East Asian Journal on Applied Mathematics
doi: 10.4208/eajam.281018.100419

Vol. **10**, No. 1, pp. 1-21
February 2020

TTSCSP-Based Iteration Methods for Complex Weakly Nonlinear Systems

Tahereh Salimi Siahkolaei and Davod Khojasteh Salkuyeh*

Faculty of Mathematical Sciences, University of Guilan, Rasht, Iran.

Received 28 October 2018; Accepted (in revised version) 10 April 2019.

Abstract. We present Picard-TTSCSP and nonlinear TTSCSP-like iteration methods for systems of weakly nonlinear equations. These methods require solving two subsystems with constant positive definite coefficient matrices. Such subsystems are solved by the conjugate gradient method. Local convergence of the methods is established and numerical experiments demonstrate the efficiency of the methods.

AMS subject classifications: 65F10, 65F50, 65W05

Key words: Weakly nonlinear system, TTSCSP iteration method, inner/outer iteration scheme.

---

1. Introduction

Let $A \in \mathbb{C}^{n\times n}$ be a large, sparse, complex symmetric matrix, $\mathbb{D} \subset \mathbb{C}^n$ and $\phi : \mathbb{D} \to \mathbb{C}^n$ a continuously differentiable function. We consider the iterative solution of the systems of weakly nonlinear equations

$$Au = \phi(u) \text{ or equivalently } F(u) = Au - \phi(u) = 0, \tag{1.1}$$

with a matrix $A = W + iT$, where $W \in \mathbb{R}^{n\times n}$ and $T \in \mathbb{R}^{n\times n}$ are symmetric positive definite and symmetric positive semidefinite matrices, respectively. The system (1.1) is called weakly nonlinear if the linear term Au is strongly dominant over the term $\phi(u)$ in a norm [14, 30]. Such systems arise in various areas of scientific computing and engineering, including the discretisation of nonlinear partial differential equations [3,4,12,13,24], collocation of nonlinear integral equations [28], saddle point problems in image processing [6,15] and other applications [20].

The Newton iteration method [2, 17] is an efficient tool for solving systems of nonlinear equations (1.1). However, at each iteration step, the method uses a Jacobian matrix, the construction of which is costly and complicated. Various approaches to improve the efficiently of the method have been proposed — cf. Refs. [2–5, 7, 11, 14, 16, 18, 29]. In

---

*Corresponding author. *Email addresses:* `khojasteh@guilan.ac.ir` (D.K. Salkuyeh), `salimi_tahereh@phd.guilan.ac.ir` (T.S. Siahkolaei)

©2020 Global-Science Press

particular, recent work of Bai *et al.* [10], where Hermitian and skew-Hermitian splitting (HSS) iteration methods [9] are used as inner solvers for the Newton's method, introduces a new class of Newton-HSS methods for large sparse systems of nonlinear equations. Bai [3] proposed sequential two-stage iteration methods for nonlinear equations (1.1) — cf. also Refs. [5, 11]. Picard-HSS and nonlinear HSS-like iteration methods are developed in [14]. Zhu *et al.* [32] considered Picard-CSCS and nonlinear CSCS-like iteration methods for weakly nonlinear systems. Using the MHSS iteration [8] as an inner solver for the Picard method, Yang *et al.* [31] developed Picard-MHSS and the nonlinear MHSS-like iteration methods. These methods can be described as follows.

Picard-MHSS iteration method. Given an initial guess $u^{(0)} \in \mathbb{D}$ and a sequence $\{l_k\}_{k=0}^{\infty}$ of positive integers, for $k = 0, 1, 2, \ldots$ compute $u^{(k+1)}$ by the iteration scheme below until $\{u^{(k)}\}$ satisfies a stopping criterion:

(a) Set $u^{(k,0)} := u^{(k)}$.

(b) To obtain $u^{(k,l+1)}$ for $l = 0, 1, 2, \ldots, l_k - 1$, solve the following linear system:

$$\begin{aligned}(\alpha I + W)u^{(k,l+1/2)} &= (\alpha I - iT)u^{(k,l)} + \phi(u^{(k)}),\\ (\alpha I + T)u^{(k,l+1)} &= (\alpha I + iW)u^{(k,l+1/2)} - i\phi(u^{(k)}),\end{aligned}$$

where α and β are given positive constants.

(c) Set $u^{(k+1)} := u^{(k,l_k)}$.

Following the works [14, 31], Li *et al.* [21] used the lopsided PMHSS (LPMHSS) iterations [22] as an inner solver for the Picard method and developed Picard-LPMHSS and nonlinear LPMHSS-like iteration methods.

Picard-LPMHSS iteration method. Given an initial guess $u^{(0)} \in \mathbb{D}$ and a sequence $\{l_k\}_{k=0}^{\infty}$ of positive integers, for $k = 0, 1, 2, \ldots$ compute $u^{(k+1)}$ by the iteration scheme below until $\{u^{(k)}\}$ satisfies a stopping criterion:

(a) Set $u^{(k,0)} := u^{(k)}$.

(b) For $l = 0, 1, 2, \ldots, l_k - 1$, solve the following linear system to obtain $u^{(k,l+1)}$:

$$\begin{aligned}Wu^{(k,l+1/2)} &= -iTu^{(k,l)} + \phi(u^{(k)}),\\ (\alpha P + T)u^{(k,l+1)} &= (\alpha P + iW)u^{(k,l+1/2)} - i\phi(u^{(k)}),\end{aligned}$$

where α is a given positive constant and P a symmetric positive definite matrix.

(c) Set $u^{(k+1)} := u^{(k,l_k)}$.

Recently, Salkuyeh and Siahkolaei [26] proposed a two-parameter two-step scale-splitting (TTSCSP) method for systems of linear equations

$$(W + iT)x = b,$$

where $W \in \mathbb{R}^{n\times n}$ and $T \in \mathbb{R}^{n\times n}$ are, respectively, symmetric positive definite and symmetric positive semidefinite matrices and $b \in \mathbb{C}^n$. The TTSCSP method reduces to two-step scale-splitting (TSCSP) method from [25]. In this work, we employ the TTSCSP method as an inner solver in Picard iterations to solve large scale systems of weakly nonlinear equations (1.1).

The rest of this paper is organised as follows. In Section 2 we provide a brief description of the TTSCSP method for symmetric systems of linear equations. A Picard-TTSCSP method is considered in Section 3 and a nonlinear TTSCSP-like iteration method is discussed in Section 4. Inexact versions of the Picard-TTSCSP and nonlinear TTSCSP-like iteration methods are introduced in Section 5. Section 6 contains results of numerical experiments and our conclusions are in Section 7.

2. TTSCSP Iteration Method

If $\phi : \mathbb{D} \to \mathbb{C}^n$ is a constant vector — i.e. if $\phi(u) = b$, the system of weakly nonlinear equations (1.1) becomes the system of linear equations

$$Au = b, \tag{2.1}$$

where $A \in \mathbb{C}^{n\times n}$ and $u, b \in \mathbb{C}^n$. There are various methods to solve (2.1) — cf. Refs [8, 19, 22, 23, 25, 26, 30]. In particular, the authors of this paper split the coefficient matrix A and studied the convergence of the following TTSCSP method — cf. [26].

TTSCSP iteration method. Let $u^{(0)} \in \mathbb{C}^n$ be an initial guess. For $k = 0, 1, 2, \ldots$, until $\{u^{(k)}\}$ converges, compute $u^{(k+1)}$ according to the following scheme:

$$\begin{aligned} (\alpha W + T)u^{(k+1/2)} &= i(W - \alpha T)u^{(k)} + (\alpha - i)b, \\ (W + \beta T)u^{(k+1)} &= i(\beta W - T)u^{(k+1/2)} + (1 - \beta i)b, \end{aligned} \tag{2.2}$$

where α and β are positive numbers .

If $\alpha = \beta$, the TTSCSP method becomes TSCSP method and it was shown in [25] that for symmetric positive definite matrices W and T, the TSCSP method converges for any positive α. We write the TTSCSP iteration method in the matrix-vector form

$$u^{(k+1)} = \mathscr{G}(\alpha,\beta)u^{(k)} + \mathscr{C}(\alpha,\beta) = \mathscr{G}(\alpha,\beta)^{k+1}\, u^{(0)} + \sum_{j=0}^{k} \mathscr{G}(\alpha,\beta)^j \mathscr{C}(\alpha,\beta)b,$$

where

$$\mathscr{C}(\alpha,\beta) := (\alpha + \beta)(W + \beta T)^{-1}(W - iT)(\alpha W + T)^{-1}$$

and

$$\mathscr{G}(\alpha,\beta) := (W + \beta T)^{-1}(T - \beta W)(\alpha W + T)^{-1}(W - \alpha T)$$

is the iteration matrix of the method.

