ASIAN JOURNAL OF CIVIL ENGINEERING (BUILDING AND HOUSING) VOL. 13, NO. 2 (2012)
PAGES 203-232

SECOND ORDER INTEGRATION ALGORITHMS FOR J_2 PLASTICITY WITH GENERALIZED ISOTROPIC/KINEMATIC HARDENING

M. Jahanshahi*
Department of Civil Engineering, School of Science and Engineering, Sharif University of Technology, International Campus, Kish Island, Iran

Received: 5 January 2011; **Accepted:** 25 April 2011

ABSTRACT

In this work four algorithms are presented for the integration of equations governing the behavior of von Mises material in plastic limit. Two of them are single-step and the others are double-step algorithms. Single-step algorithms are straight forward extension of backward Euler method or midpoint rule while double-step algorithms are a combination of two. In past similar algorithms were proposed for special cases of isotropic/kinematic hardening; however the algorithms presented in this work are applicable to general isotropic/kinematic hardening laws. Theoretical as well as numerical aspects are presented for all integration schemes and elaborated for double-step algorithms which are less explored in the literature. A comparison is made between the presented algorithms and the classical backward Euler method.

Keywords: Plasticity; backward euler method; midpoint rule; second order; nonlinear hardening

1. INTRODUCTION

The evolution of integration algorithms dates back to the pioneering work of Wilkins [15]. The radial return algorithm that he proposed in his work is the basis of many algorithms developed later. His algorithm was actually a backward Euler type of algorithm for the integration of first order ordinary differential equations as a function of time. Backward Euler algorithm has been used successfully in the work of many authors and has proven its applicability in many research and commercial softwares. Midpoint rule algorithm was first proposed in the work of Popov and Ortiz [8]. They also presented a method for rigorously investigating the stability of algorithm which was later modified and corrected in the work of Simo and Govindjee [11]. These authors used the notion of B-Stability (A-Contractivity)

* E-mail address of the corresponding author: jahanshahi@sharif.edu (M. Jahanshahi)

in the stability analysis of a given algorithm. Double step algorithms with the sense used today (enforcement of two consistency conditions per time step) were introduced by Simo [10], but were not elaborated until recently. Auricchio *et al.* used double step methods for linear hardening and a type of nonlinear hardening in some detail [3, 4]. They also proposed an exponential type of algorithm which outperforms the single and double step integration algorithms but is computationally expensive [1]. Besides the algorithms mentioned previously other integration algorithms such as Runge-Kutta and multi-step methods are available which are rather cumbersome to implement [10].

Abundant literature is available on the application of single and double step algorithms to problems with linear and special cases of nonlinear hardening. But a detailed discussion on the application of these algorithms to problems with general isotropic/kinematic hardening is missing especially for double step methods. In this work four integration algorithms are presented for the integration of governing equations of a J_2 material with general isotropic/kinematic hardening. Theoretical and numerical aspects are discussed in detail and elaborated for double step methods.

The four algorithms are labeled in [2] respectively as SMPT1, SMPT2, DMPT1 and DMPT2 and for easy reference the same labels are used here. It should be noted that in their work these algorithms are used for a linear type of hardening whilst the type of hardening (isotropic/kinematic) in this work can be general. The first two integration algorithms are single step methods and are based on the classical midpoint integration rule. On the other hand, each time step in the last two methods consists of two sub-steps and the equations are sequentially solved for each sub-step [2].

The aim of this paper is to present the formulation of the aforementioned integration schemes for general isotropic/kinematic hardening and to investigate the numerical aspects of each. The results of different algorithms are compared with backward Euler method as benchmark and their behavior is studied through error graphs and iso-error maps. The focus will be on double step methods, DMPT2 in particular, as it is believed that if it is correctly tuned it can operate faster and more accurate than others.

The paper is organized as follows. In Section 2 the time continuous model under consideration is presented. The formulation of four integration algorithms as well as the enforcement of consistency condition for each algorithm is discussed in Section 3. In Section 4 numerical examples are used to investigate the behavior of algorithms. Efficiency and accuracy of algorithms are studied through error graphs and iso-error maps. Finally the detailed derivation of tangent operators is presented in appendix A.

2. TIME CONTINUOUS MODEL

The model considered in this work is associative von Mises plasticity with general isotropic/kinematic hardening employing small deformations theory. If the stress and strain tensors are split into deviatoric and volumetric parts we can write

$$\boldsymbol{\sigma} = \mathbf{s} + \mathrm{p}\mathbf{I} \ \text{ with } \ \mathrm{p} = \frac{1}{3}\mathrm{tr}(\boldsymbol{\sigma}) \tag{1}$$

$$\boldsymbol{\varepsilon} = \mathbf{e} + \frac{1}{3}\theta\mathbf{I} \text{ with } \theta = \mathrm{tr}(\boldsymbol{\varepsilon}) \tag{2}$$

$$p = \kappa\theta \tag{3}$$

where tr is the trace operator, **I**, **s**, **e**, p and θ are, respectively, the second order identity tensor, the deviatoric stress and strain tensors, the volumetric stress and strain terms and finally κ is the bulk modulus.

Based on the previous splitting of the stress and strain tensors, the equations of the model can be written as [12]:

$$\mathbf{s} = 2G(\mathbf{e} - \mathbf{e}^p) \tag{4}$$

$$\boldsymbol{\xi} = \mathbf{s} - \boldsymbol{\beta} \tag{5}$$

$$f = \|\boldsymbol{\xi}\| - \sqrt{\frac{2}{3}}K(\alpha) \tag{6}$$

$$\dot{\mathbf{e}}^p = \gamma\mathbf{n} \tag{7}$$

$$\dot{\boldsymbol{\beta}} = \frac{2}{3}H'(\alpha)\,\gamma\,\mathbf{n} \tag{8}$$

$$\dot{\alpha} = \sqrt{\frac{2}{3}}\gamma \tag{9}$$

$$\gamma \geq 0, \quad f \leq 0, \quad \gamma f = 0 \tag{10}$$

where G is the shear modulus, **e** is the total strain, $\mathbf{e}^p$ is the plastic strain whose trace is zero, **ξ** is the relative stress, **β** is the center of yield surface in stress deviator space [12], f is the von Mises yield function, **n** is the normal to the yield surface, α is the equivalent plastic strain and the functions $K'(\alpha)$ and $H'(\alpha)$ are called the isotropic and kinematic hardening modulus respectively. Finally, equations (10) are the so called Kuhn-Tucker conditions. Since $\|\dot{\mathbf{e}}^p\| = \gamma$, relationship (9) implies that

$$\alpha = \int_0^t \sqrt{\frac{2}{3}}\|\dot{\mathbf{e}}^p(\tau)\|d\tau \tag{11}$$

which agrees with the usual definition of equivalent plastic strain.

Isotropic and kinematic hardening can be linear as is widely used in the literature. Moreover, nonlinear isotropic/kinematic hardening models are considered in which a

saturation hardening term of the exponential type is appended to linear terms [12]. The type of isotropic/kinematic hardening considered in this work is

$$\begin{gathered} h(\alpha) = \overline{K}_\infty - \left[\overline{K}_\infty - \overline{K}_0\right]\exp(-\delta\alpha) + \overline{H}'\alpha \\ K(\alpha) = \beta h(\alpha) \\ H(\alpha) = (1-\beta)h(\alpha),\ \beta \in [0, 1] \end{gathered} \tag{12}$$

where $\overline{K}_\infty$, $\overline{K}_0$, δ and $\overline{H}'$ are material constants and β provides a modulation between pure kinematic ($\beta = 0$) and pure isotropic ($\beta = 1$) hardening.

3. INTEGRATION ALGORITHMS

In this section the details of four integration algorithms introduced in previous sections are presented. These algorithms are categorized as single step and double step algorithms. In single step algorithms return mapping procedure is performed once per integration step, while in double step algorithms the integration step is divided into two sub-steps and return mapping is performed for each sub-step. As denoted in [2], these algorithms are called SMPT1 for single step midpoint method no. 1, SMPT2 for single step midpoint method no. 2, DMPT1 for double step midpoint method no. 1 and DMPT2 for double step midpoint method no. 2.

The loading history over the interval [0, T] is divided into N subintervals defined by time instants $0 = t_0 < t_1 < \ldots < t_n < t_{n+1} < \ldots < t_N = T$. An intermediate time integration interval is represented by $\Delta t = t_{n+1} - t_n$ and it is assumed to be constant for each loading step. One can choose the midpoint instant $t_{n+\alpha}$ for each subinterval $[t_n, t_{n+1}]$ such that $t_n \le t_{n+\alpha} \le t_{n+1}$ and

$$\alpha = \frac{t_{n+\alpha} - t_n}{\Delta t} \tag{13}$$

Given the variables $\{\mathbf{s}_n, \mathbf{e}_n, \mathbf{e}^p_n, \boldsymbol{\beta}_n, \alpha_n\}$ at time t_n and the deviatoric strain $\mathbf{e}_{n+1}$ at time t_{n+1}, the objective is to devise a numerical algorithm to compute the variables at time t_{n+1}.

3.1 SMPT1 and SMPT2 algorithms

In order to integrate the set of governing equations of the system using SMPT1 and SMPT2 algorithms, a generalized midpoint rule is applied over the interval $[t_n, t_{n+1}]$. The consistency condition in SMPT1 algorithm is enforced at the end of time interval, while it is enforced at midpoint instant $t_{n+\alpha}$ in SMPT2 algorithm. In both algorithms the enforcement of consistency condition is accomplished along the normal to the yield surface at midpoint instant $t_{n+\alpha}$. The algorithmic setup for SMPT1 and SMPT2 algorithms is as follows

$$\begin{gathered}
\mathbf{e}^{p}_{n+1} = \mathbf{e}^{p}_{n} + \lambda \mathbf{n}_{n+\alpha} \\
\mathbf{s}_{n+1} = 2G(\mathbf{e}_{n+1} - \mathbf{e}^{p}_{n+1}) \\
\boldsymbol{\beta}_{n+1} = \boldsymbol{\beta}_{n} + \sqrt{\frac{2}{3}}\Delta H_{n+1}\mathbf{n}_{n+\alpha},\ \Delta H_{n+1} = H(\alpha_{n+1}) - H(\alpha_{n}) \\
\boldsymbol{\xi}_{n+1} = \mathbf{s}_{n+1} - \boldsymbol{\beta}_{n+1} \\
\alpha_{n+1} = \alpha_{n} + \sqrt{\frac{2}{3}}\lambda,\ \lambda = \Delta t\, \gamma_{n+\alpha}
\end{gathered} \tag{14}$$

The scalar λ is the incremental plastic multiplier and is determined by enforcing the consistency condition $f(\boldsymbol{\xi}_{n+1}) = 0$ for SMPT1 and $f(\boldsymbol{\xi}_{n+\alpha}) = 0$ for SMPT2. The tensor $\mathbf{n}_{n+\alpha}$ is the normal to the yield surface at midpoint and can be computed by evaluating the quantities at $t_{n+\alpha}$ using the following linear equations.

$$\begin{aligned}
\mathbf{e}_{n+\alpha} &= \alpha \mathbf{e}_{n+1} + (1-\alpha)\mathbf{e}_{n} \\
\mathbf{e}^{p}_{n+\alpha} &= \alpha \mathbf{e}^{p}_{n+1} + (1-\alpha)\mathbf{e}^{p}_{n} \\
\mathbf{s}_{n+\alpha} &= \alpha \mathbf{s}_{n+1} + (1-\alpha)\mathbf{s}_{n} \\
\boldsymbol{\beta}_{n+\alpha} &= \alpha \boldsymbol{\beta}_{n+1} + (1-\alpha)\boldsymbol{\beta}_{n} \\
\boldsymbol{\xi}_{n+\alpha} &= \alpha \boldsymbol{\xi}_{n+1} + (1-\alpha)\boldsymbol{\xi}_{n}
\end{aligned} \tag{15}$$

