Preprints of the 21st IFAC World Congress (Virtual)
Berlin, Germany, July 12-17, 2020

Efficient Iterative Solvers in the Least Squares Method

Alexander Stotsky *

* *Department of Computer Science and Engineering*
Chalmers University and University of Gothenburg, SE-412 96
Gothenburg, Sweden
(e-mail: alexander.stotsky@chalmers.se)

Abstract Fast convergent, accurate, computationally efficient, parallelizable, and robust matrix inversion and parameter estimation algorithms are required in many time-critical and accuracy-critical applications such as system identification, signal and image processing, network and big data analysis, machine learning and in many others.
This paper introduces new composite power series expansion with optionally chosen rates (which can be calculated simultaneously on parallel units with different computational capacities) for further convergence rate improvement of high order Newton-Schulz iteration. New expansion was integrated into the Richardson iteration and resulted in significant convergence rate improvement. The improvement is quantified via explicit transient models for estimation errors and by simulations. In addition, the recursive and computationally efficient version of the combination of Richardson iteration and Newton-Schulz iteration with composite expansion is developed for simultaneous calculations.
Moreover, unified factorization is developed in this paper in the form of tool-kit for power series expansion, which results in a new family of computationally efficient Newton-Schulz algorithms.

Keywords: Least Squares Estimation,Power Series Factorization Tool-Kit,Computationally Efficient High Order Newton-Schulz Algorithm,Simultaneous Calculations,Convergence Acceleration of Richardson Iteration

1. INTRODUCTION

Least squares method is widely used in control, system identification, signal processing, statistics as well as in many computational applications such as emerging big data applications, machine learning and in many other areas. For accurate solution many least squares problems can be associated with calculation of the parameter vector θ_*, which satisfies the algebraic equation, Ljung (1999):

$$A\theta_* = b \tag{1}$$

where b is the vector, and A is SPD (Symmetric and Positive Definite) matrix. For example, the matrix A is SPD for the systems with harmonic regressor, Fomin et al. (1981), Bayard (2000), Stotsky (2010) and multiplication of any invertible matrix A by its transpose transforms the system to the SPD case with the Gram matrix, Björck (1996). The numerical stability problems associated with ill-conditioning of the Gram matrix can be solved using different types of preconditioning techniques, see for example, Chen (2005) and Stotsky (2015),(2017) (see also Section 7 for simulations of the ill-conditioned matrices).
Iterative methods for solving (1) are often preferable (especially for large-scale systems) due to simplicity, better accuracy and robustness, less processor time and memory space compared to direct methods. The most general and well-known method for iterative calculation of the matrix inverse is high order Newton-Schulz algorithm, described by Isaacson et al. (1966), Petryshyn (1967), Stickel (1987), Pan et al. (2016) and by many others. The second order version of Newton-Schulz iteration, see for example Schulz (1933), Demidovich et al. (1962) and Söderström et al. (1974) is the most known.
High order Newton-Schulz algorithms are well-discussed in the literature. However, the questions associated with the relation between high order Newton-Schulz algorithms and power series expansions were not properly studied. The paper by Janiszowski (2003), which was the first paper with the description of the relation between second order Newton-Schulz algorithm and power series expansion does not provide the complete solution.
Reduction of the computational complexity of high order Newton-Schulz algorithm is one of the most important challenges in this area. The computational complexity can be reduced via factorizations of the power series, see for example Stickel (1987), Soleimani et al. (2015), Pan et al. (2016), Buranay et al. (2019) and many others. Practical applications of these factorizations (excepting Horner's rule) are hampered by the lack of unified description.
Computational resources with high degree of parallelism (instead of single computing units) will be available in the future for implementation of numerical methods. The computational performance of iterative solvers can also be improved via parallel computing (especially for large scale systems), achieved for example via multiprocessor systems, Saad (2003), Peng et al. (2013). In order to improve the performance a serial algorithm is usually converted to parallel algorithm. This paper proposes a new approach for convergence rate improvement where *novel iterative*

algorithms are designed with high degree of parallelism (or enhanced parallelism). In other words, the iterative algorithm is designed as a number of independent computational parts (the number of parts is associated with the degree of parallelism) which can be executed simultaneously. The challenges associated with computational efficiency are addressed already on the design level in this case, providing new opportunities for high performance parallel processing.