Setting

$$M := \frac{1}{\alpha+\beta}(\alpha W + T)(W - iT)^{-1}(W + \beta T),$$
$$N := \frac{1}{\alpha+\beta}(T - \beta W)(W - iT)^{-1}(W - \alpha T),$$

we have $A = M - N$ and $\mathscr{G}(\alpha,\beta) = M^{-1}N$.

3. Picard-TTSCSP Method

In what follows, we write $\|\cdot\|$ for the L^2-norm of vectors or matrices. Spectrum and spectral radius of a matrix A are, respectively, denoted by $\sigma(A)$ and $\rho(A)$ and the Kronecker product of A and B by $A \otimes B$.

If linear and nonlinear terms Au and $\phi(u)$ are well-separated and Au is strongly dominant over $\phi(u)$, one can apply the Picard iteration method

$$Au^{(k+1)} = \phi(u^{(k)}), \quad k = 0,1,2,\ldots \tag{3.1}$$

to the system (1.1) — cf. [1–3, 24]. At each step of the Picard iterations, we have to determine $u^{(k+1)}$ from the system of linear equations (3.1). This can be done by the TTSCSP iteration method from [26]. As the result, we arrive at the Picard-TTSCSP iteration method described as follows.

Picard-TTSCSP iteration method. Let $\phi : \mathbb{D} \to \mathbb{C}^n$ be a continuously differentiable function and $A = W + iT$, where W and T are symmetric positive definite and symmetric positive semidefinite matrices, respectively. Given an initial guess $u^{(0)} \in \mathbb{D}$ and a sequence $\{l_k\}_{k=0}^{\infty}$ of positive integers, for $k = 0,1,2,\ldots$ we compute $u^{(k+1)}$ by the iteration scheme below until $\{u^{(k)}\}$ satisfies a stopping criterion:

(a) Set $u^{(k,0)} := u^{(k)}$.

(b) For $l = 0,1,2,\ldots,l_k - 1$ determine $u^{(k,l+1)}$ from the linear systems

$$(\alpha W + T)u^{(k,l+1/2)} = i(W - \alpha T)u^{(k,l)} + (\alpha - i)\phi(u^{(k)}),$$
$$(W + \beta T)u^{(k,l+1)} = i(\beta W - T)u^{(k,l+1/2)} + (1 - \beta i)\phi(u^{(k)}),$$

where α and β are given positive constants.

(c) Set $u^{(k+1)} := u^{(k,l_k)}$.

Note that as an inner solver in the Picard iterations (3.1), we can also use the TSCSP method, thus obtaining the Picard-TSCSP iteration method. If $\alpha = \beta$, the Picard-TTSCSP iteration method becomes the Picard-TSCSP iteration method.

In Picard-TTSCP and nonlinear TTSCSP-like methods, the coefficient matrices of the corresponding linear subsystems are symmetric positive definite. Therefore, these subsystems can be exactly solved by Cholesky factorisation of the coefficient matrices or inexactly by the conjugate gradient (CG) method. Using Picard-TTSCSP iterations, we determine $u^{(k+1)}$ in the form

$$u^{(k+1)} = \mathscr{G}(\alpha,\beta)^{l_k} u^{(k)} + \sum_{j=0}^{l_k-1} \mathscr{G}(\alpha,\beta)^j \mathscr{C}(\alpha,\beta)\phi(u^{(k)}), \quad k = 0,1,\ldots. \tag{3.2}$$

Suppose that the vector $u^* \in \mathbb{D}$ is the solution of the system (1.1). It is easily seen that

$$u^* = \mathscr{G}(\alpha,\beta)^{l_k} u^* + \sum_{j=0}^{l_k-1} \mathscr{G}(\alpha,\beta)^j \mathscr{C}(\alpha,\beta)\phi(u^*), \quad k = 0,1,\ldots. \tag{3.3}$$

Subtracting (3.3) from (3.2) yields

$$u^{(k+1)} - u^* = \mathscr{G}(\alpha,\beta)^{l_k}(u^{(k)} - u^*) + \sum_{j=0}^{l_k-1} \mathscr{G}(\alpha,\beta)^j \mathscr{C}(\alpha,\beta)\big[\phi(u^k) - \phi(u^*)\big].$$

Let a be a real number. By $\lfloor a \rfloor$ we denote the smallest integer such that $a \leq \lfloor a \rfloor$. Setting

$$\theta(\alpha,\beta) :=\| \mathscr{G}(\alpha,\beta) \|, \quad \eta :=\| A^{-1}\phi'(u^*) \|$$

and taking into account [14, Theorem 3.1], we obtain the following result.

Theorem 3.1. *Let W and T be, respectively, symmetric positive definite and symmetric positive semidefinite matrices and $A = W + iT$. If $\phi : \mathbb{D} \to \mathbb{C}^n$ is a G-differentiable function in an open neighborhood $\mathbb{N}_0 \subset \mathbb{D}$ of the point $u^* \in \mathbb{D}$ such that $\phi'(u^*)$ is continuous and $F(u^*) = Au^* - \phi(u^*) = 0$, then there is an open neighborhood $\mathbb{N} \subset \mathbb{N}_0$ of u^* such that for any $u^{(0)} \in \mathbb{N}$ and any sequence of positive integers l_k, $k = 0,1,2,\ldots$, the iteration sequence $\{u^{(k)}\}_{k=0}^{\infty}$ generated by the Picard-TTSCSP iteration method is well-defined and converges to u^*, provided that $\eta < 1$ and*

$$l_0 \geqslant \left\lfloor \ln\left(\left(\frac{1-\eta}{1+\eta}\right) / \ln(\theta(\alpha,\beta))\right) \right\rfloor.$$

Moreover,

$$\limsup_{k\to\infty} \| u^{(k)} - u^* \|^{1/k} \leqslant \eta + (1+\eta)\theta(\alpha,\beta)^{l_0}, \tag{3.4}$$

where $l_0 = \liminf_{k\to\infty} l_k$. In particular, if $\lim_{k\to\infty} l_k = \infty$, then the convergence rate is R-linear and

$$\limsup_{k\to\infty} \| u^k - u^* \|^{1/k} \leqslant \eta. \tag{3.5}$$

Proof. The proof runs similar to the proof of the convergence of the Picard-HSS iteration method in [14]. □

Theorem 3.1 shows that the convergence rate of the Picard-TTSCSP method depends on $\theta(\alpha,\beta)$ and η, so that small $\theta(\alpha,\beta)$ and η imply the fast convergence of Picard-TTSCSP iterations.

4. Nonlinear TTSCSP-Like Method

The main drawback of the Picard-TTSCSP iteration method is that in actual computations, the data in the inner iteration steps $l_k, k = 0, 1, 2, \ldots$ are often problem-dependent and are difficult to determine. Since the problem is related to the nonlinearity of the fixed-point equations

$$\begin{aligned}(\alpha W + T)u &= i(W - \alpha T)u + (\alpha - i)\phi(u),\\ (W + \beta T)u &= i(\beta W - T)u + (1 - \beta i)\phi(u),\end{aligned}$$

here we propose a nonlinear TTSCSP-like iteration method.

Nonlinear TTSCSP-like method. Let $\phi : \mathbb{D} \to \mathbb{C}^n$ be a continuously differentiable function and $A = W + iT$, where W and T symmetric positive definite and symmetric positive semidefinite matrices, respectively. Given an initial guess $u^{(0)} \in \mathbb{D}$, for $k = 0, 1, 2, \ldots$ compute $u^{(k+1)}$ by the iteration scheme below until $\{u^{(k)}\}$ satisfies a stopping criterion:

$$\begin{aligned}(\alpha W + T)u^{(k+1/2)} &= i(W - \alpha T)u^{(k)} + (\alpha - i)\phi(u^{(k)}),\\ (W + \beta T)u^{(k+1)} &= i(\beta W - T)u^{(k+1/2)} + (1 - \beta i)\phi(u^{(k+1/2)}),\end{aligned}$$

where α and β are positive constants.

