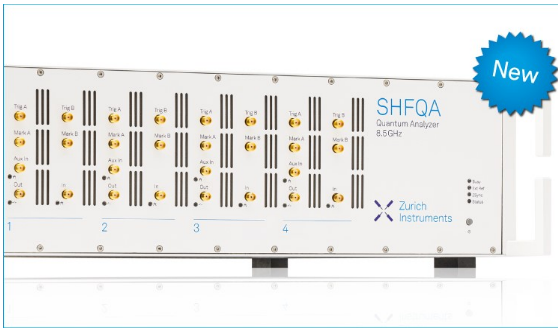
On vectorized MATLAB implementation of elastoplastic problems

Cite as: AIP Conference Proceedings **2293**, 330003 (2020); https://doi.org/10.1063/5.0026561
Published Online: 25 November 2020

Martin Cermák, Stanislav Sysala, and Jan Valdman



On Vectorized MATLAB Implementation of Elastoplastic Problems

Martin Cermák[1,2,3,4,a)], Stanislav Sysala[4,b)] and Jan Valdman[5,6,c)]

[1]Department of Mathematics, Faculty of Civil Engineering, VŠB-TU Ostrava, Ostrava, Czech Republic
[2]Department of Applied Mathematics, FEEIC, VŠB-TU Ostrava, Ostrava, Czech Republic
[3]ENET Centre, VŠB-TU Ostrava, Ostrava, Czech Republic
[4]Institute of Geonics of the Czech Academy of Sciences, Ostrava, Czech Republic
[5]Institute of Mathematics, University of South Bohemia, Ceské Budejovice, Czech Republic
[6]Institute of Information Theory and Automation of the Czech Academy of Sciences, Prague, Czech Republic

[a)]Corresponding author: martin.cermak@vsb.cz
[b)]stanislav.sysala@ugn.cas.cz
[c)]jvaldman@prf.jcu.cz

Abstract. We propose an effective and flexible way to assemble tangent stiffness matrices in MATLAB. Our technique is applied to elastoplastic problems formulated in terms of displacements and discretized by the finite element method. The tangent stiffness matrix is repeatedly assembled in each time step and in each iteration of the semismooth Newton method. We consider von Mises and Drucker-Prager yield criteria, linear and quadratic finite elements in two and three space dimensions. Our codes are vectorized and available for download. Comparisons with other available MATLAB codes show, that our technique is also efficient for purely elastic problems. In elastoplasticity, the assembly times are linearly proportional to the number of integration points.

INTRODUCTION

Vectorization in MATLAB replaces inefficient loops over long arrays by operations with matrices, mainly with sparse matrices. Vectorized codes are then reasonably scalable and fast for large size problems. In this contribution, we deal with a vectorized MATLAB implementation in 2D and 3D proposed in [3] for solution of elastoplastic problems. There are already publicly available MATLAB codes dealing with (pure) elasticity without plasticity [1, 6, 10].

Our implementation arises from a current elastoplastic solution scheme including time discretization by the implicit Euler method, construction of a constitutive operator and its generalized derivatives by the return-mapping algorithm, space discretization by the finite element method, and solution of nonlinear systems of equations by the semismooth Newton method. In [3], the implementation for models including von Mises and Drucker-Prager yield criteria is described in detail. Similar implementation has been used for other yield criteria within numerical examples introduced in recent papers [2, 4, 5, 7, 8, 9].

Further, one can optionally choose P1, P2, Q1 and Q2 finite elements with convenient quadrature rule for numerical integration. To be the codes universal, crucial functions are written uniformly regardless on the choice of elastoplastic models, finite elements or geometries.

The rest of this abstract describes main features of elastoplastic systems of nonlinear equations, assembling of the elastic and tangent stiffness matrices, and illustrative numerical results.