This paper introduces new composite power series expansion with optionally chosen rates and high degree of parallelism for further convergence rate improvement in the unified framework described by Stotsky (2019). New expansion applied to Richardson iteration resulted in significant improvement of the convergence rate. Simulation results are presented for quantification of the improvements of new algorithms compared to recent algorithms described by Stotsky (2019). Moreover, explicit transient models are derived for all new algorithms described in this paper. In addition, the recursive and computationally efficient version of the combination of Richardson iteration and Newton-Schulz iteration with composite expansion is developed for simultaneous calculations.

Finally, factorization tool-kit is developed in this paper for general power series expansion, which allows nested applications and results in a family of new computationally efficient algorithms.

This paper is organized as follows. The paper starts with redesign of Newton-Schulz iteration in the form of the power series expansion in Section 3. Unified power series factorization in the form of tool-kit for reduction of computational complexity is presented in Section 4. New high order Newton-Schulz algorithms with composite polynomial is presented in Section 5. Richardson iteration with high order convergence accelerator is described in Section 6 and compared to existing algorithms by simulation in Section 7. The paper ends with brief conclusions in Section 8.

2. SPLITTING & PRECONDITIONING

Numerical solution of the system of linear equations (1) using power series expansions requires splitting and preconditioning. Any positive definite and symmetric matrix A, whose inverse should be calculated can be split as follows, see for example Chen (2005) and references therein

$$A = S - D \tag{2}$$
$$I - S^{-1}A = S^{-1}D \tag{3}$$
$$\rho(I - S^{-1}A) = \rho(S^{-1}D) < 1 \tag{4}$$

where the spectral radius $\rho(\cdot)$ defined in (4) is less than one for symmetric and positive definite matrices A and S (where S^{-1} is the preconditioner), provided that $2S - A$ is a positive definite matrix, Hackbusch (1994).

For example, the matrix S can be chosen as a diagonal matrix, which contains the diagonal elements of SDD (Strictly Diagonally Dominant) and positive definite matrix A, Horn et al. (1985) and Stotsky (2010).

For positive definite (not SDD) matrix A the simplest preconditioner can be chosen as $S^{-1} = I/\alpha$ with $\alpha = \|A\|_\infty/2+\varepsilon$, where $\|\cdot\|_\infty$ is the maximum row sum matrix norm, and $\varepsilon > 0$, Stotsky (2015).

Notice that the spectral radius (4) is very close to one for ill-conditioned matrix A, which implies the stability problem. The problem can be solved via application of the stepwise splitting method, Stotsky (2017).

3. NEWTON-SCHULZ ITERATION AS FAST POWER SERIES EXPANSION

The results presented in this Section introduce computationally efficient factorization of the initial power series and show several steps (step by step) of fast matrix power series expansion which coincide with Newton-Schulz iteration. The relation between Newton-Schulz approach and power series expansion opens new opportunities for reduction of the computational complexity of Newton-Schulz algorithms.

The following initial power series factorization :

$$G_0 = \sum_{j=0}^{w-1}(S^{-1}D)^{(p+1)j} \ \{\sum_{d=0}^{p}(S^{-1}D)^d\}S^{-1} \tag{5}$$
$$= \sum_{j=0}^{h-1}(S^{-1}D)^j S^{-1} = (I - (S^{-1}D)^h)A^{-1} \tag{6}$$
$$F_0 = I - G_0A = (S^{-1}D)^h \tag{7}$$

where (7) defines initial inversion error, and $p = 0, 1, 2, ...$, $w = 1, 2, 3, ...$, and $h = w(p + 1) = 1, 2, 3...$, gives the starting point for the following steps of Newton-Schulz iteration. <u>Step 1:</u>

$$G_1 = \{\sum_{j=0}^{n-1} F_0^j\} \ G_0 = \{\sum_{j=0}^{n-1}(S^{-1}D)^{hj}\} \ G_0$$
$$[I + (S^{-1}D)^h + (S^{-1}D)^{2h} + ... + (S^{-1}D)^{(n-1)h}]$$
$$[I + (S^{-1}D) + (S^{-1}D)^2 + ... + (S^{-1}D)^{h-1}]S^{-1}$$
$$= [I + (S^{-1}D) + ... + (S^{-1}D)^{(hn-1)}]S^{-1} \tag{8}$$
$$F_1 = I - G_1A = (S^{-1}D)^{hn} \tag{9}$$

where G_1 in (9) is calculated via (8). <u>Step 2:</u>

$$G_2 = \{\sum_{j=0}^{n-1} F_1^j\} \ G_1 = \{\sum_{j=0}^{n-1}(S^{-1}D)^{hnj}\} \ G_1$$
$$F_2 = I - G_2A = (S^{-1}D)^{hn^2} \tag{10}$$

