AN EFFICIENT METHOD TO SIMULATE THE VARYING STATIC RECEPTANCE OF A THIN-WALLED WORKPIECE

CHRISTIAN BRECHER[1], GUIFENG ZHAO[1], MARCEL FEY[1]

[1]RWTH Aachen University, Laboratory for Machine Tools and Production Engineering, Aachen, Germany

DOI: 10.17973/MMSJ.2023_12_2023096

G.Zhao@wzl.rwth-aachen.de

A new method to simulate the varying static receptance of a thin-walled workpiece based on the Cholesky decomposition is presented in this paper. In this method, the system stiffness matrix is updated by subtracting the elemental stiffness matrices of the removed elements from the initial system stiffness matrix. The updated system stiffness matrix is permuted by using a novel method based on element removing sequence and considering fill-in reduction. The Cholesky factor of the permuted system stiffness matrix is reused to compute the static receptance of the workpiece for multiple cutter locations. The reduction of computation time by using this new method is proved by a numerical test and the accuracy of the simulation is demonstrated with a machining test.

KEYWORDS

thin-walled workpiece, static receptance, Cholesky decomposition, finite element simulation, permutation

1 INTRODUCTION

The CAM planning for thin-walled workpieces poses a significant challenge [Koike 2013] [Wiederkehr 2013]. One primary challenge stems from shape errors arising due to the workpiece's substantial static receptance [Altintas 2018] [Denkena 2007] [Dittrich 2019]. Typically, numerous machining tests are needed to iteratively optimize CAM planning until suitable process parameters are identified [Bolar 2016]. However, this approach leads to substantial material and time costs [Bolar 2016]. The finite element simulation can be used to predict the static displacement of the thin-walled workpiece. The shape errors can be compensated based on the simulated static displacement. The most time-consuming step of a finite element simulation for static problems is to solve Eq. 1:

$$\mathrm{Ku} = \mathrm{r} \tag{1}$$

with:

K: system stiffness matrix

u: static displacement vector

r: force vector

Eq. 1 can be solved using a direct solving method based on the Cholesky decomposition [Bathe 2014]. The first step involves decomposing the matrix K:

$$\mathrm{K} = \mathrm{LL^T} \tag{2}$$

with:

L: lower Cholesky factor (lower triangular matrix)

L^T: transposed matrix of L (upper Cholesky factor)

The computation time of the Cholesky decomposition for a dense symmetric positive definite (SPD) matrix is proportional to n^3 [Cormen 2022], where n is the number of the equations in Eq. 1. The matrix K for the structural mechanics problems is usually a sparse SPD matrix. The sparsity of the matrix K can be used to reduce the complexity of the Cholesky decomposition. The decomposition of a sparse SPD matrix includes a symbolic phase and a numerical factorization phase. The symbolic phase typically uses only the sparsity pattern of the matrix K to compute the nonzero structure of the Cholesky factor of K without computing the numerical values of the nonzeros. The number of nonzeros in the Cholesky factor is typically larger than the number of nonzeros in the matrix K. This phenomenon is named fill-in. The fill-in increases the memory and the computation time requirement for the numerical factorization phase. Therefore, the matrix K is typically permuted to reduce the fill-in in the symbolic phase [Scott 2023]:

$$\widetilde{\mathrm{K}} = \mathrm{P^T} \cdot \mathrm{K} \cdot \mathrm{P} \tag{3}$$

with:

P: matrix to reorder the columns

$\mathrm{P^T}$: transposed matrix of P (matrix to reorder rows)

The approximate minimum degree algorithm (AMD) is widely used to generate the permutation matrix P because of its good fill-in-reducing effect [Amestoy 1996]. Furthermore, the computation cost using the AMD algorithm is smaller than using other algorithms such as the multiple minimum degree algorithm (MMD) [Amestoy 1996]. The sparsity pattern of the Cholesky factor of the permuted matrix $\widetilde{\mathrm{K}}$ is computed at the end of the symbolic phase [Scott 2023]. This is used as the input for the numerical factorization phase to compute the values in the Cholesky factor G:

$$\widetilde{\mathrm{K}} = \mathrm{GG^T} \tag{4}$$

The displacement vector u can be computed by leveraging the Cholesky factor G. Substituting Eq. 3 into Eq. 1 yields Eq. 5:

$$\widetilde{\mathrm{K}}(\mathrm{P^T u}) = \mathrm{P^T r} \tag{5}$$

Substituting Eq. 4 into Eq. 5 yields Eq. 6:

$$\mathrm{Gv} = \mathrm{P^T r} \tag{6}$$

with:

$$\mathrm{G^T}(\mathrm{P^T u}) = \mathrm{v} \tag{7}$$

v is computed in Eq. 6 by forward substitution. Subsequently, $\mathrm{P^T} \cdot \mathrm{u}$ is obtained in Eq. 7 by backward substitution. The last step is to compute u in Eq. 8 with $\mathrm{P^T u}$ computed in Eq. 7:

$$\mathrm{u} = (\mathrm{PP^T}) \cdot \mathrm{u} = \mathrm{P}(\mathrm{P^T u}) \tag{8}$$

The most time-consuming step to solve the sparse linear problem in Eq. 1 is to compute the Choleksy factor G in the numerical factorization phase of the Cholesky decomposition [Scott 2023].

The peculiarity of the finite element simulation for the thin-walled workpiece is the variation of the system stiffness matrix K because of the material removal. Therefore, multiple simulations for the intermediate states of the thin-walled workpiece are necessary. This can require extremely high computational costs.