Elastoplastic problems and their solution

Broadly speaking, in each time step of elastoplastic problems we solve a system of nonlinear equations of the following type:

$$\text{find } \boldsymbol{u} \in \mathbb{R}^n : \quad F(\boldsymbol{u}) = \boldsymbol{f}, \tag{1}$$

International Conference of Numerical Analysis and Applied Mathematics ICNAAM 2019
AIP Conf. Proc. 2293, 330003-1–330003-4; https://doi.org/10.1063/5.0026561
Published by AIP Publishing. 978-0-7354-4025-8/$30.00

where $\boldsymbol{u}$ denotes the unknown displacement vector, $\boldsymbol{f} \in \mathbb{R}^n$ is the vector of external forces, and $F\colon \mathbb{R}^n \to \mathbb{R}^n$ is a nonlinear function representing internal forces which is usually Lipschitz continuous and semismooth but nonsmooth in $\mathbb{R}^n$. Therefore, it is necessary to use the semismooth variant of the Newton method, see, e.g., [9]. In each Newton iteration $\ell = 1, 2, \ldots$, we solve a linear system of equations

$$\text{find } \delta\boldsymbol{u}^\ell \in \mathbb{R}^n : \quad \boldsymbol{K}_{tangent}\delta\boldsymbol{u}^\ell = \boldsymbol{f} - F(\boldsymbol{u}^\ell), \tag{2}$$

where $\delta\boldsymbol{u}^\ell \in \mathbb{R}^n$ is an unknown incremental vector, $\boldsymbol{u}^\ell \in \mathbb{R}^n$ is a previous iteration of $\boldsymbol{u}$, and $\boldsymbol{K}_{tangent} \in \mathbb{R}^{n\times n}$ is a tangential stiffness matrix representing a generalized derivative of F at $\boldsymbol{u}^\ell \in \mathbb{R}^n$.

The systems of nonlinear equations (1) have other specific features. First, if the load $\boldsymbol{f}$ is sufficiently small then solutions of elastic and elastoplastic problems usually coincide, i.e., $F(\boldsymbol{u}) = \boldsymbol{K}_{elast}\boldsymbol{u}$, where $\boldsymbol{K}_{elast} \in \mathbb{R}^{n\times n}$ is the corresponding elastic stiffness matrix. Further, for larger loads these solutions significantly differ and in addition, the solution $\boldsymbol{u}$ need not exist for some elastoplastic models due to the presence of limit loads [7, 8, 9]. In vicinity of the limit load, one can also observe locking phenomena and higher order finite elements are recommended. Then, assemblies of F and $\boldsymbol{K}_{tangent}$ require suitable quadrature rules of higher order. Finally, the definition of F is based on solution of the elastoplastic constitutive problems at each integration point of the investigated body. Such solutions (constitutive operators) are given in an implicit form and depend on history of loading. Therefore, constructions of F and $\boldsymbol{K}_{tangent}$ are technically complicated and not straightforward [8, 9].

Assembly of stiffness matrices $\boldsymbol{K}_{elast}$ and $\boldsymbol{K}_{tangent}$

Stiffness matrices based on the finite element method are usually assembled elementwisely by using local stiffness matrices. For example, one can write

$$\boldsymbol{K}_{elast} = \sum_{e=1}^{n_e} \boldsymbol{R}_e^\top \boldsymbol{K}_{e,elast} \boldsymbol{R}_e, \tag{3}$$

where n_e denotes a number of finite elements, $\boldsymbol{R}_e$ is a matrix restricting the displacement vector into its components belonging to a finite element and $\boldsymbol{K}_{e,elast}$ is the local stiffness matrix of the form

$$\boldsymbol{K}_{e,elast} = \sum_{q=1}^{n_q} \omega_{e,q} \boldsymbol{B}_{e,q}^\top \boldsymbol{C}_{e,q} \boldsymbol{B}_{e,q}. \tag{4}$$

Here, n_q is a number of quadrature points at any element, $\omega_{e,q}$ denotes quadrature weights, $\boldsymbol{B}_{e,q}$ is the strain-displacement matrix, and $\boldsymbol{C}_{e,q}$ is the elastic constitutive matrix following from the Hooke's law ($\boldsymbol{C}_{e,q} \in \mathbb{R}^{3\times 3}$ in 2D and $\boldsymbol{C}_{e,q} \in \mathbb{R}^{6\times 6}$ in 3D). For homogeneous materials, $\boldsymbol{C}_{e,q}$ is fixed for any element and any quadrature point.