In order to solve the set of equations (14), it is initially assumed that the evolution of variables over $[t_n, t_{n+1}]$ is elastic. Based on this assumption, the trial step for both algorithms can be written as

$$\begin{gathered}
\mathbf{e}^{p,trial}_{n+1} = \mathbf{e}^{p}_{n} \\
\mathbf{s}^{trial}_{n+1} = \mathbf{s}_{n} + 2G\Delta\mathbf{e}_{n+1},\ \Delta\mathbf{e}_{n+1} = \mathbf{e}_{n+1} - \mathbf{e}_{n} \\
\boldsymbol{\beta}^{trial}_{n+1} = \boldsymbol{\beta}_{n} \\
\boldsymbol{\xi}^{trial}_{n+1} = \mathbf{s}^{trial}_{n+1} - \boldsymbol{\beta}^{trial}_{n+1} \\
\alpha^{trial}_{n+1} = \alpha_{n}
\end{gathered} \tag{16}$$

If the trial relative stress satisfies the yield condition, i.e.

$$\left\|\boldsymbol{\xi}^{trial}_{n+1}\right\| - \sqrt{\frac{2}{3}}K\left(\alpha^{trial}_{n+1}\right) \le 0 \ \text{ for SMPT1} \tag{17}$$

and

$$\left\|\boldsymbol{\xi}^{trial}_{n+\alpha}\right\| - \sqrt{\frac{2}{3}}K\left(\alpha^{trial}_{n+\alpha}\right) \le 0 \ \text{ for SMPT2} \tag{18}$$

the whole step is elastic and the trial variables are considered to be the correct ones. Otherwise if the relative stress does not satisfy the yield condition, a plastic correction as

specified by the following equations should be applied.

$$\begin{aligned}
\mathbf{e}^{p}_{n+1} &= \mathbf{e}^{p,trial}_{n+1} + \lambda \mathbf{n}_{n+\alpha} \\
\mathbf{s}_{n+1} &= \mathbf{s}^{trial}_{n+1} - 2G\lambda \mathbf{n}_{n+\alpha} \\
\boldsymbol{\beta}_{n+1} &= \boldsymbol{\beta}^{trial}_{n+1} + \sqrt{\frac{2}{3}}\Delta H_{n+1}\mathbf{n}_{n+\alpha} \\
\boldsymbol{\xi}_{n+1} &= \boldsymbol{\xi}^{trial}_{n+1} - \left(2G\lambda + \sqrt{\frac{2}{3}}\Delta H_{n+1}\right)\mathbf{n}_{n+\alpha} \\
\alpha_{n+1} &= \alpha^{trial}_{n+1} + \sqrt{\frac{2}{3}}\lambda
\end{aligned} \tag{19}$$

The normal tensor $\mathbf{n}_{n+\alpha} = \boldsymbol{\xi}_{n+\alpha} / \left\| \boldsymbol{\xi}_{n+\alpha} \right\|$ can be computed by observing that at midpoint the trial relative stress is obtained using

$$\begin{aligned}
\mathbf{s}^{trial}_{n+\alpha} &= \mathbf{s}_{n} + 2G\alpha\Delta\mathbf{e}_{n+1} \\
\boldsymbol{\beta}^{trial}_{n+\alpha} &= \boldsymbol{\beta}_{n} \\
\boldsymbol{\xi}^{trial}_{n+\alpha} &= \mathbf{s}^{trial}_{n+\alpha} - \boldsymbol{\beta}^{trial}_{n+\alpha}
\end{aligned} \tag{20}$$

and the updated relative stress using

$$\begin{aligned}
\mathbf{s}_{n+\alpha} &= \mathbf{s}^{trial}_{n+\alpha} - 2G\alpha\lambda \mathbf{n}_{n+\alpha} \\
\boldsymbol{\beta}_{n+\alpha} &= \boldsymbol{\beta}^{trial}_{n+\alpha} + \sqrt{\frac{2}{3}}\alpha\Delta H_{n+1}\mathbf{n}_{n+\alpha} \\
\boldsymbol{\xi}_{n+\alpha} &= \boldsymbol{\xi}^{trial}_{n+\alpha} - \alpha\left(2G\lambda + \sqrt{\frac{2}{3}}\Delta H_{n+1}\right)\mathbf{n}_{n+\alpha}
\end{aligned} \tag{21}$$

Equation $(21)_3$ implies that the following co-alignment relation exists between $\mathbf{n}_{n+\alpha}$, $\boldsymbol{\xi}_{n+\alpha}$ and $\boldsymbol{\xi}^{trial}_{n+\alpha}$.

$$\mathbf{n}_{n+\alpha} = \frac{\boldsymbol{\xi}_{n+\alpha}}{\left\| \boldsymbol{\xi}_{n+\alpha} \right\|} = \frac{\boldsymbol{\xi}^{trial}_{n+\alpha}}{\left\| \boldsymbol{\xi}^{trial}_{n+\alpha} \right\|} \tag{22}$$

The plastic multiplier λ is computed by enforcing the consistency condition at the end of time interval for SMPT1 and at midpoint for SMPT2. This is done by requiring that the yield function be satisfied for the appropriate relative stress. In the case of SMPT1 algorithm we have

$$f_{n+1} = \|\xi_{n+1}\| - \sqrt{\frac{2}{3}}K(\alpha_{n+1}) = 0 \tag{23}$$

However, since the consistency condition is enforced along $\mathbf{n}_{n+\alpha}$, it is necessary to implicitly (or explicitly) express the equation (23) in terms of $\mathbf{n}_{n+\alpha}$. Hence, it is more convenient to rewrite this equation as a function of λ in the following useful form.

$$g(\lambda) = \xi_{n+1} : \xi_{n+1} - \frac{2}{3}K^2(\alpha_{n+1}) = 0 \tag{24}$$

It should be noted that ξ_{n+1} and K can be highly nonlinear functions of α_{n+1}. Therefore in order to solve the equation (24) for λ we should recourse to numerical solution. The derivative of equation (24) with respect to λ is

$$g'(\lambda) = 2\frac{d\xi_{n+1}}{d\lambda} : \xi_{n+1} - \frac{4}{3}\sqrt{\frac{2}{3}}K(\alpha_{n+1})K'(\alpha_{n+1}) \tag{25}$$

Computing the derivative of ξ_{n+1} with respect to λ with the help of equations $(15)_5$ and $(21)_3$ and substituting into equation (25) results in the following equation.

$$g'(\lambda) = -2\left[2G + \frac{2}{3}H'(\alpha_{n+1})\right]\xi_{n+1} : \mathbf{n}_{n+\alpha} - \frac{4}{3}\sqrt{\frac{2}{3}}K(\alpha_{n+1})K'(\alpha_{n+1}) \tag{26}$$

Given $g(\lambda)$ and $g'(\lambda)$, the k^{th} step of the numerical solution of λ for SMPT1 algorithm can be summarized as follows.

$$\begin{aligned}
g(\lambda_k) &= \xi_{n+1}^k : \xi_{n+1}^k - \frac{2}{3}K^2(\alpha_{n+1}^k) \\
g'(\lambda_k) &= -2\left\{\left[2G + \frac{2}{3}H'(\alpha_{n+1}^k)\right]\xi_{n+1}^k : \mathbf{n}_{n+\alpha} + \frac{2}{3}\sqrt{\frac{2}{3}}K(\alpha_{n+1}^k)K'(\alpha_{n+1}^k)\right\} \\
\Delta\lambda_{k+1} &= -\frac{g(\lambda_k)}{g'(\lambda_k)} \\
\lambda_{k+1} &= \lambda_k + \Delta\lambda_{k+1} = \lambda_k - \frac{g(\lambda_k)}{g'(\lambda_k)}
\end{aligned} \tag{27}$$

where ξ_{n+1}^k and α_{n+1}^k are the updated values of ξ_{n+1} and α_{n+1} in step k and are computed using the following relations.

$$\boldsymbol{\xi}_{n+1}^{k} = \boldsymbol{\xi}_{n+1}^{trial} - \left(2G\lambda_k + \sqrt{\frac{2}{3}}\Delta H_{n+1}^{k} \right) \mathbf{n}_{n+\alpha},\ \Delta H_{n+1}^{k} = H(\alpha_{n+1}^{k}) - H(\alpha_n)$$
$$\alpha_{n+1}^{k} = \alpha_{n+1}^{trial} + \sqrt{\frac{2}{3}}\lambda_k \tag{28}$$

The procedure for solving λ should be repeated until $\Delta\lambda_{k+1}$ becomes smaller than a predefined tolerance. In the case of SMPT2 algorithm, the consistency condition to be satisfied is

$$f_{n+\alpha} = \left\| \boldsymbol{\xi}_{n+\alpha} \right\| - \sqrt{\frac{2}{3}} K(\alpha_{n+\alpha}) = 0 \tag{29}$$

Observing equations (21) and (22) it is evident that the equation (29) as a function of λ can be written in the following form.

$$g(\lambda) = \left\| \boldsymbol{\xi}_{n+\alpha}^{trial} \right\| - \left(2G\alpha\lambda + \sqrt{\frac{2}{3}}\Delta H_{n+\alpha} \right) - \sqrt{\frac{2}{3}} K(\alpha_{n+\alpha}) = 0 \tag{30}$$

Taking the derivative of equation (30) with respect to λ results in the following equation.

$$g'(\lambda) = -2G\alpha \left[1 + \frac{H'(\alpha_{n+\alpha}) + K'(\alpha_{n+\alpha})}{3G} \right] \tag{31}$$

Based on equations (30) and (31) the k^{th} step of the numerical solution of λ for SMPT2 algorithm can be summarized as follows.

$$\begin{aligned} g(\lambda_k) &= \left\| \boldsymbol{\xi}_{n+\alpha}^{trial} \right\| - \left(2G\alpha\lambda_k + \sqrt{\frac{2}{3}}\Delta H_{n+\alpha}^{k} \right) - \sqrt{\frac{2}{3}} K(\alpha_{n+\alpha}^{k}) \\ g'(\lambda_k) &= -2G\alpha \left[1 + \frac{H'(\alpha_{n+\alpha}^{k}) + K'(\alpha_{n+\alpha}^{k})}{3G} \right] \\ \Delta\lambda_{k+1} &= -\frac{g(\lambda_k)}{g'(\lambda_k)} \\ \lambda_{k+1} &= \lambda_k + \Delta\lambda_{k+1} = \lambda_k - \frac{g(\lambda_k)}{g'(\lambda_k)} \end{aligned} \tag{32}$$

where $\Delta H_{n+\alpha}^{k}$ and $\alpha_{n+\alpha}^{k}$ are the updated values of $\Delta H_{n+\alpha}$ and $\alpha_{n+\alpha}$ in step k and are computed using the following relations.

$$\begin{aligned} \Delta H^{k}_{n+\alpha} &= H(\alpha^{k}_{n+\alpha}) - H(\alpha_n) \\ \alpha^{k}_{n+\alpha} &= \alpha^{trial}_{n+\alpha} + \sqrt{\frac{2}{3}}\alpha\lambda_k \end{aligned} \tag{33}$$

Pictorial illustrations of SMPT1 and SMPT2 algorithms as well as a detailed discussion about these algorithms for linear isotropic/kinematic hardening can be found in [2].

