Structural Engineering and Mechanics, *Vol. 62, No. 3 (2017) 297-302*
DOI: https://doi.org/10.12989/sem.2017.62.3.297

The Cholesky rank-one update/downdate algorithm for static reanalysis with modifications of support constraints

Haifeng Liu[*1], Jihua Zhu[2a] and Mingming Li[3b]

[1]School of Mathematics and Statistics, Xi'an Jiaotong University, Xi'an, 710049, P.R. China
[2]School of Software Engineering, Xi'an Jiaotong University, Xi'an, 710049, P.R. China
[3]Network Center, Jilin University, Changchun 130025, P.R. China

(Received April 22, 2016, Revised December 22, 2016, Accepted December 22, 2016)

Abstract. Structural reanalysis is frequently utilized to reduce the computational cost so that the process of design or optimization can be accelerated. The supports can be regarded as the design variables and may be modified in various types of structural optimization problems. The location, number, and type of supports can make a great impact on the performance of the structure. This paper presents a unified method for structural static reanalysis with imposition or relaxation of some support constraints. The information from the initial analysis has been fully utilized and the computational time can be significantly reduced. Numerical examples are used to validate the effectiveness of the proposed method.

Keywords: structural reanalysis; stiffness matrix; support constraints; Cholesky factorization; computational cost

1. Introduction

The design or optimization of a large-scale structure often involves numerous modifications, and each modification requires a fresh analysis for stresses and displacements. Large amounts of time will be spent in these numerous repeated analyses. This results in extensive studies on structural static reanalysis. Static reanalysis is to calculate structural responses after modifications by utilizing the original information as much as possible so that the computational cost can be greatly reduced (Abu Kassim and Topping 1987). Static reanalysis techniques are significant for designing large structures, especially for the case in which only a small part of the structure is modified step by step (Hassan *et al*. 2010, Terdalkar and Rencis 2006).

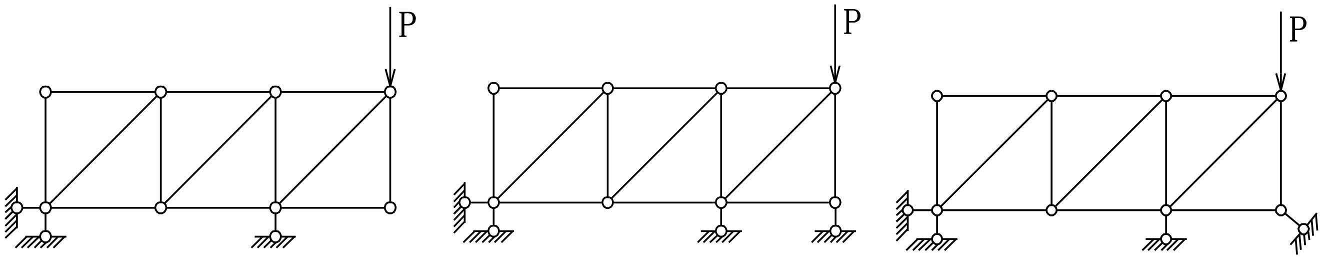
Many static reanalysis methods have already been proposed. These methods can be roughly divided into the following two categories: direct methods and approximate methods. Direct methods present exact closed-form solutions of the response of the modified structure. The computational costs of these methods are closely related to the number of the modified elements, and are unrelated to the extent of the change. Thus, this kind of methods is efficient for the cases in which the changes in design variables are large in magnitude, yet only affect a relative small part of the structure. Approximate methods provide approximate solutions, and the computational costs of these methods depend on the number of the modified elements and the extent of the change. Generally speaking, approximate methods can further be classified into the following four classes: local approximations, global approximations, combined approximations (CA), and preconditioned conjugate gradient (PCG) approximations. For the detailed derivation of these methods, we refer the readers to Kirsch (2008).

Generally, the design variables used in structural design or optimization can be classified as follows (Olhoff and Taylor 1983): cross-sectional design variables, material design variables, geometrical design variables, topological design variables, shape design variables, and support design variables. The structural modifications corresponding to the modification of the above design variables are named cross-sectional modifications, geometrical modifications, topological modifications, layout modifications and supporting modifications, respectively. Among these design variables, support design variables become increasingly widely used (Takezawa *et al.* 2006, Zhu and Zhang 2010, Kozikowska 2011, Xia *et al.* 2014, Gao *et al.* 2015, Xia and Shi 2016) and several static reanalysis methods were presented for this kind of modifications in recent years. Liu *et al.* (2012), Liu and Yue (2014) studied the cases of adding and deleting some support constraints whose orientations are the same as that of some axes of the global coordinate system, respectively. Liu *et al.* (2014) proposed a unified method that can deal with adding and deleting of supports with arbitrary orientation. The method is based on the rank-one decomposition of the element stiffness matrix and the computational cost is closely related to the number and the types of the involved elements. This paper is a follow up to Liu *et al.* (2012), Liu and Yue (2014). A unified method that can deal with adding and deleting of supports whose orientations are the same as that of some axes of the global coordinate system is presented. The Cholesky rank-one update/downdate algorithm is utilized. The algorithm