The typical methods for simulating thin-walled workpieces can be categorized into three groups. In the first group, the system stiffness matrix of the thin-walled workpiece remains constant throughout the entire machining process [Khandagale 2018]. As a result, only a single Cholesky decomposition is required for the complete machining process. However, the usability of the methods in this group is severely limited, as they do not account for variations in the workpiece's static receptance. The methods in the second group address this variation by performing a new Cholesky decomposition for each selected intermediate state of the workpiece [Budak 1995], [Ratchev 2005], [Wimmer 2019]. While this approach enhances simulation accuracy, it also leads to significant computational time demands. To mitigate this computational burden, the third group employs the substructure coupling method [Li 2018]. The basic idea of substructure coupling is that the physical properties of the entire structure can be determined by coupling the physical properties of the associated substructures [Chavan 2020]. When applying the substructure coupling method to simulate the thin-walled workpiece, the first step consists of conducting a finite element simulation to get the static receptance of the initial workpiece. Subsequently, an intermediate state of the workpiece is considered to be the coupling of the preceding intermediate state (substructure 1) and the removed material (substructure 2). The static receptance of this intermediate state is approximated based on the static receptances of substructures 1 and 2. This iterative process continues until calculations encompass the entire machining process. With substructure coupling, the computation time is reduced because the computation results of one intermediate state are used for the computation of the next intermediate state. However, a large cumulative error can arise due to the approximation for each intermediate state. Consequently, the accuracy of the calculation cannot be assured.

In summary, the current methods for simulating a thin-walled workpiece are either inefficient or lack sufficient accuracy. These limitations hinder the use of the finite element simulation in the CAM planning for the thin-walled workpiece. The purpose of the research presented in this paper is to improve efficiency without reducing the accuracy of the simulation for the thin-walled workpiece.

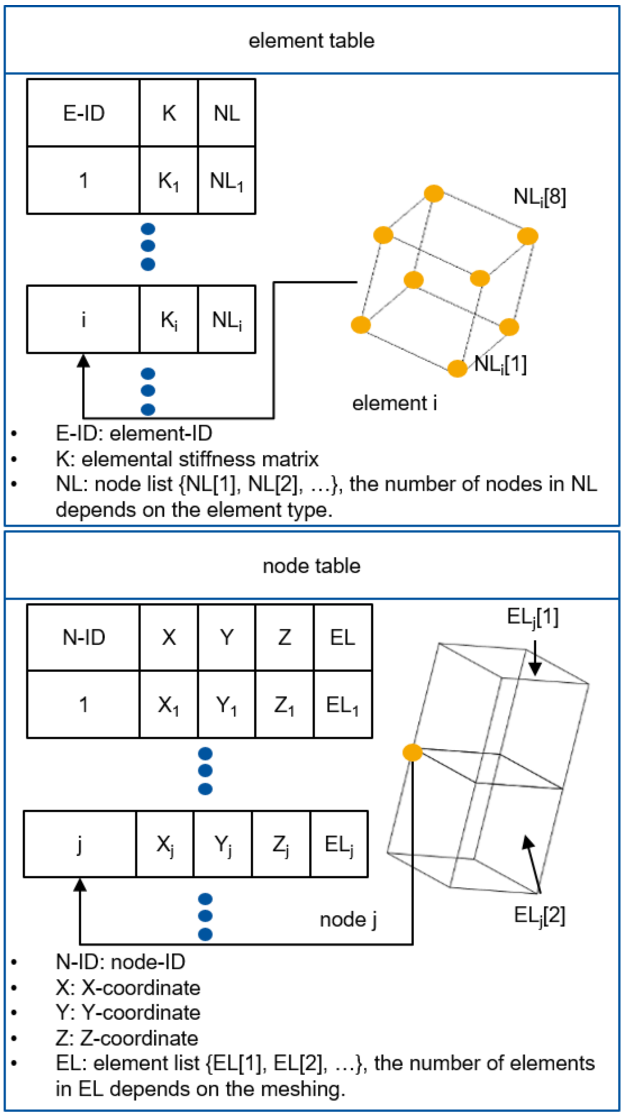
Following this introduction section, section 2 presents a method to update the system stiffness matrix without remeshing the workpiece geometry. Subsequently, an efficient method to compute the Cholesky factor of the varying system stiffness matrix is developed in section 3. Based on the method developed in section 3, a method to compute the static receptance of the thin-walled workpiece is presented in section 4. The efficiency and the accuracy of the simulation are analyzed in sections 5 and 6. The conclusions and outlooks are provided in the last section.

2 METHOD TO UPDATE SYSTEM STIFFNESS MATRICES WITHOUT REMESHING

The system stiffness matrix of the thin-walled workpiece can be updated by remeshing the workpiece geometry at each cutter location (CL). However, automating geometry remeshing is challenging. Furthermore, the system stiffness matrix is changed completely after remeshing. Hence, the Cholesky factors of the system stiffness matrices before and after the remeshing are completely distinct. As a result, reusing the Cholesky factor becomes unfeasible after remeshing.

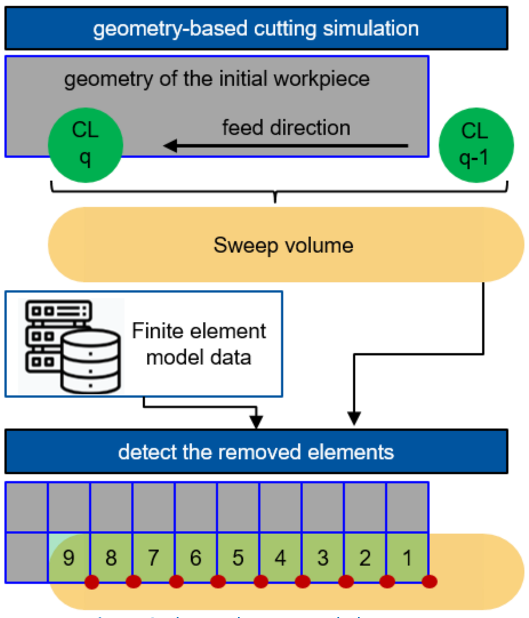
In this section, a system stiffness matrix updating method without remeshing is presented. The procedure begins by meshing the initial workpiece. The elements and the nodes are extracted from the finite element model and stored in an element table and a node table (Fig. 1). In the element table, each element is in a row where the element ID, the elemental stiffness matrix and the IDs of the corresponding nodes are stored. In the node table, each node is in a row where the node ID, the node coordinates and the IDs of the corresponding elements are stored.