3.2 DMPT1 and DMPT2 algorithms

Double step algorithms perform the return mapping procedure twice per time interval, first at $t_{n+\alpha}$ and then at t_{n+1}. Consequently the amount of computations required to accomplish a single step of DMPT1 and DMPT2 algorithms is dramatically increased compared with SMPT1 and SMPT2 algorithms. In the first step, both algorithms perform a backward Euler integration over the subinterval $[t_n, t_{n+\alpha}]$. This is the presence of the second step which makes the two algorithms different from each other.

The second step of DMPT1 algorithm comprises of a midpoint integration rule that is applied to the time interval $[t_n, t_{n+1}]$. The consistency condition is enforced at the end of time interval along the normal tensor $\mathbf{n}_{n+\alpha}$ which is computed in the first step. Conversely, as the second step, DMPT2 performs an extrapolation of the results obtained in the first step and if necessary projects them onto the yield surface [2].

It was mentioned that applying a backward Euler integration rule over the subinterval $[t_n, t_{n+\alpha}]$ constitutes the first step of DMPT1 and DMPT2 algorithms. Therefore the algorithmic setup for the first step can be formulated as follows.

$$\begin{aligned} \mathbf{e}^{p}_{n+\alpha} &= \mathbf{e}^{p}_{n} + \lambda_1 \mathbf{n}_{n+\alpha} \\ \mathbf{s}_{n+\alpha} &= 2G(\mathbf{e}_{n+\alpha} - \mathbf{e}^{p}_{n+\alpha}) \\ \boldsymbol{\beta}_{n+\alpha} &= \boldsymbol{\beta}_n + \sqrt{\frac{2}{3}}\Delta H_{n+\alpha}\mathbf{n}_{n+\alpha},\ \Delta H_{n+\alpha} = H(\alpha_{n+\alpha}) - H(\alpha_n) \\ \boldsymbol{\xi}_{n+\alpha} &= \mathbf{s}_{n+\alpha} - \boldsymbol{\beta}_{n+\alpha} \\ \alpha_{n+\alpha} &= \alpha_n + \sqrt{\frac{2}{3}}\lambda_1,\ \lambda_1 = \Delta t\, \gamma_{n+\alpha} \end{aligned} \tag{34}$$

where λ_1 is the plastic multiplier for the time interval $[t_n, t_{n+\alpha}]$ and is computed by enforcing the consistency condition at time instant $t_{n+\alpha}$.

The procedure for solving the set of equations (34) is similar to the one used for SMPT1 and SMPT2 algorithms. In other words, it is initially assumed that the evolution of variables over the time interval $[t_n, t_{n+\alpha}]$ is elastic and if the trial values violate the consistency condition, a plastic correction is employed. Based on the elastic hypothesis the trial step is

$$\begin{gathered}\mathbf{e}_{n+\alpha}^{p,trial}=\mathbf{e}_{n}^{p}\\ \mathbf{s}_{n+\alpha}^{trial}=\mathbf{s}_{n}+2G\Delta\mathbf{e}_{n+\alpha},\ \Delta\mathbf{e}_{n+\alpha}=\mathbf{e}_{n+\alpha}-\mathbf{e}_{n}\\ \boldsymbol{\beta}_{n+\alpha}^{trial}=\boldsymbol{\beta}_{n}\\ \boldsymbol{\xi}_{n+\alpha}^{trial}=\mathbf{s}_{n+\alpha}^{trial}-\boldsymbol{\beta}_{n+\alpha}^{trial}\\ \alpha_{n+\alpha}^{trial}=\alpha_{n}\end{gathered}\tag{35}$$

If the trial relative stress satisfies the yield condition, i.e.

$$\left\|\boldsymbol{\xi}_{n+\alpha}^{trial}\right\|-\sqrt{\frac{2}{3}}K\left(\alpha_{n+\alpha}^{trial}\right)\le 0\tag{36}$$

the time interval $[t_n, t_{n+\alpha}]$ is elastic and the trial variables are considered to be the correct ones. Otherwise if the relative stress does not satisfy the yield condition, a plastic correction as specified by the following equations should be applied.

$$\begin{gathered}\mathbf{e}_{n+\alpha}^{p}=\mathbf{e}_{n+\alpha}^{p,trial}+\lambda_{1}\mathbf{n}_{n+\alpha}\\ \mathbf{s}_{n+\alpha}=\mathbf{s}_{n+\alpha}^{trial}-2G\lambda_{1}\mathbf{n}_{n+\alpha}\\ \boldsymbol{\beta}_{n+\alpha}=\boldsymbol{\beta}_{n+\alpha}^{trial}+\sqrt{\frac{2}{3}}\Delta H_{n+\alpha}\mathbf{n}_{n+\alpha}\\ \boldsymbol{\xi}_{n+\alpha}=\boldsymbol{\xi}_{n+\alpha}^{trial}-\left(2G\lambda_{1}+\sqrt{\frac{2}{3}}\Delta H_{n+\alpha}\right)\mathbf{n}_{n+\alpha}\\ \alpha_{n+\alpha}=\alpha_{n+\alpha}^{trial}+\sqrt{\frac{2}{3}}\lambda_{1}\end{gathered}\tag{37}$$

The plastic multiplier λ_1 is calculated by enforcing the consistency condition at time instant $t_{n+\alpha}$, i.e. the following equation should be satisfied.

$$f_{n+\alpha}=\left\|\boldsymbol{\xi}_{n+\alpha}\right\|-\sqrt{\frac{2}{3}}K(\alpha_{n+\alpha})=0\tag{38}$$

After a few mathematical manipulations, it can be shown that the following steps should be taken in order to numerically solve the equation (38) for λ_1.

$$
\begin{aligned}
g(\lambda_1^k) &= \left\|\boldsymbol{\xi}_{n+\alpha}^{trial}\right\| - \left(2G\lambda_1^k + \sqrt{\frac{2}{3}}\Delta H_{n+\alpha}^k\right) - \sqrt{\frac{2}{3}}K(\alpha_{n+\alpha}^k) \\
g'(\lambda_1^k) &= -2G\left[1 + \frac{H'(\alpha_{n+\alpha}^k) + K'(\alpha_{n+\alpha}^k)}{3G}\right] \\
\Delta\lambda_1^{k+1} &= -\frac{g(\lambda_1^k)}{g'(\lambda_1^k)} \\
\lambda_1^{k+1} &= \lambda_1^k + \Delta\lambda_1^{k+1} = \lambda_1^k - \frac{g(\lambda_1^k)}{g'(\lambda_1^k)}
\end{aligned}
\tag{39}
$$

where $\Delta H_{n+\alpha}^k$ and $\alpha_{n+\alpha}^k$ are the updated values of $\Delta H_{n+\alpha}$ and $\alpha_{n+\alpha}$ in step k and are computed using the following relations.

$$
\begin{aligned}
\Delta H_{n+\alpha}^k &= H(\alpha_{n+\alpha}^k) - H(\alpha_n) \\
\alpha_{n+\alpha}^k &= \alpha_{n+\alpha}^{trial} + \sqrt{\frac{2}{3}}\lambda_1^k
\end{aligned}
\tag{40}
$$

In the previous derivations, it should be noted that $\mathbf{n}_{n+\alpha}$ is the normal tensor at time instant $t_{n+\alpha}$ and regarding the equation $(37)_4$ it can be computed using a co-alignment relation similar to equation (22).

The second step of DMPT1 algorithm, after applying a backward Euler integration method over the subinterval $[t_n, t_{n+\alpha}]$, consists of an integration using midpoint rule over the time interval $[t_n, t_{n+1}]$. The algorithmic setup is as follows

$$
\begin{aligned}
\mathbf{e}_{n+1}^p &= \mathbf{e}_n^p + \lambda_2\mathbf{n}_{n+\alpha} \\
\mathbf{s}_{n+1} &= 2G(\mathbf{e}_{n+1} - \mathbf{e}_{n+1}^p) \\
\boldsymbol{\beta}_{n+1} &= \boldsymbol{\beta}_n + \sqrt{\frac{2}{3}}\Delta H_{n+1}\mathbf{n}_{n+\alpha},\ \Delta H_{n+1} = H(\alpha_{n+1}) - H(\alpha_n) \\
\boldsymbol{\xi}_{n+1} &= \mathbf{s}_{n+1} - \boldsymbol{\beta}_{n+1} \\
\alpha_{n+1} &= \alpha_n + \sqrt{\frac{2}{3}}\lambda_2,\ \lambda_2 = \Delta t\,\gamma_{n+\alpha}
\end{aligned}
\tag{41}
$$

where $\mathbf{n}_{n+\alpha}$ is the normal tensor at midpoint and is computed in the previous step. The plastic multiplier λ_2 is calculated by enforcing the consistency condition at the end of time step.

As usual, the solution of the set of equations (41) is pursued by initially assuming that the evolution of variables is elastic. Thus the trial step can be written in a form similar to equations (16). If the trial relative stress satisfies the yield condition, i.e. the equation (17) holds, the trial variables are the correct ones and no further updating is necessary. However if this is not the case, a plastic correction as specified by the following equations is required.

$$\begin{aligned}
\mathbf{e}^{p}_{n+1} &= \mathbf{e}^{p,trial}_{n+1} + \lambda_2 \mathbf{n}_{n+\alpha} \\
\mathbf{s}_{n+1} &= \mathbf{s}^{trial}_{n+1} - 2G\lambda_2 \mathbf{n}_{n+\alpha} \\
\boldsymbol{\beta}_{n+1} &= \boldsymbol{\beta}^{trial}_{n+1} + \sqrt{\frac{2}{3}}\Delta H_{n+1}\mathbf{n}_{n+\alpha} \\
\boldsymbol{\xi}_{n+1} &= \boldsymbol{\xi}^{trial}_{n+1} - \left(2G\lambda_2 + \sqrt{\frac{2}{3}}\Delta H_{n+1}\right)\mathbf{n}_{n+\alpha} \\
\alpha_{n+1} &= \alpha^{trial}_{n+1} + \sqrt{\frac{2}{3}}\lambda_2
\end{aligned} \tag{42}$$

The plastic multiplier λ_2 is computed by requiring that the yield function be satisfied for the relative stress $\boldsymbol{\xi}_{n+1}$, i.e. equation (23) should hold. Following the same procedure used for SMPT1 algorithm, the numerical solution of λ_2 can be summarized as follows.

$$\begin{aligned}
g(\lambda_2^k) &= \boldsymbol{\xi}^k_{n+1} : \boldsymbol{\xi}^k_{n+1} - \frac{2}{3}K^2(\alpha^k_{n+1}) \\
g'(\lambda_2^k) &= -2\left\{\left[2G + \frac{2}{3}H'(\alpha^k_{n+1})\right]\boldsymbol{\xi}^k_{n+1} : \mathbf{n}_{n+\alpha} + \frac{2}{3}\sqrt{\frac{2}{3}}K(\alpha^k_{n+1})K'(\alpha^k_{n+1})\right\} \\
\Delta\lambda_2^{k+1} &= -\frac{g(\lambda_2^k)}{g'(\lambda_2^k)} \\
\lambda_2^{k+1} &= \lambda_2^k + \Delta\lambda_2^{k+1} = \lambda_2^k - \frac{g(\lambda_2^k)}{g'(\lambda_2^k)}
\end{aligned} \tag{43}$$

where $\boldsymbol{\xi}^k_{n+1}$ and α^k_{n+1} are the updated values of $\boldsymbol{\xi}_{n+1}$ and α_{n+1} in step k and are computed using the following relations.

$$\begin{aligned}
\boldsymbol{\xi}^k_{n+1} &= \boldsymbol{\xi}^{trial}_{n+1} - \left(2G\lambda_2^k + \sqrt{\frac{2}{3}}\Delta H^k_{n+1}\right)\mathbf{n}_{n+\alpha},\ \Delta H^k_{n+1} = H(\alpha^k_{n+1}) - H(\alpha_n) \\
\alpha^k_{n+1} &= \alpha^{trial}_{n+1} + \sqrt{\frac{2}{3}}\lambda^k_2
\end{aligned} \tag{44}$$

From the previous discussion it is evident that the second step of DMPT1 algorithm is similar to SMPT1 algorithm, but with the difference that the normal tensor $\mathbf{n}_{n+\alpha}$ in DMPT1 algorithm is computed in the first step.