[*]Corresponding author, Ph.D.
E-mail: lhflkcc@mail.xjtu.edu.cn
[a]Ph.D.
[b]M.Sc.

Copyright © 2017 Techno-Press, Ltd.

ISSN: 1225-4568 (Print), 1598-6217 (Online)



(a) (b) (c)

Fig. 1 The truss structures. (a) initial design, (b) modified design with added one support along vertical axis of the global coordinate system, (c) modified design with added one skew support

can update the Cholesky factorization for a matrix following a low rank modification with little cost. It has been used in many fields such as power systems, statistics and optimization (Davis 2006). The remainder of the paper is organized as follows. The considered problem is formulated in Section 2 and our method is derived in Section 3. Numerical examples are employed to validate the effectiveness of the proposed method in Section 4. Finally, some conclusions are drawn in Section 5.

2. Problem formulation

The static reanalysis problem with added support constraints where some node displacements along axes of the global coordinate system are specified can be stated as follows. Given an initial design, the corresponding stiffness matrix is $\mathbf{K}_0 \in R^{m\times m}$. The displacements vector $\mathbf{x}_0$ can be obtained by solving the following equilibrium equations

$$\mathbf{K}_0\mathbf{x}_0 = \mathbf{R}_0 \tag{1}$$

where $\mathbf{R}_0$ denotes the load vector. The matrix $\mathbf{K}_0$ is symmetric and positive definite (SPD). From the initial analysis, the Cholesky factorization of $\mathbf{K}_0$ has already been known

$$\mathbf{K}_0 = \mathbf{L}_0\mathbf{D}_0\mathbf{L}_0^{\mathrm{T}} \tag{2}$$

where $\mathbf{L}_0$ is a unit lower triangular matrix, $\mathbf{D}_0$ is a diagonal matrix and $\mathbf{L}_0^{\mathrm{T}}$ denotes the transpose of $\mathbf{L}_0$.

Suppose the structure is modified by increasing some support constraints so that k node displacements along axes of the global coordinate system are specified. The truss structures shown in Fig. 1 are employed to illustrate the scope of the proposed method. The initial structure is in Fig. 1(a), case of adding support constraint like Fig. 1(b) is studied in this paper, while case of adding skew support constraint like Fig. 1(c) is out of our scope.

The equilibrium equation corresponding to the modified structure is

$$\mathbf{K}\mathbf{x} = \mathbf{R} \tag{3}$$

where $\mathbf{K}_0 \in R^{(m-k)\times(m-k)}$ is the modified stiffness matrix and is also SPD, $\mathbf{x}$ denotes the displacements vector and $\mathbf{R}$ represents the load vector of the modified structure. In most cases, the number of the changed support constraints is very small compared with the number of the original degrees of freedom (DOFs), i.e., $k<<m$. The purpose is to calculate the displacements vector $\mathbf{x}$ by utilizing the original information as much as possible so that the computational cost can be significantly reduced.

In order to illustrate our method, the relationship between $\mathbf{K}_0$ and $\mathbf{K}$ is presented. Let