We want to study the convergence of the TTSCSP-like iteration method. Writing

$$\begin{aligned}F(u) &= (\alpha W + T)^{-1}[i(W - \alpha T)u + (\alpha - i)\phi(u)],\\ V(u) &= (W + \beta T)^{-1}[i(\beta W - T)u + (1 - \beta i)\phi(u)],\end{aligned}$$

and introducing the function $\psi(u) := VoF(u) = V(F(u))$, we represent the nonlinear TTSCSP-like iterations as

$$u^{(k+1)} = \psi(u^{(k)}), \quad k = 0, 1, 2, \ldots.$$

Taking into account Ostrowski theorem — cf. [14, Theorem 10.1.3], we observe that if $\rho(\psi'(u^*)) < 1$, then u^* is a point of attraction of the nonlinear TTSCSP-like iterations. Assuming that $u^* \in \mathbb{D}$ is a solution of the system (1.1), we can easily verify the identities

$$\begin{aligned}&F(u^*) = u^*, \quad V(u^*) = u^*,\\ &F'(u^*) = (\alpha W + T)^{-1}[i(W - \alpha T) + (\alpha - i)\phi'(u^*)],\\ &V'(u^*) = (W + \beta T)^{-1}[i(\beta W - T) + (1 - \beta i)\phi'(u^*)],\end{aligned}$$

and the chain rule — cf. [24, Theorem 3.1.7], yields

$$\begin{aligned}\psi'(u^*) =& V'(u^*)F'(u^*) = (W + \beta T)^{-1}[i(\beta W - T) + (1 - \beta i)\phi'(u)]\\ &\times (\alpha W + T)^{-1}[i(W - \alpha T) + (\alpha - i)\phi'(u)].\end{aligned}$$

Summarising the results above, we state the following theorem.

Theorem 4.1. *Let* $A = W + iT$, *where* W *and* T *are, respectively, symmetric positive definite and symmetric positive semidefinite matrices and*

$$\begin{aligned}\mathscr{G}(\alpha,\beta;u^*) =&(W+\beta T)^{-1}[i(\beta W - T) + (1-\beta i)\phi'(u^*)]\\ &\times (\alpha W + T)^{-1}[i(W-\alpha T) + (\alpha - i)\phi'(u^*)].\end{aligned}$$

If $\phi : \mathbb{D} \to \mathbb{C}^n$ *is an* F*-differentiable function at a point* $u^* \in \mathbb{D}$ *such that* $Au^* = \phi(u^*)$ *and* $\mathscr{G}(\alpha,\beta;u^*) < 1$, *then* $u^* \in \mathbb{D}$ *is a point of attraction of the nonlinear TTSCSP-like iteration method.*

Theorem 4.2. *Assume that the conditions of Theorem* 4.1 *are satisfied and let*

$$\begin{aligned}&\delta := \max\left\{\| \phi'(u^*)(W+\beta T)^{-1} \|, \| \phi'(u^*)(\alpha W + T)^{-1} \|\right\},\\ &a := \max\left\{\left|\frac{1-\alpha\mu_1}{\alpha+\mu_1}\right|, \left|\frac{1-\alpha\mu_n}{\alpha+\mu_n}\right|\right\},\\ &b := \max\left\{\left|\frac{\beta-\mu_1}{1+\beta\mu_1}\right|, \left|\frac{\beta-\mu_n}{1+\beta\mu_n}\right|\right\},\end{aligned}$$

where μ_1 *and* μ_n *are the smallest and the largest eigenvalues of* $W^{-1}T$. *If*

$$ab + \delta\left(b\sqrt{1+\alpha^2} + a\sqrt{1+\beta^2}\right) + \delta^2\sqrt{(1+\alpha^2)(1+\beta^2)} < 1, \tag{4.1}$$

then

$$\rho(\mathscr{G}(\alpha,\beta,u^*)) < 1.$$

Proof. By simple computations, we have

$$\begin{aligned}&(W+\beta T)\mathscr{G}(\alpha,\beta;u^*)(W+\beta T)^{-1}\\ =&(W+\beta T)\mathscr{G}(\alpha,\beta)(W+\beta T)^{-1}\\ &+(1+\alpha i)(\beta W - T)(\alpha W + T)^{-1}\phi'(u^*)(W+\beta T)^{-1}\\ &+(\beta+i)\phi'(u^*)(\alpha W+T)^{-1}(W-\alpha T)(W+\beta T)^{-1}\\ &+(1-\beta i)(\alpha - i)\phi'(u^*)(\alpha W+T)^{-1}\phi'(u^*)(W+\beta T)^{-1},\end{aligned}$$

and

$$\begin{aligned}\|\mathscr{G}(\alpha,\beta)\| &= \|(\alpha W+T)^{-1}(W-\alpha T)(W+\beta T)^{-1}(T-\beta W)\|\\ &= \|(\alpha I + S)^{-1}(I-\alpha S)(I+\beta S)^{-1}(\beta I - S)\|\\ &\leqslant \|(\alpha I+S)^{-1}(I-\alpha S)\|\|(I+\beta S)^{-1}(\beta I - S)\|\\ &= \max_{\mu\in\sigma(S)}\left\{\left|\frac{1-\alpha\mu}{\alpha+\mu}\right|\right\}\max_{\mu\in\sigma(S)}\left\{\left|\frac{\beta-\mu}{1+\beta\mu}\right|\right\}\\ &= \max\left\{\left|\frac{1-\alpha\mu_1}{\alpha+\mu_1}\right|, \left|\frac{1-\alpha\mu_n}{\alpha+\mu_n}\right|\right\}\max\left\{\left|\frac{\beta-\mu_1}{1+\beta\mu_1}\right|, \left|\frac{\beta-\mu_n}{1+\beta\mu_n}\right|\right\} = ab.\end{aligned}$$

Hence

$$\begin{aligned}
\|\mathscr{G}(\alpha,\beta;u^*)\| &= \left\|(W+\beta T)\mathscr{G}(\alpha,\beta;u^*)(W+\beta T)^{-1}\right\| \\
&\leqslant \left\|(W+\beta T)\mathscr{G}(\alpha,\beta)(W+\beta T)^{-1}\right\| \\
&\quad + \left\|(1+\alpha i)(\beta W-T)(\alpha W+T)^{-1}\phi'(u^*)(W+\beta T)^{-1}\right\| \\
&\quad + \left\|(\beta+i)\phi'(u^*)(\alpha W+T)^{-1}(W-\alpha T)(W+\beta T)^{-1}\right\| \\
&\quad + \left\|(1-\beta i)(\alpha-i)\phi'(u^*)(\alpha W+T)^{-1}\phi'(u^*)(W+\beta T)^{-1}\right\| \\
&\leqslant \left\|\mathscr{G}(\alpha,\beta)\right\| + \left\|(1+\alpha i)(\beta W-T)(W+\beta T)^{-1}\right\| \left\|\phi'(u^*)(\alpha W+T)^{-1}\right\| \\
&\quad + \left\|(\beta+i)(\alpha W+T)^{-1}(W-\alpha T)\right\| \left\|\phi'(u^*)(W+\beta T)^{-1}\right\| \\
&\quad + \left|(1-\beta i)(\alpha-i)\right| \left\|\phi'(u^*)(\alpha W+T)^{-1}\right\| \left\|\phi'(u^*)(W+\beta T)^{-1}\right\| \\
&\leqslant ab + \delta\left(b\sqrt{1+\alpha^2}+a\sqrt{1+\beta^2}\right) + \delta^2\sqrt{(1+\alpha^2)(1+\beta^2)}.
\end{aligned}$$

The condition (4.1) yields

$$\rho(\mathscr{G}(\alpha,\beta;u^*)) \le \|\mathscr{G}(\alpha,\beta;u^*)\| < 1,$$

and the proof is completed. ☐

Remark 4.1. The inequality (4.1) holds if $ab < 1$ and δ is sufficiently small. We note that sufficient conditions for $ab < 1$ are presented in [26].

Remark 4.2. For nonlinear TSCSP-like iteration method, the convergence conditions are the same as in Theorem 4.1 but one has to set $\alpha = \beta$.