The assembly of $\boldsymbol{K}_{elast}$ introduced in [3] arises from the following split:

$$\boldsymbol{K}_{elast} = \boldsymbol{B}^\top \boldsymbol{D}_{elast} \boldsymbol{B}, \tag{5}$$

where

$$\boldsymbol{B} = \begin{pmatrix} \boldsymbol{B}_{1,1}\boldsymbol{R}_1 \\ \boldsymbol{B}_{1,2}\boldsymbol{R}_1 \\ \vdots \\ \boldsymbol{B}_{1,n_q}\boldsymbol{R}_1 \\ \boldsymbol{B}_{2,1}\boldsymbol{R}_2 \\ \vdots \\ \vdots \\ \boldsymbol{B}_{n_e,n_q}\boldsymbol{R}_{n_e} \end{pmatrix}, \quad \boldsymbol{D}_{elast} = \begin{pmatrix} \tilde{\boldsymbol{C}}_{1,1} & & & & & & & \\ & \tilde{\boldsymbol{C}}_{1,2} & & & & & & \\ & & \ddots & & & & & \\ & & & \tilde{\boldsymbol{C}}_{1,n_q} & & & & \\ & & & & \tilde{\boldsymbol{C}}_{2,1} & & & \\ & & & & & \ddots & & \\ & & & & & & \ddots & \\ & & & & & & & \tilde{\boldsymbol{C}}_{n_e,n_q} \end{pmatrix},$$

with $\tilde{\boldsymbol{C}}_{e,q} = \omega_{e,q}\boldsymbol{C}_{e,q}$, $e = 1, 2, \ldots, n_e$, $q = 1, 2, \ldots, n_q$. The matrices $\boldsymbol{B}$ and $\boldsymbol{D}_{elast}$ are large and sparse. Moreover, we see that $\boldsymbol{D}_{elast}$ is block diagonal. The multiplications in (5) are possible and convenient in MATLAB if these matrices are defined as sparse.

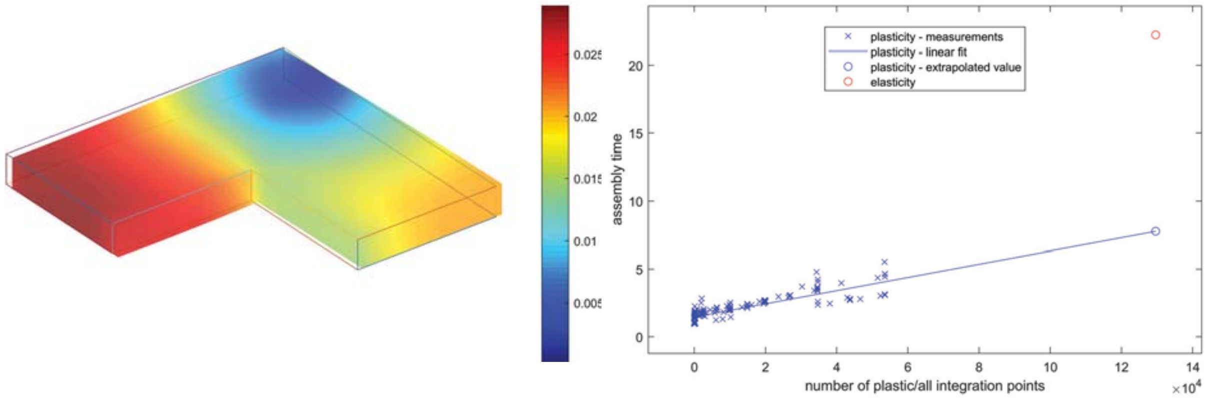


FIGURE 1. 3D problem with the von Mises yield criterion and kinematic hardening. Total displacement (left), assembly times of tangential stiffness matrix versus number of plastic integration points (right).