Subsequently, the change history of the mesh due to material removal is computed using the extracted data from the finite element model and a geometry-based cutting simulation (Fig. 2). The cutter locations (CL) and the cutter geometry are exported from the geometry-based cutting simulation and used to generate the sweep volume of the cutter. The elements in this sweep volume are detected and removed from the mesh. The nodes associated with the removed elements are influenced. These nodes are classified into two groups. As shown in Fig. 2, the red nodes are deleted from the mesh after the removal of the green marked elements 1 to 9. These nodes are in the first group. The remaining nodes of these removed elements are only influenced, but not deleted. These nodes are in the second group. In order to track the change history of the mesh, the removed elements and the two groups of the affected nodes are stored in an intermediate mesh table, as shown in Fig. 3.



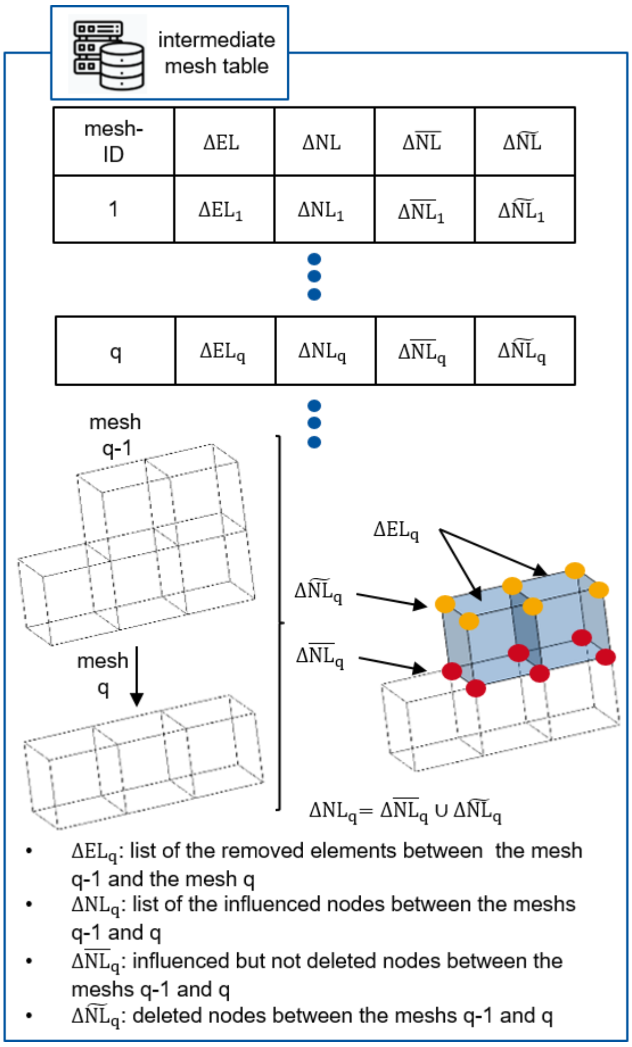
element table
E-ID | K | NL
1 | K_1 | NL_1
i | K_i | NL_i
$NL_i[8]$
$NL_i[1]$
element i
• E-ID: element-ID
• K: elemental stiffness matrix
• NL: node list {NL[1], NL[2], …}, the number of nodes in NL depends on the element type.
node table
N-ID | X | Y | Z | EL
1 | X_1 | Y_1 | Z_1 | EL_1
j | X_j | Y_j | Z_j | EL_j
$EL_j[1]$
$EL_j[2]$
node j
• N-ID: node-ID
• X: X-coordinate
• Y: Y-coordinate
• Z: Z-coordinate
• EL: element list {EL[1], EL[2], …}, the number of elements in EL depends on the meshing.

Figure 1: extract data from the finite element model



geometry-based cutting simulation
geometry of the initial workpiece
CL q
feed direction
CL q-1
Sweep volume
Finite element model data
detect the removed elements
9
8
7
6
5
4
3
2
1

Figure 2: detect the removed elements



mesh-ID	ΔEL	ΔNL	$\Delta \bar{N}L$	$\Delta \tilde{N}L$
1	ΔEL_1	ΔNL_1	$\Delta \bar{N}L_1$	$\Delta \tilde{N}L_1$
⋮				
q	ΔEL_q	ΔNL_q	$\Delta \bar{N}L_q$	$\Delta \tilde{N}L_q$

- ΔEL_q: list of the removed elements between the mesh q-1 and the mesh q
- ΔNL_q: list of the influenced nodes between the meshs q-1 and q
- $\Delta \bar{N}L_q$: influenced but not deleted nodes between the meshs q-1 and q
- $\Delta \tilde{N}L_q$: deleted nodes between the meshs q-1 and q

Figure 3. intermediate mesh table

The system stiffness matrix for the mesh q is computed by subtracting the elemental stiffness matrices of the removed elements from the system stiffness matrix of the mesh q-1:

$$K_{sys,q} = K_{sys,q-1} - \sum_i K_i \qquad (9)$$

with:

$K_{sys,q}$: system stiffness matrix of the mesh q

$K_{sys,q-1}$: system stiffness matrix of the mesh q-1

K_i: elemental stiffness matrix of the removed element i

The matrices $K_{sys,q}$, $K_{sys,q-1}$ and K_i are $n_{DoF} \times n_{DoF}$ matrices, where n_{DoF} is calculated by

$$n_{DoF} = n_{nodes} \cdot n_{dimension} \qquad (10)$$

with:

n_{DoF}: number of the degrees of freedom (DoF) of the initial mesh

n_{nodes}: number of the nodes of the initial mesh

$n_{dimension}$: number of the DoF of each node

The IDs of the removed elements are retrieved from the intermediate mesh table, and the elemental stiffness matrices of the removed elements are found in the element table. In most cases, the active nodes of the mesh q are fewer than n_{nodes} due to multiple node removals. Hence, $n_{DoF,deleted}$ rows and columns in $K_{sys,q}$ are filled by zeros, where:

$$n_{DoF,deleted} = n_{nodes,deleted} \cdot n_{dimension} \qquad (11)$$

with:

$n_{DoF,deleted}$: number of the deleted DoF from the initial mesh to mesh q

$n_{nodes,deleted}$: number of the deleted nodes from the initial mesh to mesh q

The matrix $K_{sys,q}$ is singular because of these zero rows and columns. The corresponding Cholesky decomposition is impossible because of the singularity. In order to avoid the singularity, the removal of the zero rows and columns from the system stiffness matrix is necessary. An illustrative example clarifies the removal process. The example matrix $K_{sys,q}$ is an 8 × 8 matrix in Fig. 4. The red marked columns and rows have just zero entries. The first step is to construct an 8 × 6 matrix S_{col}:

$$S_{col}(i, j+1) = 1 \qquad (12)$$

with:

i: the index of the column in $K_{sys,q}$ with at least one nonzero entry

j: the number of the already inserted ones in S_{col}

The matrix S_{col} in this example selects the columns 2, 4, 5, 6, 7 and 8:

$$K_{sys,q} S_{col} \qquad (13)$$

The matrix S_{row} to select the rows 2, 4, 5, 6, 7 and 8 is the transpose of S_{col}:

$$S_{row} = S_{col}{}^{T} \qquad (14)$$

Finally, the non-singular system stiffness matrix $K_{sys\_spd,q}$ is computed as:

$$K_{sys\_spd,q} = S_{row} K_{sys,q} S_{col} \qquad (15)$$

$$K_{sys,q} = \begin{array}{c|cccccccc|c} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & \\ \hline & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ & 0 & + & 0 & + & 0 & 0 & 0 & 0 & 2 \\ & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 \\ & 0 & + & 0 & + & + & 0 & 0 & 0 & 4 \\ & 0 & 0 & 0 & + & + & + & 0 & 0 & 5 \\ & 0 & 0 & 0 & 0 & + & + & + & 0 & 6 \\ & 0 & 0 & 0 & 0 & 0 & + & + & + & 7 \\ & 0 & 0 & 0 & 0 & 0 & 0 & + & + & 8 \end{array}$$

build matrix S_{col} to select columns 2, 4, 5, 6, 7 and 8

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad \begin{array}{l} \\ S_{col}(2,1) = 1 \\ \\ S_{col}(4,2) = 1 \\ S_{col}(5,3) = 1 \\ S_{col}(6,4) = 1 \\ S_{col}(7,5) = 1 \\ S_{col}(8,6) = 1 \end{array}$$

matrix S_{row} to select rows 2, 4, 5, 6, 7 and 8 is equal to $S_{col}{}^{T}$

$$K_{sys\_spd,q} = S_{row} \cdot K_{sys_{spd},q} \cdot S_{col}$$

$$\begin{bmatrix} + & + & 0 & 0 & 0 & 0 \\ + & + & + & 0 & 0 & 0 \\ 0 & + & + & + & 0 & 0 \\ 0 & 0 & + & + & + & 0 \\ 0 & 0 & 0 & + & + & + \\ 0 & 0 & 0 & 0 & + & + \end{bmatrix}$$

Figure 4. removal of the zero rows and columns

The method outlined in this section to update the system stiffness matrix can be concluded into the following steps:

- Extract the data from the finite element model of the initial workpiece and store the data into the element table and the node table.
- Detect the removed elements and the influenced nodes for each CL and store these data into the intermediate mesh table.

- Compute the singular system stiffness matrix using Eq. 9.
- Compute the non-singular system stiffness matrix using Eq. 15.

By following these steps, the method updates the system stiffness matrix without remeshing the workpiece geometry. This facilitates the reuse of the Cholesky factor for multiple CLs.

3 AN EFFICIENT METHOD TO COMPUTE THE CHOLESKY FACTOR

As explained in the introduction, computing Cholesky decompositions for multiple intermediate meshes demands substantial computational time. The aim of this section is to develop a method to efficiently compute the Cholesky factors for a thin-walled workpiece.

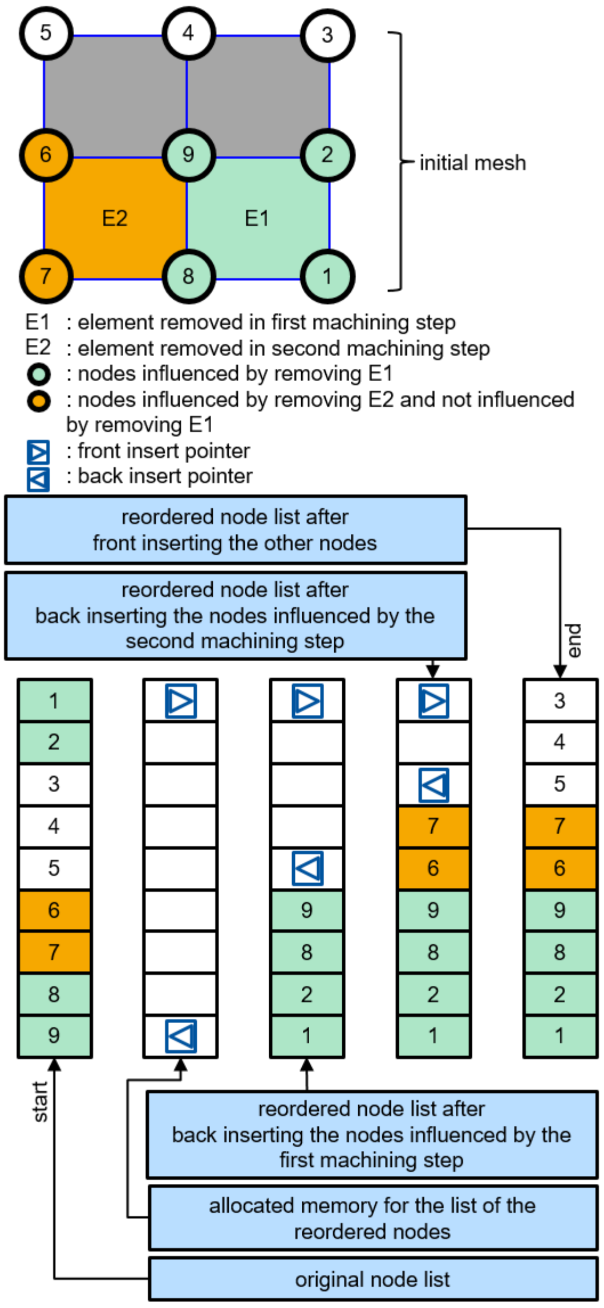
For simplicity, this section focuses on the lower Cholesky factor, since the upper Cholesky factor is its transpose. The fundamental idea is to reuse the Cholesky factor from one intermediate mesh to compute Cholesky factors for several others. This concept hinges on a specific property of the lower Cholesky factor [Scott 2023]:

- The value $L_{i,j}$ in the i-th row and j-th column of the lower Cholesky factor L is dependent on the entries $K_{v,w}$ in the system stiffness matrix K, where $v \leq i$ and $w \leq j$.