The convergence speed of the iteration methods for the system (1.1) depends on two factors — viz. on the weak nonlinearity of the system and on optimality of the parameters. These factors are problem-based and are difficult to handle. However, as is shown in [26], the spectral radius of the iteration matrix $\mathscr{G}(\alpha,\beta)$ satisfies the inequality

$$\begin{aligned}
&\rho(\mathscr{G}(\alpha,\beta)) \le \|\mathscr{G}(\alpha,\beta)\| \\
&\le \max\left\{\left|\frac{1-\alpha\mu_1}{\alpha+\mu_1}\right|, \left|\frac{1-\alpha\mu_n}{\alpha+\mu_n}\right|\right\} \max\left\{\left|\frac{\beta-\mu_1}{1+\beta\mu_1}\right|, \left|\frac{\beta-\mu_n}{1+\beta\mu_n}\right|\right\} := \sigma(\alpha,\beta),
\end{aligned}$$

and

$$(\alpha^*,\beta^*) = \arg\min_{\alpha,\beta>0} \sigma(\alpha,\beta),$$

where

$$\alpha^* = \frac{1-\mu_1\mu_n+\sqrt{(1-\mu_1\mu_n)^2+(\mu_1+\mu_n)^2}}{\mu_1+\mu_n}, \quad \beta^* = \frac{1}{\alpha^*}. \tag{4.2}$$

Let us point out that α^* and β^* minimise the upper bound $\sigma(\alpha,\beta)$ of the spectral radius $\mathscr{G}(\alpha,\beta)$, but not $\mathscr{G}(\alpha,\beta;u^*)$. On the other hand, if the linear term Au is strongly dominant over the nonlinear term $\phi(u)$, one can use the parameters α^* and β^* in the implementation of the method and forthcoming numerical experiments confirm that these parameters often provide satisfactory results.

5. Inexact Picard-TTSCSP and Nonlinear Inexact TTSCSP-Like Methods

In order to determine $u^{(k+1)}$ by Picard-TTSCSP or by nonlinear TTSCSP-like iteration methods, one has to solve two subsystems with the coefficient matrices $\alpha W+T$ and $W+\beta T$. The methods can be improved if one employs iteration methods for solving the subproblems. Since $\alpha W+T$ and $W+\beta T$ are symmetric positive definite matrices, these subsystems can be solved inexactly by the CG method such that the relative residual norms are smaller than $\epsilon_{1k}>0$ and $\epsilon_{2k}>0$, respectively.

In the Picard-TTSCSP iteration method, we suppose that

$$u^{(k,l+1/2)}=u^{(k,l)}+z^{(k,l)}$$

and substitute it in the first subsystem, so that

$$(\alpha W+T)z^{(k,l)}=(\alpha-i)r^{(k,l)}, \tag{5.1}$$

where $r^{(k,l)}=\phi(u^{(k)})-Au^{(k,l)}$. Analogously, letting

$$u^{(k,l+1)}=u^{(k,l+1/2)}+z^{(k,l+1/2)},$$

we write the second subsystem as

$$(W+\beta T)z^{(k,l+1/2)}=(1-\beta i)r^{(k,l+1/2)}, \tag{5.2}$$

where $r^{(k,l+1/2)}=\phi(u^k)-Au^{(k,l+1/2)}$. We inexactly solve the systems (5.1) and (5.2) by the CG method. The nonlinear inexact TTSCSP-like iteration method is established in the same way. The algorithms obtained have the following form:

Algorithm 5.1 Inexact Picard-TTSCSP (Picard-ITTSCSP) iteration method.

```
Choose an initial guess $u^{(0)}$.
for $k=0,1,2,\dots$ until convergence do
  Set $b:=\phi(u^k)$.
  Set $u^{(k,0)}:=u^{(k)}$.
  for $l=0,1,2,\dots,l_k-1$ do
    Compute $r^{(k,l)}=b-Au^{(k,l)}$.
    Set $\bar{r}^{(k,l)}=(\alpha-i)r^{(k,l)}$.
    Solve $(\alpha W+T)z^{(k,l)}=\bar{r}^{(k,l)}$ by CG to compute the approximate solution $z^{(k,l)}$
    satisfying $\|\bar{r}^{(k,l)}-(\alpha W+T)z^{(k,l)}\|_2\leqslant\epsilon_{1k}\|\bar{r}^{(k,l)}\|_2$, $u^{(k,l+1/2)}:=u^{(k,l)}+z^{(k,l)}$.
    Compute $r^{(k,l+1/2)}=b-Au^{(k,l+1/2)}$.
    Set $\bar{r}^{(k,l+1/2)}=(1-\beta i)r^{(k,l+1/2)}$.
    Solve $(W+\beta T)z^{(k,l+1/2)}=\bar{r}^{(k,l+1/2)}$ by CG to compute the approximate
    solution $z^{(k,l+1/2)}$ satisfying $\|\bar{r}^{(k,l+1/2)}-(W+\beta T)z^{(k,l+1/2)}\|_2\leqslant\epsilon_{2k}\|\bar{r}^{(k,l+1/2)}\|_2$,
```

```
    $u^{(k,l+1)} := u^{(k,l+1/2)} + z^{(k,l+1/2)}$.
  end for
  Set $u^{(k+1)} := u^{(k,l_k)}$.
end for
```

Algorithm 5.2 Inexact TTSCSP-like (ITTSCSP-like) iteration method.

```
Choose an initial guess $u^{(0)}$.
for $k = 0, 1, 2, \dots$ until convergence do
  Compute $r^{(k)} = \phi(u^{(k)}) - Au^{(k)}$.
  Set $\bar{r}^{(k)} = (\alpha - i)r^{(k)}$.
  Solve $(\alpha W + T)z^{(k)} = \bar{r}^{(k)}$ by CG to compute the approximate solution $z^{(k)}$
  satisfying $\|\bar{r}^{(k)} - (\alpha W + T)z^{(k)}\|_2 \leqslant \epsilon_{1k}\|\bar{r}^{(k)}\|_2$, $u^{(k+1/2)} := u^{(k)} + z^{(k)}$.
  Compute $r^{(k+1/2)} = \phi(u^{(k+1/2)}) - Au^{(k+1/2)}$.
  Set $\bar{r}^{(k+1/2)} = (1 - \beta i)r^{(k+1/2)}$.
  Solve $(W + \beta T)z^{(k+1/2)} = \bar{r}^{(k+1/2)}$ by CG to compute the approximate
  solution $z^{(k+1/2)}$ satisfying $\|\bar{r}^{(k+1/2)} - (W + \beta T)z^{(k+1/2)}\|_2 \leqslant \epsilon_{2k}\|\bar{r}^{(k+1/2)}\|_2$.
  Set $u^{(k+1)} := u^{(k+1/2)} + z^{(k+1/2)}$.
end for
```

6. Numerical Experiments

Let us now compare the Picard-TTSCSP method with Picard-MHSS, Picard-LPMHSS and Picard-TSCSP iterations as well as the nonlinear TTSCSP-like method with nonlinear MHSS-like, LPMHSS-like and TSCSP-like processes.

Consider the two-dimensional nonlinear convection-diffusion equation

$$\begin{aligned} &u_t - (\alpha_1 + i\beta_1)(u_{xx} + u_{yy}) + \varrho u = (\alpha_2 + i\beta_2)\, ue^u + \sin\sqrt{1 + u_x^2 + u_y^2}, \\ &u(0,x,y) = u_0(x,y), \quad (x,y) \in \Omega, \\ &u(t,x,y) = 0, \quad (t,x,y) \in (0,1] \times \partial\Omega, \end{aligned} \tag{6.1}$$

where $\Omega = (0,1) \times (0,1)$, $\partial\Omega$ is the boundary of Ω, $\alpha_1 = \beta_1 = 1$, $\alpha_2 = \beta_2 = 0.5$ and ϱ is a positive constant controlling the magnitude of the reaction term — cf. [21]. Using the equidistant grid $\Delta t = h = 1/(N+1)$ at each temporal step of the implicit scheme for the Eq. (6.1), we arrive at the system of weakly nonlinear equations (1.1) of the form

$$F(u) = Mu - h^2\phi(u) = 0,$$

where

$$\begin{aligned} M &= h(1 + \varrho\Delta t)I_n + (\alpha_1 + i\beta_1)(A_n \otimes I + I \otimes A_n), \\ \phi(u) &= (\alpha_2 + i\beta_2)\psi(u) + \sin(1 + B(u)) \end{aligned}$$

and

$$\psi(u) = (u_1 e^{u_1}, u_2 e^{u_2}, \dots, u_n e^{u_n})^T,$$

$$\begin{aligned}
&\sin(u) = (\sin(u_1), \sin(u_2), \dots, \sin(u_n))^T,\\
&A_N = \text{tridiag}(-1, 2, -1), \quad B = C_N \otimes C_N,\\
&C_N = \text{tridiag}(-1/h, 0, 1/h), \quad n = N \times N.
\end{aligned}$$

In all numerical experiments, the zero vector is used as the initial guess and the stopping criterion is

$$\frac{\|F(u^{(k)})\|_2}{\|F(u^{(0)})\|_2} < 10^{-6}.$$

For the inner iterations in Picard-MHSS, Picard-LPMHSS, Picard-TSCSP and Picard-TTSCSP methods, the stopping criterion is

$$\frac{\|F(u^{(k,l_k)})\|_2}{\|F(u^{(k,0)})\|_2} < \eta_k,$$

where l_k denotes the number of inner iteration steps. If η_k is fixed for all k, then we write η for η_k. The coefficient matrices of the subsystems in the Picard-TTSCSP and the nonlinear TTSCSP-like iteration methods are symmetric positive definite and in the exact versions of these algorithms, the subsystems are solved via Cholesky factorisation of the coefficient matrices. In the inexact implementation of the algorithms, the corresponding subsystems are solved by the CG method. The CG iterations are stopped when the residual norm is reduced by a factor of 10^2. The maximum number of iterations in the CG method is set to 1000.