$$\mathbf{K}_0 = \begin{bmatrix} k_{11} & \cdots & k_{1i_1-1} & k_{1i_1} & k_{1i_1+1} & \cdots & k_{1i_k-1} & k_{1i_k} & k_{1i_k+1} & \cdots & k_{1m} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ k_{i_1-11} & \cdots & k_{i_1-1i_1-1} & k_{i_1-1i_1} & k_{i_1-1i_1+1} & \cdots & k_{i_1-1i_k-1} & k_{i_1-1i_k} & k_{i_1-1i_k+1} & \cdots & k_{i_1-1m} \\ k_{i_11} & \cdots & k_{i_1i_1-1} & k_{i_1i_1} & k_{i_1i_1+1} & \cdots & k_{i_1i_k-1} & k_{i_1i_k} & k_{i_1i_k+1} & \cdots & k_{i_1m} \\ k_{i_1+11} & \cdots & k_{i_1+1i_1-1} & k_{i_1+1i_1} & k_{i_1+1i_1+1} & \cdots & k_{i_1+1i_k-1} & k_{i_1+1i_k} & k_{i_1+1i_k+1} & \cdots & k_{i_1+1m} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ k_{i_k-11} & \cdots & k_{i_k-1i_1-1} & k_{i_k-1i_1} & k_{i_k-1i_1+1} & \cdots & k_{i_k-1i_k-1} & k_{i_k-1i_k} & k_{i_k-1i_k+1} & \cdots & k_{i_k-1m} \\ k_{i_k1} & \cdots & k_{i_ki_1-1} & k_{i_ki_1} & k_{i_ki_1+1} & \cdots & k_{i_ki_k-1} & k_{i_ki_k} & k_{i_ki_k+1} & \cdots & k_{i_km} \\ k_{i_k+11} & \cdots & k_{i_k+1i_1-1} & k_{i_k+1i_1} & k_{i_k+1i_1+1} & \cdots & k_{i_k+1i_k-1} & k_{i_k+1i_k} & k_{i_k+1i_k+1} & \cdots & k_{i_k+1m} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ k_{m1} & \cdots & k_{mi_1-1} & k_{mi_1} & k_{mi_1+1} & \cdots & k_{mi_k-1} & k_{mi_k} & k_{mi_k+1} & \cdots & k_{mm} \end{bmatrix}_{m\times m} \tag{4}$$

Assume that the support constraints are imposed so that displacements of the i_1th,…,i_kth DOFs are specified. The stiffness matrix $\mathbf{K}$ for the modified structure can be obtained by deleting the i_1th,…i_k rows and columns from $\mathbf{K}_0$. Therefore,

$$\mathbf{K} = \begin{bmatrix} k_{11} & \cdots & k_{1i_1-1} & k_{1i_1+1} & \cdots & k_{1i_k-1} & k_{1i_k+1} & \cdots & k_{1m} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ k_{i_1-11} & \cdots & k_{i_1-1i_1-1} & k_{i_1-1i_1+1} & \cdots & k_{i_1-1i_k-1} & k_{i_1-1i_k+1} & \cdots & k_{i_1-1m} \\ k_{i_1+11} & \cdots & k_{i_1+1i_1-1} & k_{i_1+1i_1+1} & \cdots & k_{i_1+1i_k-1} & k_{i_1+1i_k+1} & \cdots & k_{i_1+1m} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ k_{i_k-11} & \cdots & k_{i_k-1i_1-1} & k_{i_k-1i_1+1} & \cdots & k_{i_k-1i_k-1} & k_{i_k-1i_k+1} & \cdots & k_{i_k-1m} \\ k_{i_k+11} & \cdots & k_{i_k+1i_1-1} & k_{i_k+1i_1+1} & \cdots & k_{i_k+1i_k-1} & k_{i_k+1i_k+1} & \cdots & k_{i_k+1m} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ k_{m1} & \cdots & k_{mi_1-1} & k_{mi_1+1} & \cdots & k_{mi_k-1} & k_{mi_k+1} & \cdots & k_{mm} \end{bmatrix}_{(m-k)\times(m-k)} \tag{5}$$

At first glance, Eqs. (4) and (5) may mislead one that only the i_1th and i_kth rows and columns are deleted from $\mathbf{K}_0$. In fact, k rows and k columns are deleted from $\mathbf{K}_0$, i.e, the i_1th, i_2th…,i_{k-1}th, i_k th rows and columns. Let

$$\tilde{\mathbf{K}}=\begin{bmatrix} k_{11} & \cdots & k_{1i_1-1} & 0 & k_{1i_1+1} & \cdots & k_{1i_k-1} & 0 & k_{1i_k+1} & \cdots & k_{1m} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ k_{i_1-11} & \cdots & k_{i_1-1i_1-1} & 0 & k_{i_1-1i_1+1} & \cdots & k_{i_1-1i_k-1} & 0 & k_{i_1-1i_k+1} & \cdots & k_{i_1-1m} \\ 0 & \cdots & 0 & k_{i_1i_1} & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ k_{i_1+11} & \cdots & k_{i_1+1i_1-1} & 0 & k_{i_1+1i_1+1} & \cdots & k_{i_1+1i_k-1} & 0 & k_{i_1+1i_k+1} & \cdots & k_{i_1+1m} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ k_{i_k-11} & \cdots & k_{i_k-1i_1-1} & 0 & k_{i_k-1i_1+1} & \cdots & k_{i_k-1i_k-1} & 0 & k_{i_k-1i_k+1} & \cdots & k_{i_k-1m} \\ 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & k_{i_ki_k} & 0 & \cdots & 0 \\ k_{i_k+11} & \cdots & k_{i_k+1i_1-1} & 0 & k_{i_k+1i_1+1} & \cdots & k_{i_k+1i_k-1} & 0 & k_{i_k+1i_k+1} & \cdots & k_{i_k+1m} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ k_{m1} & \cdots & k_{mi_1-1} & 0 & k_{mi_1+1} & \cdots & k_{mi_k-1} & 0 & k_{mi_k+1} & \cdots & k_{mm} \end{bmatrix}_{m\times m} \tag{6}$$

