPA works considerably slower than the other implemented methods for the test problems. Moreover, it has the largest deviation between maximal and average time, which creates difficulties for task parallelization. Note that the results in Fig. 11 only represent the variation simulation analysis of one fastener pattern. Different methods can yield the best results for different patterns.

5 Conclusion

The paper presents and discusses computational aspects related to the solution of a certain class of quadratic programming problems related to variation simulation analysis.

The contact problem arising in assembly simulation is reformulated in dual and relative form. Data preprocessing (e.g., matrix transformations) is only performed once for the assembly model. Therefore the time for each of the multiple problem solving as well as time for the complete assembly analysis is reduced, since the variation simulation analysis requires solving of problems with similar data.

The presented numerical experiments demonstrate the following features of the methods. The computation time of the interior point method is not affected by the number of active constraints at the optimum. Thus, this method is suggested for such problems as optimization of fastening pattern for simple joints where the number of active constraints may vary considerably. The Goldfarb-Idnani active set method is revealed to be the best for middle-scaled ill-conditioned problems such as verification analysis of complex joints. Newton projection method is suitable for problems of various types, to the fullest extent for the large-scale ones and for problems with big number of active constraints. The Complementarity pivot algorithm should be preferred for problems with small number of active constraints. For the example of variation simulation analysis of an assembly simulation problem, Newton projection method was revealed to be the leader in terms of average and maximal computation time.

The adaptation of the discussed numerical methods combined with an appropriate modification of the contact problem leads to a considerable reduction of computation time for the described problems.

References

Acary V, Brogliato B (2008) Numerical methods for nonsmooth dynamical systems: applications in mechanics and electronics, vol 35. Springer, Berlin

Acary V, Pérignon F (2007) An introduction to Siconos. Technical Report RT-0340, INRIA

Bertsekas D (1976) On the Goldstein–Levitin–Polyak gradient projection method. IEEE Trans Autom Control 21(2):174–184

Bertsekas D (1982) Projected Newton methods for optimization problems with simple constraints. SIAM J Control Optim 20(2):221–246

Boland NL (1996) A dual-active-set algorithm for positive semi-definite quadratic programming. Math Program 78(1):1–27

Burton D, Toint PL (1992) On an instance of the inverse shortest paths problem. Math Program 53(1–3):45–61

Byrd RH, Hribar ME, Nocedal J (1999) An interior point algorithm for large-scale nonlinear programming. SIAM J Optim 9(4):877–900

Cottle RW, Pang JS, Stone RE (1992) The linear complementarity problem. Academic Press, San Diego

Dahlström S, Lindkvist L (2007) Variation simulation of sheet metal assemblies using the method of influence coefficients with contact modeling. J Manuf Sci Eng 129(3):615–622

D'Apuzzo M, De Simone V, di Serafino D (2010) Starting-point strategies for an infeasible potential reduction method. Optim Lett 4(1):131–146

Ferris MC, Munson TS (2018) GAMS documentation 25.1: PATH 4.7

Galin LA (1961) Contact problems in the theory of elasticity. North Carolina State College, Raleigh

Gaspero LD, Tollo GD, Roli A, Schaerf A (2011) Hybrid metaheuristics for constrained portfolio selection problems. Quant Financ 11(10):1473–1487

Goldfarb D, Idnani A (1983) A numerically stable dual method for solving strictly convex quadratic programs. Math Program 27(1):1–33

Goldstein AA (1964) Convex programming in Hilbert space. Bull Am Math Soc 70(5):709–710

Gondzio J, Grothey A (2006) Direct solution of linear systems of size 109 arising in optimization with interior point methods. In: Wyrzykowski R, Dongarra J, Meyer N, Waśniewski J (eds) Parallel processing and applied mathematics. Springer, Berlin, pp 513–525

Gondzio J, Terlaky T (1995) A computational view of interior point methods. In: Beasley J (ed) Advances in linear and integer programming. Oxford University Press, Oxford, pp 103–144

Goswami N, Mondal SK, Paruya S (2012) A comparative study of dual active-set and primal-dual interior-point method. IFAC Proc Vol 45(15):620–625

Guyan RJ (1965) Reduction of stiffness and mass matrix. J Math Ind 3:380

Kinderlehrer D, Stampacchia G (1980) An introduction to variational inequalities and their applications, vol 31. SIAM, Philadelphia

Lemke CE (1968) On complementary pivot theory. Math Decis Sci 1:95–114

Levitin ES, Polyak BT (1966) Constrained minimization problems. USSR Comput Math Math Phys 6:1–50 **(Eng. transl. of paper in Zh. Vychisl. Mat. i Mat. Fiz (1965) 6:787–823)**

Lin CJ, Saigal R (2000) An incomplete Cholesky factorization for dense symmetric positive definite matrices. BIT Numer Math 40(3):536–558