All runs are implemented in MATLAB R2014b on a laptop with 2.40 GHz central processing unit (Intel(R) Core(TM) i7-5500), 8 GB memory and Windows 10 operating system. Numerical experiments are carried out for $N = 32, 64, 128$ — i.e. for $n = 32^2, 64^2, 128^2$ and for the tolerance $\eta = 0.1$, $\eta = 0.01$ and $\eta = 0.001$. The terms IT_{int}, IT_{out}, IT, CPU show the average number of the inner iterations, the outer iteration, the total iteration and the total CPU-time, respectively. As was already mentioned, optimal parameters in the methods tested are problem-based even if the nonlinear term $\phi(u)$ is neglected. However, in the TTSCSP-based methods, the parameters α^* and β^* often provide suitable results — cf. (4.2). The optimal parameters α_{opt} and β_{opt} used in all the experiments are found experimentally and ensure the smallest number of iterations. The parameters α_{opt} and β_{opt} in Picard-MHSS, Picard-LPMHSS, Picard-TSCSP and Picard-TTSCSP are presented in Tables 1-3.

Table 7 compares Picard-TTSCSP and Picard-MHSS, Picard-LPMHSS and Picard-TSCSP iteration methods. The Picard-HSS, Picard-LPMHHS, Picard-TSCSP and Picard-TTSCSP are respectively denoted by “P-MHSS”, “P-LPMHHS”, “P-TSCSP” and “P-TTSCSP”. It shows that the Picard-TSCSP and Picard-TTSCSP methods outperform other iteration methods in terms of iteration numbers and CPU time. For nonlinear TTSCSP-like, MHSS-like, the LPMHSS-like and TSCSP-like iteration methods, numerical results are displayed in Table 10 and they show that TTSCSP-like method performs better then others. Table 9 demonstrates the work of the Picard-TTSCSP method if optimal parameters α_{opt}, β_{opt} and the parameters α^* and β^* obtained from the Eq. (4.2) are used. We note that there is no substantial difference in results concerning iteration numbers and CPU time.

Table 1: Experimentally optimal parameters for various iteration methods, $N = 32$.

	Iteration	ϱ	0.1	1	10
$\eta = 0.1$	Picard-MHSS	α_{opt}	0.34	0.34	0.35
	Picard-LPMHSS	α_{opt}	1.3	1.3	1.5
	Picard-TSCSP	α_{opt}	0.5	0.5	0.5
	Picard-TTSCSP	α_{opt}	1.17	1.17	1.17
		β_{opt}	0.31	0.30	0.30
$\eta = 0.01$	Picard-MHSS	α_{opt}	0.33	0.34	0.36
	Picard-LPMHSS	α_{opt}	1.0	1.0	1.0
	Picard-TSCSP	α_{opt}	0.45	0.45	0.45
	Picard-TTSCSP	α_{opt}	1.17	1.17	1.17
		β_{opt}	0.30	0.30	0.30
$\eta = 0.001$	Picard-MHSS	α_{opt}	0.34	0.34	0.34
	Picard-LPMHSS	α_{opt}	1.5	1.5	1.6
	Picard-TSCSP	α_{opt}	0.43	0.43	0.42
	Picard-TTSCSP	α_{opt}	1.17	1.17	1.17
		β_{opt}	0.30	0.30	0.30

Table 2: Experimentally optimal parameters for various iteration methods, $N = 64$.

	Iteration	ϱ	0.1	1	10
$\eta = 0.1$	Picard-MHSS	α_{opt}	0.20	0.20	0.20
	Picard-LPMHSS	α_{opt}	1.1	1.1	1.1
	Picard-TSCSP	α_{opt}	0.36	0.36	0.36
	Picard-TTSCSP	α_{opt}	1.17	1.17	1.17
		β_{opt}	0.30	0.30	0.30
$\eta = 0.01$	Picard-MHSS	α_{opt}	0.20	0.20	0.21
	Picard-LPMHSS	α_{opt}	1.0	1.0	1.0
	Picard-TSCSP	α_{opt}	0.36	0.36	0.35
	Picard-TTSCSP	α_{opt}	1.17	1.17	1.17
		β_{opt}	0.30	0.30	0.30
$\eta = 0.001$	Picard-MHSS	α_{opt}	0.19	0.19	0.20
	Picard-LPMHSS	α_{opt}	0.9	0.9	0.9
	Picard-TSCSP	α_{opt}	0.35	0.35	0.35
	Picard-TTSCSP	α_{opt}	1.17	1.17	1.17
		β_{opt}	0.30	0.30	0.30

Tables 8-11 provide numerical results for the inexact versions of Picard-TTSCSP and nonlinear TTSCSP-like methods for three values of η, where inexact version of Picard-MHSS, Picard-LPMHSS, Picard-TSCSP and Picard-TTSCSP are denoted by “P-IMHSS”, “P-ILPMHSS”, “P-ITSCSP and “P-ITTSCSP, respectively. The optimal values of the parameters

Table 3: Experimentally optimal parameters for various iteration methods, $N = 128$.

	Iteration	ϱ	0.1	1	10
$\eta = 0.1$	Picard-MHSS	α_{opt}	0.12	0.12	0.12
	Picard-LPMHSS	α_{opt}	0.80	0.80	0.80
	Picard-TSCSP	α_{opt}	0.26	0.24	0.26
	Picard-TTSCSP	α_{opt}	1.17	1.17	1.17
		β_{opt}	0.20	0.20	0.20
$\eta = 0.01$	Picard-MHSS	α_{opt}	0.13	0.13	0.13
	Picard-LPMHSS	α_{opt}	0.80	0.80	0.80
	Picard-TSCSP	α_{opt}	0.24	0.24	0.25
	Picard-TTSCSP	α_{opt}	1.17	1.17	1.17
		β_{opt}	0.30	0.30	0.30
$\eta = 0.001$	Picard-MHSS	α_{opt}	0.12	0.12	0.12
	Picard-LPMHSS	α_{opt}	0.80	0.80	0.80
	Picard-TSCSP	α_{opt}	0.23	0.23	0.23
	Picard-TTSCSP	α_{opt}	1.17	1.17	1.17
		β_{opt}	0.30	0.30	0.30

Table 4: Experimentally optimal parameters for various inexact iteration methods, $N = 32$.

	Iteration	ϱ	0.1	1	10
$\eta = 0.1$	Picard-IMHSS	α_{opt}	0.43	0.43	0.45
	Picard-ILPMHSS	α_{opt}	0.76	0.76	0.74
	Picard-ITSCSP	α_{opt}	0.50	0.49	0.44
	Picard-ITTSCSP	α_{opt}	1.17	1.17	1.17
		β_{opt}	0.30	0.30	0.30
$\eta = 0.01$	Picard-IMHSS	α_{opt}	0.43	0.42	0.45
	Picard-ILPMHSS	α_{opt}	0 .83	0.83	0.83
	Picard-ITSCSP	α_{opt}	0.45	0.45	0.45
	Picard-ITTSCSP	α_{opt}	1.17	1.17	1.17
		β_{opt}	0.30	0.30	0.30
$\eta = 0.001$	Picard-MHSS	α_{opt}	0.40	0.41	0.42
	Picard-ILPMHSS	α_{opt}	0.83	0.83	0.81
	Picard-ITSCSP	α_{opt}	0.43	0.43	0.43
	Picard-ITTSCSP	α_{opt}	1.17	1.17	1.17
		β_{opt}	0.30	0.30	0.30

α and β are given in Tables 4-6. The inexact version of the nonlinear MHSS-like, LPMHSS-like, TSCSP-like and TTSCSP-like are denoted by “IMHSS-like”, “ILPMHSS-like”, “ITSCSP-like” and “ITTSCSP-like”, respectively. We also note that in both exact and inexact versions of Picard-TTSCSP and nonlinear TTSCSP-like iteration methods, the optimal values of the

Table 5: Experimentally optimal parameters for various inexact iteration methods, $N = 64$.