As the second step, DMPT2 algorithm extrapolates the results obtained in the first step to the time instant t_{n+1}. If it is assumed that the evolution of variables over the time interval $[t_n, t_{n+1}]$ is linear, then the extrapolation can be performed with the help of equations (15) by expressing the quantities at time instant t_{n+1} in terms of the quantities at time instants t_n and

$t_{n+\alpha}$. The variables obtained in this way constitute the trial state for the second step. Therefore the trial state can be written as

$$\begin{aligned}
\mathbf{e}_{n+1}^{p,trial} &= \frac{1}{\alpha}\mathbf{e}_{n+\alpha}^{p} - \frac{1-\alpha}{\alpha}\mathbf{e}_{n}^{p} \\
\mathbf{s}_{n+1}^{trial} &= \frac{1}{\alpha}\mathbf{s}_{n+\alpha} - \frac{1-\alpha}{\alpha}\mathbf{s}_{n} \\
\boldsymbol{\beta}_{n+1}^{trial} &= \frac{1}{\alpha}\boldsymbol{\beta}_{n+\alpha} - \frac{1-\alpha}{\alpha}\boldsymbol{\beta}_{n} \\
\boldsymbol{\xi}_{n+1}^{trial} &= \frac{1}{\alpha}\boldsymbol{\xi}_{n+\alpha} - \frac{1-\alpha}{\alpha}\boldsymbol{\xi}_{n} \\
\alpha_{n+1}^{trial} &= \frac{1}{\alpha}\alpha_{n+\alpha} - \frac{1-\alpha}{\alpha}\alpha_{n}
\end{aligned} \tag{45}$$

If the trial relative stress $\boldsymbol{\xi}_{n+1}^{trial}$ satisfies the yield condition, i.e.

$$\left\|\boldsymbol{\xi}_{n+1}^{trial}\right\| - \sqrt{\frac{2}{3}}K\left(\alpha_{n+1}^{trial}\right) \le 0 \tag{46}$$

the trial variables are the correct ones and no further updating is required. However, if the trial relative stress $\boldsymbol{\xi}_{n+1}^{trial}$ violates the yield condition, a plastic correction according to the following equations is required.

$$\begin{aligned}
\mathbf{e}_{n+1}^{p} &= \mathbf{e}_{n+1}^{p,trial} + \lambda_2 \mathbf{n}_{n+1} \\
\mathbf{s}_{n+1} &= \mathbf{s}_{n+1}^{trial} - 2G\lambda_2 \mathbf{n}_{n+1} \\
\boldsymbol{\beta}_{n+1} &= \boldsymbol{\beta}_{n+1}^{trial} + \sqrt{\frac{2}{3}}\Delta\overline{H}_{n+1}\mathbf{n}_{n+1},\ \Delta\overline{H}_{n+1} = H(\alpha_{n+1}) - H(\alpha_{n+1}^{trial}) \\
\boldsymbol{\xi}_{n+1} &= \boldsymbol{\xi}_{n+1}^{trial} - \left(2G\lambda_2 + \sqrt{\frac{2}{3}}\Delta\overline{H}_{n+1}\right)\mathbf{n}_{n+1} \\
\alpha_{n+1} &= \alpha_{n+1}^{trial} + \sqrt{\frac{2}{3}}\lambda_2
\end{aligned} \tag{47}$$

where $\mathbf{n}_{n+1}$ is the normal tensor at time instant t_{n+1} and can be computed using the following relation.

$$\mathbf{n}_{n+1} = \frac{\boldsymbol{\xi}_{n+1}}{\left\|\boldsymbol{\xi}_{n+1}\right\|} = \frac{\boldsymbol{\xi}_{n+1}^{trial}}{\left\|\boldsymbol{\xi}_{n+1}^{trial}\right\|} \tag{48}$$

Straightforward algebraic manipulations lead to the following set of equations for numerical solution of λ_2.

$$\begin{aligned} g(\lambda_2^k) &= \left\|\xi_{n+1}^{trial}\right\| - \left(2G\lambda_2^k + \sqrt{\frac{2}{3}}\Delta\overline{H}_{n+1}^k\right) - \sqrt{\frac{2}{3}}K(\alpha_{n+1}^k) \\ g'(\lambda_2^k) &= -2G\left[1 + \frac{H'(\alpha_{n+1}^k) + K'(\alpha_{n+1}^k)}{3G}\right] \\ \Delta\lambda_2^{k+1} &= -\frac{g(\lambda_2^k)}{g'(\lambda_2^k)} \\ \lambda_2^{k+1} &= \lambda_2^k + \Delta\lambda_2^{k+1} = \lambda_2^k - \frac{g(\lambda_2^k)}{g'(\lambda_2^k)} \end{aligned} \tag{49}$$

where $\Delta\overline{H}_{n+1}^k$ and α_{n+1}^k are the updated values of $\Delta\overline{H}_{n+1}$ and α_{n+1} in step k and are computed using the following relations.

$$\begin{aligned} \Delta\overline{H}_{n+1}^k &= H(\alpha_{n+1}^k) - H(\alpha_{n+1}^{trial}) \\ \alpha_{n+1}^k &= \alpha_{n+1}^{trial} + \sqrt{\frac{2}{3}}\lambda_2^k \end{aligned} \tag{50}$$

Pictorial illustrations of DMPT1 and DMPT2 algorithms as well as a detailed discussion about these algorithms for linear isotropic/kinematic hardening can be found in [2].

4. NUMERICAL EXAMPLES

In this section, numerical examples are used to compare the SMPT1, SMPT2, DMPT1 and DMPT2 algorithms under various loading conditions. In order to solve examples, a finite element program has been developed in C++ and the four integration algorithms as well as the backward Euler method have been implemented into it. The backward Euler method is adopted as the reference in all numerical examples.

The isotropic and kinematic hardening rules are of the exponential type as specified by equations (12). The elastic properties of the material are taken as E = 7000MPa, $\nu = 0.3$, and the parameters in the saturation type of hardening rule are $\overline{K}_\infty = 35\text{MPa}$, $\overline{K}_0 = 30\text{MPa}$, $\delta = 0.1$, $\overline{H}' = 280\text{MPa}$ and $\beta = 0.8$.

The numerical examples fall into three groups. In the first one, a typical mixed stress-strain loading history is considered [2]. Error graphs are used to investigate the convergence of algorithms to the exact solution. In the second example, a set of non-proportional loading histories are employed to build iso-error maps. The iso-error maps are then used to study the accuracy of algorithms. The third example is a perforated strip subject to uniaxial tension. The

aim of this example is to assess the reliability of algorithms in real engineering applications.

4.1 Mixed stress-strain loading history

The biaxial non-proportional stress-strain loading history shown in figure 1 is considered in this section. The loading history is obtained by controlling ε_{11} and ε_{22} [2]. In order to study the convergence history, error graphs are used to evaluate the error produced by different algorithms. To provide a relationship between the error and the size of time step, the following error measure is introduced.

$$E_n^{\sigma} = \frac{\left\| \sigma - \sigma_n^{ex} \right\|}{\left\| \sigma_n^{ex} \right\|} \tag{51}$$

where $\|\bullet\|$ is the Euclidean norm and σ_n^{ex} is the stress at time instant t_n computed using the backward Euler method with a very fine time discretization. Figures 2-3, present the error graphs for SMPT1, SMPT2, DMPT1, and DMPT2 algorithms employing $\Delta t = 0.1$s.

Regarding the error graphs, it is clear that the performance of DMPT2 algorithm is considerably better than the other algorithms while the performance of SMPT2 algorithm is the worst. Other aspects such as the linear and quadratic accuracy of algorithms can be investigated as well [2].

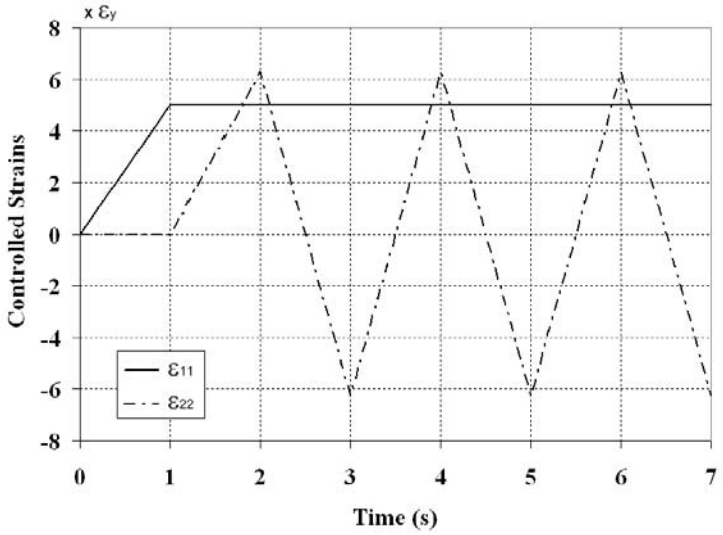


Figure 1. Mixed stress-strain loading history

4.2 Iso-error maps

Iso-error maps are extensively employed by researchers as a systematic procedure to assess the accuracy of a proposed algorithm. In the literature, iso-error maps are generated by applying a set of normalized strains in principal directions. A detailed description of this procedure can be found in [7] or in [12]. In order to build the iso-error maps, the following formula is used to measure the error in stress compared with the exact value.

$$E_{iso}^{\sigma} = \frac{\sqrt{(\sigma - \sigma^{ex}) : (\sigma - \sigma^{ex})}}{\sqrt{\sigma^{ex} : \sigma^{ex}}} \times 100 \tag{52}$$

where σ^{ex} is the stress computed using backward Euler method with a very fine discretization. The iso-error maps shown in figures 5-7 correspond to points **A**, **B** and **C** of the yield surface in figure 4 and are generated for $\alpha = 0.5$.