Further evaluation in <u>Step k</u> gives classical high order Newton-Schulz algorithm (11) and error model (12) :

$$G_k = \{\sum_{j=0}^{n-1} F_{k-1}^j\} \ G_{k-1} \tag{11}$$
$$F_k = I - G_kA = F_0^{n^k} = (S^{-1}D)^{hn^k} \tag{12}$$

where G_k is the estimate of A^{-1}, $n = 2, 3, ...$ and $k = 1, 2, 3, ...$

Notice that the factorization similar to (5) can be applied to the power series (11) for improvement of computational efficiency. To this end the unified factorization method is developed in the next Section.

4. FACTORIZATION TOOL-KIT: A UNIFIED APPROACH

The following unified framework is presented for factorization of the Newton-Schulz iteration :

$$Z = \{\sum_{j=0}^{n-1} Y^j\}\ X \tag{13}$$

$$Y = I - XA \tag{14}$$

$$Z = \{\sum_{j=0}^{w-1} Y^{(p+1)j}\}\ \{\sum_{d=0}^{p} Y^d\}\ X \tag{15}$$

$$Y^{p+1} = I - \{\sum_{d=0}^{p} Y^d\}\ X\ A \tag{16}$$

$$h = w\ (p+1),\quad p = 0, 1, 2, ...,\quad w = 1, 2, 3, ... \tag{17}$$

$$Z_1 = (I + Y\{\sum_{j=0}^{w-1} Y^{(p+1)j}\}\ \{\sum_{d=0}^{p} Y^d\})\ X \tag{18}$$

$$h_1 = h + 1 \tag{19}$$

where the equations (13),(14) represent the Newton-Schulz iteration of the order n, and the equations (15), (18) represent factorizations Z and Z_1 of orders h and h_1 respectively.
The technique is illustrated on the example (described by Buranay et al., 2019) of the step-wise factorization of the algorithm of order 45 that requires 10 mmm (matrix-by-matrix multiplications) only:

$$Z = (I + Y^9 + ... + Y^{36})\ \ [I + Y + ... + Y^8]\ X \tag{20}$$

$$= (I + Y^9 + ... + (Y^9)^4)\ \ [I + Y^3 + Y^6]\ [I + Y + Y^2]\ X \tag{21}$$

$$= \{I + (I + (Y^9)^2)\ (Y^9 + (Y^9)^2)\}\ \ [I + Y^3 + Y^6]$$
$$[3I + (XA)\ (-3I + XA)]\ X \tag{22}$$

$$Y^3 = I - [\sum_{d=0}^{2} Y^d]\ X\ A,\ Y^9 = I - [\sum_{d=0}^{8} Y^d]\ X\ A$$

The factorization (15) is valid for the orders, which are presented as the composite numbers[1], $h = 2, 4, 6, ...$ (excepting $h = 2$). For the orders which represent the prime numbers[2], $h_1 = 3, 5, 7, ...$ the factorization (18) is valid. Factorizations, based on the unified framework described above for the orders $n = 2, ..., 19$ are presented in Table 1.
Notice that the factorization (18) is simple application of the idea known as Schröder–Traub sequence (Traub, 1964) to the polynomial factorized in (15). The factorization (18) increases the order of (15) by one, see (17) and (19).
Notice that the factorizations which require computational efforts and additional memory may result in error accumulation in finite-digit calculations. The factorizations in high order Newton-Schulz iterations can be seen as the additional sub-steps (nested calculations for order reduction). The idea of order reduction is also associated with the Newton-Schulz iteration. Therefore Newton-Schulz algorithms of low orders ($h = 2, 3$) being iterated for a number

[1] A composite number is a positive integer that has at least one divisor other than one and itself
[2] A prime number is a positive integer that has exactly two distinct divisors, namely one and the number itself