Lindau B, Lorin S, Lindkvist L, Söderberg R (2016) Efficient contact modeling in nonrigid variation simulation. J Comput Inf Sci Eng 16(1):011002

Lions JL, Stampacchia G (1967) Variational inequalities. Commun Pure Appl Math 20:493–519

Liu SC, Hu SJ (1997) Variation simulation for deformable sheet metal assemblies using finite element methods. J Manuf Sci Eng 119(3):368–374

Lupuleac S, Kovtun M, Rodionova O, Marguet B (2010) Assembly simulation of riveting process. SAE Int J Aerosp 2(2009–01–3215):193–198

Lupuleac S, Petukhova M, Shinder Y, Bretagnol B (2011) Methodology for solving contact problem during riveting process. SAE Int J Aerosp 4(2011–01–2582):952–957

Lupuleac S, Shinder Y, Petukhova M, Yakunin S, Smirnov A, Bondarenko D (2013) Development of numerical methods for simulation of airframe assembly process. SAE Int J Aerosp 6(1):101–105. https://doi.org/10.4271/2013-01-2093

Lupuleac S, Zaitseva N, Stefanova M, Berezin S, Shinder J, Petukhova M, Bonhomme E (2018) Simulation and optimization of airframe assembly process. In: ASME 2018 international mechanical engineering congress and exposition. American Society of Mechanical Engineers, v002T02A109

Lupuleac S, Smirnov A, Churilova M, Shinder J, Zaitseva N, Bonhomme E (2019a) Simulation of body force impact on the assembly process of aircraft parts. In: ASME 2019 international mechanical engineering congress and exposition. American Society of Mechanical Engineers

Lupuleac S, Zaitseva N, Stefanova M, Berezin S, Shinder J, Petukhova M, Bonhomme E (2019b) Simulation of the wing-to-fuselage assembly process. J Manuf Sci Eng 10(1115/1):4043365

Marron J, Turlach B, Wand M (1997) Local polynomial smoothing under qualitative constraints. Comput Sci Stat 28:647–652

McCormick G, Tapia R (1972) The gradient projection method under mild differentiability conditions. SIAM J Control 10(1):93–98

Mehrotra S (1992) On the implementation of a primal-dual interior point method. SIAM J Optim 2(4):575–601

Osborne M (2010) Bayesian Gaussian processes for sequential prediction, optimisation and quadrature. PhD thesis, University of Oxford

Petukhova MV, Lupuleac SV, Shinder YK, Smirnov AB, Yakunin SA, Bretagnol B (2014) Numerical approach for airframe assembly simulation. J Math Ind 4:8

Pogarskaia T, Churilova M, Petukhova M, Petukhov E (2018) Simulation and optimization of aircraft assembly process using supercomputer technologies. In: RuSCDays 2018 communications in computer and information science, vol 965. Springer, Berlin

Powell M (1985) On the quadratic programming algorithm of Goldfarb and Idnani. Mathematical programming essays in honor of George B. Dantzig part II. Springer, pp 46–61

Rogue Wave Software (2016) IMSL C Library 8.6.0 User Guide, volume 1—Math Library

Šantin O, Havlena V (2011) Combined gradient and Newton projection quadratic programming solver for MPC. IFAC Proc Vol 44(1):5567–5572

Šantin O, Jarošová M, Havlena V, Dostál Z (2016) Proportioning with second-order information for model predictive control. Optim Methods Softw 32(3):436–454. https://doi.org/10.1080/10556788.2016.1213840

Scilab (2018) Scilab 6.0.1 help, optimization and simulation

Stefanova M, Yakunin S, Petukhova M, Lupuleac S, Kokkolaras M (2018) An interior-point method-based solver for simulation of aircraft parts riveting. Eng Optim 50(5):781–796

Stewart GW (1998) Matrix algorithms: basic decompositions, vol i. Society for Industrial and Applied Mathematics, Philadelphia

Tu YO, Gazis D (1964) Gazis the contact problem of a plate pressed between two spheres. J Math Ind 31:659–666

Turner MJ, Topp LJ, Martin HC, Clough RW (1956) Stiffness and deflection analysis of complex structures. J Math Ind 23:805–823

van der Vorst HA (2003) Iterative Krylov methods for large linear systems. Cambridge University Press, Cambridge

Wriggers P (2006) Computational contact mechanics, 2nd edn. Springer, Berlin

Wright M (2005) The interior-point revolution in optimization: history, recent developments, and lasting consequences. Bull Am Math Soc 42(1):39–56

Yang D, Qu W, Ke Y (2016) Evaluation of residual clearance after pre-joining and pre-joining scheme optimization in aircraft panel assembly. Assem Autom 36(4):376–387

Yassine A (2008) Comparative study between Lemke's method and the interior point method for the monotone linear complementary problem. Studia Univ Babes-Bolyai Math 53:119–132

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.