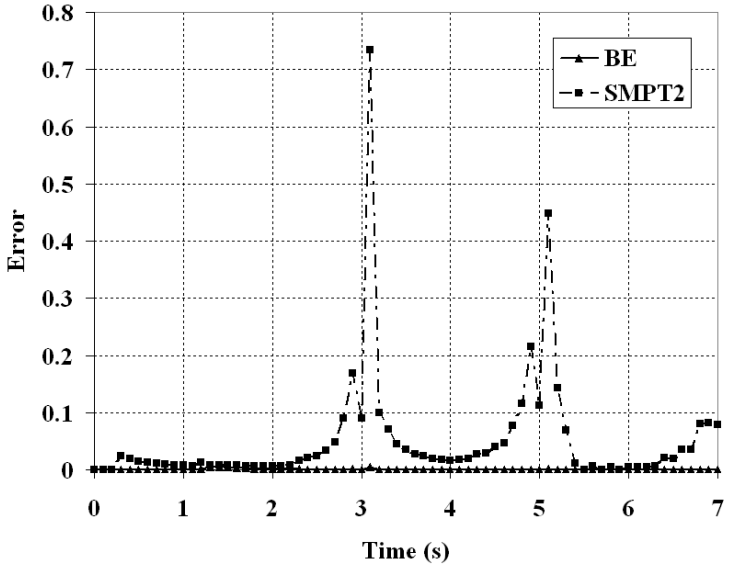


Figure 2. Error for SMPT2 algorithm

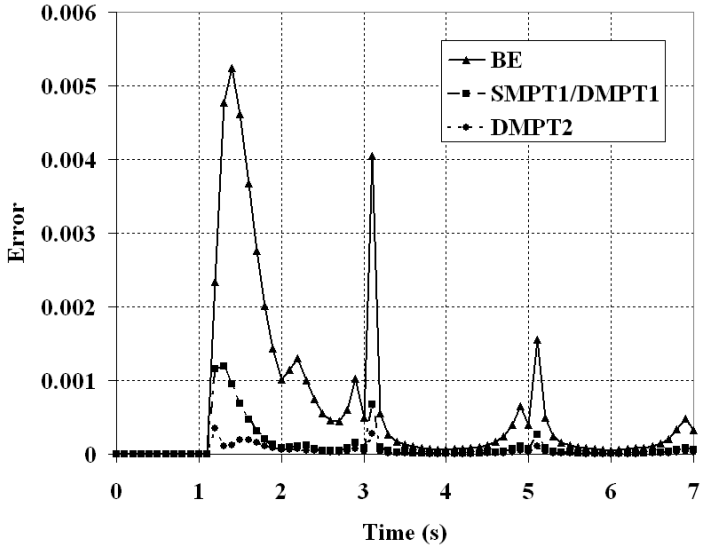


Figure 3. Error for SMPT1/DMPT1 and DMPT2 algorithms

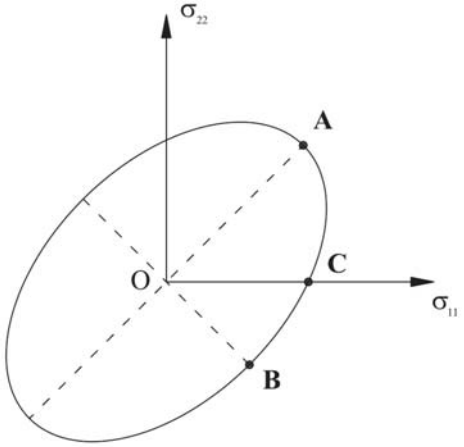


Figure 4. Plane stress yield surface. Points for iso-error maps

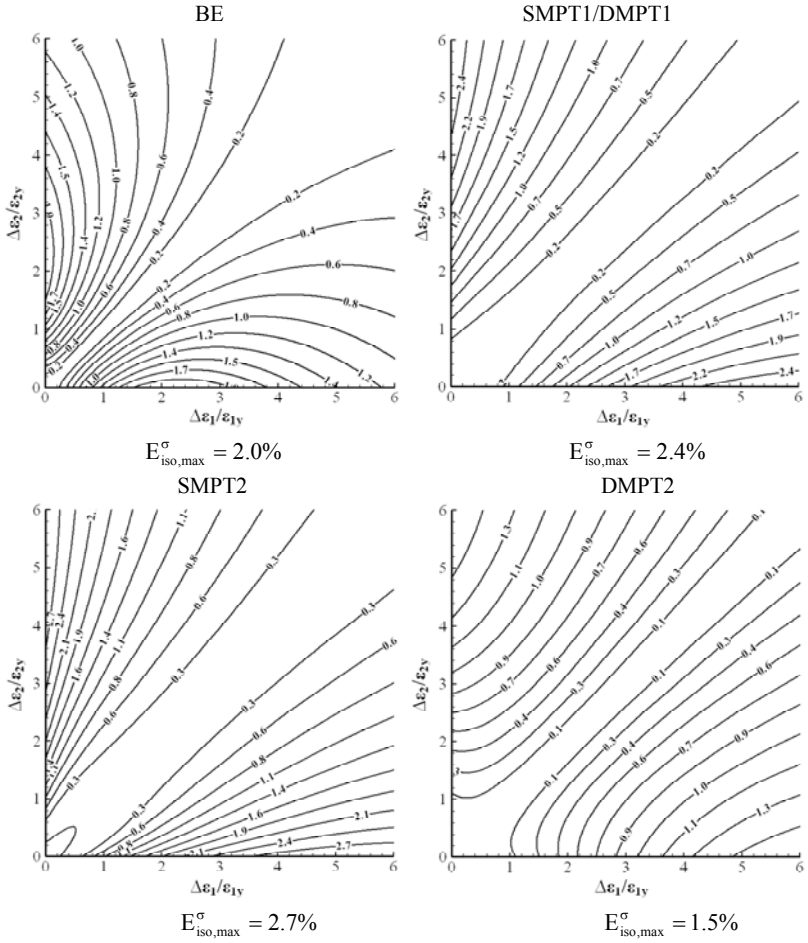


Figure 5. Iso-error maps for point **A** of the yield surface

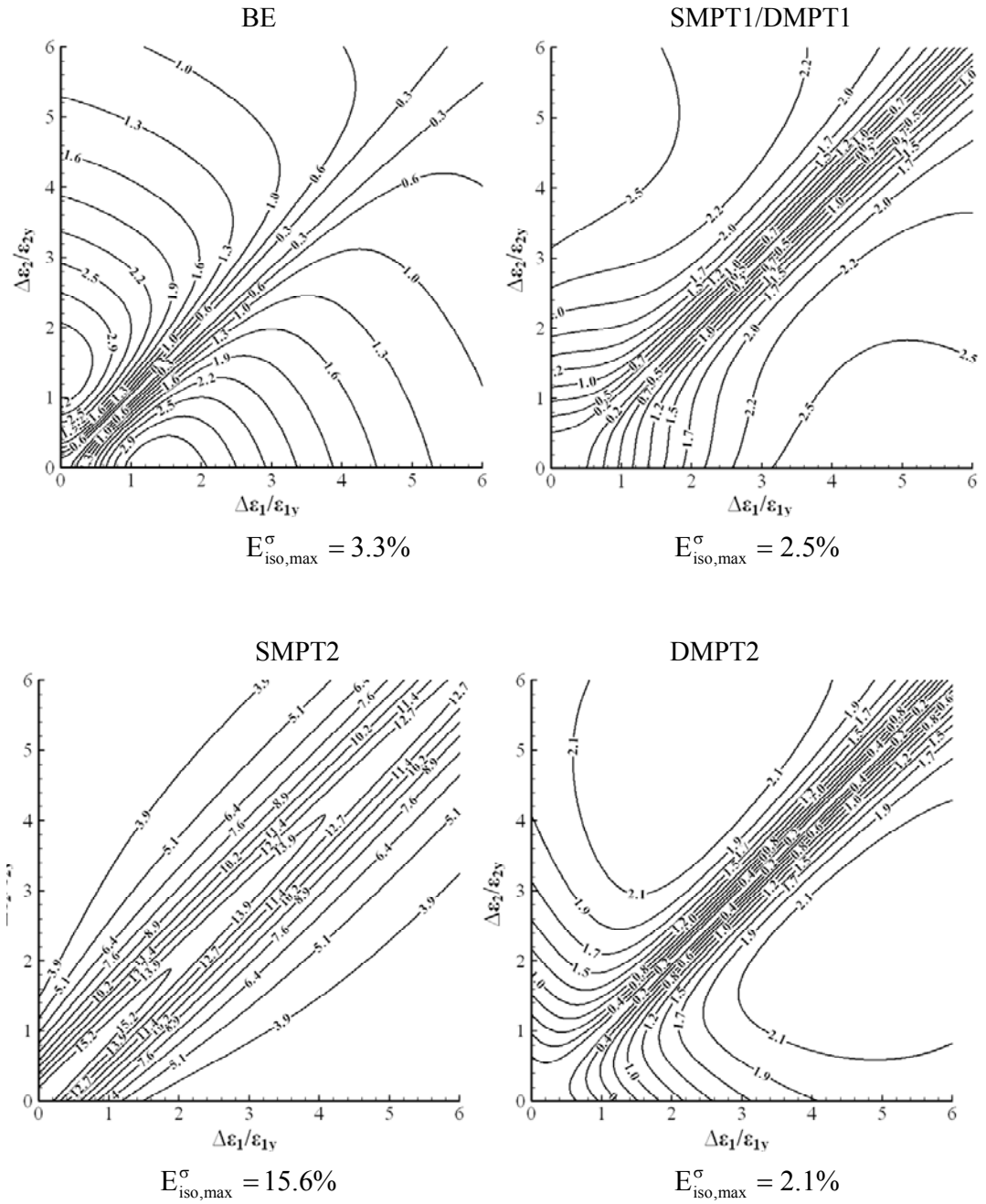


Figure 6. Iso-error maps for point **B** of the yield surface

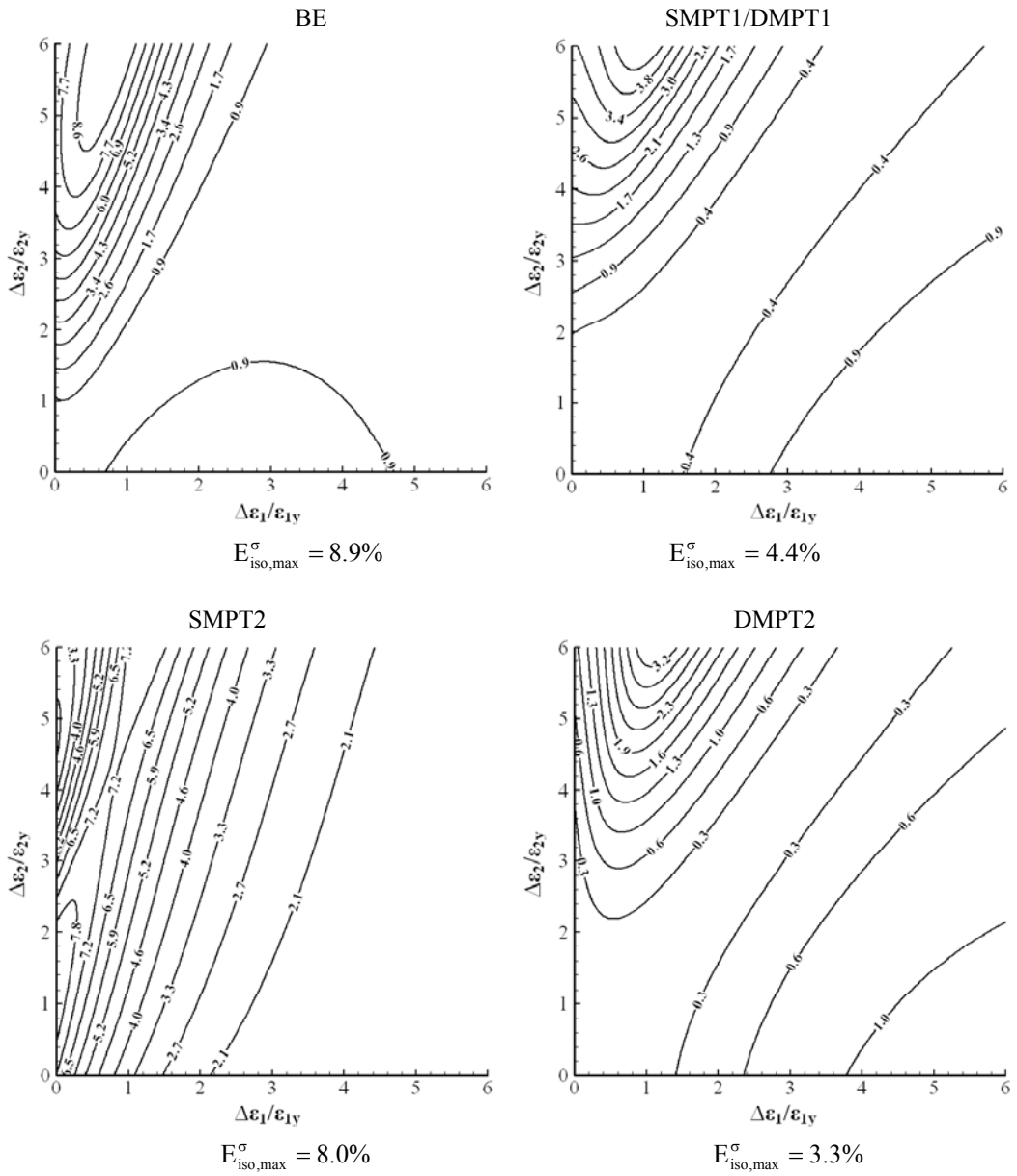
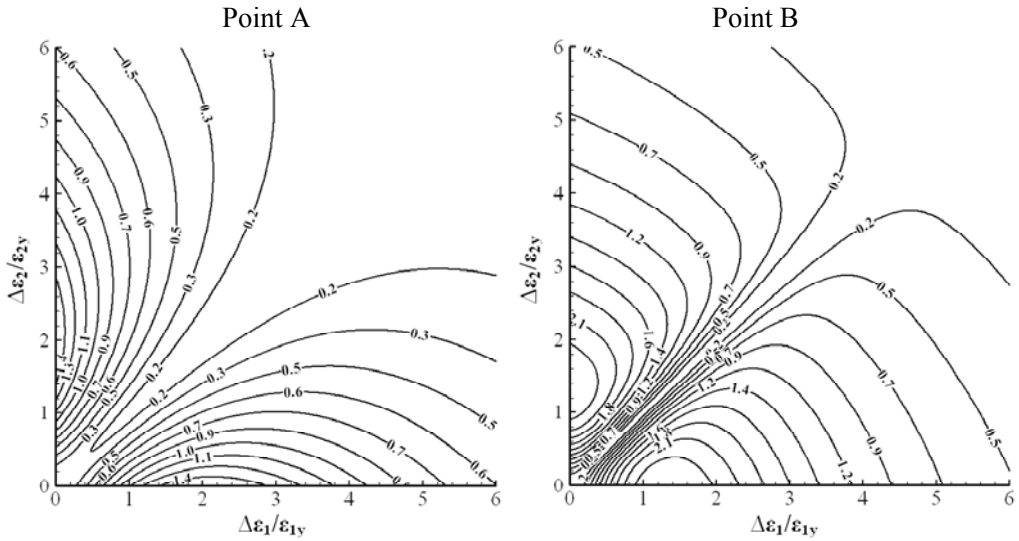


Figure 7. Iso-error maps for point **C** of the yield surface



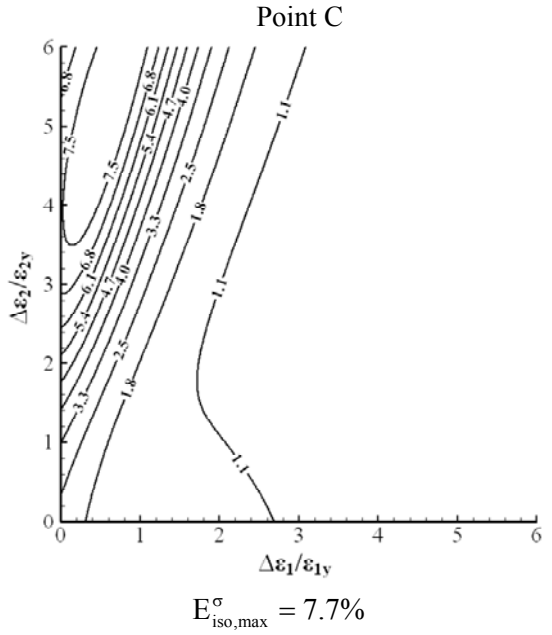


Figure 8. Iso-error maps for SMPT2 algorithm for $\alpha = 0.86$

The error range for each iso-error map is divided into 10 error levels. The maximum error for each iso-error map is also reported. Regarding figures 5-7 the following observations can be made:

- As a first order method, the behavior of backward Euler is quite satisfactory showing low error levels for large time steps [2]. On the other hand, DMPT2 method has the

lowest error levels compared with other integration methods.

- SMPT2 method has the worst performance among the five integration methods analyzed here. The poor performance of SMPT2 is attributable to two factors; the first is the choice of $\alpha = 0.5$ for which the accuracy of second order methods in large time steps deteriorates [5, 6] and the second is that the consistency condition for SMPT2 method is imposed at midpoint while the values of stresses are reported at the end. The lowest value of α for which the results of SMPT2 algorithm are reliable is 0.86. The iso-error maps for this case are shown in figure 8.
- From the iso-error maps shown in figure 8, it can be inferred that the accuracy of SMPT1, SMPT2 and DMPT1 improves for values of α close to 1. This result agrees with the conclusion made previously by many authors [6].

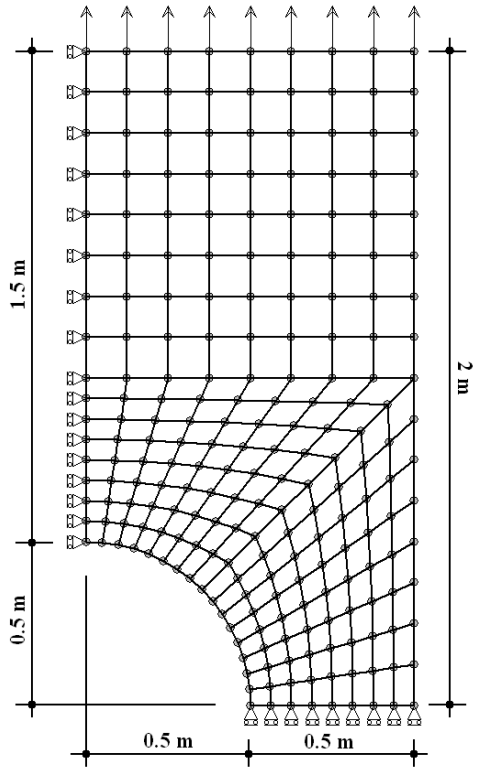


Figure 9. Perforated strip under uniaxial tension

4.3 Perforated strip subject to uniaxial tension

In this example, the extension of a perforated strip in plane strain condition is considered. Due to the symmetry of geometry and loading only a quarter of the plate is analyzed. The geometry of the plate and the finite element mesh are shown in figure 9. A total of 192 quadrilateral plane strain elements with 4 integration points and linear displacement interpolation along each direction are used to model the plate. The load that applies to the