and

$$\tilde{\mathbf{R}}=\left(R_1,\cdots,R_{i_1-1},k_{i_1i_1}\bar{u}_{i_1},R_{i_1},\cdots,R_{i_k-k},\; k_{i_ki_k}\bar{u}_{i_k},R_{i_k-k+1},\cdots,R_{m-k}\right)^{\mathrm{T}}\in R^m \tag{7}$$

where the i_1th,…,i_kth, components of $\tilde{\mathbf{R}}$ are set to $k_{i_1i_1}\bar{u}_{i_1},\cdots,k_{i_ki_k}\bar{u}_{i_k}$, respectively, the remaining components are the same as the components of **R** and the order is also kept unchanged. Where the parameters $\bar{u}_{i1},\cdots,\bar{u}_{ik}$ are imposed node displacements. In this way, Eq. (3) can be rewritten as follows

$$\tilde{\mathbf{K}}\tilde{\mathbf{x}}=\tilde{\mathbf{R}} \tag{8}$$

For the modification of deleting support constraints where some node displacements along axes of the global coordinate system are relaxed, the modified stiffness matrix is obtained by inserting some rows and columns into the initial stiffness matrix symmetrically and the locations of the added rows and columns are only related to the numbers of the involved nodes. For the details, we refer the readers to Liu and Yue (2014).

3. The proposed method

In this section, the Cholesky rank-one update/downdate algorithm is first reviewed, then the proposed method is derived, finally the efficiency of the method is studied.

3.1 The Cholesky rank-one update/downdate algorithm

In 1974, the Cholesky rank-one update/downdate algorithm was presented (Gill *et al*. 1974). Suppose $\mathbf{A}\in R^{m\times m}$ is a SPD matrix, and the Cholesky factorization of **A** has already been known

$$\mathbf{A}=\mathbf{L}_{\mathrm{A}}\mathbf{D}_{\mathrm{A}}\mathbf{L}_{\mathrm{A}}^{\mathrm{T}} \tag{9}$$

Let $\bar{\mathbf{A}}$ be a symmetric rank-one modification of **A**, i.e.

$$\bar{\mathbf{A}}=\mathbf{A}+\eta\mathbf{w}\mathbf{w}^{\mathrm{T}}\quad(\eta\in R,\eta\neq 0) \tag{10}$$

where $\mathbf{w}\in R^m$ is a column vector. The following algorithm can be utilized to calculate the Cholesky factorization of $\bar{\mathbf{A}}$

$$\bar{\mathbf{A}}=\bar{\mathbf{L}}\bar{\mathbf{D}}\bar{\mathbf{L}}^{\mathrm{T}} \tag{11}$$

by using the factorization of **A** instead of directly factoring $\bar{\mathbf{A}}$.

Algorithm 1.

1. Define $\eta_1=\eta$, $\mathbf{w}^{(1)}=\mathbf{w}$.
2. For $j=1,2,\cdots,m$, compute

$$p_j=w_j^{(j)}$$
$$\bar{d}_j=d_j+\eta_j p_j^2$$
$$\tau_j=\frac{p_j\eta_j}{\bar{d}_j}$$
$$\eta_{j+1}=\frac{d_j\eta_j}{\bar{d}_j}$$

for $r=j+1,\cdots,m$

$$w_r^{(j+1)}=w_r^{(j)}-p_j l_{rj}$$
$$\bar{l}_{rj}=l_{rj}+\tau_j w_r^{(j+1)}$$

end

end

When η>0, $\bar{\mathbf{A}}$ is always SPD and the algorithm is called a rank-one update. When η<0, the algorithm is called a rank-one downdate (Davis 2006). Note that in this case $\bar{\mathbf{A}}$ may not be SPD and it is require to be SPD, otherwise the algorithm above is numerically unstable, even if the factorization of $\bar{\mathbf{A}}$ exists. The number of operations of Algorithm 1 is roughly 2 m^2 floating point operations (flops), and is equal to the cost of a matrix-vector product when *m* is large enough. If the width of $\bar{\mathbf{A}}$ is considered, the computational cost of the algorithm can be further decreased. For the details and progresses of the above algorithm, we refer the readers to Davis (2006).