Order	Unified Factorization	Reference
$h = 2$	$Z = (I + Y)\ X$ $p = 1,\ w = 1$	Schulz (1933)
$h_1 = 3$	$Z_1 = (I + Y(I + Y))\ X$ $p = 1,\ w = 1$	Pan et al. (2016)
$h = 4$	$Z = (I + Y^2)(I + Y)\ X$ $p = 1,\ w = 2$	Esmaeilic et al. (2017)
$h_1 = 5$	$Z_1 = (I + Y(I + Y^2)(I + Y))\ X$ $p = 1,\ w = 2$	Soleimani et al. (2015)
$h = 6$	$Z = (I + Y^3)(I + Y + Y^2)\ X$ $p = 2,\ w = 2$	
$h_1 = 7$	$Z_1 = (I + (Y + Y^4)(I + Y + Y^2))\ X$ $p = 2,\ w = 2$	Soleimani (2015)
$h = 8$	$Z = (I + Y^4)(I + Y^2)(I + Y)X$ $p = 3,\ w = 2$	
$h = 9$	$Z = (I + Y^3 + Y^6)(I + Y + Y^2)\ X$ $p = 2,\ w = 3$	
$h = 10$	$Z = (I{+}Y^5)(I{+}(Y{+}Y^2)(I{+}Y^2))\ X$ $p = 4,\ w = 2$	Soleimani et al. (2015)
$h_1 = 11$	$Z_1 = (I + Y(I + (Y^2 + Y^4)(I + Y^4))$ $(I + Y))\ X$ $p = 1,\ w = 5$	Stanimirovic et al. (2019)
$h = 12$	$Z = (I + Y^4 + Y^8)$ $(I + Y + Y^2 + Y^3)\ X$ $p = 3,\ w = 3$	Soleimani et al. (2015)
$h_1 = 13$	$Z_1 = (I + Y(I + Y^4 + Y^8)$ $(I + Y + Y^2 + Y^3))\ X$ $p = 3,\ w = 3$	Soleimani et al. (2015)
$h = 14$	$Z = (I + Y^7)$ $(I{+}Y{+}Y^2{+}Y^3{+}Y^4{+}Y^5{+}Y^6)\ X$ $p = 6,\ w = 2$	Soleimani et al. (2015)
$h = 15$	$Z = (I + (I + (Y^3)^2)\ ((Y^3)^2 + Y^3))$ $(I + Y + Y^2)\ X$ $p = 2,\ w = 5$	Soleimani et al. (2015)
$h = 16$	$Z = (I + Y^4 + Y^8 + Y^{12})$ $(I + Y + Y^2 + Y^3)\ X$ $p = 3,\ w = 4$	Soleimani et al. (2015)
$h_1 = 17$	$Z_1 = (I + (Y + Y^2 + Y^3 + Y^4)$ $(I + Y^4 + Y^8 + Y^{12}))\ X$ $p = 3,\ w = 4$	Soleimani et al. (2015)
$h = 18$	$Z = (I + Y^6 + Y^{12})$ $(I + Y + Y^2 + Y^3 + Y^4 + Y^5)\ X$ $p = 5,\ w = 3$	Soleimani et al. (2015)
$h_1 = 19$	$Z_1 = (I + (Y + Y^2)(I + Y^2 + Y^4)$ $(I + Y^6 + Y^{12}))\ X$ $p = 5,\ w = 3$	Soleimani et al. (2015)

Table 1. Factorization of the algorithms of orders $n = 2, ..., 19$

of steps can be applied instead of factorized Newton-Schulz iterations of higher orders for the sake of robustness and efficiency. Indeed, two and three steps of the second order Newton-Schulz iteration are equivalent to one step of the fourth and eighth order iterations with 4 and 6 mmm respectively, see Table 1, $h = 4$ and $h = 8$. Robustness, efficiency and accuracy arguments motivate application of the second order and the third order Newton-Schulz iterations instead of higher orders in some cases. However, application of the eleventh order algorithm, see Table 1 requires also 6 mmm and provides faster convergence than three steps of the second order Newton-Schulz iteration. Therefore the proper choice of the order and factorization that is made for each particular application should represent the trade-off between the robustness and convergence rate.

5. NOVEL NEWTON-SCHULZ ALGORITHM

The following new composite power series expansion for Newton-Schulz iteration with different expansion rates for convergence rate improvement extends the framework described by Stotsky (2019) as follows :

$$G_k = \underbrace{T_c}_{\substack{\text{Composite}\\\text{Polynomial}}} + \underbrace{\Gamma_c}_{\substack{\text{Composite}\\\text{Residual}}} \underbrace{\{\sum_{j=0}^{n-1} F_{k-1}^j\}\ G_{k-1}}_{\substack{\text{Newton-Schulz}\\\text{Iteration}}} \tag{23}$$

$$T_c = \sum_{i=1}^{w}\{\prod_{p=0}^{i-1}\Gamma_p\}T_i = T_1 + \Gamma_1 T_2 + ... + \prod_{p=1}^{w-1}\Gamma_p T_w \tag{24}$$