Similarly, one can assemble the tangent stiffness matrix for an elastoplastic problem:

$$\boldsymbol{K}_{tangent} = \boldsymbol{B}^\top \boldsymbol{D}_{tangent} \boldsymbol{B}. \tag{6}$$

Here, the matrix $\boldsymbol{D}_{tangent}$ has the same size and structure as $\boldsymbol{D}_{elast}$. Each block of $\boldsymbol{D}_{tangent}$ represents a generalized derivative of the elastoplastic constitutive operator at any integration point. Moreover, one can write [3]:

$$\boldsymbol{K}_{tangent} = \boldsymbol{K}_{elast} + \boldsymbol{B}^\top (\boldsymbol{D}_{tangent} - \boldsymbol{D}_{elast}) \boldsymbol{B}, \tag{7}$$

Although (6) and (7) are algebraically identical, the form (7) is more convenient for MATLAB implementation since the sparse matrix $\boldsymbol{D}_{tangent} - \boldsymbol{D}_{elast}$ is typically sparser than $\boldsymbol{D}_{tangent}$. This occurs when most of integration points remains in the elastic phase. Therefore, for problems with smaller plastic regions, the assembly of the tangential stiffness matrix can be faster than for problems with larger plastic regions, see Figure 1 (left).

Finally, it is important to note that the matrices $\boldsymbol{K}_{elast}$, $\boldsymbol{B}$, $\boldsymbol{D}_{elast}$ can be precomputed and only the matrix $\boldsymbol{D}_{tangent}$ depends on a particular plasticity model and needs to be partially reassembled in each Newton iteration. Additionally, $\boldsymbol{B}$ can be also used for the assembly of the function F.

Illustrative numerical results

The first illustrative result is depicted in Figure 1. It is considered a 3D problem with L-shaped geometry and cycling loading. The body obeys the associative plastic flow rule and the linear kinematic hardening law. The von Mises yield criterion is used. The left figure visualizes the total displacement. The right figure compares assembly times of $\boldsymbol{K}_{tangent}$ at particular time steps and Newton iterations. These times do not include the assembly of the elastic stiffness matrix $\boldsymbol{K}_{elast}$ which is precomputed and fixed. We see that the assembly times of $\boldsymbol{K}_{tangent}$ linearly depend on the numbers of elements with plastic response and are always lower than the assembly time of $\boldsymbol{K}_{elast}$.

The second illustrative result is depicted in Figure 2. It is considered a strip-footing problem under the plane strain assumption. The aim is to analyze bearing capacity of a soil foundation and visualize the plastic collapse of the body. Monotone displacement loading is prescribed on the left part of the top. The body is perfectly plastic with the Drucker-Prager yield criterion. Failure mechanism is visualized by displacement fields and deformed shape. We observe significant jumps in displacement fields. The interface between the blue and yellow regions defines the expected failure zone.

CONCLUSION

The paper is focused on an efficient and flexible implementation of elastoplastic problems. We have mainly proposed the innovative assembly of elastoplastic FEM matrices based on the split (7). Additional effort to build