3.2 The proposed method

Assume $\mathbf{B}\in R^{m\times m}$ is a SPD matrix and the Cholesky factorization of **B** has already been known

$$\mathbf{B}=\mathbf{L}_{\mathbf{B}}\mathbf{D}_{\mathbf{B}}\mathbf{L}_{\mathbf{B}}^{\mathrm{T}} \tag{12}$$

Partitioning **B** into

$$\mathbf{B}=\begin{bmatrix}\mathbf{B}_{11} & \mathbf{b}_1 & \mathbf{B}_{12}\\ \mathbf{b}_1^{\mathrm{T}} & b_{ii} & \mathbf{b}_2^{\mathrm{T}}\\ \mathbf{B}_{12}^{\mathrm{T}} & \mathbf{b}_2 & \mathbf{B}_{22}\end{bmatrix} \tag{13}$$

where $\mathbf{B}_{11}\in R^{(i-1)\times(i-1)}$, $\mathbf{B}_{22}\in R^{(m-i)\times(m-i)}$, $\mathbf{b}_1\in R^{i-1}$, $\mathbf{b}_2\in R^{m-i}$, $i\ (1\le i\le m)$ is an arbitrary positive integer.

Let

$$\tilde{\mathbf{B}}=\begin{bmatrix}\mathbf{B}_{11} & \mathbf{0} & \mathbf{B}_{12}\\ \mathbf{0}^{\mathrm{T}} & b_{ii} & \mathbf{0}^{\mathrm{T}}\\ \mathbf{B}_{12}^{\mathrm{T}} & \mathbf{0} & \mathbf{B}_{22}\end{bmatrix} \tag{14}$$

where **0** denotes the corresponding zero vector. It can be easily proven that $\tilde{\mathbf{B}}$ is SPD and its Cholesky factorization can be calculated by using the factorization of **B** and Algorithm 1. The details are given as follows (Sun and Yuan 2006):

Using Eqs. (13) and (14) yields

$$\mathbf{B} = \tilde{\mathbf{B}} + \mathbf{b}\mathbf{e}_i^{\mathrm{T}} + \mathbf{e}_i\mathbf{b}^{\mathrm{T}} \tag{15}$$

where

$$\mathbf{b} = \begin{bmatrix}\mathbf{b}_1^{\mathrm{T}} & 0 & \mathbf{b}_2^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}} \tag{16}$$

is a m-dimensional vector, $\mathbf{e}_i$ is the ith column of the identity matrix of order m. Let

$$\mathbf{y} = \frac{\mathbf{b}+\mathbf{e}_i}{\sqrt{2}}, \quad \mathbf{z} = \frac{\mathbf{b}-\mathbf{e}_i}{\sqrt{2}} \tag{17}$$

Substituting Eq. (17) into (15) yields

$$\mathbf{B} = \tilde{\mathbf{B}} + \mathbf{y}\mathbf{y}^{\mathrm{T}} - \mathbf{z}\mathbf{z}^{\mathrm{T}} \tag{18}$$

The above equation can be rewritten as

$$\tilde{\mathbf{B}} = \mathbf{B} + \mathbf{z}\mathbf{z}^{\mathrm{T}} - \mathbf{y}\mathbf{y}^{\mathrm{T}} \tag{19}$$

Eq. (19) shows that the factorization of $\tilde{\mathbf{B}}$ can be calculated by using Algorithm 1 twice, which results in

$$\tilde{\mathbf{B}} = \mathbf{L}\mathbf{D}\mathbf{L}^{\mathrm{T}} \tag{20}$$

Implementing the above procedure for $\tilde{\mathbf{K}}$ and $\mathbf{K}_0$ one after another, yields

$$\tilde{\mathbf{K}} = \mathbf{K}_0 + \sum_{l=1}^{k}\left(\mathbf{z}_{i_l}\mathbf{z}_{i_l}^{\mathrm{T}} - \mathbf{y}_{i_l}\mathbf{y}_{i_l}^{\mathrm{T}}\right) \tag{21}$$

Based on the factorization of $\mathbf{K}_0$ in Eq. (2), the Cholesky factorization of $\tilde{\mathbf{K}}$ can be obtained by carrying out 2k times Algorithm 1. In fact, the number of Algorithm 1 required is equal to two times of the number of the modified supports. If a free node with s DOFs is constrained in all directions, the number of the modified supports is s and Algorithm 1 will be required 2s times. Since k is usually very small compared with m, the computational cost is inexpensive. Using the Cholesky factorization of $\tilde{\mathbf{K}}$ and solving Eq. (8) by the forward and backward substitutions yield the displacements vector of the modified structure.