$$\Gamma_c = \prod_{p=1}^{w}\Gamma_p \tag{25}$$

where T_c is the composite power series expansion and the composite residual is defined as the product of the residual terms $\Gamma_c = \Gamma_1\Gamma_2\Gamma_3...\Gamma_w$ with the spectral radius $\rho(\Gamma_i) < 1$, and $\Gamma_0 = I$. Power series expansions T_i satisfy the following relations:

$$T_i A = I - \Gamma_i\ , \quad i = 1, ..., w \tag{26}$$

$$T_c A = I - \Gamma_c \tag{27}$$

Multiplication of both sides of equation (23) by A together with the relation (27) results in following error model:

$$F_k = \Gamma_c F_{k-1}^n \tag{28}$$

where the spectral radius $\rho(\Gamma_c) \le \rho(\Gamma_1)...\rho(\Gamma_w) < 1$ according to Gelfand's formula provided that the matrices Γ_i commute.

The power series expansions T_i (which can be calculated simultaneously on parallel computational units) can be taken as follows:

$$T_i = \{\sum_{j=0}^{x_i-1}(S^{-1}D)^j\}S^{-1} = (I - \Gamma_i)A^{-1} \tag{29}$$

$$\Gamma_i = (S^{-1}D)^{x_i} \tag{30}$$

The rate of expansion x_i can be chosen using computational capacity of each parallel computational unit (fast power series expansion should be implemented on more powerful computational unit). For example x_i can be taken as a polynomial which is a function of step number k or as rapidly expanding power series associated with high order Newton-Schulz iteration with $x_i = m^k$, $m = 2, 3, ...$, see for example Stotsky (2019) for this and other choices.

5.1 Newton-Schulz Algorithm with High Order Residual

Fast and computationally efficient algorithms can be designed for a number of the same rapid expansions T_i (high order Newton-Schulz iterations) with the expansion rate associated with the order n in (23).

The algorithm (23) with $T_1 = T_2 = ... = T_k$, where T_k and Γ_k are defined in (29),(30) with $w = n$ and $x_i = hn^k$, $h, n = 1, 2, ...$ has the following form :

$$\Gamma_k = I - L_{k-1}A \tag{31}$$

$$L_k = \{\sum_{j=0}^{n-1}\Gamma_k^j\}\ L_{k-1} \tag{32}$$

$$\Gamma_k^n = I - L_k A \tag{33}$$

$$G_k = \underbrace{L_k}_{\substack{\text{Newton-Schulz}\\\text{Iteration}}} + \underbrace{\Gamma_k^n}_{\substack{\text{High Order}\\\text{Convergence}\\\text{Accelerator}}} \underbrace{\{\sum_{j=0}^{n-1} F_{k-1}^j\}\ G_{k-1}}_{\substack{\text{Newton-Schulz}\\\text{Iteration}}} \tag{34}$$

$$F_k = I - G_k A = \Gamma_k^n\ F_{k-1}^n \tag{35}$$

$$F_k = (S^{-1}D)^h\ (k\ n^{k+1} +\ n^k) \tag{36}$$

where $L_0 = \{\sum_{j=0}^{n-1}\Gamma_0^j\}\ T_0$, $\Gamma_0 = I - T_0 A$, and $T_0 = G_0$ (G_0 is calculated via (5)) are precalculated. The algorithm (31) - (36) has two Newton-Schulz loops (which can be calculated simultaneously) of the same order n associated with inversion errors (31) and (35). The sums $\sum_{j=0}^{n-1}\Gamma_k^j$ and $\sum_{j=0}^{n-1}F_{k-1}^j$ can be calculated recursively.

Remark 1. Comparison of the error model (36) with the error model (12) of classical high order Newton-Schulz algorithm shows that the algorithm (31) - (35) has significantly higher convergence rate due to the term $k\ n^{k+1}$.

Remark 2. The algorithm similar to (31) - (36) was proposed by Srivastava et al. (2014). The algorithm written in the following form:

$$Z_k = \sum_{j=0}^{p-1}(I - Z_{k-1}A)^j\ Z_{k-1} \tag{37}$$

$$G_k = G_{k-1} + (I - G_{k-1}A)\ Z_k \tag{38}$$

has also two Newton-Schulz loops, where both Z_k and G_k are the estimates of the matrix inverse and $p = 2, 3, ...$ is the order. Algorithm (37), (38) has the following error model :

$$L_k = L_{k-1}^p, \quad L_k = I - Z_k A \tag{39}$$

$$F_k = F_{k-1}L_{k-1}^p, \quad F_k = I - G_k A \tag{40}$$

where L_k and F_k are the estimation errors. The algorithm (31) - (36) has faster convergence due to the high order error F_{k-1}^n in the error model (35) compared to algorithm (37), (38) which has the error model (40) with the first order error F_{k-1}.