It is assumed, that the initial mesh of the thin-walled workpiece has n DoF. An intermediate mesh emerges by removing multiple elements from the initial mesh, and m DoF are affected by the removal of these elements. If these m DoF correspond to the last m DoF of the initial mesh, the top left $(n-m)\times(n-m)$ submatrices for both the initial and the intermediate meshes are identical. According to the aforementioned property of the lower Cholesky factor, the top left $(n-m)\times(n-m)$ submatrices of the lower Cholesky factors for both the initial and the intermediate meshes are identical. Hence, the computed $(n-m)\times(n-m)$ submatrix of the Cholesky lower factor can be reused for multiple intermediate meshes. However, it's not guaranteed that the affected nodes stored in the intermediate mesh table are the last nodes of the mesh. Consequently, the correspondingly influenced DoF may not necessarily be the last DoF. To address this, a node permutation method is introduced as follows:

- The memory with the size of the original node list stored in the node table is allocated for the reordered node list. The front insert pointer points to the first position of the allocated memory, and the back insert pointer points to the last position of the allocated memory.
- The influenced nodes stored in the intermediate mesh table (Fig. 3) are back inserted into the reordered node list. The back insert pointer shifts up one position after each insertion.
- The uninfluenced nodes (nodes in the node table but not in the intermediate mesh table) are front inserted into the reordered node list. The front insert pointer shifts down one position after each insertion.

This method is illustrated with an example in Fig. 5. Initially, memory is allocated for the new node list containing nine integers. The front insert pointer points to the first position of the new node list and the back insert pointer points to the 9th position. The machining process involves two machining steps. The nodes 1, 2, 8 and 9 are influenced by removing the element E1 during the first machining step. These nodes are back inserted into the new node list. The position to start the back insertion for this machining step is the 9th position of the new node list, where the back insert pointer points to. The back insert pointer is pointed to the 5th position of the new node list after inserting these 4 nodes. The nodes 6, 7, 8 and 9 are influenced by removing the element E2 during the second machining step. However, the nodes 8 and 9 are already considered in the last machining step. Hence, only the nodes 6 and 7 are inserted at the back of the new node list. In the last step, the uninfluenced nodes 3, 4 and 5 are inserted into the front of the new node list.



5
4
3
6
9
2
E2
E1
7
8
1
initial mesh
E1 : element removed in first machining step
E2 : element removed in second machining step
: nodes influenced by removing E1
: nodes influenced by removing E2 and not influenced by removing E1
: front insert pointer
: back insert pointer
reordered node list after front inserting the other nodes
reordered node list after back inserting the nodes influenced by the second machining step
end
start
reordered node list after back inserting the nodes influenced by the first machining step
allocated memory for the list of the reordered nodes
original node list

Figure 5. reorder the node list

A permutation matrix P1 is created based on the reordered node list, as shown in Fig. 6. Firstly, a list of DoF corresponding to the list of the reordered nodes is constructed. In this example, the reordered node list has nine nodes, each of them has two DoF. A list of DoF with a length of 18 is constructed by iterating over the reordered node list. After any machining step i, the number of the DoF which not affected by the steps from 1 to i is stored in a map data structure. In Fig. 6, the numbers of the unaffected DoF after machining step 1 and machining step 2 are stored in the map as two key to value pairs: 1→10 and 2→6. The permutation matrix P1 is created by assigning the ones to a zero matrix according to the list of DoF. For the DoF i in the position j of the DoF list, a value 1 is inserted to the matrix P1 as follows:

$$p1_{i,j} = 1 \quad (16)$$

with i equal to $(\text{node}_{\text{pos}} * n_{\text{DoF}} - 1)$, where node_{pos} is the position of the node in the reordered node list, and n_{DoF} is the number of DoF for each node.

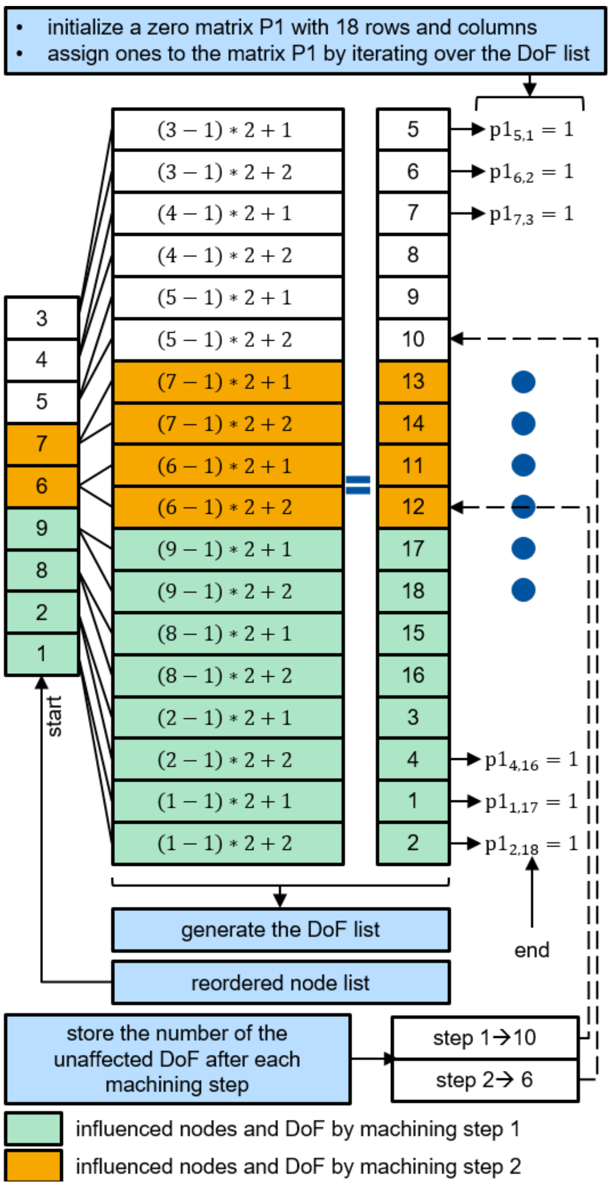


Figure 6. construct the permutation matrix P1

In this example, as shown in Fig. 6, 18 ones are inserted into the permutation matrix P1.