For the case of deleting support constraints where some node displacements along axes of the global coordinate system are relaxed, the above process can also be used. In order to avoid duplication, the details are omitted.

3.3 The efficiency of the proposed method

The computational cost of above method can be quantified by the number of flops. Suppose the number of DOFs of the original structure is m, and k is the number of the changed support constraints. Assume the half-band widths of the initial stiffness matrix and the modified stiffness matrix are the same, and b represents the half-band width. The case of b<<m is considered. The computational cost of using Algorithm 1 one time is roughly 2mb flops since the entries of $\bar{\mathbf{L}}$ outside the width are all zeros. Solving one linear system of order $m-k$ (for the case of adding k support constraints) or $m+k$ (for the case of deleting k support constraints) with half-band width b by utilizing the forward and backward substitutions requires $4(m-k)b$ or $4(m+k)b$ flops (Golub and Van Loan 2013). Thus, the total computational cost of the proposed method is $4(m-k)b+4kmb$ or $4(m+k)b+4kmb$ flops since Algorithm 1 is employed 2k times. It is roughly equal to $4mb+4kmb$ flops due to k<<m. Direct analysis costs $(m-k)(b^2+8b+1)$ or $(m+k)(b^2+8b+1)$ flops (Golub and Van Loan 2013) since the Cholesky factorization of the modified stiffness matrix is involved, and is roughly equal to $m(b^2+8b+1)$ flops. The theoretical speed up S_t is defined as the ratio of the flops using the direct analysis method to that using the proposed method (Leu and Tsou 2000), that is

$$S_t = \frac{m(b^2+8b+1)}{4mb+4kmb} \tag{22}$$

Eq. (22) can be approximated by

$$S_t \approx \frac{m(b^2+8b)}{4mb+4kmb} = \frac{b+8}{4+4k} \tag{23}$$

From Eq. (23), it can be observed that the smaller k is, the larger S_t is. Using Eq. (23) yields $S_t \geq 1$, if

$$k \leq \frac{1}{4}(b+4) \tag{24}$$

i.e., if the number of the changed support constraints k satisfies the inequality (24), the computational cost of the proposed method will be equal to or less than that of the direct analysis method.

4. Numerical examples

In this section, two examples are given to illustrate the effectiveness of the proposed method. All the computations are completed on a PC: Pentium 4, quad-core CPU with 2.66 GHz, 2 GB RAM. Compaq Visual Fortran 6.5 is used.

Example 1

A ceiling structure shown in Fig. 2 is studied for case of adding support constraints. The length and width of the structure are 36 m and 24 m, respectively. The modulus of elasticity of the material is $E=2.07\times10^{11}$ Pa, and the Poisson's ratios is $\upsilon=0.3$. The structure has 2652 pipe elements and 925 nodes. The outer radius of every pipe element is 0.06 m, and the thickness is 0.01 m. Every node has 6 DOFs except the 52 constrained nodes and the total number of DOFs of the structure is 5238. The top view of the structure is shown in Fig. 3 where '○' denotes the nodes of support constraints. In order to reinforce the structure, two free nodes as shown in Fig. 3 with '□' are constrained, and 12 displacement constraints are imposed. The number of DOFs of the modified structure is 5226. Every free node of the modified structure is subjected to a vertical load

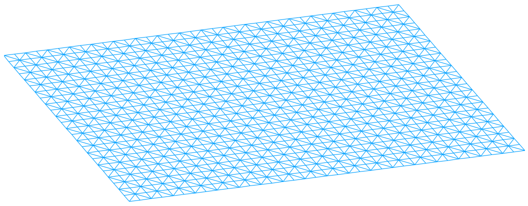


Fig. 2 The ceiling structure

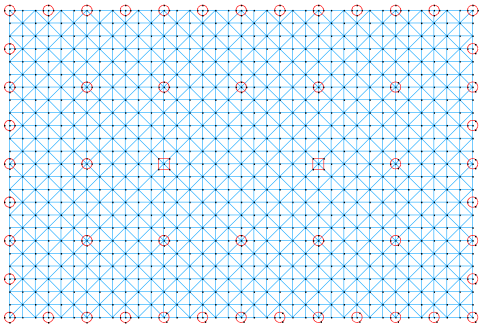


Fig. 3 The top view of the ceiling structure

Table 1 The maximum vertical displacements calculated by the proposed method, the direct analysis method and the reanalysis methods presented in Liu *et al.* (2012) and Liu *et al.* (2014) for the modified ceiling structure