plate increases proportionally up to six times the initial yielding.

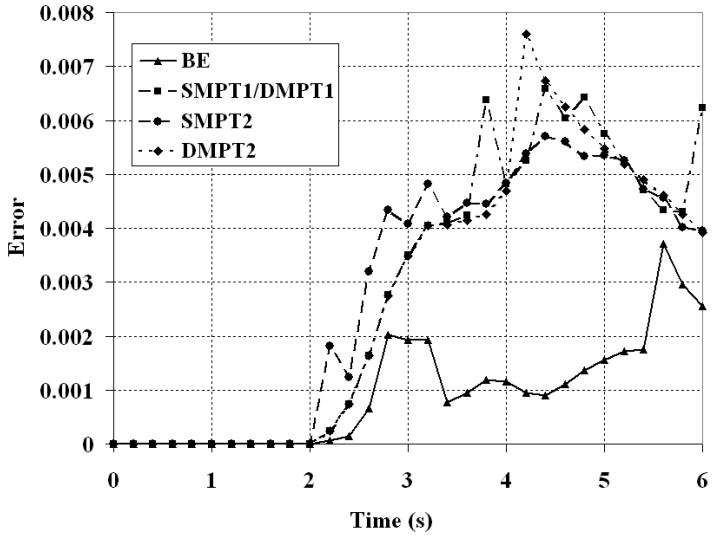


Figure 10. Error graphs for different algorithms with $\alpha = 0.5$

The error graphs for BE, SMPT1, SMPT2, DMPT1 and DMPT2 algorithms with $\Delta t = 0.2s$ are shown in figures 10 and 11 for $\alpha = 0.5$ and $\alpha = 0.80$ respectively. Regarding the error graphs the following observations can be made:

- For single step, second order methods such as SMPT1 and SMPT2 choosing a larger value for α (preferably between 0.7 and 0.8) leads to more accurate results [5, 6].
- Increasing the value of α also improves the behavior of double step methods such as DMPT1 and DMPT2. In the case of $\alpha = 0.80$, the behavior of DMPT1 and DMPT2 is considerably better than BE.

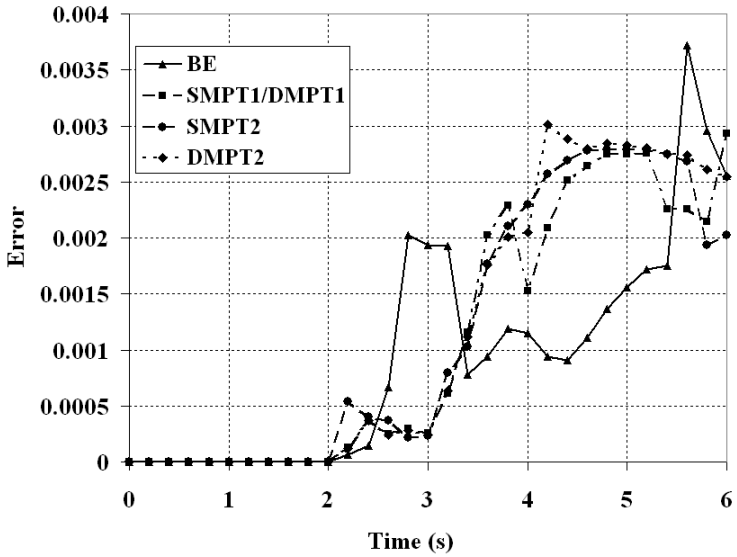


Figure 11. Error graphs for different algorithms with $\alpha = 0.80$

- At $\alpha = 0.80$, the second order algorithms are competitive in terms of the error that is produced. From figure 11, it can be seen that the error level at different stages of loading is approximately the same for all second order algorithms. Considering the peaks and valleys in the graph, it can also be inferred that second order algorithms are more stable compared with backward Euler method.
- As a performance test with $\alpha = 0.80$, the model was fully loaded in one step only. Time elapses and the number of iterations required to complete the analysis on a machine with dual core 3.0GHz CPU are listed in table 1. Regarding the values it is observed that the time elapses and the number of iterations for second order algorithms except SMPT2 are close to backward Euler.

Table 1: Time elapses and number of iterations for different algorithms

Algorithm	**No. iterations**	**Time (s)**
BE	23	12
SMPT1/DMPT1	22	11
SMPT2	40	21
DMPT2	24	12

5. CONCLUDING REMARKS

In this work, existing second order integration methods were extended to general isotropic/kinematic hardening for classical J_2 plasticity. Several numerical examples including mixed stress-strain loading histories, iso-error maps and boundary value problems were presented to assess the numerical aspects of these algorithms.