In order to reduce the computation time and the memory requirement of a single Cholesky decomposition, the AMD algorithm based permutation is combined with the element removing sequence based permutation. Firstly, the system stiffness matrix K is permuted using the matrix P1 (based on the sequence of the element removal). The first m rows and the first m columns of the permuted matrix are extracted and stored in the matrix $\widetilde{K}_{\text{top\_lef}}$, where m is the number of the unaffected DoF after the whole machining process. The AMD method uses this matrix as the input and generates a fill-in reducing $m \times m$ permutation matrix $P2_{\text{top\_lef}}$. The permutation matrix $P2_{\text{top\_lef}}$ is extended to a $n \times n$ matrix P2 by inserting ones on the diagonal after the m-th row. The system stiffness matrix permuted by P1 is permuted again with the matrix P2.

The Cholesky factor of the system stiffness matrix can be reused, as shown in Fig. 7. The initial system stiffness matrix K_{initial} is permuted with the permutation matrix $P = P1 \cdot P2$, and the corresponding Cholesky factor is computed. The system stiffness matrix K_q for the intermediate mesh q is obtained by using Eq. 9. The matrix K_q is permuted and the zero columns and rows are deleted according to Eq. 15. The top left submatrix $G_{q\_\text{top\_lef}}$ of the Cholesky factor G_q is equal to the top left submatrix $G_{\text{initial\_top\_lef}}$ of the Cholesky factor G_{initial}, because the corresponding submatrices of the permuted system stiffness matrices are identical. Therefore, the values in

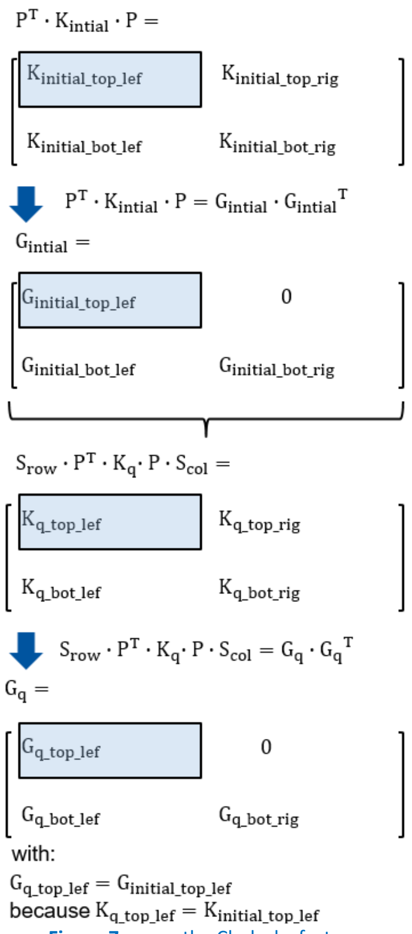


Figure 7. reuse the Cholesky factor

$G_{\text{initial\_top\_lef}}$ can be reused for multiple intermediate meshes to save the computation time.

The presented method to compute the varying Cholesky factor for thin-walled workpieces can be summarized in the following steps:

- Generate the permutation matrix P1 based on the sequence of the element removal.
- Generate the permutation matrix P2 based on the AMD algorithm.
- Compute the Cholesky factor of the initial system stiffness matrix permuted with the permutation matrix $P = P1 \cdot P2$.
- Compute the Cholesky factor for the permuted system stiffness matrix of arbitrary intermediate mesh by reusing the Cholesky factor of the permuted initial system stiffness matrix.

The computed Cholesky factors for the intermediate meshes are used in the next section to get the varying static receptance of the workpiece.

4 METHOD TO CALCULATE THE STATIC RECEPTANCE FOR EACH CUTTER LOCATION

The finite element model of the thin-walled workpiece is assumed to be a linear model in this paper. As a result, the static receptance at each cutter location remains a system characteristic, unrelated to the process force. The calculation of this static receptance involves the following steps (Fig. 8):

1. Identify the influenced but not deleted nodes in the search circle of the radius r around the tool center point. The system stiffness matrix is updated based on the method introduced in section 2 and permuted using the method introduced in section 3.
2. Define a concentrated static force $f_{\text{concentrated}}$ with an arbitrary amplitude in any orthogonal direction.
3. Distribute the concentrated force to multiple identical forces $f_{\text{distributed}}$ on the nodes found in the first step.
4. Calculate the displacement vector of the finite element model under the distributed load defined at the previous step.
5. Extract the displacements of the nodes identified in the first step and compute the average displacement.
6. Compute the static receptance by

$$g_{i,j} = \frac{u_j}{f_i} \tag{17}$$

with:

i: direction of the force

j: direction of the displacement

$g_{i,j}$: static receptance

u_j: average displacement in direction j

f_i: concentrated force in direction i

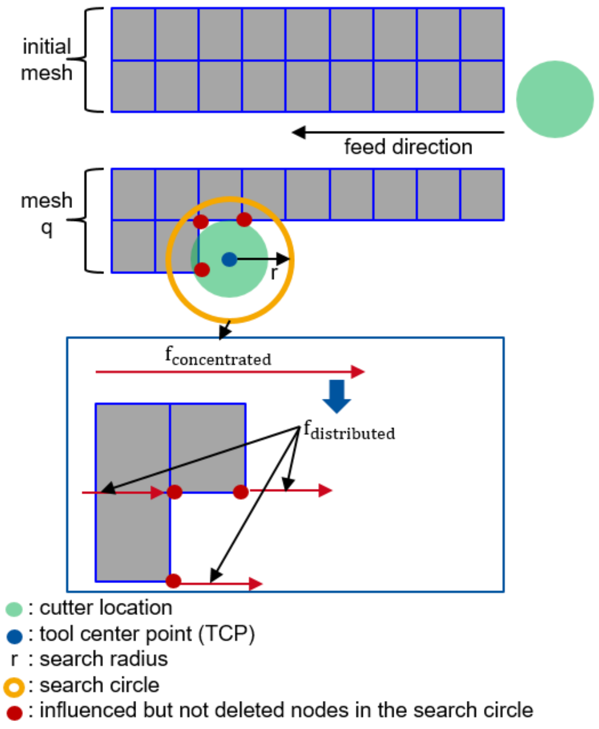


Figure 8. compute the static receptance

It's important to note that Step 4, where the system stiffness matrix's Cholesky factor must be updated for each cutter location, is the most time-consuming part of the process. The method introduced in the previous section aims to mitigate this time requirement by reusing the Cholesky factor. The real efficiency improvement is assessed in the upcoming section through a numerical example.

5 NUMERICAL TEST

In this section, the efficiency improvement achieved by reusing the Cholesky factor is assessed through a numerical example involving the flank milling of a thin wall, as depicted in Fig. 9. The initial mesh comprises 236,715 tetrahedron elements and 47,489 nodes. The stiffness matrix is updated and the static receptance is calculated for 47 CLs. These 47 CLs are evenly spaced along the feed direction. The distance between any two adjacent CLs is 2 mm.

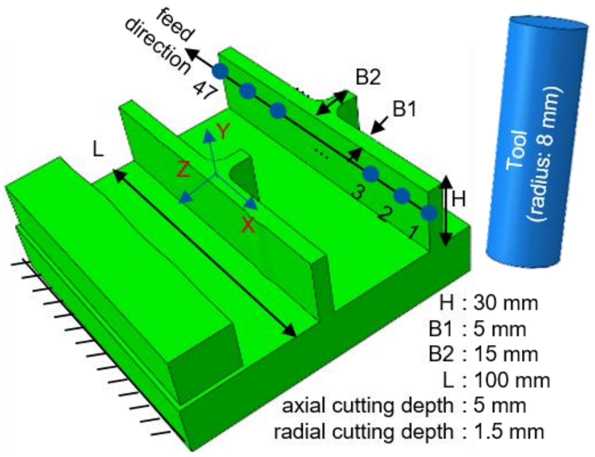


Figure 9. numerical example

The computation times for each CL are detailed in Fig. 10. Utilizing the method that reuses the Cholesky factor results in an average computation time of 33.3 s, compared to an average of 63.1 s without the reuse of the Cholesky factor. This demonstrates a substantial efficiency enhancement with a computation time reduction of 47.2 %.

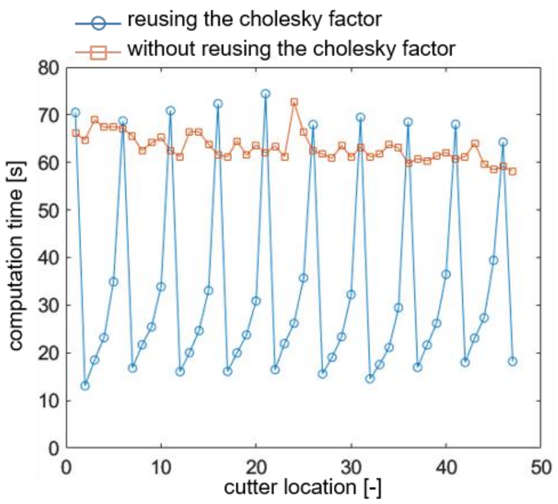


Figure 10. time to compute the static receptance

When not reusing the Cholesky factor, the computation time for a single CL varies within a relatively small range of 58 s to 73 s. On the other hand, the computation time with the new method exhibits a cyclic variation. This can be explained by considering the computation times for the first five CLs. The initial CL necessitates a time-consuming complete Cholesky decomposition, leading to a computation time of around 70 s, which is obviously larger than the computation times for the other four CLs. The second CL, being the most similar to the initial one in terms of the system stiffness matrix, results in the shortest computation time of 13.1 seconds. Subsequently, the computation time increases due to the growing difference between the corresponding system stiffness matrices. This pattern continues until the sixth CL, which is considered to be the new initial mesh for the subsequent five CLs.

The numerical example demonstrates the considerable efficiency improvement brought about by the new method. The next section delves into the accuracy of this method through a machining test.

6 VALIDATION

The objective of this section is to validate the accuracy of the simulated static receptance through a machining test depicted in Fig. 11, corresponding to the numerical example discussed in the previous section. The process force is measured with a force measurement platform. The displacement of the workpiece along the feed direction is computed from the simulated static receptance and the measured process force. The cutting tool exhibits significantly greater stiffness compared to the workpiece. As a result, any deformation of the cutting tool is considered negligible in terms of its impact on the displacement of the workpiece. Furthermore, the workpiece displacement along the feed direction is determined by measuring the finished workpiece with a coordinate measuring machine (CMM). The computed displacement and the displacement determined using the CMM in z direction are nearly

material of the workpiece: C45
tool: 16 mm end mill with 4 blades
spindle speed: 3183 rotation/min
feed rate: 752 mm/min
axial cutting depth: 5 mm
radial cutting depth: 1.5 mm