	Proposed method	Direct analysis	Liu *et al.* (2012)	Liu *et al.* (2014)
The maximum vertical displacements	−0.0394 m	−0.0394 m	−0.0394 m	−0.0394 m

Table 2 The computational times for the modified ceiling structure

	Proposed method	Direct analysis	Liu *et al.* (2012)	Liu *et al.* (2014)
The computational times	0.4188 s	6.4547 s	0.5632 s	2.6215 s

P=200 N.

Table 1 gives the maximum vertical displacement of the modified structure calculated by the proposed method, the direct analysis method and the reanalysis methods presented in Liu *et al.* (2012), Liu *et al.* (2014). It can be observed that, all the results are the same. The computational times are listed in Table 2. It shows that the computational time of the proposed method is the shortest among all the methods.

Example 2

Consider the framework of the building shown in Fig. 4, the case of deleting support constraints is studied. The length, width and height of the framework are 36 m, 18 m and 20 m, respectively. The height of each floor is 5 m. The material modulus of elasticity and the Poisson's ratio are E=3×10^{10} Pa and υ=0.2, respectively. A finite element model is employed to simulate the framework under a given load. The model has 672 elements (288 plate elements and 384 beam elements) and 439 nodes. Every node has 6 DOFs except the 15 constrained nodes and the total number of DOFs of the structure is 2544. All the constrained nodes are at the bottom of the structure, as showed in Fig. 5, where '•' denotes the support constraint nodes. The thickness and the size of all the plates are 0.2 m and 3 m×3 m, respectively. The beams have two cross-section sizes, those perpendicular to the ground have the square 0.5 m×0.5 m cross-sections, others have the rectangular 0.3 m×0.6 m cross-sections. Each free node of the framework are subjected to a vertical load P=2×10^4 N. The modification is made by deleting the rotation constraints of two nodes denoted by '△' in Fig. 6. Thus, the total number of DOFs of the modified structure is 2550.

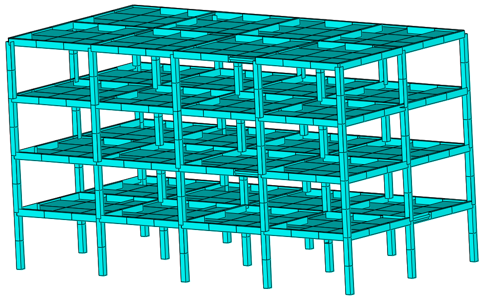


Fig. 4 The framework of building

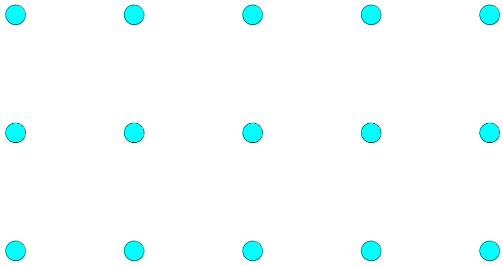


Fig. 5 The constrained nodes of the initial framework

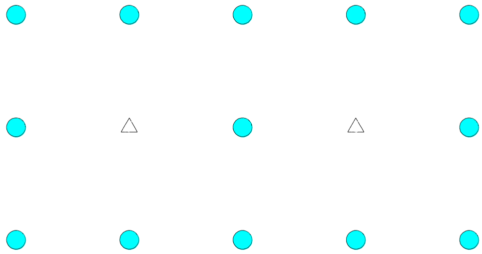


Fig. 6 The constrained nodes of the modified framework

Table 3 gives the maximum vertical displacement of the modified structure calculated by the proposed method, the direct analysis method and the reanalysis methods presented in Liu and Yue (2014) and Liu *et al.* (2014). The computational times are listed in Table 4. It can be seen from Tables 3 and 4 that, the maximum vertical displacement calculated by the four methods are the same, while the computational time of the proposed method is the shortest.