6. MODIFICATION OF THE RICHARDSON ITERATION

The parameter vector in (1) can be estimated using combination of the fast matrix inversion algorithm described above and Richardson iteration as follows:

$$\theta_k = \theta_{k-1} - \underbrace{[\ L_k + \Gamma_k^n \ \{\sum_{j=0}^{q-1} F_k^j\}\ G_k]}_{\text{Fast Matrix Inversion Algorithm}} \ \underbrace{\{A\theta_{k-1} - b\}}_{\text{Parameter Estimation Error}} \quad (41)$$

where θ_k is the estimate of θ_* and Γ_k, L_k, F_k and G_k are calculated in (31) - (35) and $q = 1, 2, ...$ is the order of Neumann series. The following model is valid for estimation error $\tilde{\theta}_k = \theta_k - \theta_*$:

$$\tilde{\theta}_k = \Gamma_k^n \ F_k^q \ \tilde{\theta}_{k-1} \quad (42)$$

$$\tilde{\theta}_k = (S^{-1}D)^{h\ \{(k\ n^{k+1} +\ n^k)\ q + n^{k+1}\}} \ \tilde{\theta}_{k-1} \quad (43)$$

The error model has especially simple form for the case where $q = n$:

$$\tilde{\theta}_k = \Gamma_k^n \ F_k^n \ \tilde{\theta}_{k-1} \quad (44)$$

$$\tilde{\theta}_k = (S^{-1}D)^{h\ (k\ n^{k+2} +\ 2\ n^{k+1})} \ \tilde{\theta}_{k-1} \quad (45)$$

$$\tilde{\theta}_k = (S^{-1}D)^{\gamma_k} \ \tilde{\theta}_0 \quad (46)$$

$$\gamma_k = h\ n^2 \ \frac{(k\ n^{k+2} - (k-1)\ n^{k+1} - 2\ n^k - n + 2)}{(n-1)^2} \quad (47)$$

where $n > 1$, $h = 1, 2, ...$, $\tilde{\theta}_0 = \theta_0 - \theta_*$ and $\theta_0 = L_0 b$.
For development of the computationally efficient version the algorithm (41) is presented in the following form:

$$\theta_k = \theta_{k-1} - \omega_k \ (A\theta_{k-1} - b) \quad (48)$$

$$\omega_k = L_k + \Gamma_k^n \ \{\sum_{j=0}^{n-1} F_k^j\} \ G_k \quad (49)$$

Recursive algorithm for calculation of ω_k described below is divided in two independent computational parts for simultaneous calculations. Calculations of both parts start with calculation of the $\sum_{j=0}^{n-1} \Gamma_k^j$, where $\Gamma_k = \Gamma_{k-1}^n$.
1) The first part is associated with the calculation of G_k in (34) via ω_{k-1} as follows:

$$G_k = [\sum_{j=0}^{n-1} \Gamma_k^j]\ [\omega_{k-1} - \sum_{j=0}^{n-1} F_{k-1}^j \ G_{k-1}] + \sum_{j=0}^{n-1} F_{k-1}^j \ G_{k-1} \quad (50)$$

which requires one matrix multiplication only and further calculation of $\sum_{j=0}^{n-1} F_k^j \ G_k$ with G_k defined in (50) which in turn can be further divided in independent parts.
2) The second part is associated with calculations of L_k and Γ_k^n in (32) and (33) respectively using $\sum_{j=0}^{n-1} \Gamma_k^j$.
The results of both parts are merged in (49) to be included in the Richardson iteration (48). The matrix-by-vector product $A\theta_{k-1}$ in (48) can be easily calculated in parallel via methods described for example by Saad (2003) .

7. COMPARISONS & QUANTIFICATION OF THE PERFORMANCE

Numerical calculation of the parameter vector θ_* for the system (1) where the ill-conditioned SPD information matrix A associated with the system with harmonic regressor with three frequencies is chosen for comparisons. The performance evaluation is presented in the following two parts.