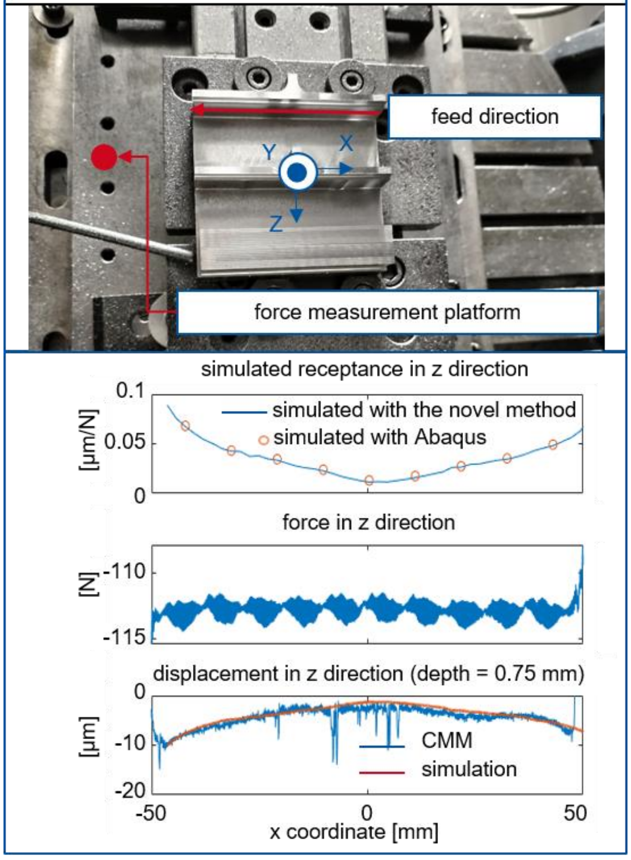


Figure 11. machining test

indistinguishable. Apart from validating through machining test, the workpiece receptance simulated using the novel method is also compared to the receptance simulated with Abaqus. The simulation results demonstrate a good match between the two methods. The machining test and the simulation with Abaqus show that the presented method in this paper can predict the varying static receptance of the thin-walled workpiece with a good accuracy.

7 CONCLUSIONS

An efficient algorithm to simulate the varying static stiffness of the thin-walled workpiece is presented in this paper. The first part of this algorithm is a method to update the system stiffness matrix without remeshing the workpiece geometry. Subsequently, a method to permute the system stiffness matrix based on the sequence of the element removal is introduced. This cutting sequence based permutation method is combined with a fill-in reducing permutation method based on the AMD algorithm. The Cholesky factors for multiple intermediate meshes are computed by reusing the Cholesky factor of the permuted initial system stiffness matrix. These Cholesky factors are used to simulate the varying static receptance of the thin-walled workpiece. A large efficiency improvement is shown in a numerical example by using the new simulation method. Finally, the accuracy of the simulation method is validated by comparing the simulated displacement with the displacement determined using the CMM for a thin-walled workpiece.

The combination of high efficiency and good accuracy underscores the substantial potential of this novel approach. In future work, the method will be applied to real workpieces during CAM planning to validate its efficiency and accuracy under practical conditions.

ACKNOWLEDGMENTS

The authors would like to thank the German Research Foundation (DFG) for supporting this research under the project: methods for efficient and accurate calculation of the static receptance and the eigenpairs of the thin-walled workpiece. The project number is 518689593.

REFERENCES

[Altintas 2018] Altintas, Y., Tuysuz, O., Habibi, M., Li, Z. L.: Virtual compensation of deflection errors in ball end milling of flexible blades. In: Cirp Annals, 2018.

[Amestoy 1996] Amestoy, P. R., Timothy, A. D., Iain, S. D.: An approximate minimum degree ordering algorithm. In: SIAM Journal on Matrix Analysis and Applications, 1996.

[Bathe 2014] Bathe, K. J.: Finite element procedures. In: Watertown, 2014.

[Bolar 2016] Bolar, G., Joshi, S. N.: Three-dimensional numerical modeling, simulation and experimental validation of milling of a thin-wall component. In: Journal of Engineering Manufacture, 2016.

[Budak 1995] Budak, E., Altintas, Y.: Modeling and avoidance of static form errors in peripheral milling of plates. In: International Journal of Machine Tools and Manufacture, 1995.

[Chavan 2020] Chavan, P., Brecher, C., Fey, M., Loba, M.: Workpiece Coupling in Machine Tools Using Experimental-Analytical Dynamic Substructuring. In: Dynamic Substructures. Springer, 2020.

[Cormen 2022] Cormen, T. H., Leiserson, C. E., Rivest, R. L., Stein, C.: Introduction to algorithms. In: MIT press, 2022.

[Denkena 2007] Denkena, B., Schmidt, C.: Experimental investigation and simulation of machining thin-walled workpieces. In: Production engineering, 2007.

[Dittrich 2018] Dittrich, M. A., Böß, V., Wichmann, M., Denkena, B.: Simulation-based compensation of deflection errors in helical flute grinding. In: CIRP Journal of Manufacturing Science and Technology, 2019.

[Khandagale 2018] Khandagale, P., Bhakar, G., Kartik, V., Joshi, S. S.: Modelling time-domain vibratory deflection response of thin-walled cantilever workpieces during flank milling. In: Journal of Manufacturing Processes, 2018.

[Koike 2013] Koike, Y., Matsubara, A., Yamaji, I.: Design method of material removal process for minimizing workpiece displacement at cutting point. In: CIRP Annals, 2013.

[Li 2018] Li, Z. L., Tuysuz, O., Zhu, L. M., Altintas, Y.: Surface form error prediction in five-axis flank milling of thin-walled parts. In: International Journal of Machine Tools and Manufacture, 2018.

[Ratchev 2005] Ratchev, S., Liu, S., Becker, A. A.: Error compensation strategy in milling flexible thin-wall parts. In: Journal of Materials Processing Technology, 2005.

[Scott 2023] Scott, J., Miroslav, T.: Algorithms for sparse linear systems. In: Springer Nature, 2023.

[Wiederkehr 2013] Wiederkehr, P., Biermann, D.: Modeling techniques for simulating workpiece deflections in NC milling. In: CIRP Journal of Manufacturing Science and Technology, 2013.

[Wimmer 2019] Wimmer, S., Hunyadi, P., Zaeh, M. F.: A numerical approach for the prediction of static surface errors in the peripheral milling of thin-walled structures. In: Production Engineering, 2019.

CONTACTS:

M.Sc. Guifeng Zhao
RWTH Aachen University
Laboratory for Machine Tools and Production Engineering
Steinbachstraße 19, Aachen, 52074, Germany
+492418028235
G.Zhao@wzl.rwth-aachen.de