Table 3 The maximum vertical displacements calculated by the proposed method, the direct analysis method and the methods presented in Liu and Yue (2014) and Liu *et al.* (2014) for the modified framework

	Proposed method	Direct analysis	Liu and Yue (2014)	Liu *et al.* (2014)
The maximum vertical displacements	−0.048 m	−0.048 m	−0.048 m	−0.048 m

Table 4 The computational times for the modified framework structure

	Proposed method	Direct analysis	Liu and Yue (2014)	Liu *et al.* (2014)
The maximum vertical displacements	0.3048 s	11.4328 s	0.4182 s	0.6527 s

5. Conclusions

A unified method for structural static reanalysis with imposition or relaxation of some support constraints along axes of the global coordinate system has been presented. The Cholesky rank-one update/downdate algorithm is employed. The proposed method is easy to implement and the computational time can be significantly reduced. Although the scope of the method is less than that of the method based on the rank-one decomposition of the element stiffness matrix, the computational cost is less expensive. In addition, the Cholesky factorization of the modified stiffness matrix is obtained, it can be used as the initial information when the structure is further modified. Numerical examples have demonstrated the effectiveness of the proposed approach.

Acknowledgments

The work was supported by the Fundamental Research Funds for the Central Universities and the Natural Science Foundation of China (Grant Nos. 11401463 and 61573273).

References

Abu Kasim, A.M. and Topping, B.H.V. (1987), "Static reanalysis: a review", *J. Struct. Eng.*, ASCE, **113**(5), 1029-1045.

Davis, T.A. (2006), *Direct Methods for Sparse Linear Systems*, SIAM, Philadephia, PA, USA.

Gao, H.H., Zhu, J.H., Zhang, W.H. and Zhou, Y. (2015), "An improved adaptive constraint aggregation for integrated layout and topology optimization", *Comput. Meth. Appl. M.*, **289**, 387-408.

Gill, P.E., Golub, G.H., Murray, W. and Saunders, M.A. (1974), "Methods for modifying matrix factorizations", *Math. Comput.*, **28**(126), 505-535.

Golub, G.H. and Van Loan, C.F. (2013), *Matrix Computations*, 4th Edition, The Johns Hopkins University Press, Baltimore, MD, USA.

Hassan, M.R.A., Azid, I.A., Ramasamy, M., Kadesan J., Seetharamu, K.N., Kwan, A.S.K. and Arunasalam P. (2010), "Mass optimization of four bar linkage using genetic algorithms with dual bending and buckling constraints", *Struct. Eng. Mech.*, **35**(1), 83-98.

Kirsch, U. (2008), *Reanalysis of Structures*, Springer, Dordrecht, Netherlands.

Kozikowska, A. (2011), "Topological classes of statically determinate beams with arbitrary number of supports", *J. Theor. Appl. Mech.*, **49**(4), 1079-1100.

Leu, L.J. and Tsou, C.H. (2000), "Application of a reduction method for reanalysis to nonlinear dynamic analysis of framed structures", *Comput. Mech.*, **26**(5), 497-505.

Liu, H.F. and Yue, S.G. (2014), "An efficient method to structural static reanalysis with deleting support constraints", *Struct. Eng. Mech.*, **52**(6), 1121-1134.

Liu, H.F., Wu, B.S., and Li, Z.G. (2012), "An efficient approach to structural static reanalysis with added support constraints", *Struct. Eng. Mech.*, **43**(3), 273-285.

Liu, H.F., Wu, B.S., Li, Z.G. and Zheng, S.P. (2014), "Structural static reanalysis for modification of supports", *Struct. Multidisc. Optim.*, **50**(3), 425-435.

Olhoff, N. and Taylor, J.E. (1983), "On structural optimization", *J. Appl. Mech.*, ASME, **50**(4), 1139-1151.

Sun, W.Y. and Yuan, Y.X. (2006), *Optimization Theory and Methods: Nonlinear Programming*, Springer, New York, NY, USA.

Takezawa, A., Nishiwaki, S., Izui, K. and Yoshimura, M. (2006), "Structural optimization using function-oriented elements to support conceptual designs", *J. Mech. Des.*, ASME, **128**(4), 689-700.

Terdalkar, S.S. and Rencis, J.J. (2006), "Graphically driven interactive finite element stress reanalysis for machine elements in the early design stage", *Finite. Elem. Anal. Des.*, **42**(10), 884-899.

Xia, Q. and Shi, T.L. (2016), "Topology optimization of compliant mechanism and its support through a level set method", *Comput. Meth. Appl. M.*, **305**, 359-375.

Xia, Q., Wang, M.Y. and Shi, T.L. (2014), "A level set method for shape and topology optimization of both structure and support of continuum structures", *Comput. Meth. Appl. M.*, **272**, 340-353.

Zhu, J.H. and Zhang, W.H. (2010), "Integrated layout design of supports and structures", *Comput. Meth. Appl. M.*, **199**(9-12), 557-56.

CC