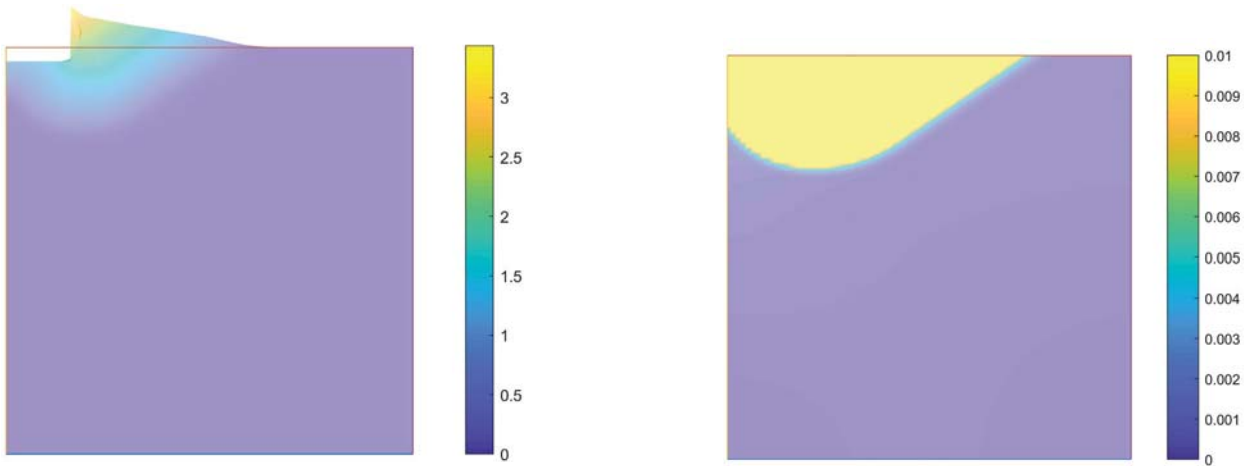


FIGURE 2. Strip-footing 2D problem solved by perfect plasticity with the Drucker-Prager yield criterion. Failure mechanism is visualized by the deform shape (left) and jumps in displacement fields (right).

the tangential stiffness matrices in each Newton iteration and each time step of elastoplastic problems does not exceed the cost for the elastic stiffness matrix. The smaller is the number of the plastic integrations points, the faster is the assembly. Our techniques are explained and implemented in the vectorized code available for download at `https://github.com/matlabfem/matlab_fem_elastoplasticity`.

ACKNOWLEDGMENT

This work has been supported by the Czech Science Foundation, project No. 17-04301S and 19-11441S; to VSB-Technical University of Ostrava by the Czech Ministry of Education, Youth and Sports from the budget for conceptual development of science, research and innovations for the year 2019; by The Ministry of Education, Youth and Sports of the Czech Republic – project LO1404 – Sustainable development of CENET "Z.1.05/2.1.00/19.0389: Research Infrastructure Development of the CENET".

REFERENCES

[1] J. Alberty, C. Carstensen, S.A. Funken, and R. Klose. *Matlab implementation of the finite element method in elasticity.* Computing **69(3)** (2002) 239–263.

[2] M. Čermák, T. Kozubek, S. Sysala, J. Valdman. *A TFETI domain decomposition solver for elastoplastic problems.* Applied Mathematics and Computation **231** (2014) 634–653.

[3] M. Čermák, S. Sysala, and J. Valdman. *Efficient and flexible Matlab implementation of 2D and 3D elasto-plastic problems.* Applied Mathematics and Computation **355** (2019) 595–614.

[4] M. Čermák. *Elastoplastic one dimensional problem.* AIP Conference Proceedings **2116** (2019) art. no. 320002.

[5] P. Horyl, R. Šňupárek, P. Maršálek. *Behaviour of frictional joints in steel arch yielding supports.* Archives of Mining Sciences, **59** (2014), 781–792.

[6] J. Koko. *Fast MATLAB assembly of fem matrices in 2D and 3D using cell-array approach.* International Journal of Modeling, Simulation, and Scientific Computing **7(02)** (2016) 1650010.

[7] S. Repin. S. Sysala and J. Haslinger: *Computable majorants of the limit load in Hencky's plasticity problems.* Computer and Mathematics with Applications **75** (2018) 199–217.

[8] S. Sysala, M. Čermák, T. Koudelka, J. Kruis, J. Zeman, and R. Blaheta. *Subdifferential- based implicit return-mapping operators in computational plasticity.* Zeitschrift für Angewandte Mathematik und Mechanik **96** (2016) 1318–1338.

[9] S. Sysala, M. Čermák, and T. Ligurský. *Subdifferential-based implicit return-mapping op- erators in Mohr-Coulomb plasticity.* Zeitschrift für Angewandte Mathematik und Mechanik, **97** (2017) 1502–1523.

[10] T. Rahman, J. Valdman, *Fast MATLAB assembly of FEM matrices in 2D and 3D: nodal elements*. Applied Mathematics and Computation **219(13)** (2013), 7151–7158.