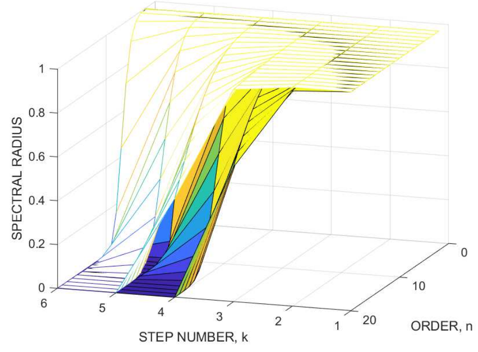

Figure 1. The Figure shows the spectral radius $\rho^{(k\ n^{k+1}\ +\ n^k)}$ for the matrix inversion algorithm defined in (36) (described in Section 5.1), plotted as colored surface. The spectral radius for Newton-Schulz iteration with improved convergence rate described by Stotsky (2019) $\rho^{(k\ +\ 1)\ n^k}$ is plotted as white surface. Both surfaces are plotted for the spectral radius of ill-conditioned case as functions of the order n and step number k.

1) The convergence rate of new matrix inversion algorithm (31) - (36) is compared to the convergence rate of recent algorithm with improved convergence rate described by Stotsky (2019) in Figure 1. The Figure shows that convergence rate improvements are more pronounced for higher orders and larger step numbers.
2) Comparison of the convergence rate of the parameter estimation algorithm (41) and the Richardson iteration with improved convergence rate described by Stotsky (2019) is presented in Figure 2. The algorithm (41) with high order convergence accelerator improves essentially the convergence rate compared to existing algorithms even for lower orders and small step numbers. Indeed, comparison of the Figure 1 and Figure 2 shows that new algorithms are the most beneficial in the Richardson framework.

8. CONCLUSION

This paper shows that the most general and well-known Newton-Schulz iteration is fast power series expansion and presents unified framework and tool-kit for power series factorization and reduction of the computational complexity. The framework allows reduction of complexity of many algorithms and factorization of the algorithm of the order 45 that requires 10 mmm only is presented as example.
Main result of the paper is new composite power series expansion for Newton-Schulz iteration with high degree of parallelism for the convergence rate improvement and computational efficiency. Comparative analysis of the convergence rates of new algorithms and exiting ones is performed via explicit transient models. New algorithms have faster convergence than known Newton-Schulz iterations. Moreover, new expansion resulted in significant improvement of the convergence rate of Richardson iteration for which recursive and computationally efficient version was developed. The results were also confirmed by simulations.

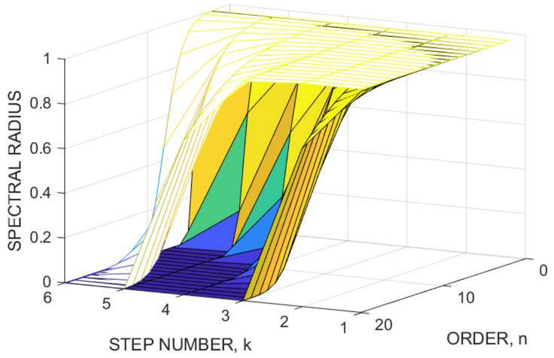

Figure 2. The Figure shows the spectral radius $\rho \quad h\, n^2\, \frac{(k\, n^{k+2} - (k-1)\, n^{k+1} - 2\, n^k - n + 2)}{(n-1)^2}$ for Richardson iteration (for parameter estimation) defined in (46) (plotted as colored surface). The spectral radius for Richardson iteration with improved convergence rate described by Stotsky (2019) $\rho \quad 2h\{\frac{n^{k+3} - n^4}{(n-1)^3} - (k-1)\{\frac{n^3}{(n-1)^2} + \frac{k}{2(n-1)}\} + k(n+2)\}$ is plotted as white surface. Both surfaces are plotted for the spectral radius of ill-conditioned case as functions of the order n and step number k for $h = 1$.

The paper opens new opportunities for convergence rate improvement of Newton-Schulz and Richardson iterations via computationally efficient composite expansions to be implemented on parallel machines with different computational performance.

REFERENCES

[1] Bayard, D. (2000). A General Theory of Linear Time-Invariant Adaptive Feedforward Systems with Harmonic Regressors, IEEE Trans. Autom. Control, vol. 45, N 11, pp. 1983-1996.

[2] Björck Å. (1996). Numerical Methods for Least Squares Problems, SIAM, First edition, April 1.

[3] Buranay S. and Iyikal O. (2019). A Predictor-Corrector Iterative Method for Solving Linear Least Squares Problems and Perturbation Error Analysis, Journal of Inequalities and Applications, vol. 203, pp.1-14.

[4] Chen K. (2005). Matrix Preconditioning Techniques and Applications, Cambridge University Press, Cambridge, UK.