	Iteration	ϱ	0.1	1	10
$\eta = 0.1$	Picard-IMHSS	α_{opt}	0.26	0.26	0.27
	Picard-ILPMHSS	α_{opt}	0.78	0.78	0.78
	Picard-ITSCSP	α_{opt}	0.35	0.35	0.31
	Picard-ITTSCSP	α_{opt}	1.17	1.17	1.17
		β_{opt}	0.30	0.30	0.30
$\eta = 0.01$	Picard-IMHSS	α_{opt}	0.27	0.26	0.27
	Picard-ILPMHSS	α_{opt}	0.87	0.87	087
	Picard-ITSCSP	α_{opt}	0.35	0.35	0.35
	Picard-ITTSCSP	α_{opt}	1.17	1.17	1.17
		β_{opt}	0.30	0.30	0.30
$\eta = 0.001$	Picard-IMHSS	α_{opt}	0.25	0.25	0.26
	Picard-ILPMHSS	α_{opt}	0.87	0.879	0.86
	Picard-ITSCSP	α_{opt}	0.32	0.32	0.32
	Picard-ITTSCSP	α_{opt}	1.17	1.17	1.17
		β_{opt}	0.30	0.30	0.30

Table 6: Experimentally optimal parameters for various inexact iteration methods, $N = 128$.

	Iteration	ϱ	0.1	1	10
$\eta = 0.1$	Picard-IMHSS	α_{opt}	0.17	0.17	0.18
	Picard-ILPMHSS	α_{opt}	0.80	0.80	0.80
	Picard-ITSCSP	α_{opt}	0.27	0.28	0.27
	Picard-ITTSCSP	α_{opt}	1.17	1.17	1.17
		β_{opt}	0.30	0.30	0.30
$\eta = 0.01$	Picard-IMHSS	α_{opt}	0.18	0.18	0.18
	Picard-ILPMHSS	α_{opt}	0.80	0.80	0.80
	Picard-ITSCSP	α_{opt}	0.28	0.28	0.28
	Picard-ITTSCSP	α_{opt}	1.16	1.17	1.17
		β_{opt}	0.30	0.30	0.30
$\eta = 0.001$	Picard-IMHSS	α_{opt}	0.17	0.17	0.17
	Picard-ILPMHSS	α_{opt}	0.74	0.74	0.74
	Picard-ITSCSP	α_{opt}	0.24	0.24	0.23
	Picard-ITTSCSP	α_{opt}	1.17	1.17	1.17
		β_{opt}	0.30	0.30	0.30

parameters α and β do not depend on the problem size. We see that both of the inexact Picard-TTSCSP and the TTSCSP-like methods provide suitable results for different values of η and N.

Table 7: Numerical results for various Picard iteration methods.

			$N=32$			$N=64$			$N=128$		
η	Iteration	ϱ	0.1	1	10	0.1	1	10	0.1	1	10
0.1	P-MHSS	IT_{int}	1.01	1.01	1.02	1.01	1.01	1.01	1.01	1.01	1.01
		IT_{out}	70	70	67	107	106	103	155	155	153
		IT	71.01	71.01	68.02	108.01	107.01	104.01	156.01	156.01	156.01
		CPU	0.082	0.084	0.079	0.358	0.359	0.348	2.166	2.142	2.116
	P-LPMHSS	IT_{int}	1.03	1.03	1.03	1.03	1.03	1.03	1.03	1.03	1.03
		IT_{out}	35	35	35	34	34	34	34	34	34
		IT	36.03	36.03	36.03	35.03	35.03	35.03	35.03	35.03	35.03
		CPU	0.048	0.047	0.048	0.169	0.165	0.170	1.171	1.169	1.172
	P-TSCSP	IT_{int}	1.2	1.2	1.2	1.14	1.14	1.25	1.11	1.11	1.11
		IT_{out}	6	6	6	8	8	9	10	10	9
		IT	7.2	7.2	7.2	9.14	9.14	10.25	11.11	11.11	10.11
		CPU	0.025	0.025	0.025	0.065	0.065	0.067	0.398	0.397	0.393
	P-TTSCSP	IT_{int}	1.33	1.33	1.33	1.25	1.25	1.25	1.25	1.25	1.25
		IT_{out}	4	4	4	5	5	5	5	5	5
		IT	5.33	5.33	5.33	6.25	6.25	6.25	6.25	6.25	6.25
		CPU	0.024	0.024	0.024	0.057	0.057	0.056	0.318	0.310	0.323
0.01	P-MHSS	IT_{int}	1.02	1.02	1.02	1.02	1.02	1.02	1.01	1.01	1.01
		IT_{out}	57	57	53	84	83	79	122	122	117
		IT	58.02	58.02	54.02	85.02	84.02	80.02	123.01	123.01	118.01
		CPU	0.079	0.078	0.072	0.340	0.331	0.315	2.057	2.023	1.960
	P-LPMHSS	IT_{int}	1.04	1.04	1.04	1.04	1.04	1.04	1.04	1.04	1.04
		IT_{out}	27	27	27	27	27	27	27	27	28
		IT	28.04	28.04	28.04	28.04	28.04	28.04	28.04	28.04	29.04
		CPU	0.046	0.048	0.048	0.165	0.165	0.165	1.162	1.159	1.157
	P-TSCSP	IT_{int}	1.25	1.25	1.25	1.17	1.17	1.17	1.11	1.11	1.11
		IT_{out}	5	5	5	7	7	7	10	10	9
		IT	6.25	6.25	6.25	8.17	8.17	8.17	11.11	11.11	10.11
		CPU	0.025	0.025	0.025	0.064	0.065	0.067	0.402	0.397	0.393
	P-TTSCSP	IT_{int}	1.50	1.50	1.50	1.33	1.33	1.33	1.20	1.20	1.25
		IT_{out}	3	3	3	4	4	4	6	6	5
		IT	4.50	4.50	4.50	5.33	5.33	5.33	7.20	7.20	6.25
		CPU	0.024	0.023	0.024	0.056	0.055	0.056	0.330	0.330	0.324
0.001	P-MHSS	IT_{int}	1.02	1.02	1.03	1.02	1.02	1.02	1.02	1.02	1.02
		IT_{out}	43	43	39	63	62	60	92	92	90
		IT	44.02	44.02	40.03	64.02	63.02	61.02	93.02	93.02	91.02
		CPU	0.074	0.074	0.069	0.335	0.326	0.318	2.014	1.996	1.942
	P-LPMHSS	IT_{int}	1.05	1.05	1.05	1.05	1.05	1.05	1.05	1.05	1.05
		IT_{out}	21	21	22	21	21	21	21	21	21
		IT	22.05	22.05	23.05	22.05	22.05	22.05	22.05	22.05	22.05
		CPU	0.046	0.047	0.047	0.168	0.161	0.158	1.145	1.152	1.169
	P-TSCSP	IT_{int}	1.25	1.25	1.25	1.20	1.20	1.20	1.17	1.17	1.17
		IT_{out}	5	5	5	6	6	6	7	7	7
		IT	6.25	6.25	6.25	7.20	7.20	7.20	8.17	8.17	8.17
		CPU	0.025	0.025	0.026	0.072	0.070	0.072	0.397	0.395	0.396
	P-TTSCSP	IT_{int}	1.50	1.50	1.50	1.33	1.33	1.33	1.25	1.25	1.25
		IT_{out}	3	3	3	4	4	4	5	5	5
		IT	4.50	4.50	4.50	5.33	5.33	5.33	6.25	6.25	6.25
		CPU	0.023	0.023	0.024	0.055	0.055	0.056	0.327	0.325	0.336

Table 8: Numerical results for various inexact Picard iteration methods.