Based on numerical tests it was demonstrated that the behavior of DMPT2 algorithm in a displacement controlled mechanism is notably better than the others. On the other hand, regarding the quadratic convergence of a second order algorithm, the behavior of SMPT2 algorithm is very poor. In a force controlled mechanism, it is more appropriate to choose α between 0.7 and 0.8 in order to have an improved performance compared with backward Euler method. Numerical examples show that for $\alpha = 0.8$ the CPU time that is consumed by DMPT1 and DMPT2 algorithms is approximately the same as backward Euler method but with higher accuracy.

It should however be noted that the accuracy of second order algorithms is sensitive to many parameters such as the value of α, the size of time step, the tolerances that are imposed and even the efficiency of method that is used to solve the set of equations in each iteration. Consequently the combination of parameters that works fine for one algorithm might not be suitable for another. The choice between second order algorithms is therefore influenced by the set of parameters being selected, the type of problem under consideration (displacement/force controlled) and of course the burden that is anticipated for coding the

specific algorithm.

APPENDIX A

In this appendix, the derivation of tangent operators for SMPT1, SMPT2, DMPT1 and DMPT2 algorithms is presented.

A.1 SMPT1 algorithm

Taking into account the deviatoric and volumetric part of the stress, using equations (1) - (3) one can write the tangent operator in the following form:

$$\mathbf{C}^{ep} = \frac{\partial \boldsymbol{\sigma}_{n+1}}{\partial \boldsymbol{\varepsilon}_{n+1}} = \frac{\partial \mathbf{s}_{n+1}}{\partial \boldsymbol{\varepsilon}_{n+1}} + \kappa \mathbf{I} \otimes \mathbf{I} \tag{A1}$$

Taking the derivative of equation $(14)_2$ with respect to $\boldsymbol{\varepsilon}_{n+1}$ and using the equation $(14)_1$ we have

$$\frac{\partial \mathbf{s}_{n+1}}{\partial \boldsymbol{\varepsilon}_{n+1}} = 2G\left(\boldsymbol{\iota} - \frac{1}{3}\mathbf{I} \otimes \mathbf{I} \right) - 2G\alpha \left(\mathbf{n}_{n+\alpha} \otimes \frac{\partial \lambda}{\partial \boldsymbol{\varepsilon}_{n+\alpha}} + \lambda \frac{\partial \mathbf{n}_{n+\alpha}}{\partial \boldsymbol{\varepsilon}_{n+\alpha}} \right) \tag{A2}$$

where $\mathbf{I}$ and $\boldsymbol{\iota}$ are respectively the second and fourth order identity tensors. Using equation (24) as the starting point and after straightforward algebraic manipulations one finds

$$\frac{\partial \lambda}{\partial \boldsymbol{\varepsilon}_{n+\alpha}} = \frac{B}{\alpha}[(1-C)\boldsymbol{\xi}_{n+1} + C(\boldsymbol{\xi}_{n+1} : \mathbf{n}_{n+\alpha})\mathbf{n}_{n+\alpha}] \tag{A3}$$

with

$$B = 2G\left\{ \left[2G + \frac{2}{3}H'(\alpha_{n+1}) \right](\boldsymbol{\xi}_{n+1} : \mathbf{n}_{n+\alpha}) + \frac{2}{3}\sqrt{\frac{2}{3}}K(\alpha_{n+1})K'(\alpha_{n+1}) \right\}^{-1} \tag{A4}$$

and

$$C = \frac{\alpha\left(2G\lambda + \sqrt{\frac{2}{3}}\Delta H_{n+1} \right)}{\left\| \boldsymbol{\xi}^{trial}_{n+\alpha} \right\|} \tag{A5}$$

Using the second equality of equation (22) it is found that

$$\frac{\partial \mathbf{n}_{n+\alpha}}{\partial \boldsymbol{\varepsilon}_{n+\alpha}} = \frac{2G}{\left\| \boldsymbol{\xi}^{trial}_{n+\alpha} \right\|}\left(\boldsymbol{\iota} - \frac{1}{3}\mathbf{I} \otimes \mathbf{I} - \mathbf{n}_{n+\alpha} \otimes \mathbf{n}_{n+\alpha} \right) \tag{A6}$$

Substituting equations (A6) and (A3) into equation (A2) and the result into equation (A1) finally leads to the following equation for tangent operator.

$$\mathbf{C}^{ep} = \kappa\mathbf{I}\otimes\mathbf{I} + 2G(1-A)\left(\boldsymbol{\iota} - \frac{1}{3}\mathbf{I}\otimes\mathbf{I}\right) + 2G\left[A - BC(\boldsymbol{\xi}_{n+1} : \mathbf{n}_{n+\alpha})\right]\mathbf{n}_{n+\alpha}\otimes\mathbf{n}_{n+\alpha} - 2GB(1-C)\,\mathbf{n}_{n+\alpha}\otimes\boldsymbol{\xi}_{n+1} \tag{A7}$$

with

$$A = \frac{2G\alpha\lambda}{\left\|\boldsymbol{\xi}^{trial}_{n+\alpha}\right\|} \tag{A8}$$

It is clear that the presence of the last term in equation (A7) renders the tangent operator unsymmetric.

A.2 SMPT2 algorithm

For SMPT2 algorithm, the tangent operator is

$$\mathbf{C}^{ep} = \kappa\mathbf{I}\otimes\mathbf{I} + 2G(1-A)\left(\boldsymbol{\iota} - \frac{1}{3}\mathbf{I}\otimes\mathbf{I}\right) + 2G(A-B)\,\mathbf{n}_{n+\alpha}\otimes\mathbf{n}_{n+\alpha} \tag{A9}$$

with

$$A = \frac{2G\alpha\lambda}{\left\|\boldsymbol{\xi}^{trial}_{n+\alpha}\right\|} \tag{A10}$$

and

$$B = \frac{1}{1 + \dfrac{H'(\alpha_{n+\alpha}) + K'(\alpha_{n+\alpha})}{3G}} \tag{A11}$$

A.3 DMPT1 algorithm

In the case of DMPT1 algorithm, the equation (A1) still holds. Taking the derivative of equation $(41)_2$ with respect to $\boldsymbol{\varepsilon}_{n+1}$ and using the equation $(41)_1$ we have

$$\frac{\partial\mathbf{s}_{n+1}}{\partial\boldsymbol{\varepsilon}_{n+1}} = 2G\left(\boldsymbol{\iota} - \frac{1}{3}\mathbf{I}\otimes\mathbf{I}\right) - 2G\left(\mathbf{n}_{n+\alpha}\otimes\frac{\partial\lambda_2}{\partial\boldsymbol{\varepsilon}_{n+1}} + \lambda_2\frac{\partial\mathbf{n}_{n+\alpha}}{\partial\boldsymbol{\varepsilon}_{n+1}}\right) \tag{A12}$$

Using the first equality in equation (22) it is found that

$$\frac{\partial\mathbf{n}_{n+\alpha}}{\partial\boldsymbol{\varepsilon}_{n+1}} = \frac{\alpha}{\left\|\boldsymbol{\xi}_{n+\alpha}\right\|}\left(\boldsymbol{\iota} - \mathbf{n}_{n+\alpha}\otimes\mathbf{n}_{n+\alpha}\right) : \frac{\partial\boldsymbol{\xi}_{n+\alpha}}{\partial\boldsymbol{\varepsilon}_{n+\alpha}} \tag{A13}$$

The derivative of $\boldsymbol{\xi}_{n+\alpha}$ with respect to $\boldsymbol{\varepsilon}_{n+\alpha}$ can be obtained with the help of equation $(37)_4$ as follows.

$$\frac{\partial \boldsymbol{\xi}_{n+\alpha}}{\partial \boldsymbol{\varepsilon}_{n+\alpha}} = 2G\left[(1-\overline{A})\left(\boldsymbol{\imath} - \frac{1}{3}\mathbf{I}\otimes\mathbf{I}\right) + (\overline{A}-\overline{B})\,\mathbf{n}_{n+\alpha}\otimes\mathbf{n}_{n+\alpha}\right] \tag{A14}$$

with

$$\overline{A} = \frac{2G\lambda_1 + \sqrt{\frac{2}{3}}\Delta H_{n+\alpha}}{\left\|\boldsymbol{\xi}^{\text{trial}}_{n+\alpha}\right\|} \tag{A15}$$

and

$$\overline{B} = \frac{1 + \frac{H'(\alpha_{n+\alpha})}{3G}}{1 + \frac{H'(\alpha_{n+\alpha}) + K'(\alpha_{n+\alpha})}{3G}} \tag{A16}$$

Substituting equation (A14) into equation (A13) leads to the following equation.

$$\frac{\partial \mathbf{n}_{n+\alpha}}{\partial \boldsymbol{\varepsilon}_{n+1}} = \frac{2G\alpha}{\left\|\boldsymbol{\xi}_{n+\alpha}\right\|}(1-\overline{A})\left(\boldsymbol{\imath} - \frac{1}{3}\mathbf{I}\otimes\mathbf{I} - \mathbf{n}_{n+\alpha}\otimes\mathbf{n}_{n+\alpha}\right) \tag{A17}$$

It should be noted that if $\lambda_1 = 0$, i.e. the yield condition at midpoint is not violated, then the following equation replaces the equation (A17).

$$\frac{\partial \mathbf{n}_{n+\alpha}}{\partial \boldsymbol{\varepsilon}_{n+1}} = \frac{2G\alpha}{\left\|\boldsymbol{\xi}^{\text{trial}}_{n+\alpha}\right\|}\left(\boldsymbol{\imath} - \frac{1}{3}\mathbf{I}\otimes\mathbf{I} - \mathbf{n}_{n+\alpha}\otimes\mathbf{n}_{n+\alpha}\right) \tag{A18}$$

The derivative of λ_2 with respect to $\boldsymbol{\varepsilon}_{n+1}$ is

$$\frac{\partial \lambda_2}{\partial \boldsymbol{\varepsilon}_{n+1}} = B\left[\boldsymbol{\xi}_{n+1} - \frac{2G\lambda_2 + \sqrt{\frac{2}{3}}\Delta H_{n+1}}{2G}\,\boldsymbol{\xi}_{n+1} : \frac{\partial \mathbf{n}_{n+\alpha}}{\partial \boldsymbol{\varepsilon}_{n+1}}\right] \tag{A19}$$

with B defined in equation (A4). Substituting equation (A17) or (A18), whichever applicable, into equation (A19) leads to the following equation.

$$\frac{\partial \lambda_2}{\partial \boldsymbol{\varepsilon}_{n+1}} = B[(1-C)\boldsymbol{\xi}_{n+1} + C(\boldsymbol{\xi}_{n+1} : \mathbf{n}_{n+\alpha})\mathbf{n}_{n+\alpha}] \tag{A20}$$

with

$$C = \frac{2G\lambda_2 + \sqrt{\frac{2}{3}}\Delta H_{n+1}}{2G}\,\widetilde{A} \tag{A21}$$

and

$$\widetilde{\mathrm{A}} = \begin{cases} \dfrac{2\mathrm{G}\alpha}{\left\|\boldsymbol{\xi}_{\mathrm{n}+\alpha}\right\|}(1-\overline{\mathrm{A}}); & \lambda_1 \neq 0 \\ \dfrac{2\mathrm{G}\alpha}{\left\|\boldsymbol{\xi}^{\mathrm{trial}}_{\mathrm{n}+\alpha}\right\|}; & \lambda_1 = 0 \end{cases} \tag{A22}$$

Observing equations (A1), (A12), (A17), (A18) and (A20), the tangent operator for DMPT1 algorithm can be written in the following form.

$$\begin{aligned} \mathbf{C}^{\mathrm{ep}} &= \kappa\mathbf{I}\otimes\mathbf{I} + 2\mathrm{G}(1-\mathrm{A})\left(\boldsymbol{\iota} - \frac{1}{3}\mathbf{I}\otimes\mathbf{I}\right) + \\ &2\mathrm{G}\left[\mathrm{A} - \mathrm{BC}(\boldsymbol{\xi}_{\mathrm{n}+1} : \mathbf{n}_{\mathrm{n}+\alpha})\right]\mathbf{n}_{\mathrm{n}+\alpha}\otimes\mathbf{n}_{\mathrm{n}+\alpha} - 2\mathrm{GB}(1-\mathrm{C})\,\mathbf{n}_{\mathrm{n}+\alpha}\otimes\boldsymbol{\xi}_{\mathrm{n}+1} \end{aligned} \tag{A23}$$

with

$$\mathrm{A} = \lambda_2\widetilde{\mathrm{A}} \tag{A24}$$

The similarity between equations (A7) and (A23) is noteworthy. It is easy to check that if $\lambda_1 = 0$, the equation (A23) reduces to the equation (A7).