[5] Demidovich B., Maron I. (1963). Basics of Numerical Mathematics, Moscow, Fizmatgiz, 660 pages (in Russian).

[6] Esmaeilic H., Erfanifar R. and Rashidi M. (2017). A Fourth-Order Iterative Method for Computing the Moore-Penrose Inverse, Journal of Hyperstructures vol. 6, N1, pp. 52-67.

[7] Fomin V., Fradkov A. and Yakubovich V.(1981). Adaptive Control of Dynamic Objects, Nauka, Moscow, (in Russian).

[8] Hackbusch W.(1994). Iterative Solution of Large Sparse Systems of Equations, Springer, New York.

[9] Horn R. and Johnson C. (1985). Matrix Analysis, Cambridge University Press.

[10] Isaacson E. and Keller H. (1966). Analysis of Numerical Methods, John Wiley & Sons, New York.

[11] Janiszowski K. (2003). Inversion of Square Matrices in Processors with Limited Calculation Abillities, International Journal of Applied Mathematics and Computer Science, AMSC, vol. 13, N 2, pp. 199-204.

[12] Ljung L. (1999). System Identification: Theory for the User, Prentice-Hall, Upper Saddle River, NJ.

[13] Pan V., Soleymani F. and Zhao L. (2016). Highly Efficient Computation of Generalized Inverse of a Matrix, arXiv:1604.07893v1 [math.RA].

[14] Peng R. and Spielman D. (2013). An Efficient Parallel Solver for SDD Linear Systems, arXiv:1311.3286v1 [cs.NA], 13 Nov.

[15] Petryshyn W. (1967). On Generalized Inverses and on the Uniform Convergence of $(I-\beta K)^n$ with Application to Iterative Methods, J. Math. Anal. Appl., vol. 18, pp. 417-439.

[16] Saad Y. (2003). Iterative Methods for Sparse Linear Systems, 2-nd edition, SIAM, Philadelpha, PA.

[17] Schulz G. (1933). Iterative Berechnung Der Reziproken Matrix, Zeitschrift für Angewandte Mathematik und Mechanik, vol. 13, pp. 57-59.

[18] Soleymani, F., Stanimirovic, P., Haghani, F. (2015). On Hyperpower Family of Iterations for Computing Outer Inverses Possessing High Efficiencies, Linear Algebra Appl., vol. 484,pp. 477-495.

[19] Soleymani, F. (2015). An Efficient and Stable Newton-type Iterative Method for Computing Generalized Inverse, $A^{(2)}_{T,S}$, Numer. Algorithms vol. 69, N3, pp. 569-578.

[20] Srivastava S. and Gupta D. (2014). A Higher Order Iterative Method for $A^{(2)}_{T,S}$, Journal of Applied Mathematics and Computing, vol.46, N 1/2, pp. 147 - 168.

[21] Stickel E. (1987). On a Class of High Order Methods for Inverting Matrices, ZAMM Z. Angew. Math. Mech. 67, pp. 331-386.

[22] Stanimirovic P., Kumar A. and Katsikis V. (2019) Further Efficient Hyperpower Iterative methods for the Computation of Generalized Inverses $A^2_{T,S}$, RACSAM, vol.113, pp. 3323-3339.

[23] Stotsky A. (2010). Recursive Trigonometric Interpolation Algorithms, Journal of Systems and Control Engineering, vol. 224, N 1, pp. 65-77.

[24] Stotsky A. (2015). Accuracy Improvement in Least-Squares Estimation with Harmonic Regressor: New Preconditioning and Correction Methods, 54-th CDC, Dec. 15-18, Osaka, Japan, pp. 4035-4040.

[25] Stotsky A. (2017). Grid Frequency Estimation Using Multiple Model with Harmonic Regressor: Robustness Enhancement with Stepwise Splitting Method, IFAC PapersOnLine 50-1, pp. 12817 - 12822.

[26] Stotsky A. (2019). Unified Frameworks for High Order Newton-Schulz and Richardson Iterations: A Computationally Efficient Toolkit for Convergence Rate Improvement, Journal of Applied Mathematics and Computing, vol. 60, N 1 - 2, pp. 605-623.

[27] Söderström T. and Stewart G. (1974). On the Numerical Properties of an Iterative Method for Computing the Moore–Penrose Generalized Inverse, SIAM J. Numer. Anal. vol. 11, pp. 61-74.

[28] Traub J. (1964). Iterative Methods for Solution of Equations, Englewood Cliffs, NJ: Prentice-Hall.