			$N=32$			$N=64$			$N=128$		
η	Iteration	ϱ	0.1	1	10	0.1	1	10	0.1	1	10
0.1	P-IMHSS	IT_{int}	1.01	1.01	1.01	1.01	1.01	1.01	1.00	1.00	1.00
		IT_{out}	86	86	81	138	137	133	214	214	208
		IT	87.01	87.01	82.01	139.01	138.01	134.01	215.00	215.00	209.00
		CPU	0.218	0.219	0.205	1.242	1.226	1.160	5.663	5.611	5.337
	P-ILPMHSS	IT_{int}	1.03	1.03	1.03	1.03	1.03	1.03	1.03	1.03	1.03
		IT_{out}	34	34	34	34	34	34	34	34	34
		IT	35.03	35.03	35.03	35.03	35.03	35.03	35.03	35.03	35.03
		CPU	0.274	0.281	0.263	1.192	1.200	1.350	3.976	4.261	4.219
	P-ITSCSP	IT_{int}	1.20	1.20	1.20	1.14	1.14	1.13	1.10	1.10	1.10
		IT_{out}	6	6	6	8	8	9	11	11	11
		IT	7.20	7.20	7.20	9.14	9.14	10.13	12.10	12.10	12.10
		CPU	0.064	0.067	0.070	0.430	0.421	0.472	2.089	2.161	2.115
	P-ITTSCSP	IT_{int}	1.33	1.33	1.33	1.25	1.25	1.25	1.20	1.20	1.17
		IT_{out}	4	4	4	5	5	5	6	6	8
		IT	5.33	5.33	5.33	6.25	6.25	6.25	7.20	7.20	8.17
		CPU	0.055	0.055	0.055	0.213	0.209	0.201	0.781	0.771	0.824
0.01	P-IMHSS	IT_{int}	1.01	1.01	1.02	1.01	1.01	1.01	1.01	1.01	1.01
		IT_{out}	70	70	65	109	109	107	168	167	163
		IT	71.14	71.01	66.02	110.01	110.01	108.01	169.01	168.01	164.01
		CPU	0.213	0.214	0.200	1.203	1.179	1.134	5.224	5.189	5.113
	P-ILPMHSS	IT_{int}	1.04	1.04	1.04	1.04	1.04	1.04	1.04	1.04	1.04
		IT_{out}	27	27	27	27	27	27	27	27	27
		IT	28.04	28.04	28.04	28.04	28.04	28.04	28.04	28.04	28.04
		CPU	0.248	0.263	0.241	1.081	1.231	1.078	4.066	4.275	4.235
	P-ITSCSP	IT_{int}	1.25	1.25	1.20	1.17	1.17	1.17	1.13	1.13	1.13
		IT_{out}	5	5	6	7	7	7	9	9	9
		IT	6.25	6.25	7.20	8.17	8.17	8.17	10.13	10.13	10.13
		CPU	0.067	0.069	0.070	0.430	0.424	0.414	2.315	2.308	2.161
	P-ITTSCSP	IT_{int}	1.33	1.33	1.33	1.33	1.33	1.33	1.20	1.20	1.25
		IT_{out}	4	4	4	4	4	4	6	6	5
		IT	5.33	5.33	5.33	5.33	5.33	5.33	7.20	7.20	6.25
		CPU	0.059	0.062	0.059	0.207	0.201	0.196	0.935	0.905	0.956
0.001	P-IMHSS	IT_{int}	1.02	1.02	1.02	1.01	1.01	1.01	1.01	1.01	1.01
		IT_{out}	51	50	48	81	81	77	127	127	127
		IT	52.02	51.02	49.02	82.01	82.01	78.01	128.01	128.01	128.01
		CPU	0.230	0.217	0.219	1.238	1.223	1.157	5.428	5.437	5.192
	P-ILPMHSS	IT_{int}	1.05	1.05	1.05	1.05	1.05	1.05	1.05	1.05	1.05
		IT_{out}	20	20	20	20	20	20	21	21	21
		IT	21.05	21.05	21.05	21.05	21.05	21.05	21.05	21.05	21.05
		CPU	0.287	0.248	0.247	1.326	1.404	1.147	4.162	3.922	4.179
	P-ITSCSP	IT_{int}	1.25	1.25	1.25	1.20	1.20	1.20	1.17	1.17	1.14
		IT_{out}	5	5	5	6	6	6	7	7	8
		IT	6.25	6.25	6.25	7.20	7.20	7.20	8.17	8.17	8.14
		CPU	0.075	0.078	0.077	0.491	0.496	0.477	2.718	2.696	2.967
	P-ITTSCSP	IT_{int}	1.33	1.33	1.33	1.33	1.33	1.33	1.25	1.25	1.25
		IT_{out}	4	4	4	4	4	4	5	5	5
		IT	5.33	5.33	5.33	5.33	5.33	5.33	6.25	6.25	6.25
		CPU	0.059	0.059	0.058	0.233	0.223	0.226	0.995	0.970	0.999

Table 9: Numerical results for Picard-TTSCSP iteration method, α_{opt}, β_{opt} and α^*, β^*.

	$N=32$			$N=64$			$N=128$		
ϱ	0.1	1	10	0.1	1	10	0.1	1	10
α_{opt}	1.17	1.17	1.17	1.17	1.17	1.17	1.17	1.17	1.17
β_{opt}	0.31	0.31	0.30	0.30	0.30	0.30	0.20	0.20	0.20
IT_{int}	1.33	1.33	1.33	1.25	1.25	1.25	1.25	1.25	1.25
IT_{out}	4	4	4	5	5	5	5	5	5
IT	5.33	5.33	5.33	6.25	6.25	6.25	6.25	6.25	6.25
CPU	0.024	0.024	0.024	0.057	0.057	0.056	0.327	0.331	0.329
α^*	1.56	1.57	1.65	1.80	1.81	1.85	2.03	2.03	2.05
β^*	0.61	0.64	0.60	0.56	0.56	0.54	0.49	0.49	0.49
IT_{int}	1.25	1.25	1.25	1.20	1.20	1.20	1.17	1.17	1.20
IT_{out}	5	5	5	6	6	6	7	7	6
IT	6.25	6.25	6.25	7.20	7.20	7.20	8.17	8.17	7.20
CPU	0.027	0.027	0.027	0.071	0.071	0.073	0.338	0.338	0.331

Table 10: Numerical results for MHSS-like, LPMHSS-like, TSCSP-like, TTSCSP-like iteration methods.

	Iteration	ϱ	0.1	1	10
$N=32$	MHSS-like	α_{opt}	0.34	0.34	0.35
		IT	82	81	78
		CPU	0.0854	0.0863	0.0859
	LPMHSS-like	α_{opt}	3.0	3.0	3.0
		IT	35	35	35
		CPU	0.0589	0.0543	0.0530
	TSCSP-like	α_{opt}	0.43	0.43	0.43
		IT	7	7	7
		CPU	0.0274	0.0282	0.0286
	TTSCSP-like	α_{opt}	1.06	1.06	1.06
		β_{opt}	0.36	0.36	0.36
		IT	4	4	4
		CPU	0.0241	0.0255	0.0236
$N=64$	MHSS-like	α_{opt}	0.2	0.2	0.2
		IT	125	124	120
		CPU	0.3611	0.3574	0.3498
	LPMHSS-like	α_{opt}	2.7	2.7	2.7
		IT	35	35	35
		CPU	0.1477	0.1500	0.1480
	TSCSP-like	α_{opt}	0.35	0.35	0.35
		IT	9	9	10
		CPU	0.0726	0.0743	0.0734
	TTSCSP-like	α_{opt}	0.94	0.94	0.94
		β_{opt}	0.29	0.29	0.29
		IT	5	5	5
		CPU	0.0596	0.0594	0.0614

$N=128$	MHSS-like	α_{opt}	0.12	0.12	0.12
		IT	181	181	178
		CPU	2.1688	2.1335	2.1049
	LPMHSS-like	α_{opt}	2.35	2.35	2.35
		IT	35	35	35
		CPU	0.7675	0.7805	0.7681
	TSCSP-like	α_{opt}	0.24	0.24	0.24
		IT	13	13	13
		CPU	0.3731	0.3748	0.3796
	TTSCSP-like	α_{opt}	0.82	0.82	0.82
		β_{opt}	0.22	0.22	0.22
		IT	6	6	6
		CPU	0.3138	0.3094	0.3136

Table 11: Numerical results for inexact MHSS-like, LPMHSS-like, TSCSP-like, TTSCSP-like iteration methods.