A.4 DMPT2 algorithm

In this case, the equation (A1) still holds. Using equations $(47)_2$ and $(45)_2$ the derivative of $\mathbf{s}_{\mathrm{n}+1}$ with respect to $\boldsymbol{\varepsilon}_{\mathrm{n}+1}$ can be computed as

$$\frac{\partial\mathbf{s}_{\mathrm{n}+1}}{\partial\boldsymbol{\varepsilon}_{\mathrm{n}+1}} = \frac{\partial\mathbf{s}_{\mathrm{n}+\alpha}}{\partial\boldsymbol{\varepsilon}_{\mathrm{n}+\alpha}} - 2\mathrm{G}\left(\mathbf{n}_{\mathrm{n}+1}\otimes\frac{\partial\lambda_2}{\partial\boldsymbol{\varepsilon}_{\mathrm{n}+1}} + \lambda_2\frac{\partial\mathbf{n}_{\mathrm{n}+1}}{\partial\boldsymbol{\varepsilon}_{\mathrm{n}+1}}\right) \tag{A25}$$

where the derivative of $\mathbf{s}_{\mathrm{n}+\alpha}$ with respect to $\boldsymbol{\varepsilon}_{\mathrm{n}+\alpha}$ is

$$\frac{\partial\mathbf{s}_{\mathrm{n}+\alpha}}{\partial\boldsymbol{\varepsilon}_{\mathrm{n}+\alpha}} = 2\mathrm{G}(1-\mathrm{A})\left(\boldsymbol{\iota} - \frac{1}{3}\mathbf{I}\otimes\mathbf{I}\right) + 2\mathrm{G}(\mathrm{A}-\mathrm{B})\,\mathbf{n}_{\mathrm{n}+\alpha}\otimes\mathbf{n}_{\mathrm{n}+\alpha} \tag{A26}$$

with

$$\mathrm{A} = \frac{2\mathrm{G}\lambda_1}{\left\|\boldsymbol{\xi}^{\mathrm{trial}}_{\mathrm{n}+\alpha}\right\|} \tag{A27}$$

and

$$\mathrm{B} = \frac{1}{1 + \dfrac{\mathrm{H}'(\alpha_{\mathrm{n}+\alpha}) + \mathrm{K}'(\alpha_{\mathrm{n}+\alpha})}{3\mathrm{G}}} \tag{A28}$$

Using equations (23) and $(47)_4$, the derivative of λ_2 with respect to $\boldsymbol{\varepsilon}_{\mathrm{n}+1}$ is obtained as

$$\frac{\partial \lambda_2}{\partial \boldsymbol{\varepsilon}_{n+1}} = D\left(\frac{1}{2G}\mathbf{n}_{n+1} : \frac{\partial \boldsymbol{\xi}_{n+1}^{trial}}{\boldsymbol{\varepsilon}_{n+1}} - \overline{C}\mathbf{n}_{n+\alpha} \right) \tag{A29}$$

with

$$\overline{C} = \frac{H'(\alpha_{n+1}) - H'(\alpha_{n+1}^{trial}) + K'(\alpha_{n+1})}{3G + H'(\alpha_{n+\alpha}) + K'(\alpha_{n+\alpha})} \tag{A30}$$

and

$$D = \frac{1}{1 + \dfrac{H'(\alpha_{n+1}) + K'(\alpha_{n+1})}{3G}} \tag{A31}$$

Observing the equation $(45)_4$, it is evident that

$$\frac{\partial \boldsymbol{\xi}_{n+1}^{trial}}{\partial \boldsymbol{\varepsilon}_{n+1}} = 2G\left[(1-\overline{A})\left(\mathbf{\imath} - \frac{1}{3}\mathbf{I}\otimes\mathbf{I} \right) + (\overline{A}-\overline{B})\mathbf{n}_{n+\alpha}\otimes\mathbf{n}_{n+\alpha} \right] \tag{A32}$$

with $\overline{A}$ and $\overline{B}$ given by equations (A15) and (A16). Substituting the equation (A32) into (A29) leads to

$$\frac{\partial \lambda_2}{\partial \boldsymbol{\varepsilon}_{n+1}} = D\left\{ (1-\overline{A})\mathbf{n}_{n+1} + \left[(\overline{A}-\overline{B})(\mathbf{n}_{n+1} : \mathbf{n}_{n+\alpha}) - \overline{C} \right]\mathbf{n}_{n+\alpha} \right\} \tag{A33}$$

The derivative of $\mathbf{n}_{n+1}$ with respect to $\boldsymbol{\varepsilon}_{n+1}$ is computed as

$$\frac{\partial \mathbf{n}_{n+1}}{\partial \boldsymbol{\varepsilon}_{n+1}} = \frac{1}{\left\| \boldsymbol{\xi}_{n+1}^{trial} \right\|}\left(\mathbf{\imath} - \mathbf{n}_{n+1}\otimes\mathbf{n}_{n+1} \right) : \frac{\partial \boldsymbol{\xi}_{n+1}^{trial}}{\partial \boldsymbol{\varepsilon}_{n+1}} \tag{A34}$$

Substituting the equation (A32) into (A34) leads to

$$\begin{aligned} \frac{\partial \mathbf{n}_{n+1}}{\partial \boldsymbol{\varepsilon}_{n+1}} = {} & \frac{2G}{\left\| \boldsymbol{\xi}_{n+1}^{trial} \right\|}(1-\overline{A})\left(\mathbf{\imath} - \frac{1}{3}\mathbf{I}\otimes\mathbf{I} - \mathbf{n}_{n+1}\otimes\mathbf{n}_{n+1} \right) + \\ & \frac{2G}{\left\| \boldsymbol{\xi}_{n+1}^{trial} \right\|}(\overline{A}-\overline{B})[\mathbf{n}_{n+\alpha}\otimes\mathbf{n}_{n+\alpha} - (\mathbf{n}_{n+1} : \mathbf{n}_{n+\alpha})\mathbf{n}_{n+1}\otimes\mathbf{n}_{n+\alpha}] \end{aligned} \tag{A35}$$

Substituting equations (A26), (A33) and (A35) into equation (A25) and the result into equation (A1) finally leads to the following equation for tangent operator.

$$\begin{aligned}\mathbf{C}^{ep} = \kappa\mathbf{I}\otimes\mathbf{I} + 2G[1-A-(1-\overline{A})C]\left(\mathbf{\imath} - \frac{1}{3}\mathbf{I}\otimes\mathbf{I}\right) - \\ 2G[(\overline{A}-\overline{B})C-(A-B)]\mathbf{n}_{n+\alpha}\otimes\mathbf{n}_{n+\alpha} - \\ 2G(1-\overline{A})(D-C)\mathbf{n}_{n+1}\otimes\mathbf{n}_{n+1} - \\ 2G[(\overline{A}-\overline{B})(D-C)(\mathbf{n}_{n+1}:\mathbf{n}_{n+\alpha})-\overline{C}]\mathbf{n}_{n+1}\otimes\mathbf{n}_{n+\alpha}\end{aligned} \tag{A36}$$

with

$$C = \frac{2G\lambda_2}{\left\|\boldsymbol{\xi}_{n+1}^{\text{trial}}\right\|} \tag{A37}$$

The last term in equation (A36) renders the tangent operator unsymmetric.

REFERENCES

1. Artioli E, Auricchio F, Beirao da Veiga L. A novel 'optimal' exponential-based integration algorithm for von-Mises plasticity with linear hardening: theoretical analysis on yield consistency, accuracy, convergence and numerical investigations, *International Journal of Numerical Methods in Engineering,* No. 4, **67**(2006) 449-98.
2. Artioli E, Auricchio, F, Beirao da Veiga L. Generalized midpoint integration algorithms for J_2 plasticity with linear hardening, *International Journal of Numerical Methods in Engineering,* **72**(2007) 422-63.
3. Artioli E, Auricchio F, Beirao da Veiga L. Second order accurate integration algorithms for von-Mises plasticity with a nonlinear kinematic hardening mechanism, *Computer Methods in Applied Mechanics and Engineering,* **196**(2007) 1827-46.
4. Auricchio F, Taylor RL. Two material models for cyclic plasticity: nonlinear kinematic hardening and generalized plasticity, *International Journal of Plasticity,* **11**(1992) 65-98.
5. de Souza Neto EA, Peric D, Owen DRJ. *Computational Methods for Plasticity: Theory and Applications*, John Wiley & Sons Ltd, Chichester, UK, 2008.
6. Fuschi P, Peric D, Owen DRJ. Studies on generalized midpoint integration in rate-independent plasticity with reference to plane stress J_2-flow theory, *Computers and Structures,* No. 6, **43** (1992) 1117-33.
7. Krieg RD, Krieg DB. Accuracies of numerical solution methods for the elastic-perfectly plastic model, *Journal of Pressure Vessel Technology, Transaction of ASME,* **99**(1977) 510-5.
8. Ortiz M, Popov EP. Accuracy and stability of integration algorithms for elastoplastic constitutive relations, *International Journal of Numerical Methods in Engineering,* **21**(1985) 1561-76.
9. Ortiz M, Simo JC. An analysis of a new class of integration algorithms for elastoplastic constitutive relations, *International Journal of Numerical Methods in Engineering,* **23**(1986) 353-66.
10. Simo JC. Topics on the numerical analysis and simulation of plasticity, In *Handbook of*

Numerical Analysis, P.G. Ciarlet, J.L. Lions, Eds., Vol. 4, Elsevier, Amsterdam, Netherland, 1998.
11. Simo JC, Govindjee S. Non-linear B-stability and symmetry preserving return mapping algorithms for plasticity and visco-plasticity, *International Journal of Numerical Methods in Engineering,* **31**(1991) 151-76.
12. Simo JC, Hughes TJR. *Computational Inelasticity*, Springer Verlag, New York, USA, 1998.
13. Simo JC, Taylor RL. Consistent tangent operators for rate-independent elastoplasticity, *Computer Methods in Applied Mechanics and Engineering,* **48**(1985) 101-18.
14. Simo JC, Taylor RL. A return mapping algorithm for plane stress elastoplasticity, *International Journal Numerical Methods in Engineering,* **22**(1986) 649-70.
15. Wilkins ML. Calculation of elastic-plastic flow, In *Methods of Computational Physics*, B. Adler et al., Eds., Vol. 3, Academic Press, New York, USA, 1964.
16. Zienkiewicz OC, Taylor RL. *The Finite Element Method*, Vol. 2, 5th Ed., McGraw-Hill, New York, USA, 2002.