	Iteration	ϱ	0.1	1	10
$N=32$	IMHSS-like	α_{opt}	0.34	0.34	0.35
		IT	82	81	78
		CPU	0.4466	0.4339	0.4094
	ILPMHSS-like	α_{opt}	3.0	3.0	3.0
		IT	35	35	35
		CPU	0.2342	0.2326	0.2252
	ITSCSP-like	α_{opt}	0.43	0.43	0.43
		IT	7	7	7
		CPU	0.0702	0.0704	0.0702
	ITTSCSP-like	α_{opt}	1.06	1.06	1.06
		β_{opt}	0.36	0.36	0.36
		IT	4	4	4
		CPU	0.0520	0.0505	0.0492
$N=64$	IMHSS-like	α_{opt}	0.20	0.20	0.20
		IT	125	124	120
		CPU	2.8561	2.8323	2.7401
	ILPMHSS-like	α_{opt}	2.7	2.7	2.7
		IT	35	35	35
		CPU	1.0628	1.0487	1.0167
	ITSCSP-like	α_{opt}	0.35	0.35	0.35
		IT	9	9	10
		CPU	0.3495	0.3405	0.3473
	ITTSCSP-like	α_{opt}	0.94	0.94	0.94
		β_{opt}	0.29	0.29	0.29
		IT	5	5	5
		CPU	0.1953	0.1930	0.1845

$N = 128$	IMHSS-like	α_{opt}	0.12	0.12	0.12
		IT	181	181	178
		CPU	11.7168	11.4980	11.5725
	ILPMHSS-like	α_{opt}	2.37	2.37	2.37
		IT	35	35	35
		CPU	4.7721	4.7076	4.5706
	ITSCSP-like	α_{opt}	0.24	0.24	0.24
		IT	13	13	13
		CPU	2.2445	2.2148	2.1373
	ITTSCSP-like	α_{opt}	0.82	0.82	0.82
		β_{opt}	0.22	0.22	0.22
		IT	6	6	6
		CPU	0.9811	0.9842	0.9589

7. Conclusion

We propose Picard-TSCSP, Picard-TTSCSP, nonlinear TSCSP-like and TTSCSP-like iteration methods and their inexact versions for a class of large sparse systems of weakly nonlinear equations. Numerical experiments show that new iteration methods are efficient, implementable and outperform well-known Picard-MHSS, Picard-LPMHSS, Picard-TSCSP and Picard-TTSCSP iteration methods.

Acknowledgements

The authors would like to thank anonymous referees for their comments and suggestions.

The work of Davod Khojasteh Salkuyeh was partially supported by the University of Guilan.

References

[1] H.-B. An and Z.-Z. Bai, *NGLM: A globally convergent Newton-GMRES method (in Chinese)*, Math. Numer. Sinica **27**, 151–174 (2005).

[2] H.-B. An and Z.-Z. Bai, *A globally convergent Newton-GMRES method for large sparse systems of nonlinear equations*, Appl. Numer. Math. **57**, 235–252 (2007).

[3] Z.-Z. Bai, *A class of two-stage iterative methods for systems of weakly nonlinear equations*, Numer. Algorithms **14**, 295–319 (1997).

[4] Z.-Z. Bai, *On the convergence of parallel chaotic nonlinear multisplitting Newton-type methods*, J. Comput. Appl. Math. **80**, 317–334 (1997).

[5] Z.-Z. Bai, *Parallel multisplitting two-stage iterative methods for large sparse systems of weakly nonlinear equations*, Numer. Algorithms **15**, 347–372 (1997).

[6] Z.-Z. Bai, *Optimal parameters in the HSS-like methods for saddle-point problems*, Numer. Linear Algebra Appl. **16** 447–479 (2009).

[7] Z.-Z. Bai and H.-B. An, *On efficient variants and global convergence of the Newton-GMRES method (in Chinese)*, J. Numer. Meth. Comput. Appl. **26**, 291–300 (2005).

[8] Z.-Z. Bai, M. Benzi and F. Chen, *Modified HSS iteration methods for a class of complex symmetric linear systems*, Computing **87**, 93–111 (2010).

[9] Z.-Z. Bai, G.H. Golub and M.K. Ng, *Hermitian and skew-Hermitian splitting methods for non-Hermitian positive definite linear systems*, SIAM J. Matrix Anal. Appl. **24** 603–626 (2003).

[10] Z.-Z. Bai and X.-P. Guo, *On Newton-HSS methods for systems of nonlinear equations with positive-definite Jacobian matrices*, J. Comput. Math. **28**, 235–260 (2010).

[11] Z-.Z. Bai and Y.-G. Huang, *Asynchronous multisplitting two-stage iterations for systems of weakly nonlinear equations*, J. Comput. Appl. Math. **93**, 13–33 (1998).

[12] Z.-Z. Bai, Y.-M. Huang and M.K. Ng, *On preconditioned iterative methods for Burgers equations*, SIAM J. Sci. Comput. **29**, 415–439 (2007).

[13] Z.-Z. Bai, Y.-M. Huang and M.K. Ng, *On preconditioned iterative methods for certain time-dependent partial differential equations*, SIAM J. Numer. Anal. **47** 1019–1037 (2009).

[14] Z.-Z. Bai and X. Yang, *On HSS-based iteration methods for weakly nonlinear systems*, Appl. Numer. Math. **59** 2923–2936 (2009).

[15] M. Benzi and M.K. Ng, *Preconditioned iterative methods for weighted toeplitz least squares problems*, SIAM J. Matrix Anal. Appl. **27**, 1106–1124 (2006).

[16] R.S. Dembo, S.C. Eisenstat and T. Steihaug, *Inexact Newton methods*, SIAM J. Numer. Anal. **19** 400–408 (1982).

[17] E. Galligani, *The newton-arithmetic mean method for the solution of systems of nonlinear equations*, Appl. Math. Comput. **134**, 9–34 (2003).

[18] X.-P Guo, *On semilocal convergence of inexact Newton methods*, J. Comput. Math. **25**, 231–242 (2007).

[19] D. Hezari, D.K. Salkuyeh and V. Edalatpour, *A new iterative method for solving a class of complex symmetric system of linear equathions*, Numer. Algorithms **73**, 927–955 (2016).

[20] C.T. Kelley, *Iterative methods for linear and nonlinear equations*, SIAM (1995).

[21] C.-X. Li and S.-L. Wu, *On LPMHSS-based iteration methods for a class of weakly nonlinear systems*, Comp. Appl. Math. **37**, 1232–1249 (2018).

[22] X. Li, A.-L. Yang and Y.-J. Wu, *Lopsided PMHSS iteration method for a class of complex symmetric linear systems*, Numer. Algorithms **66**, 555–568 (2014).

[23] L.D. Liao and G.F. Zhang, *Efficient preconditioner and iterative method for large complex symmetric linear algebraic systems*, East Asian J. Appl. Math. **7**, 530–547 (2017).

[24] J.M. Ortega and W.C. Rheinboldt, *Iterative solution of nonlinear equations in several variables*, Academic Press (1970).

[25] D.K. Salkuyeh, *Two-step scale-splitting method for solving complex symmetric system of linear equations*, arXiv:1705.02468.

[26] D.K. Salkuyeh and T.S. Siahkolaei, *Two-parameter TSCSP method for solving complex symmetric system of linear equations*, Calcolo **55**, 8 (2018).

[27] A.H. Sherman, *On Newton-iterative methods for the solution of systems of nonlinear equations*, SIAM J. Numer. Anal. **15**, 755–771 (1978).

[28] T. Tang, *Superconvergence of numerical solutions to weakly singular volterra integro-differential equations*, Numer. Math. **61**, 373–382 (1992).

[29] J. Wang, X.-P. Guo and H.-X. Zhong, *Accelerated GPMHSS method for solving complex systems of linear equations*, East Asian J. Appl. Math. **7**, 143–155 (2017).

[30] Y.-M. Wang, *Parallel multisplitting methods for a class of systems of weakly nonlinear equations without isotone mapping*, Appl. Math. Comput. **109**, 135–150 (2000).

[31] W. Yang, Y.-J. Wu and J. Fu, *On Picard-MHSS methods for weakly nonlinear systems (in Chinese)*,

Math. Numer. Sinica **36**, 291–302 (2014).

[32] M.-Z. Zhu and G.-F. Zhang, *On CSCS-based iteration methods for Toeplitz system of weakly nonlinear equations*, J. Comput. Appl. Math. **235**, 5095–5104 (2011).